SOLVE	RESOLVE	License for SOLVE and RESOLVE	Download SOLVE and RESOLVE	Version: 2.10

SOLVE / RESOLVE

SOLVE:Automated crystallographic structure solution for MIR, SAD, and MAD
RESOLVE: Statistical density modification, local pattern matching, automated model-building, automated ligand-fitting, and prime-and-switch minimum bias phasing

Tom Terwilliger, Los Alamos National Laboratory
Last updated 8-July-2005 (version 2.10)

Subscribe to the SOLVE / RESOLVE mailing list!

Download SOLVE/RESOLVE manual (without sample scripts) as PDF (2 Mb)

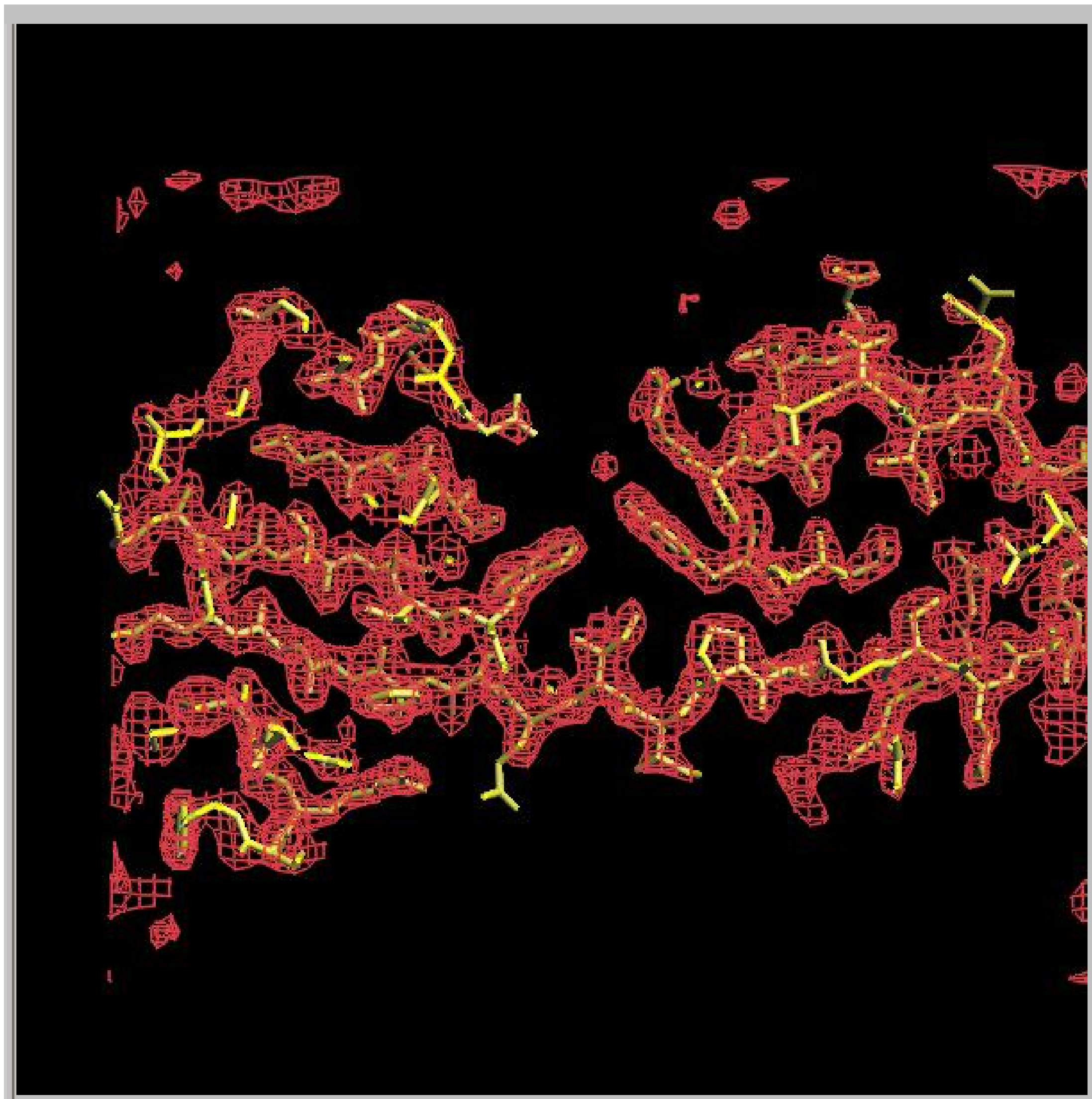
References for SOLVE and RESOLVE

SOLVE: Terwilliger, T.C. and J. Berendzen. (1999) "Automated MAD and MIR structure solution". Acta Crystallographica D55, 849-861.
RESOLVE: Terwilliger, T. C. (2000) "Maximum likelihood density modification," Acta Cryst. D56, 965-972.
RESOLVE model-building: Terwilliger, T.C. (2002) "Automated main-chain model-building by template-matching and iterative fragment extension." Acta Cryst . D59, 34-44.

(for full list see SOLVE references and RESOLVE references)

Features of version 2.10 of SOLVE / RESOLVE

- ***Automated loop fitting***
  RESOLVE now has a very powerful loop-fitting algorithm. You provide the residues on the ends of the loop and an mtz file with information for calculating a map, tell it how many residues are in the loop including the



SOLVE/RESOLVE electron-density map and auto-built atomic model for initiation factor 5a at a resolution of 2.1A. Multiwavelength diffraction data courtesy of Tom Peat; refinement with Garib Murshudov's refmac5; graphics drawn with Alwyn Jones' O

More features of version 2.10 of SOLVE / RESOLVE...

- ***PHENIX Alpha release NOW AVAILABLE:*** The PHENIX project (www.phenix-online.org) is a collaboration aimed at developing a comprehensive and integrated platform for determing macromolecular structures. The PHENIX software has graphical tools to let you choose your structure determination pathway from pre-packaged modules and to combine them in many ways, and it has Wizards that lead you through structure determination. You can run the PHENIX Wizards from a GUI or with scripts. PHENIX has full refinement (phenix.refine), dual-space heavy-atom search (HYSS), automated model-building (RESOLVE,TEXTAL), molecular replacement and maximum-likelihood phasing (PHASER), and many other powerful

ends and how hard to try, and resolve will fit the loop. It takes just a few seconds to run in most cases. If you want to do this in a fully automated way, try using the ResolveBuild and IterativeBuild wizards in PHENIX!

- ***Automated fitting of flexible ligands to electron density maps with the resolve_ligand_fit script:*** RESOLVE now is capable of fitting ligands with many rotatable bonds to maps. RESOLVE starts by finding the location and orientation of the largest fixed part of the ligand, then builds all the other parts sequentially to this core. You can even give resolve a list of ligands and the resolve_ligand_id script will fit each one to the map and score them all, identifying which may be correct.
- ***Merging of NCS copies*** is now automatically carried out during iterative model-building and refinement with the RESOLVE_BUILD script.
- ***New standard procedures:*** The best way to use SOLVE/RESOLVE now on SAD/MAD/MIR data is: (1) edit and run one of the standard SOLVE scripts,

features. Just about anything that SOLVE/RESOLVE scripts can do, PHENIX can do (better)! Give it a try, and be sure to update as new versions become available because it is changing rapidly!

- ***Fragment identification:*** RESOLVE can identify the presence of fragments of structure (helices, strands) in your map and use them to improve your phases. Like local pattern matching, this can make a big difference if your map is of moderate quality. This is also automatically carried out using the new RESOLVE_BUILD script.
- ***Superquick model-building:*** RESOLVE now can build your model at a rate of up to 1 residue every 2-3 seconds if you have a good map ("superquick_build"). Even the more thorough standard model-building in RESOLVE is now 3 times faster than earlier versions.
- ***Swap-space needs for SOLVE/RESOLVE:*** For the standard versions, 1 GB or more of swap space is recommended (700 MB minimum). On linux machines you can now run resolve_huge, and even go as high as "isizeit = 36" if you have 4 GB of swap space. RESOLVE runs best on linux machines if they have 1 GB or more of memory
- You can give SOLVE and RESOLVE all your MAD/SAD data and it will decide at what resolution the signal-to-noise is high enough to use for phasing.
- SOLVE carries out all the steps of macromolecular structure determination from scaling data to calculation of an electron density map, automatically.
- RESOLVE uses statistical density modification (previously called maximum-likelihood density modification) to improve electron density maps
- Prime-and-switch phasing in RESOLVE removes model bias from model-phased maps. See some amazing prime-and-switch examples!
- RESOLVE automatically identifies NCS in heavy-atom sites and applies it for you.
- Version 2.10 contains all of earlier SOLVE and RESOLVE versions. Now you download SOLVE and RESOLVE both at once.
- Your version 2 license is good for all versions 2.xx of SOLVE/RESOLVE. No need for new access codes.

(2) edit and run the RESOLVE_BUILD script, which does pattern-matching, the new fragment identification procedure, density modification, and iterative autobuilding and is improved from the original script in version 2.06. With fast model-building, the whole process takes only a few hours for a small protein and overnight to a few days for a moderate-sized one.

- ***Evaluate your final model automatically:*** The RESOLVE_BUILD script will automatically calculate a prime-and-switch composite omit map at the very end of model-building and evaluate the model and give you a report on how well your model fits the map. You can also use this script to evaluate a model you built yourself.
- ***Local pattern matching:*** RESOLVE_PATTERN can identify local patterns in your map and use them to improve your phases. Local pattern matching can make a big difference if your map is of moderate quality. This is automatically carried out using the new RESOLVE_BUILD script.
- ***CCP4 version 5 libraries:*** CCP4 version

	5.0 libraries are used in versions of SOLVE/ RESOLVE starting with version 2.08. Version 2.10 will work with output from previous versions, except that ha. pdb files from previous versions need to have the CELL, SCALE and CRYST headers added (these can be copied from your resolve.pdb or other model PDB files).

How do I get SOLVE and RESOLVE?	***Do you have a SOLVE license?***	***You can download SOLVE and RESOLVE upgrades at any time, once you have your version 2 license, there are no more forms to fill out. (Version 1 users need a new license.)***
	Do you need a SOLVE license?	***You need to fill out the license form; you can get a free 45-day license to try it out.***

Disclaimer/Privacy

SOLVE / RESOLVE home	SOLVE on-line manual	SOLVE Examples

SOLVE

Automated crystallographic structure solution for MIR, SAD, and MAD

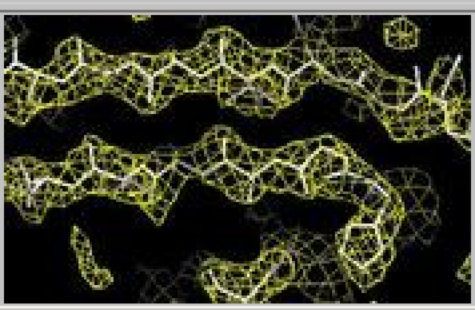
Tom Terwilliger, Los Alamos National Laboratory

Last updated 3-June-2005 (version 2.10)

What is SOLVE?	SOLVE is a program that can carry out all the steps of macromolecular structure determination from scaling data to calculation of an electron density map, automatically.
What can SOLVE do?	SOLVE does everything crystallographers do to solve an MIR or MAD structure, but automatically. It scales data, solves Patterson functions, calculates difference Fouriers, looks at a native Fourier to see if there are distinct solvent and protein regions, and can score partial MAD and MIR solutions to build up a complete solution. SOLVE has solved MIR and MAD structures with up to 66 heavy-atom sites. Check out some examples and look at the on-line manual for more info!

Many new features for versions 1.18/1.19/2.01/2.02/2.03/2.04/2.05/2.06/2.07/2.08/2.09/2.10 of SOLVE:

- You can tell SOLVE to ignore big peaks in the Patterson with the flag uvw_remove.
- SOLVE now will analyze your MAD/SAD data to identify to what resolution there is significant signal in your data; then it will carry out heavy-atom searches and phasing to this resolution. It will write out all the data for RESOLVE to read in and use in phase extension. You can set the working resolution with



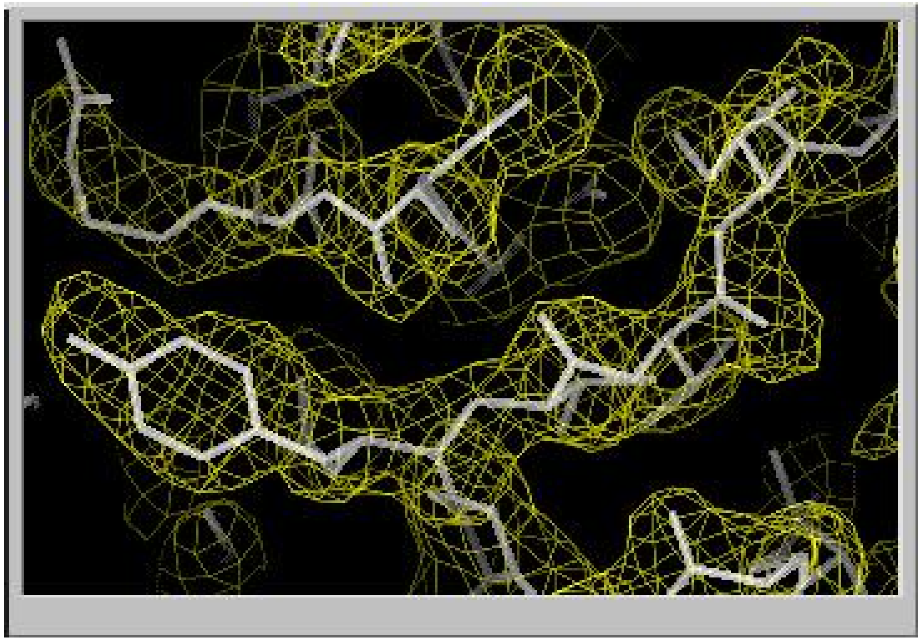
SOLVE / RESOLVE electron density map calculated after automatic solution at 2.4 A of a MAD structure with 56 selenium atoms followed by automatic maximum-likelihood density modification. Data and model courtesy of Ward Smith and Cheryl Janson

"res_phase" too.

- SOLVE now has easy scripts for SAD phasing.
- SOLVE now writes out heavy atom sites in fractional coordinates in solve.xyz and also the inverse in solve_inverse.xyz. You can copy these into your script and use ANALYZE_SOLVE to easily re-run either a modification or the inverse.
- SOLVE has even better MAD phasing! SOLVE now re-refines scattering factors using the final heavy atom parameters, improving the final phases. In combination with RESOLVE, the final maps are greatly improved over earlier versions.
- Must faster search for solutions: SOLVE now follows only the very best solution by default, greatly speeding up the search in most cases.
- You can start from MR or other input phases.
- You can read CCP4 unmerged intensities directly.
- You can specify a SOLVETMPDIR where SOLVE will write scratch files.
- You can tell SOLVE where some sites are and go on from there easily with ANALYZE_SOLVE and with ADDSOLVE
- There is now a SOLVE FAQS page.
- You can load the SOLVE manual with a web browser just by typing "solvehelp" on your terminal!
- Versions 1.19 and 2.01 correct bugs in SAD phasing and allows 6 derivatives again

Disclaimer

SOLVE/RESOLVE home	RESOLVE examples	RESOLVE on-line manual



RESOLVE

Statistical density modification

Prime-and-switch minimal-bias phasing

Local pattern matching

Automated model-building

Automated ligand-fitting

Tom Terwilliger, Los Alamos National Laboratory

What is resolve?

RESOLVE is a program that improves electron density maps.

RESOLVE uses a statistical approach to combine experimental X-ray diffraction information with knowledge about the expected characteristics of an electron density map of a macromolecule.

You can run RESOLVE to improve your maps right after using SOLVE or another program to solve your structure.

Version 2.02 and higher of RESOLVE will build a model of your structure automatically.

	Version 2.03 and higher can be used for iterative model-building and refinement.
How is RESOLVE different than other density-modification programs?	RESOLVE uses a new mathematical formulation to directly maximize the total probability of the phases. Most other approaches rely on phase recombination where the optimal statistical weighting of experimental and modified phases is not known.
What is prime-and-switch phasing and how does it minimize model bias?	Prime-and-switch phasing primes density modification with biased model phases, then switches entirely to an independent source of phase information (the probability of the map) to remove the model bias. Prime-and-switch phasing can give excellent unbiased maps even for crystals with very low solvent content (provided that the biased model did have substantial correct phase information!)
What are other new features of version 2 of resolve?	In addition to prime-and-switch phasing, version 2.0 of RESOLVE can find non-crystallographic symmetry in your heavy-atom sites and apply it automatically. Version 2.02 and higher can build a model for you as well. Versions 2.05 and higher can identify local patterns and use them to improve your phases.Versions 2.06 and higher can carry out iterative pattern id, fragment id, and model-building. Resolve versions 2.08 and higher can carry out automated ligand fitting as well.
How can I read more about resolve?	The mathematics behind RESOLVE and examples of its use are described in the article "Maximum likelihood density modification," appearing in Acta Cryst. D56, 965---972 (2000).

Last updated: 3-June-2005 (version 2.10)

SOLVE / RESOLVE Home Page

SOLVE/RESOLVE licensing information

U.S. Universities, non-profits, other non-governmental institutions A one-time $500 license fee is good for all versions from version 2.00 to version 2.99 for all machines at your institution. You can get a free 45-day trial license too.
U.S. Government institutions For U.S. government institutions, the SOLVE/RESOLVE license is free of charge. One license is good for all machines at your institution.
U.S. Commercial institutions Contact the Los Alamos Technology Transfer Division for commercial licenses. You can get a free 45-day trial license if you wish.

Non-U.S. Non-commercial institutions A one-time $600 license and export fee is good for all versions from version 2.00 to version 2.99 for all machines at your institution.
Non-U.S. Commercial institutions Contact the Los Alamos Technology Transfer Division for non-U.S. commercial licenses.

< br > < br >

SOLVE/RESOLVE home page

Downloading and upgrading SOLVE/RESOLVE

The current version of the SOLVE/RESOLVE package is version 2.10.

Please note: One license is good for all machines at your institution for all versions from 2.00 to 2.99 (it goes in "solve2.access" and the upgrade is free for academic/non-profits)

There are three easy parts to installing SOLVE/RESOLVE on your system

- you need to figure out which version of SOLVE/RESOLVE matches your system
- you need to ftp a compressed file to your computer, uncompress it, and run install.csh to put the files in the right places on your system and set up solvehelp (a link to the local version of the manual).
- you need to put the "solve2.access" file you get from us by email in the right place on your system

Once you have SOLVE/RESOLVE installed you can view the manuals just by typing "solvehelp".

1. First you need to choose the file that matches your computer. If none does, then you can get a version compiled on your computer directly by emailing me at "terwilliger@lanl.gov". Each download file contains "solve" and "solve_giant" and "solve_huge". Same for resolve. The giant-size and even bigger huge versions are for huge unit cells and are not usually necessary. Also if you have a really enormous unit cell you may need an even bigger version which you can receive by emailing me directly.

If your computer is a...	*Then ftp this file...*
Alpha running Digital (Tru-64) Unix	solve-2.10-alpha.tar.gz
SGI (R5000 or higher, most Octanes)	solve-2.10-sgi.tar.gz
linux (Red Hat 7.3 or Fedora 3 or higher on Pentium III or higher)	solve-2.10-linux.tar.gz

All other machines (and some SGI Octanes which won't run SGI version of SOLVE)	email terwilliger@lanl.gov for source to compile on your machine. You will need to have the CCP4 suite installed on your machine

2. Here is how to get the file containing the SOLVE and RESOLVE programs by ftp, uncompress it, and put the files in the right places on your system. (Note that you can substitute any directory you want for "/usr/local/lib/solve" in the install.csh script and then set the environmental variable SOLVEDIR to point to that directory in everyone's .login or .login_custom file.)

- Make the directory /usr/local/lib/solve on your computer and get into it as "root"
- Click on the ftp links above or this one to download the file you need:

ftp solve.lanl.gov/pub/solve/2.10/

Please note: If this download does not work please try directly running ftp from a terminal with:

```
ftp solve.lanl.gov
anonymous
your-email-address
cd pub/solve
ls
binary
get name-of-the-download-file-you-want
quit
```

- uncompress and extract the files (Note: on OSX use bzip2 to unzip the file):

```
gunzip solve*.gz
tar xof solve*.tar
```

- install solve by running install.csh (after editing it if you wish):

```
cd solve-2.10/
./install.csh
```

- Please note: on some systems the binaries supplied use shared libraries. If your system does not have the shared libraries you will get an error message when you run the programs. The solution is to compile the program on your own machine. Just email terwilliger@lanl.gov and I'll send you instructions.

- You are just about ready to go. Now you can run SOLVE or RESOLVE (usually in /usr/local/bin/solve and /usr/local/bin/resolve) for regular-size runs, and solve_giant and resolve_giant or the bigger solve_huge and resolve_huge for huge unit cells.

- Now this directory also has in it all the symmetry files
- This directory also has the local on-line manual. You can access the on-line manual just by typing "solvehelp" now. (It may only take effect after you log in again).

- Change for versions 2.08 and higher: RESOLVE uses the CCP4 version 5 libraries. In your scripts you will now want to specify both SYMOP (as in earlier versions) and SYMINFO. These are located in the same place. If you use the ones supplied with SOLVE/RESOLVE, they are located in:

```
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
```

- Finally, you need to set the CCP4_OPEN environmental variable. Put in your .login_custom or else at the beginning of all files to run solve the following command. (It allows ccp4 routines to overwrite existing files. If you don't do this solve will stop the second time you run it when it tries to open solve.mtz)
  - setenv CCP4_OPEN UNKNOWN # for sh or csh shells
  - export CCP4_OPEN=UNKNOWN # for bash or ksh shells
- You may also wish to set the environmental variable SOLVEDIR which tells SOLVE and RESOLVE where to look for solve2.access and symmetry files. (Change the directory if you don't use the default one.)
  - setenv SOLVEDIR /usr/local/lib/solve/ # for sh or csh shell
  - export SOLVEDIR=/usr/local/lib/solve/ # for bash or ksh shells
- You are now ready to go as soon as your solve2.access file is ready

3. Here is how to set up your solve2.access file. After your license is completed you should receive an email from us with the two lines of information for the solve2.access file. (Note that the old solve.access file for version 1 won't work any more.) They will look like:

```
License for SOLVE/RESOLVE version 2 expiring 01-jan-03
X89A943951
```

- There should be exactly 10 characters left justified on the second line. This is the access code.
- Please note: There must be a carriage return at the end of the second line of solve2.access for RESOLVE to read it correctly.
- Put the 2 lines into a file called "solve2.access" and put this file in the directory /usr/local/lib/

solve or else in the directory named by the environmental variable SOLVEDIR. (See "Intro/ Getting Started/The SOLVEDIR environmental variable" for more information on setting that variable.)

- You are ready to go. You should be able to run solve by typing "solve" from any directory if your path is set up to look in /usr/local/bin for programs. You can always type "/usr/local/bin/solve" to run solve if your path isn't set.

If you have problems getting SOLVE/RESOLVE going then:

- try again a couple times following the instructions as closely as possible
- check that the machine you are working on matches the file you have ftp'd
- email me for help at terwilliger@lanl.gov

Mailing list for SOLVE and RESOLVE

Now you can correspond with other SOLVE/RESOLVE users with this email newsgroup.

To subscribe to or unsubscribe from the mailing list, send an email message containing one of the following:

- subscribe solve
- unsubscribe solve

to:

- listmanager@listserv.lanl.gov

To send a message to everyone on the list, just send an email to

- solve@lanl.gov

SOLVE Table of Contents	Alphabetical Index

References for SOLVE

If you use SOLVE for structure determination, please cite the appropriate papers from this list and also please cite the web site "www.solve.lanl.gov". The overall paper describing SOLVE is:

Terwilliger, T.C. and J. Berendzen. (1999) "Automated MAD and MIR structure solution". Acta Crystallographica D55, 849-861.

For automated MAD structure determinations, references 1-3, 5 , 6, and 11 are appropriate; for automated MIR structure determinations, references 1-4 are appropriate.

1. Terwilliger, T. C., Kim, S.-H., and D. Eisenberg. (1987). Generalized method of determining heavy-atom positions using the difference Patterson function. Acta Cryst. A43, 1-5.

2. Terwilliger, T. C. and D. Eisenberg. (1983). Unbiased three-dimensional refinement of heavy-atom parameters by correlation of origin-removed Patterson functions. Acta Cryst. A39, 813-817.

3. Terwilliger, T. C. and D. Eisenberg. (1987). Isomorphous replacement: effects of errors on the phase probability distribution. Acta Cryst. A43, 6-13.

4. Terwilliger, T. C. and J. Berendzen (1996) Correlated phasing of multiple isomorphous replacement data. Acta Cryst. D52, 749-757.

5. Terwilliger, T. C. (1994). MAD phasing: treatment of dispersive differences as isomorphous replacement information. Acta Cryst. D50, 17-23.

6. Terwilliger, T. C. (1994) MAD phasing: Bayesian estimates of Fa. Acta Cryst. D50, 11-16.

7. Terwilliger, T. C. and J. Berendzen (1995). Difference refinement: a method for estimating differences between related structures. Acta Cryst. D51, 609-618.

8. Terwilliger, T. C. and Berendzen, J. (1996). Bayesian difference refinement. Acta Crystallographica section D52, 1004-1011.

9. Terwilliger, T. C. and Berendzen, J. (1996). Bayesian weighting for macromolecular crsytallographic refinement. Acta Cryst. D52, 743-748.

10. Terwilliger, T. C. and Berendzen, J. (1996). Correlated phasing in multiple isomorphous replacement. Acta Cryst D52, 749-757.

11. Terwilliger, T. C. and Berendzen, J. (1997). Bayesian MAD phasing. Acta Cryst. D53, 571-579.

Back to RESOLVE table of contents

RESOLVE references

1. T. C. Terwilliger (1999) "Reciprocal-space solvent flattening," *Acta Crystallographica* **D**55,1863-1871.
2. T. C. Terwilliger (2000) "Maximum likelihood density modification," *Acta Cryst.* D**56**, 965-972
3. T. C. Terwilliger (2001) "Maximum-likelihood density modification with pattern recognition of structural motifs" *Acta Cryst.*, **D**57, 1755-1762
4. T. C. Terwilliger (2001) "Map-likelihood phasing" *Acta Cryst.*, **D**57, 1763-1775
5. Terwilliger, T. C. (2004) Removing model bias in macromolecular crystallography. (in preparation)
6. Terwilliger, T. C. (2002). Statistical density modification with non-crystallographic symmetry. Acta Cryst. D58, 2082-2086.
7. Terwilliger, T. C. (2002). Rapid Automatic NCS identification Using Heavy-Atom Substructures. Acta Cryst. D58, 2213-2215.
8. Terwilliger, T. C. (2002). Automated main-chain model-building by template-matching and iterative fragment extension. Acta Cryst. D59, 34-44.
9. Terwilliger, T. C. (2002). Automated side-chain model-building and sequence assignment by template-matching. Acta Cryst. D59, 45-49.
10. Terwilliger, T. C. (2002). Improving atomic models at moderate resolution by automated iterative model-building and refinement. Acta Cryst. D59, 1174-1182.
11. Terwilliger, T. C. (2003) Statistical density modification using local pattern matching. Acta Cryst. D59, 1688-1701.

Back to RESOLVE table of contents

Sample scripts for resolve

The minimal script for running resolve: (Some sample data and input files are located the library directory where SOLVE and RESOLVE are installed, usually located at /usr/local/lib/solve/solve-2.03/lib/examples_resolve/)

```
#!/bin/csh
#
# Here is a minimal script to run resolve:
#
# Set CCP4 variables for symmetry information and
# for file handling:
#
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
setenv CCP4_OPEN UNKNOWN
#
# Now run resolve:
#
resolve<<EOD
solvent_content 0.4             ! your solvent content goes here. Next line is your
sequence file
seq_file protein.seq
EOD
```

- RESOLVE assumes "solve.mtz" as input with FP PHIB FOM HLA HLB HLC HLD
- Output is "resolve.mtz" with FP PHIM FOMM HLAM HLBM HLCM HLDM (M is for modified)
- You need to tell RESOLVE the solvent content (approximately)

- If you want to save time and do not want a model, specify the keyword: no_build

- The sequence file for your protein should look like:

```
MIVLTVHYSSEGILV   [put the sequence of chain type 1 here, up to 80 characters per line]
>>>      [this defines the end of chain 1]
MKLVERWISSTV      [put the sequence of chain type 2 here, up to 80 characters per line]
```

NOTE: just input 1 copy of each unique chain

Keywords for running RESOLVE with a different input file:

```
hklin other.mtz
LABIN FP=F PHIB=PHI FOM=W HLA=HLA HLB=HLB HLC=HLC HLD=HLD
hklout resolve.mtz
solvent_content 0.4             ! your solvent content goes here. Next line is your
sequence file
seq_file protein.seq
```

- The input file is named with HKLIN
- The column names can be specified with the LABIN keyword
- The output file can be specified with the HKLOUT keyword.

Running RESOLVE with a different number of cycles or changing resolution:

```
mask_cycles 4                   ! number of cycles of solvent mask generation
(default = 5)
minor_cycles 3                  ! cycles of iteration for each mask cycle (default =
10)
solvent_content 0.4             ! your solvent content goes here. Next line is your
sequence file
seq_file protein.seq
mask_cycles 4                   ! number of cycles of solvent mask generation
resolution 20 3.5               ! You can limit the resolution if you want
```

- You might want fewer mask cycles to save time if the phase change in the last cycles is very small.
- Alternatively you might want more minor cycles for each mask cycle if the phase change in the last minor cycles of each mask cycle is not very small.
- You can limit the resolution if you want. Currently, RESOLVE does not fill in reflections that are not in the input file. RESOLVE will fill in phase information for any reflections that it reads in that have a non-zero structure factor amplitude

Running RESOLVE with NCS

You can have RESOLVE identify NCS from atoms in a PDB file (default file is the "ha.pdb" file output by SOLVE). You can also generate an appropriate file for molecular replacement cases by selecting 3 equivalent atoms in each molecule and creating a PDB file with just these 3*N atoms.

Alternatively, you can enter NCS operators directly to resolve. In this case you also need to enter an estimate of the center-of-mass of molecule 1.

You need to input N sets of rotation matrices/translations/centers-of-mass, 1 for each copy in the asymmetric unit. Start with the identity for molecule 1. You can conveniently get all the numbers you need from the CCP4 program lsqkab if you have a model that you are starting with. The rotation/translation matrices and the center-of-mass are all input in a form appropriate for operating on orthogonal Angstrom coordinates (not fractional coordinates).

```
        rota_matrix  .92  .01  .33         !rotation matrix for molecule j ->molecule 1
        rota_matrix -.01 -.99  .03
        rota_matrix  .05  .10 -.95
        tran_orth    .50  .00 .00          ! translation for molecule j -> molecule 1
        center_orth   25.  39. 44.         !  center of mass, molecule j

        fraction_ncs 0.15                  ! fraction of the asymmetric unit in 1 copy of
NCS
                        ! (only needed if it is not equal to (fraction protein)/(number of
NCS copies)
```

- RESOLVE automatically identifies a mask for NCS, but requires an estimate of the center of molecule 1, in orthogonal A units ("center_orth")
- RESOLVE expects rotation matrices ("rota_matrix") and translation vectors ("trans_orth") in orthogonal Angstrom units that map each of the N molecules onto molecule 1 in the asymmetric unit. This is the form produced by the CCP4 program lsqkab with xyzin1 (reference)= molecule 1 and xyzin2 (working)= molecule j. If you have transformations that go from molecule 1 to molecule N, you can specify "invert" to invert the transformations.
- Input the matrices first (three lines of "rota_matrix" for each matrix); then the translation vector
- Input the identity as the first rotation matrix, with a zero translation vector

For at least the first molecule, you need to enter an estimate of the center of the molecule ("center_orth"). If you enter centers for others, resolve will verify that the centers match the positions expected from the symmetry you input

Prime-and-switch phasing starting from a SIGMAA-weighted map (see also script above starting right from an MR model)

To carry out prime-and-switch phasing starting with a model and SIGMAA phases: *(Sample data and input files are located the library directory where SOLVE and RESOLVE are installed, usually located at /usr/local/lib/solve/solve-2.06/lib/examples_resolve/)*

1. Run Randy Read's SIGMAA program to get as unbiased a starting map as possible into "sigmaa.mtz"
2. Go to the directory where you ran sigmaa and type (or put in a command file):

```
#!/bin/csh
# Here is a very minimal script to run prime-and-switch phasing
#
# Set CCP4 variables for symmetry information and
# for file handling:
#
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
setenv CCP4_OPEN UNKNOWN
#
# Now run prime-and-switch phasing:
#
resolve<<EOD
hklin sigmaa.mtz
labin FP=FP FC=FC PHIC=PHIC FOM=WCMB FWT=FWT
hklout ps.mtz
solvent_content 0.4             ! your solvent content goes here. Next line is your
sequence file
seq_file protein.seq
prime_and_switch
EOD
#
# Now "ps.mtz" has the output amplitudes, phases,
# figure of merit and HL-coeffs in columns labelled: FP PHIM FOMM HLAM HLBM HLCM HLDM
#
```

Note: The least biased map from SIGMAA is FWT exp(i PHIC), so RESOLVE works best starting with FWT from SIGMAA. However, you can leave out FWT if you wish. You need FP FC PHIC at a minimum.

- Output is "resolve.mtz" with FP PHIM FOMM HLAM HLBM HLCM HLDM (M is for modified)

Iterative model-building with RESOLVE and refmac5

You can carry out model-building and refinement iteratively with RESOLVE and refmac5 . A script that you can edit and use is resolve_build.csh. This is convenient for starting with solve.mtz (right from SOLVE)

You can also carry out iterative model-rebuilding (starting from an MR model, for example) with resolve_build.csh.

Adding HL coeffs

You can convert from PHI FOM to PHI FOM HL ABCD in an approximate way with this script:

```
hklin phases.mtz
labin FP=FP PHIB=PHIB FOM=FOM
scale_refl 0.  ! just use prior phase information
mask_cycles 1
minor_cycles 1
solvent_content 0.5  !any number, not used but needed
no_build
hklout phases_hl.mtz
```

Adjusting origin to match another phase set

You can apply an offset of the origin for a dataset to match the origin in another phase set. It uses an FFT convolution to find the best allowed offset between the maps calculated with the two phase sets. Unlike merging two phase sets (below) HL coefficients are optional. Note: the FP and SIGFP in the output file will come from the hklin file NOT the hklmerge file.

```
hklin random_1_hl.mtz   ! File with target map (fixed)
labin FP=FP PHIB=PHIM FOM=FOMM
hklmerge random_2_hl.mtz  ! File to be offset
labmerge FP=FP PHIB=PHIM FOM=FOMM HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM
hkl_offset_file resolve_off.mtz  ! The hklmerge file, offset all by itself
```

Merging phase sets with different origins

You can merge phase sets with different origins with this script. It uses an FFT convolution to find the best allowed offset between the maps calculated with the two phase sets. Merging the files REQUIRES HL coefficients (see above for generating them if you do not have them). Note: the FP and SIGFP in the output file will come from the hklin file NOT the hklmerge file:

```
hklin random_1_hl.mtz   ! File with target map (fixed)
labin FP=FP PHIB=PHIM FOM=FOMM HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM
hklmerge random_2_hl.mtz  ! File to be offset
labmerge FP=FP PHIB=PHIM FOM=FOMM HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM
hkl_offset_file resolve_off.mtz  ! The hklmerge file, offset all by itself
hkl_merge_file resolve_merge.mtz  ! The hklmerge file offset, then merged with hklin
```

Fitting a loop (with fixed ends)

You can now fit a loop if you know the ends. Put the two amino acids before the loop and the two after the loop in a pdb file (loop_ends. pdb) and put all the rest of your model in another PDB file (all_but_loops.pdb) that will be used as a mask to prevent building where your model exists. Then run the script below, specifying the number of residues in the loop, including the 2 on each end that you are supplying.

```
hklin resolve.mtz
labin FP=FP PHIB=PHIM FOM=FOMM
pdb_in loop_ends.pdb     ! Just 2 residues at beginning of loop, 2 at end.
seq_file seq.dat
extend_only
loop_only                ! just fit the loop
model all_but_loop.pdb
build_outside_model      ! don't build in region defined by model
no_sub_segments          ! fit the whole loop to sequence, not parts
n_random_loop 20
loop_length 8            ! exact length of loop, including ends in pdb_in
                         ! you can leave it out if unknown
rms_random_loop 0.3      ! how much to vary each try of building the loop
loop_cc_min   0.4        ! minimum correlation of density for the loop
EOD
```

Fitting flexible ligands to difference electron density maps

You can fit a flexible ligand into a difference electron density map.

You have two choices of input for the ligand. You can either input a single copy (preferred method) and the program will guess acceptable configurations, or you can make about 10 copies of the ligand in different allowed conformations and put them all into a one PDB file.

In either case, RESOLVE will identify the rigid parts of the molecule and all allowed relationships between them from this group of structures. It will then build the ligand into the electron density map. See the list of resolve keywords for additional options (i.e., delta_phi_ligand...)

```
hklin difference_density.mtz   ! File with map
labin FP=FP PHIB=PHIM FOM=FOMM
ligand_file ligand_in_1_or_10_conformations.pdb
```

That's it. RESOLVE will fit the ligand into the density and write it out to "ligand_fit.pdb" along with an analysis of the fit.

If you want to input an experimental map and a model for most of the structure you can add the keyword model:

```
model protein_model.pdb
```

and RESOLVE will subtract off the density from the model and just fit the remaining (ligand) density.

For more information on ligand fitting, see the introduction in ligand fitting.

A script you can edit is resolve_ligand_fit.com.

To score many ligands against your map, use resolve_ligand_id.com.

If you want to calculate a map with difference density for the ligand, then use resolve_difference script. This script starts with an mtz file containing native amplitudes (FP) and a PDB file containing a model for the structure without the ligand and calculates a simple difference map. You can also use any other suitable program to calculate this map. The output file resolve_diff.mtz (with FP PHIM FOMM) can then be used for ligand fitting with resolve_ligand_fit.com (for fitting one type of ligand), resolve_ligand_id.com (for testing a list of ligands), or a simple script like:

```
hklin resolve_map.mtz   ! File with map
labin FP=FP PHIB=PHIM FOM=FOMM
ligand_file ligand_in_1_or_10_conformations.pdb
model protein_model.pdb
```

If the ligand-fitting routine does not fit your ligand well, there are some options to get it to try harder:

- You can tell resolve to screen more potential locations of fragments in the FFT convolution search (default=300) with:

  ```
  n_ligand_pos 400
  ```

- You can tell resolve to refine more potential locations of fragments in the FFT convolution search (default=100) with:

  ```
  n_ligand_pos_ref 200
  ```

- You can tell resolve to search more finely for potential locations of fragments in the FFT convolution search (default=40 degrees) with:

```
 delta_phi_ligand 20
```

- You can tell resolve to keep track of more placements (default=100) of partially-built ligands with:

```
n_keep_plac 200
```

- Resolve will always take all the refined placements of fragments as a group and try to build from them, keeping the top n_keep_plac all the time. Resolve also tries one at a time if n_indiv_tries_min > 0. You can tell resolve to try to build starting with more individual placements (defaults min:0, max:5) of each sub-fragment with:

```
n_indiv_tries_min 20

n_indiv_tries_max 20
```

- You can tell resolve to try more sub-fragments as possible starting points for ligand building with:

```
n_group_search 10
```

- You can center the search at a desired position (A) with:

```
search_center  3 25 6
```

[Back to RESOLVE table of contents]

Introduction to RESOLVE

Why another density modification approach?

Although density modification (solvent flattening, non-crystallographic symmetry, phase extension, histogram matching, etc.) has been a very powerful tool, its potential is much greater than has been achieved so far. There are two reasons for this:

- The statistical basis of density modification has not been well developed
- The range of potential information included in density modification has not been fully utilized.

Problems with the phase recombination approach to density modification.

RESOLVE uses a statistical approach to density modification, while other methods use an approach in which a map is modified to meet expectations and the new phases are recombined with experimental phases. For the mathematical details, see the references for RESOLVE . You might also wish to see the discussion and extensions in Kevin Cowtan's article "Gaussian Likelihoods in real and reciprocal space" in the CCP4 newsletter.

Principal problems with the phase recombination method

What is the optimal relative weighting of modified and experimental phases?	Incorrect relative weighting means that the final results will not be optimal
	Incorrect weighting terms mean that the final figures of merit are almost always inflated
When do you stop iterating?	In some approaches the maps initially get better, then get worse unless you stop

A statistical approach to density modification

Density modification can be thought of as a way to adjust crystallographic phases (or amplitudes) to make them simultaneously consistent with the experimental data and with our expectations of what an electron density map should look like. The statistical approach is a mathematical way to formulate this statement. By using this formulation, the weighting factors and problems with convergence are taken care of automatically.

In RESOLVE, any set of structure factor amplitudes and phases has an associated probability composed of two simple parts:

Probability of a set of phases (and amplitudes)

The probability of the experimental phases	This is the probability that you would have observed your experimental data if this set of phases (and amplitudes) were correct
The lprobability of the map	This is the probability that the electron density map calculated from this set of phases is drawn from the set of plausible electron density maps for this structure

RESOLVE adjusts your crystallographic phases so as to maximize the total (posterier) probability of those phases. The mathematics is a little complicated but the idea is very simple. To see the mathematics in detail, have a look at T. C. Terwilliger (2000) "Maximum likelihood density modification," *Acta Cryst.* D**56**, 965-972.

Note on terminology: The approach used by resolve is now called "Statistical density modification," a name suggested by Kevin Cowtan. It used to be called "Maximum-likelihood density modification", using the term "likelihood" in a colloquial sense of probability. The old name (as pointed out by Gerard Bricogne and others) is confusing because the maximum-likelihood method is a specific technique that uses a specific definition of "likelihood" that is not used in this approach. Sorry to all for the confusion, and hoping that it will now be more clear. The mathematics remains exactly the same.

Using all the available information for density modification

Density modification is usually thought of as a process that is carried out on an experimental electron density map prior to model building, but iterative model-building methods such as ARP/wARP can also be thought of as density modification techniques. With the statistical approach, partial model information can be seamlessly incorporated into the total expression for the probability of the phases. This allows a hierachical approach to incorporating information about phase probability:

Types of information that can be used in statistical density modification

Experimental phases (if available)

Low-resolution structural information (solvent boundary)

Non-crystallographic symmetry

Partial model information (molecular replacement or model building)

Full atomic model information

The current version of RESOLVE can incorporate all of these types of information.

Carrying out density modification with RESOLVE

RESOLVE carries out density modification on several levels:

Each "mask cycle":

RESOLVE estimates the probability that each point in the map is within the protein or solvent region (a probabilistic "mask")
RESOLVE refines NCS symmetry operators, if present
RESOLVE then carries out one or more minor cycles:

Fitting of the histogram of density in the protein and solvent regions to model histograms (yielding beta = quality of this fit, and sigma= the overall error in the map)
Estimation of target density (a probility function)at each point based on these histograms for solvent and protein regions
Estimation of target density and uncertaingy at each point from NCS or a model map, if present
Calculations of derivatives of map probability with respect to phases
Estimation of phase probability from experimental phase probabilities and the map probability function

RESOLVE carries out mask cycles (up to 5) until no further changes occur in the phases.

If NCS is present, then RESOLVE carries out an initial mask cycle, not including any NCS, to estimate uncertainties in density estimated from NCS copies. Then RESOLVE carries out another initial mask cycle, using NCS but not solvent flattening, to estimate "sigma", the overall error in the map.

If "use_input_solv" is not set and "hklstart" is not specified, then RESOLVE uses the R factor to estimate the solvent content of the crystal. Solvent contents from 0.1 to 0.9 are tested, and the value leading to the minimum R is chosen. This optimal solvent content is written to the file "resolve.solvent." Note: if "use_input_solv" is specified, then RESOLVE assumes that the solvent content is already known and reads it from "solvent_content" if specified, or else from "resolve.solvent" if present, or else the default (0.40) is used.

RESOLVE also uses the R-factor to identify which histogram of solvent densities and protein densities to use in density modification. The file "rho.list" in $SOLVEDIR/segments/ contains several histogram profiles, all based on model electron density maps. These are at resolutions from 1.2 A to 4 A. RESOLVE carries out a test of each histogram initially and chooses the one leading to the lowest R factor. The histogram can be set using "database". The optimal database entry is written to "resolve. database".

Resolve estimates the optimal smoothing radius using a simple formula. For cycles where no density modification has occurred yet (first cycle normally, unless "phases_from_resolve" has been set), R is set with the equation: R=2.41 (dmin)**0.9 (fom)**-0.26.   For all other cycles (after density modification has begun), the smoothing radius is 4 A. These can also be set with "wang_radius", "wang_radius_cycle", "wang_radius_start", or "wang_radius_finish".

If "n_restore" is set by the user to be non-zero (default = 0), then after the phases have converged, the whole process is repeated again, starting with the original phases, but using the current probabilistic solvent mask. This allows an optimized mask to be used in the "first" cycle of density modification.

Removing model bias with prime-and-switch phasing

Electron density maps obtained using phases calculated from atomic models often show peaks at the coordinates of atoms in the models, even when those atoms are incorrectly placed. This effect can be reduced by careful weighting such as can be accomplished by Randy Read's SIGMAA approach, but it cannot be eliminated unless the phases are changed.

Prime-and-switch phasing is a way to remove model bias by using statistical density modification, ***but without including the phase information coming from the model*** once an initial map has been calculated.

The basic procedure is simple:

- Use the best existing amplitudes, phases, and weights to calculate a map
- Identify the probability that each point in the map is in solvent/macromolecule/NC-symmetry regions, etc
- Identify the expected distribution of electron density for points in each class (solvent/macromolecule etc)
- Calculate the log-probability of this map
- Identify how the log-probability of the map would change if the phases were changed
- Adjust the phases to maximize the log-probability of the map

The initial biased phase information from the model is required to get the procedure going. The final phases are essentially unbiased by the model because they are based on the features of the map, not on the prior phase probabilities.

The final phases are generally improved the most when:

- The starting phases are accurate (even if they are biased!)
- There is substantial solvent (25% is enough, the more the better)
- The data are accurate and high resolution (3 A is fine, the higher resolution the better provided there is accurate starting phase information at the high resolution limit)

There are some ways that prime-and-switch phasing can have residual bias:

- If there is low solvent content and not enough cycles are carried out (prime-and-switch phasing converges more slowly for low solvent content)
- If the starting model is highly refined and not enough cycles are carried out (if the model is refined, then the phases have already been adjusted to minimize the density in the solvent region, and prime-and-switch phasing converges more slowly)

There are some cases where prime-and-switch phasing does not yield a nice-looking map

- Usually these are cases where the model-based phases were very inaccurate (though they might have made a nice-looking biased map)
- The estimated corrected figure of merit output by RESOLVE will generally be very low in these cases, so you know that there just was not enough phase information

---

NCS averaging in RESOLVE

Non-crystallographic symmetry is an important source of information about the probabiltiy of an electron density map. RESOLVE can begin with transformation matrices and an estimate of the center-of-mass of molecule 1 that you input. RESOLVE can also figure out the transformations and center-of-mass automatically from the NCS in heavy-atom sites in a PDB file (if the default file "ha.pdb" exists and you don't specify NCS transformations, RESOLVE will try to find the NCS in those sites).

- RESOLVE uses NCS information in the following way (see Terwilliger, T. C. 2002 "Statistical density modification with non-crystallographic symmetry". Acta Cryst. D58, 2082-2086 and Terwilliger, T. C. (2002). "Rapid Automatic NCS identification Using Heavy-Atom Substructures" Acta Cryst. D58, 2213-2215.)

  - Use the best existing amplitudes, phases, and weights to calculate a map
  - Identify the region near the center-of-mass of molecule 1 for which the NCS transformations (on average) result in significant correlation
  - Define the asymmetric unit of NCS such that all points in it are close together and within the region of significant correlation, and such that no point in it is equivalent to any other point either by crystallographic or non-crystallographic symmetry.
  - Estimate the overall correlation among NCS-related molecules as a function of position in the NCS asymmetric unit (so that the edges might be allowed to vary more than the centers, for example)
  - Map all N copies of the NCS asymmetric unit on to identical grids
  - Generate target density for each molecule using all the other N-1 copies only
  - Map all the target densities back onto the original asymmetric unit of the crystal and use agreement between target density and the map as part of the probability of the map

Local pattern matching in RESOLVE

RESOLVE can use the local patterns of density in your electron density map in statistical density modification to improve crystallographic phases. The basic idea is that on a local level (within a sphere of radius 2 A) there are patterns of electron density that are associated with high density at the center of the pattern, and other patterns associated with low density at the center. RESOLVE goes through your electron density map, and at each point it compares the nearby density with a set of 20 templates (it does not use the density at the point of interest or right around it in this analysis). RESOLVE_PATTERN uses this analysis to come up with a new estimate of the density at each point in the map. This new estimate of density (the "image") has the remarkable property that errors in the image are almost uncorrelated with errors in the map used to create it. This means that phase information from the "image" can be combined with phase information from other sources in a simple way. You can see the details of all this in Terwilliger, T. C. (2003) Statistical density modification using local pattern matching. Acta Cryst. D59, 1688-1701.

The resolve_build script below uses image-based phasing. Image-based phasing is the use of an electron density map that typically comes from either an atomic model or from pattern-matching or from NCS, along with observed values of FP, to estimate phases. The process results in phases and figures of merit similar to those obtained with Randy Read's SIGMAA, but the values come directly from map-probability phasing. The electron density map provided is used as a target for statistical density modification: crystallographic phases are found that, when combined with observed amplitudes, give a map that is as close as possible to the target map. The figures of merit reflect how precisely each phase can be determined using this approach. The phases from image-based phasing are not the same as those from an FC calculation and they are not always unimodal like FC, SIGMAA or Sim-weighted phases.

Fragment identification in RESOLVE

RESOLVE can carry out an FFT-based search for fragments of structure (currently helices, strands), refine the locations of these fragments, and use them in density modification even if a complete model cannot be built. The approach to finding fragments ("Maximum-likelihood density modification with pattern recognition of structural motifs",Terwilliger, T. Acta Cryst D. 57, 1755-1762; 2001) is very similar to Kevin Cowtan's FFT-based search (Cowtan, K., Acta Cryst D54, 750-756, 1998). A template consisting of averaged helical density (or strand density) is rotated over a range of orientations designed to cover most possibilities within about 20 degrees and an FFT convolution is carried out for each orientation to find locations where the template and map match. The best matches are identified and the orientiations and positions are refined. Then a pseudo-map is constructed consisting of the original templates, oriented based on the refined positions found in the search, and weighted by the local correlation coefficient. This pseudo-map is used as a source of phase information through map-probability phasing (Map-likelihood phasing", Terwilliger, T., Acta Cryst., D57, 1763-1775). This approach is similar to the one described in the original publication ("Maximum-likelihood density modification with pattern recognition of structural motifs",Terwilliger, T. Acta Cryst D. 57, 1755-1762; 2001) but works much better than the original method.

Fragment identification is normally carried out right after model-building because the same FFT search can be used for both. The resolve build script includes it.

Automated model-building and iterative model-building in RESOLVE

After the completion of density modification, RESOLVE builds a model of your structure. For versions 2.02 and higher, the model needs sequence information from you. You specify a file with the keyword "seq_file" and RESOLVE expects a sequence of amino acids in 1-letter format. If there are more than one type of chain, RESOLVE expects them separated by a line containing ">>>". . Typically RESOLVE can build 70-90% of the residues for a good map at 2-3 A resolution. You can tell if the model is correct by noting how good the match is to the sequence and by noting the NCS correspondence among chains (if NCS exists). The PDB file that RESOLVE writes out will have the model and also as HETATM records at the end with the heavy atom sites from SOLVE output file ha.pdb.

You can read all the details about RESOLVE automated model-building in Terwilliger, T. C. (2002). Automated main-chain model-building by template-matching and iterative fragment extension. Acta Cryst. D59, 34-44 and Terwilliger, T. C. (2002). Automated side-chain model-building and sequence assignment by template-matching. Acta Cryst. D59, 45-49.

RESOLVE now has superquick model building! The standard RESOLVE model-building for version 2.05 and higher is about 3 times faster than earlier versions. This is made possible by a more selective choice of which fragments to consider extending (no need to work on a fragment that covers a region that is already built). Versions 2.05 and higher also have the option of "superquick_build" which is about 10 times faster than previous versions of RESOLVE model-building. For a very good map (one where RESOLVE can build >80% of the model) superquick_build typically gives almost the same model as the standard build. For a moderate-quality map, the standard build or even the "thorough_build" may give up to 10% more model built.

RESOLVE versions 2.05 and higher include cycles of model-building in which the thresholds for fit of the model to the map are sequentially lowered. This allows much more of the model to be built, while keeping the accuracy of most of the model high. You can use "aggressive_build" to try and build as much as possible, or "conservative_build" to build only the best parts.

RESOLVE versions 2.06 and higher include the capability of identifying fragments (helices; strands) in a map and including them in density modification

RESOLVE builds a model in the following way.

- Use the best existing amplitudes, phases, and weights to calculate a map
- Identify locations of helices and strands using an FFT-based correlation search with a standard set of helix/strand templates
- Using a library of actual helical/beta templates, find the best match to density near each helix/strand
- Trim the templates down to match the density
- Extend the templates using a template library of short fragments

- Assemble fragments into longer fragments
- Match side-chain density to library of side-chain densities, get probability of each possible sequence alignment, choose those with very high probability
- Map all the fragments to one asymmetric unit so that they are as close together as possible
- Write out PDB file with the fragments as a main-chain model.
- Note: RESOLVE writes out "resolve.mtz" before it starts building the model, so if you don't want to wait, your phases are already ready for you.

RESOLVE (versions 2.06 and higher) can carry out pattern identification, fragment identification, density modification, and iterative model-building and refinement in combination with refmac5 (versions 5.1.24 and higher only!)

- In the first cycle, resolve carries out density modification and builds a model as usual. The model is refined and extended as much as possible. Then the current electron density map is searched for local patterns and for fragments (helices/strands), and maps are created based just on these features.
- On the next cycles, resolve uses the pattern and fragment maps, along with a map created from all models built so far, to generate model density for the asymmetric unit. This density is used along with solvent flattening, NCS, and histogram matching in the next cycle of density modification.
- Additionally, a prime-and-switch composite omit map is created in which all the above information is used except that in each "omit" region of the map, the model-based information is left out; and the omit regions are then spliced together to form a composite omit map. This omit map is used in identifying the "patterns" for the next cycle so as to minimize bias in this step.
- RESOLVE then builds a new model based on the combined phase information. It also uses fragments from the last model as candidates for parts of the new model.
- The process is iterated as long as desired (typically 5-10 cycles are plenty). If the model is not as complete as desired after about 10 cycles (i.e., R-factor > .40) then the model is used in model rebuilding with phase information. This is just like model rebuilding (below) except that experimental phase information is included throughout.
- You can use a standard script " resolve_build.csh " to carry this out.

RESOLVE (versions 2.03 and higher) can also carry out iterative model-rebuilding. This is like model-building except that you start with just a model of some kind and measured amplitudes and resolve does everything from there. This works much more slowly than model-building with experimental phases.

- Rebuilding can be carried out either with or without composite omit maps. Omit maps are recommended for rebuilding of a model with possible model errors.
- Each cycle, RESOLVE starts with input model and create an electron density map (image). Then resolve uses this map as a target in statistical density modification along with the measured FP to estimate phases. Then these phases are used as the starting phases (but not probabilities) in a cycle of density modification including (1) any experimental phase probabilities, (2) solvent flattening, histogram matching, NCS, and (3) model density based on a composite of the last 20 models.
- RESOLVE then builds a model, the model is refined with refmac5, and the model is then extended and rebuilt. On each cycle that is not an omit cycle, RESOLVE uses fragments from the previous model along with fragments identified from the map itself as possibilities for constructing the new model.
- This process is repeated (typically 100 cycles)
- You can use a standard script " resolve_build.csh " to carry this out (it is the same script as for autobuilding). It can take a long time for rebuilding!

RESOLVE_BUILD (versions 2.06 and higher) can automatically evaluate a model, given a set of amplitudes FP (and phases PHIB and FOM if available). First RESOLVE will rebuild the model (to reduce any bias due to refinement). Then RESOLVE will calculate a prime-and-switch composite omit map (as used in rebuilding) based on the rebuilt model and any phase information you give it. Then RESOLVE will compare the original model to this map and summarize the fit for you.

RESOLVE (versions 2.08 and higher) can carry out fitting of FLEXIBLE LIGANDS to an electron density map. The only inputs needed are an electron density map (or difference map), and either just one (recommended) or else 5-10 copies (ok also) of the ligand in random but stereochemically ideal conformations in a PDB format file. The routine will figure out the allowed bond rotations from the copies of the ligand, and then will fit the ligand into the density starting with the biggest rigid part of the ligand. Parts of the ligand that do not fit are built as reasonably as possible, but may be built out of density or may be left off.

You can use the sample script resolve_ligand_fit.com script which allows you to find one or more than one copy of a ligand in a map.

You can even take a list of PDB files containing different ligands, fit each one to your map, and score them to identify which ligand may be bound, using the sample script resolve_ligand_id.com.

See the additional descriptions in resolve_sample_scripts too.

Also see the list of resolve keywords for additional options.

Thanks to Herb Klei for emphasizing the need for ligand fitting and for suggesting the idea of first finding the biggest fixed part of the ligand and then building the rest from this core!

RESOLVE (versions 2.08 and higher) will automatically merge NCS-related copies of your model during iterative model-building and refinement. The merging is done in the "extend_only" mode of model-building. An mtz file with FP PHIB FOM, a model (with >1 NCS copy) and a coordinate file with positions of atoms or pseudo-atoms (ha_file) used to deduce the NCS relationships are read in. The coordinates of each NCS-related copy are placed at all NCS-related positions, merged (if possible) and then are extended if possible into the density. If you do not want this to be done, use the flag no_merge_ncs_copies. You can merge models yourself with RESOLVE too: use the extend_only flag for model-building and include your model with: pdb_in your-current-model . Note that you need to supply an mtz file with a map to do this. You can specify the keywords trim or no_trim to tell RESOLVE to trim the resulting model back to the density or not.

```
#!/bin/csh
#
#
#    Resolve ligand-fitting script
#     T. Terwilliger 05-Jan-2005
#
#    This script requires resolve version 2.08 and ccp4 version 5.0.2
#
#    Before running this script, please get map coefficients with
#    resolve_completion.com script(output is resolve_map.mtz optimized for ligand )
#    or else input map coefficients of your own in hklin and labin
#
#
setenv SOLVEDIR /usr/local/lib/solve/
setenv SOLVETMPDIR .
setenv SYMOP $SOLVEDIR/symop.lib
setenv SYMINFO $SOLVEDIR/syminfo.lib
setenv CCP4_OPEN UNKNOWN
unlimit
#
echo "Resolve ligand-fitting script version 2.08.8 of 05-Jan-2005"
echo ""
#   get N ligands in N passes through map,
#    excluding previous as we go
#
set hklin = resolve_diff.mtz
set labin = "FP=FP PHIB=PHIM FOM=FOMM"
set number_of_ligands = 1    # number of copies of ligand to find
set ligand_template = ligand_template.pdb    # file with 1-10 copies of ligand
set model = PARTIAL_MODEL.pdb    #  model without ligand or ""
set resolve = /u1/terwill/resolve/work/resolve.linux  # where is resolve
set dmin = NONE   # NONE to use all data
set dmax = NONE   # NONE to use all data
#
#  some parameters you can set if you want to...
#
set n_indiv_tries_min = 10    # usually 0-10, but set up to 100 to try harder to find soln
set n_indiv_tries_max = 10    # usually 5-10, but set up to 100 to try harder to find soln
set n_group_search = 3    # usually 3, but set up to 10 to try harder to find soln
set search_dist = 10    # usually 10  A; always at least 5; smaller speeds up search
set no_local_search = ""    # usually "";  "no_local_search" to force complete search
set delta_phi_ligand = 40    # usually 40 degree increments; set lower to search more
```

```
#
if ( $dmin == NONE || $dmax == NONE ) then
 set resolution_line = ""
else
 set resolution_line = "resolution $dmin $dmax"
 echo "Resolution limits: $dmin $dmax A"
endif
#
echo "Input mtz file with map: $hklin"
if  (! -f $hklin ) then
  echo "Sorry, the file $hklin does not seem to exist?"
  exit
endif
echo "Labin line for $hklin is $labin"
echo "Number of copies of ligand to find: $number_of_ligands"
echo "Template PDB file containing 1 to 10 copies of ligand: $ligand_template"
if  ( ! -f $ligand_template) then
  echo "Sorry, the file $ligand_template does not seem to exist?"
  exit
endif
if ($model != "") then
 echo "The model $model will be used to mask out part of the map"
 if  ( ! -f $model) then
  echo "Sorry, the file $model does not seem to exist?"
  exit
 endif
set model_use = "model ALL.PDB"
else
set model_use = ""
endif
#
echo "Location of resolve: $resolve"
if  ( ! -f $resolve) then
  echo "Sorry, the program $resolve does not seem to exist?"
  exit
endif

echo "Number of groups (fragments) to search for with FFT: $n_group_search"
echo "FFT search will be in increments of $delta_phi_ligand degrees"
if ($no_local_search != "") then
 echo "Entire map will be searched"
endif
echo "Fitting will finish when number of tries is $n_indiv_tries_max or when"
echo "all atoms are found and the number of tries is at least $n_indiv_tries_min"
#
echo ""
#-----------------------------------------------------------------------------
#-------------------------------------------------------------
#  figure out if this machine uses grep -a or just grep for text files:
echo "A" > test_a.dat
 set test_grep = `grep -a "a" test_a.dat >& tmp.dat`
if ( $status ) then
#  there was an error...do not use grep -a
 set grep_type = "grep"
else
 set grep_type = "grep -a"
endif
set test_grep = `$grep_type "A" test_a.dat`
if ( $#test_grep != 1 ) then
 echo "Sorry, unable to set the grep command on this system...giving up"
 exit
endif
rm test_a.dat
#-------------------------------------------------------------

#-----------------------------------------------------------------------------
#
if ($model != "")then
cp $model ALL.PDB
cat ALL.PDB|$grep_type 'CRYST1' > resolve_ligand.pdb
cat ALL.PDB|$grep_type 'ORIGX[1-3]' >> resolve_ligand.pdb
cat ALL.PDB|$grep_type 'SCALE[1-3]' >> resolve_ligand.pdb
#  make sure the headers exist...
@ header_lines = `cat resolve_ligand.pdb |wc -l`
if ( $header_lines >= 4 ) goto ok1
 echo "The input PDB file $model needs to have at least 4 lines of headers with "
 echo "CRYST1 SCALE1 SCALE2 SCALE3"
 echo "Yours seems to have instead $header_lines of headers"
 cat resolve_ligand.pdb
 exit
ok1:
else
 if ( -f ALL.PDB ) rm -f ALL.PDB
 if ( -f resolve_ligand.pdb ) rm -f resolve_ligand.pdb
endif
#
```

```
set cycle = 1
while ($cycle <= $number_of_ligands)
#
echo "Searching for copy $cycle of ligand..."
$resolve<<EOD
$resolution_line
hklin $hklin
labin $labin
ha_file NONE
no_build
delta_phi_ligand $delta_phi_ligand
$no_local_search
search_dist  $search_dist
$model_use   ! exclude region defined by model
ligand_file $ligand_template
ligand_resno $cycle   # label ligand with $cycle
n_indiv_tries_min  $n_indiv_tries_min
n_indiv_tries_max  $n_indiv_tries_max
n_group_search $n_group_search  ! how many groups to search for
! some more parameters that can be set:
! search_center  3 25 6 ! center the search here
! group_search  3     ! use this group in FFT search
! fit_phi_range -180 180  ! range of torsion angles to check
! fit_phi_inc   20    ! increment for torsion angle check
EOD
#
if ($cycle == 1 ) then
  cp ligand_map.mtz ligand_map_all.mtz
  cp ligand_map.map ligand_map_all.map
endif
#
cp ligand_fit.pdb ligand_$cycle.pdb
#
#  add new ligand to ALL.PDB and to resolve_ligand.pdb
#
if ( -f ALL.PDB )then
 cat  ligand_fit.pdb |$grep_type ATOM >> ALL.PDB
else
 cp  ligand_fit.pdb ALL.PDB
endif
if ( -f resolve_ligand.pdb ) then
 cat  ligand_fit.pdb |$grep_type ATOM >> resolve_ligand.pdb
else
```

```
 cp ligand_fit.pdb resolve_ligand.pdb
endif
#
@ cycle ++
end
#
cp ligand_map_all.mtz ligand_map.mtz
cp ligand_map_all.map ligand_map.map
#
echo "Evaluating fit of ligand to map...."
$resolve<<EOD>TEMP.DAT
hklin $hklin
labin $labin
$resolution_line
model resolve_ligand.pdb
evaluate_model
EOD
$grep_type 'region of model' TEMP.DAT|$grep_type Map
rm -f TEMP.DAT
echo ""
#
echo "All done...your fitted $number_of_ligands copies of ligand are in resolve_ligand.pdb"
echo "and coefficients for a map showing just the ligand regions are in ligand_map.mtz"
```

```
#!/bin/csh
#
#
#    Resolve ligand identification script
#     T. Terwilliger 05-Jan-2005
#
#    This script requires resolve version 2.08 and ccp4 version 5.0.2
#
#    Before running this script, please get map coefficients with
#    resolve_completion.com script(output is resolve_map.mtz optimized for ligand )
#    or else input map coefficients of your own in hklin and labin
#
#
setenv SOLVEDIR /usr/local/lib/solve/
setenv SOLVETMPDIR .
setenv SYMOP $SOLVEDIR/symop.lib
setenv SYMINFO $SOLVEDIR/syminfo.lib
setenv CCP4_OPEN UNKNOWN
unlimit
#
echo "Resolve ligand identification script version 2.08.5 of 05-Jan-2005"
echo ""
#
#   Test many ligands for fit to map and identify the best
#
set hklin = resolve_diff.mtz
set labin = "FP=FP PHIB=PHIM FOM=FOMM"
set ligand_list_file = ../ligand_list_1.dat    # file containing list of ligand PDB files
set model = PARTIAL_MODEL.pdb
set resolve = /u1/terwill/resolve/work/resolve.linux  # where is resolve
set dmin = 2.6
set dmax = 200
#
#  some parameters you can set if you want to...
#
set n_indiv_tries_min = 10    # usually 0 to 10, but set up to 20 to try harder to find soln
set n_indiv_tries_max = 10    # usually 5-10, but set up to 20 to try harder to find soln
set n_group_search = 3    # usually 3, but set up to 10 to try harder to find soln
set search_dist = 10    # usually 10  A; always at least 5; smaller speeds up search
set no_local_search = ""    # usually "";  "no_local_search" to force complete search
set delta_phi_ligand = 40    # usually 40 degree increments; set lower to search more
#
```

```
echo "Resolution limits: $dmin $dmax A"
echo "Input mtz file with map: $hklin"
if  (! -f $hklin ) then
  echo "Sorry, the file $hklin does not seem to exist?"
  exit
endif
echo "Labin line for $hklin is $labin"
echo "File containing list of template PDB files is $ligand_list_file"
if  ( ! -f $ligand_list_file) then
  echo "Sorry, the file $ligand_list_file does not seem to exist?"
  exit
endif
if ($model != "") then
 echo "The model $model will be used to mask out part of the map"
 if  ( ! -f $model) then
  echo "Sorry, the file $model does not seem to exist?"
  exit
 endif
set model_use = "model $model"
else
set model_use = ""
endif
#
echo "Location of resolve: $resolve"
if  ( ! -f $resolve) then
  echo "Sorry, the program $resolve does not seem to exist?"
  exit
endif

echo "Number of groups (fragments) to search for with FFT: $n_group_search"
echo "FFT search will be in increments of $delta_phi_ligand degrees"
if ($no_local_search != "") then
 echo "Entire map will be searched"
endif
@ n1 = $n_indiv_tries_max + 1
@ nn = $n_indiv_tries_min + 1
echo "Fitting will finish when number of tries is $n1 or when"
echo "all atoms are found and the number of tries is at least $nn"
#
echo ""
#---------------------------------------------------------------------------
#-------------------------------------------------------------
#  figure out if this machine uses grep -a or just grep for text files:
```

```
echo "A" > test_a.dat
 set test_grep = `grep -a "a" test_a.dat >& tmp.dat`
if ( $status ) then
#  there was an error...do not use grep -a
 set grep_type = "grep"
else
 set grep_type = "grep -a"
endif
set test_grep = `$grep_type "A" test_a.dat`
if ( $#test_grep != 1 ) then
 echo "Sorry, unable to set the grep command on this system...giving up"
 exit
endif
rm test_a.dat
#------------------------------------------------------------

#----------------------------------------------------------------------------
#
if ($model != "")then
#  make sure the headers exist...
@ header_lines = `cat $model |wc -l`
if ( $header_lines >= 4 ) goto ok1
 echo "The input PDB file $model needs to have at least 4 lines of headers with "
 echo "CRYST1 SCALE1 SCALE2 SCALE3"
 echo "Yours seems to have instead $header_lines of headers"
 cat resolve_ligand.pdb
 exit
ok1:
endif
#
#  make sure all the ligand files in $ligand_list_file exist
set count = 0
set keep_going = 1
set lines = `cat $ligand_list_file |wc -l`
echo "Number of files in $ligand_list_file is $lines"
set count = $lines
while ($count > 0)
 set file = `tail -${count} $ligand_list_file |head -1 `
 #echo "File: $file"
 if ( ! -f $file ) then
  echo "Sorry the file $file from $ligand_list_file does not exist?"
  exit
 endif
```

```
@ count--
end
#
echo "Evaluating all ligands in $ligand_list_file now...and placing fitted"
echo "ligand xx in resolve_ligand_xx.pdb"
echo ""
echo " LIGAND    ATOMS PLACED      CC          SCORE        TEMPLATE FILE"
set count = $lines
set count_up = 0
while ($count > 0)
@ count_up++
 set file = `tail -${count} $ligand_list_file |head -1 `
 #echo "Evaluating ligand $file"
#
$resolve<<EOD   > resolve_fit_id_$count_up.log
resolution $dmax $dmin
hklin $hklin
labin $labin
ha_file NONE
no_build
delta_phi_ligand $delta_phi_ligand
$no_local_search
search_dist  $search_dist
$model_use   ! exclude region defined by model
ligand_file $file
ligand_resno 1 # label ligand with
n_indiv_tries_min  $n_indiv_tries_min
n_indiv_tries_max  $n_indiv_tries_max
n_group_search $n_group_search  ! how many groups to search for
! some more parameters that can be set:
! search_center  3 25 6 ! center the search here
! group_search  3     ! use this group in FFT search
! fit_phi_range -180 180  ! range of torsion angles to check
! fit_phi_inc   20    ! increment for torsion angle check
EOD
#
if ( -f ligand_fit.pdb )then
 if($count_up == "1" ) then
  cat ligand_fit.pdb  > all_ligands.pdb
 else
  cat ligand_fit.pdb|$grep_type ATOM  >> all_ligands.pdb
 endif
$resolve<<EOD>TEMP.DAT
```

```
hklin $hklin
labin $labin
resolution $dmin $dmax
model ligand_fit.pdb
evaluate_model
EOD
else
echo "" > TEMP.DAT
echo "" > resolve.ligand_scores
endif
#
set correl_line = `$grep_type 'region of model' TEMP.DAT|$grep_type Map`
#if (-f TEMP.DAT ) rm -f TEMP.DAT
if ( $#correl_line > 0 ) then
 set correl = $correl_line[$#correl_line-$#correl_line]
else
 set correl = 0
endif
#
set score_line = `cat resolve.ligand_scores |$grep_type MODIFIED`
set placed_line = `cat resolve.ligand_scores |$grep_type "ATOMS PLACED"`
set poss_line = `cat resolve.ligand_scores |$grep_type TOTAL`
if ( $#score_line > 0 ) then
 set score = $score_line[1-1]
else
 set score = 0
endif
if ( $#placed_line > 0 ) then
 set placed = $placed_line[1-1]:r
else
 set placed = 0
endif
if ( $#poss_line > 0 ) then
 set poss = $poss_line[1-1]:r
else
 set poss = 0
endif
cp ligand_fit.pdb resolve_ligand_${count_up}.pdb
echo "    $count_up        ${placed}/${poss}          $correl         $score     $file"
@ count --
end
#---------------------------------------------------------------
#---------------------------------------------------------------
```

```
echo ""
echo "Re-evaluating all ligands in $ligand_list_file now over common region"
echo ""
echo " LIGAND   CC     TEMPLATE FILE"
set count = $lines
set count_up = 0
head -10000 all_ligands.pdb > model_2.pdb   # limit to 10000 atoms
while ($count > 0)
@ count_up++
 set file = `tail -${count} $ligand_list_file |head -1 `
 #echo "Evaluating ligand $file"
#
$resolve<<EOD>TEMP.DAT
hklin $hklin
labin $labin
resolution $dmin $dmax
model resolve_ligand_${count_up}.pdb
model_2 model_2.pdb    ! evaluate in region of model_2
evaluate_model
EOD
#
set correl_line = `$grep_type 'region of model' TEMP.DAT|$grep_type Map`
#if (-f TEMP.DAT ) rm -f TEMP.DAT
if ( $#correl_line > 0 ) then
 set correl = $correl_line[$#correl_line-$#correl_line]
else
 set correl = 0
endif
#
echo "    $count_up      $correl  $file"
@ count --
end
#
#
#
```

```
#!/bin/csh
set main_version = 2
set minor_version = 09
set edit_version = 12
#
#                 NOTE:  This script requires resolve version 2.08 or higher
echo " "
echo "RESOLVE build script version $main_version.$minor_version.$edit_version "
echo " "
echo "Date: `date`"
echo "Working directory: `pwd`"
echo " "
#
#          RESOLVE iterative model-building script
#                      26-Jun-2005
#
#                     Tom Terwilliger
#             Los Alamos National Laboratory
#
#
#    Input:  FP and SIGFP, experimental phases.
#           Input model  (optional)
#
#    Autobuilding:  Iteratively improve phases using statistical density
#           modification including solvent flattening, histogram-matching,
#           NCS, local pattern recognition, local fragment identification,
#           and model-building and refinement. Optionally include cycles of
#           rebuilding (see below); accept if R decreases.
#
#    Rebuilding: Start with input model and calculate phases using
#           density modification including the model information. Optionally
#           calculate and use prime-and-switch composite omit map.
#           Build new model into the map...iterate
#
#    Model evaluation: as rebuilding, but just evaluate the model using the
#           prime-and-switch composite omit map
#
#    Output: Optimized phases in resolve_best.mtz
#           Partially-refined model in resolve_best.pdb
#           Log file for best cycle in resolve_best.log
#           CCP4-style map file in resolve_best.map
#
#    Major changes:  Version 2.08 allows you to build
#           outside of a region defined by a model.
#           Version 9 allows definition of a mask with a PDB file
#
#======================================================================
#     EDIT THE NEXT FEW SETS OF LINES TO MATCH YOUR DATA AND SYSTEM
```

```
#
#     PLEASE NOTE: each of the " = " below must have a SPACE on either
#       side (hklin = solve.mtz  NOT hklin=solve.mtz )
#
#==============================================================================
#
#  Anything special for your location (SOLVEDIR etc);
#
setenv SOLVEDIR /usr/local/lib/solve/
setenv TMPDIR .                         # large scratch files go here
#
# location of resolve and resolve libraries;  place for large scratch files
#
set resolve =  resolve
set resolve_pattern =  resolve_pattern
#
#   location of refmac5
#
set refmac5 = refmac5 # NONE if you do not have refmac5
#
#   Input mtz file with at least FP and SIGFP and PHIB
#
set hklin = ../fobs.mtz
set labin = 'FP=FP SIGFP=SIGFP '
set labin_cont = ''
#
#   Optional input model to be rebuilt
#
set pdb_in =  ../2gn5.pdb #  NONE if no model to rebuild
#
#   resolution, solvent_content, sequence, heavy-atom file
#
set dmin =  2.6                  #   high-res limit
set dmax = 200                   #   low-res limit
set solvent_content = 0.40       #   solvent content (required)
set seq_file = ../seq.dat        #   amino acid sequence file (up to 80 char/line)
set ha_file =  NONE              #     e.g., ha.pdb or NONE; used to get NCS
#
set refine_model =  YES        #  say NO if resolution > 3 A usually
set compare_file =  ../coords.pdb #   NONE if no comparison file
#
#    Just build outside the region defined by input model? (Only use if input model has
#    side-chains. Not for a backbone model)
#
set mask_pdb_file = NONE   # PDB file specifying mask for density modification
#
#   criteria for ending..
#
set n_cycle_build_max = 20   # try 20
set n_cycle_build_min = 3    # try 3
set unacceptable_r = 0.4 # try 0.4 keep going if R > unacceptable_r
set n_cycle_rebuild_max = 50  # try 30 for build, 50 for rebuild
set n_cycle_rebuild_min = 20  # try 20 (Rebuild skipped if R < unacceptable_r)
set n_cycle_rebuild_omit = 10 #  every n'th cycle do omit  if
                              # omit_on_rebuild is YES (starting with nth cycle)
#
#   misc optional parameters...
#
set use_resolve_fragments = AUTO  #  AUTO or YES or NO  (AUTO usually)
set use_resolve_pattern   =  AUTO #  AUTO or YES or NO  (AUTO usually)
#
set superquick_build =     NO #  usually NO. Use YES to get a quick answer
#
set clean_up = YES          # YES usually
set extra_verbose = NO      # only for debugging
set omit_on_rebuild = NO    # NO usually
set quick_as_possible = NO  # NO usually
set n_cycle_image_min = 3   # 3 usually use image info for at least these cycles
set cycle_start = 1         # 1 usually
set evaluate_only =  NO   # NO usually; YES ok; NEVER skip
set use_prob_mask = NO    # NO usually; yes for version 2.0 probability mask
#
#=============================================================================
#              Normally no need to edit below here...
#=============================================================================
#
set skip =  NO
#
set main_version_minimum = 2
set minor_version_minimum = 08
set build_outside_model = NO  # Does not always work. refmac5 fails if resolve writes a residue
                              # with same residue number as a residue in pdb_in
#
setenv SYMOP $SOLVEDIR/symop.lib
setenv SYMINFO $SOLVEDIR/syminfo.lib
setenv CCP4_OPEN UNKNOWN
unlimit
limit coredumpsize 0
#
if ( $extra_verbose == YES ) then
 set ip = 1
else
 set ip = 0
endif
#------------------------------------------------------------
#  figure out if this machine uses grep -a or just grep for text files:
echo "A" > test_a.dat
 set test_grep = `grep -a "a" test_a.dat >& tmp.dat`
```

```
if ( $status ) then
#  there was an error...do not use grep -a
 set grep_type = "grep"
else
 set grep_type = "grep -a"
endif
set test_grep = `$grep_type "A" test_a.dat`
if ( $#test_grep != 1 ) then
 echo "Sorry, unable to set the grep command on this system...giving up"
 exit
endif
rm -f test_a.dat
#-------------------------------------------------------------
#  check for all the library files we will need etc
#
if ( $resolve == NONE || $resolve_pattern == NONE ) then
 echo "Sorry, you need to define resolve and resolve_pattern..."
 exit
endif
foreach program ($resolve $resolve_pattern $refmac5)
if ( $ip ) echo "Checking for the program $program ..."
if ( -f $program ) then
 if ( $ip ) echo "OK"
 goto ok
endif
if (  $program == NONE ) then
 if ( $ip ) echo "This program is not used"
 goto ok
endif
which $program >& tmp.dat
set test = `head -1 tmp.dat`
if ( $#test != 1 ) goto bad
if ( -f $test ) then
 if ( $ip ) echo "OK"
 goto ok
endif
bad:
 echo "Sorry, the program $program does not exist?"
 echo "Please check its definition in this script..."
 exit
endif
ok:
end
#
#
foreach lib_file ( symop.lib segments/rho.list \
 segments/fragment_1_0.pdb segments/fragment_2_0.pdb  \
 segments/segment_001_1.pdb segments/segment_001_2.pdb \
 segments/side_chains_best.dat segments/side_chains_rho.dat \
 segments/gap_library.dat \
 segments/template_30_mean_1.ezd segments/template_30_mask_1.ezd \
 segments/template_30_corr_1.ezd segments/template_30_mean_2.ezd \
 segments/template_30_mask_2.ezd segments/template_30_corr_2.ezd \
 hist.sol_91 hist.sol_92 \
 patterns/hist_cc_0.8.dat patterns/index_95.dat \
 patterns/patterns.info patterns/box001.dat )
#
if ( $ip ) echo "Checking for library file $SOLVEDIR/$lib_file..."
if ( ! -f $SOLVEDIR/$lib_file) then
 echo "Sorry, the library file $lib_file does not exist?"
 echo "Please check the definition of SOLVEDIR in your script: $SOLVEDIR"
 exit
endif
end
#------------------------------------------------------------
#
if ( $quick_as_possible == YES ) then
 echo "Trying to do everything as quickly as possible..."
 echo "Setting superquick_build, no refinement, no rebuild, no pattern ID,"
 echo "no fragment ID, minimum cycles"
 set refine_model = NO
 set superquick_build = YES
 set use_resolve_pattern = NO
 set use_resolve_fragments = NO
endif
if ( $evaluate_only == YES ) then
 echo "Just evaluating the model $pdb_in"
 set n_cycle_rebuild_max = 0
 set n_cycle_rebuild_min = 0
endif
if ( ( $pdb_in != NONE ) && ( $evaluate_only == YES ) && ( ! -f $pdb_in ) ) then
 echo "Sorry cannot find the model $pdb_in ... required for evaluate_only"
 exit
endif
if ( $build_outside_model == NO || $pdb_in == NONE )then
  # do nothing
else
 echo "Building only around model $pdb_in (excluding region in existing model)"
endif
if ( -f HEADER.pdb ) rm -f HEADER.pdb
if ($pdb_in != NONE )then
$grep_type CRYST1 $pdb_in >>HEADER.pdb
$grep_type ORIGX1 $pdb_in >>HEADER.pdb
$grep_type ORIGX2 $pdb_in >>HEADER.pdb
$grep_type ORIGX3 $pdb_in >>HEADER.pdb
$grep_type SCALE1 $pdb_in >>HEADER.pdb
$grep_type SCALE2 $pdb_in >>HEADER.pdb
$grep_type SCALE3 $pdb_in >>HEADER.pdb
```

```
#  make sure the headers exist...
@ header_lines = `cat HEADER.pdb |wc -l`
if ( $header_lines >= 4 ) goto ok1
 echo "The input PDB file $pdb_in needs to have at least 4 lines of headers with "
 echo "CRYST1 SCALE1 SCALE2 SCALE3"
 echo "Yours seems to have instead $header_lines of headers"
 cat HEADER.pdb
 exit
ok1:
echo "REMARK RESOLVE MODEL" >> HEADER.pdb
endif
#
if ( $n_cycle_build_min < 1 ) then
   echo "Sorry, please set n_cycle_build_min to at least 1"
   exit
endif
if ( $n_cycle_build_max < 1 ) then
   echo "Sorry, please set n_cycle_build_max to at least 1"
   exit
endif
if ( $n_cycle_rebuild_min < 0 ) then
   set n_cycle_rebuild_min = 0
endif
if ( $n_cycle_rebuild_max < 0 ) then
    set n_cycle_rebuild_max = 0
endif
#
#
set past_cycle = $n_cycle_build_min
set n_cycle = $n_cycle_build_max
set n_cycle_rebuild = $n_cycle_rebuild_max
#   (set n_cycle to n_cycle_rebuild_max later)
#
set minimum_coverage = 0
#
#
#
# check for resolve and resolve_pattern
if ( -f resolve.ok ) rm -f resolve.ok
if ( -f resolve.version ) rm -f resolve.version
$resolve <<EOD  >& resolve.log_0
 quit
EOD
if ( ! -f resolve.ok ) then
 echo "Sorry please check the location of resolve..."
 echo "    is it really $resolve?"
 exit
endif
#  make sure this sort of worked...
```

```
set test_ok = `cat resolve.ok`
set test_ok_mem = $test_ok[$#test_ok-$#test_ok]
if ( $test_ok_mem != "ALLOCATED" ) then
 echo "Sorry, resolve was not able to run properly at all"
 echo "The end of the resolve log file says..."
 echo "-------------------------------------------------------"
 tail -12 resolve.log_0
 echo "-------------------------------------------------------"
 exit
endif
if ( ! -f resolve.version ) then
 echo "Sorry, this script requires version $main_version_minimum.$minor_version_minimum or higher of resolve"
 exit
endif
@ version = `cat resolve.version|head -1`
set version_minimum = "$main_version_minimum$minor_version_minimum"
if ( $version < $version_minimum )then
 echo "Sorry, this script requires version $main_version_minimum.$minor_version_minimum or higher of resolve"
 exit
else
 if ( $ip ) echo "Running version $version of resolve"
endif
if ( -f resolve_pattern.ok ) rm -f resolve_pattern.ok
$resolve_pattern <<EOD >& /dev/null
end
EOD
if ( ! -f resolve_pattern.ok ) then
 echo "Sorry please check the location of resolve_pattern... "
 echo "   is it really $resolve_pattern?"
 exit
endif
#
# make sure we have these files...
#
if ( -f $hklin ) then
   echo "Data are in the mtz file $hklin"
else
   echo "Sorry, cannot find the mtz file $hklin"
   exit
endif
echo "LABIN information: $labin $labin_cont"
echo " "
#
if ( $seq_file != NONE &&  -f $seq_file  ) then
  echo "Sequence information from ${seq_file}"
else if ( $seq_file != NONE ) then
  echo "Sorry, cannot find the seq_file file $seq_file"
  exit
endif
```

```
#
if ( ( $ha_file != NONE ) && ( -f $ha_file  ) ) then
  echo "Atom information for NCS from $ha_file"
else if ( $ha_file != NONE ) then
  echo "Sorry, cannot find the ha_file file $ha_file"
  exit
endif
#
if ( ( $pdb_in != NONE ) && ( ! -f $pdb_in ) ) then
 echo "Sorry, cannot find your pdb_in file $pdb_in"
 exit
else if ( ( $cycle_start == 1 ) && ( $pdb_in != NONE ) ) then
 set use_resolve_fragments = NO
 set use_resolve_pattern   =  NO
endif
#
if ( $superquick_build == YES ) then
 set superquick_build = "superquick_build"
else
 set superquick_build = " "
endif
#
@ grep_phib = `echo $labin $labin_cont | $grep_type 'PHIB='|wc -m`
@ grep_fp = `echo $labin $labin_cont | $grep_type 'FP='|wc -m`
#
if ( $grep_fp == 0 ) then
 echo "Sorry, FP is always required in the labin line..."
 exit
endif
if ( $grep_phib == 0 && $pdb_in == NONE ) then
 echo "Sorry if you have no pdb_in file you need phases (PHIB=...)"
 exit
else if ( $pdb_in != NONE ) then
 echo "Using  phase probabilities from $hklin in rebuilding"
endif
if ( $pdb_in == NONE ) then
 set scale = 1.0
else if ( $grep_phib != 0 ) then
 set scale = 1.0
else
 set scale = 0.2
endif
#
echo " "
echo "solvent content ${solvent_content}"
echo "resolution ${dmin} ${dmax}"
if ( $refine_model == NO ) then
 echo "Model will not be refined"
else
```

```
 echo "Model will be refined"
endif
echo " "
if ( $compare_file != NONE ) then
 echo "Coordinates for comparison from $compare_file"
endif
if ( $compare_file != NONE && ! ( -f $compare_file ) ) then
  echo "Sorry, cannot find the comparison PDB file $compare_file"
  exit
endif
#
set blank = " "
#
if ( $pdb_in != NONE ) then
  set combine_resolve_fragments = $blank
  echo "Not using fragment ID"
else if ( $use_resolve_fragments == YES ) then
  set combine_resolve_fragments = "combine_map fragments.map"
  echo "Using fragment ID"
else if ( $use_resolve_fragments == AUTO ) then
  set combine_resolve_fragments = "combine_map fragments.map"
  echo "Testing using fragment ID"
else
  set combine_resolve_fragments = $blank
  echo "Not using fragment ID"
endif
if ( $pdb_in != NONE ) then
  set combine_resolve_pattern = $blank
  echo "Not using resolve_pattern"
else if ( $use_resolve_pattern == YES) then
  set combine_resolve_pattern = "combine_map pattern.map"
  echo "Using resolve_pattern"
else if ( $use_resolve_pattern == AUTO) then
  set combine_resolve_pattern = "combine_map pattern.map"
  echo "Testing using resolve_pattern"
else
  set combine_resolve_pattern = $blank
  echo "Not using resolve_pattern"
endif
#  make n_unacceptable_r  have 5 digits exactly
set n_unacceptable_r = $unacceptable_r:e
if ( $n_unacceptable_r < 1 ) then
 set n_unacceptable_r = 10000
endif
while ( $n_unacceptable_r < 10000 )
 @ n_unacceptable_r = $n_unacceptable_r * 10
end
if ( ( $pdb_in != NONE ) && ( $evaluate_only != YES ) ) then
 echo "If R-factor < 0.$n_unacceptable_r then no rebuilding will be done"
 echo "If R-factor >= 0.$n_unacceptable_r then maximum cycles will be done"
endif
if ($mask_pdb_file != NONE  && ! -f $mask_pdb_file)then
  echo "Sorry, could not find the mask pdb file $mask_pdb_file"
  exit
else if ($mask_pdb_file != NONE )then
  set mask_line = "model $mask_pdb_file"
  set use_mask_line = "use_model_mask"
  echo "Using $mask_pdb_file to define mask for density modification"
else if ($use_prob_mask == YES )then
  set mask_line = ""
  set use_mask_line = "use_prob"
  echo "Using probability-based mask (version 2.0)"
else
  set mask_line = ""
  set use_mask_line = ""
endif
#
#
#          SETUP:
#
# ==========================================================================
# ==========================================================================
# hklin_image will be our combined phase information
#
set hklin_image = start_image.mtz
set labin_image = 'FP=FP PHIB=PHIM FOM=FOMM HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM'
set labin_image_cont = 'FreeR_flag=FreeR_flag'
#
#  hklin_exp is the file we will use to evaluate phase quality.
set hklin_exp = exp.mtz
set labin_exp = 'FP=FP PHIB=PHIM FOM=FOMM HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM'
set labin_exp_cont = 'FreeR_flag=FreeR_flag'
#
set build_score_best_overall = -1000
set build_score_best_overall_cycle = -1000
set r_best_overall = 99999
#
# ==========================================================================
set cycle = $cycle_start
set cycle_prev = $cycle
@ cycle_prev --
# ==========================================================================
if ( $cycle_start == 1 ) then
 goto cycle_one
endif
echo "Starting at cycle $cycle_start"
#
cycle_one:
```

```
# ===========================================================================
set jump = NO
if ( $jump == YES ) then
set best_cc_value = 0000
set read_mask = " "
echo "JUMPING TO MIDDLE OF FILE AT CYCLE $cycle_start"
endif
if ( $jump == YES ) goto jump_point
# ===========================================================================
#
while ( $cycle <= $n_cycle )
#
set cycle_prev = $cycle
@   cycle_prev --
#
 if ( $ip ) echo " "
if ( $ip ) echo "Cycle $cycle -------------------------------------------"
#
#
if ( -f resolve.log_$cycle ) then
 if ( $ip ) echo "removing previously-existing log file resolve.log_$cycle"
 rm -f resolve.log_$cycle
endif
set build_score_best = -1000
set placed_best = 0
set built_best = 0
if ( -f work_model.pdb ) then
  rm -f work_model.pdb
endif
#
if ( $cycle > 1 ) then
 goto cycle_point
endif
#
# ===========================================================================
#
#  First cycle:
#
#   remove all the files from previous runs..
if ( -f pattern.map ) then
 if ( $ip ) echo "removing existing pattern.map"
 rm -f pattern.map
endif
if ( -f fragments.map ) then
 if ( $ip ) echo "removing existing fragments.map"
 rm -f fragments.map
endif
if ( -f combine.map ) then
 if ( $ip ) echo "removing existing combine.map"
 rm -f combine.map
endif
if ( -f current_model.pdb ) then
  if ( $ip ) echo "Removing current_model.pdb"
  rm -f current_model.pdb
endif
if ( -f resolve_peaks.dat ) then
  if ( $ip ) echo "removing resolve_peaks.dat"
  rm -f resolve_peaks.dat
endif
if ( -f resolve_peaks.use) then
  if ( $ip ) echo "removing resolve_peaks.use"
  rm -f resolve_peaks.use
endif
if ( -f fragments.dat ) then
  if ( $ip ) echo "removing fragments.dat"
  rm -f fragments.dat
endif
foreach best_file (resolve_best.pdb resolve_best.log resolve_best.mtz \
    refmac_best.log )
if ( -f $best_file ) rm -f $best_file
end
#
#===========================================================================
echo "Copying phase information from $hklin to $hklin_exp"
${resolve}<<EOD >>resolve.log_0
! Copying over $hklin to $hklin_exp
hklin $hklin
labin $labin
labin $labin_cont
hklout $hklin_exp
solvent_content $solvent_content
resolution $dmin $dmax
res_start $dmin
mask_cycles 1
minor_cycles 0
ha_file NONE
no_build
EOD
#
#  make sure this sort of worked...
if ( `cat resolve.ok` != 'OK' ) then
 echo "Sorry, resolve was not able to finish even the first cycle"
 echo "The end of the resolve log file says..."
 echo "----------------------------------------------------------"
 tail -10 resolve.log_0
 echo "----------------------------------------------------------"
 exit
endif
```

```
#
#  for model start we go right on to rebuilding...cycle 2
#
if (  $pdb_in != NONE )  then
 echo " "
 echo "Going right on to rebuilding/model evaluation now..."
 break
endif
#
#  First cycle just do std density modification...2x with
#   2 sets of parameters
#
set best_dm_type = "quick"
#
if ( $quick_as_possible == YES ) then
 set types = "quick"
else
 set types = "quick"
endif
foreach build_type ( $types )
if ( $build_type == "quick" ) then
 set build_1 = "mask_cycles 3"
 set build_2 = "minor_cycles 1"
else
 set build_1 = $blank
 set build_2 = $blank
endif
echo "Testing $build_type density modification..."
#
if ( -f resolve.built ) rm -f resolve.built
$resolve <<EOD >>resolve.log_$cycle
!  Resolve_build script
!  Cycle $cycle...testing $build_type density modification
resolution $dmin $dmax
hklin $hklin
labin $labin
labin $labin_cont
solvent_content $solvent_content
ha_file $ha_file
seq_file $seq_file
$superquick_build
compare_file $compare_file
hklout dummy.mtz
$build_1
$build_2
$mask_line
$use_mask_line
EOD
#
```

```
#  make sure this sort of worked...
if ( `cat resolve.ok` != 'OK' ) then
 echo "Sorry, resolve was not able to finish even the first cycle"
 echo "The end of the resolve log file says..."
 echo "-----------------------------------------------------"
 tail -10 resolve.log_0
 echo "-----------------------------------------------------"
 exit
endif
#
#   score this build:
#
if ( -f resolve.built ) then
    set built = `cat resolve.built|head -1`
    set placed = `cat resolve.built | head -2|tail -1`
    set build_score = `cat resolve.built | head -4|tail -1`
 echo "Score for $build_type : $build_score   built: $built  placed: $placed"
else
 set build_score = 0
 set built = 0
 set placed = 0
endif
#
if ( $build_score > $build_score_best ) then
 set best_dm_type = $build_type
 set build_score_best = $build_score
 set built_best = $built
 set placed_best = $placed
 cp dummy.mtz resolve.mtz_$cycle
 cp resolve.mtz_$cycle omit.mtz
 cp resolve_peaks.dat resolve_peaks.use
 cp resolve.pdb  work_model.pdb
endif
#
end
echo "Best density modification obtained with $best_dm_type approach"
echo $best_dm_type > resolve.type
#
echo " "
echo "Starting building at cycle $cycle"
echo " "
echo "--------------------------------------------------------"
echo "         RESIDUES  SIDE-CHAINS"
echo " CYCLE     BUILT      PLACED        SCORE      R       Free R   BEST"
#
goto finish
#
#==================================================================
cycle_point:
```

```
#
# Figure out if we need to try alternate ncs:
if ( -f resolve.ncs_used ) then
 set ncs_used = `cat resolve.ncs_used|head -1`
else
 set ncs_used = NO
endif
if ( ( $ncs_used == NO ) &&  ( -f ncs_test.pdb ) ) then
 set ha_file = ncs_test.pdb
 echo "Trying NCS from $ha_file based on last model built..."
endif
#
#  for first few cycles always use image info if available...
if ( ( $cycle <= $n_cycle_image_min) && ( -f combine.map ) ) then
 goto use_image
endif
#
# density-modify with model information only:
#
if ( -f resolve.built ) rm -f resolve.built
${resolve}<<EOD>>resolve.log_$cycle
! map with model information only cycle $cycle
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
hklout resolve.mtz_$cycle
solvent_content $solvent_content
ha_file $ha_file
seq_file $seq_file
$superquick_build
compare_file $compare_file
composite_all
composite_pdb refine.pdb_
composite_pdb_first 1
composite_pdb_last $cycle_prev
pdb_in refine.pdb_$cycle_prev
image
mask_cycles 1
nohl
$mask_line
$use_mask_line
EOD
#
if ( -f resolve.built ) then
    set built = `cat resolve.built|head -1`
    set placed = `cat resolve.built | head -2|tail -1`
    set build_score = `cat resolve.built | head -4|tail -1`
 if ( $ip ) echo "Build score with model but no image info: $build_score   built: $built  placed: $placed"
else
```

```
 set build_score = 0
 set built = 0
 set placed = 0
 echo "REMARK No model built" > resolve.pdb
 echo "REMARK No model built"
endif
#
set best_method = "no_image"
set build_score_best = $build_score
set built_best = $built
set placed_best = $placed
cp resolve_peaks.dat resolve_peaks.use
cp resolve.pdb  work_model.pdb
#
#  Include model information with image info and build a new model:
#
if ( ! -f combine.map ) then
 goto skip_image
endif
#
use_image:
#
#
 if ( $ip ) echo "Adding in information from combine.map to $hklin_image"
#
${resolve}<<EOD >>resolve.log_$cycle
! applying combine info cycle $cycle
resolution $dmin $dmax
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
solvent_content $solvent_content
pattern_phase
hklout $hklin_image
cc_map_file combine.map
ha_file NONE
no_build
EOD
#
if ( -f resolve.built ) rm -f resolve.built
${resolve}<<EOD>>resolve.log_$cycle
! Get map with model and image information cycle $cycle
hklin $hklin_image
labin $labin_image
labin $labin_image_cont
hklout dummy.mtz
solvent_content $solvent_content
ha_file $ha_file
seq_file $seq_file
```

```
$superquick_build
compare_file $compare_file
composite_all
composite_pdb refine.pdb_
composite_pdb_first 1
composite_pdb_last $cycle_prev
pdb_in refine.pdb_$cycle_prev
image
mask_cycles 1
nohl
$mask_line
$use_mask_line
EOD
#
#   score this build:
#
if ( -f resolve.built ) then
    set built = `cat resolve.built|head -1`
    set placed = `cat resolve.built | head -2|tail -1`
    set build_score = `cat resolve.built | head -4|tail -1`
 if ( $ip ) echo "Score for build using image info  : $build_score   built: $built  placed: $placed"
else
 set build_score = 0
 set built = 0
 set placed = 0
endif
if ( $build_score > $build_score_best ) then
 set best_method = "image"
 set build_score_best = $build_score
 set built_best = $built
 set placed_best = $placed
 cp resolve_peaks.dat resolve_peaks.use
 cp resolve.pdb  work_model.pdb
 cp dummy.mtz resolve.mtz_$cycle
 if ( $ip ) echo "Including image information"
else if ( $use_resolve_fragments == AUTO ) then
 echo "No longer including image information"
  set combine_resolve_pattern = $blank
  set combine_resolve_fragments = $blank
  rm -f combine.map
  goto skip_image
else
 if ( $ip ) echo "Not using image information on this cycle"
endif
#
# Also get the omit map if we are using images...
#
if ( $ip ) echo "Getting composite ps omit map..."
if ( -f omit.mtz ) then
```

```
 rm -f omit.mtz
endif
$resolve <<EOD >>resolve.log_$cycle
!  Cycle $cycle...composite prime-and-switch omit density modification
hklin $hklin_image
labin $labin_image
labin $labin_image_cont
hklout omit.mtz
solvent_content $solvent_content
no_build
ha_file $ha_file
composite_all
composite_pdb refine.pdb_
composite_pdb_first 1
composite_pdb_last $cycle_prev
add_mask
image
omit
mask_cycles 1
nohl
$mask_line
$use_mask_line
EOD
#
skip_image:
#
finish:
#
#  Start of refine-extend cycles
#
foreach extend_cycle ( 0 1 2)
#
if ( $ip ) echo "Extend cycle $extend_cycle"
if( $extend_cycle > 0 ) then
 if ( $ip ) echo "Extending model"
#
if ( -f resolve.built ) rm -f resolve.built
${resolve}<<EOD >> resolve.log_$cycle
hklin resolve.mtz_$cycle
labin FP=FP PHIB=PHIM FOM=FOMM
hklout dummy.mtz
solvent_content $solvent_content
resolution ${dmin} ${dmax}
extend_only
ha_file $ha_file
seq_file $seq_file
compare_file $compare_file
pdb_in resolve.pdb
no_trim
```

```
nohl
EOD
endif
if ( ( $extend_cycle > 0 ) && ( -f resolve.built ) ) then
    set built = `cat resolve.built|head -1`
    set placed = `cat resolve.built | head -2|tail -1`
    set build_score = `cat resolve.built | head -4|tail -1`
 if ( $ip ) echo "Score for extension: $build_score   built: $built  placed: $placed"
else if ( $extend_cycle > 0 ) then
 set build_score = 0
 set built = 0
 set placed = 0
endif
if ( ( $extend_cycle > 0 ) && ( $build_score > $build_score_best ) ) then
 set build_score_best = $build_score
 set built_best = $built
 set placed_best = $placed
 cp resolve.pdb work_model.pdb
else if ( $extend_cycle > 0 ) then
 if ( $ip ) echo "Extension did not improve score...going on"
#  break to end of extend cycles..
 break
endif
#
#
#  decide if we refine model this cycle..
#
if ( -f resolve.coverage ) then
 set coverage1 = `cat resolve.coverage|head -3|tail -1`
 set coverage = 0
 if ( $#coverage1> 0 ) set coverage= $coverage1[$#coverage1-$#coverage1]:e
else
 set coverage = 0
endif
if ( $refine_model == NO ) then
 set refine_it = NO
else if ( $build_score_best < 2 ) then
 set refine_it = NO
else if (  $coverage < $minimum_coverage )then
 set refine_it = NO
else if ( ! -f work_model.pdb ) then
 set refine_it = NO
 echo "REMARK No model built" > work_model.pdb
 echo "REMARK No model built extend cycle $extend_cycle"
else
 set refine_it = YES
endif
if ( $refine_it == YES ) then
 set ncyc = 20
```

```
else
 set ncyc = 0
endif
if ( ( $refmac5 != NONE ) && ( $build_score_best > 1 ) ) then
if ( $ip ) echo "Refining model $ncyc cycles and placing it in refine.pdb_$cycle"
#
cat work_model.pdb | $grep_type -v HETATM > resolve.pdb
#
# --------refine with refine.--------------
#
#  NOTE: refine against "best" map = resolve.mtz_$cycle
#
$refmac5 xyzin resolve.pdb hklin resolve.mtz_$cycle xyzout refine.pdb  \
PROTOUT $CCP4_SCR/protout.dat \
PROTCOUNTS $CCP4_SCR/counts.dat \
PROTSCR $CCP4_SCR/counts.scr \
<<EOD > refmac.log_${cycle}
MAKE HYDRogens No
MAKE CHECK 0
LABI FP=FP SIGFP=SIGFP HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM FREE=FreeR_flag
REFI TYPE RESTrained RESOlution ${dmax} ${dmin}
#   Maximum likelihood refinement
REFI RESI MLKF
#   Overall B value refinement
REFI BREF OVER      ! Refine overall B-values
#   Tight restraints  Lowered further 051403
WEIG MATR 0.05
#   Scale down shifts at every cycle by factor 0.5
DAMP 0.5 0.5
#   No Blur (scale down reliability of) phases
REFI PHASed  BBLUrring 00.0  SCBLurring 1.
#  Babinet's bulk solvent scale parameters. But fix B value
SCALe TYPE BULK
SCALe LSSCale FIXBulk BBULk 200.0
#   Anisotropic scaling. It is default
SCALe LSSCale ANISotropic
#   Fix Babinet's bulkd solvent for sigmaA
SCAL MLSC FIXBulk BBULk 200.0 SCBUlk -0.05
NCYC  $ncyc
MONI MEDI
BINS 10
EOD
#--------------------------------------------------------
#
cp refine.pdb refine.pdb_$cycle
else if (  $build_score_best > 1  ) then
#   no refmac5...just copy over
 cat work_model.pdb | $grep_type -v HETATM > resolve.pdb
 cp resolve.pdb refine.pdb
 cp refine.pdb refine.pdb_$cycle
else
if ( $ip ) echo "No model built this try"
 echo "REMARK  No model built cycle $cycle " > refine.pdb_$cycle
 echo "REMARK  No model built cycle $cycle "
endif
#
end_of_extend:
end
#
#
#  Get R and freeR value on refine.pdb_$cycle:
#
set freer_line = `$grep_type 'FREE R VALUE                     :' refine.pdb_$cycle|tail -1`
set r_line = `$grep_type 'R VALUE            (WORKING SET) :' refine.pdb_$cycle|tail -1`
if ( $#freer_line > 0 ) then
 set freer = $freer_line[$#freer_line-$#freer_line]:e
else
 set freer = 99999
endif
if ( $#r_line > 0 ) then
 set r = $r_line[$#r_line-$#r_line]:e
else
 set r = 99999
endif
if ( $ncyc == 0 )then
 set freer = $r
endif
#
if ( $ip ) echo " R/freeR for this model: 0.$r/0.$freer"
#
if ( $cycle < 10 ) then
 set b1 = " "
else
 set b1 = ""
endif
if ( $built_best < 100 ) then
 set b2 = " "
else
 set b2 = ""
endif
if ( $placed_best >= 100 ) then
 set b3 = ""
else if ( $placed_best >= 10 )then
 set b3 = " "
else
 set b3 = "  "
endif
if ( $build_score_best < 100 )then
```

```
 set b4 = " "
else
 set b4 = ""
endif
# Decide how to score. If R < unacceptable_r or r_best_overall < unacceptable_r
#   then use R. Otherwise use score
#
if ( ( $r < $n_unacceptable_r ) && ( $r < $r_best_overall ) ) then
 set this_is_best = YES
else if ( ( $r_best_overall >= $n_unacceptable_r ) && \
   ( $build_score_best > $build_score_best_overall ) ) then
 set this_is_best = YES
else
 set this_is_best = NO
endif
if ( $this_is_best == YES ) then
 if ( $ip ) echo "This is the best model overall so far. Previous best had score of $build_score_best_overall"
 if ( $ip ) echo "Copied refine.pdb_$cycle to resolve_best.pdb"
 if ( $ip ) echo "Copied resolve.log_$cycle resolve_best.log"
 if ( $ip ) echo "Copied resolve.mtz_$cycle to resolve_best.mtz"
 @ build_score_best_overall = $build_score_best
 @ build_score_best_overall_cycle = $cycle
 @ r_best_overall = $r
 cp refine.pdb_$cycle resolve_best.pdb
 cp resolve.log_$cycle resolve_best.log
 cp resolve.mtz_$cycle resolve_best.mtz
 if( -f refmac.log_$cycle ) cp refmac.log_$cycle refmac_best.log
 echo "  ${b1}$cycle      ${b2}$built_best        ${b3}$placed_best         ${b4}$build_score_best     0.$r    0.$freer    *"
else
 echo "  ${b1}$cycle      ${b2}$built_best        ${b3}$placed_best         ${b4}$build_score_best     0.$r    0.$freer    "
endif
#
#
#=============================================================================
#=============================================================================
#
#
set have_map = NO
if ( ( "$combine_resolve_fragments" != $blank ) && ( -f resolve_peaks.dat )  )  then
if ( $ip ) echo "Carrying out fragment ID based on resolve.mtz_$cycle"
cp resolve_peaks.use resolve_peaks.dat
#
$resolve <<EOD >> resolve.log_$cycle
hklin resolve.mtz_$cycle
labin FP=FP PHIB=PHIM FOM=FOMM
solvent_content $solvent_content
nohl
build_image
noget_peaks      ! Don't need peaks because we just got them!
```

```
mask_cycles 0
minor_cycles 0
superquick_build
ha_file NONE
$mask_line
$use_mask_line
EOD
cp dump.map fragments.map
set have_map = YES
endif
#
if ( "$combine_resolve_pattern" != $blank ) then
if ( $ip ) echo "Carrying out pattern ID based on omit.mtz"
$resolve_pattern<<EOD >> resolve.log_$cycle
hklin omit.mtz
labin FP=FP PHIB=PHIM FOM=FOMM
recover_image
cc_map_file dump.map
EOD
cp dump.map pattern.map
set have_map = YES
endif
#
#  Decide which to combine..
#
if ( $have_map == YES ) then
if ( $ip ) echo "Combining all image information into combine.map"
#
$resolve<<EOD >> resolve.log_$cycle
hklin omit.mtz
labin FP=FP PHIB=PHIM FOM=FOMM
$combine_resolve_pattern
$combine_resolve_fragments
ha_file NONE
EOD
#
else
 if ( $ip ) echo "No image information generated this cycle"
 if ( -f combine.map ) rm -f combine.map
endif
#
#
#  Decide if we should keep going or quit...
#   quit if
#   r_best_overall < n_unacceptable_r and
#   it has been at least past_cycle cycles since things got better
#
@ cutoff_cycle = $cycle - $past_cycle
if ( ( $build_score_best_overall_cycle <= $cutoff_cycle ) && ( $r_best_overall < $n_unacceptable_r ) ) then
```

```
 echo "Best model was built on cycle $build_score_best_overall_cycle which was"
 echo "   more than $past_cycle cycles ago..."
 @ cycle ++
 break
endif
#
end_of_cycle:
#
#  clean up...
#
if ( $clean_up == YES  && -f resolve.log_$cycle ) then
 if ( -f resolve.log_$cycle ) rm -f resolve.log_$cycle
 if ( -f refmac.log$cycle ) rm -f refmac.log_$cycle
 if ( -f resolve.mtz_$cycle ) rm -f resolve.mtz_$cycle
endif
#===========================================================================
#
if ( $ip ) echo "END OF CYCLE $cycle"
@ cycle ++
end
#
if ( $pdb_in != NONE ) then
 echo "Copying $pdb_in to resolve_group.pdb and resolve_best.pdb for rebuilding/evaluation..."
 cp $pdb_in resolve_best.pdb
 cp $pdb_in resolve_group.pdb
 goto start_rebuild
endif
#
echo "" > resolve_group.pdb
@ cyc = $cycle - 5
if ( $cyc < 1 ) set cyc = 1
while ( $cyc < $cycle )
 echo "Adding refine.pdb_$cyc to resolve_group.pdb"
 cat refine.pdb_$cyc >>resolve_group.pdb
 @ cyc ++
end
echo "Starting PDB group model for rebuild is in resolve_group.pdb"
#
if ( -f resolve_best.pdb ) then
 echo "Best model was built on cycle $build_score_best_overall_cycle"
 echo "This model is now in resolve_best.pdb"
else
 echo "No model was built...quitting"
 exit
endif
#
#===================================================================
if (  $quick_as_possible == YES ) then
 goto wrap_up
```

```
endif
if (  $r_best_overall < $n_unacceptable_r ) then
 goto wrap_up
endif
if ( $n_cycle_rebuild_max == 0 ) then
 goto wrap_up
endif
 echo " "
 echo "Our best R-factor (0.$r_best_overall) is > cutoff of $unacceptable_r"
 echo "so we are going to try and rebuild the model..."
 echo " "
#============================================================
#
start_rebuild:
#
set past_cycle = $n_cycle_rebuild_min
@ cycle_start = $cycle + 1
@ n_cycle = $cycle_start + $n_cycle_rebuild - 1
#
#
#
set pdb_start = resolve_group.pdb           #   starting PDB file
#
if ( $pdb_in == NONE ) then
 set scale = 1.0
 set extend = NO
 set reuse_chain = NO
else if ( $grep_phib != 0 ) then
#   we have phases, so use scale = 1.0...but we are rebuilding model, so
#    use extend and reuse_chain to get maximum building
 set scale = 1.0
 set extend = YES
 set reuse_chain = YES
else
#  slightly different rules for rebuilding from scratch from a model
#   without any phase information:
 set scale = 0.2
 set extend = YES
 set reuse_chain = YES
endif
#
set r_max = 0.$r_best_overall # maximum free R to write anything to resolve_best.pdb
#
if ( $ip ) echo "Output density-modified phases will be in image_only_dm_image.mtz_[cycle]"
#
#
if ( $evaluate_only != YES && $extend == NO )then
 echo "Model will not be extended each cycle"
 set extend_flag = ' 0 '
```

```
else if ( $evaluate_only != YES ) then
 echo "Model will be extended each cycle"
 set extend_flag = ' 0 1 2 '
endif
#
if ( $evaluate_only != YES && $omit_on_rebuild == YES )then
 echo "Omit prime-and-switch phasing will be used every  $n_cycle_rebuild_omit cycle"
 set omit = 'omit'
else
 echo "Standard image-based phasing will be used"
 set omit = ' '
endif
if ( $ip ) echo "Scale on map weighting is $scale"
set n_r_max = $r_max:e
if ( $cycle_start > 2 ) then
 echo "Maximum R to save model: 0.$n_r_max "
endif
#
echo " "
#
@ cycle = $cycle_start - 1
@ cycle_pdb_start = $cycle
#
echo "Copying ${pdb_start} to refine.pdb_$cycle"
cp ${pdb_start} refine.pdb_$cycle
#
#================================================================
@ cycle ++
if ( $evaluate_only != YES ) then
echo "Starting rebuilding at cycle $cycle of total of $n_cycle cycles"
#========================================================
#========================================================
#
echo " "
echo "------------------------------------------------------------"
echo "         RESIDUES  SIDE-CHAINS"
echo " CYCLE     BUILT      PLACED        SCORE      R       Free R   BEST"
endif
#
set n_omit_cycle = 0
while ( $cycle <= $n_cycle )
@ n_omit_cycle ++
if ( $n_cycle_rebuild_omit == 0 ) then
#  is_omit_cycle is NO for this one; defaults for omit_use and pdb_in
#
 set is_omit_cycle = NO
 set omit_use = $omit
 set pdb_in_current_model = 'pdb_in current_model.pdb'
else if ( $n_omit_cycle == 1 || $n_omit_cycle == $n_cycle_rebuild_omit ) then
```

```
 set n_omit_cycle = 1
 set is_omit_cycle = YES
 if ( $ip ) echo "Using $omit model phasing on this cycle and building from scratch"

 set omit_use = $omit
 set pdb_in_current_model = ' '
else
 set omit_use = ' '
 set pdb_in_current_model = 'pdb_in current_model.pdb'
 set is_omit_cycle = NO
endif
#
set cycle_prev = $cycle
@ cycle_prev --
#
if ( -f resolve.log_$cycle ) then
 if ( $ip ) echo "removing previously-existing log file resolve.log_$cycle"
 rm -f resolve.log_$cycle
endif
set build_score_best = -1000
set built_best = 0
set placed_best = 0
#
if ( -f work_model.pdb ) then
  if ( $ip ) echo "removing work_model.pdb"
  rm -f work_model.pdb
endif
#
if ( $build_outside_model == YES ) then
# put all current models in all.pdb
if ( -f all.pdb ) rm -f all.pdb
set c_cycle = $cycle_pdb_start
set c_end = $cycle_prev
while ($c_cycle <= $c_end)
if ( -f refine.pdb_${c_cycle} ) cat refine.pdb_${c_cycle} >> all.pdb
@ c_cycle ++
end
set add_mask = 'add_mask'
set model_all = 'model all.pdb'
set build_outside = 'build_outside_model'
else
set add_mask = ''
set model_all = ''
set build_outside = ''
endif
#
#  get image of pdb_start and all models up to now into image.mtz.
#  Reflns from hklin_exp
#
```

```
if ( $ip ) echo "Getting map of ${pdb_start} and all models built into binary file dump.map"
#
${resolve}<<EOD >> resolve.log_$cycle
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
resolution ${dmin} ${dmax}
composite_pdb refine.pdb_
composite_pdb_start $cycle_pdb_start
composite_pdb_end $cycle_prev
composite_all
dump_composite
EOD
#
if ( $ip ) echo "Getting image of ${pdb_start} and built models using dump.map into image_only.mtz"
#
${resolve}<<EOD >> resolve.log_$cycle
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
resolution ${dmin} ${dmax}
no_build
pattern_phase
prior_weight 1.0
hklout image_only.mtz
read_cc_map
cc_map_file dump.map
nohl
EOD
#  make sure this sort of worked...
if ( `cat resolve.ok` != 'OK' ) then
 echo "Sorry, resolve was not able to finish on the first cycle"
 echo "The end of the resolve log file says..."
 echo "-------------------------------------------------------"
 tail -10 resolve.log_$cycle
 echo "-------------------------------------------------------"
 exit
endif
#
if ( $ip ) echo "Density-modifying image of $pdb_start and prev models to get image_only_dm.mtz"
#
${resolve}<<EOD >> resolve.log_$cycle
hklin image_only.mtz
labin FP=FP PHIB=PHIM FOM=FOMM  HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM
hklout image_only_dm.mtz
solvent_content $solvent_content
prior_weight 1.0
database 5
resolution ${dmin} ${dmax}
```

```
$add_mask      ! optionally add mask based on all.pdb (model_all)
$model_all
mask_cycles 1
no_build
nohl
EOD
#
if ( $ip ) echo "Carrying out $omit_use phasing using image_only_dm.mtz "
if ( $ip ) echo " and ${pdb_start} and building model"
#
#
#   either build using this map, or build around starting model only!
#
if ( -f resolve.built ) rm -f resolve.built
if ( -f built_model.pdb ) rm -f built_model.pdb
if ( -f resolve.pdb ) rm -f resolve.pdb
#
if ( $build_outside_model == NO || $pdb_in == NONE )then
 if ($ip) echo "Standard build"
#
${resolve}<<EOD >> resolve.log_${cycle}
!standard build
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
hklstart image_only_dm.mtz
labstart FP=FP PHIB=PHIM FOM=FOMM
hklout image_only_dm_image.mtz_${cycle}
solvent_content $solvent_content
database 5
resolution ${dmin} ${dmax}
composite_pdb refine.pdb_
composite_pdb_start $cycle_pdb_start
composite_pdb_end $cycle_prev
composite_all
prior_weight 1.0
scale_refl $scale
image
$omit_use
mask_cycles 0
$superquick_build
ha_file $ha_file
seq_file $seq_file
compare_file $compare_file
$pdb_in_current_model
nohl
$mask_line
$use_mask_line
EOD
```

```
if ( -f resolve.pdb ) cp resolve.pdb work_model.pdb
if ( -f resolve.pdb ) cp resolve.pdb built_model.pdb
cp HEADER.pdb work_model_start.pdb
#
else
#
 if ($ip) echo "Building around model $pdb_in in 2 steps"
cp HEADER.pdb work_model.pdb
cp HEADER.pdb built_model.pdb
cat $pdb_in |$grep_type ATOM >>work_model.pdb
cat $pdb_in |$grep_type HETATM >>work_model.pdb
cp work_model.pdb work_model_start.pdb
#
${resolve}<<EOD >> resolve.log_${cycle}
! building around model $pdb_in in 2 steps
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
hklstart image_only_dm.mtz
labstart FP=FP PHIB=PHIM FOM=FOMM
hklout image_only_dm_image.mtz_${cycle}
solvent_content $solvent_content
database 5
resolution ${dmin} ${dmax}
composite_pdb refine.pdb_
composite_pdb_start $cycle_pdb_start
composite_pdb_end $cycle_prev
composite_all
prior_weight 1.0
scale_refl $scale
image
$omit_use
mask_cycles 0
no_build
ha_file $ha_file
seq_file $seq_file
compare_file $compare_file
$pdb_in_current_model
$add_mask   ! add mask from model to rmsd mask
$model_all
$mask_line
$use_mask_line
nohl
EOD
#
${resolve}<<EOD >> resolve.log_${cycle}
! Building outside input model $pdb_in only
hklin  image_only_dm_image.mtz_${cycle}
labin FP=FP PHIB=PHIM FOM=FOMM
```

```
ha_file NONE
add_mask
model work_model_start.pdb
database 5
solvent_content $solvent_content
resolution ${dmin} ${dmax}
nohl
build_only
build_outside_model
ha_file $ha_file
seq_file $seq_file
compare_file $compare_file
$superquick_build
EOD
 if ( -f resolve.built)    $grep_type ATOM resolve.pdb  >>work_model.pdb
 if ( -f resolve.built)    $grep_type ATOM resolve.pdb  >>built_model.pdb
endif
#
if ( $omit_use == omit )then
 cp resolve_composite_map.mtz  build.mtz
 if ( $ip ) echo "Copied resolve_composite_map.mtz to build.mtz"
else
 cp image_only_dm_image.mtz_${cycle} build.mtz
 if ( $ip ) echo "Copied image_only_dm_image.mtz to build.mtz"
endif
 cp image_only_dm_image.mtz_${cycle} refine.mtz
 if ( $ip ) echo "Copied image_only_dm_image.mtz to refine.mtz"
#
#
if (  -f resolve.built ) then
    set built = `cat resolve.built|head -1`
    set placed = `cat resolve.built | head -2|tail -1`
    set build_score = `cat resolve.built | head -4|tail -1`
    cp resolve.pdb built_model.pdb
else
 set build_score = 0
 set built = 0
 set placed = 0
 echo "REMARK No model built" > work_model.pdb
 if ( -f built_model.pdb ) rm -f built_model.pdb
 echo "REMARK No model built"
endif
set build_score_best = $build_score
set built_best = $built
set placed_best = $placed
 if ( $ip ) echo "Score : $build_score   built: $built  placed: $placed"
#
#=================================================================
#
```

```
#  Start of refine-extend cycles
#
if  ( $build_outside_model == YES && $pdb_in != NONE )then
set add_mask_use = 'add_mask'
set model_in = 'model work_model_start.pdb'
set build_outside_model_use = 'build_outside_model'
else
set add_mask_use = ''
set model_in = ''
set build_outside_model_use = ''
endif
foreach extend_cycle ( $extend_flag )
#
if ( $ip ) echo "Extend cycle $extend_cycle"
if( -f built_model.pdb && $extend_cycle > 0 ) then
 if ( $ip ) echo "Extending model"
#
if ( -f resolve.built ) rm -f resolve.built
${resolve}<<EOD >> resolve.log_$cycle
hklin build.mtz
labin FP=FP PHIB=PHIM FOM=FOMM
hklout dummy.mtz
solvent_content $solvent_content
resolution ${dmin} ${dmax}
extend_only
$add_mask_use
$model_in     ! building outside work_model_start only
$build_outside_model_use    ! and map to the region of this model too
ha_file $ha_file
seq_file $seq_file
compare_file $compare_file
pdb_in built_model.pdb    ! extend what we just built (only)
no_trim
nohl
$mask_line
$use_mask_line
EOD
endif
if ( ( $extend_cycle > 0 ) && ( -f resolve.built ) ) then
    set built = `cat resolve.built|head -1`
    set placed = `cat resolve.built | head -2|tail -1`
    set build_score = `cat resolve.built | head -4|tail -1`
 if ( $ip ) echo "Score for extension: $build_score   built: $built  placed: $placed"
else if ( $extend_cycle > 0 ) then
 set build_score = 0
 set built = 0
 set placed = 0
endif
if ( ( $extend_cycle > 0 ) && ( $build_score > $build_score_best ) ) then
```

```
 set build_score_best = $build_score
 set built_best = $built
 set placed_best = $placed
 cp resolve.pdb built_model.pdb
 cp work_model_start.pdb work_model.pdb
 $grep_type ATOM built_model.pdb >>work_model.pdb
 if ( $ip ) echo "Adding extended model to starting work model"
else if ( $extend_cycle > 0 ) then
 if ( $ip ) echo "Extension did not improve score...going on"
#  break to end of the extend cycles...
 break
endif
#
#
#  decide if we refine model this cycle..
#
if ( -f resolve.coverage ) then
 set coverage1 = `cat resolve.coverage|head -3|tail -1`
 set coverage = 0
 if ( $#coverage1> 0 ) set coverage= $coverage1[$#coverage1-$#coverage1]:e
else
 set coverage = 0
endif
if ( $refine_model == NO ) then
 set refine_it = NO
else if ( $build_score_best < 2 ) then
 set refine_it = NO
else if ( ( $coverage < $minimum_coverage ) && ( $build_outside_model == NO) )then
 set refine_it = NO
else if ( ! -f work_model.pdb ) then
 echo "REMARK No model built" > work_model.pdb
 echo "REMARK No model built extend cycle $extend_cycle"
 set refine_it = NO
else
 set refine_it = YES
endif
if ( $refine_it == YES ) then
 set ncyc = 20
else
 set ncyc = 0
endif
if ( ( $refmac5 != NONE ) && ( $build_score_best > 1 ) ) then
if ( $ip ) echo "Refining model $ncyc cycles and placing it in refine.pdb_$cycle"
#
cat work_model.pdb | $grep_type -v HETATM > resolve.pdb
#
# --------refine with refine.--------------
#
#  NOTE: refine against "best" map = refine.mtz
```

```
#
cp resolve.pdb refine.pdb   # in case it doesn't work...
#
$refmac5 xyzin resolve.pdb hklin refine.mtz xyzout refine.pdb  \
PROTOUT $CCP4_SCR/protout.dat \
PROTCOUNTS $CCP4_SCR/counts.dat \
PROTSCR $CCP4_SCR/counts.scr \
<<EOD > refmac.log_${cycle}
MAKE HYDRogens No
MAKE CHECK 0
LABI FP=FP SIGFP=SIGFP HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM FREE=FreeR_flag
REFI TYPE RESTrained RESOlution ${dmax} ${dmin}
#   Maximum likelihood refinement
REFI RESI MLKF
#   Overall B value refinement
REFI BREF OVER      ! Refine overall B-values
#   Tight restraints  Lowered further 051403
WEIG MATR 0.05
#   Scale down shifts at every cycle by factor 0.5
DAMP 0.5 0.5
#   No Blur (scale down reliability of) phases
REFI PHASed  BBLUrring 00.0  SCBLurring 1.
#  Babinet's bulk solvent scale parameters. But fix B value
SCALe TYPE BULK
SCALe LSSCale FIXBulk BBULk 200.0
#   Anisotropic scaling. It is default
SCALe LSSCale ANISotropic
#   Fix Babinet's bulkd solvent for sigmaA
SCAL MLSC FIXBulk BBULk 200.0 SCBUlk -0.05
NCYC $ncyc
MONI MEDI
BINS 10
EOD
#-----------------------------------------------------------
#
cp refine.pdb refine.pdb_$cycle
else if (  $build_score_best > 1  ) then
#   no refmac5...just copy over
 cat work_model.pdb | $grep_type -v HETATM > resolve.pdb
 cp resolve.pdb refine.pdb
 cp refine.pdb refine.pdb_$cycle
else
 if ( $ip ) echo "No model built this try"
 echo "REMARK  No model built cycle $cycle " > refine.pdb_$cycle
 echo "REMARK  No model built cycle $cycle "
endif
#
end_of_extend:
end
```

```
#
#  Get R and freeR value on refine.pdb_$cycle:
set freer_line = `$grep_type 'FREE R VALUE                     :' refine.pdb_$cycle|tail -1`
set r_line = `$grep_type 'R VALUE            (WORKING SET) :' refine.pdb_$cycle|tail -1`
#
if ( $#freer_line > 0 ) then
 set freer = $freer_line[$#freer_line-$#freer_line]:e
else
 set freer = 99999
endif
if ( $#r_line > 0 ) then
 set r = $r_line[$#r_line-$#r_line]:e
else
 set r = 99999
endif
if ( $ncyc == 0 )then
 set freer = $r
endif
#
if ( $ip ) echo " R/freeR for this model: 0.$r/0.$freer"
#
if ( $cycle < 10 ) then
 set b1 = " "
else
 set b1 = ""
endif
if ( $built_best < 100 ) then
 set b2 = " "
else
 set b2 = ""
endif
if ( $placed_best >= 100 ) then
 set b3 = ""
else if ( $placed_best >= 10 )then
 set b3 = " "
else
 set b3 = "  "
endif
if ( $build_score_best < 100 )then
 set b4 = " "
else
 set b4 = ""
endif
# Decide how to score. If R < unacceptable_r or r_best_overall < unacceptable_r
#   then use R. Otherwise use score
#
if ( ( $r < $n_unacceptable_r ) && ( $r < $r_best_overall ) ) then
 set this_is_best = YES
else if ( ( $r_best_overall >= $n_unacceptable_r ) && \
  ( $r <= $n_r_max ) && \
  ( $build_score_best > $build_score_best_overall ) ) then
 set this_is_best = YES
else
 set this_is_best = NO
endif
if ( $this_is_best == YES ) then
 if ( $ip ) echo "This is the best model overall so far. Previous best had score of $build_score_best_overall"
 if ( $ip ) echo "Copied refine.pdb_$cycle to resolve_best.pdb"
 if ( $ip ) echo "Copied resolve.log_$cycle resolve_best.log"
 if ( $ip ) echo "Copied refine.mtz to resolve_best.mtz"
 @ build_score_best_overall = $build_score_best
 @ build_score_best_overall_cycle = $cycle
 @ r_best_overall = $r
 cp refine.pdb_$cycle resolve_best.pdb
 cp resolve.log_$cycle resolve_best.log
 cp refine.mtz resolve_best.mtz
 if( -f refmac.log_$cycle ) cp refmac.log_$cycle refmac_best.log
 echo "  ${b1}$cycle       ${b2}$built_best        ${b3}$placed_best         ${b4}$build_score_best     0.$r    0.$freer    *"
else
 echo "  ${b1}$cycle       ${b2}$built_best        ${b3}$placed_best         ${b4}$build_score_best     0.$r    0.$freer    "
endif
#
#=================================================================
#
if ( $reuse_chain == YES )then
 cat  refine.pdb > current_model.pdb
 if ( $ip ) echo " Using refine.pdb fragments as part of next build cycle"
else if ( -f current_model.pdb ) then
  rm -f current_model.pdb
endif
#
#
if ( $ip ) echo " "
if ( $ip ) echo "End of cycle $cycle with score of $build_score_best"
if ( $ip ) echo " "
#
#  Decide if we should keep going or quit..
@ cutoff_cycle = $cycle - $past_cycle
if ( ( $build_score_best_overall_cycle <= $cutoff_cycle ) && ( $r_best_overall < $n_unacceptable_r ) ) then
 echo "Our best model was built on cycle $build_score_best_overall_cycle which was"
 echo "   more than $past_cycle cycles ago...calling it quits with this model,"
 echo "   now in resolve_best.pdb"
 break
endif
#
#  clean up...
#
if ( $clean_up == YES  ) then
```

```
 if ( -f resolve.log_$cycle ) rm -f resolve.log_$cycle
 if ( -f refmac.log_$cycle ) rm -f refmac.log_$cycle
 if ( -f resolve.mtz_$cycle ) rm -f resolve.mtz_$cycle
 if ( -f refine.mtz ) rm -f refine.mtz
 if ( -f image_only_dm_image.mtz_$cycle ) rm -f image_only_dm_image.mtz_$cycle
endif
#
if ( is_omit_cycle == YES ) then
#  set this cycle as first one to consider when creating composite from PDB files
 set cycle_pdb_start = $cycle
endif
#
@ cycle ++
#
#   end of build cycle
#
end
#
wrap_up:
if ($evaluate_only == NEVER)then
  exit
endif
#
#============================================================================
#============================================================================
echo " "
echo "Analyzing model based on prime-and-switch composite omit map"
#
echo "First rebuilding current model to remove refinement bias..."
if ( ! -f resolve_best.pdb ) then
 echo "Sorry cannot find resolve_best.pdb?"
 exit
else
 cp resolve_best.pdb test.pdb_0
endif
if ( $skip == YES ) then
 goto evaluate_it
endif
#
${resolve}<<EOD > resolve_eval.log_1
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
resolution ${dmin} ${dmax}
composite_pdb test.pdb_
composite_pdb_start 0
composite_pdb_end 0
composite_all
dump_composite
```

```
EOD
#
${resolve}<<EOD >> resolve_eval.log_1
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
resolution ${dmin} ${dmax}
no_build
pattern_phase
prior_weight 1.0
hklout image_only.mtz
read_cc_map
cc_map_file dump.map
nohl
EOD
#
if ( $ip ) echo "Density-modifying image of resolve_best.pdb to get image_only_dm.mtz"
#
${resolve}<<EOD >> resolve_eval.log_1
hklin image_only.mtz
labin FP=FP PHIB=PHIM FOM=FOMM  HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM
hklout image_only_dm.mtz
solvent_content $solvent_content
prior_weight 1.0
database 5
resolution ${dmin} ${dmax}
no_build
nohl
EOD
#
if ( $ip ) echo "Carrying out phasing using image_only_dm.mtz "
if ( $ip ) echo " and resolve_best.pdb and rebuilding model"
#
if ( -f resolve.built ) rm -f resolve.built
${resolve}<<EOD >> resolve_eval.log_1
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
hklstart image_only_dm.mtz
labstart FP=FP PHIB=PHIM FOM=FOMM
hklout image_only_dm_image.mtz
solvent_content $solvent_content
database 5
resolution ${dmin} ${dmax}
composite_pdb test.pdb_
composite_pdb_start 0
composite_pdb_end 0
composite_all
prior_weight 1.0
```

```
scale_refl $scale
image
mask_cycles 0
$superquick_build
seq_file $seq_file
compare_file $compare_file
ha_file $ha_file
hklout test.mtz_1
nohl
$mask_line
$use_mask_line
EOD
if ( -f resolve.pdb ) then
 cp resolve.pdb test.pdb_1
else
 echo "Sorry, no model obtained on rebuilding...quitting"
 exit
endif
if (  -f resolve.built  ) then
    set built = `cat resolve.built|head -1`
    set placed = `cat resolve.built | head -2|tail -1`
    set build_score = `cat resolve.built | head -4|tail -1`
else
 set build_score = 0
 set built = 0
 set placed = 0
 echo "Sorry, no model obtained on rebuilding...quitting"
 exit
endif
#
#  report what happened in our rebuilding...
echo "Rebuilt model...$built residues built, $placed with side-chains in test.pdb_1"
#
echo "Generating composite prime-and-switch composite omit map based on rebuilt model"
#
${resolve}<<EOD >> resolve_eval.log_1
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
resolution ${dmin} ${dmax}
composite_pdb test.pdb_
composite_pdb_start 1
composite_pdb_end 1
composite_all
dump_composite
EOD
#
${resolve}<<EOD >> resolve_eval.log_1
hklin $hklin_exp
```

```
labin $labin_exp
labin $labin_exp_cont
resolution ${dmin} ${dmax}
no_build
pattern_phase
prior_weight 1.0
hklout image_only.mtz
read_cc_map
cc_map_file dump.map
nohl
EOD
#
if ( $ip ) echo "Density-modifying image of test.pdb_1 to get image_only_dm.mtz"
#
${resolve}<<EOD >> resolve_eval.log_1
hklin image_only.mtz
labin FP=FP PHIB=PHIM FOM=FOMM  HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM
hklout image_only_dm.mtz
solvent_content $solvent_content
prior_weight 1.0
database 5
resolution ${dmin} ${dmax}
no_build
nohl
$mask_line
$use_mask_line
EOD
#
if ( $ip ) echo "Getting composite ps omit map using image_only_dm.mtz "
if ( $ip ) echo " and test.pdb_1"
#
${resolve}<<EOD >> resolve_eval.log_1
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
hklstart image_only_dm.mtz
labstart FP=FP PHIB=PHIM FOM=FOMM
hklout image_only_dm_image.mtz
solvent_content $solvent_content
database 5
resolution ${dmin} ${dmax}
composite_pdb test.pdb_
composite_pdb_start 1
composite_pdb_end 1
composite_all
prior_weight 1.0
scale_refl $scale
image
omit
```

```
mask_cycles 0
no_build
nohl
$mask_line
$use_mask_line
EOD
cp resolve_composite_map.mtz build.mtz
if ( $ip ) echo "Composite omit is in build.mtz"
#
evaluate_it:
#
#  Get R-factor for the model resolve_best.pdb...relative to hklin_exp
#
#
$refmac5 xyzin resolve_best.pdb hklin $hklin_exp xyzout refine.pdb  \
PROTOUT $CCP4_SCR/protout.dat \
PROTCOUNTS $CCP4_SCR/counts.dat \
PROTSCR $CCP4_SCR/counts.scr \
<<EOD > refmac_eval.log_${cycle}
MAKE HYDRogens No
MAKE CHECK 0
LABI FP=FP SIGFP=SIGFP FREE=FreeR_flag
REFI TYPE RESTrained RESOlution ${dmax} ${dmin}
#   Maximum likelihood refinement
REFI RESI MLKF
#   Overall B value refinement
REFI BREF OVER      ! Refine overall B-values
#   Tight restraints  Lowered further 051403
WEIG MATR 0.05
#   Scale down shifts at every cycle by factor 0.5
DAMP 0.5 0.5
#   No Blur (scale down reliability of) phases
REFI PHASed  BBLUrring 00.0  SCBLurring 1.
#  Babinet's bulk solvent scale parameters. But fix B value
SCALe TYPE BULK
SCALe LSSCale FIXBulk BBULk 200.0
#   Anisotropic scaling. It is default
SCALe LSSCale ANISotropic
#   Fix Babinet's bulkd solvent for sigmaA
SCAL MLSC FIXBulk BBULk 200.0 SCBUlk -0.05
NCYC 0
MONI MEDI
BINS 10
EOD
#--------------------------------------------------------
#
#  Get R and freeR value on refine.pdb:
set freer_line = `$grep_type 'FREE R VALUE                    :' refine.pdb|tail -1`
set r_line = `$grep_type 'R VALUE            (WORKING SET) :' refine.pdb|tail -1`
```

```
#
if ( $#freer_line > 0 ) then
 set freer = $freer_line[$#freer_line-$#freer_line]:e
else
 set freer = 99999
endif
if ( $#r_line > 0 ) then
 set r = $r_line[$#r_line-$#r_line]:e
else
 set r = 99999
endif
#
echo " "
echo "=========================================================================="
if ( $evaluate_only == YES ) then
 echo "Work R-factor for input model $pdb_in : 0.$r   free R: 0.$freer"
 echo "Evaluation of $pdb_in using prime-and-switch composite omit map:"
else
 echo "Work R-factor for resolve_best.pdb: 0.$r   free R: 0.$freer"
 echo "Evaluation using prime-and-switch composite omit map:"
endif
#
#
#   Evaluate the model relative to the composite omit ps map:
#
${resolve} <<EOD > evaluate.log
hklin build.mtz
labin FP=FP PHIB=PHIM FOM=FOMM
model resolve_best.pdb
evaluate_model
EOD
#
#==========================================================================
echo "See evaluate.log for more details on the analysis of this model"
echo " "
echo "All done...please carefully evaluate your model."
echo "Your best indicators are the free R factor and the match"
echo "to the prime-and-switch composite omit map in evaluate.log."
echo "Note that no waters are built in this model."
echo " "
echo "Output files are:"
if ( -f resolve_best.pdb ) then
 echo "resolve_best.pdb  ... partially-refined model"
endif
if ( -f resolve_best.log ) then
 echo "resolve_best.log  ... log file in which this model was built"
endif
if ( -f resolve_best.mtz ) then
$resolve<<EOD>final.log
```

```
hklin resolve_best.mtz
labin FP=FP PHIB=PHIM FOM=FOMM
mask_cycles 1
minor_cycles 0
no_build
ccp4_map_file resolve_best.map
EOD
 echo "resolve_best.mtz  ... FP, SIGFP, best overall phases, freeR_flag...."
 echo "resolve_best.map  ... ccp4-style map from resolve_best.mtz"
endif
if ( -f refmac_best.log ) then
 echo "refmac_best.log   ... log file for refinement of resolve_best.pdb"
endif
if ( -f build.mtz ) then
$resolve <<EOD>> final.log
hklin build.mtz
labin FP=FP PHIB=PHIM FOM=FOMM
mask_cycles 1
minor_cycles 0
no_build
ccp4_map_file resolve_composite_omit.map
EOD
#
 echo "resolve_composite_omit.map ... composite omit map "
endif
#=============================================================
```

Sample scripts for SOLVE

Choose one of the following sample control files, edit it to match your data (you may need to check on data formats), and run SOLVE with it. You can watch SOLVE run by looking at the end of the "solve.status" with "tail -30f solve.status". This file will tell you where to look if something got typed in wrong and it will keep you informed about the structure determination as it goes.

If you want to change some parameters that are not listed in this file you should have a look at the list of keywords for SOLVE automated operation.

Each control file also comes with a script you can use to generate a dummy dataset and an output solve.prt file that will be similar to the one you should get if you run the control file on the dummy dataset. (Note: it won't necessarily be identical, because slight differences between machines can lead to somewhat different output files. The overall answers are usually about the same, though. The files here were generated on a DEC alpha.) You can use these to make sure everything is working properly.

Note in particular the keywords

- READDENZO/READFORMATTED/READTREK
- UNMERGED/PREMERGED
- READ_INTENSITIES/READ_AMPLITUDES

  LABIN / HKLIN (for CCP4 files, see example ccp4 MADor MIRscripts)

ADDSOLVE and ANALYZE_SOLVE. Note also if you know some of your sites at the beginning you can tell SOLVE to use them:

- under "lambda 1" or "derivative 1" specify
  - atomname xx   ! xx is your atom
  - xyz x y z   ! coordinates of this atom
  - .. more xyz values
- before ANALYZE_MAD or ANALYZE_MIR or SAD specify:
  - ADDSOLVE   ! ADDSOLVE (add on more sites) or ANALYZE_SOLVE (refine input sites)
- SOLVE will refine these sites and go on from there.
- See sample solve.com file for ADDSOLVE.
- See sample solve.com file for ANALYZE_SOLVE.

Single datasets

Command files for running SOLVE with...

- 1 SAD dataset: generate one or solve one or look at the solve.prt output from generate followed by solve.
- 1 MAD dataset: generate one or solve one or look at the solve.prt output from generate followed by solve.
- 1 MIR dataset: generate one or solve one or look at the solve.prt output from generate followed by solve.
- 1 MIR dataset with anomalously scatter ing atoms in the native: generate one or solve one (requires version 1.10 or higher) or look at the solve.prt output from generate followed by solve.

Multiple datasets

If you have several types of datasets on the same crystal, then you can combine them into one pseudo-MIR dataset (see also the documentation on COMBINE). You can also analyze MAD data with more than one type of anomalous scatterer with this method. Edit these control files for combining datasets (Note: all these require version 1.10 or higher):

- MAD + MIR datasets: generate one or solve one or look at the solve.prt output from generate followed by solve.
- 2 MAD datasets: generate one or solve one or look at the solve.prt output from generate followed by solve.
- 2 MIR datasets: generate one or solve one or look at the solve.prt output from generate followed by solve.
- 1 MAD dataset with 2 anomalously scattering atoms: generate one or solve one or look at the solve.prt output from generate followed by solve.

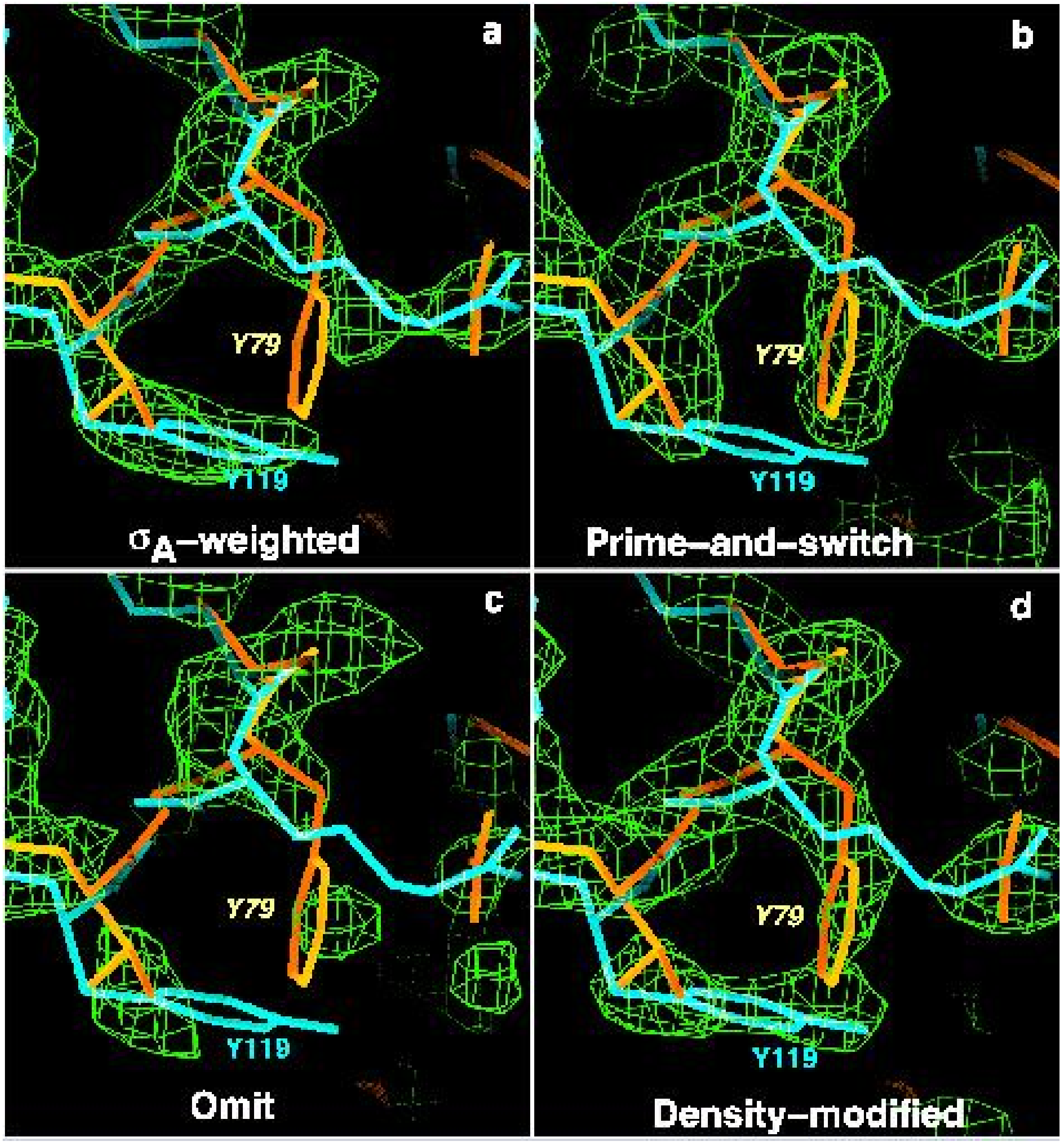
SOLVE/RESOLVE home page	***Back to RESOLVE table of contents***

RESOLVE examples

Removal of model bias with prime-and-switch phasing

IF5A molecular replacement phasing

- **The system: initiation factor 5A (IF5A)**
- **X-ray data on *P. aerophilum* IF5A to 2.2 A** *(Peat T.S., Newman J, Waldo G.S, Berendzen J. & Terwilliger T.C.   Structure Of Translation Initiation-Factor 5a From Pyrobaculum aerophilum At 1.75 Angstrom Resolution. Structure 6, 1207-1214 (1998))*
- **60% solvent**
- **atomic model from *M. jannaschii* IF5A (***Kim, K. K, Hung, L. W., Yokota, H., Kim, R. & Kim, S.-H. Crystal structures of eukaryotic translation initiation factor 5A from Methanococcus jannaschii at 1.8 angstrom resolution. Proc. Natl. Acad Sci. USA 95, 10419-10424 (1998))*
- **RMSD between atomic model from *M. jannaschii* and refined *P. aerophilum* structure is 1.7 A.**
- **The test: phase the *P. aerophilum* data with the *M. jannaschii* structure**
- **Methods tested: (a) sigma-A, (b) prime-and-switch, (c) omit map, and (d) density modification (with RESOLVE)**
- **Results: (MR model in blue, correct model in yellow). Sigma-A map looks a lot like the model in blue used to calculate it; prime-and-switch map looks just like the correct model in yellow. Omit map is noisy, and density-modified map is halfway between sigmaA and prime-and-switch.**

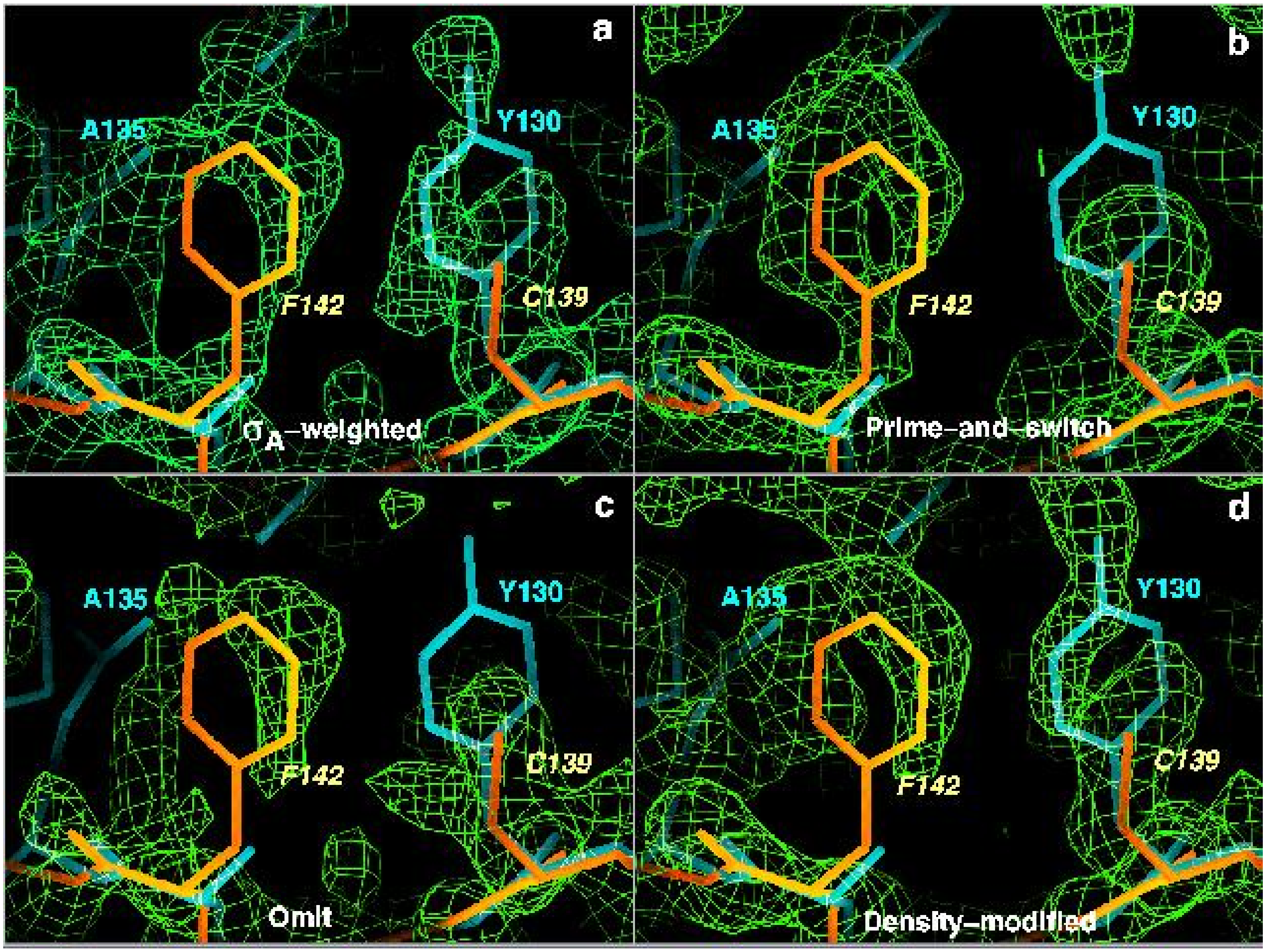


Removal of model bias with prime-and-switch phasing

Dehalogenase molecular replacement phasing

- **The system: *Rhodococcus* dehalogenase (*dhlA*)**
- **X-ray data on *Rhodococcus dhlA* to 1.5 A** *(Newman, J., Peat, T. S., Richard, R., Kan, L., Swanson, P. E., Affholter, J. A., Holmes, I. H., Schindler, J. F., Unkefer, C. J., & Terwilliger, T. C. Haloalkane dehalogenases: Structure of a Rhodococcus enzyme. Biochemistry 38, 16105-16114 (1999).)*
- **30% solvent**
- **atomic model from dehalogenase *linB* from *S. paucimobilis***(*Marek, J., Vevodova, J., Smatanova, I. K., Nagata, Y., Svensson, L. A., Newman, J., Takagi, M. & Damborsky, J. Crystal structure of the haloalkane dehalogenase from Sphingomonas paucimobilis UT26. Biochemistry 39, 14082-14086 (2000).)*

- **RMSD between atomic model from *S. paucimobilis lin B* and refined *RhodococcusdhlA* structure is 1.4 A.**
- **The test: phase the *Rhodococcus dhlA* data with the *S. paucimobilis linB* structure**
- **Methods tested: (a) sigma-A, (b) prime-and-switch, (c) omit map, and (d) density modification (with RESOLVE)**
- **Results: (MR model structure is in blue, correct model in yellow). SigmaA map looks like the MR model used to calculate it; prime-and-switch map looks like the correct model.**

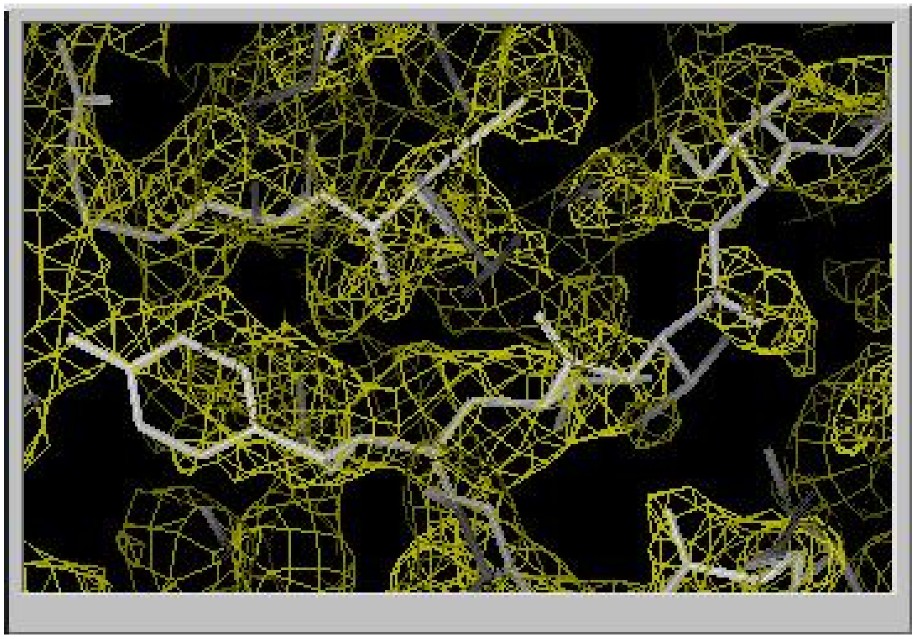


Improvement of experimentally-phased maps

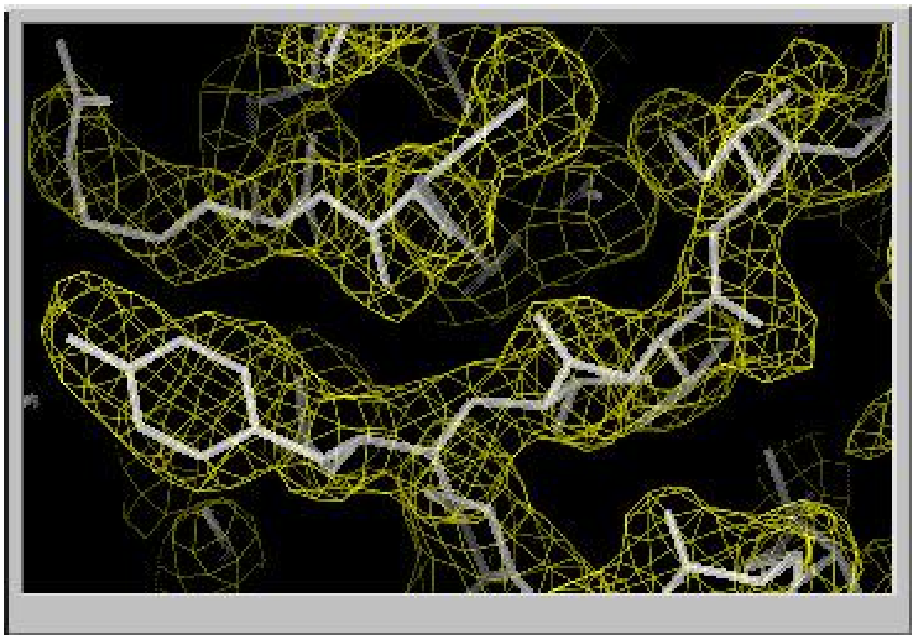
IF5A MAD map

(See: Peat, T. S., Newman, J., Waldo, G. S., Berendzen, J. & Terwilliger, T. C. (1998) "Structure of translation initiation factor 5A from Pyrobaculum aerophilum at 1.75 A resolution" Structure 15, 1207-1214.

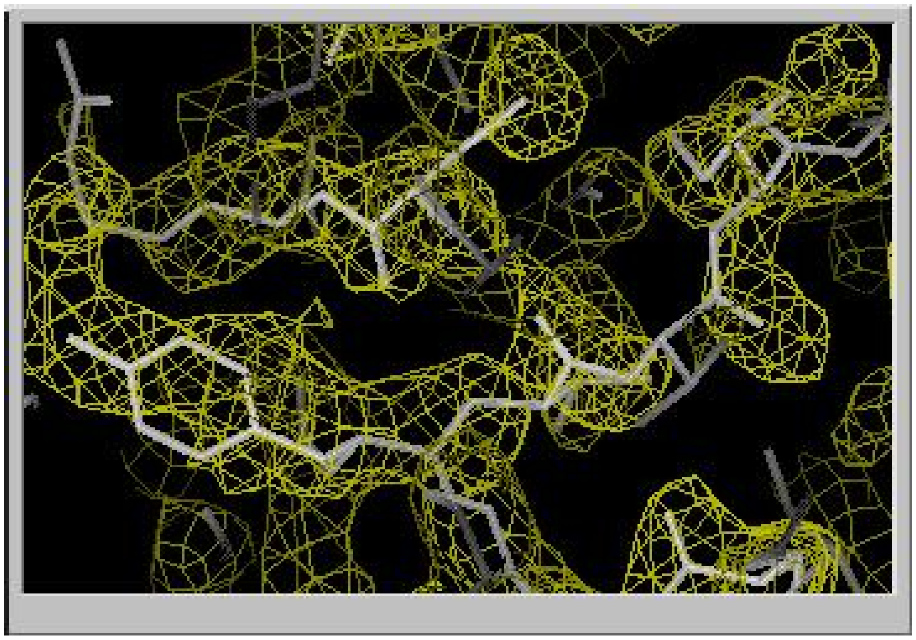
- **MAD dataset on initiation factor 5A**
- **Only 1 of 3 selenium atoms in 147 residues used in phasing (just for this test example)**
- **Solvent content was 60%**
- **Resolution of 2 A, space group I4**
- **Initial correlation of experimental map with one based on refined model was 0.37**
- **Correlation after RESOLVE was 0.79**
- **Correlation after conventional solvent flattening and histogram matching (using dm) was 0.65**



Initial MAD map (CC=0.37)



RESOLVE map (CC=0.79)



Conventional solvent flattening & histogram matching (CC=0.65)

SOLVE / RESOLVE home page	RESOLVE Table of Contents	SOLVE Alphabetical Index

Table of Contents for SOLVE on-line manual version 2.09

SOLVE Examples

This page gives three examples of what SOLVE can do. Each is an automatic structure determination carried out starting with raw data and a minimal amount of information from the user. In each case the top solution was correct and had the correct handedness (anomalous differences were used in each structure determination). The examples are:

- Gene 5 protein (MAD data, 2800 reflections to 2.6 A, 87 amino acids, 2 selenium sites)
- Armadillo repeat of beta-catenin (MAD data, 17000 reflections to 2.7 A, 537 amino acids, 15 selenium sites; data courtesy of Andy Huber and Bill Weis)
- Granulocyte-macrophage colony stimulating factor (MIR data, 4800 reflections to 3.5 A, 4 derivatives, 254 amino acids; data courtesy of Kay Diederichs)

For each structure determination, this page shows:

- A description of the structure determination by SOLVE
- The input script file used to run SOLVE
- The summary of the structure solution from the output "solve.prt"
- The end of the solve.status file that showed the progress of the structure solution file

Gene 5 protein

Summary of this structure solution:

This is a dataset with 3-wavelengths of MAD data, 2800 reflections to 2.6 A, 87 amino acids, and 2 selenium sites (Met-1, Met-77). SOLVE found both selenium sites in 34 minutes on an DEC alpha 500 MHz workstation. The Met-1 site has a very high thermal factor.

Solve.setup file listing basic information about the crystals:

```
CELL 76.08 27.97 42.36 90 103.2 90        ! cell params
symfile /usr/local/lib/solve/c2.sym       ! space group symmetry
resolution 2.6 20.0                       ! resolution limits
```

Input script file used to run SOLVE on gene 5 protein

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
```

```
#
solve <<EOD > solve.log
!command file to read in raw MAD data, scale, analyze and solve it----
checksolve                      ! compare solution with known h.a. sites
comparisonfile gvp.fft          ! FFT map using FCALC from refined model
@solve.setup                    ! get our standard information read in
logfile mad.logfile             ! write out most information to this file.
                                ! summary info will be written to solve.prt
readformatted                   ! alternatives are readdenzo, readtrek
premerged                       ! alternative is unmerged
read_intensities                ! alternative is read_amplitudes
refscattfactors                 ! alternative is fixscattfactors


mad_atomname se                 ! anomalously scattering atom is Se

lambda 1                       ! info on wavelength #1 follows
label Wavelength #  1          ! a label for this wavelength
rawmadfile test_wva.fmt        ! datafile with h k l Intensity sigma or
                               ! h k l I+ sigma+ I- sigma-
wavelength 0.9000              ! wavelength value
fprimv_mad  -1.6               ! f' value at this wavelength
fprprv_mad  3.4                ! f doubleprime value at this wavelength

! input refined h.a. coordinates (used only for comparison in "checksolve")
atomname se
 XYZ   0.4813319      0.9972169      9.4140753E-02
 XYZ   0.9731338      0.2875228      0.9446641

lambda 2
rawmadfile test_wvb.fmt
wavelength 0.9794
fprimv_mad  -8.5
fprprv_mad  4.8

lambda 3
rawmadfile test_wvc.fmt
wavelength 0.9797
fprimv_mad  -9.85
fprprv_mad  2.86
premerged
readformatted
nres 100                   [approx # of residues in protein molecule]
nanomalous 2               [approx # of anomalously scattering atoms per protein]
SCALE_MAD                  ! read in and localscale the data
ANALYZE_MAD                ! run MADMRG and MADBST and analyze all the Pattersons
SOLVE                      ! Solve the structure
EOD
```

Summary information from the "solve.prt" output file produced after completion of the automated structure determination

Correlation of anomalous differences. These indicate that the data beyond about 2.7 A are not contributing much to the phasing, as the correlation is less than 0.3.

```
             CORRELATION FOR

            WAVELENGTH PAIRS

 DMIN    1 VS 2   1 VS 3   2 VS 3

 5.20     0.90     0.72     0.80

 3.90     0.76     0.54     0.69

 3.64     0.69     0.44     0.69

 3.44     0.65     0.44     0.52

 3.25     0.59     0.27     0.50

 3.12     0.57     0.34     0.47

 2.99     0.58     0.18     0.39

 2.86     0.42     0.30     0.46

 2.73     0.33     0.17     0.31

 2.60     0.19     0.09     0.32
```

List of sites analyzed for compatibility with difference Patterson

```
   PEAK         X         Y         Z     OPTIMIZED
                                             RELATIVE OCCUPANCY
      1     0.484     1.000     0.090      91.024
      2     0.026     0.292     0.062      37.211

 Evaluation of this test soln with    2 sites after optimizing
 occupancy of each site

 Cross-vectors for sites  1 and  1 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1   -0.969   0.000  -0.181   16389.9     16570.9          2

 Cross-vectors for sites  2 and  1 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1   -0.458  -0.708  -0.028   4053.98     3387.13          1
   2   -0.510  -0.708  -0.153   4583.90     3387.13          1

 Cross-vectors for sites  2 and  2 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1   -0.052   0.000  -0.125   1640.05     2769.35          2

 Overall quality of this Patterson soln =  7815.40
 Overall quality of the fit to patterson = 0.798126
 Avg normalized peak height =  3495.15
```

```
Selenium atom occupancy, coordinates, and thermal factors, and
Cross-validation fouriers calculated with all heavy atoms in
all derivs except the site being evaluated and any sites equivalent to it.

(Peak height is height of peak at this position/rms of map)

  Site    x       y       z       occ       B     -- PEAK  HEIGHT --

    1   0.484   0.997   0.093   0.526  49.596             4.90
    2   0.028   0.285   0.059   0.369  60.000             3.56
```

Re-refinement of f' and f" values:

```
                    Final refined values of f-prime and f"

    Wavelength  ------- f-prime --------       --------f"-------------
        last refinement       Refined     last refinement       Refined

     1          0.327          0.327             3.817          2.928

     2         -8.419         -7.357             6.146          5.860

     3         -9.609         -8.668             2.272          1.774
```

Figure of merit versus resolution, and anomalous and dispersive FH/E versus resolution

```
 FIGURE OF MERIT WITH RESOLUTION
 DMIN:            TOTAL    8.81   5.75   4.55   3.88   3.44   3.12   2.88   2.68
 N:                2668     153    234    301    343    388    417    452    380
 MEAN FIG MERIT:   0.46    0.65   0.64   0.62   0.56   0.52   0.37   0.34   0.26

 RMS ANOMALOUS FH/E  [f" PART OF FH / RMS ANO ERROR]:

 LAMBDA:  1          0.5    0.7    0.8    0.6    0.6    0.5    0.3    0.3    0.2
 LAMBDA:  2          0.6    0.8    0.8    0.7    0.7    0.6    0.5    0.4    0.3
 LAMBDA:  3          0.3    0.6    0.6    0.5    0.3    0.3    0.2    0.2    0.1

 RMS DISPERSIVE FH/E  [Delta-f-prime PART OF FH / RMS DISPERSIVE ERROR]:

 L1 VS L2:           0.8    1.0    1.1    1.0    0.9    0.8    0.5    0.4    0.3
 L1 VS L3:           0.9    1.2    1.3    1.0    1.0    0.9    0.6    0.5    0.4
 L2 VS L3:           0.2    0.4    0.4    0.3    0.2    0.2    0.1    0.1    0.1
```

The summary of scoring for this solution

```
 Summary of scoring for this solution:
                           -- over many solutions--    -- this solution --
 Criteria                       MEAN          SD          VALUE        Z-SCORE
 Pattersons:                   3.11          1.86         5.83          1.46
 Cross-validation Fourier:     3.26          2.44         7.05          1.55
 NatFourier CCx100:            27.4          5.39         31.3         0.729
 Mean figure of meritx100:    0.000E+00      8.26         58.5          7.08
 Correction for Z-scores:                                              -2.85

 Overall Z-score value:                                                 7.97
```

Note that the Patterson and cross-validation Fourier scores are 1.5 sigma above the starting solutions, but native fourier analysis is just 1 sigma above. This is both because the asymmetric unit is small and the map is fairly noisy.

The end of the solve.status file:

```
****************************************************************************
                    SOLVE STATUS      07-oct-00 10:34:53

 TIME ELAPSED:    34 MIN

 ----------------------------------------------------------------------------
 CURRENT STEP:SOLVE MAIN PROGRAM
 STATUS:   DONE
 ----------------------------------------------------------------------------
 ----------------------------------------------------------------------------
     ---TOP SOLUTION FOUND BY SOLVE  (<m> = 0.59; score =   8.03) ---

           X        Y        Z         OCCUP     B          HEIGHT/SIGMA

   2     0.484    0.997    0.093     0.526     49.6              4.3
   2     0.028    0.285    0.059     0.369     60.0              4.4

        TIME REQUIRED TO OBTAIN THIS SOLUTION:    28 MIN
 ----------------------------------------------------------------------------
 CURRENT RESOLUTION:   2.6 A.    FINAL RESOLUTION:   2.6 A.
```

Armadillo repeat region of beta-catenin (data courtesy of Andy Huber and Bill Weis)

Summary of this structure solution:

This is a dataset with 4 wavelengths of MAD data, 17000 reflections to 2.7 A, 537 amino acids, and 15 selenium sites. SOLVE found 14 selenium sites in 2 hours on a DEC Alpha 500 MHz workstation. The remaining 2 sites (one selenium has 2 positions) are very weak and were not included by SOLVE.

Solve.setup file listing basic information about the beta-catenin crystals:

```
resolution 2.7 20
symfile /usr/local/lib/solve/c2221.sym
cell 64.1 102.0 187.0 90 90 90
```

***Input script* file used to run SOLVE on beta-catenin**

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#
solve <<EOD > solve.log

!command file to read in raw MAD data, scale, analyze and solve it----
title armadillo repeat of beta catenin 4-wavelength MAD data
logfile mad.logfile              ! write out most information to this file.
                                 ! summary info will be written to "solve.prt"
@solve.setup                     ! get our standard information read in
readformatted                     ! or: readdenzo, readtrek, readccp4_unmerged
unmerged                          ! or; premerged

mad_atom se

refscattfactors                 ! do not refine scattering factors (you can if
                                ! you want though)

        !  Comment out next line if you don't know any sites
checksolve                      ! compare solutions to the one input below

        !  Comment out next lines if you don't know the structure
! native.fft is fft calculated from catenin_y.pdb (offset +0.5 in y)
comparisonfile native.fft

lambda 1                        ! info on wavelength #1 follows
label Wavelength #  1           ! a label for this wavelength
```

```
rawmadfile l1.int
wavelength 0.9000              ! wavelength value
fprimv_mad  -1.6               ! f' value at this wavelength
fprprv_mad  3.4                ! f" value at this wavelength

! list of all SE positions in refined beta-catenin
! structure (offset by 0.5 in y from PDB file). Only if you know them

atomname se
xyz  0.2631041      0.6633824      2.8978506E-02
xyz  0.4166300      0.6113137      7.7325497E-03
xyz  0.4765674      0.7249608      2.5320712E-02
xyz  0.4591554      0.7427059      0.4719517
xyz  0.4083922      0.7455686      0.1403100
xyz  0.4416372      0.8393628      7.6342024E-02
xyz  0.1327285      0.4970000      0.4364171
xyz  9.6379094E-02  0.5855882      0.3802352
xyz  7.6066948E-02  0.6245000      0.3974865
xyz  0.1150683      0.7795883      0.3715025
xyz  0.1385160      0.7238529      0.4098982
xyz  9.2073016E-02  0.7063529      0.4022779
xyz  0.2152710      0.8265882      0.3764597
xyz  0.3304202      0.6161765      0.2311389
xyz  0.1806852      0.8512745      0.1618233

lambda 2
rawmadfile l2.int
wavelength 0.9794
fprimv_mad  -11.44
fprprv_mad  8.74

lambda 3
rawmadfile l3.int
wavelength 0.9797
fprimv_mad  -12.83
fprprv_mad  2.56

lambda 4
rawmadfile l4.int
wavelength 0.9897
fprimv_mad -2.42
fprprv_mad 1.13

nres 700                   [approx # of residues in protein molecule]
nanomalous 15               [approx # of anomalously scattering atoms per protein]
acceptance 0.10
SCALE_MAD                  ! read in and localscale the data
ANALYZE_MAD                ! run MADMRG and MADBST and analyze all the Pattersons
SOLVE                      ! Solve the structure
EOD
```

Summary information from the "solve.prt" output file produced after completion of the automated structure determination of beta-catenin

Correlation of anomalous differences. These indicate that the data all the way to about 2.7 A are contributing to the phasing, as the correlation >0.3 for wavelengths 1 vs 2.

```
          CORRELATION FOR

          WAVELENGTH PAIRS

 DMIN    1 VS 2   1 VS 3   1 VS 4   2 VS 3   2 VS 4   3 VS 4

 5.40     0.84     0.66     0.42     0.79     0.41     0.35

 4.05     0.75     0.53     0.36     0.69     0.35     0.33

 3.78     0.65     0.43     0.21     0.60     0.23     0.19

 3.58     0.67     0.38     0.24     0.58     0.27     0.22

 3.38     0.56     0.31     0.19     0.50     0.19     0.17

 3.24     0.53     0.28     0.12     0.40     0.14     0.14

 3.11     0.48     0.21     0.14     0.36     0.18     0.16

 2.97     0.44     0.25     0.11     0.38     0.18     0.11

 2.84     0.41     0.21     0.08     0.32     0.13     0.06

 2.70     0.33     0.11     0.10     0.25     0.13     0.11

 ALL      0.63     0.37     0.22     0.52     0.24     0.19
```

List of sites analyzed for compatibility with difference Patterson

```
   PEAK        X         Y         Z     OPTIMIZED
                                            RELATIVE OCCUPANCY
      1     0.833     0.115     0.231      66.078
      2     0.424     0.125     0.102      60.406
      3     0.944     0.337     0.076      55.988
      4     0.368     0.997     0.062      56.693
      5     0.681     0.354     0.162      49.508
      6     0.403     0.083     0.118      54.275
      7     0.049     0.243     0.028      41.397
      8     0.972     0.226     0.025      52.006
      9     0.389     0.281     0.127      42.309
     10     0.292     0.326     0.125      31.510
     11     0.410     0.212     0.095      27.341
     12     0.361     0.226     0.090      39.093
     13     0.910     0.250     0.144      22.225
     14     0.889     0.111     0.012      24.421

 Evaluation of this test soln with   14 sites after optimizing
 occupancy of each site

 Cross-vectors for sites  1 and  1 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1   -1.667  -0.229   0.500   7041.00     8732.56          2
   2   -1.667   0.000   0.037   6518.80     8732.56          2
   3    0.000  -0.229  -0.463   6185.17     8732.56          2

 Cross-vectors for sites  2 and  1 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1   -0.410   0.010  -0.130   6028.85     3991.49          1
   2   -1.257  -0.240   0.370   6627.21     3991.49          1
   3   -1.257   0.010   0.167   7920.79     3991.49          1
   4   -0.410  -0.240  -0.333   5973.46     3991.49          1

 Cross-vectors for sites  2 and  2 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1   -0.847   0.000   0.296   4925.09     7297.73          2
   2    0.000  -0.250  -0.204   5302.47     7297.73          2

(etc. for many many more cross-vectors)
```

Selenium atom occupancy, coordinates, and thermal factors, and Cross-validation fouriers calculated with all heavy atoms in all derivs except the site being evaluated and any sites equivalent to it.

```
(Peak height is height of peak at this position/rms of map)

  Site    x       y       z       occ       B     -- PEAK  HEIGHT --

    1   0.830   0.116   0.231   0.691  38.214             30.54
    2   0.422   0.124   0.103   0.671  44.433             25.98
    3   0.943   0.338   0.076   0.706  42.818             25.65
    4   0.367   0.996   0.063   0.527  15.000             24.49
    5   0.679   0.353   0.162   0.641  60.000             20.66
    6   0.406   0.084   0.119   0.574  32.160             22.10
    7   0.045   0.243   0.028   0.539  48.318             17.88
```

```
   8   0.970   0.225   0.026   0.764  60.000             18.08
   9   0.386   0.281   0.128   0.306  15.000             15.63
  10   0.289   0.326   0.125   0.303  33.767             12.07
  11   0.409   0.211   0.095   0.310  36.022             10.80
  12   0.362   0.225   0.091   0.192  15.000             12.00
  13   0.910   0.250   0.145   0.289  31.524              8.22
  14   0.891   0.110   0.011   0.456  60.000              8.36
```

Re-refinement of f' and f" values:

```
                    Final refined values of f-prime and f"

    Wavelength  ------- f-prime --------       --------f"--------------
        last refinement       Refined     last refinement       Refined

     1          -2.206         -2.206              5.365          4.357

     2         -10.957        -11.069             11.971          7.525

     3         -12.631        -12.740              3.032          2.000

     4          -2.714         -2.507              1.232          0.563
```

Figure of merit versus resolution, and anomalous and dispersive FH/E versus resolution

```
 FIGURE OF MERIT WITH RESOLUTION

 DMIN:           TOTAL    9.09   5.96   4.72   4.03   3.57   3.24   2.99   2.79
 N:               17155    946   1466   1815   2122   2386   2623   2798   2999
 MEAN FIG MERIT:   0.55   0.70   0.74   0.67   0.59   0.56   0.52   0.47   0.41

 RMS ANOMALOUS FH/E  [f" PART OF FH / RMS ANO ERROR]:

 LAMBDA:  1         0.6    1.1    1.2    0.9    0.7    0.6    0.5    0.4    0.4
 LAMBDA:  2         1.0    1.2    1.3    1.2    1.1    1.0    0.9    0.8    0.6
 LAMBDA:  3         0.3    0.4    0.6    0.4    0.3    0.3    0.3    0.2    0.2
 LAMBDA:  4         0.1    0.1    0.2    0.1    0.1    0.1    0.1    0.1    0.1

 RMS DISPERSIVE FH/E  [Delta-f-prime PART OF FH / RMS DISPERSIVE ERROR]:

 L1 VS L2:          1.0    1.4    1.5    1.3    1.1    1.0    0.8    0.7    0.6
 L1 VS L3:          1.1    1.5    1.6    1.4    1.2    1.1    0.9    0.8    0.7
 L1 VS L4:          0.0    0.1    0.1    0.1    0.1    0.0    0.0    0.0    0.0
 L2 VS L3:          0.3    0.6    0.5    0.4    0.4    0.3    0.2    0.2    0.1
 L2 VS L4:          1.0    1.3    1.5    1.3    1.1    1.0    0.9    0.8    0.6
 L3 VS L4:          1.2    1.4    1.7    1.4    1.2    1.1    1.0    0.9    0.8
.1
```

The summary of scoring for this solution

```
 Summary of scoring for this solution:
                          -- over many solutions--    -- this solution --
 Criteria                      MEAN          SD         VALUE        Z-SCORE
 Pattersons:                  4.89        0.745         14.2           12.5
 Cross-validation Fourier:    21.7         4.61         185.           35.5
 NatFourier CCx100:           11.0         5.29         58.9           9.05
 Mean figure of meritx100:   0.000E+00     5.00         69.3           13.9
 Correction for Z-scores:                                             -12.1

 Overall Z-score value:                                                58.8
```

Note that the Z-score for this solution (59) is much higher than for the gene 5 protein example, even though the phasing is about the same. This is because SOLVE scoring gets higher for datasets with more sites.

The end of the solve.status file:

```
 ****************************************************************************
                SOLVE STATUS      07-oct-00 11:46:31

 DATASET TITLE: armadillo repeat of beta catenin 4-wavelength MAD data
 TIME ELAPSED:     2 HR

 --------------------------------------------------------------------------
 CURRENT STEP:SOLVE MAIN PROGRAM
 STATUS:   DONE
 --------------------------------------------------------------------------
 --------------------------------------------------------------------------
    ---TOP SOLUTION FOUND BY SOLVE  (<m> = 0.69; score =  58.80) ---

         X       Y       Z        OCCUP     B        HEIGHT/SIGMA

   2    0.830   0.116   0.231    0.691    38.2         30.5
   2    0.422   0.124   0.103    0.671    44.4         26.0
   2    0.943   0.338   0.076    0.706    42.8         25.7
   2    0.367   0.996   0.063    0.527    15.0         24.5
   2    0.679   0.353   0.162    0.641    60.0         20.7
   2    0.406   0.084   0.119    0.574    32.2         22.1
   2    0.045   0.243   0.028    0.539    48.3         17.9
   2    0.970   0.225   0.026    0.764    60.0         18.1
   2    0.386   0.281   0.128    0.306    15.0         15.6
   2    0.289   0.326   0.125    0.303    33.8         12.1
   2    0.409   0.211   0.095    0.310    36.0         10.8
   2    0.362   0.225   0.091    0.192    15.0         12.0
   2    0.910   0.250   0.145    0.289    31.5          8.2
   2    0.891   0.110   0.011    0.456    60.0          8.4

      TIME REQUIRED TO OBTAIN THIS SOLUTION:    38 MIN
 --------------------------------------------------------------------------
 CURRENT RESOLUTION:   2.7 A.    FINAL RESOLUTION:   2.7 A.
~
```

Granulocyte-macrophage colony stimulating factor

Summary of this structure solution:

This is an MIR dataset with 4800 reflections to 3.5 A, 4 derivatives, and 254 amino acids. The data is courtesy of Kay Diederichs. The derivatives are not that good and the overall figure of merit of the structure is only 0.54 to 3.5 A. Using all the data and including anomalous differences, SOLVE took 18 minutes to solve this MIR problem on a DEC Alpha 500 MHz workstation. Solve only used 3 of the 4 derivatives in phasing.

Solve.setup file listing basic information about the crystals:

```
cell 47.6 59.1 126.7 90 90 90
symfile p212121.sym
resolution 20 3.5
```

Input script file used to run SOLVE on Granulocyte-macrophage colony stimulating factor

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
solve <<EOD > solve.log
! solve.com for gmf 7-25-97
! include known h.a. sites for comparison and fft map as well

logfile mir.logfile ! write out most information to this file.
@solve.setup
title gm native + 4 derivatives
!
readformatted
premerged
checksolve
comparisonfile gm_offset.fft
rawnativefile gmnat.fmt
derivative 1
```

```
inano
noanorefine
label deriv 1  gm18 pcmbs
rawderivfile gm18.fmt
 ATOMNAME Hg
 nsolsite_deriv 2
 XYZ   0.4069387      0.5974227      0.1901610
 XYZ   0.4322624      0.5161777      0.2020042
derivative 2
inano
noanorefine
label deriv 2 gmPt(EtNH2)2Cl2 derivative #40
rawderivfile gm40.fmt
 ATOMNAME Pt
 nsolsite_deriv 6
derivative 3
inano
noanorefine

label mersalyl acid # 52
rawderivfile gm52.fmt
 ATOMNAME Hg
 nsolsite_deriv 2
 XYZ   0.9070011      0.4427668      0.1972121
 XYZ   0.4240658      0.6003970      0.1951167
derivative 4
inano
noanorefine
label HgI2 #57
rawderivfile gm57.fmt
 ATOMNAME Hg
 nsolsite_deriv 3
 XYZ   0.9747854      0.4725027      0.2089491
 XYZ   0.3438405      0.6067868      0.1840420
acceptance 0.35       ! accept new sites with ~50% of height of avg
scale_native
scale_mir
analyze_mir
SOLVE
EOD
```

Summary information from the "solve.prt" output file produced after completion of the automated structure determination

Selenium atom occupancy, coordinates, and thermal factors, andCross-validation fouriers calculated with all heavy atoms inall derivs except the site being evaluated and any sites equivalent to it.

```
(Peak height is height of peak at this position/rms of map)

  Site    x       y       z       occ       B     -- PEAK  HEIGHT --
```

```
Deriv 1:
    1   0.404   0.600   0.192   0.130  60.000             14.80
    2   0.923   0.440   0.200   0.112  60.000             12.27

Deriv 2:

Deriv 3:
    1   0.423   0.600   0.195   0.169  60.000             21.90
    2   0.908   0.443   0.197   0.223  60.000             24.02

Deriv 4:
    1   0.973   0.481   0.210   0.126  60.000             12.80
    2   0.326   0.568   0.188   0.052  16.111              9.09
    3   0.357   0.637   0.182   0.040  25.935              3.11
```

Figure of merit versus resolution

```
 DMIN:            TOTAL   11.16   7.54   6.04   5.18   4.61   4.19   3.87   3.61
 N:                 4801    297    418    525    589    663    713    778    818
 MEAN FIG MERIT:    0.54   0.70   0.76   0.73   0.64   0.52   0.48   0.42   0.34
```

List of sites analyzed for compatibility with difference Patterson

(Height is 1000 x height of peak in Patterson/rms of map. Predicted height is expected height based on occupancy of sites)

```
Derivative 1:

   PEAK        X         Y         Z     OPTIMIZED
                                            RELATIVE OCCUPANCY
      1     0.406     0.597     0.191      47.139
      2     0.927     0.438     0.198      40.744

 Evaluation of this test soln with    2 sites after optimizing
 occupancy of each site

 Cross-vectors for sites  1 and  1 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1   -0.312  -1.194   0.500   4539.24     4444.25          2
   2   -0.812   0.500   0.118   2537.75     4444.25          2
   3    0.500  -0.694  -0.382   4848.71     4444.25          2

 Cross-vectors for sites  2 and  1 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1    0.521  -0.160   0.007   4568.46     1920.64          1
   2   -0.833  -1.035   0.507   4088.19     1920.64          1
   3   -1.333   0.340   0.111   3195.59     1920.64          1
   4    1.021  -0.535  -0.389   2322.98     1920.64          1
 Cross-vectors for sites  2 and  2 (excluding origin; 1000 = 1 sigma):
  #      U        V      W      HEIGHT   PRED HEIGHT      SYMM#
   1   -1.354  -0.875   0.500   1348.34     3320.13          2
   2    0.500  -0.375  -0.396   3381.48     3320.13          2
 Total of           1 of          11 patterson peaks used more than once.

 Overall quality of this Patterson soln =  5946.31
 Overall quality of the fit to patterson =  1.38812
 Avg normalized peak height =  1792.88
```

(... etc for derivatives 2,3,4).

.

Summary of scoring for this solution:

```
 Summary of scoring for this solution:
                           -- over many solutions--    -- this solution --
 Pattersons:                   1.87        0.500         3.82         3.89
 Cross-validation Fourier:     6.63         3.49         36.3         8.51
 NatFourier CCx100:            12.4         5.31         32.8         3.83
 Mean figure of meritx100:   0.000E+00      7.53         53.5         7.11
 Correction for Z-scores:                                            -3.92

 Overall Z-score value:                                               19.4
```

Tail end of the solve.status file:

```
 ***************************************************************************
                   SOLVE STATUS      07-oct-00 10:18:51

 DATASET TITLE: gm native + 4 derivatives
 TIME ELAPSED:    18 MIN

 ---------------------------------------------------------------------------
 CURRENT STEP:SOLVE MAIN PROGRAM
 STATUS:   DONE
 ---------------------------------------------------------------------------
 ---------------------------------------------------------------------------
     ---TOP SOLUTION FOUND BY SOLVE  (<m> = 0.54; score =  19.43) ---

 Deriv     X        Y        Z          OCCUP     B          HEIGHT/SIGMA



   1     0.404    0.600    0.192     0.130     60.0             14.8
   1     0.923    0.440    0.200     0.112     60.0             12.3

   3     0.423    0.600    0.195     0.169     60.0             21.9
   3     0.908    0.443    0.197     0.223     60.0             24.0
```

```
  4     0.973    0.481    0.210     0.126     60.0             12.8
  4     0.326    0.568    0.188     0.052     16.1              9.1
  4     0.357    0.637    0.182     0.040     25.9              3.1

       TIME REQUIRED TO OBTAIN THIS SOLUTION:    16 MIN
---------------------------------------------------------------------------
CURRENT RESOLUTION:   3.5 A.    FINAL RESOLUTION:   3.5 A.
```

e!doctype html public "-//w3c//dtd html 4.0 transitional//en">

Contents	Index

All keywords for SOLVE

This is a list of all the keywords for SOLVE. Keywords set values of parameters (number of heavy atom sites, f' value), while commands cause SOLVE to do something (solve a structure, scale data, search for heavy atoms, draw a map)

Also see the list of common keywords that apply to automated SOLVE operation and the list of all commands.

The keywords for SOLVE are listed here in the following groups:

Overall control parameters

```
VERBOSE                write out a lot of output to logfile
NSHELLS n              Number of shells for analysis [default=10]. Does not
                       apply to heavy atom refinement (fixed at 8).
SAVE_FILES             Save scratch files instead of cleaning up
```

Crystal data (cell, resolution, etc)

```
SYMFILE xxxxx       symmetry file for this space group
CELL   a b c alpha beta gamma
RESOLUTION dmin dmax
TITLE      xxxxx      CCP4 overall Title (60 characters, spaces allowed)
PROJECTNAME xxxxx     CCP4 Project Name (20 characters, no spaces)
CRYSTALNAME  xxxxx     CCP4 Crystal Name (20 characters, no spaces)
DATASETNAME xxxxx     CCP4 Dataset Name (20 characters, no spaces)
```

```
RES_PHASE   XX       Just use data up to XX in phasing and heavy-atom
                     searches, write out all data defined by dmin and dmax
NRES   n             # of residues in asymmetric unit [default=100].  Used to
                     estimate overall scale and (along with nanomalous) how big
                     Fa values might be.

NANOMALOUS n         # of anomalously scattering atoms in asymmetric unit.
                     Used to estimate how big the Fa values might be



SN_MIN  xx           Identify working resolution as the point where signal-
                     to-noise in the data goes down to about XX. Default =0.5

SN_RATIO_MIN  xx     Identify working resolution as the point where signal-
                     to-noise in the data goes down to about XX times its
                      maximum value. Larger of value of S/N obtained by SN_MIN and
                      SN_RATIO_MIN used. Default = 0.1
```

Reading in scaling, and rejecting data

```
READ_INTENSITITES  (default)  The raw data files contain intensity measurements
READ_AMPLITUDES    The raw data files contain amplitudes (F) not intensities (I)
                   (This is valid only with READFORMATTED)

PREMERGED          The data in all RAWMADFILEs have H K L and 4 other columns:
                   I+/F+, sigma, I-/F-, sigma
UNMERGED           The data in all RAWMADFILEs have H K L and 2 other columns:
                   I/F, sigma

READDENZO        All datafiles are written by Scalepack.  For unmerged data
                 they will be read with the formatting:(6i4,i6,2i2,i3,2f8.0) and
                 nsym*2+1 lines are skipped at the top of the file. For
                 merged data the formatting is:  (3i4,4f8.0) and
                 3 lines are skipped at the top of the file.

READFORMATTED      All datafiles will be read with "*" formatting and
                   contain H K L I/F sigma or H K L I+/F+ sigma I-/F- sigma

READTREK          The datafiles were written by d*trek and contain columns
                  with intensities

READCCP4_UNMERGED  All datafiles will be read as CCP4 mtz files assuming that
                   LABIN is defined as I=I SIGI=SIGI. No LABIN lines are allowed
                   (you cannot redefine LABIN).

LABIN             specify column assignments for HKLIN in standard CCP4
                  fashion (FC=FC1 PHIC=PHIC FOM=FOM) etc

HKLIN  xxx.mtz    mtz file containing scaled amplitudes
```

```
PHASES_FORMATTED xxx.fmt   File  xxx.fmt contains H K L FC PHIC FOM and
                    the phases and fom will be used in SOLVE with
                    difference Fouriers to find initial sites

PHASES_LABIN        specification of column assignments for PHASES_MTZ file
                    Normal use is FC=FC PHIC=PHIC FOM=FOM. NOTE: MUST
                    come before PHASES_MTZ!

PHASES_MTZ xxx.mtz    as PHASES_FORMATTED, but mtz-file. FC PHIC FOM required

NSKIP n            Skip exactly n lines at the top of each data file
NSKIP 0            Do not skip any lines at the top of each data file
NSKIP -1           Skip 0 lines at the top of each data file
                   unless the keywords READDENZO and PREMERGED
                   are set in which case the default number of lines
                   are skipped (see above)

RATMIN  xx         Minumum ratio of F/sig or I/sig to read in data for a
                   reflection at all is xx [default=2.0].  This is
                   used to eliminate weak data.

SIGMA_I_RATIO XX   Data read in with "premerged" will be given sigmas of
                   at least XX. Default=0.0

FPFM_ONLY          Toss all acentric reflections where either F+
                   or F- is missing [this is the default for MAD data]
FP_OR_FM           Use F+ or F- as an estimate of Fbar if F+ and
                   F- are not both present.

SWAP_ANO           Swap H K L -> -H -K -L as data are read in to
                   SOLVE in scale_native, scale_derivative, and
                   scale_mad. This is to correct for a detector or indexing
                   that swapped F+ for F-
OVERALLSCALE       Do not do local scaling; just an overall
                   scale factor for F+, F- at each wavelength.
                   Use this if you already have scaled the data
                   and you don't want any more scaling done.
KEEPALL           keep reflections even with high differences
TOSSBAD           (default)Toss reflections if differences between native and
                  derivative are more than 3 * the rms found for other
                  reflections.
                  Note: KEEPALL and TOSSBAD apply to MERGE, LOCALSCALE,
                  SCALE_MAD, SCALE_MIR, SCALE_NATIVE. This is the
                  place to reject derivative reflections with very large del F
                  if you want to reject them at all.
ANCUT             minimum # of reflections to use to scale a reflection (30.)
RATMIN            minimum ratio of F/sigma to include (default=2)
NOBFACTOR         if specified, do not apply overall Wilson scaling before doing
                  local scaling. Generally used only along with DAMPING=0.
BFACTOR           undoes NOBFACTOR. Do apply Wilson scaling before local scaling
DAMPING xx        scale factor (after Wilson scaling) is damped by taking it
                  to the power xx. Generally used with NOBFACTOR and a value of
                  0 to not do any scaling at all.
NODAMPING         undoes DAMPING by resetting damping factor to 1.0
OVERALLSCALE      just get 1 scale factor for the whole dataset. No local
                  scaling, no wilson scaling. Same as NOBFACTOR + DAMPING 0.0
NOOVERALLSCALE    undoes OVERALLSCALE. SAME AS BFACTOR + DAMPING 1.0
```

Specifying file names

```
LOGFILE  xx       name of output file for summary of results is

SYMFILE      xxxxx       symmetry file for this space group
INFILE  data.drg    Standard datafile for input.  Some routines require that
                    other input file names are specified.
OUTFILE  data.xplor Standard output file.   Some routines require that
                    other output file names are specified.

RAWMADFILE     xxx.int   read in xxx.int as data for the current mad wavelength
RAWNATIVEFILE  xxx.int   read xxx.int as data for the native
RAWDERIVFILE   xxx.int   read xxx.intas data for the current derivative
EXPORTFILE   xxx    Formatted file with Fp,phase, m, and Hendrickson-
                    Lattman coefficients will be written to this
                    file at the end  of SOLVE.  Only the top solution
                    will be written out in this way.  The file has
                    a header that you can edit and make into a little
                    script file that will read the data into ccp4
                    format. Default is phases-hl.export

PHASEFILE           binary .drg file with the same data as the EXPORTFILE
                    Default is phases-hl.drg

NEWSCRIPTFILE       script file that will run HEAVY and write out a
                    new EXPORTFILE,PHASEFILE, and NEWSCRIPTFILE.
                    Default name is "phases-hl.script".  All
                    this script file does is calculate phases.  The file
                    does contain the final heavy atom parameters. If you
                    want to take the heavy atom parameters and continue
                    in SOLVE with them (i.e., running ADDSOLVE or
                    ANALYZE_SOLVE), you should copy these heavy atom
                    parameters into the solve_mad.script or
                    solve_mir.script file that was
                    written by "ANALYZE_MAD" or
                    "ANALYZE_MIR" and make
                    a new script file this way.
MADFBARFILE xx.scl  Output file from SCALE_MAD with (Fbar,sigma,DelAno,sigma)
                    for each wavelength will be xx.scl
                    (DEFAULT="mad_fbar.scl")

MADFPFMFILE yy.scl Output file from SCALE_MAD with (F+,sigma,F-,sigma) for each
                    wavelength will be yy.scl
                    (DEFAULT="mad_fpfm.scl")
```

```
madmrgfile xxx.out    SIRAS-like MAD dataset from MADMRG
                    [     default="madmrg.out"]

madbstfile yyy.out  coefficients for a Bayesian Patterson to yyy.out from MADBST
                          [default="madbst.out"]

SOLVEDATAFILE  xxx  Output datafile with MADMRG and MADBST data

NEWFILE  xx       file with updated script file for HEAVY
OUTFILE   xx     output file name for HEAVY, required if KOUT is not 0



FFTFILE   xx        name of FFT-containing file
EZDMAPFILE   xx       name of output EZD format map file
CCP4MAPFILE   xx       name of output ccp4 format map file
MAPVIEWFILE xx      name of output MAPVIEW format file
COMPARISONFILE   xx   name of file with FFT to be compared with all native
                    fouriers from trial solutions (goes with checksolve).

FILETITLE xxx       optional title for a file
```

Column numbers for data in data files

```
NNATF n           column # for F of native data
NNATS n           column # of sigma of F of native data
NDERF n           column # for F of deriv data
NDERS n           column # for sigma of F of deriv data
NANOF n           column # of anomalous difference (Fplus-Fminu) of deriv data
NANOS n           column # of sigma of anomalous difference
NCOLF_MERGE n     column number in input file for F (default = 1)
NCOLSIG_MERGE n   column number in input file for sigma of F (default =2)

NCOLFBAR   n        Fbar for this wavelength
NCOLSFBAR  n        sigma of Fbar
NCOLDELF   n        Del F ano (Fplus - Fminus)
NCOLSDELF  n        sigma of del F ano
NCOLFPLUS               column number for Fplus
NCOLSIGPLUS             column number for sigma of Fplus
NCOLFMINUS              column number for Fminus
NCOLSIGMINUS            column number for sigma of Fminus
NCOLFC n                WT Fc (WT Fcalc) for Fdiff
NCOLFOWT n              WT Fo (WT Fobs) for Fdiff
NCOLSWT n               sigma of WT Fo for Fdiff
NCOLFOMUT n             MUT Fo (MUT Fobs) for Fdiff
NCOLSMUT n              sigma of MUT Fo for Fdiff
NCOLFCMUT n             MUT Fc for Fdiff
NCOLRTEST n             RTEST indicator (0 if missing) for Fdiff
NXPLORF  n          column for F in drgtoxplor
NXPLORSIG n         column for sigma in drgtoxplor
```

```
JSTD    n           wavelength ID for wavelength to be considered the STANDARD
```

Control parameters for SOLVE structure determination

```
new_dataset        Used to identify the beginning of a new dataset
                        in combination with COMBINE_ALL_DATA.
                        Cannot be used with mtz files.
ntopfour xx         Number of Fourier peaks to pick from a map
ntopderiv xx        Number of Fourier peaks to be tested for
nsolsite xx         Maximum number of sites in a derivative unless overridden by
                    nsolsite_deriv
nseedtest xx        Number of seeds per derivative to try (before sorting)
nseedsolve xx       Number of seeds (total) to try after sorting them
ntopsolve xx        Number of solutions to print out at the end and number
                    of solutions to keep track of at any one time
addsolve            Add on to solution that is input[default=off]
checksolve          Compare all solutions to input solution [default=off]

analyze_solve       Analyze input solution without doing anything else
                    [default=off]

no_fom             do not use figure of merit in solve scoring
                   (this is default in P1 where figure of merit is poorly
                   estimated)
no_patterson       do not use patterson in solve scoring
no_fourier         do not use cross-validation difference fouriers in scoring
no_native_fourier  do not use analysis of native fourier in solve scoring

[no]delete          do [not] check out all solutions by testing
                    all one-site deletions [default=delete]

[no]inverse         do [not] check out all solutions by testing
                    their inverses (does not apply if a solution
                    is centrosymmetric or if anomalous differences
                    are not used). [default=inverse]
full_inverse        if inverse is set, calculate all scores for inverse, do not
                    copy over patterson and cross-validation difference fourier

SCORING_TABLE (8 values)  Scoring table (usually generated by SOLVE) consisting
                      of mean and standard deviation of scores for trial
                      solutions for Pattersons, Cross-fouriers, Native Fourier
                      maps, and mean figure of merit.  This keyword is useful
                      when you are running SOLVE after modifying the script
                      file it writes out at the end.

QUICK                once a plausible solution is found, don't keep looking,
                     just add on sites to it and check it at the end.

QUICKER              Go with the best solution at each stage, but try all seeds ,
                     unless a very good solution is found.

VERYQUICK            SOLVE will go with the best solution at each step, even if
                     it is not very good.  This will speed up SOLVE a lot for good
data.

THOROUGH             keep looking anyways until a limit set by ntopsolve,
                     nseedsolve, etc is reached.  Opposite of QUICK.

FINISH     control parameter for solve.control file indicating that SOLVE should
           finish up as soon as possible without looking for any new solutions.

RESOLUTION_STEPS    number of steps of resolution to use in the search for
                    heavy-atom solutions.  Default=3.  If you specify 0 or 1 it
                    will go right to the highest resolution available



NTOL_SITE            a site within ntol_site grid units of an existing site is
                     considered to be a duplicate and is ignored. [default=8]

NTOL_SOLN            a heavy-atom solution for which every site matches another
                     solution within ntol_soln grid units is considered to be
                     a duplicate and is ignored. [default=2]

ACCEPTANCE  xx       the weighting function for scoring patterson and free-
                     difference fourier peak heights is adjusted so that a new
                     site with height relative to the previous average height
                     of ACCEPTANCE or higher will generally give a solution
                     with a higher score than the solution without this site.
                     [default =0.2]

CUT_DELETE  xx       Only sites with free-difference Fourier peak heights less
                     than XX *sigma of map will be will be considered for removal
                     in generating new test solutions. Default = 5.0

bayes                Use Bayesian MAD phasing at the very end of SOLVE.
                     (This is the default)

nobayes              Use the compressed MADMRG datafile for all
                     phasing when program gets to SOLVE.

NO_ANISOTROPIC_B     no anisotropic b used in automated structure determination

no_duplicate_sites   Do not allow the same site to appear in more than
                     one derivative.(default=off)

CONTINUE_SAD         Continuing a SAD dataset (keyword used by SOLVE in
                     solve_fast_sad.script

CONTINUE_MAD         Continuing a SAD dataset (keyword used by SOLVE in
                     solve_mad.script

USE_INPUT_PHASES     Using input phases (must be defined in input file)
                     (keyword used by SOLVE in solve_mad.script and solve_mir.script)
```

Defining heavy atom scattering factors

```
The atom types recognized by SOLVE are:
H, H-1, He, Li, Li+1, Be, Be+2, B, C, Cv, N, O, O-1,
F, F-1, Ne, Na, Na+1, Mg, Mg+2, Al, Al+3, Si, Siv, Si+4,
P, S, Cl, Cl-1, Ar, K, K+1, Ca, Ca+2, Sc, Sc+3, Ti, Ti+2,
Ti+3, Ti+4, V, V+2, V+3, V+5, Cr, Cr+2, Cr+3, Mn, Mn+2, Mn+3,
Mn+4, Fe, Fe+2, Fe+3, Co, Co+2, Co+3, Ni, Ni+2, Ni+3, Cu,
Cu+1, Cu+2, Zn, Zn+2, Ga, Ga+3, Ge, Ge+4, As, Se, Br,
Br-1, Kr, Rb, Rb+1, Sr, Sr+2, Y, Y+3, Zr, Zr+4, Nb, Nb+3,
Nb+5, Mo, Mo+3, Mo+5, Mo+6, Tc, Ru, Ru+3, Ru+4, Rh, Rh+3,
Rh+4, Pd, Pd+2, Pd+4, Ag, Ag+1, Ag+2, Cd, Cd+2, In, In+3,
Sn, Sn+2, Sn+4, Sb, Sb+3, Sb+5, Te, I, I-1, Xe, Cs, Cs+1,
Ba, Ba+2, La, La+3, Ce, Ce+3, Ce+4, Pr, Pr+3, Pr+4, Nd,
Nd+3, Pm, Pm+3, Sm, Sm+3, Eu, Eu+2, Eu+3, Gd, Gd+3, Tb,
Tb+3, Dy, Dy+3, Ho, Ho+3, Er, Er+3, Tm, Tm+3, Yb, Yb+2,
Yb+3, Lu, Lu+3, Hf, Hf+4, Ta, Ta+5, W, W+6, Re, Os, Os+4,
Ir, Ir+3, Ir+4, Pt, Pt+2, Pt+4, Au, Au+1, Au+3, Hg, Hg+1,
Hg+2, Tl, Tl+1, Tl+3, Pb, Pb+2, Pb+4, Bi, Bi+3, Bi+5, Po,
At, Rn, Fr, Ra, Ra+2, Ac, Ac+3, Th, Th+4, Pa, U, U+3, U+4,
U+6, Np, Np+3, Np+4, Np+6, Pu, Pu+3, Pu+4, Pu+6, Am, Cm, Bk, Cf

newatomtype xxxx    define scattering properties of atom xxxx
aval a1 a2 a3 a4    4 real numbers (a1,a2,a3,a4)  from International Tables for
                    the most recently defined newatomtype
bval b1 b2 b3 b4    b values for newatomtype (NOTE: not the same as BVALUE).
cval c              c value for newatomtype
fprimv xx           f' value for newatomtype
fprprv xx           f" value for newatomtype

                    Where the fo scattering from this atom is:
       cval +  sum_{i=1 to 4} (aval(i)*exp(-bval(i)*(sin(theta)/lambda)**2)
       and the f" value is fprimv and the f"" value is fprprv

    For CLUSTER compounds, you may wish to specify instead fprimv, fprprv,
plus 3 separate formulas, one for the f_o part of the scattering factor,
1 for NORMALIZED f", and one for NORMALIZED f"".
All three of these quantities will depend strongly on scattering angle.

NOTE: when you put in a cluster compound, it is a good idea to look at
the form factors as a function of sin(theta)/lambda using the keyword

plot_formfactors xxxx



The formulas used are:
```

```
                    f_o formula:

The "clus_aval",
"clus_bval","clus_cval(1)" values
will be used almost as for regular atoms, but with 2 additional
factors cval(2) and cval(3) that put in a sinc function as described
in Fu et al., Cell 98, 799 (1999):

clus_aval           4 real numbers (a1,a2,a3,a4)
clus_bval           4 real numbers
clus_cval           3 real numbers

 fo= clus_cval(1) +
 sinc(clus_cval(2)*(sin(theta)/lambda)**clus_cval(3))**2  *
   [sum_{i=1 to 4} (clus_aval(i)*exp(-bval(i)*(sin(theta)/lambda)**2)]

Note the sinc function multiplies the whole sum over the aval/bval terms
but NOT the cval(1) term (it is unclear how this was done
in the Fu et al paper).

                    f" and f"" formulas

clus_fp_aval           4 real numbers (a1,a2,a3,a4) for f"
clus_fp_bval           4 real numbers
clus_fp_cval           3 real numbers

clus_fpp_aval           4 real numbers (a1,a2,a3,a4) for f""
clus_fpp_bval           4 real numbers
clus_fpp_cval           3 real numbers


The formulas for f" and f"" are a little different so as to
preserve compatibility with the definitions for normal atoms.  The
definitions here are not used in the Fu et al article mentioned above.
In each case, the value used for f" and f""
in all the routines is equal to their INPUT values (fprimv and fprprv or
fprimv_mad and fprprv_mad)
TIMES an overall form factor given by (for f"):

 form_factor= clus_fp_cval(1) +
 clus_fp_cval(2)*sinc((sin(theta)/lambda)**clus_fp_cval(3))**2  *
   [sum_{i=1 to 4} (clus_fp_aval(i)*exp(-bval(i)*(sin(theta)/lambda)**2)]

and for f"":

 form_factor= clus_fpp_cval(1) +
 clus_fpp_cval(2)*sinc((sin(theta)/lambda)**clus_fpp_cval(3))**2  *
   [sum_{i=1 to 4} (clus_fpp_aval(i)*exp(-bval(i)*(sin(theta)/lambda)**2)]


Example of CLUSTER form factor input:
```

```
newatomtype wclu

fprimv 100.0    ! fprime value, to be multiplied by clus_fp form factor
fprprv 50.0   ! fprime value, to be multiplied by clus_fpp form factor

clus_aval 2093 5109.4 -1197.1 5254.3   ! form factors for f_o scattering
clus_bval 509.3 -37.8 849.4 108.5
clus_cval 184 30 1.2

clus_fp_aval 0.185886 0.453782 -0.10632 0.466651  ! form factors for fprime
clus_fp_bval 509.3 -37.8 849.4 108.5
clus_fp_cval 184 30 1.2

clus_fpp_aval 0.185886 0.453782 -0.10632 0.466651  !form factors, fdoubleprime
clus_fpp_bval 509.3 -37.8 849.4 108.5
clus_fpp_cval 184 30 1.2



Example of CLUSTER form factor input for mad atom:

Same as above for the newatomtype.  Then specify:
mad_atom wclu

and for each wavelength specify (just as usual)
fprimv_mad 100.
fprprv_mad 50.

Then the clus_fp_aval etc will be applied to the fprimv_mad value, and the
clus_fpp_aval etc will be applied to the fprprv_mad value as a function
of sin(theta)/lambda.

mad_atom  xxxx      name of the anomalously scattering atom is xxxx.

fprimv_mad       f' value for anomalously scattering atom at a particular
                 wavelength (must be input after each wavelength)
fprprv_mad       f" value for a particular wavelength

FIXSCATTFACTORS     Fix scattering factors at their input values [default]
REFSCATTFACTORS     refine scattering factors f' and f".
```

Heavy atom parameters

```
derivative n        begin input of information for derivative/wavelength n
                    This command is used to start entering information on a
                    derivative.  If you want to modify something after you've
                    gone on to another derivative then you need to use the
                    command GOTODERIV
lambda n            identical to derivative n
gotoderiv n         go to previously specified derivative (wavelength) n
                    and get readyto read some modifications of the parameters
                    for this derivative.
```

```
LABEL   text        label for this  wavelength

IEGROUP n           Group of correlated derivatives that this derivative
                    belongs to.  Determined automatically if GETGROUPS is set
DERSCALE        Dividing scale factor applied to all this derivative data
                  after overall scale factor has been applied.  DEFAULT=1.0
DERTEMP         Dividing B-factor to apply to deriv data.  DEFAULT =0.
SIGDERSCALE     Scale factor to apply to derivative sigmas after all above
                  scaling is applied. DEFAULT = 1.0


ATOMNAME XXXX   XXXX is the atom type of an atom to be refined.
                Please note:  the f' and f" values in SOLVE's database are
                for lambda=1.54 A.  If you collected MIR data at a
                synchrotron then you should define a new atom type with
                NEWATOMTYPE and input the correct f' and f" values.
                HINT:  you can get the aval, bval, and cval values for an
                atom recognized by SOLVE by typing madatom [atom name].

                When you type ATOMNAME, SOLVE assumes you are typing in a new
                atom and it zeroes out all the parameters for this new atom.
                If you want to go back to this atom later (i.e., in another
                cycle) use the keywords GOTODERIV and GOTOATOM to identify
                this atom.

                When you have multiple sites for a particular derivative,
                use ATOMNAME XXXX for the first, then input all the data on
                that site, then start the next site with ATOMNAME YYYY, and
                so forth.
gotoatom n      go to the n'th atom in this derivative/wavelength
                and get ready to read some modifications of its parameters


OCCUPANCY x     Fractional occupancy of this atom
                Note: if occupancy is equal to 0.000 or refines to 0.000,
                the atom is ignored in refinement.

BVALUE  b       Temperature factor for this atom.  Anisotropic temperature
                factors are also supported (just input 6 numbers.)
                NOTE: bval is not the same as bvalue.  BVAL refers to
                scattering factors for a newatomtype.

XYZ     x y z   Fractional coordinates of this atom
                Note: if coordinates move dramatically during refinement,
                the occupancy is set to zero and the atom ignored.

PDB_XYZ_IN        PDB file with orthogonal A coordinates
                of all heavy-atoms for this derivative/wavelength
                NOTE: you cannot use XYZ/OCC/BVALUE/REFINEMENT parameters along with
pdb_xyz_in.
```

Control parameters for HEAVY atom refinement and phasing

```
NO_SIM             Do not use Sim weighing with heavy-atom structure factors in
SAD
CORRELPHASE        Use Bayesian correlated MIR phasing (default)
GETGROUPS          Automatically get groups of correlated derivatives for
                   correlated MIR phasing
IMADPHASE    n     wavelength to phase using Bayesian MAD phasing
                   (default =0, no Bayesian MAD phasing). See  description below.
                   n refers to the wavelength to be defined as a reference.
                   Note that you cannot refine with madphase, you can only phase.

Note: Flags for refinement of a heavy atom are cumulative, so you can
refine x and y using REFINEX and REFINEY.

Note: Flags for refinement of a heavy atom do not apply when the keyword
SOLVE is used (only with HEAVY)

REFINENONE       Don't refine anything...reset all the refinement flags to zero
REFINEALL        Refine x,y,z,occupancy, and B
REFINEOCCB       Refine occupancy and B
REFINEXYZ        Refine x,y,z
REFINEX          Refine x
REFINEY          Refine y
REFINEZ          Refine z
REFINEOCC        Refine occupancy
REFINEB          Refine B



NOREFINESCALE    Do not refine overall scale factor. Default = refined
REFINETEMP       Refine B-factor applied to deriv data. DEFAULT= not refined

nsolsite_deriv       Maximum number of sites for this derivative only
cutoff_deriv 200. 3.5        resolution limits for this derivative/wavelength only
RES_PHASE 2.2    high-resolution limit for phasing only
SN_MIN     Set RES_PHASE so that signal-to-noise is bigger than this
SN_RATIO_MIN     Set RES_PHASE so that signal-to-noise is bigger than this
                     ratio times the value at low-resolution
INPHASE              include this wavelength/derivative in phasing.
NOINPHASE            do not this include this derivative/wavelength in phasing
INANO                include anomalous differences for current wavelength/deriv

noanorefine         use anomalous differences in phasing but not
                    refinement for this derivative.
                    (this is usually the best option for MIR
                    unless your anomalous differences are really
                    big, as from a synchrotron MIR dataset at an
                    absorption edge).  Note: you still have to specify
                    for each derivative "inano" to include anomalous
                    differences for that derivative.

anorefine           For this derivative with "inano" specified, use
```

```
                   anomalous differences in both refinement and phasing.
                   This is best for MAD data. (This is the default also).
                   Applies to current derivative/wavelength

ISOONLY            use only isomorphous differences in phasing and
                   refinement this deriv
ANOONLY            use only anomalous differences in phasing and refinement
                   this deriv
ISOANO             use both isomorphous and anomalous differences this deriv
                   (cancels ISOONLY and ANOONLY)(default)

KOUT     n      type of output from HEAVY if any. DEFAULT = 0 (no binary output)

                KOUT...TYPE OF OUTPUT
        0.... NONE
        2.... DIFFERENCE FOURIER FOR KDER
           A=m(Fder-Fnat)cos(PhiBest)
           B=m(Fder-fnat)sin(PhiBest)
        3.... ANOM DIFF FOURIER FOR KDER
           A=m(DelAno)cos(PhiBest-90)
           B=m(DelAno)sin(PhiBest-90)
        4.... RESIDUAL MAP FOR KDER
           A=m(Fder-|Fnat+FH|)cos(PhiBest)
           B=m(fder-|Fnat+FH|)sin(PhiBest)
            (where Fnat+FH is the vector sum of Fnat
              and the heavy atom FH)
        6.... NATIVE FOURIER
           A=m(Fnat)cos(PhiBest)
           B=m(Fnat)sin(PhiBest)
        7.... PHASES AND FIGURE OF MERIT and fnat
              PhiBest (in degrees), PhiMostProbable,
                and figure of merit, and Fnat
        8.... Fnat,phibest, phi most probable, figure of
              merit, HENDRICKSON-LATTMAN COEFFS
        9.... HEAVY ATOM S. FACTORS FOR KDER
           A, B= real and imaginary parts of normal scattering from heavy atom.
           C, D= real and imaginary parts of anomalous scattering
         NOTE: m=the figure of merit, PhiBest is the "Best" phase,
          PhiMostProbable is the the most probable phase.

KDER    n       derivative n is to be included in output

IANGLE   x      phasing angle, minimum=5, default=5
INANAL   n      PHASE ANALYSIS.  DEFAULT=0
                  1 for printing of extensive heavy atom statistics

INRESD   n      RESIDUAL AND STATISTICS.  DEFAULT = 0
                -1  No residuals or statistics calculated.
                0   zeroth cycle added before first refinement
                    cycle. During zeroth cycle residuals and
                    statistics are calculated and printed.
                    No statistics are calculated on other cycles.
                1   Residuals and statistics calculated every
                    cycle and printed according to INPRNT.
                    Note: residuals are only calculated for
                    derivatives with INPHAS = 1.
INOSIG  n       USE OF SIGMAS.  DEFAULT = 0 (use sigmas).
                 1  if sigmas from input data file are not to be used.
INHEND  n       USE OF HENDRICKSON-LATTMAN COEFFICIENTS.  DEFAULT=0 (don't use)
                 1  if Hendrickson-Lattman coefficients are to be calculated and
                    used for phasing instead of Blow-Crick phasing.
                    HEAVY does not do phase combination.  If KOUT=8 and
                    INHEND=1, then standard H-L ABCD are output.  If KOUT=8 and
                    INHEND=0, then HL ABCD are fit by least squares to the
                    phase probability distribution.
INPRNT   n      PRINTING OF SHIFTS. DEFAULT =0 (don't print)
                 1  if shifts (and statistics, if any) are
                    to be printed on every cycle.  Default
                    is to print statistics on first cycle,
                    shifts and statistics on last.
JALT     n      USE OF PHASES IN REFINEMENT.  DEFAULT = 0 (Patterson refinement)
                   0 is to use origin-removed Patterson refinement.
                   1 is to use phase refinement at most probable phase
                   JALT and KALT are set automatically to 0
                   if you use a procedure (IHEAVYPROC > 0)

KALT     n      USE OF DERIVATIVE BEING REFINED IN PHASING. DEFAULT=0(don't use)
                   0 is not to use derivative being refined in phases
                   1 is to use all available derivatives in phasing
                   JALT and KALT are set automatically to 0
                   if you use a procedure (IHEAVYPROC > 0)

NCYCLE    n     Number of cycles of refinement to be carried out if a
                PROCEDURE is NOT used (see IHEAVYPROC).  Maximum = 30
                Default = 0
IREFCY    n     List of derivative numbers to be refined during the NCYCLE
                cycles of refinement if a procedure is NOT used. Default = 0
                  i.e., 1,1,1,1,0 means refine deriv #1 on cycles 1-4 and
                  calculate phases, get residuals, figure of merit, etc
                  on cycle #5.  Note that you don't get these statistics on
                  cycles in which you refine with Patterson refinement.

IHEAVYPROC  n   RUN a procedure with HEAVY.  Default =5 (refine everything)
                Available procedures:
                1 = NREP cycles of refinement of each deriv that has INPHASE
                  specified, refining only occupancy.
                2 = as 1, but refining only xyz.  Fixes coordinates of best
                  atom in each deriv in polar space group in polar directions
                  unless another atom is already fixed by user.
                3 = as 2, but refining xyz and occ
                4 = as 3, but refining xyz, occ, B
                5 = 1, then 2, then 3, then 4
                6 = phased refinement to obtain relative coordinates among
                  derivatives for polar directions in polar space groups.
```

```
                    Fix and phase with best derivative.  Refine just coordinates
                    in polar directions for all other derivatives with INPHASE
                    specified.  This should be followed by #5 again.
                  Note: when you use this procedure you must still set which
                  refinement flags you want to (ever) be refined.  The
                  procedures only turn OFF flags, they do not turn on ones
                  you have never set.

NREP    n         # of refinements of each deriv in procedures with
                   IHEAVYPROC > 0 (default=5)

SMALL   xx         minimum ratio of derivative structure factor amplitude
                   (F Deriv) to RMS lack-of-closure for use in
                   refinement or residuals. DEFAULT=0.
FMIN    xx        minimum native F for any action. Default=0.
FOMMIN  xx        minimum figure of merit for use in phased
                     portion of refinement. Default =0.
BMAX                maximum allowed bvalue default = 60
BMIN                minimum allowed bvalue default = 15



THR     xx        Keywords to set threshold and damping factors for shifts:
ACL     xx        if SHIFT > THR *sigma of SHIFT, SHIFT=SHIFT*ACL
                           Defaults are 0. and 0.5

FSIGMIN  xx       MINIMUM ratio of F/sig to include. DEFAULT =1.0

NNATF     n       column number in input file for native F
NNATS     n       column number for sigma of native F

NNATF_DERIV n     column number for native F that pairs with this derivative
NNATS_DERIV n     sigma of native F that pairs with this derivative

NBST     n        optional column number for "best" phase in input file
NMP      n        optional column number for most probable phase in input file
NFIGM    n        optional column number for figure of merit in input file
                    Note: if you set NFIGM for routine HEAVY it will be
                    applied in MAPS too.  Be sure you reset it to the value you
                    want.

INOLD    n        Flag for using phases from input file in phasing when they
                    are not available from current data. default = 0.  To use
                    input phases, inold=1

ANATSCALE xx   Overall scale factor applied to ALL data before any
                    other scaling.  DEFAULT = 1.0
SIGNATSCALE  xx   Scale factor applied to native sigmas after overall scaling
                    DEFAULT = 1.0
EIS               Optional list of estimated rms isomorphous lack-of-closure
                    residuals in 8 resolution ranges
EAD               Optional list of estimated rms anomalous lack-of-closure
                    residuals in 8 resolution ranges
```

```
FPHBAR          Optional list of estimated rms derivative F in 8
                  resolution ranges
FHBAR           Optional list of estimated rms heavy atom F in 8
                  resolution ranges
SIGBAR          Optional list of estimated rms derivative sigma in 8
                  resolution ranges

USE_F_BINS      Use |F| and resolution to set bins for phasing statistics

                 (only used internally, does not affect printout)

NO_USE_F_BINS      Use only resolution to set bins for phasing statistics

N_F_BINS        Number of bins (for each resolution range) based on F for

                phasing
```

Control parameters for HASSP

```
searchregion            region to search (xs,xe,ys,ye,zs,ze) (default = region
                            covered by the FFTGRID)

ihassptype n            control of what is to be done (default=0)
                        0    search for single-site solutions, then 2-sites
                        2    search for single-site solutions
                        5    same as 0
                       -5    use trial solutions given in keyword trialsite
                              as trial cross-vectors
                       -6    use trial sites given in trialsite as a
                              trial solution. evaluate it and look for more
                              sites

discrm  xx              ratio of peak height over surroundings to use
                          (default=1.0)
icrmax     n            maximum # of peaks to try in 2-site search (default=10)
nospec     n            control over ignoring symmetry #'s of special positions
                          (default=0, do not ignore)
nsignf     n            0 if significance of peaks is to be tested (default=0)
spat       xx            minimum probability for non-randomness to keep a peak
                          in routine "patpk". (default=0.0)
ssin       xx           as spat, but for single-site searches (default=0.0)
sdub       xx           as spat, for two-site searches (default=0.95)
strp       xx           as spat, for 3-site searches (default=0.0)
ssft       xx           as spat, for sifting through 3-site solutions
                         (default=0.95)

ihassplist              print out value of patterson function
                          at every predicted self- and cross-vector for every
                          solution.  this is very helpful for analyzing a
                          solution in detail. default =0 (not to print out).
```

```
trialsite   x y z       fractional coordinates of a trial site or cross vector
                          (read in if ihassptype < 0)

ntophassp       n       a maximum of n solutions will be saved in routines
                        that use hassp output to build up solutions

sigma_remove       n       peaks labelled with uvw_remove will be set to this value
uvw_remove u v w       n      remove this uvw peak from the Patterson (set to
sigma_remove*sigma)
```

Working with maps

```
PATTERSON            this is a Patterson map
FOURIER              this is a fourier map
```

Grids used for FFT calculations

```
FFTGRID  xs xe xtotal ys ye ytotal zs ze ztotal    grid for FFT calculations
PATTGRID xs xe xtotal ys ye ytotal zs ze ztotal    grid for Patterson
EZDGRID  xs xe    ys ye    zs ze                   grid for NEWEZD map
BOSSGRID  xs xe    ys ye    zs ze                  grid for MAPVIEW map .
```

Control parameters for GENERATE

```
coordinatefile          pdb file with coordinates for GENERATE
percent_error           % error added to intensities in GENERATE
cell_derivative a b c alpha beta gamma  (only for generate_mir) cell parameters for
this derivative.
```

How SOLVE works

MAD structure determination and overall SOLVE operation

The real power of the SOLVE package is the automated solution of MAD and MIR datasets. For a MAD dataset, all you need to do is tell the program about your space group and cell dimensions, where the unscaled intensity data files are, and what the scattering factors are for the MAD dataset. The program scales the data using localscaling in SCALE_MAD , calculates difference Patterson maps, compresses the MAD data into SIRAS-like data using MADMRG, calculates an optimized Bayesian heavy-atom Patterson using MADBST, and iteratively builds up and scores potential heavy-atom solutions for the MAD dataset with the SOLVE routine. Potential solutions are scored on the basis of (1) agreement with the Patterson, (2) "free" difference Fouriers, (3) the presence of "solvent" and "protein" regions in a native electron density map, and (4) the figure of merit of phasing.

SOLVE can either look exhaustively for solutions (trying all possible additions/deletions/inversions etc) or else just follow the best solution and keep adding on to it. The default is to follow the best solution (keyword is "mediumquick"), and SOLVE keeps looking at seeds until this process leads to a result with a figure of merit > 0.5 and a Z-score over 10. If you set the "veryquick" keyword, SOLVE will just look at one seed this way.

Final phases for the top solutions are written out along with Hendrickson-Lattman coefficients for calculating maps and solvent flattening. SOLVE refines scattering factors for the MAD data after heavy-atom parameters are found by comparing the effective occupancies of sites when refined against each possible set of dispersive or anomalous differences. SOLVE calculates final phases using Bayesian correlated MAD phasing.

SAD structure determination

SAD data is single-wavelength data with anomalous differences. SOLVE treats SAD data as a native (the averaged F+ and F-) and anomalously-scattering derivative (F+, F-) where the f' value is zero and the f" value is non-zero. The heavy-atom sites are found using the anomalous differences and phasing uses just the anomalous differences. Scattering factors are not refined. SAD phases require solvent flattening or other density modification before they are useful. RESOLVE works very well for SAD data.

MIR structure determination

MIR structure determination is almost the same as for MAD structure determination. The program scales the data using localscaling in SCALE_NATIVE and SCALE_MIR , calculates difference Patterson maps, and iteratively builds up and scores potential heavy-atom solutions for the MAD dataset with the SOLVE routine.

Combinations of multiple MAD or MIR datasets

SOLVE can treat multiple MAD or MIR datasets by converting them into one super-dataset that uses each native dataset other than the first one as a pseudo- derivative with no heavy atom. Then it uses correlated bayesian phasing to take into account any non-isomorphisms among the different datasets. In this way you can combine several MAD datasets, several MIR datasets, or MAD and MIR datasets. See the command COMBINE.

Datasets with anomalously-scattering native

SOLVE treats native datasets with anomalously-scattering atoms as two datasets. You enter the same data twice, once as a "native", and again as a "derivative" for which you specify a new heavy atom type with a zero real part and a non-zero imaginary part.

Data formats for automated structure determination with SOLVE

Your choices about input data include:

- Premerged or unmerged data?
- One data file or more per dataset?
- What data format? Intensities or amplitudes? (Scalepack, formatted, CCP4 mtz, d*trek)
- An additional file with phases (i.e. from MR)?

These choices are discussed below in more detail. See also the SAMPLE SCRIPTS.

Should you merge your data to the asymmetric unit before running SOLVE?

- SOLVE can read unmerged data or data merged to the asymmetric unit.
  - PREMERGED data is best if your data is already well scaled
  - UNMERGED data is best if your data has not been thoroughly scaled already

Can you input more than one data file for a native, derivative, or wavelength?

- For each native, derivative, or wavelength dataset, you can input one or more separate data files.
  - If a dataset has just one data file, just read in the datafile
  - If a dataset consists of several data files, just read them in one after another

What data format? Amplitudes or intensities?

- if you have DENZO/SCALEPACK output as your raw data...
  - ...and the data is NOT MERGED to the asymmetric unit, you will use the flags:
    - READDENZO
    - UNMERGED
    - READ_INTENSITIES
  - if the data is ALREADY MERGED to the asymmetric unit, substitute the flag:
    - PREMERGED

- if you have FREE-FORMAT intensities or amplitudes as your raw data...
  - ...and the data looks like: H K L I SIGMA, use the flags
    - READFORMATTED
    - UNMERGED

        - READ_INTENSITIES
    - if the data looks like: H K L I+ SIGMA+ I- SIGMA-, substitute the flag:
        - PREMERGED
    - if you have free-format F(hkl) instead of intensities:
        - substitute the flag READ_AMPLITUDES

- if you have a CCP4 MTZ file with amplitudes scaled and reduced to the asymmetric unit as your raw data...
    - You will have to make sure that this mtz file contains only the data you want and not lots of other columns of data
    - Note what you have called your data columns
    - The column names that SOLVE will want assigned are:
        - MAD data:
            - FPH1 (amplitude at wavelength 1)
            - SIGFPH1 (sigma of FPH1)
            - DPH1 (anomalous difference wavelength 1)
            - SIGDPH1 (sigma of DPH1)
            - FPH2 (etc for wavelength 2, 3 ...)
        - MIR data:
            - FP (amplitude for native)
            - SIGFP (sigma of FP)
            - FPH1 (amplitude for deriv 1)
            - SIGFPH1 (sigma of FPH1)
            - DPH1 (anomalous difference deriv 1)
            - SIGDPH1 (sigma of DPH1)
            - FPH2 (etc for derivs 2, 3 ...)
    - use the flags LABIN and HKLIN to tell SOLVE how to read your mtz file. You can use multiple LABIN statements if you can't fit it all on one line. A sample LABIN statement where native F is called FP and sigma is SIG and deriv F is called FHG and sig of deriv F is SIGHG and anom diff for deriv is called DELHG and its sigma is SIGDELHG and with an input file of input.mtz is:
        - LABIN FP=FP SIGFP=SIG FPH1=FHG SIGFPH1=SIGHG
        - LABIN DPH1=DELHG SIGDPH1=SIGDELHG
        - HKLIN input.mtz
        - NOTE: use uppercase letters (unless your column names are lowercase) because case matters here
    - SOLVE figures out if this is MIR or MAD data based on whether or not you define FP and SIGFP.
    - When SOLVE reads the HKLIN statement it will read in the file using the information in all previous LABIN statements. HKLIN can be specified only once in a solve run.
    - You do not need to input cell dimensions or space group if you use HKLIN. The values read from the mtz file are used unless you change them with a keyword after the HKLIN statement. SOLVE writes out a symmetry file in the local directory based on the symmetry

    information in the mtz file that you can use later if you wish. It is named with the space group name.
  - NOTE: remove the SCALE_MAD command from your script file as your data is assumed to be scaled already

- if you have a set of CCP4 MTZ files with unmerged intensities (LABIN I=I SIGI=SIGI)use the flag:
    - READCCP4_UNMERGED !(instead of readdenzo or readformatted or readtrek)
    - Enter data file names just as for readdenzo or premerged
    - You may not specify a LABIN line with this option. Your mtz file must contain I and SIGI as the column labels.

- if you have a d*TREK file with intensities as your raw data...
  - use the flag READTREK (just one flag needed)

What if I have phases from molecular replacement?

- If you have an "mtz" file containing FC PHIC FOM then specify (myFC is your column name for FC, etc):
  - PHASES_LABIN FC=myFC PHIC=myPHIC FOM=myFOM
  - PHASES_MTZ xxxx.mtz
- If you have a formatted file with H K L FC PHIC FOM (one record per line; there can be text in between the numbers, such as in CNS or X-PLOR formatted files), then specify:
  - PHASES_FORMATTED xxxxx.fmt
- That's it. Put these lines somewhere in your input file before "SOLVE" and SOLVE will read in these phases and use them in initial difference Fouriers to find sites.

Contents	Index

Introduction to SOLVE

What SOLVE does

SOLVE is a program that can carry out all the steps of structure determination for MAD and MIR, starting with raw intensities (e.g., Scalepack output), and ending with phases and a NEWEZD electron density map that you can read right into "O". SOLVE can make all the decisions in this process and come up with a map all by itself if you want. On the other hand, if you know something about your system that SOLVE doesn't seem to realize, you can guide it through the process by editing script files that SOLVE writes out and continuing in the fashion that you choose.

To use SOLVE on your data, first read the rest of this introduction and the Getting Started section that follows. Then go right on to Automated structure determination to solve your structure.

To get an idea of what SOLVE does, you might want to have a look at some examples of MAD and MIR structure solutions using SOLVE.

Then you might want to try SOLVE out on some test data in your space group using the GENERATE feature. This allows you to create a MAD or MIR dataset with any heavy atom sites you want and then run SOLVE on it. If you start with a PDB coordinate file, you can generate a dataset, solve it, and use "O" to display the EZD electron density map that SOLVE creates along with the correct structure.

Getting Started

[Scripts | Useful commands | Symmetry files | Data files]

[Your license |SOLVEDIR, SOLVETMPDIR, CCP4_OPEN environmental variables| More help]

Scripts

This manual is set up to provide you with short scripts that you can edit to carry out MIR and MAD structure determinations and to carry out any of the other functions of SOLVE.

If you want, you can put the standard information for your dataset in to a file "solve.setup":

```
!------------------ solve.setup: standard setup for a dataset --------------
SYMFILE /usr/local/lib/solve/c2.sym     ! symmetry file for this space group
                                        !  (most common space groups should be in
                                        !   this directory)
CELL  76.08 27.97 42.08 90 103.2 90     ! a, b, c, alpha, beta, gamma
RESOLUTION 3 20                         ! Resolution limits in A.
!-----------------------------------------------------------------------------
```

Now each time you start up SOLVE, just type:@solve.setup and the program will read in this information. You can put any keywords or commands into script files that you can type at the terminal. You can also put this information right into your scripts for running SOLVE if you prefer.

Useful commands and information about running SOLVE

The "solve.status" file

SOLVE continuously updates a file called "solve.status" in your working directory. The tail end of the file shows the current status of your SOLVE job. You may find it convenient in unix to open a window, get in this directory, and type "tail -30f solve.status". This will continuously update your screen with the current status of SOLVE. In general, good datasets take a lot less time than poor ones because SOLVE recognizes a correct partial solution very early in a good dataset and can just follow it until all the sites are found.

The "solve.control" file

SOLVE will read a file called "solve.control" in your working directory while it is running. This file can be used to tell SOLVE to finish up as soon as practical if you think it has found the best solution already and you don't want it to keep looking any longer. Once SOLVE has read the file it writes the keyword IGNORE at the end of the file. This tells it not to read the file again (until you delete or comment out the keyword IGNORE). The most useful commands that can be read from this file are:

```
!-------- solve.control: control solve operation in real time-------------
FINISH          ! finish up cleanly as soon as possible (do not look for any new
solutions)
NTOPSOLVE  n    ! only consider n solutions at a time
NSEEDSOLVE  n   ! only use n seeds as starting points for SOLVE
NTOPDERIV n     ! only try out up to n peaks from a difference Fourier when adding
                             ! on new sites to a solution
```

Useful commands

Some commands that you might find useful right away are "Help", "?" "@" and "History".

HELP will print out a list of available commands.

? [keyword or command]

?ALL will print out a list of all keywords applying to each command and the keyword values.

? followed by a command will print out a list of all keywords that apply to that command and their values. Example: "? maps" will list all keywords that apply to "MAPS".

? followed by a keyword will print out the value of that keyword. Example: "?cell" will print out the current cell constants a, b, c, alpha, beta, gamma. This is useful for making sure that the values are what you think they are and for finding out what the defaults are.

@ filename

The "@" symbol is used to read in the contents of a file as commands. In the above example, "filename" will be opened and interpreted as keywords and commands just as if they were entered from the terminal.

HISTORY

This command will print out a list of the most recent commands and keywords input. The number listed is controlled by the keyword NLIST. Example:

```
nlist 100
history
```

will print out the 100 most recent commands and keywords.

Abbreviating commands

You can abbreviate commands as long as what you type is unique. That is, you can type "hist" and SOLVE will recognize this as "history". Any text appearing after a "!" or a "[" on a keyword or command line is ignored. This lets you enter comments in your script files. SOLVE reads commands one line at a time. This means any information for a keyword has to appear on the same line as the keyword. Also you can only enter one keyword per line. You can end the program by typing "stop".

Alias

If you don't like a command name SOLVE uses, you can create an alias of your choice. For example, if you tell SOLVE:

```
alias quit stop
```

then any time you type "quit", SOLVE will interpret this as "stop" (and it will stop). If you want to know what the current aliases are, type,

```
?alias
```

and SOLVE will list them for you.

Running more than one SOLVE job at a time.

You can run more than one SOLVE job on your computer at one time, but they must be in separate directories. This is because SOLVE writes out lots of temporary files with standard names and they would be mixed up if two jobs were running simultaneously in one directory.

Symmetry files

You need to tell SOLVE about your space group symmetry with a symmetry file using the information in the International Tables. The directory /usr/local/lib/solve (or else the directory named if you type "echo $SOLVEDIR") on your computer should contain the most common space groups as they are supplied with the installation of SOLVE.

Here is the symmetry file for space group C2 which you would find in "/usr/local/lib/solve/c2.sym":

```
4          !  Line 1 of the file is the number of symmetry elements to follow
x,y,z
-x,y,-z
x+1/2,y+1/2,z
-x+1/2,y+1/2,-z
```

Enter the identity "x,y,z" as the first operation always. For centered cells, enter the members of the centered sets in the same order for each centering operation.

NOTE: dashes (-) and underscores (_) in the space group file name have meanings. Don't confuse r3.sym with r-3.sym!

For space groups like P2/m that contain slashes, the space group file name has an underscore instead of the slash (p2_m.sym is P2/m). Similarly, for space groups like P1-bar, the file name for the space group has a dash (p-1.sym is p1-bar)

Data Files

The data files used by SOLVE are binary. To look at one, you can use the keywords "INFILE", "NLIST", and "VIEW". Here is a script to look at the file "data.drg":

```
!----------------------------------------------------------------------------
! script file to VIEW a .drg file
@solve.setup
INFILE data.drg
NLIST 20
VIEW
!----------------------------------------------------------------------------
```

If you want to convert SOLVE datafiles to another format, then you can use the IMPORT and EXPORT commands. See the "Importing and Exporting data" section for details on this and for details on converting binary maps to other formats.

If you want to look at all occurrences of the reflection "-1 3 6" and its symmetry equivalents, you can use the keyword "HKL_VIEW" with "NLIST -1":

```
!----------------------------------------------------------------------------
! script file to VIEW a reflection and sym equivalents in  .drg file
@solve.setup
INFILE data.drg
NLIST -1
HKL -1 3 6
VIEW
!----------------------------------------------------------------------------
```

For automated analysis of MAD and MIR data, you can start with Scalepack output or other formatted data. If you used mosflm and have an .mtz file with your data, you can read that right into SOLVE as well.

For other operations of SOLVE, you will need to IMPORT your data into "dorgbn-style" files. See the description of "IMPORT" later in this documentation. Missing data is represented by either "-1.0" or "0.0" in this package.

Your license

Users of SOLVE are issued an "solve2.access" license file that looks like:

```
License for SOLVE expiring 3-jan-1999
AX9481VA991
```

You should name your license file "solve2.access" and put it in the directory named in the environmental variable "SOLVEDIR" (see below). That is, if you type "echo $SOLVEDIR" and your computer responds "/usr/local/lib/solve" then your license file should be located in "/usr/local/lib/solve/solve2.access".

The SOLVEDIR, SOLVETMPDIR and CCP4_OPEN environmental variables

SOLVE expects to find the file "solve2.access" and your symmetry files (e.g., c2.sym) in your working directory or else in the directory named by the environmental variable "SOLVEDIR". If you put these files in the directory "/usr/local/lib/solve" then you should add a line something like the following to your .cshrc or .cshrc_custom login file (or the corresponding files if you use another shell):

```
setenv SOLVEDIR /usr/local/lib/solve
```

You should then be able to check that the variable is set by typing

```
echo $SOLVEDIR
```

and your computer should respond

```
/usr/local/lib/solve
```

By default, SOLVE writes temporary files in your working directory. If you are running on one machine and your directories are on another, this can slow SOLVE down. SOLVE now allows you to set a SOLVETMPDIR environmental variable that specifies where these files are written. You can set it to a directory on your local machine to speed up SOLVE. (If the directory does not exist, solve will stop.)

```
setenv SOLVETMPDIR /var/tmp
```

SOLVE now uses ccp4 library routines. To allow overwriting of existing files, you need to set the CCP4_OPEN environmental variable:

```
setenv CCP4_OPEN UNKNOWN
```

Getting more help

You can get more help with this program and report bugs by emailing the SOLVE/Resolve mailing list. You can also email the author at "terwilliger@LANL.gov". Before you do this, however, please check that you have the most recent version of the program and documentation as your bug may already have been corrected.

Answers to SOLVE Frequently-asked questions

Where can I find all the questions and answers that have been sent to the SOLVE/RESOLVE newsgroup?" ?
You can find them archived at Jiansheng Jiang's very nice site at http://asdp.bnl.gov. On that site, click the blue button "BBnML" (Bulletin Boards and Mailing Lists) and then click "solve" under the Archive column.

Where is SYMINFO?
SOLVE/RESOLVE versions 2.08 and higher use the CCP4 version 5.0 libraries. These require both SYMOP and SYMINFO to be defined. (They are both symmetry libraries). They can be defined (if you are using csh, for example) with:

```
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
```

What should I try if SOLVE cannot solve my MAD dataset?
Try the SAD script right away if using your MAD data fails. In many cases, the data for the wavelengths collected later are severely affected by crystal decay, and in these cases using just the first wavelength collected (hopefully at the peak wavelength) may work much better than using all the data.

What is the best way to tell if my data are good?
For MAD data , have a look at the correlation of anomalous differences. See the table in the beta-catenin dataset for example. For MIR data, check that your R-factors between derivative and native start out large at low resolution, get smaller, and then finally get bigger again (the last rise is due to the errors in measurement and indicate where to cut off).

For SOLVE version 2.01 and higher, you can look at the analysis of signal-to-noise in the data listed in solve.prt. SOLVE estimates the noise from the sigmas in the data and the signal from MAD anomalous or dispersive differences, and from SAD anomalous differences. Note: If the sigmas in the data are clearly overestimated (rms(sigma) > rms (difference) then SOLVE rescales all the sigmas so as to yield rms(sigma)=rms(difference) in the highest resolution shell.

Can I input sites that I already know into SOLVE? Yes, you can. You just put them right in under the correct wavelength or derivative, add the keywords "addsolve" or "analyze_solve" before the scaling command, and SOLVE will use them to find new sites (addsolve) or to refine and calculate phases (analyze_solve). See addsolve and analyze_solve instructions for examples.

Can I use a solution at low resolution to run SOLVE at high resolution? Yes, you can. The easiest way is with addsolve and analyze_solve.

What do "checksolve" and "comparisonfile" do? Checksolve tells SOLVE to compare all the solutions it gets with the one that you input. SOLVE finds the origin (and hand, if you do not have anomalous data) that best matches its trial solutions with the one you entered, and reports the solution relative to this origin and hand. Comparisonfile allows you to input an FFT that SOLVE has previously calculated (at the same resolution as SOLVE is working); in combination with checksolve, SOLVE will calculate the correlation coefficient of every map that it examines to the one you input. This is handy when you have used "generate" to create a dataset.

Will SOLVE give me the right hand for my structure? Usually if you have good anomalous differences and MAD data, then yes, SOLVE will give you the correct hand. For SAD data, it is common for SOLVE not to get the correct hand. Run RESOLVE or the RESOLVE_BUILD script. after SOLVE, and if it fails to build anything useful, rerun SOLVE with analyze_solve and use the heavy-atom sites located in "solve_inverse.xyz" which are just the inverse of the sites in your original SOLVE run. In very rare cases your anomalous differences might be reversed (due to incorrect analysis of data or detector hooked up backwards). In that case you can use "swap_ano" to reverse the signs of the differences.

How do I get a bigger version of SOLVE? The distribution comes with the regular sized SOLVE and solve_giant and solve_huge. Try these first. If you need even a bigger version, then email me at terwilliger@lanl.gov and I'll give you the source so you can compile a bigger version. You will need the CCP4 library file libccp4.a to compile SOLVE.

Do I need a new access file for a new version of SOLVE? No, the same access file is good for all versions from version 2.0 through 2.99. If you are upgrading from version 1 to version 2, yes you do need a new access file (and the new one goes in "solve2.access").

Why will SOLVE read my solve2.access file but not RESOLVE ? This is a bug triggered by not having a carriage return after the access code on the second line of solve2.access. Just put in the carriage return and it should work for RESOLVE too. Sorry!

Why do I have to set BMIN=0 for high-resolution SAD data or other high-resolution data? You just need to do this for SOLVE versions earlier than 2.03 (20-Sept-2002). The reason is that the default minimum B value for heavy-atom site is B=15. For high-resolution data, typically B values are 5-10 so this default is much too high. If you set BMIN=0 then SOLVE can properly refine these B values. For version 2.03 and higher, the default is 2.

Where do I get f' and f" scattering factors? The best place to get f' and f" values for your MAD experiment is from the beamline staff where you collected your data. They will usually have made careful measurements of these for standard settings on their beamline, so if you do a Se experiment, for example, their values should be very good. You can also measure X-ray fluorescence from your own crystal and use the Kramers-Kronig transformation to estimate these values with the same programs the beamline staff used for their standard cases.

SOLVE does use the f' and f" values and they are very important. The wavelength values are not used in any important ways by SOLVE

Where do I get scattering factors for atoms that SOLVE has not heard of? They are on pp. 500-501 of Volume C of the international tables. For example (Nb:)

```
NEWATOMTYPE NB
AVAL 17.6142 12.0144 4.04183 3.53346
BVAL  1.18865 11.7660 0.204785 69.7957
CVAL 3.75591
FPRIMV -.248
FPRPRV 2.48
```

Why are the figures of merit in the solve.status file not quite the same as the final values? The reason that the final phases look better for MAD data than the ones reported in the solve.status file is that SOLVE calculates phases at the very end using bayesian correlated mad phasing, which gives much better phases than the SIRAS-like phases used during the main part of the run (when the solve.status file is being written). The reason the full phasing is not used all the time is that it is very slow.

Should I use all my data, or just the good data? Though it would be nice to use all the data, it is far better to use just the good data. Unless your sigmas are perfect and the statistics were done perfectly, it is really hard to get rid of the interference caused by data containing noise and essentially no signal.

Will SOLVE use NCS? Regrettably, no.

Why should I use NO MERGE ORIGINAL INDEX in scalepack? You should use "no merge original index" in scalepack so that SOLVE can re-scale the data with local scaling. This flag tells scalepack to write out the place in reciprocal space that each reflection was measured. Then SOLVE can compare it to its neighbors in reciprocal space.

Can I compare Z-scores for SOLVE runs in different space groups? At different resolutions? No, Z-scores are relative and therefore cannot be compared for different space groups or resolutions.

Can I read in data in 2 different formats? Unfortunately not.

Can I convert solve files like mad_fbar.scl into mtz files? Yes, you can. You will need to use "export" to export the data to a flat file, then use the ccp4 routine f2mtz to import into mtz.

Can I look at my patterson maps? Yes, you can. SOLVE writes some of them out as ".ezd" files which you can read right into "O" or convert to anything else with "mapman". Others you can convert to ezd with "ffttoezd".

Why do I get an execution error with no output when I try to run SOLVE? On an SGI, if you run a version of solve that does not match your computer, you get an "exec error". Try a version of solve for a lower version of your machine (i.e., r5000 instead of r12000).

Why does SOLVE say CELL DIMENSION <1 OR > 1000 FOUND? This happens if you try to use a really huge unit cell that SOLVE didn't expect. You'll have to cut back on the resolution a bit if it happens.

Why does SOLVE say /sbin/loader: Fatal Error: set_program_attributes failed to set heap start? This is an error that your Compaq Alpha might give you if you don't have enough memory allocated to you. The solution is to add a line to your .cshrc file that just says: "unlimit". This tells the system to give you all available resources.

Why doesn't COMBINE_ALL work for me? For combine_all to work, you have to be sure and input two or more complete datasets, separated by "new_dataset".

Why can't I use COMBINE_ALL_DATA with mtz files? SOLVE won't let you read in more than one mtz file, unfortunately. Sorry about this limitation. This means you can't use COMBINE with mtz files. You would need to dump your mtz data into flat files and read it in with "readformatted" instead.

Why can't SOLVE find 2 sites that are close together? SOLVE won't let you find sites that are closer than a specified number of grid units. The distance depends on the grid size, which is typically 1/3 the resolution. The default ("ntol_site") is 8, or about 2 to 3x the resolution. You can decrease it if you want; in which case SOLVE will have to consider more solutions and may have trouble identifying the best.

What does it mean when SOLVE says "error in reading this file" when reading a scalepack file ? When SOLVE encounters "********" in a data file it will give you this error message. In scalepack (.sca) files this occurs if there is a reflection with a very large intensity that does not fit in the format of the file. One solution is just to edit the .sca file to remove these lines. Another is to re-run scalepack, specifying a scale factor to apply to all the intensities. You can do this with the keyword:

```
scale factor 10.0
```

in Scalepack.

On-line manual for RESOLVE

SOLVE/RESOLVE home page	SOLVE Table of Contents

RESOLVE Table of Contents

Licensing for commercial organizations

For commercial institutions, you need to contact our licensing office to obtain the forms. Please contact T. Allen Morris in the Technology Transfer Division at the address:

T. Allen Morris
Industrial Business Development Division Office, MS C334
Los Alamos National Laboratory
Los Alamos, NM 87545

Tel: 505-665-9597
Fax: 505-665-0154

email: tamorris@lanl.gov

Contents	Index

Index for SOLVE manual

Note: The links in this index will put you before the position of the word you are looking for. You may need to use FIND (control-F) on your browser to find the word itself.

Back to RESOLVE table of contents

Running RESOLVE

Summary

To run RESOLVE, you need:

- *SOLVE and RESOLVE installed on your computer*
- *the CCP4 suite installed on your computer (or at least the file "symop.lib" somewhere)*
- *A CCP4 mtz file with at least FP, PHI, FOM for your dataset*
- *An estimate of the solvent content of your crystal*
- *To set the CCP4 environmental variables for file control and symmetry (modifying as appropriate for the location of symop.lib on your system):*

```
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
setenv CCP4_OPEN UNKNOWN
```

The most basic RESOLVE script (see the sample scripts for more cases):

```
#!/bin/csh
#
# Here is a minimal script to run RESOLVE on MAD/MIR/SAD etc data:
#
# Set CCP4 variables for symmetry information and
# for file handling:
#
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
setenv CCP4_OPEN UNKNOWN
#
# Now run RESOLVE:
#
resolve<<EOD
hklin solve.mtz
```

```
LABIN FP=FP PHIB=PHIB FOM=FOM HLA=HLA HLB=HLB HLC=HLC HLD=HLD
hklout resolve.mtz
solvent_content 0.4             ! your solvent content goes here.
             ! Next line is the file with your protein sequence.
seq_file protein.seq
EOD
#
# Now "resolve.mtz" has the output amplitudes, phases,
# and figure of merit in columns labelled: FP PHIM FOMM.
# A model of your structure is in resolve.pdb.
#
```

- RESOLVE will read from "solve.mtz" and write out new structure factor amplitudes and phases.
- If you have a file "ha.pdb" in your directory, RESOLVE will try to use the PDB coordinates in it to identify and apply NCS.
- Additionally, Hendrickson-Lattman coefficients are written.
- A model of your structure is built and placed in resolve.pdb
- You can use "resolve.mtz" in the same way as you use any mtz file in the CCP4 suite.

Notes on input and output mtz files (see the sample scripts)

- **RESOLVE expects to read data from a CCP4 mtz file**
  - **at least FP, PHI, FOM are needed for MAD/MIR/SAD data**
  - **for prime-and-switch, you need FP FC PHIC FOM, with optional FWT (from sigmaa)**
  - **it works much better for MAD/MIR/SAD data if you also input Hendrickson-Lattman coefficients (HLA HLB HLC HLD)**
  - **you specify the columns with the LABIN keyword as in the CCP4 suite**
  - **(LABIN FP=yourFPcolumn PHIB=yourPHIBestcolumn FOM=yourFOMcolumn HLA=yourHLAcolumn etc)**
  - **If your input file has reflections with non-zero F within your resolution range, it will automatically attempt to phase all of them.**
  - **You can input a FreeR_flag column.**
- **RESOLVE will write a CCP4 mtz file**
  - **The output columns of data are H K L FP SIGFP PHIM FOMM HLAM HLBM HLCM HLDM FreeR_flag**
  - **FP is just what was input; SIGFP is 1.0 if not input. NOTE: the SIGFP behavior may not be what you expect: If you do not specify SIGFP in your labin statement, resolve will write out SIGFP = 1.0. If you specify SIGFP=something, the data in column "something" will be copied out to SIGFP in the output file. If you use the default labin (FP=FP SIGFP=SIGFP PHIB=PHIB FOM=FOM HLA=HLA HLB=HLB HLC=HLC HLD=HLD) it will copy it out correctly.**
  - **PHIM is the modified "best" or centroid phase**
  - **FOMM is the figure of merit of PHIM. Use in a map with F = FP * FOMIM * exp(i PHIM).**
  - **HLAM, HLBM, HLCM, HLDM are Hendrickson-Lattman coefficients for the final phases.**
  - **Reflections with FreeR_flag = 0 are used as the "free" set in RESOLVE. Normally all reflections are used for all steps except the identification of the solvent fraction and a database for density statistics (i.e., the FreeR_flag is ignored for most steps).**
- **RESOLVE will write out a PDB file with a model of your structure**
  - **The model will usually not be fully complete, depending on the quality of the map**
  - **Some loops won't be present in most cases**
  - **Side chains will be built on the better parts of the model if you specify a file with the protein sequence**
  - **Side chains not in density will be truncated to CB**

Keywords for resolve

```
KEYWORD         DEFAULT                 WHAT IT IS

access_file     solve2.access            Name of solve2.access file.  If it is not in
the
                                             /usr/local/lib/solve/ directory or in the
                                             current directory or in the directory
$SOLVEDIR
                                             then you will want to tell RESOLVE where
it is

hklin           solve.mtz               mtz file with input phases and phase
probabilities

hklout          resolve.mtz             mtz file with output phases

LABIN FP=FP PHIB=PHIB FOM=FOM HLA=HLA HLB=HLB HLC=HLC HLD=HLD
                                        LABIN statement identifying the columns of
                                        data in the hklin mtz file




mask_cycles     5                       Number of cycles in which masks are redone and
                                        images are compared to map

minor_cycles    10                      Number of minor cycles per mask_cycle


no_build                                Don't build a model

build_only                              Just build the model (no density modification)

assemble_only                           Just assemble the model and write it out
(using info in peak_file and fragment_file)

build_outside_model                              Just build the model outside the
region defined by model xxx.pdb

superquick_build                        Build the model as quickly as possible,
searching for fragments on coarse grid
                                        (works great for good maps and 10x faster than
version 2.03 model-building)
                                        NOTE: superquick_build/quick_build/
thorough_build are 3 choices for 1 parameter

quick_build                             (Default) Standard model-building protocol (3x
faster than version 2.03)

thorough_build                          Look exhaustively for ways to build the model.
Similar to version 2.03.
                                        Not always any better than "quick_build"
```

```
aggressive_build                        Build aggressively, allowing some incorrect
residues or even entirely incorrect residues.
                                        Same as "macro_cycles 3".  Not recommended.

conservative_build                      Build conservatively (same as version 2.03).
Same as "macro_cycles 1".



seq_file protein.seq                    File containing protein sequence
                                        Format: 1-letter code sequence of each chain,
separated
                                        by lines starting with &rt&rt&rt .  No need to
put in duplicate chains.

no_expand_ncs                           Don't expand number of copies of each chain
beyond what was found with ha sites

seq_prob_min 0.95                       Minimum confidence of sequence match to place
side chains on a fragment

noget_peaks                             Skip searching for helices/strands and use
data from peak_file

noget_fragments                         Skip searching for fragments and use data from
fragment_file

peak_file      resolve_peaks.dat        Intermediate file with locations of all
helices/strands considered

fragment_file  fragments.dat            Intermediate file with coordinates of all
fragments considered in model-building

fom_cut  0.15                           Set initial resolution for density
modification to be where the FOM is about .15

s_step   0.02                           Steps in s=1/d to take during phase extension



solvent_content 0.3                     Fraction of unit cell in solvent region (if
specified, do not search for optimal fraction)

use_input_solv                          Do not search for optimal solvent content
                                        Use value from solvent_content if specified
                                        Use value from resolve.solvent if it exists
and solvent_content is not specified.

resolution                              Resolution limits (default=whatever is in
input mtz file)

res_start   2.5                         Start out density modification at this
resolution, then extend to maximum

phase_extend                           Start out density modification at resolution
res_start (set automatically), then extend to maximum
```

```
no_phase_extend                          Start out density modification at final
resolution. This is default (changed in version 2.09)


phases_from_solve                       Input phases are not yet density-modified
(default if "hklstart" is not set)




phases_from_resolve                     Input phases are already density-modified
(default if "hklstart" is set)




use_free_for_test                       Use free set for solvent content/histogram
tests (default unless Nfree<500)




use_all_for_test                        Use all data for solvent content/histogram
tests




create_free                             Create FreeR_flag if it does not exist
(default)
                                        (This does not mean that the FreeR_flag is
used)




include_free                            Use all data for main cycles (default)




exclude_free                            Do not include free data for main cycles (does
not work well)

database  1                             Use database entry #1 for histograms of
protein/solvent density

use_input_db                            Do not search for optimal histogram from
database
                                        Use value from "database" if specified
                                        Use value from resolve.database if it exists
```

```
and "database" is not specified.




wang_radius_cycle  6. 4.              (default: variable start; 4 end) Starting and
ending radius (A) in Wang method for getting solvent mask
                                        NOTE:  RESOLVE will automatically set this for
you in a reasonable way if
                                        you don't specify it.

wang_radius_finish  4.                (default:  4.0) Ending radius (A) in Wang
method for getting solvent mask

wang_radius_start  6.                 (default:  variable) Starting radius (A) in
Wang method for getting solvent mask

wang_radius  6.                         Radius to be used for all cycles
                                        NOTE:  In RESOLVE you don't want or need a
small radius for getting the
                                        solvent probability.  It works better with a
medium-large radius (but really it
                                        makes very little difference what radius you
use)

hklstart                                mtz file with a starting set of phases.
RESOLVE will start with
                                        these phases (but use probabilities from
hklin).  Useful for
                                        running a few cycles, getting an output
resolve_1.mtz, then
                                        continuing on from there.  Goes with labstart.

labstart  FP=FP PHIB=PHIB FOM=FOM       LABIN statement for hklstart.  only PHIB and
FOM used.

hklperfect                              mtz file with a model set of phases
labperfect  FP=FP PHIB=PHIB FOM=FOM     LABIN statement for hklperfect
difference_map                          calculate difference map FP hklin-FP hklperfect
phase_with_perf                         use PHIB and FOM from hklperfect in diff map
(default is use PHIB and FOM from hklin)


cc_ratio      0.1                       Cut off prime-and-switch phasing at resolution
where CC
                                        of FP with FC < cc_ratio

verbose                                 print out a lot of data every cycle, not just
at first and last.

nohl                                    don't calculate HL coefficients at end of
resolve (saves time)

rota_matrix                             rotation matrix for NCS symmetry. Three
rota_matrix lines define a matrix:
                                        rEnter the identity as first molecule 1.
                                        rota_matrix 1 0 0
                                        rota_matrix 0 1 0
                                        rota_matrix 0 0 1
                                        The rotation matrix applies to orthogonal
Angstrom coordinates.
                                        The matrix and translation maps molecule j on
to molecule 1.
                                        This is what you get from the ccp4 program
lsqkab if molecule 1 is
                                        the reference (xyzin1) and molecule j is the
working molecule (xyzin2)

tran_orth                               Translation vector for NCS symmetry element,
in orthogonal Angstroms
                                        Enter after rota_matrix

center_orth                             Approximate center of mass of NCS symmetry
element, in orthogonal Angstroms
                                        Enter after rota_matrix. Required only for
element 1

fraction_ncs  0.15                      fraction of asymmetric unit occupied by 1
molecule (default = fraction protein / N )

invert                                  Invert NCS matrices (they correspond to
mapping molecule 1 on to molecule j)

fix_ncs                                 Do not refine NCS operators




ncs_restrict   n                        Only consider NCS if there are n operators.
ncs_restrict 3 looks only for trimers.

force_ncs                               Use input or heavy-atom-site NCS symmetry even
if there is very low correlation

rad_ncs_mask_max  6.                    Do not allow NCS mask to have radius greater
than rad

ha_file          ha.pdb                 Use the entries in this file (PDB format) to
look for non-crystallographic symmetry

compare_file    xxx.pdb                 Use coords in this file to define asymmetric
unit for output resolve_compare.pdb
                                        NOTE: has no effect on contents of resolve.pdb

no_find_ncs                             Don't try to find NCS from heavy-atom sites
even if the file "ha.pdb" exists

ncs_only                                Find NCS and do nothing else (no density
modification, nothing)

do_all_cycles                          Do all the assigned cycles, even if nothing is
happening

no_fill                                Don't fill in missing reflections (default)
(best usually)

fill                                   Fill in missing reflections (risky)

r_match                                (Default=1.0) Maximum distance between CA in
different fragments to link during assembly

r_min                                  (Default=1.) Lowest minimum rho/sigma for main-
chain atoms

r_min_side                             (Default=0.3) Lowest minimum rho/sigma for
side-chain atoms

r_overall                              (Default=0.75) Minimum rho_bar/avg for
starting a segment in model-building

r_end                                  (Default=0.5) Minimum rho_bar/avg for
continuing a segment in model-building

z_cut                                  (Default=0.5) Minimum Z-score to keep a
fragment after refinement

z_cut_extend                           (Default=-0.5) Minimum Z-score to keep a
fragment after extension

macro_cycles   3                       (Default = 2) Number of iterations of lowering
thresholds in model-building.
                                       NOTE: normally do not use more than 3 or the
model becomes poor.
                                       Each cycle beyond 1 r_min and r_min_side
decrease by 0.5 and
                                       z_cut and z_cut_extend decrease by 1.




no_unassigned                          Do not write out residues not assigned to
sequence in model-building

rad_mask      2.5                      Radius for calculation of solvent mask and for
inclusion in image-based phasing

image                                  Use map calculated from PDB file
("composite_pdb") or from input phases as
                                       target for electron density. Only density
within rad_mask of atoms used if from PDB file.
```

```
image_only                               Do not do solvent flattening/NCS etc. Just use
the image as a target. Requires
                                         the keyword "image" as well.   Produces phases
similar to "sigmaa".





prior_weight    1.0                      (Default=1.0) Weight on the input phases.
Prior_weight 0 is used in prime-and-switch phasing.





prime_and_switch                         Use prime-and-switch phasing. Use input phases
only to calculate initial map.
                                         Input phase probabilities not used at all.

no_erase_protein                         Do not let P(protein) be less than the
starting value in prime-and-switch phasing.


n_image_cycle                            Number of cycles of using the image (map)
based on a model in density modification

composite_pdb  xxx.pdb_                  root name of a set of PDB files to be used to
construct a composite image

composite_pdb_first 0                    first PDB file for composite is xxx.pdb_0

composite_pdb_last 20                    last PDB file for composite is xxx.pdb_20

pdb_in  refmac.pdb                       Name of PDB file to be used in starting model-
building

extend_only                              Just trim and extend the chains in pdb_in;
don't rebuild from scratch

no_merge_ncs_copies                      Do not merge NCS copies during extend (default
= merge_ncs_copies)

side_avg_min  0.0                        Truncate side-chain atoms in model-building if
mean density is < side_avg_min





LABIN FP=FP FC=FC PHIC=PHIC FOM=WCMB  FWT=FWT    LABIN statement suitable for prime-
and-switch using SIGMAA phases.
                                         FWT is optional; if present initial map
calculated with FWT exp(i PHIC).
```

```
pattern_phase                        Use image in cc_map_file as target for image-
based phasing (no prior phase information,
                                       no solvent flattening, no NCS.




cc_map_file   recovered_map.dat         binary file with map for pattern_phase
targeting (only read by RESOLVE)




coarse_grid                          Use coarse grid (same as RESOLVE 2.02) for
maps. Must match RESOLVE_PATTERN




n_restore       1                    Number of times to restart density
modification using mask
                                     from previous run but starting with original
phases. Default = 0 (no restarting)

no_restore                           Same as n_restore 0




scale_refl    0.5                    Weighting on map probability function (default
= 0.5)



scale_refl_start    0.05             Weighting on map probability function (default
= 0.5) on first cycle of density modification



scale_refl_end    0.5                Weighting on map probability function (default
= 0.5) on last cycle of density modification




trim                                 Trim pdb_in file back to match density
(default)
```

```
no_trim                                Take main-chain of pdb_in file as is and use
as a basis for model-building without trimming




no_cut_up_model                        Don't cut up pdb_in into little pieces
(default)




cut_up_model                           Cut pdb_in up into little and big pieces and
try them all as starting points for model-building




build_image                            Use FFT-based search to find helices/strands
and create an output map (dump.map)
                                             with reconstructed image of the map. You
can then use dump.map with "pattern_phase"




evaluate_model                           Compare the model defined by the model
keyword with the map calculated from hklin




model  resolve_best.pdb                            model to evaluate with
evaluate_model




ncs_mask_file    mask.mtz             write out a CCP4-style mtz file with FP PHIM
FOM that yield a map showing the NCS asymmetric unit.




protein_mask_file  mask.mtz           write out a CCP4-style mtz file with FP PHIM
FOM that yield a map showing the protein region.




add_mask             Require that the region defined by the PDB file read in with
```

```
model xxx.pdb is protein
                                              use rad_mask radius in definition of
region



no_ha                                  do not write heavy-atom sites out to the
resolve.pdb model file



start_segment     n                    read segments files starting with number "n"



max_segment       m                    read up to "m" segments files.  These are files
with information about helices/strands etc.



score_only                             score an electron density map (skew, ha_ncs,
correlation of map from map-probability phases with
                                              original map, correlation of local rms)



score_tert                             score tertiary structure



loop_only                             Fit a loop only. Requires pdb_in and extend_only



n_random_loop  20                           Number of loop conformations to try



loop_cc_min  0.4                           Minimum density correlation to keep a loop



rms_random_loop  0.3                           RMS random variation in loop
coordinates
```

```
no_sub_segments                             Fit entire segments to sequence (do not
break up)



omit_box  n                           Omit all points in omit box n from density
modification



n_box_target  m                             Try to set up m omit boxes



omit_boundary 2.0                             Increase the size of the omit region
defined by omit_box
                             in all directions by this amount



complete_omit                           Do not include solvent flattening or
histogram matching in omit region (default)



no_complete_omit                            Include solvent flattening and histogram
matching even in omit region



complete_omit_hist                            Include histogram matching even in omit
region
```

Keywords for RESOLVE ligand fitting

```
ligand_file      ligand.pdb            PDB file with N copies of a ligand in random
stereochemically ideal conformations. All atoms must be
                                              in the same order in all copies
```

```
n_ligand_pos      300                    Number of rotation/translation positions for a
ligand to consider



n_ligand_pos_ref      100                Number of rotation/translation positions for a
ligand to refine



n_group_search    3                      Number of large groups within ligand to search
for.
                                         If zero, groups from input file used without
rotation/translation to start fitting



group_search  xx                         Use group xx in ligand to start ligand fitting



n_keep_plac     100                      Number of placements of groups to keep in
ligand search



n_indiv_tries_min  20                    Minimum nmber of top placements of large groups
to try individually in ligand search



n_indiv_tries_max  20                    Maximum nmber of top placements of large groups
to try individually in ligand search



ligand_resno  20                        Residue # for ligand



n_template_atom  45                        Number of atoms in ligand (overrides guess
made by resolve)
```

```
no_local_search                       Search whole map for ligand



search_dist  10                      Search within this distance of top rms region in
map for local searches



search_center   3 10 5                     Local search is around this point (in A)



fit_phi_range   -20 20                     Try angles from -20 to 20 degrees relative
to the torsion angles in ligand_file



fit_phi_inc   20                      Sample angles in range fit_phi_range with
increment fit_phi_inc



delta_phi_ligand 50                    Rotation angle to sample overall placements of
large groups in ligand fitting



acceptable_offset 1.0                  Maximum allowed offset of atoms from
overlapping placements in ligand search (i.e., placements of
                                              an atom from each of 2 directions in a
cyclic molecule)



cut_close 1.5                          Minimum allowed separation between atoms in
separate rigid groups.
```

```
ignore_map                         Ignore the map and just generate a random
conformation.



model  xxx.pdb                         Generate FC from model and use Fo-Fc map to
fit ligand.
```

Keywords for resolve_pattern

```
KEYWORD         DEFAULT                 WHAT IT IS

access_file     solve2.access            Name of solve2.access file.  If it is not in
the
                                              /usr/local/lib/solve/ directory or in the
                                              current directory or in the directory
$SOLVEDIR
                                              then you will want to tell RESOLVE where
it is

hklin           solve.mtz               mtz file with input phases and phase
probabilities

hklout          resolve.mtz             mtz file with output phases

LABIN FP=FP PHIB=PHIB FOM=FOM HLA=HLA HLB=HLB HLC=HLC HLD=HLD
                                        LABIN statement identifying the columns of
                                        data in the hklin mtz file




resolution                              Resolution limits (default=whatever is in
input mtz file)

                                        Use value from resolve.database if it exists
and "database" is not specified.




recover_image                           Analyze map calculated with hklin phases and
amplitudes for local patterns;
                                              write out binary map with the recovered
image to cc_map_file
```

```
cc_map_file  recovered_map.dat          File with recovered map (binary file; only
read by RESOLVE)



coarse_grid                             Use coarse grid (same as RESOLVE 2.02) for
maps. Must match RESOLVE



path_patterns  $SOLVEDIR/patterns/      Location of library files for RESOLVE_PATTERN
```

```
#!/bin/csh
set main_version = 2
set minor_version = 08
set edit_version = 07
echo " "
echo "RESOLVE difference map script version $main_version.$minor_version.$edit_version "
echo " "
echo "Date: `date`"
echo "Working directory: `pwd`"
echo " "
#
#                RESOLVE difference map script
#                  05-Jan-2005
#
#                 Tom Terwilliger
#           Los Alamos National Laboratory
#
#
#    Input:  FP, model
#
#    Methods:  Just calculate a difference map using phases from model
#
#    Output: Map coefficients FP PHIM FOMM in resolve_diff.mtz
#          CCP4-style map file in resolve_diff.map
#
#=======================================================================
#     EDIT THE NEXT FEW SETS OF LINES TO MATCH YOUR DATA AND SYSTEM
#
#     PLEASE NOTE: each of the " = " below must have a SPACE on either
#       side (hklin = solve.mtz  NOT hklin=solve.mtz )
#
#=======================================================================
#
#  Anything special for your location (SOLVEDIR etc);
#
setenv SOLVEDIR /usr/local/lib/solve/
#
# location of resolve and resolve libraries;  place for large scratch files
#
set resolve =  /u1/terwill/resolve/work/resolve.linux
setenv TMPDIR .                          # large scratch files go here
#
set hklin = FOBS.mtz
set labin = 'FP=FP SIGFP=SIGFP'
set labin_cont = ''
#
```

```
#   input model
#
set pdb_in =  PARTIAL_MODEL.pdb
#
#   resolution
#
set dmin =  NONE #   NONE to use all data...high-res limit
set dmax =  NONE #   NONE to use all data...low-res limit
#
#=================================================================
#               Normally no need to edit below here...
#=================================================================
#=================================================================
#
set main_version_minimum = 2
set minor_version_minimum = 08
#
setenv SYMOP $SOLVEDIR/symop.lib
setenv SYMINFO $SOLVEDIR/syminfo.lib
setenv CCP4_OPEN UNKNOWN
unlimit
limit coredumpsize 0
#
set ip = 0
#-------------------------------------------------------------
#  check for all the library files we will need etc
#
if ( $resolve == NONE ) then
 echo "Sorry, you need to define resolve ..."
 exit
endif
foreach program ($resolve)
if ( $ip ) echo "Checking for the program $program ..."
if ( -f $program ) then
 if ( $ip ) echo "OK"
 goto ok
endif
if (  $program == NONE ) then
 if ( $ip ) echo "This program is not used"
 goto ok
endif
which $program >& tmp.dat
set test = `head -1 tmp.dat`
if ( $#test != 1 ) goto bad
if ( -f $test ) then
 if ( $ip ) echo "OK"
 goto ok
```

```
endif
bad:
 echo "Sorry, the program $program does not exist?"
 echo "Please check its definition in this script..."
 exit
endif
ok:
end
#
#
foreach lib_file ( symop.lib syminfo.lib)
#
if ( $ip ) echo "Checking for library file $SOLVEDIR/$lib_file..."
if ( ! -f $SOLVEDIR/$lib_file) then
 echo "Sorry, the library file $lib_file does not exist?"
 echo "Please check the definition of SOLVEDIR in your script: $SOLVEDIR"
 exit
endif
end
#-------------------------------------------------------------
#
# check for resolve
if ( -f resolve.ok ) rm resolve.ok
if ( -f resolve.version ) rm resolve.version
$resolve <<EOD  >& resolve.log
 quit
EOD
if ( ! -f resolve.ok ) then
 echo "Sorry please check the location of resolve..."
 echo "    is it really $resolve?"
 exit
endif
#  make sure this sort of worked...
set test_ok = `cat resolve.ok`
set test_ok_mem = $test_ok[$#test_ok-$#test_ok]
if ( $test_ok_mem != "ALLOCATED" ) then
 echo "Sorry, resolve was not able to run properly at all"
 echo "The end of the resolve log file says..."
 echo "-------------------------------------------------------"
 tail -12 resolve.log
 echo "-------------------------------------------------------"
 exit
endif
if ( ! -f resolve.version ) then
 echo "Sorry, this script requires version $main_version_minimum.$minor_version_minimum or higher of resolve"
 exit
endif
```

```
@ version = `cat resolve.version|head -1`
set version_minimum = "$main_version_minimum$minor_version_minimum"
if ( $version < $version_minimum )then
 echo "Sorry, this script requires version $main_version_minimum.$minor_version_minimum or higher of resolve"
 exit
else
 if ( $ip ) echo "Running version $version of resolve"
endif
#
# make sure we have these files...
#
if ( -f $hklin ) then
   echo "Data are in the mtz file $hklin"
else
   echo "Sorry, cannot find the mtz file $hklin"
   exit
endif
echo "LABIN information: $labin $labin_cont"
echo " "
#
if ( ( $pdb_in != NONE ) && ( ! -f $pdb_in ) ) then
 echo "Sorry, cannot find your pdb_in file $pdb_in"
 exit
endif
#
#-------------------------------------------------------------
#  figure out if this machine uses grep -a or just grep for text files:
echo "A" > test_a.dat
 set test_grep = `grep -a "a" test_a.dat >& tmp.dat`
if ( $status ) then
#  there was an error...do not use grep -a
 set grep_type = "grep"
else
 set grep_type = "grep -a"
endif
set test_grep = `$grep_type "A" test_a.dat`
if ( $#test_grep != 1 ) then
 echo "Sorry, unable to set the grep command on this system...giving up"
 exit
endif
rm test_a.dat
#-------------------------------------------------------------

#
@ grep_phib = `echo $labin $labin_cont | $grep_type 'PHIB='|wc -m`
@ grep_fp = `echo $labin $labin_cont | $grep_type 'FP='|wc -m`
#
```

```
if ( $grep_fp == 0 ) then
 echo "Sorry, FP is always required in the labin line..."
 exit
endif
if (  $pdb_in == NONE ) then
 echo "Sorry you need a pdb_in file"
 exit
endif
#
if ( $dmin == NONE || $dmax == NONE )then
  set resolution_line = ""
else
  set resolution_line = "resolution $dmin $dmax"
endif
echo " "
echo "${resolution_line} "
echo " "
#
set blank = " "
#
#          SETUP:
#
# ===================================================================
# ===================================================================
# hklin_image will be our combined phase information
#
set hklin_image = start_image.mtz
set labin_image = 'FP=FP PHIB=PHIM FOM=FOMM HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM'
set labin_image_cont = 'FreeR_flag=FreeR_flag'
#
set hklin_exp = exp.mtz
set labin_exp = 'FP=FP PHIB=PHIM FOM=FOMM HLA=HLAM HLB=HLBM HLC=HLCM HLD=HLDM'
set labin_exp_cont = 'FreeR_flag=FreeR_flag'
#
echo "Copying information from $hklin to $hklin_exp"
${resolve}<<EOD >>resolve.log
! Copying over $hklin to $hklin_exp
hklin $hklin
labin $labin
labin $labin_cont
hklout $hklin_exp
$resolution_line
mask_cycles 1
minor_cycles 0
no_build
EOD
#
```

```
#  make sure this sort of worked...
if ( `cat resolve.ok` != 'OK' ) then
 echo "Sorry, resolve was not able to finish even the first cycle"
 echo "The end of the resolve log file says..."
 echo "------------------------------------------------------"
 tail -10 resolve.log
 echo "------------------------------------------------------"
 exit
endif

#
#
set pdb_start = $pdb_in            #   starting PDB file
cp $pdb_in refine.pdb_0
#
echo " "
#
${resolve}<<EOD >>resolve.log
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
$resolution_line
fcalc $pdb_in
hklout fcalc.mtz
EOD
#
#  make sure this sort of worked...
if ( `cat resolve.ok` != 'OK' ) then
 echo "Sorry, resolve was not able to finish on the first cycle"
 echo "The end of the resolve log file says..."
 echo "------------------------------------------------------"
 tail -10 resolve.log
 echo "------------------------------------------------------"
 exit
endif
#
$resolve<<EOD>>resolve.log
hklin $hklin_exp
labin $labin_exp
labin $labin_exp_cont
hklperfect fcalc.mtz
labperfect FP=FP PHIB=PHIM
difference_map
phase_with_perf
hklout resolve_diff.mtz
EOD
#
```

```
if ( -f resolve_map.mtz ) then
$resolve<<EOD>>resolve.log
hklin resolve_diff.mtz
labin FP=FP PHIB=PHIM FOM=FOMM
mask_cycles 1
minor_cycles 0
no_build
ccp4_map_file resolve_diff.map
EOD
 echo "resolve_diff.mtz  ... Coefficients for difference map only FP PHIM FOMM"
 echo "resolve_diff.map  ... ccp4-style map from resolve_diff.mtz"
endif
#=============================================================================
```

How to interpret the solve.status file

The solve.status file keeps a running summary of the status of SOLVE. As additional steps are carried out by SOLVE, the results are added to the end of the solve.status file. It is convenient to look at this file continuously using the unix command "tail -30f solve.status" which will show the last 30 lines in the file, updating them every few seconds.

The solve.status file will show you:

- The title of the experiment if you have input one
- The step that SOLVE is currently on (e.g., SCALE_MAD, SOLVE)
- The time SOLVE has spent on this step
- If the program has begun automated structure determination (SOLVE) then the solve.status file will show statistics on the best solution found so far:
    - The figure of merit (m) and overall Z-score
    - The coordinates, occupancies and overall B-factors for each atom in the solution
    - The peak heights (normalized to the rms of the maps) of each heavy atom site in cross-validation difference Fouriers
    - The current resolution that SOLVE is working at
- The total time that SOLVE has been running
- If SOLVE is unable to carry out a command (e.g., an input file is missing) then the bottom of the solve.status file will tell you where to look for more detailed information about what happened (usually it will tell you to look at the end of the logfile).

The solve.status file can give you a pretty good indication of whether SOLVE has found a good solution. A correct solution will have high values of peak heights for all the heavy atom sites in the cross-validation difference Fouriers (for most good solutions these peak heights will all be more than 5 or 10 sigma). A good solution will have a high Z-score (typically above 20; this score is higher for structures with more heavy atoms). A good solution will have a figure of merit around 0.6 to 0.7 as well.

See How to interpret the output from SOLVE as well for information on this.

Commands for controlling how SOLVE works in automated mode

This is a list of the most commonly-used keywords for SOLVE that apply to SOLVE operation in automated mode. Also see the list of all SOLVE keywords and the list of all commands .

```
These keywords are usually in your solve.setup file:

SYMFILE xxxxx       symmetry file for this space group
CELL    a b c alpha beta gamma
RESOLUTION dmin dmax

RES_PHASE  xx        Carry out phasing heavy-atom searches at resolution XX
                     (but write out all data in resolution range set by the
                      keyword "resolution"). Usually you should let SOLVE set this.

FFTGRID  xs xe xtotal ys ye ytotal zs ze ztotal    grid for FFT calculations
PATTGRID xs xe xtotal ys ye ytotal zs ze ztotal    grid for Patterson
EZDGRID  xs xe    ys ye    zs ze                   grid for NEWEZD map

These keywords are usually in your main command file:

VERBOSE                         write out a lot of output to logfile
RAWMADFILE  xxx.int   File xxx.int will be read in as data for
                      the current mad wavelength
RAWNATIVEFILE  xxx.int   File &xxx.int& will be read in as data for
                         the native
RAWDERIVFILE  xxx.int   File xxx.int will be read in as data for
                        the current derivative
READ_INTENSITITES  (default)  The raw data files contain intensity measurements
READ_AMPLITUDES    The raw data files contain amplitudes (F) not intensities (I)
                   (This is valid only with READFORMATTED)

PREMERGED          The data in all RAWMADFILEs have H K L and 4 other columns:
                   I+/F+, sigma, I-/F-, sigma
UNMERGED           The data in all RAWMADFILEs have H K L and 2 other columns:
                   I/F, sigma

READDENZO        All datafiles are written by Scalepack.  For unmerged data
                 they will be read with the formatting:(6i4,i6,2i2,i3,2f8.0) and
                 nsym*2+1 lines are skipped at the top of the file. For
                 merged data the formatting is:  (3i4,4f8.0) and
                 3 lines are skipped at the top of the file.

READFORMATTED      All datafiles will be read with "*" formatting and
                   contain H K L I/F sigma or H K L I+/F+ sigma I-/F- sigma

READTREK          The datafiles were written by d*trek and contain columns
```

```
                   with intensities

LABIN              specify column assignments for HKLIN in standard CCP4
                   fashion (FC=FC1 PHIC=PHIC FOM=FOM) etc

HKLIN  xxx.mtz     mtz file containing scaled amplitudes

PHASES_FORMATTED xxx.fmt   File  xxx.fmt contains H K L FC PHIC FOM and
                     the phases and fom will be used in SOLVE with
                     difference Fouriers to find initial sites

PHASES_LABIN         specification of column assignments for PHASES_MTZ file
                     Normal use is FC=FC PHIC=PHIC FOM=FOM. NOTE: MUST
                     come before PHASES_MTZ!

PHASES_MTZ xxx.mtz     as PHASES_FORMATTED, but mtz-file. FC PHIC FOM required

NSKIP n             Skip exactly n lines at the top of each data file
NSKIP 0             Do not skip any lines at the top of each data file
NSKIP -1            Skip 0 lines at the top of each data file
                     unless the keywords READDENZO and PREMERGED
                     are set in which case the default number of lines
                     are skipped (see above)

RATMIN  xx           Minumum ratio of F/sig to read in data for a
                     reflection at all is xx [default=2.0].  This is
                     useful for eliminating weak data.

FPFM_ONLY           Toss all acentric reflections where either F+
                     or F- is missing [this is the default for MAD data]

FP_OR_FM            Use F+ or F- as an estimate of Fbar if F+ and
                     F- are not both present.  This is useful if
                     your data is not that complete.  It is much
                     better to obtain complete data however.

OVERALLSCALE        Do not do local scaling; just an overall
                     scale factor for F+, F- at each wavelength.
                     Use this if you already have scaled the data
                     and you don't want any more scaling done.



SWAP_ANO             Swap H K L -> -H -K -L as data are read in to
                     SOLVE in scale_native, scale_derivative, and
                     scale_mad. This is to correct for a detector or indexing
                     that swapped F+ for F-

The atom types recognized by SOLVE are:
H, H-1, He, Li, Li+1, Be, Be+2, B, C, Cv, N, O, O-1,
F, F-1, Ne, Na, Na+1, Mg, Mg+2, Al, Al+3, Si, Siv, Si+4,
P, S, Cl, Cl-1, Ar, K, K+1, Ca, Ca+2, Sc, Sc+3, Ti, Ti+2,
Ti+3, Ti+4, V, V+2, V+3, V+5, Cr, Cr+2, Cr+3, Mn, Mn+2, Mn+3,
Mn+4, Fe, Fe+2, Fe+3, Co, Co+2, Co+3, Ni, Ni+2, Ni+3, Cu,
Cu+1, Cu+2, Zn, Zn+2, Ga, Ga+3, Ge, Ge+4, As, Se, Br,
Br-1, Kr, Rb, Rb+1, Sr, Sr+2, Y, Y+3, Zr, Zr+4, Nb, Nb+3,
Nb+5, Mo, Mo+3, Mo+5, Mo+6, Tc, Ru, Ru+3, Ru+4, Rh, Rh+3,
Rh+4, Pd, Pd+2, Pd+4, Ag, Ag+1, Ag+2, Cd, Cd+2, In, In+3,
Sn, Sn+2, Sn+4, Sb, Sb+3, Sb+5, Te, I, I-1, Xe, Cs, Cs+1,
Ba, Ba+2, La, La+3, Ce, Ce+3, Ce+4, Pr, Pr+3, Pr+4, Nd,
Nd+3, Pm, Pm+3, Sm, Sm+3, Eu, Eu+2, Eu+3, Gd, Gd+3, Tb,
Tb+3, Dy, Dy+3, Ho, Ho+3, Er, Er+3, Tm, Tm+3, Yb, Yb+2,
Yb+3, Lu, Lu+3, Hf, Hf+4, Ta, Ta+5, W, W+6, Re, Os, Os+4,
Ir, Ir+3, Ir+4, Pt, Pt+2, Pt+4, Au, Au+1, Au+3, Hg, Hg+1,
Hg+2, Tl, Tl+1, Tl+3, Pb, Pb+2, Pb+4, Bi, Bi+3, Bi+5, Po,
At, Rn, Fr, Ra, Ra+2, Ac, Ac+3, Th, Th+4, Pa, U, U+3, U+4,
U+6, Np, Np+3, Np+4, Np+6, Pu, Pu+3, Pu+4, Pu+6, Am, Cm, Bk, Cf


newatomtype xxxx    define scattering properties of atom xxxx not recognized
                    by SOLVE
                    HINT: to get the aval, bval, cval values from Int tables
                    for atoms recognized by SOLVE, type mad_atom [atomname] and
                    SOLVE will list them for you.
aval a1 a2 a3 a4    4 real numbers (a1,a2,a3,a4)  from International Tables
for
                     the most recently defined newatomtype
bval b1 b2 b3 b4     b values for newatomtype
cval c              c value for newatomtype
fprimv xx           f' value for newatomtype
fprprv xx           f" value for newatomtype

                    For CLUSTERS, follow the instructions in
                    all_keywords.



mad_atom  xxxx   name of the anomalously scattering atom is xxxx.  If SOLVE does
                 not recognize this atom, first input its scattering parameters
                 with newatomtype
fprimv_mad       f' value for anomalously scattering atom
                 at this wavelength (must be input after each wavelength)
fprprv_mad       f" value for this wavelength

FIXSCATTFACTORS     Fix scattering factors at their input values.
                    This is a good idea if you have a reasonable
                    idea of the f' and f" values.  [this is the
                    default]

REFSCATTFACTORS     refine scattering factors f' and f".  If you
                    refine them, be sure to look at their new
                    values at the end of the routine MADMRG and
                    verify that they are reasonable.

derivative n        begin input of information for derivative/wavelength n
                    This command is used to start entering information on a
                    derivative.  If you want to modify something after you've
                    gone on to another derivative then you need to use the
                    command GOTODERIV
lambda n            identical to derivative n

nsolsite_deriv      Maximum number of sites for this derivative
                    (overrides nsolsite)

cutoff_deriv 200 3.5   resolution limits for this derivative/wavelength only

RES_PHASE 2.8   high-resolution limit for phasing only
SN_MIN    Set RES_PHASE so that signal-to-noise is bigger than this
SN_RATIO_MIN    Set RES_PHASE so that signal-to-noise is bigger than this
                    ratio times the value at low-resolution

INPHASE             include this wavelength/derivative in phasing.
NOINPHASE           do not this include this derivative/wavelength in phasing

INANO               include anomalous differences for this wavelength/deriv

noanorefine        use anomalous differences in phasing but not
                   refinement for this derivative.
                   (this is usually the best option for MIR
                   unless your anomalous differences are really
                   big, as from a synchrotron MIR dataset at an
                   absorption edge).  Note: you still have to specify
                   for each derivative "inano" to include anomalous
                   differences for that derivative.

anorefine          For this derivative with "inano" specified, use
                   anomalous differences in both refinement and phasing.
                   This is best for MAD data. (This is the default also).
                   Applies to current derivative/wavelength


ATOMNAME XXX      Start reading in information about a new site with atomname
                     "XXX" at the current wavelength/deriv.

OCCUPANCY xx       occupancy of current atom
BVALUE xx          B-value of current atom
XYZ  xx yy zz      fractional coordinates of current atom
REFINEALL          refine x,y,z,occ and b for this atom
REFINENONE         don't refine anything for this atom

Note: Flags for refinement of a heavy atom do not apply when the keyword
SOLVE is used (only with HEAVY)



PDB_XYZ_IN        PDB file with orthogonal A coordinates
                of all heavy-atoms for this derivative/wavelength
                NOTE: you cannot use XYZ/OCC/BVALUE/REFINEMENT parameters along with
pdb_xyz_in.
```

```
NRES   n            # of residues in asymmetric unit  [default=100]

NANOMALOUS n        # of anomalously scattering atoms in asymmetric unit.
                    Used to estimate how big the Fa values might be.
                    Also used to set max # of heavy atoms if nsolsite is not set

nobayes        do not use Bayesian correlated phasing in SOLVE.

ntopfour xx         Number of Fourier peaks to pick from a map

ntopderiv xx        Number of Fourier peaks to be tested for
                    inclusion in the model

nsolsite xx         Maximum number of sites in a derivative unless overridden by
                    nsolsite_deriv

nseedtest xx        Number of seeds per derivative to try (before
                    sorting)

nseedsolve xx       Number of seeds (total) to try after
                    sorting them

ntopsolve xx        Number of solutions to print out at the end
                    and number of solutions to keep track of at
                    any one time

addsolve           Add on to solution that is input
                    [default=off]
checksolve         Compare all solutions to input solution
                    [default=off]

analyze_solve      Analyze input solution without doing anything
                    else [default=off]

[no]delete          do [not] check out all solutions by testing
                    all one-site deletions [default=delete]

[no]inverse         do [not] check out all solutions by testing
                    their inverses (does not apply if a solution
                    is centrosymmetric or if anomalous differences
                    are not used). [default=inverse]

SCORING_TABLE (8 values)  Scoring table (usually generated by SOLVE) consisting
                      of mean and standard deviation of scores for trial
                      solutions for Pattersons, Cross-fouriers, Native Fourier
                      maps, and mean figure of merit.  This keyword is useful
                      when you are running SOLVE after modifying the script
                      file it writes out at the end.

QUICK                once a plausible solution is found, don't keep looking,
                     just add on sites to it and check it at the end. [default]
```

THOROUGH	keep looking anyways until a limit set by ntopsolve, nseedsolve, etc is reached.
NTOL_SITE	a site within ntol_site grid units of an existing site is considered to be a duplicate and is ignored. [default=8]
NTOL_SOLN	a heavy-atom solution for which every site matches another solution within ntol_soln grid units is considered to be a duplicate and is ignored. [default=2]
ACCEPTANCE xx	the weighting function for scoring patterson and free-difference fourier peak heights is adjusted so that a new site with height relative to the previous average height of ACCEPTANCE or higher will generally give a solution with a higher score than the solution without this site. [default =0.2]
SN_MIN xx	Identify working resolution as the point where signal-to-noise in the data goes down to about XX. Default =0.5
SN_RATIO_MIN xx	Identify working resolution as the point where signal-to-noise in the data goes down to about XX times its maximum value. Larger of value of S/N obtained by SN_MIN and SN_RATIO_MIN used. Default = 0.1
NO_SIM	Do not use Sim weighing with heavy-atom structure factors in SAD

```
# solve.com  -- take 1 mad dataset and solve it
# version that reads ccp4 mtz data file
#
setenv CCP4_OPEN UNKNOWN
solve <<EOD
logfile solve.logfile            ! write out most information to this file.
resolution 20  3.0               ! you need resolution.  Cell params read
                                 ! from the mtz file.

! get the data from ccp4 file mad_fbar-cad.mtz:
! NOTE:  do not put in FP=FP SIGFP=SIGFP because SOLVE will assume it is MIR!
! Also you can EITHER use labin/hklin  OR readccp4_unmerged but not both.

labin FPH1=FPH1 SIGFPH1=SIGFPH1 DPH1=DPH1 SIGDPH1=SIGDPH1
labin FPH2=FPH2 SIGFPH2=SIGFPH2 DPH2=DPH2 SIGDPH2=SIGDPH2
labin FPH3=FPH3 SIGFPH3=SIGFPH3 DPH3=DPH3 SIGDPH3=SIGDPH3
hklin mad_fbar-cad.mtz
! scaled data is all ready

mad_atom se
lambda 1
label set 1 with 2 se atoms, lambda 1
wavelength .9782                        ! wavelength value
fprimv_mad  -10                         ! f' value at this wavelength
fprprv_mad  3                           ! f doubleprime value
ATOMNAME Se                             ! we're about to enter data on an
atom

lambda 2
wavelength 0.977865
fprimv_mad  -7.5
fprprv_mad  5

lambda 3
wavelength 0.8856
fprimv_mad  -2
fprprv_mad  3.5
nres 80
nanomalous 2
!
analyze_mad
solve
!--------all done----------
EOD
```

```
#  command file to solve a 2-deriv MIR dataset using ccp4 mtz input
#  It looks just like one_mir_dataset.html except it has LABIN and HKLIN
#  and it has no rawnativefile or rawderivfile and no scale_native or
#  scale_derivative



# CCP4_OPEN environmental variable set to UNKNOWN so file overwriting will work
setenv CCP4_OPEN UNKNOWN

solve <<EOD
!   ccp4 mtz file input
!   solve a 2-deriv MIR dataset
logfile mir.logfile             ! write out most information to this file.
                                ! summary info will be written to solve.prt
resolution 20 3                 ! you need resolution.  space group and cell
                                ! dimensions read from mtz file.
!  get the mtz file information and read it in:
labin FP=FP SIGFP=SIGFP FPH1=FPH1 SIGFPH1=SIGFPH1 DPH1=DPH1 SIGDPH1=SIGDPH1
labin FPH2=FPH2 SIGFPH2=SIGFPH2 DPH2=DPH2 SIGDPH2=SIGDPH2
hklin mir_fbar-cad.mtz
!  now we're ready with scaled data

derivative 1                    ! about to enter information on derivative #1
label deriv 1 HG                ! a label for this deriv
atomname hg
inano                           ! You need to tell it if anomalous diffs are
                                ! to be used
noanorefine                      ! use anomalous differences in phasing
                                 ! but not refinement (best option for MIR)
nsolsite_deriv 2                 ! max 2 sites this deriv

derivative 2
label deriv 2 also hg
atomname hg
inano
noanorefine                      ! use anomalous differences in phasing
                                 ! but not refinement (best option for MIR)

acceptance 0.35                  ! accept a new site if it has a
                                 ! peak height about 1/3 of avg or more
nsolsite 2                       ! number of sites per deriv
                                 ! (use nsolsite_deriv to set individual
values)
ANALYZE_MIR                      ! analyze this mir data and set up for SOLVE
SOLVE
EOD
```

What to do next after you've run SOLVE

After you have run SOLVE you should first:

- Have a look at how to interpret your output

If SOLVE successfully solved your structure you might:

- look at your solve.ezd map directly with Alwyn Jones' "O" program (www.imsb.au.dk/~mok/o/) or your equivalent solve.ccp4_map with CCP4.\
- go on to density modification and automated model-building with RESOLVE
- Check out how to export your output to other packages

If SOLVE got most of the heavy atom solution you can:

- analyze variations on your solution with ANALYZE_SOLVE
- try and add more sites using ADDSOLVE

and if SOLVE was not able to find a good solution you should:

- See if you should rerun SOLVE with different control parameters
- If you have a MAD dataset...try SOLVE with just the peak wavelength and the SAD script.

ANALYZE_SOLVE: Analyzing a known solution with SOLVE
ADDSOLVE: Getting more heavy atom sites if you have a partial solution.

You can use ANALYZE_SOLVE or ADDSOLVE to input some sites you have obtained already. Here is the easiest way to do it:

- In your solve.com file, under "lambda 1" or "derivative 1" specify
  - atomname xx ! xx is your atom
  - xyz x y z ! coordinates of this atom
  - .. more xyz values
- anywhere ***before*** ANALYZE_MAD or ANALYZE_MIR or SAD specify:
  - ANALYZE_SOLVE (refine input sites) or ADDSOLVE (start from input sites)
- SOLVE will refine these sites and either just phase with them (ANALYZE_SOLVE) or go on

and find more sites (ADDSOLVE)
- See the sample solve.com file for ANALYZE_SOLVE and the one for ADDSOLVE for examples.

There is another way to do the same thing that is more complicated, but does not require re-running scaling and analysis. If you have already run SOLVE, you can also use SOLVE to analyze a MAD or MIR solution that you have already obtained. Start with your solve_mad.script file or solve_mir.script file that was written out during automated SOLVE operation by the routine ANALYZE_MAD or ANALYZE_MIR and:

- edit it to include the heavy atom sites that you want to check. For MAD data, you only need to specify the heavy atom sites for the one wavelength designated as the standard wavelength (this is indicated in your solve_mad.script file by the lines "This is the "standard" wavelength from MADMRG" and "Put your heavy atom parameters here"), they are copied automatically to the others. For more info on the keywords in the solve_mad.script and solve_mir.script files see the writeups for solve_mad.script (MAD data) or solve_mir.script (MIR data)
- copy the 1-line scoring table from your phases-hl.script output file into your solve_mad.script or solve_mir.script file. The keyword is scoring_table.
- Then specify the following keywords:

```
ANALYZE_SOLVE
SOLVE
```

- then move all your ouput files from previous runs to somewhere safe so that SOLVE does not overwrite them
- then run solve using the edited solve_mad.script or solve_mir.script file as input

The program will read in these heavy atom parameters and will refine them. It will do a phasing calculation and will analyze the derivatives relative to the pattersons and with cross-difference Fouriers. This analysis is the same as you would get at the end of SOLVE ordinarily. (SOLVE will also write out the files "phases-hl.export; .drg; .script as it does when running originally. Be sure you move your original ones out of the way or use the keywords "EXPORTFILE", "PHASEFILE", and "NEWSCRIPTFILE" to define new filenames for these output files so SOLVE does not overwrite them.)

SOLVE can be used to generate additional sites if you have a partial heavy atom solution. You do this in exactly the same way as described above for ANALYZE_SOLVE except you substitute ADDSOLVE for ANALYZE_SOLVE. Then run SOLVE. It will read in your solution and generate other solutions that are based on it and then it will analyze them for you and write out the best ones. This is a good thing to do once SOLVE has found a good, but not complete solution for you. You may need to increase the number of sites allowed in a solution (NSOLSITE) to allow more sites to be found. You may also want to reduce the value of "ACCEPTANCE" so as to include weak sites.

If you use ADDSOLVE, then you might also wish to specify the keyword "CHECKSOLVE". If you do this, then the analysis routine will compare each solution you obtain to the one you input at the beginning. This way you can see how things are going.

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
#
#  command file to use addsolve on a SAD dataset
#
#
solve<<EOD > solve.log
logfile solve.logfile

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym


readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors

addsolve        ! look for more sites then refine and phase

mad_atom se

lambda 1
label SAD data for Pt
rawmadfile lam1.intensities
fprprv_mad 5.0
xyz   0.9770      0.2439      1.1071   ! put sites here
xyz   1.1877      0.2625      0.9833
xyz   1.1669      0.2943      0.9205



nsolsite_deriv 4              ! 4 atoms max
SAD                          ! solve an SAD dataset
EOD
#
#
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
#
#  command file to use analyze_solve on a SAD dataset
#
#
solve<<EOD > solve.log
logfile solve.logfile

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym


readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors

analyze_solve        ! do not look for sites...just refine and phase

mad_atom se

lambda 1
label SAD data for Pt
rawmadfile lam1.intensities
fprprv_mad 5.0
xyz   0.9770      0.2439      1.1071   ! put sites here
xyz   1.1877      0.2625      0.9833
xyz   1.1669      0.2943      0.9205

refinenone     !  put this and the next line here to fix xyz
refineoccb     !



nsolsite_deriv 4              ! 4 atoms max
SAD                          ! solve an SAD dataset
EOD
#
```

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
#
#  generate SAS dataset (ano scatterer in native)
#
solve  <<EOD

! generate Se SAD dataset

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym

logfile generate.logfile
solvefile generate.prt
coordinatefile coords.pdb
percent_error  0.5
iranseed 31933

mad_atom Pt
lambda 1
label 0.9797 A Se SAD dataset
wavelength 0.9797
fprimv_mad  -8              ! f' value at this wavelength
fprprv_mad  10
ATOMNAME Pt
xyz  0.44 0.16 0.38
occ 1.0
bvalue 20
ATOMNAME Pt
xyz  0.14 0.36 0.28
occ 1.0
bvalue 20
generate_mad
! now lam_1.intensities = SAS intensity data at 0.9797 A
end
EOD
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
#
#  command file to solve an SAD dataset (Pt 1-wavelength)
#
#
solve<<EOD > solve.log
logfile solve.logfile

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym


readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors

mad_atom se

lambda 1
label SAD data for Pt
rawmadfile lam1.intensities
fprprv_mad 5.0

nsolsite_deriv 2              ! 2 atoms max
SAD                           ! solve an SAD dataset
```

```
EOD
#
#  Now do density modification and build a model:
resolve<<EOD>resolve.log
!solvent_content 0.40
seq_file seq.dat        ! sequence file
EOD
#  All done; your output phases are in resolve.mtz and
#   your model is in resolve.pdb
#
```

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
#
#
#  generate 1 mad dataset
#
solve <<EOD

!  GENERATE 1 MAD DATASET:

@solve.setup
percent_error 0.5
coordinatefile coords.pdb
iranseed -199753
logfile generate.logfile
solvefile generate.prt

mad_atom se                             ! define the scattering factors...
!
lambda 1
label set 1 with 2 se atoms, lambda 1
wavelength 0.9782            ! wavelength value
fprimv_mad  -10              ! f' value at this wavelength
fprprv_mad  3
ATOMNAME Se
xyz  0.44 0.16 0.38
occ 1.5
bvalue 20.
ATOMNAME Se
xyz  0.23 0.45 0.165
occ 1.5
bvalue 20.
!
lambda 2
wavelength 0.977865
fprimv_mad  -7.5
fprprv_mad  5
```

```
lambda 3
wavelength 0.8856
fprimv_mad  -2
fprprv_mad  3.5
!
GENERATE_MAD                        ! generate the MAD dataset now.
!
EOD
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
#   solve a MAD dataset
#
solve<<EOD > solve.log
logfile solve.logfile

resolution 20 3.0
cell 60 60 50 90 90 90
symfile /usr/local/lib/solve/p21212.sym



readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
refscattfactors      ! fixscattfactors/refscattfactors

mad_atom se     ! the anomalously-scattering atom
lambda 1
label set 1 with 2 se atoms, lambda 1
wavelength .9782                     ! wavelength value
fprimv_mad  -9                      ! f' value at this wavelength
fprprv_mad  3.5                     ! f doubleprime value
rawmadfile lam1.intensities            ! datafile for this wavelength
lambda 2
wavelength 0.977865
fprimv_mad  -8
fprprv_mad  4.5
```

```
rawmadfile lam2.intensities

lambda 3
wavelength 0.8856
fprimv_mad  -2.5
fprprv_mad  3.0
rawmadfile lam3.intensities
nres 80                    ! number of residues in au
nanomalous 2               ! number of anomalously-scattering atoms in au
!
scale_mad
analyze_mad
solve
EOD
#
# Now run Resolve to do density modification and build a model
#
resolve << EOD > resolve.log
!solvent_content 0.70       !    solvent fraction
seq_file seq.dat            !    sequence file
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#  and resolve.pdb has your model
#
```

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
#
#  command file to generate a 2-deriv MIR dataset
#
solve << EOD
resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym

coordinatefile coords.pdb
percent_error 10
iranseed -124093
OVERWRITE                              ! overwrite duplicate file names without
                                 !     asking
logfile generate.logfile
solvefile generate.prt
!
deriv 1                            ! enter parameters for deriv 1
!
inano                              ! use anom diffs
atom hg
occ 0.8
bvalue 40.
xyz -0.620932 -0.0765346 -0.637333
atom hg
occ 0.4
bvalue 25.
xyz -0.315098 -0.512727 -0.664318
!
deriv 2
inano
atom hg
occ  0.8
bvalue 30.
xyz 0.620932 0.5765346 0.337333
atom hg
occ 0.6
```

```
bvalue 25.
xyz 0.515098 0.212727 0.364318

GENERATE_MIR                    ! generate the MIR dataset now.
!
EOD
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
#  solve.com   solve an MIR problem
#
solve<<EOD > solve.log
logfile solve.logfile

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym



readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors

rawnativefile native.intensities ! native data H K L Iobs Sigma usually
                         ! NOTE: all datafiles must be in the same format

derivative 1                   ! about to enter information on derivative #1
label deriv 1 HG               ! a label for this deriv
atom hg

rawderivfile der1.intensities  !  derivative data
inano             ! You need to tell it if anomalous diffs are to be used
noanorefine       ! use anomalous differences in phasing
                  ! but not refinement (best option for MIR)
```

```
nsolsite_deriv 2         ! max 2 sites this deriv

derivative 2
label deriv 2 also hg
atom hg

inano
noanorefine                  ! use anomalous differences in phasing
                             ! but not refinement (best option for MIR)
rawderivfile der2.intensities        ! the derivative data is in this file



acceptance 0.35               ! accept a new site if it has a
                             ! peak height about 1/3 of avg or more
nsolsite 2                   ! number of sites per deriv
                             ! (use nsolsite_deriv to set individual values)
SCALE_NATIVE                 ! scale the native dataset
SCALE_MIR                    ! scale the derivs to the native
ANALYZE_MIR                  ! analyze this mir data and set up for SOLVE
SOLVE
EOD
#
# Now run Resolve to do density modification and build a model
#
resolve << EOD > resolve.log
!solvent_content 0.40        !    solvent fraction
seq_file seq.dat            !   sequence file
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#  and resolve.pdb has your model
#
```

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
#
#  generate MIR dataset with ano scatterer in native
#
solve  <<EOD

! generate fe-native + MIR dataset

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym

logfile generate.logfile
solvefile generate.prt
coordinatefile coords.pdb
percent_error  1
iranseed 31933

! native dataset treated as 1-wavelength Fe data
!
mad_atom fe
lambda 1
label 1.5418 A Fe-native dataset
wavelength 1.5418
fprimv_mad  -1              ! f' value at this wavelength
fprprv_mad  3
ATOMNAME fe
xyz  0.44 0.16 0.38
occ 1.0
bvalue 20
generate_mad
! now lam_1.intensities = native intensity data at 1.5418 A
end
EOD

solve  <<EOD
```

```
! generate fe-native + MIR dataset
@solve.setup
logfile generate_mir.logfile
solvefile generate_mir.prt
coordinatefile coords.pdb
percent_error 4
iranseed 184231

deriv 1
inano
atom hg
occ 1.0
bvalue 31.
xyz 0.15 0.25 0.35

deriv 2
inano
atomname hg
occ 1.0
bvalue 25
xyz 0.51 0.31 0.145
GENERATE_MIR                    ! generate the MIR dataset now.
!  now der1.intensities, der2.intensities have deriv data
!  native.intensities has native data WITHOUT anom data (don't use)
EOD
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
#  command file to solve a 2-deriv MIR dataset + anom diffs in native
#
solve<<EOD > solve.log
logfile solve.logfile

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym



readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors

rawnativefile lam1.intensities ! native data

! now read in native data again as a pseudo-deriv 1:
! put in scattering factors for pseudo-atom fel1 with only ano
! scattering

newatomtype FEL1
aval 0 0 0 0
bval 0 0 0 0
cval 0.001
fprimv 0
```

```
fprprv 3

derivative 1
label native data with ano diffs, treated as deriv 1
nsolsite_deriv 1            ! just 1 site
rawderivfile lam1.intensities
inano
anoonly                     ! no isomorphous diffs, just ano
atomname fel1               ! use our pseudo-atom with no iso scattering

! now read in the regular 2 derivs:
derivative 2                ! about to enter information on derivative #1
label deriv 1 HG            ! a label for this deriv
nsolsite_deriv 1            ! one site in this deriv
rawderivfile der1.intensities  ! data from generate.com
inano                       ! You need to tell it if anomalous diffs are
                           ! to be used
noanorefine                 ! do not use ano diffs for refinement
atomname Hg                 ! tell SOLVE what the heavy atom is

derivative 3
label deriv 2 also hg
rawderivfile der2.intensities      ! the derivative data is in this file
nsolsite_deriv 2
inano
noanorefine                 ! do not use ano diffs for refinement
atomname hg
acceptance 0.35             ! accept a new site if it has a
                           ! peak height about 1/3 of avg or more
SCALE_NATIVE                ! scale the native dataset
SCALE_MIR                   ! scale the derivs to the native
ANALYZE_MIR                 ! analyze this mir data and set up for SOLVE
SOLVE
EOD
#
# Now run Resolve to do density modification and build a model
#
resolve << EOD > resolve.log
!solvent_content 0.4      !    solvent fraction
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#  and resolve.pdb has your model
```

```
#
```

COMBINE: Combining multiple MAD and MIR datasets

Combining multiple MAD or MIR datasets is easy with SOLVE. You simply read in one dataset, scale and analyze it, then you give SOLVE the keyword "NEW_DATASET" and go on and read in the next dataset, scale and analyze it. When all datasets are input, you tell SOLVE to go ahead and solve the combined datasets with the command "COMBINE_ALL_DATA", followed by "SOLVE" as usual.

When combining datasets, SOLVE first converts MAD datasets to pseudo-MIR + anomalous scattering (i.e., native + 1 derivative + anomalous differences). Then SOLVE uses the first native dataset as the native. It treats the derivatives associated with that native as derivatives in the usual way. SOLVE treats the native for the other datasets as "derivatives" without heavy atoms, and the derivatives for these datasets as ordinary derivatives.

When the COMBINE command is used, solve writes out individual script files (e.g., solve_mad_01. script) that could be used to analyze the individual datasets, then a final script file (solve_combine.script) that can be used to analyze them all together.

SOLVE takes into account any non-isomorphisms among the different datasets by using Bayesian correlated phasing. This phasing method automatically corrects for non-isomorphism that is shared among a group of derivatives (or in this case derivatives and pseudo-derivatives).

NOTE: Unfortunately COMBINE cannot be used with multiple mtz files.   If you have mtz data and want to use COMBINE you will need to dump your data out to a flat file and read it in using "readformatted".

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
#
# generate 1 mad and 1 mir dataset to solve
#
solve <<EOD

!  GENERATE MAD  and MIR DATASETS:

! dataset 1:   MAD dataset
resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym

percent_error 1
coordinatefile coords.pdb
iranseed -199753
logfile generate_mad.logfile
solvefile generate_mad.prt

mad_atom se                              ! define the scattering factors...
!
lambda 1
label set 1 with 2 se atoms, lambda 1
wavelength 0.9782            ! wavelength value
fprimv_mad  -10              ! f' value at this wavelength
fprprv_mad  3
ATOMNAME Se
xyz  0.44 0.16 0.38
occ 1.0
bvalue 20.
ATOMNAME Se
xyz  0.23 0.45 0.165
occ 1.0
bvalue 20.
!
lambda 2
wavelength 0.977865
```

```
fprimv_mad  -7.5
fprprv_mad  5

lambda 3
wavelength 0.8856
fprimv_mad  -2
fprprv_mad  3.5
!
GENERATE_MAD                    ! generate the MAD dataset now.
!
EOD

solve <<EOD

! NOW DATASET 2 (MIR)
@solve.setup
! change cell slightly for non-isomorphism:
cell 77 27 43 90 102 90
percent_error 1
coordinatefile coords.pdb
iranseed -189753
logfile generate_mir.logfile
solvefile generate_mir.prt

!
derivative 1
label set 1 with 1  pt atom
atomname pt
xyz 0.18 0.53 0.77  ! coords
occ 1.0
bvalue 20.

!
GENERATE_MIR                    ! generate the Pt  MIR  dataset now.
!
EOD
# end of generating mad and mir  datasets
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
solve<<EOD > solve.log
!
!   Take mad and mir datasets that may or may not
!   be exactly isomorphous, combine them into one pseudo-mir dataset
!   and solve it
!
logfile solve.logfile  ! another log file

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym



readformatted       ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged           ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors



mad_atom se   ! define the anomalously-scattering atom

lambda 1
label set 1 with 2 se atoms, lambda 1
wavelength .9782                    ! wavelength value
fprimv_mad  -10                     ! f' value at this wavelength
fprprv_mad  3                       ! f doubleprime
```

```
rawmadfile lam1.intensities          ! data file

lambda 2
wavelength 0.977865
fprimv_mad  -7.5
fprprv_mad  5
rawmadfile lam2.intensities

lambda 3
wavelength 0.8856
fprimv_mad  -2
fprprv_mad  3.5
rawmadfile lam3.intensities

nres 100                [approx # of residues in protein molecule]
nanomalous 2            [approx # of anomalously scattering atoms per protein]
SCALE_MAD               ! read in and localscale the data
ANALYZE_MAD             ! run MADMRG and MADBST and analyze all the Pattersons

!------------------------end of first dataset -------------

new_dataset

!----------------second dataset (MIR with Pt atoms) ----------

rawnativefile native.intensities
!
derivative 1
label set 1 with 1  pt atoms, deriv 1
rawderivfile der1.intensities
atomname pt

nsolsite 1
scale_native
scale_mir
analyze_mir

!--------------------------end of second dataset --------------

!  combine the datasets into one now...

combine
```

```
!------ go -------
solve
!--------all done----------

EOD
#
# Now run Resolve to do density modification
#
resolve << EOD > resolve.log
!solvent_content 0.4        !    solvent fraction
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#  and resolve.pdb has your model
#
echo 'All done.'
```

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
#
#  generate 2 mad datasets
#
solve <<EOD

!  GENERATE 2 MAD DATASETS:

! dataset 1

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym

percent_error 4
coordinatefile coords.pdb
iranseed -199753
logfile generate.logfile
solvefile generate.prt

mad_atom se                              ! define the scattering factors...
!
lambda 1
label set 1 with 2 se atoms, lambda 1
wavelength 0.9782            ! wavelength value
fprimv_mad  -10              ! f' value at this wavelength
fprprv_mad  3
ATOMNAME Se
xyz  0.44 0.16 0.38
occ 1.0
bvalue 20.
ATOMNAME Se
xyz  0.23 0.45 0.165
occ 1.0
bvalue 20.
!
lambda 2
```

```
wavelength 0.977865
fprimv_mad  -7.5
fprprv_mad  5

lambda 3
wavelength 0.8856
fprimv_mad  -2
fprprv_mad  3.5
!
GENERATE_MAD                    ! generate the MAD dataset now.
!
EOD
mv lam1.intensities lam1_se.intensities
mv lam2.intensities lam2_se.intensities
mv lam3.intensities lam3_se.intensities
mv lambda_1.fft dataset_1.fft

solve <<EOD

! NOW DATASET 2 (FE)
@solve.setup
percent_error 2
coordinatefile coords.pdb
iranseed -179753
logfile generate_2.logfile
solvefile generate_2.prt

mad_atom fe                    ! define the scattering factors...
!
lambda 1
label set 1 with 1  fe atoms, lambda 1
wavelength 1.74          ! wavelength value
fprimv_mad  -9           ! f' value at this wavelength
fprprv_mad  2.5            ! f doubleprime value at this wavelength
atomname fe
xyz 0.18 0.53 0.77  ! coords
occ 1.0
bvalue 20.

lambda 2
label Fe PK
wavelength 1.73647
fprimv_mad  -5
```

```
fprprv_mad  4.5

lambda 3
label Fe RM
wavelength 0.978
fprimv_mad  1
fprprv_mad  1.5
!
GENERATE_MAD                    ! generate the FE MAD dataset now.
!
EOD
mv lam1.intensities lam1_fe.intensities
mv lam2.intensities lam2_fe.intensities
mv lam3.intensities lam3_fe.intensities
mv lambda_1.fft dataset_2.fft
# end of generating 2 mad datasets
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
# solve.com  -- take 2 mad datasets that may or may not
#   be exactly isomorphous, combine them into one pseudo-mir dataset
# and solve it
#
solve<<EOD > solve.log
logfile solve.logfile

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym



readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors

!
!---------first MAD dataset (se atoms)---------
mad_atom se
lambda 1
label set 1 with 2 se atoms, lambda 1
wavelength .9782                        ! wavelength value
fprimv_mad  -10                         ! f' value at this wavelength
fprprv_mad  3                           ! f doubleprime value
rawmadfile lam1_se.intensities          ! datafile
```

```
ATOMNAME Se                          ! we're about to enter data on an atom
lambda 2
wavelength 0.977865
fprimv_mad  -7.5
fprprv_mad  5
rawmadfile lam2_se.intensities

lambda 3
wavelength 0.8856
fprimv_mad  -2
fprprv_mad  3.5
rawmadfile lam3_se.intensities
nres 80
nanomalous 2
!
scale_mad
analyze_mad

!------------------------end of first dataset -------------

new_dataset                  ! tell solve we're about to start a new one

!----------------second MAD dataset (fe atoms) ----------
mad_atom fe
!
lambda 1
label set 1 with 1  fe atoms, lambda 1
wavelength 1.74              ! wavelength value
fprimv_mad  -9               ! f' value at this wavelength
fprprv_mad  2.5              ! f doubleprime value at this wavelength
rawmadfile lam1_fe.intensities
atomname fe

lambda 2
label Fe PK
wavelength 1.73647
fprimv_mad  -5
fprprv_mad  4.5
rawmadfile lam2_fe.intensities

lambda 3
label Fe RM
wavelength 0.978
```

```
fprimv_mad  1
fprprv_mad  1.5
rawmadfile lam3_fe.intensities
nres 80
nanomalous 1
scale_mad
analyze_mad
!-------------------------end of second dataset --------------

!  combine the datasets into one now...

combine

!  solve it...
solve
!--------all done----------

EOD
#
# Now run Resolve to do density modification and build a model
#
resolve << EOD > resolve.log
!solvent_content 0.40        !    solvent fraction
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#  and resolve.pdb has your model
```

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
#
#  generate 2 mir datasets
#
solve <<EOD

!  GENERATE 2 MIR DATASETS:

! dataset 1
resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym

percent_error 4
coordinatefile coords.pdb
iranseed -199753
logfile generate.logfile
solvefile generate.prt

!
derivative 1
label set 1 with 1 hg atoms, derivative 1
cell_deriv 76 28 42 90 103 90
ATOMNAME hg
xyz  0.44 0.16 0.38
occ 1.0
bvalue 20.
derivative 2
label deriv 2 set 1, Iodine
cell_deriv 76 28 42 90 103 90
ATOMNAME I
xyz  0.23 0.45 0.165
occ 1.0
bvalue 20.
!
GENERATE_MIR                    ! generate the MAD dataset now.
!
```

```
EOD
mv der1.intensities der1_hg.intensities
mv der2.intensities der2_i.intensities
mv native.intensities native_1.intensities

solve <<EOD

! NOW DATASET 2
@solve.setup
cell 76 28 42 90 103 90
percent_error 4
coordinatefile coords.pdb
iranseed 532131
logfile generate_2.logfile
solvefile generate_2.prt

derivative 1
label set 2 with 1 Pt atoms, derivative 1
cell_deriv 76 28 42 90 103 90
ATOMNAME  pt
xyz 0.71241 0.315 0.2167
occ 1.0
bvalue 25
GENERATE_MIR                         ! generate the MAD dataset now.
!
EOD
mv der1.intensities der1_pt.intensities
mv native.intensities native_2.intensities

# end of generating 2 mir datasets
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
# solve.com  -- take 2 mir datasets that may or may not
#   be exactly isomorphous, combine them into one pseudo-mir dataset
# and solve it
#
solve<<EOD > solve.log
logfile solve.logfile

! solve.setup for test case
resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym


readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors

!
!---------first MIR dataset (hg, i atoms)---------
rawnativefile native_1.intensities
derivative 1
label set 1 with 1 hg atoms, derivative 1
rawderivfile der1_hg.intensities
ATOMNAME hg
derivative 2
```

```
label deriv 2 set 1, Iodine
rawderivfile der2_i.intensities
ATOMNAME I

!
scale_native
scale_mir
analyze_mir

!------------------------end of first dataset -------------

new_dataset                    ! tell solve we're about to start a new one

!----------------second MAD dataset (Pt atoms) ----------
rawnativefile native_2.intensities
derivative 1
label set 2 with 1 hg atoms, derivative 1
rawderivfile der1_pt.intensities
ATOMNAME pt
scale_native
scale_mir
analyze_mir

!  combine the datasets into one now...

combine

!  solve it...
solve
!--------all done----------

EOD
#
# Now run Resolve to do density modification and build a model
#
resolve << EOD > resolve.log
!solvent_content 0.40         !    solvent fraction
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#  and resolve.pdb has your model
#
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
#
#   use all system resources:
#
unlimit
#
solve  <<EOD

! generate se + fe mad dataset
resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym

logfile generate.logfile
solvefile generate.prt
coordinatefile coords.pdb
percent_error  1.0
iranseed 11933
! need to define Fe scattering factors explicitly as it is the 2nd mad atom
! new atom types for Fe (2nd mad atom) at each wavelength:
newatomtype fel1
aval 11.76950       7.357300       3.522200  2.304500
bval 4.761100      0.3072000       15.35350     76.88050
cval 1.036900
fprimv 0.3
fprprv 1.5
newatomtype fel2
aval 11.76950       7.357300       3.522200  2.304500
bval 4.761100      0.3072000       15.35350     76.88050
cval 1.036900
fprimv 0.3
fprprv 1.5
newatomtype fel3
aval 11.76950       7.357300       3.522200  2.304500
```

```
bval 4.761100      0.3072000       15.35350     76.88050
cval 1.036900
fprimv 0.3
fprprv 1.5
newatomtype fel4
aval 11.76950       7.357300       3.522200  2.304500
bval 4.761100      0.3072000       15.35350     76.88050
cval 1.036900
fprimv -9.0
fprprv 2.5
newatomtype fel5
aval 11.76950       7.357300       3.522200  2.304500
bval 4.761100      0.3072000       15.35350     76.88050
cval 1.036900
fprimv -5.0
fprprv 4.5



! dataset treated as se MAD data with Fe as extra atom
mad_atom se                   ! define the scattering factors...
!
lambda 1
label set 1 with se + Fe , lambda 1
wavelength  0.9000            ! wavelength value
fprimv_mad  -1.6              ! f' value for Se (fe defined in fel1)
fprprv_mad  3.4
ATOMNAME Se
! generate automatically sets up se at all the other wavelengths too
xyz  0.44 0.16 0.38
occ 1.0
bvalue 20
atomname fel1
! generate automatically sets up fel2, fel3 etc at other wavelengths
xyz 0.15 0.33 0.40
occ 1.0
bvalue 20

lambda 2
wavelength .9794
fprimv_mad  -8.5
fprprv_mad 4.8

lambda 3
```

```
wavelength 0.9797
fprimv_mad  -9.85
fprprv_mad   2.86

lambda 4
wavelength 1.74             ! wavelength value
fprimv_mad -0.6           ! f prime value for Se (fe defined in fel4)
fprprv_mad  1.4

lambda 5
wavelength 1.736             ! wavelength value
fprimv_mad -0.6           ! f prime value for Se (fe defined in fel4)
fprprv_mad  1.4


generate_mad

EOD
```

```
#!/bin/csh
#
#   solve.com file to run SOLVE/RESOLVE version 2.01
#
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#  set CCP4 and SOLVETMPDIR  and SYMOP variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
#
#   use all system resources:
#
unlimit
#
solve<<EOD > solve.log

! Solve 1 mad dataset with 2 anomalously-scattering atoms

logfile solve.logfile

resolution 20 3.0
cell 76 28 42 90 103 90
symfile /usr/local/lib/solve/c2.sym


readformatted        ! readformatted/readdenzo/readtrek/readccp4_unmerged
premerged            ! premerged/ unmerged
read_intensities     ! read_intensities/read_amplitudes
fixscattfactors      ! fixscattfactors/refscattfactors

title  5-wavelength 2-ano scatterer MAD dataset   !  a title for this dataset

! Read in the data twice: once treated as Se and once as Fe.

mad_atom se                  ! the anomalously scattering atom is selenium
lambda 1                     ! info on wavelength #1 follows
label Wavelength #  1        ! a label for this wavelength
rawmadfile lam1.intensities  ! datafile for lambda 1 containing raw Intensities

wavelength 0.9000            ! wavelength value
```

```
fprimv_mad  -1.6             ! f' value at this wavelength (for Se)
fprprv_mad  3.4              ! f doubleprime value at this wavelength

atomname se                  !  the atom name

lambda 2
rawmadfile lam2.intensities
wavelength 0.9794
fprimv_mad  -8.5
fprprv_mad  4.8

lambda 3
rawmadfile lam3.intensities
wavelength 0.9797
fprimv_mad  -9.85
fprprv_mad  2.86

nres 100                [approx # of residues of protein in asymmetric unit]
nanomalous 1              [approx # of anomalously scattering Se atoms in a.u.]
SCALE_MAD                ! read in and localscale the data
ANALYZE_MAD              ! run MADMRG and MADBST and analyze all the Pattersons

! now do it all over for Fe instead of Se:

NEW_DATASET                  ! tell solve we're starting a new dataset

mad_atom Fe                  ! the anomalously scattering atom is Fe

lambda 1                     ! info on wavelength #1 follows
label Wavelength #  1         ! a label for this wavelength
rawmadfile lam4.intensities  ! datafile for lambda 1 containing raw Intensities
atomname fe

wavelength 1.74             ! wavelength value
fprimv_mad  -9              ! f' value for Fe at this wavelength
fprprv_mad  2.5             ! f doubleprime at this wavelength

lambda 2
rawmadfile lam5.intensities
wavelength 1.73647
fprimv_mad  -5
fprprv_mad  4.5
```

```
lambda 3
rawmadfile lam3.intensities
wavelength 0.9797
fprimv_mad 0.3
fprprv_mad  1.5

nres 100                [approx # of residues of protein in asymmetric unit]
nanomalous 1            [approx # of anomalously scattering Fe atoms in a.u.]
SCALE_MAD               ! read in and localscale the data
ANALYZE_MAD             ! run MADMRG and MADBST and analyze all the Pattersons

! now combine the datasets into one

COMBINE_ALL_DATA

! now solve it as a whole

SOLVE                   ! Solve the structure

!--------all done----------

EOD
#
# Now run Resolve to do density modification and build a model
#
resolve << EOD > resolve.log
!solvent_content 0.4       !    solvent fraction
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#  and resolve.pdb has your model
#
```

Contents	Index

All commands for SOLVE

This is a list of all the commands for SOLVE. Commands cause SOLVE to do something (solve a structure, scale data, search for heavy atoms, draw a map), while keywords set values of parameters (number of heavy atom sites, f' value)

Also see the list of common keywords that apply to automated SOLVE operation and the list of all keywords.

```
 ALIAS: set an alternate name for a keyword
 ANALYZE_MAD: analyze MAD data with MADBST and MADMRG
 ANALYZE_MIR: analyze MIR data
 AVG_OMIT: average key parts of a set of omit maps
 BTOF: convert from binary dorgbn file to formatted one

 CART_TO_FRACT: convert Cartesian coordinates to fractional
 COMBINE_ALL_DATA: put together several datasets into pseudo-MIR dataset
 COMPARE_SOLN:  compare two heavy-atom solutions and see if they are equivalent
 COMPLETE: determine completeness of a dataset
 DRGTOXPLOR: write out FOBS, SIGMA in X-PLOR format
 END: end the program
 EXPORT: write out h,k,l, data without any titles
 FDIFF: Create pseudo-mutant dataset for difference refinement
 FFTTOCCP4: convert asymmetric unit of FFT to asymmetric unit in ccp4 map format
 FFTTOEZD: convert asymmetric unit of FFT to any portion of cell in EZDNEW format
(for O)
 FFTTOMAPVIEW: convert asymmetric unit of FFT to any portion of cell in a format for
MAPVIEW
 FILEMERGE: combine or extract data columns from dorgbn files
 FTOB: convert from formatted dorgbn file to binary one
 FRACT_TO_CART: convert fractional coordinates to Cartesian
 GENERATE_MAD:  generate MAD dataset
 GENERATE_MIR:  generate MIR dataset
 GETANOM: convert from F+ and F- to Fbar and DelAnom
 GETISO: calculate differences between two data columns
 GETPHASES: convert from A,B coefficients to F and phase
 HA_PDB:  write out current set of heavy atoms in pdb format
 HASSP:  search for solutions to a patterson map
 HISTORY: list commands and keywords input to SOLVE
 HEAVY:  refine heavy atom parameters
 HELP: get information on this program
 IMPORT:  read in formatted file with h,k,l,data, stripping off any text
 LOCALSCALE: scale one dataset to another with local scaling
 MADBST: estimate heavy-atom structure factors from MAD data
 MADMRG: create pseudo-SIR+anom data from MAD data
 MAPS:  calculate Fourier and Patterson maps
 MAPTOASYM: Map a PDB file onto the asymmetric unit of the crystal
 MAPTOOBJECT: Map atoms to equivalent position close to  specified object
 MATH: Add, subtract, generate columns of data, used to  construct test datasets and
```

```
operate on data.
 MERGE: merge equivalent reflections and write out asymmetric unit
 NEW_DATASET: start reading in a new dataset to be combined later
 NOLOG: stop writing log file

 PEAKSEARCH: find peaks in Fourier map
 PREFERENCES: display symmetry file, cell dimensions, and grids for FFT
 RHO: Write out value of a map at positions of atoms in a PDB file.

 SAD: solve a SAD dataset
 SCALE_NATIVE: read in and scale native data for a MIR dataset
 SCALE_MAD: read in and scale MAD data
 SCALE_MIR: read in and scale MIR data following SCALE_NATIVE
 SOLVE:  solve an MIR or MAD dataset
 STOP: end the program

 VIEW: view a binary dorgbn file
 WEIGHTS: set up weighting scheme for atomic refinement
```

Automated structure determination with SOLVE

What you will need to input to SOLVE

To solve a structure automatically using SOLVE you need the following things:

- Your unit cell information and resolution.
- some raw datafiles in Scalepack or d*TREK or just a plain text format
- knowledge of what your heavy atoms are, and for MAD data, approximate values of the scattering factors (f' and f") at each wavelength. The best place to get these f' and f" numbers are from the beamline staff where you collected your data. If you have no other source of f' and f" information, you can use tables of values, but this won't work as well.

How to run SOLVE

To run SOLVE, all you need to do is:

- Set the environmental variable CCP4_OPEN to UNKNOWN with the following command:
  - setenv CCP4_OPEN UNKNOWN
  - (You can do this in your .login_custom file if you like)
- Choose a sample control file, edit it to match your data, and run SOLVE with it.
- You can watch SOLVE run by looking at the end of the "solve.status" with "tail -30f solve. status". This file will tell you where to look if something got typed in wrong and it will keep you informed about the structure determination as it goes.
- After SOLVE finishes, have a look at how to interpret your SOLVE output.
- If SOLVE is looking through more solutions than you want it to, and you would like SOLVE to finish up as quickly as possible, use the solve.control file to tell SOLVE to finish up.
- You may want to look at the keywords that you can use to control how SOLVE carries out its search for solutions.
- Also have a look at what to do next such as the ways that you can try and get more solutions
- Have a look at how to pass your data on to other packages after SOLVE has finished.

If you want to run solve more interactively or use it to do a host of other things such as peaksearches on maps, converting from binary to formatted files, or calculating coefficients for difference refinement, have a look at the table of contents for a list of things you can do.

You might want to try SOLVE out on some test data in your space group using the generate feature. This allows you to create a MAD or MIR dataset with any heavy atom sites you want and then run SOLVE on it. If you start with a PDB coordinate file, you can generate a dataset, solve it, and use "O" to display the NEWEZD electron density map that SOLVE creates along with the correct structure.

How to interpret the output from SOLVE

SOLVE writes out some files that have information on how the structure solution went and others that allow you to look at a map and export phases. Here are the most important files that SOLVE writes out when it is done:

- solve.prt (Summary of SOLVE results)
- solve.ezd (Portable electron density map)
- solve.ccp4_map (CCP4 electron density map)
- solve.mtz (ccp4 mtz file with F, Phases, Hendrickson-Lattman coefficients)
- phases-hl.export (Formatted file with phases and Hendrickson- Lattman coefficients ready to read into ccp4)
- phases-hl.script (Final parameters ready to use again in SOLVE)

Other files SOLVE writes out include:

- patt_Fa.ezd (Portable Fa Patterson, MAD data only)
- patt_iso_orig_removed_der1.ezd (Portable origin-removed isomorphous difference patterson, MIR only)
- patt_ano_orig_removed_der1.ezd (Portable origin-removed anomalous difference patterson, MIR with ano differences only)
- ha.pdb (heavy atoms in pdb format)
- solve.fft
- phases-hl.drg

solve.prt: an analysis of the top solutions found.

- If there is a really clear solution, then just one solution is written out
- For each solution, the solve.prt file lists (see examples):
    - the Patterson solution,
    - values of cross-validation difference Fouriers for each heavy atom site
    - phasing statistics (see HEAVY, for details)
    - Z-scores for
        - the Patterson
        - cross-validation difference Fouriers
        - analysis of the native Fourier
        - figure of merit

        - overall scoring
- You want to look for:
    - mostly positive peaks in your patterson solution,
    - high peaks (5 to 15 sigma) for all your cross-difference Fouriers
    - a decent figure of merit (0.5 or better)
    - Z-score for analysis of the native Fourier of 2 or more if you have good MAD data and a figure of merit over 0.65
    - an overall Z-score of 5-10 for a 2-site solution or 20-50 for a 10-site solution
    - If SOLVE printed out several solutions, then you should expect to see some duplication among the top solutions
        - this is because the solutions are each from a different seed and different seeds should give the same overall solution
        - See the analysis of relationships among solutions at the end of solve.prt
        - If you have no duplication at all and all solutions seem to have about the same poor scores, then SOLVE probably has not obtained the right solution
        - If there is one really high score or lots of duplication, then it is probably on the right track
- See what to do next after you've run SOLVE for how to look at your map, export your data, or improve your solution

Files for exporting output from SOLVE

solve.ezd: a portable native electron density map.

The "solve.ezd" file is an NEWEZD electron density map. Read this into "O" and you're ready to look at your electron density! In "O", just type,

```
@solve.ezd
ezd_draw
<cr>
<cr>
```

and you've got your map.

You can also read this map into the program mapman with the flag NEWEZD and convert it to ccp4 or other formats.

solve.ccp4_map: a CCP4-formatted native electron density map.

The solve.ccp4_map file is ready to be read into CCP4 programs for viewing or modification.

solve.mtz: a ccp4 mtz file with F, phases and Hendrickson- Lattman coefficients.

The "solve.mtz" file is a binary file in ccp4 mtz format containing Fnative, sigma, phases, figure of merit, and Hendrickson-Lattman coefficients calculated using Bayesian phasing. The column names in this file are:

- H K L FP SIGFP PHIB FOM HLA HLB HLC HLD
- H K L are indices
- FP SIGFP PHIB FOM are F, sigma, best phase, and figure of merit. For MAD data, the FP and phase are values estimated at the reference wavelength (usually the shortest wavelength).
- HLA HLB HLC HLD are Hendrickson-Lattman coefficients

phases-hl.export: a portable list of phases.

The "phases-hl.export" file is a formatted file containing Fnative, sigma, phases, figure of merit, and Hendrickson-Lattman coefficients calculated using Bayesian phasing. Copy and edit the header at the top into a command file and use it to read the "phases-hl.export" file into ccp4 mtz format. Then you're ready to do density modification or whatever you like with your map.

Also see Importing and Exporting data with SOLVE for more information about importing datafiles and maps.

phases-hl.script: a script file with heavy atom parameters

The routine also writes out a file "phases-hl.script" containing your refined parameters for the heavy atoms.

- You can paste the heavy atom parameters in phases-hl.script back into the solve_mad.script file or solve_mir.script file written by ANALYZE_MAD (or ANALYZE_MIR)
- Then you are set to continue on with ADDSOLVE or ANALYZE_SOLVE

patt_Fa.ezd -- a portable Fa Patterson map (MAD data)

patt_iso_orig_removed_der1.ezd and patt_ano_orig_removed_der1.ezd -- portable isomorphous and anomalous origin-removed patterson maps

These maps are in NEWEZD format and can be viewed with O or converted to other formats with mapman in the same way as the solve.ezd electron density map. (For derivative 1 the origin-removed maps are called ...der1 and for derivative 2 ...der2, etc). Note that if you have a multiple-MAD dataset the names of all these files will have _02, _03 etc appended to indicate the dataset number they came from.

ha.pdb -- heavy atoms in PDB format

Solve writes out the heavy atom positions for the top solution in PDB format in the file ha.pdb. For MAD data the heavy atom names are something like LAM1 (for lambda-1). The residue names are the derivative (or wavelength) number (DE1 means derivative 1).

solve.fft -- UCLA format map file

Here solve.fft is the same map as "solve.ezd", but in the ucla format.

phases-hl.drg -- binary output file

The "phases-hl.drg" contains the same information as "phases-hl.export" but it is in the binary ucla format. "phases-hl.drg" can be used to calculate electron density maps in this package or in another package.

Generating and solving model datasets with errors

Generate allows you to construct an MIR or MAD dataset in which you specify the heavy atom locations and types. You can even specify the cell parameters for each derivative of an MIR dataset to simulate non-isomorphism. The output from GENERATE is suitable as input to SOLVE and you can run them in one script file to generate, then SOLVE, a dataset.

If you start with a PDB file in "coords.pdb" and specify "checksolve", then you can generate a dataset, solve it, and display the "solve.ezd" electron density map that SOLVE comes up with using "O". The map will automatically be referred to the same origin as the coords.pdb structure so you can overlay your map and the model to see how good the solution is. Please note that the EZD map will cover the asymmetric unit only. You may need to put your model in the asymmetric unit or else use Gerard Kleywegt's program mapman; manipulate your map (read it in to mapman as "NEWEZD") before you overlay the map.

Here are sample files that generate and solve MIR and MAD datasets. The keywords for generate_mir and generate_mad follow after the samples.

If you specify "checksolve" when you run one of these command files then SOLVE will automatically compare all the solutions it is getting with the one that you started with.

```
!----------------------------------------------------------------
!gensolvemir.script
! command file to generate an MIR dataset and solve it

CELL 76 28 42 90 103 90
SYMFILE /usr/local/lib/solve/c2.sym
resolution 3.0 20.0
logfile gensolvemir.logfile
solvefile gensolvemir.prt
percent_error 3.0                       ! 3% error added to intensities
coordinatefile coords.pdb              ! coordinate file used to generate
                                        ! the starting I's (if none supplied,
                                        ! the routine makes up I's
deriv 1
cell_derivative  77 28 41 90 103 90    ! Try cell params for derivatives that
                                        ! are about 1% different from wt
inano
atom hg
occ 1.0
bvalue 31.
xyz 0.15 0.25 0.35

deriv 2
cell_derivative  75 28 42 90 102.5 90
inano
atom au
```

```
occ 0.8
bvalue 25.
xyz 0.33 0.15 0.17

GENERATE_MIR                            ! generate the MIR dataset now.

! Now the data are in: native.intensities, der1.intensities, and der2.intensities

!...  now analyze this MIR dataset...

rawnativefile native.intensities        !file for native data

premerged
readformatted

gotoder 1
rawderivfile der1.intensities           ! We have to use "gotoder" because we're in the
                                        ! middle of SOLVE, not starting from the
                                        ! beginning, and we have already specified
                                        ! more than one derivative.
gotoder 2
rawderivfile der2.intensities

nres 87                                 [approx # of residues in protein molecule]
nsolsite 1                              ! one site per derivative
checksolve                              ! compare the solutions to the correct one
comparisonfile native.fft         ! get correlation coefficient of map
                                 !calculated from each solution along the
                                 !way with the true map in native.fft

scale_native
scale_mir
analyze_mir
solve
!----------------------------------------------------------------
```

... and now for a MAD dataset:

```
!----------------------------------------------------------------
!gensolvemad.script
! command file to generate a MAD dataset and solve it
CELL 72 28 42 90 103 90
SYMFILE /usr/local/lib/solve/c2.sym
resolution 3.0 20.0
logfile gensolvemad.logfile
solvefile gensolvemad.prt
percent_error 3.0                       ! 3% error added to intensities
coordinatefile coords.pdb              ! coordinate file used to generate
                                        ! the starting I's (if none supplied,
                                        ! the routine makes up I's
mad_atom se                              ! define the scattering factors...
lambda 1
```

```
wavelength 0.90
fprimv_mad -1.6
fprprv_mad 3.4
atomname se
xyz 0.197 0.377 0.216
occ 1.0
bfactor 20
atomname se                             ! you only have to specify the coords for
                                        ! this one wavelength (they're copied to the
                                        ! others)
xyz 0.216 0.115 0.399
occ 1.0
bfactor 20
lambda 2
wavelength 0.9794
fprimv_mad -8.5
fprprv_mad 4.8
lambda 3
wavelength 0.9797
fprimv_mad -9.85
fprprv_mad  2.9

GENERATE_MAD                            ! generate the MAD dataset now.

! Now the data are in: lam1.intensities, lam2.intensities, and lam3.intensities for
!  the 3 wavelengths of data

! solve the dataset

premerged
readformatted

gotoder 1
rawmadfile lam1.intensities
gotoder 2
rawmadfile lam2.intensities
gotoder  3
rawmadfile lam3.intensities

nres 87                  [approx # of residues in protein molecule]
nanomalous 2
checksolve
comparisonfile lambda_1.fft         ! get correlation coefficient of map
                             !calculated from each solution along the
                             !way with the true map in lambda_1.fft

scale_mad
analyze_mad
solve
!----------------------------------------------------------------
```

Notes on using GENERATE_MAD

You can have your generated MAD dataset contain more than one anomalously-scattering atom. You input information on the first atom type in the usual way as described above. For the second atom type, you need to:

- input one NEWATOMTYPE with scattering factors for each wavelength of MAD data. The NEWATOMTYPE for the various wavelengths must be of the Form PTL1 PTL2 PTL3 etc., for lambda 1, 2, 3.
- input the heavy atom parameters for this atom for lambda 1
- SOLVE will generate heavy atom parameters for this atom for all the other wavelengths and will include it in the generate procedure.

Keywords for GENERATE_MIR and GENERATE_MAD

```
coordinatefile          pdb file with coordinates.  Used to generate the starting
                        values of F and phase for the structure. Only C N O and S
                        atoms are read in.

percent_error           % error added to intensities

cell_derivative a b c alpha beta gamma    (only for generate_mir) cell parameters
                                          for this derivative.

derivative nn
lambda nn               derivative or wavelength number

atomname   xx           name of an atom about to be specified
xyz x y z               coordinates of this atom
bvalue b                b-factor
occupancy               occupancy value
```

SCALE_MAD: Scaling MAD datasets

SCALE_MAD is a routine to read in MAD data from a Scalepack output file or from another formatted file and to scale it and put it all in a file suitable for ANALYZE_MAD and SOLVE. It is ordinarily called as the first step in a completely automated structure determination and is followed by ANALYZE_MAD and SOLVE. It can also be run on its own. The routine works well when the MAD data has been collected in very nearly the same way at all wavelengths and anomalous differences have been measured at all wavelengths either by inverse beam or by mirror plane symmetry. The routine assumes that either:

- you have already merged your anomalous data and your data files contain 4 columns (I+,sigma,I-, sigma), or
- you have not merged the data and your data columns contain 2 columns (I, sigma).

(Note: the data can be amplitudes, not intensities if you read in the data with READFORMATTED and you specify the flag READ_AMPLITUDES)

For the second case, reflections with H K L corresponding to I+ and I- are treated as such.

SCALE_MAD first estimates the overall absolute scale of the data using the data in the lowest resolution shell from the first datafile and the number of protein residues in the asymmetric unit (NRES). SCALE_MAD separates Bijvoet pairs into separate files. SCALE_MAD scales all the input data files to the first data file using just an overall scale and B. Then it merges all the data from all datafiles into one averaged dataset and rescales all the datasets to the averaged dataset using LOCALSCALE.

A typical input script for SCALE_MAD looks just like the one for automated MAD structure determination except you leave off the commands "ANALYZE_MAD" and "SOLVE".

```
                     KEYWORDS for SCALE_MAD:


LAMBDA n          The next RAWMADFILE(s) that are read in will be
                   for lambda n




MADFBARFILE xx.scl Output file with (Fbar,sigma,DelAno,sigma)
                   for each wavelength will be xx.scl
```

```
                    (DEFAULT="mad_fbar.scl")

MADFPFMFILE yy.scl Output file with (F+,sigma,F-,sigma) for each
                    wavelength will be yy.scl
                    (DEFAULT="mad_fpfm.scl")
```

See also the commonly-used keywords for automated SOLVE operation.

ANALYZE_MAD: Analyzing a MAD dataset

ANALYZE_MAD is a routine to run MADMRG and MADBST. You usually do not have to worry about this routine at all because it is ordinarily called right after running SCALE_MAD for automatic structure determination, and all the parameters are already set for you. It is ordinarily followed by running SOLVE with the solve_mad.script file written out by this routine or by using the keyword "SOLVE" after running this routine.

ANALYZE_MAD assumes that you have a datafile ("mad_fbar.scl") that contains (Fbar,sigma,DelAno,sigma) for each wavelength of MAD data. That is, there are exactly 4 data columns for each wavelength. This is a dorgbn file. It is ordinarily created by SCALE_MAD, but you can create your own if you like.

To run ANALYZE_MAD, you need to input your standard setup file ("solve.setup"), and the wavelength to define as the standard (JSTD, see below). You need to input the scattering factors for the anomalously scattering atom at the various wavelengths too. You also need to input an estimate of the number of anomalously scattering atoms in the asymmetric unit (NANOMALOUS) and the number of protein residues in the asymmetric unit (NRES). It is only the ratio of NANOMALOUS to NRES that is important.

ANALYZE_MAD will run MADMRG (output ="madmrg.out") and MADBST ("madbst.out"). See the writeups below for these two routines. It will calculate origin-removed difference Pattersons for all dispersive and anomalous differences and Pattersons based on the output of MADMRG and MADBST. These maps are all compared to each other and the correlation coefficients are displayed in a table. In a typical MAD experiment the anomalous and dispersive difference Pattersons have correlations with each other on the order of 0.1 to 0.3 or so (pretty low, so don't be worried).

A typical input script for ANALYZE_MAD follows.

```
                     Script for ANALYZE_MAD

!--------------------------Run ANALYZE_MAD---------------------- ----------
@solve.setup                  !  standard information for this crystal

madfbarfile mad_fbar.scl      !  input dorgbn file with (Fbar,sig,Delano,Sig)
                              !    for each wavelength
madfpfmfile mad_fpfm.scl      !  input dorgbn file with (F+,sig,F-,Sig)
                              !    for each wavelength

logfile analyze_mad.logfile   !  write out most information to this file

mad_atom se                   !  atom type is selenium

fixscattfactors               ! do not refine scattering factors (you can if
                              ! you want and the data is good)
```

```
jstd 1                        ! Define wavelength #1 as the reference wavelength
lambda 1                      ! wavelength #1 information is to follow
label   Wavelength #1         ! label for lambda 1
wavelength 0.9000             ! wavelength value
fprimv_mad  -1.6              ! f' value at this wavelength
fprprv_mad  3.4               ! f" value at this wavelength

lambda 2
wavelength 0.9794
fprimv_mad  -8.5
fprprv_mad  4.8

lambda 3
wavelength 0.9797
fprimv_mad  -9.85
fprprv_mad  2.86

nres 100                  [approx # of residues in asymmetric unit]
nanomalous 2              [approx # of anomalously scattering atoms in a.u.]

madmrgfile madmrg.out    ! write the SIRAS-like MAD dataset to madmrg.out
madbstfile madbst.out    ! write the Bayesian Patterson to madbst.out

ANALYZE_MAD              ! run MADMRG and MADBST and analyze all the Pattersons

!------------------------------------------------------------------------------
```

Keywords for ANALYZE_MAD:

```
LOGFILE xx.logfile  log file for output will be xx.logfile

madfbarfile xx.scl  input dorgbn file with (Fbar,sig,Delano,Sig)
                        for each wavelength will be xx.scl
                        [Default="mad_fbar.scl"]
madfpfmfile xx.scl  input dorgbn file with (F+,sig,F-,Sig)
                        for each wavelength will be xx.scl
                        [Default="mad_fpfm.scl"]

madmrgfile xxx.out   write the SIRAS-like MAD dataset to xxx.out
                    [     default="madmrg.out"]

madbstfile yyy.out   write coefficients for a Bayesian Patterson to yyy.out
                          [default="madbst.out"]

SOLVEDATAFILE  xxx  Output datafile with MADMRG output and MADBST
                    output  combined together, suitable for use
                    with routine SOLVE, will be xxx.   (DEFAULT file name =
                    "solve.data") The datafile has the following
                    columns of data:

                    1         Fnative (from MADMRG)
                    2         Sigma of Fnative (from MADMRG)
                    3         Fderiv (from MADMRG)
                    4         Sigma of Fderiv (from MADMRG)
                    5         DelAno (from MADMRG)
                    6         Sigma of DelAno (from MADMRG)
                    7         <Fh cos(theta)> (from MADBST)
                    8         <Fh sin(theta)> (from MADBST)

SCRIPTFILE xxx      Output script file containing instructions
                    for running SOLVE written to xxx
                    [default="solve_mad.script"]. Starting SOLVE with
                    this script is equivalent to following
                    ANALYZE_MAD with SOLVE.

FIXSCATTFACTORS     Fix scattering factors at their input values.
                    This is a good idea if you have a reasonable
                    idea of the f' and f" values.  [this is the
                    default]

REFSCATTFACTORS     refine scattering factors f' and f".  If you
                    refine them, be sure to look at their new
                    values at the end of the routine MADMRG and
                    verify that they are reasonable.

mad_atom xx         anomalously scattering atom is "xx"

The atom types recognized by SOLVE are:
H, H-1, He, Li, Li+1, Be, Be+2, B, C, Cv, N, O, O-1,
F, F-1, Ne, Na, Na+1, Mg, Mg+2, Al, Al+3, Si, Siv, Si+4,
P, S, Cl, Cl-1, Ar, K, K+1, Ca, Ca+2, Sc, Sc+3, Ti, Ti+2,
Ti+3, Ti+4, V, V+2, V+3, V+5, Cr, Cr+2, Cr+3, Mn, Mn+2, Mn+3,
Mn+4, Fe, Fe+2, Fe+3, Co, Co+2, Co+3, Ni, Ni+2, Ni+3, Cu,
Cu+1, Cu+2, Zn, Zn+2, Ga, Ga+3, Ge, Ge+4, As, Se, Br,
Br-1, Kr, Rb, Rb+1, Sr, Sr+2, Y, Y+3, Zr, Zr+4, Nb, Nb+3,
Nb+5, Mo, Mo+3, Mo+5, Mo+6, Tc, Ru, Ru+3, Ru+4, Rh, Rh+3,
Rh+4, Pd, Pd+2, Pd+4, Ag, Ag+1, Ag+2, Cd, Cd+2, In, In+3,
Sn, Sn+2, Sn+4, Sb, Sb+3, Sb+5, Te, I, I-1, Xe, Cs, Cs+1,
Ba, Ba+2, La, La+3, Ce, Ce+3, Ce+4, Pr, Pr+3, Pr+4, Nd,
Nd+3, Pm, Pm+3, Sm, Sm+3, Eu, Eu+2, Eu+3, Gd, Gd+3, Tb,
Tb+3, Dy, Dy+3, Ho, Ho+3, Er, Er+3, Tm, Tm+3, Yb, Yb+2,
Yb+3, Lu, Lu+3, Hf, Hf+4, Ta, Ta+5, W, W+6, Re, Os, Os+4,
Ir, Ir+3, Ir+4, Pt, Pt+2, Pt+4, Au, Au+1, Au+3, Hg, Hg+1,
Hg+2, Tl, Tl+1, Tl+3, Pb, Pb+2, Pb+4, Bi, Bi+3, Bi+5, Po,
At, Rn, Fr, Ra, Ra+2, Ac, Ac+3, Th, Th+4, Pa, U, U+3, U+4,
U+6, Np, Np+3, Np+4, Np+6, Pu, Pu+3, Pu+4, Pu+6, Am, Cm, Bk, Cf
```

```
                    If xx is not recognized by SOLVE you need to
                    specify a newatomtype and then refer to it

lambda n            wavelength number (n=1,2,3...) for values
                        to be entered next of f' and f"

wavelength xx       wavelength for lambda xx (i.e., 0.9798)

fprimv_mad  xx      1 real number for f' value for anomalously
                    scattering atom at the current wavelength.
                    Wavelength is defined by the most recent
                    value of the keyword "LAMBDA".  You always
                    need to enter f' and f" explicitly for MAD
                    data.

fprprv_mad  xx      1 real number for f" value for anomalously
                    scattering atom at the current wavelength.
                    Wavelength is defined y the most recent value
                    of the keyword "LAMBDA"

JSTD  n             Wavelength "n" will be used as the standard
                    to which all data is referred (see MADMRG)
                    [default= the lowest wavelength]

NRES   n            # of residues in asymmetric unit
                    [default=100]

NANOMALOUS n        # of anomalously scattering atoms in asymmetric unit.
                    Used to estimate how big the Fa values might be

NSHELLS n           Number of shells for analysis is n
                    [default=10]

bayes               Use Bayesian MAD phasing at the
                       very end of SOLVE. Requires that madfpfmfile
                        exists.  It also requires that INPHASE be
                       specified for all wavelengths. (This is the default)

nobayes             Use the compressed MADMRG datafile for all
                       phasing when program gets to SOLVE.
```

MADMRG: Conversion of MAD data to pseudo-SIR with anomalous scattering

MADMRG is a routine for compressing a MAD dataset into a nearly-equivalent MIR + anomalous scattering pseudo-dataset. This is useful for refinement of parameters describing the anomalously scattering atoms for several reasons, not the least of which is the very large improvement in speed of refinement. This fast refinement is critical for the SOLVE automated structure determination routine. Ordinarily you will not run MADMRG by itself, but rather as part of ANALYZE_MAD or an automated structure determination. This means that you usually will not need to be familiar with all the keywords for MADMRG.

MADMRG reads in measurements of Fbar and the anomalous differences DelAno at several wavelengths for each reflection. From this data and the known values of f' and f" for the anomalously scattering atoms at these wavelengths, the program estimates (1) the magnitude of the structure factor corresponding to all atoms except the anomalous scatterer (Fo), (2) the "isomorphous" difference that would be measured +/- the anomalous scatterer at a standard wavelength, and (3) the anomalous difference that would be measured at this standard wavelength. In this way, the MAD data is converted to a form identical to that used in the analysis of SIR+anomalous differences data.

MADMRG assumes that structure factor due to anomalous scatterer is not large compared to that due to all other atoms. In this case iso differences among various wavelengths are proportional to differences in (f+f') for the anomalous scatterer, and ano diffs at each wavelength are proportional to f". MADMRG scales all the ano diffs to a common wavelength, then averages them. It takes all the iso diffs (e.g., L3-L1, L3-L2, L1-L2), and scales each iso diff by: (f+f' at std wavelength)/(difference in f+f' at the 2 wavelengths) to obtain estimate of what would be measured for the structure factor amplitude due to the entire structure at the standard wavelength minus the structure factor amplitude of the entire structure without the anomalously scattering atoms. That is, it estimates, delta Fiso (+/- ano scatterer at the standard wavelength). Finally, MADMRG obtains estimates of what delta Fiso would be at each wavelength by scaling std Fiso by f+f' at that lambda. This allows the program to obtain estimates of Fo, the structure factor amplitude due to all non-anomalously scattering atoms from each value of (Fbar - Fiso at that lambda). These estimates of Fo are averaged.

The result of these manipulations is a pseudo-SIR+anomalous differences dataset. The "native" structure factor amplitude is Fo, the estimate of the structure factor amplitude due to all non-anomalously scattering atoms at the standard wavelength. The "derivative" structure factor amplitude is Fo plus the isomorphous difference, delta Fiso, corresponding to the contribution of the anomalously scattering atoms at the standard wavelength. The anomalous difference is the averaged anomalous difference, scaled to the value at the standard wavelength. Generally, the standard wavelength is chosen to be one well away from the absorption edge of the anomalously scattering atoms, so that f' is small or negligible. This is not essential, however.

Keywords for MADMRG

```
NSHELLS   n      number of shells of resolution used to group data is
                   n(default=10)

INFILE  xxx      input file is xxx
OUTFILE xxx      output file is xxx

mad_atom xx              anomalously scattering atom is "xx".

lambda xx                wavelength number (1,2,3...) for values to be entered
                         next of f' and f" and for column numbers
wavelength  xx           wavelength value for lambda xx (e.g., 0.9798)
fprimv_mad  xx           1 real number for f' value for anomalously scattering
                         atom at the current wavelength.  Wavelength is defined
                         by the most recent value of the keyword "LAMBDA". You
                         need f' and f" regardless of whether you input
                         mad_atom or aval_mad...
fprprv_mad  xx           1 real number for f" value for anomalously scattering
                         atom at the current wavelength.  Wavelength is defined
                         by the most recent value of the keyword "LAMBDA"
LABEL           label for this  wavelength
NCOLFBAR   n     Fbar for this wavelength
NCOLSFBAR  n     sigma of Fbar
NCOLDELF   n     Del F ano (Fplus - Fminus)
NCOLSDELF  n     sigma of del F ano

JSTD    n        wavelength ID for wavelength to be considered the STANDARD
                 [default=1]
FIXSCATTFACTORS   Fix scattering factors at their input values
REFSCATTFACTORS   refine scattering factors f' and f"
                         [default=fix (i.e., do not refine them)]
```

Notes on MADMRG:

OUTPUT FILE: This a DORGBN-style output file containing 8 columns of data. They are:

1 madmrg est of Fp-zero ("Fnative")
2 madmrg sig of fp-zero ("sig of Fnative")
3 madmrg: MOCK FDER ("Fderiv";
equal to Fp-zero + del iso)
4 madmrg sig of del iso ("Sig of Fderiv")

5 madmrg est of del ano ("Delano")
6 madmrg sig of del ano ("Sig of Delano")
7 madmrg weighted est of del iso for Patterson
8 madmrg weighted est of del ano for Patterson

To use this data as MIR + anomalous differences, simply use columns 1 and 2 as Fp (native F) and sigma; columns 3 and 4 as Fder (derivative F) and sigma; and columns 5 and 6 as delano and sigma.

The last two output columns are for use in drawing Patterson maps only. Column 7 is equal to column 3 - column 1, multiplied by a weighting factor. Column 8 is equal to column 5, multiplied by a weighting factor. These weighting factors are based on a Bayesian weighting scheme. The mean square iso difference in a range of resolution, less the mean square sigma of this, is a good estimate of the mean square true iso difference, Del**2, in this shell. The observed iso differences are weighted down by a factor equal to Del**2/(Del**2+sigma**2). If sigma is small relative to Del, the weight will be about 1. If sigma is big relative to Del, the difference will be weighted down. A similar approach is taken for the ano differences. Look at the Patterson maps you get with columns 7 and 8 and compare them to the ones you get using column 3 - column 1 and column 5.

Please note: The output of MADMRG is set up to be used as "Mock" native and derivative data. When you refine heavy atom parameters using this mock dataset, you must define a heavy atom type that has scattering factors identical to those you use in MADMRG at the "standard" wavelength. That is, if you define lambda 3 as "standard" in madmrg and f" at lambda 3 is 8.9933, then when you get to heavy atom refinement with routine HEAVY you will need to define an atom type "L3" (or something) that has all the right scattering factor information including an f" of 8.9933. Use the keyword NEWATOMTYPE in HEAVY to do this easily.

Also note: when you use this data in your MIR program, DO NOT refine an overall scale factor and B for the "derivative." The overall scale factor and B of the derivative relative to the (pseudo) native are absolutely perfect to start with (because of the way the derivative has been set up). In this package, use the flag "NOREFINESCALE" for the derivative.

The reason to use column 4 as sigma of Fder is that heavy atom refinement programs such as HEAVY assume that errors in Fp and Fder are independent. In this case they are not. Suppose you estimated the error in Fder by combining errors in Fo and deliso. Then your heavy atom refinement program would estimate the error in Fder-Fnat by combining the errors you give it for Fder (based on errors in deliso + Fo) and the errors you give it for Fnat (the error in Fo). The estimates of errors in Fder-Fnat would therefore contain the errors in Fo twice. If you use column 4 as sigma of Fder, the errors in Fder-Fnat will be correctly calculated based on deliso and Fo.

Input data file

This input data must be scaled carefully. MADMRG does not scale your data for you.

Number of protein residues and anomalous scatterers

The program assumes that the B-factor for the anomalously scattering atoms is similar to that for all other atoms. Using the number of protein residues and the number of anomalously scattering atoms on the next line, the program estimates the rms value of structure factor amplitudes due to anomalously scattering atoms as a function of resolution. Each of these are for the asymmetric unit.

MADBST: Calculation of Bayesian Fa values from MAD data

Madbst reads in MAD data from a dorgbn file. It estimates the components of the heavy (anomalously scattering) atom structure factor (Fa) parallel to and perpendicular to the structure factor for all non-anomalously scattering atoms. It also estimates the magnitude of Fa and its uncertainty.

Generally you will want to use "ANALYZE_MAD" to run "MADMRG" and "MADBST" for you. Ordinarily, you will take your scaled MAD data (the exact same data you have used for MADMRG) and use it with MADBST. The output of MADBST can be used in (1) Patterson maps, (2) difference fouriers, (3) direct methods, and (4) in automated structure determination with SOLVE in this package.

How MADBST works

Here is how MADBST works. MADBST first estimates rms heavy atom structure factor for anom scattering atoms at each wavelength used from form factor tables and # of atoms. Then MADBST goes through all the reflections and estimates probability distributions for the components of Fh parallel (Fha) and perpendicular (Fhb) to the structure factor corresponding to non-anomalously scattering atoms. Possible values of Fha, Fhb are deduced from rms value of Fh and Wilson statistics. Range of Fh tested are from -3 sigma to +3 sigma. Relative probabilities of each possible (Fha,Fhb) are calculated from Wilson statistics.

For each reflection, all above possiblities for (Fha, Fhb) are tested. For each (Fha,Fhb), values of Fp (structure factor for protein atoms only, assumed to be along the x-axis as we have no information on it) are tested. For any set of values of (Fha,Fhb) and Fp, it is easy to calculate values of Fbar and DelFano at each wavelength. The relative likelihood that a particular set of values of (Fha,Fhb) and Fp is correct is estimated (from Bayes' Rule), as the weighted residual: exp(- sum of (calc - obs)/sigma**2) for Fbar and DelfAno at all wavelengths. The a priori probability of (Fha,Fhb) is also included (Wilson statistics). The final "best" value of (Fha,Fhb) is just the weighted average over all possibilities. Similarly the "best" FH**2 is the weighted average of FH**2 over all possibilities (note that the best (FH**2) is not the (best FH)**2. Reflections for which no values of Fha,Fhb are likely (P<0.001) are rejected.

MADBST output

The output file is a copy of the input file, with 6 data columns appended to it, and with all reflections out of the resolution range or with no data tossed. The extra 6 data columns are:

ncol+1 -- <Fa cos theta> = Fh component along Fo weighted by figure of merit
ncol+2 -- <Fa sin theta> = weighted Fh component perpendicular to Fo
ncol+3 -- <Fa> = best estimate of Fa
ncol+4 -- sigma of <Fa>
ncol+5 -- sqrt(<Fa**2>) = sqrt of best estimate of Fa**2
ncol+6 -- sigma of sqrt(<Fa**2>)

The first two of these data columns can be used in difference Fourier maps to show the positions of anomalously scattering atoms. For example, if you have an estimate of phases for Fo, the non-anomalously scattering part of the structure from some partial solution, then you can add the phase angle for Fo on to the phase angle for Fa and draw a Fourier for Fa, the anomalously scattering atoms. This is done in SOLVE.

The third or fifth data columns can be used in a Patterson map (after squaring Fa) or in direct methods. SOLVE can use this Patterson map that you calculate.

Keywords for MADBST

(many are the same as for MADMRG):

```
nshells xx               # of shells of resolution to use (usu. 5 to max of 10)
nres  xx                 # of protein residues in the asymmetric unit
nanomalous  xx           # of anomalously scattering atoms in the a.u.

mad_atom xx               Name of anomalously scattering atom.  This generates
                          aval_mad, bval_mad and cval_mad for you.

lambda n                 wavelength # n (n=1,2,3..) for data to follow
ncolfbar n              column # of Fbar at wavelength 1
ncolsfbar n              column # of sigma of Fbar, wavelength 1
ncoldelf  n              column # of DelAno at wavelength 1
ncolsdelf n             column # of sigma fo DelAno at wavelength 1
                         ... (same for wavelength 2 etc...)
fprimv_mad  xx           1 real number for f' value for anomalously scattering
                         atom at the current wavelength.  Wavelength is defined
                         by the most recent value of the keyword "LAMBDA".
                         Note that f' and f" are required regardless of whether
                         you input aval_mad... or mad_atom
fprprv_mad  xx           1 real number for f" value for anomalously scattering
                         atom at the current wavelength.  Wavelength is defined
                         by the most recent value of the keyword "LAMBDA"

infile xxxx              Input data file (.drg file)
outfile xxxx             Output data file (.drg file, same as input except
                         reflections out of range are tossed and there are
                         6 additional columns of data)
```

Using SOLVE with MAD data analyzed with MADBST

If you want to phase the difference Fouriers calculated by SOLVE using both the anomalous and dispersive data data then you can specify,

NCOLFHCOS xx

NCOLFHSIN xx

which will, for lambda (1), use the value in ncolfhcos and ncolfhsin as estimates of the heavy atom structure factor components parallel to and perpendicular to the native structure factor. This is done automatically by ANALYZE_MAD if you are using automated structure determination. Here ncolfhcos(1) is identical to ncolfhcos. If you specify ncolfhcos(2) it refers to "derivative" 2.

The output of MADBST provides ncolfhcos(1) and ncolfhsin(1) as "<fa cos theta>" and "<fa sin theta>" (See MADBST writeup). You might also want to use a "combined" Bayesian patterson map as output by MADBST or an optimized difference Patterson map as calculated by MADMRG for your Patterson searches. If you want to specify a previously calculated Patterson map for lambda 1, use the command

PATTFFTFILE xxxxxx

where xxxxxx is the name of the FFT file containing this patterson. The FFT must have been calculated with this package using the same grid as currently specified. PATTFFTFILE is equivalent to PATTFFTFILE(1), where the 1 refers to lambda 1. This keyword will result in the use of file xxxxxx as the patterson for lambda #1.

Contents	Index

SCALE_NATIVE: Scaling native data

SCALE_NATIVE and SCALE_MIR are routines to read in MIR data from Scalepack output files or from other formatted files and to scale it and put it all in a file suitable for ANALYZE_MIR and SOLVE. They are ordinarily called as the first steps in a completely automated structure determination and are followed by ANALYZE_MIR and SOLVE. They can also be run on their own. The routines work best when the MIR data is reasonably complete. The routines assume that either:

- you have already merged your anomalous data and your data files contain 4 columns (I+,sigma,I-,sigma), or
- you have not merged the data and your data columns contain 2 columns (I, sigma).

(Note: the data can be amplitudes, not intensities if you read in the data with READFORMATTED and you specify the flag READ_AMPLITUDES)

For the second case, reflections with H K L corresponding to I+ and I- are treated as such. These choices are specified with the keywords "merged" or "unmerged". SCALE_NATIVE can read up to 4 data files and SCALE_MIR can read in up to 4 data files per derivative.

SCALE_NATIVE uses the first dataset you give it as a basis for setting the scaling of all the others. It reads in all the data, converts from I to F, maps it to the asymmetric unit, and uses the mapped data to localscale all the original unmapped datasets. The scaled datasets are then mapped to the asymmetric unit, merged, and written out to the file you specify with the keyword "scalednativefile" (default = "native_f.drg").

Sample script file for SCALE_NATIVE

```
!-----Command file to read in raw native data and scale it----

@solve.setup              ! get our standard information read in
logfile native.logfile    ! write out most information to this file.
rawnativefile native.int ! file with native data H K L Iobs Sigma usually
unmerged                  ! data have not yet been reduced to the a.u.
readdenzo                 ! data written by Scalepack
read_intensities        ! alternative is read_amplitudes
scalednativefile native_f.drg    ! output file with scaled native data
scale_native              ! scale this native dataset
!--------------------------------------------------------------------------------
```

Keywords for SCALE_NATIVE:

```
RAWNATIVEFILE xxx.int       File "xxx.int" will be read in as data for native



PREMERGED          The data in all RAWMADFILEs have H K L and 4 columns:
                   I+, sigma, I-, sigma

UNMERGED           The data in all RAWMADFILEs have H K L and 2 columns:
                   I, sigma

READDENZO          The datafiles are written by Scalepack.  For unmerged data
                   they will be read with the formatting:(6i4,i6,2i2,i3,2f8.0)
and
                   nsym*2+1 lines are skipped at the top of the file. For
                   merged data the formatting is:  (3i4,4f8.0) and
                   3 lines are skipped at the top of the file.

READFORMATTED      The datafiles were not written by Scalepack.
                    They will be read with "*" formatting.

READ_INTENSITIES     The data are intensities (default)
READ_AMPLITUDES    The data are amplitudes
                   (This is valid only with READFORMATTED)



NSKIP n              Skip exactly n lines at the top of each data file

NSKIP 0              Do not skip any lines at the top of each data file

NSKIP -1             Skip 0 lines at the top of each data file
                     unless the keywords READDENZO and PREMERGED
                     are set, in which case,skip NS*4+1 lines
                     where NS is the number of symmetry elements
                     in the space group (this is the # of lines in
                     a Scalepack header) [Default: NSKIP = -1]

RATMIN xx             Minumum ratio of F/sig to read in data for a
                      reflection at all is xx [default=0.0]. This is
                      useful for eliminating weak data.

OVERALLSCALE          Do not do local scaling; just an overall
                      scale factor for F+, F- at each wavelength.
                      Use this if you already have scaled the data
                      and you don't want any more scaling done.

SCALEDNATIVEFILE aa.drg       Scaled native data Fnat,sig will be written to and
                              read from "aa.drg" (default="native_f.drg")
```

SCALE_MIR: Scaling derivative data to a native dataset

SCALE_NATIVE and SCALE_MIR are routines to read in MIR data from Scalepack output files or from other formatted files and to scale it and put it all in a file suitable for ANALYZE_MIR and SOLVE. They are ordinarily called as the first steps in a completely automated structure determination and are followed by ANALYZE_MIR and SOLVE. They can also be run on their own. The routines work best when the MIR data is reasonably complete. The routines assume that either:

- you have already merged your anomalous data and your data files contain 4 columns (I+,sigma,I-,sigma), or
- you have not merged the data and your data columns contain 2 columns (I, sigma).

(Note: the data can be amplitudes, not intensities if you read in the data with READFORMATTED and you specify the flag READ_AMPLITUDES)

For the second case, reflections with H K L corresponding to I+ and I- are treated as such. These choices are specified with the keywords "merged" or "unmerged". SCALE_NATIVE can read up to 4 data files and SCALE_MIR can read in up to 4 data files per derivative.

SCALE_MIR reads in and converts the data from I to F, scales all the F- data in the derivative datasets to the F+ data, then it scales each derivative (F-, F+) to the native. All the scaling is done using LOCALSCALE. The F- and F+ data for the derivatives are separately merged and use to create two data files. The first contains Fnat,sigma, and (Fbar,sig,Delanom, sig) for each derivative. It is specified by the keyword "mirfbarfile" and is usually called "mir_fbar.drg". The other contains Fnat,sigma, and (F+,sig,F-,sig) for each derivative. It is specified by the keyword "mirfpfmfile" and is usually called "mir_fpfm.drg". If you don't want to use anomalous differences later, you can specify the "noinano" keyword for that derivative in SOLVE. At the end of the output for SCALE_MIR will be a summary of the scaling R-factors for isomorphous and anomalous differences and of the completeness of the datasets.

Sample script file for SCALE_MIR

```
!-----Command file to read in raw derivative data and scale it to the native----

@solve.setup                      ! get our standard information read in
logfile deriv.logfile             ! write out most information to this file.

scalednativefile native_f.drg       ! you need a native to scale the derivatives...

derivative 1                      ! about to enter information on derivative #1
rawderivfile deriv1.int           ! the derivative data is in deriv1.int

derivative 2                      ! next derivative
rawderivfile deriv2.int

unmerged                          ! data have not yet been reduced to the a.u.
readdenzo                         ! data written by Scalepack
read_intensities               ! data are intensities (alternative is
read_amplitudes)
scale_mir                        ! scale the derivs to the native
!-------------------------------------------------------------------------------
```

Keywords for SCALE_MIR:

```
RAWNATIVEFILE xxx.int       File "xxx.int" will be read in as data for native

DERIVATIVE n                The next RAWMIRFILE(s) that are read in will be
                            for derivative n

RAWDERIVFILE xxx.int          File "xxx.int" will be read in as data for
                            the current derivative

PREMERGED          The data in all RAWMADFILEs have H K L and 4 columns:
                   I+, sigma, I-, sigma

UNMERGED           The data in all RAWMADFILEs have H K L and 2 columns:
                   I, sigma

READDENZO          The datafiles are written by Scalepack.  For unmerged data
                   they will be read with the formatting:(6i4,i6,2i2,i3,2f8.0) and
                   nsym*2+1 lines are skipped at the top of the file. For
                   merged data the formatting is:  (3i4,4f8.0) and
                   3 lines are skipped at the top of the file.

READFORMATTED      The datafiles were not written by Scalepack.
                    They will be read with "*" formatting.

READ_INTENSITIES     The data are intensities (default)
READ_AMPLITUDES    The data are amplitudes
                   (This is valid only with READFORMATTED)


NSKIP n              Skip exactly n lines at the top of each data file

NSKIP 0              Do not skip any lines at the top of each data file

NSKIP -1             Skip 0 lines at the top of each data file
                     unless the keywords READDENZO and PREMERGED
                     are set, in which case,skip NS*4+1 lines
                     where NS is the number of symmetry elements
                     in the space group (this is the # of lines in
                     a Scalepack header) [Default: NSKIP = -1]

RATMIN xx             Minumum ratio of F/sig to read in data for a
```

```
                      reflection at all is xx [default=0.0]. This is
                      useful for eliminating weak data.

FPFM_ONLY             Toss all acentric reflections where either F+
                      or F- is missing.

FP_OR_FM              Use F+ or F- as an estimate of Fbar if F+ and
                      F- are not both present. This is useful if
                      your data is not that complete. It is
                      better to obtain complete data however.
                      This is the default for MIR data.

OVERALLSCALE          Do not do local scaling; just an overall
                      scale factor for F+, F- at each wavelength.
                      Use this if you already have scaled the data
                      and you don't want any more scaling done.

SCALEDNATIVEFILE aa.drg       Scaled native data Fnat,sig will be
                              read from "aa.drg" (default="native_f.drg")

MIRFBARFILE xx.scl         Output file with Fnat,sig, and (Fbar,sigma,DelAno,sigma)
                           for each derivative will be xx.scl
                           (DEFAULT="mir_fbar.scl")

MIRFPFMFILE yy.scl         Output file with Fnat, sig, and (F+,sigma,F-,sigma)
                           for each derivative will be yy.scl
                           (DEFAULT="mir_fpfm.scl")
```

Contents	Index

ANALYZE_MIR: Analyzing an MIR dataset

ANALYZE_MIR is a routine to calculate difference Pattersons and anomalous difference Pattersons for an MIR dataset, to calculate correlation coefficients among them all, and to set up a "solve_mir.script" file that can be used to run SOLVE on the dataset. You usually do not have to worry about this routine at all because it is ordinarily called right after running SCALE_MIR for automatic structure determination. It is ordinarily followed by running SOLVE with the solve_mir.script file written out by this routine or by using the keyword "SOLVE" after running this routine.

ANALYZE_MIR assumes that you have a datafile ("mir_fbar.scl") that contains Fnat, sigma, and (Fbar,sigma,DelAno,sigma) for each derivative of MIR data. That is, there are exactly 2 columns of data for the native and 4 data columns for each derivative. This is a dorgbn file. It is ordinarily created by SCALE_MIR, but you can create your own if you like.

To run ANALYZE_MIR, you need to input your standard setup file ("solve.setup"), and the name of the mir datafile. ANALYZE_MIR will calculate origin-removed difference Pattersons for all isomorphous and anomalous differences. These maps are all compared to each other and the correlation coefficients are displayed in a table. In a typical MIR experiment the anomalous and isomorphous difference Pattersons have correlations with each other on the order of 0.0 to 0.2 or so (pretty low, so don't be worried).

ANALYZE_MIR writes out a script file "solve_mir.script" that you can use to go on with the SOLVE routine to solve this mir structure. You can edit this script file to modify it if you like or you can run it as is. You can also go right on with SOLVE by adding the command "SOLVE" after "ANALYZE_MIR".

Please note that there is a slight difference between running ANALYZE_MIR right after SCALE_MIR and in a separate session. If you run them right after each other, you should input all the information about each derivative at once, as in the sample script for automatic analysis of MIR data. (If you wanted to input information about derivative #1 in two places in your script file, you would need to use the keyword "gotoderiv 1" the second time. You can avoid this by putting all the information for derivative #1 together.)

Script for ANALYZE_MIR

A typical input script for ANALYZE_MIR follows:

```
!-------------------------Run ANALYZE_MIR--------------------------------

@solve.setup                       ! standard information for this crystal
mirfbarfile mir_fbar.scl           ! input dorgbn file with Fnat,sig, and
                                   ! (Fbar,sig,Delano,Sig) for each wavelength
logfile analyze_mir.logfile        ! write out most information to this file

derivative 1                       ! derivative #1 information is to follow
label deriv #1 Hg                  ! label for deriv 1
atomname HG                        ! this is an HG derivative
```

```
derivative 2
label deriv #2 Iodine
atomname I-

ANALYZE_MIR                                   !analyze all the Pattersons and
                                              !write solve_mir.script

!------------------------------------------------------------------------------
```

Keywords for ANALYZE_MIR:

```
LOGFILE xx.logfile                  log file for output will be xx.logfile

mirfbarfile xx.scl                   input dorgbn file with Fnat, sig, and
                                     (Fbar,sig,Delano,Sig)
                                     for each wavelength will be xx.scl
                                     [Default="mir_fbar.scl"]

SCRIPTFILE xxx                      Output script file containing instructions
                                    for running SOLVE written to xxx
                                    [default="solve_mir.script"]. Starting SOLVE with
                                    this script is equivalent to following
                                    ANALYZE_MIR with SOLVE.

derivative n                        derivative number (n=1,2,3...) for values
                                    to be entered next of atomname and label

label xxxx                          xxx is a label for this derivative

ATOMNAME XXXX                       XXXX is the atom name of the atoms in
                                    this derivative.  This name can be in
                                    SOLVE's database or you can
                                    enter it using the NEWATOMTYPE keyword.
                                    PLEASE NOTE: the f' and f" values assumed
                                    by SOLVE are for Cu Kalpha radiation (1.54 A).
                                    If you collected your MIR data at a synchrotron
                                    then you will want to define a new atom type
                                    with the appropriate values of f' (fprimv) and
                                    f" (fprprv).  See the HEAVY or SOLVE keyword
                                    lists for entering a NEWATOMTYPE.
                                    The atom types recognized by SOLVE are:
                                    H, H-1, He, Li, Li+1, Be, Be+2, B, C, Cv, N, O, O-1,
                                    F, F-1, Ne, Na, Na+1, Mg, Mg+2, Al, Al+3, Si, Siv, Si
+4,
                                    P, S, Cl, Cl-1, Ar, K, K+1, Ca, Ca+2, Sc, Sc+3, Ti,
Ti+2,
                                    Ti+3, Ti+4, V, V+2, V+3, V+5, Cr, Cr+2, Cr+3, Mn, Mn
+2, Mn+3,
                                    Mn+4, Fe, Fe+2, Fe+3, Co, Co+2, Co+3, Ni, Ni+2, Ni
+3, Cu,
```

Cu+1, Cu+2, Zn, Zn+2, Ga, Ga+3, Ge, Ge+4, As, Se, Br,
Br-1, Kr, Rb, Rb+1, Sr, Sr+2, Y, Y+3, Zr, Zr+4, Nb,
Nb+3,
Nb+5, Mo, Mo+3, Mo+5, Mo+6, Tc, Ru, Ru+3, Ru+4, Rh,
Rh+3,
Rh+4, Pd, Pd+2, Pd+4, Ag, Ag+1, Ag+2, Cd, Cd+2, In,
In+3,
Sn, Sn+2, Sn+4, Sb, Sb+3, Sb+5, Te, I, I-1, Xe, Cs,
Cs+1,
Ba, Ba+2, La, La+3, Ce, Ce+3, Ce+4, Pr, Pr+3, Pr+4,
Nd,
Nd+3, Pm, Pm+3, Sm, Sm+3, Eu, Eu+2, Eu+3, Gd, Gd+3,
Tb,
Tb+3, Dy, Dy+3, Ho, Ho+3, Er, Er+3, Tm, Tm+3, Yb, Yb
+2,
Yb+3, Lu, Lu+3, Hf, Hf+4, Ta, Ta+5, W, W+6, Re, Os,
Os+4,
Ir, Ir+3, Ir+4, Pt, Pt+2, Pt+4, Au, Au+1, Au+3, Hg,
Hg+1,
Hg+2, Tl, Tl+1, Tl+3, Pb, Pb+2, Pb+4, Bi, Bi+3, Bi
+5, Po,
At, Rn, Fr, Ra, Ra+2, Ac, Ac+3, Th, Th+4, Pa, U, U
+3, U+4,
U+6, Np, Np+3, Np+4, Np+6, Pu, Pu+3, Pu+4, Pu+6, Am,
Cm, Bk, Cf

NSHELLS n	Number of shells for analysis is n [default=10]

HASSP: Patterson searches by the superposition method

[Script for HASSP | More about HASSP | Analyzing a solution]

HASSP is a program for automatically searching for solutions to a Patterson function. It is used by SOLVE as a means of generating plausible starting solutions for a MAD or MIR dataset. You can use it to analyze any Patterson or difference Patterson function that you have calculated with this package. Although HASSP is good at finding possible solutions to a Patterson, it is not as good as SOLVE at evaluating these solutions. It is a good idea to run HASSP on your Patterson functions to get an idea of what they look like, but you should really run SOLVE to get a complete solution to your dataset.

Using HASSP is very easy. Here is a script file that will run HASSP on a patterson that you have calculated and put into "patterson.patt".

Script for HASSP:

```
!------------------Script for running HASSP -------------------------------
@solve.setup                     !  set up those standard keywords
fftfile patterson.patt           !  name of file with the patterson
logfile hassp.prt                !  write out the results to "hassp.prt"
hassp                            !  run hassp
!--------------------------------------------------------------------------
```

HASSP will then analyze your patterson in "patterson.patt" and will write out a sorted list of likely solutions to this patterson.

More about HASSP

The HASSP routine uses a space-group symmetry minimum method to obtain sets of atomic sites consistent with a patterson function. The usual procedure followed in using this program is to search for single-atom solutions to patterson function, then to search for two-atom solutions to the patterson function. The two-atom solutions are obtained from two sources: combination of single-atom solutions (after figuring out origin shifts and translations along polar axes), and cross-vector searches. The idea behind a cross-vector search is that many of the peaks in general positions of a patterson are cross-vectors between sites. If you know the cross-vector between two sites then once you know the position of 1 site you know the position of the second. Considering a particular cross-vector, HASSP tests all possible positions for atom 1, generates the position of atom 2 and all the predicted peaks it the patterson. These are compared to the patterson itself and the solution is scored.

Searching for single-site solutions to patterson function.

A map is calculated over the range supplied (XS-XE; YS-YE; ZS-ZE), except that search is not carried out over axis which are not fixed (all three in space group P1). The value of the map at each grid point is the minimum of values of the patterson functon at the (NEQUIV-1) Harker vectors corresponding to this grid point. Peaks in this map are stored, sorted according to symmetry, and listed after elimination of redundancies.

For points in general positions, the peak height listed is simply the minimum value of the patterson function at the (NEQUIV-1) Harker vector associated with this point. For points in special positions, the listed height is the minimum of the values of the patterson function at each of the (NEQUIV-1) Harker vectors divided by the number of times a Harker vector associated with this point falls on that position. For example, in space group P222, an atom at (x,y,z) yields Harker vectors (0,0,0), (2x,2y,0), (2x,0,2z), and (0,2y,2z). If x=0 and y=0, though, (0,0,0)=(2x,2y,0) and (2x,0,2z)=(0,2y,2z) and there is only one unique Harker vector (excluding the origin), which is repeated twice. The value of the peak height listed would be 1/2 the height at (0,0,2z).

The probability that a given peak of height A in this function is due to a random combination of peaks is roughly given by:

P=(1.- (1.- p(A)**M)**N) , where,

- A= minimum value of (value of patterson function at Harker vectors divided by expected noise at that position).
- p(A) is probability of observing a value of A or higher on a given try.
- M=number of independent Harker vectors examined for this peak
- N=number of independent grid points used in search for peaks.

The noise in the map is taken to be the RMS value of the patterson function if this is a general position. If it is a position of higher symmetry, the noise = sqrt(SIGMA) * the symmetry number of this position. The number of independent grid points used in the search for peaks would roughly be equal to the number of reflections used to make up the map if reflections at all resolutions contributed equally. A better estimate of this numbr is probably the number of peaks+valleys in the patterson map. In this routine, we actually use 2* the number of peaks.

The grid used for all searches is exactly the same as the grid for the input patterson map, but each time a peak is found, all neighboring grid points are tested on a grid twice as fine and the highest of these test values is used. Values of the patterson function between grid points are interpolated. Do not use a grid coarser than 1/3 the resolution for the input patterson map. Also don't bother to use a grid finer than 1/6 the resolution. NOTE: the input patterson map must be on a grid such that the symmetry elements lie on grid points. That is, if there is a two-fold axis at 1/12 in z, then the z-axis must be divided into a number of grid points that is a multiple of 12. The easiest way to be safe is to make sure all unit cell translations are multiples of 12.

Significance tests

Difference patterson functions have a considerable amount of noise if acentric reflections are present. (For each acentric reflection, the expected error [|Fph-Fp| - |fh|] is roughly equal to |Fph-Fp|). It can be shown that SIGMA, the RMS noise in the map is roughly equal to the RMS value of the patterson function.

Peaks in the patterson map which have a height much less than SIGMA are therefore likely to be unrelated to atomic sites. On special positions, the RMS noise in the map will be sqrt(NSYM)*SIGMA, where NSYM is the symmetry number of this position.

In order not to include too many peaks due to this noise in any of the searches carried out, a significance test is made for each peak if ISIGNF=0 (default). A peak is rejected if there is a probability less than SIGNIF that no peak of this height or higher would occur by chance in this search.

Symmetry numbers of positions in real and patterson cells

For this program, the symmetry number of a position in a real or patterson cell is the number of ways that a symmetry operator in the group (patterson or real cell) can map the point onto itself (within a toler- ance of 2 grid units). The symmetry number of a general position is 1, for a point on a dyad, it is 2, etc..

Local symmetry

Local symmetry is not yet implemented in SOLVE

.

Obtaining an analysis of a solution that you input yourself.

In order to generate a list of local symmetry-related points and minimum self- and cross-vectors corresponding to a set of unique sites you specify, use ITYPE = -6 with a very small search region (keyword searchregion; if you set it to zero, though, it will replace your zeroes with the asymmetric unit of the cell as defined by your FFTGRID).

If ITYPE=-6 is specified along with a small (but non-zero) search region, an analysis of the sites you input on lines 9a-... will be printed. This analysis includes the minimum values of the self- and cross-vectors for this set of sites. This procedure will help you determine if there is anything unusual about your map.

HEAVY: Heavy atom refinement and phasing

HEAVY is a general-purpose heavy atom refinement routine. It can be used to carry out either phase refinement or origin-removed Patterson refinement, as well as to calculate coefficients for native Fourier and difference Fourier maps. Ordinarily you will use HEAVY as part of an automated structure solution with SOLVE. In this case SOLVE will write out a "phases-hl.script" script file that you can edit and use for further refinement. This should usually mean that you will not have to generate the rather lengthy inputs to HEAVY by yourself.

Keywording inputs are most conveniently entered using a script file. Note that any values previously defined do not need to be specified. If you run HEAVY a second time without quitting the main program and do not specify any new parameters, the routine will start where it left off and carry out another set of refinements of the same type that you specified the last time you ran it. Note also that average residuals are maintained throughout. This means that if you want to refine a completely new set of data, you should start the program over.

An easy way to get a file containing all the keywords you can use for HEAVY is to edit the file generated by running HEAVY specifiying "newfile heavy.new".

MAD phasing with HEAVY

There are 3 ways to phase MAD data using HEAVY.

(1) You can use MADMRG to compress MAD data to data that look like SIR+anomalous data at one wavelength (e.g., L1), then refine heavy atom parameters and phase just as if you did have SIRAS data. This gives you F and phase for the structure without the anomalously scattering atoms.

(2) You can use MADMRG and then Bayesian phasing. This is what SOLVE does. In this case you convert MAD data to SIRAS using MADMRG, then refine heavy atom parameters all as in #1. Then you use these heavy atom parameters with the original MAD data to phase it using Bayesian MAD phasing. As you have already refined the heavy atom parameters at L1, you do not need to redo the refinement at L2, L3 because they are all the same. You simply run HEAVY again, specifying the keyword IMADPHASE n (where n=the wavelength # for which heavy atom structure factors are to be calculated), and specifying REFINENONE for each heavy atom site. You also need to specify a new input file that has (instead of the MADMRG data or FBAR,DELANO data) the complete F+,sig+,F-,sig- scaled MAD data. This can be the data file used to create the Fbar,delano data file. You specify column numbers with

NCOLFPLUS, NCOLSIGPLUS, NCOLFMINUS, NCOLSIGMINUS. Derivative 1 is now your L1 data, derivative 2, L2, etc. You include the refined heavy atom parameters for your L1 derivative and dummy atoms with the correct scattering factors for the L2 and L3 derivatives. The dummy atoms are used to calculate scattering at these other wavelengths; they are not used in phasing per se. You don't even have to put in any occupancies or xyz for these atoms. In this way, you have 1 set of heavy atom parameters that are applied to all 3 wavelengths. If your heavy atoms are not selenium at L1, L2, and L3, you will need to use the NEWATOMTYPE keyword to input their scattering factors.

The SOLVE routine with MAD data does all this without even leaving SOLVE. You can also run HEAVY once after MADMRG, then edit the HEAVY.NEW (or whatever you have called it) file as described above, then run HEAVY again with IMADPHASE specified.

If you specify IMADPHASE 1 then your output map will be calculated at the lambda of "derivative" 1, if you specify n then it will be at the wavelength of dataset n. The phasing will be the same in any case. Note that the value of "n" you specify will determine which heavy atom values are used in the MAD phasing calculation. If you specify "2" then the heavy atom parameters that you type in for derivative 2 will be used.

(3) Refinement as if the MAD data were MIR data. In this case, you choose one wavelength (the one with a small f" and the most negative f' usually, often called "L1") as "native" and treat the other wavelength data as derivatives. In this case you will need to define new "atoms" with the NEWATOMTYPE keyword that have values of f" that are actual values, but values of f' that are the difference between the value at that wavelength and the value at L1, and values of all the other parts (a1, b1, etc) of zero. For example, if the L1 values of f' and f" are -9.6 and 2.2, and at L2 they are -7.6 and 5.8, then you need a new atom type as follows for the f' difference:

```
NEWATOMTYPE L2L1
AVAL      0.00000000     0.00000000     0.00000000     0.00000000
BVAL      0.00000000     0.00000000     0.00000000     0.00000000
CVAL      0.00000000
FPRIMV      2.0
FPRPRV      5.8
```

You then use "L2L1" as your atomname for heavy atoms in derivative 2 (L2). If you have 3 wavelength MAD data, you now have 1 native and 2 derivatives. You can refine the heavy atom parameters of the 2 derivatives in just the usual way and obtain phases for the L1 data (including the heavy atoms) as if this were MIR data.

Refinement against the origin-removed Patterson map

Refinement against an origin-removed Patterson map is a way of refining heavy atom parameters of each derivative independently, and is particularly useful because the occupancies of heavy atom sites are quite accurately estimated and because the refinement is very fast. When using this package, the recommended refinement method is this one, with JALT=0 and KALT=0.

This refinement minimizes the sum over all reflections of, R = WGT * DEL**2

with respect to heavy atom parameters. WGT is a weighting factor, and DEL is defined as:

DEL = (Fph-Fnat)**2 - K*FH**2 - < (Fph-Fnat)**2 - K*FH**2 >

where the average <> is taken in a shell of resolution and FH is the magnitude of the calculated heavy atom structure factor. K is 1 for centric reflections, 1/2 for acentric reflections.

Rejecting data with large Del F's:

HEAVY uses all the data that you give it that satisfy the criteria of minimum F, FOM, etc that are set above. If you want it to reject data with especially large Del F, then you need to specify this when you scale the data with LOCALSCALE. There is an option in localscale to "reject large del F" (TOSSBAD). Use this to get rid of the large Del F before going into HEAVY.

Interpreting statistical output from HEAVY

Many of the values listed at the end of a set of refinements are more-or-less self explanatory. This should include the number of reflections read, within resolution limits, and greater than the minimum figure of merit. As these statistics are usually printed for a cycle in which refinement is not carried out, the number of reflections used to refine is usually zero in this listing.

The statistical output for MAD phasing using Bayesian correlated MAD phasing is not as intuitive as the output for MIR phasing because the standard phasing statistics do not really apply. If are running SOLVE in automatic format and you want approximate phasing statistics, you can run SOLVE specifying "NOBAYES". This will suppress the Bayesian correlated MAD phasing at the end of SOLVE and use SIRAS phasing which isn't quite as good but for which the statistics are easy to understand.

Other values listed at the end of a set of refinements include:

RMS HEAVY ATOM F: The rms value of the calculated heavy atom F in the resolution range

RMS PHASE AVG'D RESIDUAL: This is the rms value of the difference between calculated and observed derivative F, where it is averaged not only over all reflections, but over all phases for each reflection, weighted by the phase probability

RMS(FH)/RMS(E): This is the ratio of the rms heavy atom F to the rms phase averaged residual

CENTRIC R FACTOR: This is <| |Fder-Fnat| - |FH| | >/< |Fder-Fnat| >

RMS DERIVATIVE F: This is the rms value of Fder

RMS SIGMA OF FPH: This is the rms sigma of Fder

RMS SIGMA OF FP: This is the rms sigma of Fnat

RMS OBSERVED DIFFERENCE: For anomalous differences, this is the rms value of DelAno= (F+ - F-)

RMS CALCULATED DIFFERENCE: This is the rms calculated anomalous difference

MEAN RATIO OF ISO TO ANO: This is the ratio of calculated |FH| due to normal scattering relative to that due to anomalous scattering. If all anomalous scatterers are identical, this is equal to (f+f')/f" for that anomalous scatterer.

RMS(RES HA SF+LACK OF ISO SF): This is an estimate of the total errors in the heavy atom model plus lack of isomorphism that remain. It is obtained from the rms phase averaged residual and the rms native and derivative sigmas.

RMS LACK OF ISOMORPHISM SF: This is an estimate of the remaining lack of isomorphism. It is based on a comparison of the anomalous and isomorphous differences that remain

RMS RESIDUAL HEAVY ATOM SF: This is an estimate of the remaining heavy atom structure factor, based on the anomalous differences and the errors in measurement.

CENTRIC LOC: This is an estimate of the "centric" lack-of-closure residual, obtained using both centric and acentric reflections and correcting acentric lack-of-closure residuals by a factor of 2. These residuals are all corrected for errors in measurement, so that if the derivative is "solved" and there is little lack of isomorphism, these values should all be near zero.

ANOMALOUS LOC: This is the lack-of-closure error for anomalous differences, corrected for errors in measurement.

Correlated phasing statistics

If you specify the keyword "CORRELPHASE" then HEAVY will use a routine in phasing that takes into account the correlations in non-isomorphism errors among the derivatives. The derivatives must be grouped into sets with correlated errors. You can specify this grouping using IEGROUP (see below) or you can let HEAVY group them for you using the flag GETGROUPS. Note that correlated phasing makes a major improvement in the phasing power of a set of derivatives if the errors are highly correlated (>50%). If they are not highly correlated, the routine yields essentially the same results as the standard routine.

Whether or not correlated phasing is being used, the correlation of errors among derivatives is analyzed by HEAVY. An example of part of a log file that illustrates this is shown below:

```
--------------------------example------------------------------------
 Analysis of correlated modeling and non-isomorphism errors
 obtained using phased residuals.
 The derivatives were grouped into 1 sets  where the members of a set
 had some mutual correlation.


 Set 1 contains derivatives  1 2 3

 SUMMARY OF CORRELATED ERRORS AMONG DERIVATIVES

 DERIVATIVE:           1
 CENTRIC REFLECTIONS:
 DMIN:            ALL     10.81   6.94   5.46   4.65   4.11   3.73   3.43   3.20
 RMS errors correlated and uncorrelated with others in group:
      Correlated:  363.5  322.2  291.7  253.8  458.7  434.5  371.2  404.0  337.8
    Uncorrelated:  285.9  362.3  340.3  292.9  288.9  279.9  229.0  201.9  174.7

 Correlation of errors with other derivs:
 DERIV 2:           0.83   0.66   0.76   0.70   0.88   0.89   0.89   0.98   1.00
 DERIV 3:           0.74   0.59   0.64   0.51   0.78   0.83   0.82   0.86   0.95
--------------------------------------------------------------------------------
```

In this example, there are 3 derivatives, all in the same group (IEGROUP=1 for each). Of the lack-of-closure errors for derivative 1, most (363.5, arbitrary units) were correlated with derivatives 2 and 3, and some (285.9) were unique to this derivative. The overall correlation of errors with derivatives 2 and 3 were 83% and 74%, respectively. In fact, correlated phasing made a major improvement in the phasing for this group of derivatives.

Normal refinement/phasing cycles.

A. Refinement vs. origin-removed Patterson map.

Input parameters: all defaults used

NCYCLE = 1 to 30

IREFCY(I) = 1,1,1,2,2,2.....6,6,6,0

results:

Zeroth cycle: phases calculated for all derivatives identified with INPHASE using input lack-of-closure residuals. New lack-of-closure residuals are calculated for these derivatives. Statistics are printed.

Cycles 1 through NCYCLE-1: in this example, IREFCY(I) is zero onlast cycle, but non-zero for all other cycles. For each cycle when IREFCY(I) is non-zero: no phases are calculated no new residuals are calculated derivative IREFCY (I) is refined as described above

Note that only 1 derivative is refined at a time and all are independent. Therefore in polar space groups, the coordinate

(s) of at least one atom in each derivative must be fixed. In space group P1 parameters for a single heavy atom may not be refined at all. If two atoms are present, the occupancy, xyz, B of one of them only may be refined. If you use IHEAVYPROC then all this is taken care of for you (in P1 SOLVE will refine with phase refinement if any derivs have fewer than 3 sites, otherwise it will use patterson refinement).

Cycle NCYCLE: IREFCY(NCYCLE)=0 in this example, so this cycle is like the zeroth cycle: phases are calculated, new residuals calculated. If KOUT is non-zero, output data are calculated as well.

B. Refinement by minimization of lack-of-closure at most probable phase.

Input parameters: all default except JALT=1, KALT=0.

Results: identical to the above example except:

(1) phases will be calculated every cycle

(2) derivatives will be refined by minimization of (Fph-Fc)**2

This is not the recommended manner of using HEAVY in this package. In most circumstances origin-removed Patterson refinement is much more accurate. There are some instances in which phase refinement may be useful, however. One is when it is necessary to correlate the origins in different derivatives. In space group C2, for example, the y-coordinate is indeterminate. That means that if you have two derivatives and refine them independently, you will not have refined the relative y-coordinates of the atoms in the two derivatives (though you will have refined the relative y-coordinates of atoms within each derivative). You might wish to use phase refinement to carry this out, using one derivative to phase and refining y-coordinates in the other derivative. In practice, however, these relative y-coordinates can be obtained even more accurately by simply calculating a difference Fourier for one derivative, phasing with the other derivative. The centroid of the peak corresponding to the heavy atom site (which can be found, for example, by PEAKSEARCH in this package) will give you the relative y-coordinate you need with very good accuracy, and refinement of this coordinate is unnecessary. This is how SOLVE does this.

Note that still only 1 derivative may be refined at one time. (If you really want to phase only once per refinement of all derivatives, calculate phases during one run and write them out with KOUT=7. Then merge file containing phases with input DORGBN file (3 extra columns). Then run HEAVY with INPHAS=0 for each derivative and specifying INOLD=1. Also set INRESD=-1. The program will then use the input phases during phase refinement if JALT=1. Its probably faster to just phase each time.)

C. Just calculating phases and a map or other output.

Input: all default, except NCYCLE >0

If the input lack-of-closure residuals are ok., you can set INRESD = -1 so that new residuals will not be calculated and a zeroth cycle will not be included. Otherwise leave INRESD = 0.

Specify the type of map with KOUT, the derivative (if applicable) with KDER.

D. Carrying out a procedure with IHEAVYPROC. Heavy has the capability of carrying out an ordered sequence of refinements. These are useful if you want to carry out refinement in a semi-automatic fashion. When you specify a procedure with iheavyproc, you need to specify all the parameters that you want refined at all. Then the procedure you choose decides which parameters to refine on which cycles. Usually you will specify REFINEALL for all atoms, then let the procedures decide which to refine. If you use a procedure, the program will automatically fix all coordinates that cannot possibly be refined. For example, in space group C2 one atom in each derivative must have y fixed if origin-removed Patterson refinement is used, because the y-direction is polar. The program will fix the coordinate(s) of the atom that is the strongest in each derivative. If you have already fixed the coordinate(s) of an atom in a derivative (by not specifying that they be refined) then the program will just fix the atom you chose and not fix any others.

Note that you can carry out any series of refinements that you wan by setting up all your keywords for the first type of refinement, initiating refinement with the command HEAVY, then going back to KEYWORD mode, specifying the next type of refinement without changing or setting any other parameters unless you want to, then initiating the next refinement cycles with HEAVY, and so on. For example, you might type in all your heavy atom parameters, finishing with

```
...
NREP 5
IHEAVYPROC 2
!  now refine 5 cycles with iheavyproc=2
HEAVY


NREP 7
IHEAVYPROC 4
! now refine 7 cycles with iheavyproc=4
HEAVY
```

This sequence of commands results in 5 cycles of refinement of xyz of all atoms that you specified refinement of xyz, then 7 cycles of refinement of xyz,occ, and B of all atoms that you specified these parameters to be refined in. You can do this sort of thing in any order and ad infinitum if you wish.

Note that there is no procedure to refine just thermal factors. With this package there is no need to alternately refine occupancies and thermal factors. If there is insufficient data (i.e., very low resolution) to refine both occupancies and thermal factors, then set the thermal factors to any reasonable value and just refine the occupancies.

Changing heavy atom parameters after you have gone on to the next atom or derivative

If you want to change which parameters for which atoms are refined after you have already set up the atoms and refinement parameters, then you have to use a special way to reset them. The reason you have to do something special is that if you say "DERIVATIVE" then the routine assumes you are inputting data for a new derivative, so you can't go back to a previous one with that command. Instead, you type:

```
GOTODERIV 2    ( to go to derivative #2)
GOTOATOM 3     (to atom #3 in deriv #2)
REFINENONE     (set all refinement flags back to zero)
REFINEXYZB     (or whatever you want to refine for this atom)

GOTOATOM 1    ( now do atom 1 in deriv 2)
GOTODERIV 1    (now do derivative 1)
```

Contents	Index

Local scaling and merging of data

[Script for localscale | Keywords for localscale]

[Merge | Keywords for Merge | More on Merge]

[Complete]

LOCALSCALE

LOCALSCALE is a routine to scale a "derivative" dataset to a "native" dataset using local scaling. In this method the scale factor for a particular reflection is based on the ratio of derivative:native for reflections surrounding this reflection. This method is useful because the scale factor is not restricted to any particular function of position in reciprocal space.

In this implementation, at least 30 reflections surrounding the reflection to be scaled are used to obtain a scale factor. Additionally, the reflections used in obtaining a scale factor are always chosen so that they form a complete sphere around the reflection of interest (inasmuch as possible). Initial Wilson scaling is carried out before local scaling.

Data files: The program expects to read in two data files: one for the native dataset and one for the derivative. The two files may in fact be the same if desired. The native dataset is expected to have h,k,l, F and sigma (at least). The derivative dataset is expected to have h,k,l, F, and sigma, and, if desired, del F ano and sigma of del F ano. The scale factor obtained for the derivative F is applied to all of the derivative data.

A dorgbn-style file is written out containing the scaled derivative data. If you wish to have the derivative and native data in the same file, then follow this with the routine "FILEMERGE" and merge the two files. The output data file is NOT mapped to the asymmetric unit. Ordinarily you will want to follow LOCALSCALE with MERGE to merge the symmetry-related reflections and map everything to the asymmetric unit. You may need to run MERGE on your native data as well, to map it to the asymmetric unit.

Sample script to localscale der.drg to nat.drg:

```
!---------------Script for localscaling of derivative F to native F -----
@solve.setup                  ! standard parameters for this dataset
infile nat.drg                ! native in infile
nnatf 1                       ! column for native F
nnats 2                       ! column for native sigma
infile(2) der.drg             ! derivative in infile(2)
nderf 1                       ! column for deriv F
nders 2                       ! column for deriv sigma
outfile der.scl               ! output file
```

```
localscale                    ! do local scaling
!---------------------------------------------------------------------------
```

Keywords for LOCALSCALE

```
NSHELLS n         number of shells of resolution used to group data(default=10)
INFILE(1) xx      file with Native which is not further scaled
INFILE(2) xx      file with Derivative data to scale to Native
OUTFILE xx        output file with scaled derivative data
NNATF n           column # for F of native data
NNATS n           column # of sigma of F of native data
NDERF n           column # for F of deriv data
NDERS n           column # for sigma of F of deriv data
NANOF n           column # of anomalous difference (Fplus-Fminu) of deriv data
NANOS n           column # of sigma of anomalous difference
note: be sure to set those columns you don't want to 0

FILETITLE         optional title for output file

KEEPALL           keep reflections even with high differences
TOSSBAD           (default)Toss reflections if differences between native and
                  derivative are more than 3 * the rms found for other
                  reflections.
                  Note: KEEPALL and TOSSBAD apply to MERGE, LOCALSCALE,
                  SCALE_MAD, SCALE_MIR, SCALE_NATIVE. This is the
                  place to reject derivative reflections with very large del F
                  if you want to reject them at all.
ANCUT             minimum # of reflections to use to scale a reflection (30.)
RATMIN            minimum ratio of F/sigma to include (default=2)
NOBFACTOR         if specified, do not apply overall Wilson scaling before
doing
                  local scaling. Generally used only along with DAMPING=0.
BFACTOR           undoes NOBFACTOR. Do apply Wilson scaling before local scaling
DAMPING xx        scale factor (after Wilson scaling) is damped by taking it
                  to the power xx. Generally used with NOBFACTOR and a value of
                  0 to not do any scaling at all.
NODAMPING         undoes DAMPING by resetting damping factor to 1.0
OVERALLSCALE      just get 1 scale factor for the whole dataset. No local
                  scaling, no wilson scaling. Same as NOBFACTOR + DAMPING 0.0
NOOVERALLSCALE    undoes OVERALLSCALE. SAME AS BFACTOR + DAMPING 1.0
```

More on localscale:

1. A value of 0 or less for fnat or fder is assumed to mean data are not measured. A value of 0.0 or -1.0 for del f ano is assumed to mean the data are not measured also.

2. If sigmas are not supplied at all, then a value of 1.0000 will be assumed. This can affect what data are read in if you specify a minimum F/sig >0.0

3. If a particular (h,k,l) is found more than once, only the first is used. This is because localscale uses neighboring reflections to scale each (h,k,l) and if it is found more than once there is no way to know which observations are really its neighbors in both time and position.

.

MERGE

MERGE is a routine that merges measurements of structure factor amplitudes and rejects outliers. It summarizes the quality of the dataset in a listing of R-factors on I and on F.

Sample script file for MERGE

```
!------------Script file for merging of native F from 2 data files------
@solve.setup              ! standard data for this dataset
nset 2                    ! number of input files to follow
infile(1) nat1.drg        ! input data file with F's unmerged
infile(2) nat2.drg        ! another input data file
ncolf_merge 1             ! get native F from column 1 in each data file
ncolsig_merge 2           ! get sigma from column 2
outfile native.mrg        ! output file with cols 1, 2 of "Favg" and sigma
merge
!---------------------------------------------------------------------------
```

The method followed by the program is:

1. group equivalent reflections together, analyze 1 group at a time.

2. get mean, sd for this group

3. reject observations differing from mean by >4 sigma

4. reject reflection outright if Chi-squared is greater than 20 and ikeepflag=0

5. calculate stats based on what's left

6. figure out the relationship between sigmas in the input files and reasonable estimates of the true sigmas by assuming that the reduced chi-square would equal 1.0 if the correct sigmas were present. The data are fit to the equation,

Sig**2(I)=Sig**2(Poisson)+(A*I)**2

and all sigmas are corrected with this factor.

6. write out mean, SEM for the reflection

Keywords for MERGE

```
NSHELLS n        number of shells of resolution used to group data (default=10)
NFILES n         # of input files (1 to 4)
INFILE(1) xx     input file 1
INFILE(2) xx     input file 2 (up to 4 files)
NCOLF_MERGE n    column number in input file for F (default = 1)
NCOLSIG_MERGE n  column number in input file for sigma of F (default =2)
KEEPALL          keep all reflections, regardless of merging chisqr
TOSSBAD          toss reflections with merging chisqr> 20 (default)
                 Note: KEEPALL and TOSSBAD also apply to LOCALSCALE
OUTFILE xx       output file with 2 columns (F,sig)
```

More on MERGE:

It is ASSUMED that columns 1,2 are your values of F and sigma. (If this is not true, you need to run FILEMERGE first to create such a file). If your data is I and sigma of I, then run MATH with I_TO_F to convert from I to F.

The input data files do not need to have data in any particular order or to have complete datasets.

The data are written out starting with minimum H,K,L and incrementing L fastest, then K, then H.

The routine reports the number of rejects as NNN + MMM where NNN = the number rejected as being too far from the mean for that reflection and MMM is the number of reflections rejected completely with chisqr > 20.

Estimating completeness of a dataset

COMPLETE

COMPLETE a routine to determine the completeness of a dataset. It maps input data to the asymmetric unit of the space group and calculates the percentage of data that is present.

Sample script file for COMPLETE:

```
!----------------Script to estimate completeness of a dataset ---------------
@solve.setup        !  standard information about this dataset
infile  data.drg    !  input dorgbn file with data to be examined
nnatf 1             !  column for F
nnats 2             !  column for sigma
ratmin 2.0          !  only use data with F/sigma > 2.0
complete            !  figure out completeness of this dataset
!----------------------------------------------------------------------------
```

Binary data formats and map formats used by SOLVE

DORGBN-style data files.

These files can all be viewed with VIEW and can be exported with EXPORT. ASCII files can be imported with IMPORT (see Importing and Exporting for more information on this

DORGBN files

The DORGBN-style files used in this package are binary files with data sorted by hkl. Format of the binary (FORTRAN unformatted) data file:

record 1 INTEGER*4 NCOL - the number of columns of data in the file.

record 2 (LOGICAL*1) TITLE(80) - An overall title

records 3... NCOL more titles, one for each column of data.

record 4 Data records - IH,IK,IL,RES,(F(I),I=1,NCOL)

1. IH,IK,IL - INTEGER*4 The indices of the reflection.

2. RES - The d-spacing in Angstroms.

3. F(I) - Data. These can be structure factors, sigmas, phase information stored as phase, figure-of-merit, etc. When data are missing for one or more columns the value -1.0 is stored in the appropriate columns.

```
MADFBARFILE xx.scl Output file with (Fbar,sigma,DelAno,sigma)
                    for each wavelength will be xx.scl
                    (DEFAULT="mad_fbar.scl")

MADFPFMFILE yy.scl Output file with (F+,sigma,F-,sigma) for each
                    wavelength will be yy.scl
                    (DEFAULT="mad_fpfm.scl")

madfbarfile xx.scl   dorgbn file with (Fbar,sig,Delano,Sig)
```

```
                        for each wavelength will be xx.scl
                        [Default="mad_fbar.scl"]
madfpfmfile xx.scl  dorgbn file with (F+,sig,F-,Sig)
                        for each wavelength will be xx.scl
                        [Default="mad_fpfm.scl"]

madmrgfile xxx.out    SIRAS-like MAD dataset .See MADMRG documentation.
[     default="madmrg.out"].
The file has 8 columns of data:

       1 madmrg est of Fp-zero ("Fnative")
       2 madmrg sig of fp-zero ("sig of Fnative")
       3 madmrg: MOCK FDER ("Fderiv"; equal to Fp-zero + del iso)
       4 madmrg sig of del iso ("Sig of Fderiv")
       5 madmrg est of del ano ("Delano")
       6 madmrg sig of del ano ("Sig of Delano")
       7 madmrg weighted est of del iso for Patterson
       8 madmrg weighted est of del ano for Patterson

madbstfile yyy.out  coefficients for a Bayesian Patterson to yyy.out
[default="madbst.out"]  See MADBST documentation.
The file has the following columns of data:

       1 madmrg est of Fp-zero ("Fnative")
       2 madmrg sig of fp-zero ("sig of Fnative")
       3 madmrg: MOCK FDER ("Fderiv"; equal to Fp-zero + del iso)
       4 madmrg sig of del iso ("Sig of Fderiv")
       5 madmrg est of del ano ("Delano")
       6 madmrg sig of del ano ("Sig of Delano")
       7 madmrg weighted est of del iso for Patterson
       8 madmrg weighted est of del ano for Patterson
         9--  = Fh component along Fo weighted by figure of merit
       10 --  = weighted Fh component perpendicular to Fo
       11 --  = best estimate of Fa
       12 -- sigma of
       13 -- sqrt() = sqrt of best estimate of Fa**2
       14 -- sigma of sqrt()


SOLVEDATAFILE  xxx  Output datafile with MADMRG output and MADBST
                    output  combined together, suitable for use
                    with routine SOLVE, will be xxx.   (DEFAULT file name =
                    "solve.data") The datafile has the following
                    columns of data:

                    1         Fnative (from MADMRG)
```

2	Sigma of Fnative (from MADMRG)
3	Fderiv (from MADMRG)
4	Sigma of Fderiv (from MADMRG)
5	DelAno (from MADMRG)
6	Sigma of DelAno (from MADMRG)
7	<Fh cos(theta)> (from MADBST)
8	<Fh sin(theta)> (from MADBST)

Map formats for SOLVE

The format of all maps in SOLVE is a simple binary file with the z-axis varying most rapidly and the x-axis the most slowly. They are all written with something like:

```
        do 100 iz=izs,ize
            do 100 iy=iys,iye
              write(2)(rho(ix,iy,iz),ix=ixs,ixe)
 100  continue
```

You can convert from this format of maps to NEWEZD maps (that can be read into O or mapman using FFTtoEZD (see the section on FFTtoEZD in Importing and Exporting.)

Importing and Exporting data with SOLVE

IMPORT and EXPORT are utilities to bring formatted data into dorgbn format and to write out formatted data without titles. BTOF and FTOB write out and read formatted data with titles. Also DRGTOXPLOR converts from some dorgbn files to XPLOR-style files.

FFTTOEZD and FFTTOMAPVIEW convert from the binary format for maps to the formatted NEWEZD or binary MAPVIEW (PHASES) formats.

IMPORT

The routine IMPORT has several options. You can simply read the data from a formatted file in, assuming it is h,k, l, and columns of data. You can also swap indices (as H->K, K->L, L->K) as you read it in. You can also sort the data and map it to the asymmetric unit of the space group. Ordinarily you will want to sort and map the data, as some of the other routines in the package (notably FILEMERGE) assume that the data has been sorted in a particular order of hkl. When you sort the data, the program asks if any columns are to be interpreted in terms of phases in degrees. Such data has to be correctly adjusted when it is mapped to the asymmetric unit used by SOLVE. If the data has not yet been merged to the asymmetric unit and you are about to scale the data with localscale, then you may not want to sort and map the data now. If you choose not to sort it now, then you must sort and map it with MERGE after running localscale so that filemerge can recognize the data.

When you IMPORT data, it is essential that the input file has the same number of data columns for every hkl in the file. You have two options for the format of the data in the input file.

(1) You can specify that there is exactly ONE line for each HKL record. In this case, the input file can have text in the middle of a data column which will be ignored.

(2) Alternatively, you can specify the exact number of data columns for each HKL record, in which case they can be spread over any number of lines. Data "columns" refer to the assumption that for each HKL in the data file, there are a fixed number of "columns" of associated data.

That is, if your data file looks like:

```
 -3  -5  9   300.39 1.6 420.3 1.5
          265.9 5.6 991.2
           0.2
 -3  -5 10   225.35 2.5 413.1 3.4
             441.9 3.4 114.2 0.25
```

then the HKL are (-3, -5, 9) and (-3,-5,10). For the record with HKL=(-3,-5,9) the data columns are 300.39, 1.6, 420.3, etc... In this case you can use option (2), specifying that there are 8 data columns for each HKL. You can not use option (1) for this data file because the data for each hkl are not on a single line.

The commands for IMPORTing data are:

1. Input formatted data file name (the program will then type the first 3 lines of the file as read in, and then again after stripping off any text)

2. Option (1) or option (2)

3. Output dorgbn-style file name

4. Overall title for output file

5. Number of columns of data (not counting h,k,l) in input file

6a...Title for each of these columns of data

7. Overall scale factor to apply to all data

8. lsort: 'y' to sort and map data, 'n' to leave it as is

9. lswap: 'y' to swap indices hkl [only read if lswap='y']:

8a. HNEW: index H will be mapped to HNEW. That is, if you want to map old H->new K, old K->new L, old L ->new K, then you specify HNEW = "K"

10. [only if you have said 'y' to #8, lsort] # of columns of data to interpret as phases in degrees.

11. column #'s to be interpreted as phases in degrees. Note: for equivalent reflections the phase varies depending on the associated translation.

12. dmin, dmax = resolution range to consider. All data outside of this range is ignored. Note: don't set the dmin much lower than you really want it or the routine will be very slow.

Sample IMPORT scripts

```
!-------Import script: formatted datafile with 2 data columns on one line-----
IMPORT
infile.dat
1,        !  option 1 = one line per record
```

```
output.drg
Output data after infile.dat is converted to dorgbn-style file
2,        !  2 columns of data.
Fobs (title for column 1 of data)
Sigma (title for column 2 of data)
1.0,      ! overall scale factor = 1.0
y          ! Yes sort and map data
n         ! No, do not swap indices
0         ! don't interpret any columns as phases in degrees
3.5 50    ! resolution range to consider
!-------------------------------------------------------------------------------
```

```
!-------Import script: formatted datafile with 2 data columns on 2 lines -----
IMPORT
infile.dat
2,        !  option 2 = fixed # of data columns per record
output.drg
Output data after infile.dat is converted to dorgbn-style file
2,        !  2 columns of data.
Fobs (title for column 1 of data)
phase (title for column 2 of data)
1.0,      ! overall scale factor = 1.0
y          ! Yes sort and map data
n         ! No, do not swap indices
1         ! Interpret 1 column as phases in degrees
2         ! column 2 of data is to be interpreted as phases in
degrees
3.5 50    ! resolution range to consider
!------------------------------------------------------------------------------
```

EXPORT

Export is a lot easier than IMPORT. All it does is write out a formatted file with h k l and the contents of each column of data. Here is a sample script to run export:

```
!----------------------Sample script to run EXPORT----------------------------
INFILE  data.drg         !    input file name
OUTFILE data.export      !    output file name
export
!-------------------------------------------------------------------------------
```

Binary TO Format conversion (BTOF)

Format TO Binary conversion (FTOB)

These routines convert data in DORGBN-structured binary files to and from formatted files with titles. They are most useful for transferring data files from one computer system to another. A sample script for doing each is shown next:

```
!-----------Script to convert from dorgbn -> formatted file ---------------
INFILE data.drg
OUTFILE data.fmt
BTOF
!--------------------------------------------------------------------------


!---Script to convert from formatted file with titles -> dorgbn file---------
INFILE data.fmt
OUTFILE data.drg
FTOB
!--------------------------------------------------------------------------
```

DORGBN files

The files used in this package are binary files with data sorted by hkl. Format of the binary (FORTRAN unformatted) data file:

record 1 INTEGER*4 NCOL - the number of columns of data in the file.

record 2 (LOGICAL*1) TITLE(80) - An overall title

records 3... NCOL more titles, one for each column of data.

record 4 Data records - IH,IK,IL,RES,(F(I),I=1,NCOL)

1. IH,IK,IL - INTEGER*4 The indices of the reflection.

2. RES - The d-spacing in Angstroms.

3. F(I) - Data. These can be structure factors, sigmas, phase information stored as phase, figure-of-merit, etc. When

data are missing for one or more columns the value -1.0 is stored in the appropriate columns.

DRGTOXPLOR

The routine DRGTOXPLR writes out X-PLOR format file with INDIC, FOBS, SIGMA based on data in a DORGBN-style file. Here is a sample script:

```
!---------------------Script for DRGTOXPLOR --------------------------------
INFILE  data.drg
OUTFILE  data.xplor
NXPLORF  1              ! column 1 of data.drg is F
NXPLORSIG 2             ! column 2 of data.drg is Sigma of F
DRGTOXPLOR
!---------------------------------------------------------------------------
```

FFTtoEZD and FFTtoCCP4 and FFTtoMAPVIEW

FFTTOEZD

FFTTOEZD is a routine that converts an asymmetric unit of an FFT in the UCLA FFT format to any region of the map in the newezd format suitable for reading right into "O". This routine is applicable to Fourier maps, but can be used with Patterson maps as long as the output region is contained within the input FFT.

The output map is calculated using the same grid as the input FFT, but the endpoints in x,y, and z can be different. The grid points in x and y must not be more negative than -512 or greater than 512. The program generates the entire unit cell from the input FFT if the endpoints of the output EZD map are not contained within the input FFT. For Patterson maps, the output EZD map MUST be contained within the calculated FFT (i.e, no expansion is allowed).

The output map is scaled so that the rms value of the map is 1.0.

If you are in "O" and want to read in and display "fourier.ezd" all you need to type is:

```
@fourier.ezd
ezd_draw
<cr>
<cr>
```

Script file for converting from FFT to EZD

```
!-----------Script for conversion from UCLA FFT format to EZD ---------
@solve.setup                 !standard parameters for this dataset
fftfile fourier.fft          ! input FFT file
bossfile fourier.ezd          ! output EZD file
fourier                      ! this is a fourier, not a patterson
ezdgrid -35 37 -23 96 5 23  ! Output grid covers region from -35 to 37
                             ! in x (on the same grid that the fourier was
                             ! calculated with)
ffttoezd
!-------------------------------------------------------------------------
```

Keywords for FFTTOEZD

```
FFTFILE   xx      name of FFT-containing file
EZDMAPFILE   xx      name of output EZD format file
FILETITLE xxx      optional title for file
PATTERSON       this is a Patterson map
FOURIER         this is a fourier map
EZDGRID   ixstart ixend iystart iyend izstart izend -- these are
                  starting and ending grid units on the PATTGRID
                  or FFTGRID you have specified.
```

FFTTOCCP4

FFTTOCCP4is a routine that converts an asymmetric unit of an FFT in the UCLA FFT format to CCP4 map format. The region defined by FFTGRID is output

Keywords for FFTTOCCP4

```
FFTFILE   xx      name of FFT-containing file
CCP4MAPFILE   xx      name of output EZD format file
```

FFTTOMAPVIEW

FFTTOMAPVIEW is a routine that converts an asymmetric unit of an FFT in the UCLA FFT format to a format compatible with the PHASES package. This is useful for displaying the map using MAPVIEW in the PHASES package.

Note: this routine is only valid on an SGI machine.

The output map is calculated using the same grid as BOSS-style output maps. This grid may be set with the keyword BOSSGRID. The output map range of grid points must be contained, for x and y, between -512 and 512. For Pattersons, this grid must be contained within the FFT grid, set with FFTGRID

Script file for FFTTOMAPVIEW

```
!---------------Script for conversion of UCLA FFT to MAPVIEW format--------
!  Note:  this only works on an SGI
@solve.setup                 ! standard information for this dataset
fftfile fourier.fft          ! input fft file
mapviewfile fourier.mapview  ! output mapviewfile for PHASES
fourier                      ! this is a fourier, not a patterson
bossgrid -35 37 -23 96 5 23  ! Output grid covers region from -35 to 37
                             ! in x (on the same grid that the fourier was
                             ! calculated with)
ffttomapview
!-----------------------------------------------------------------------------
```

Keywords for FFTTOMAPVIEW

```
FFTFILE   xx      name of FFT-containing file
MAPVIEWFILE xx    name of output MAPVIEW format file
PATTERSON       this is a Patterson map
FOURIER         this is a fourier map
BOSSGRID  ixstart ixend iystart iyend izstart izend -- these are
                  starting and ending grid units on the PATTGRID
                  or FFTGRID you have specified.
```

Contents	Index

FILEMERGE

Merging dorgbn data files

[Script]

FILEMERGE allows manipulation of binary data files that are in the "dorgbn" format. You can extract one or more "columns" of data from a file, duplicate columns from a file, or combine parts of different files.

This routine is based on the UCLA program DORGBN. Data are stored in binary files in the form h,k,l,resolution,and columns of "data". At the beginning of the file are an overall title for the dataset and individual titles for each "data" column.

FILEMERGE runs interactively. The command format is:

1. # of Files to be opened for input (up to 4)

2a... Input file names 1...n.

3. Output file name

4. Title for output file. This title should describe the

contents of the entire file.

5. Command lines. Each command can specify a range of columns of data from some particular file to be incorporated into the output file. These data columns are incorporated in the order in which the commands are specified. Command input parameters: (these are 1 or 2 digit integers and an optional title, all separated by commas and no blanks):

1. NFILE: The number of the file treated.

2. ICOL,JCOL,TITLE: the RANGE of columns to be copied. If you want to copy just column #3, specify: 3,3 here.

The title is an optional title to be substituted for that already associated with the first column in the range. If the specified range includes more than one column and this title field is used, the titles for all the other columns in the range specified by this command must be input in the very next records. If this field is blank, the old title will be used.

PLEASE NOTE: if you are running filemerge from a command file, you cannot leave any blanks or other characters afer the column numbers here. That is, 3,3

is fine, but

3,3 !xxx

is not.

7. More command lines.

8. A "0" to signify the end of input.

An additional useful feature of the program is that if a command is entered with a valid file number but with the column numbers missing or incorrect, the titles of the columns on that file are printed for the user.

Script for FILEMERGE

Here is a sample script that will take columns 1,2 from "file1.drg" and columns 1 and 3 from "file2.drg" and write them out to "fileout.drg".

```
!----------------------Sample script for FILEMERGE --------------------------
FILEMERGE
2
file1.drg
file2.drg
fileout.drg
Title for fileout: cols 1,2 from file1 and cols 1,3 from file2
1
1,2
2
1,1
2
3,3
0
!------------------------------------------------------------------------------
```

Calculating and working with maps

MAPS

```
This routine reads in data from a dorgbn-style file and calculates any of the
following types of maps:

PATTERSON                        !Fourier coefficients are F or F**2
ORIGIN_REMOVED PATTERSON         !As Patterson, but origin-removed
FOURIER                          !Fourier coefficients are A+iB
NATFOURIER/ISOFOURIER            !Fourier coefficients are mF exp[i Phi]
ANOFOURIER                       !Fourier coefficients are mF exp[i (Phi + 90)]
2FOFC                            !Fourier coefficients are (2Fo-Fc) exp[i Phi]
FOFC                             !Fourier coefficients are (Fo-Fc) exp[i Phi]
```

Note that:

In all cases a "0.0" or "-1.0" in a column is interpreted as missing data.

DELMAX can be used to cut off the biggest |F| or |DelF|

If NCOLFIGM is specified then the map is weighted by figure of merit

If NCOLPATTSIG is specified then Patterson maps are weighted.

If you specify a PEAKFILE then routine PEAKSEARCH is called automatically.

If you specify a MAPVIEWFILE then FFTTOMAPVIEW is called.

If you specify a BOSSFILE then FFTTOBOSS is called.

Keywords for MAPS

```
PATTERSON                    Patterson map
```

```
ORIGIN_REMOVED_PATT  Origin-removed Patterson map
NCOLPATT**2          Column # in input map for squared Patterson coefficients
            -or-
NCOLPATT                column # containing Patterson coefficients
NCOLPATTSIG          Column # for uncertainty in values of ncolpatt. If you
                     specify ncolpattsig then you will get a weighted
                     patterson or difference patterson.

FOURIER         Fourier map, reading in A, B coefficients directly
NCOLFA                  column for A
NCOLFB                  column for B

NATFOURIER      same as isofourier, below
ISOFOURIER      F exp(iPHIc) map (F = del iso or F, phase = PHI)
ANOFOURIER      F exp(i[PHIc-90]) map (F = del Ano, phase = PHI - pi/2)
NCOLF           column for F or Del F
NCOLPHI         column for PHI (in degrees)
NFIGM           column for figure of merit if you want a weighted map
                  Note: if you set NFIGM for routine HEAVY it will be
                  applied here too.  Be sure you reset it to the value you
                  want.

FOFC            Fo-Fc exp(i PHIc) map
2FOFC           2Fo-Fc exp(i PHIc) map
NCOLFOBS        column for Fo
NCOLFC          column for Fc
NCOLPHI         column for PHI (in degrees)

DELMAX          maximum |F| or |del F| to include . If zero-use all
                          data.  DEFAULT=0

INFILE   xxx       name of input DORGBN file with data
FOURCOFILE xxx     name of optional output DORGBN file with A,B coefficients
                  (this can be used as input to another program for calculation
                  of maps, if desired, by converting it with EXPORT)

FFTFILE   xxx      name of output file with FFT in UCLA format
PEAKFILE  xxx      name of optional file with nlist high and low peaks
                  (program finds peaks if this file is defined)
BOSSFILE  xxx      name of optional file with FFT in BOSS format
                  (program writes bossfile if it is defined, see FFTTOBOSS)
MAPVIEWFILE xxx    name of optional file with FFT in MAPVIEW format
                    (program writes MAPVIEW file if it is defined,
                   see FFTTOMAPVIEW)



NLIST   n        number of high and low peaks to list to peakfile
ISYMMETRY   n     number of symmetry equivalents for each peak to list
```

```
                        (put a big number to get all within all adjacent
                        unit cells)
PDB             list peaks in PDB format
FRACT           list peaks in fractional format
```

Script using MAPS to generate a difference Patterson function:

```
!------------Script to calculate a difference Patterson --------------
@solve.setup        ! standard information for this dataset
!  wtder.drg contains Fnat,sig,Fder,sig:

INFILE wtder.drg        ! input datafile with Fnat,sig, Fder, sig
OUTFILE iso.drg         ! intermediate file with Fder-Fnat
NNATF 1                 ! column for native F
NNATS 2                 ! column for sigma of native F
NDERF 3                 ! column for derivative F
NDERS 4                 ! column for sigma of derivative F
GETISO                  ! get isomorphous differences
INFILE iso.drg          ! now read in those isomorphous diffs again
NCOLPATT 1              ! use column 1 as del F for a patterson
PATTERSON               ! map type is patterson
FFTFILE ha.patt         ! output fft goes to ha.patt
MAPS                    ! calculate the map

[HASSP]                 ! analyze the patterson with HASSP, if desired
!---------------------------------------------------------------------------
```

PEAKSEARCH

PEAKSEARCH is a routine that finds high and low points in a Fourier map. It is not applicable to Patterson maps (use HASSP for that purpose). It assumes that the FFT has been calculated over the entire asymmetric unit and uses the symmetry file to map neighboring grid points on to the asymmetric unit. It reports the highest peaks in the map, with the height being the highest value of the FFT on a grid point and the coordinates being the centroid of the peak.

The routine can read in an FFT written by MAPS or it can be called at the end of routine MAPS. Note that it cannot read BOSS format data.

The routine will write out NPEAK highest and NPEAK lowest peaks to the output file. The "B-factor" in the PDB format file is the peak height/1000. The final column in the fractional-format file is the peak height/1000.

Script file for PEAKSEARCH

```
!-----------Script file to run peaksearch on a fourier-----------------------
@solve.setup          ! standard information for this dataset
fftfile ha.fft        ! input map name
peakfile ha.peaks     ! write list of peaks to "ha.peaks"
peaksearch
!-----------------------------------------------------------------------------
```

Keywords for PEAKSEARCH

```
FFTFILE  xxx       name of fft-containing file
NLIST    n       number of peaks (high, same # of low) to list
ISYMMETRY  n     # of symmetry equivalents to list for each peak
                   -1 for all within FFTTOBOSS region, 0 for all within
                    FFT region. 1= default= just 1 for each peak
PEAKFILE   xx     name of output file
PDB             write peaks in PDB format
FRACT           write peaks in fractional format
POSITIVEONLY    only list positive peaks
NEGATIVEONLY    only list negative peaks
```

MAPTOASYM

MAPTOOBJECT

Routines MAPTOASYM and MAPTOOBJECT map the atoms in a PDB file, one by one, using crystallographic symmetry. MAPTOASYM maps atoms into the asymmetric unit of the space group, as defined by the symmetry file and the (FFTGRID) grid specified for FFT calculations, assumed to contain the asymmetric unit. MAPTOOBJECT maps atoms to their symmetry equivalents closest to an atom in a second PDB ("object") file.

Sample script files for MAPTOASYM and MAPTOOBJECT

```
!--------Script file to map atoms in a PDB file to the asymmetric unit ----
infile(1)  atoms.pdb      !  input file with coordinates to be mapped
outfile    atoms.mapped   !  atoms mapped to asymmetric unit
maptoasym                 !  map them to asymmetric unit
!---------------------------------------------------------------------------

!---Script file to map atoms in a PDB file close to atoms in another file ----
infile(1)  atoms.pdb      !  input file with coordinates to be mapped
infile(2)  object.pdb     !  pdb file with object that we want to be close to
outfile    atoms.mapped   !  atoms mapped close to object
dismin     0.0            !  Only atoms with a closest distance to an atom
dismax 1000.0             !  in object between dismin and dismax will be
                          !  written out. Default= 0 to 1000000.
maptoobject                 !  map them as close to the object as possible
!---------------------------------------------------------------------------
```

RHO

RHO is a routine to write out the values of a map at the coordinates of atoms in a PDB file. The values of the map at the grid point nearest the input coordinates are listed, coordinate-by-coordinate. This routine is very useful for evaluating how good the fit of a model to a map is.

Sample script file for RHO

```
!--------Script file to write out values of a map at coords in a PDB file ----
@solve.setup              !  standard setup for this dataset
infile(1)  atoms.pdb      !  input file with coordinates to be examined
fftfile fourier.fft       !  map file
rho                       !  interpolate value of fft at all coords in model
!---------------------------------------------------------------------------
```

OMIT_MAP

OMIT_MAP is a routine to calculate a set of overlapping omit maps and then to average them together using AVG_OMIT. The routine requires a .PDB file with coordinates and a .DRG data file with Fobs values. A set of omit maps in which 7 residues at a time are deleted, overlapping by 2 (1-7, 6-11, etc) is calculated, and the atoms used to calculate each map are written out in a set of pdb files as well. Then AVG_OMIT is called to average the parts of the maps that are in the regions not occupied by atoms used in the phase calculations. If you are doing simulated annealing-omit maps, then you need to use routine "AVG_OMIT" instead.

Sample script file for OMIT_MAP

```
!--------Script file to calculate overlapping omit maps ------------------
@solve.setup              !  standard setup for this dataset
INFILE(1) atoms.pdb       !  Name of PDB file with coordinates
INFILE(2) data.drg        !  Name of .DRG data file
NNATF 1                   !  Column number for F in data file
NNATS 2                   !  Column number for sigma of F in data file
RATMIN 2.0                !  Minimum ratio of F/sigma to use (default=0.)
OUTFILE avg.fft           !  File name for file with output averaged FFT map
XCUT 2.0                  !  Density within XCUT of an atom is not used
                          !        default=2.0 (Angstroms)
OMIT_MAP
!---------------------------------------------------------------------------
```

AVG_OMIT

AVG_OMIT is a routine to average key parts of a set of omit maps. For each map, only the region of the map that is further than DCUT from all atoms in a PDB file is used in the averaging. Omit maps or simulated annealing-omit maps can be used here.

The basic idea is that an omit map contains useful information only for parts of the map that are not occupied by the atoms used to phase that map.

To use this routine, calculate a series of omit maps for your structure that sequentially leave out residues 1-5, 4-9, 8-12, etc... (or whatever is a good size to leave out). Then you read in each omit map and the coordinates of atoms left in for that calculation. If you are using just one PDB file, then use the routine "OMIT_MAP" which will do everything for you. If you are doing a bunch of simulated annealing-omit maps, this is the routine to use.

NOTE: any points in the map that are not more than XCUT away from all atoms in at least 1 map will get a value of 0.0. This can lead to planes and other small regions in the map being zeroed out if all omit maps contain atoms near them. This is a problem in cases where two atoms distant in sequence are close together so that one or the other is always there in your omit maps. The solution is to construct a location-based omit map to delete all the atoms in that region.

Sample script file to run AVG_OMIT

```
!-------Script file to average overlapping omit maps -----------------------
@solve.setup            !  standard setup information

                        ! next call avg_omit.  Inputs are:
                        ! 1.  XCUT
                        ! 2.  NMAPS
                        ! 3a. file with coordinates of all atoms used to
                        !     calculate map 1
                        ! 3b. name of fft file with map 1
                        ! 4a.  as 3a, but coordinates for map 2
                        ! 4b.  as 3b, but map 2
                        ! output map file
avg_omit                ! call avg_omit.  It reads from standard input
2.0,                    !  XCUT = region around atoms to be excluded
2,                      ! # of maps to average
pdbfile1.pdb
mapfile1.fft
pdbfile2.pdb
mapfile2.fft
average.fft
! Now average.fft has in it the averaged parts of each input map that are not
!  near an atom in the corresponging pdb file
!---------------------------------------------------------------------------
```

Misc commands

HA_PDB: Write out a heavy atom solution in PDB format

Solve will write out your current heavy-atom model in PDB format if you simply specify the command HA_PDB. This can be done, for example at the very end of an automated run of SOLVE to output the final heavy-atom model in pdb format.

The output file will be called "ha.pdb".

COMPARE_SOLN: Comparing two heavy-atom input files

Compare_soln reads in heavy atom sites from 2 files and compares them. It tests all possibilities for inversion and origin shifts that could make the solutions indistinguishable. It reports back if the solutions are the same and how many sites are the same if not . The files need to be in the format used by HEAVY and SOLVE. All lines in the 2 files except those that say "DERIVATIVE" or "XYZ" or "INANO" are ignored. Each time "DERIVATIVE" is encountered, a new derivative is started. Each XYZ is read as a keyword for xyz of a new atom in the current derivative. If "INANO" is specified for a derivative then the handedness of the solution is considered in comparing the solutions. Otherwise mirror images are considered the same. In all cases solutions that simply differ by an origin shift are considered the same. The grid used for FFT calculations is used to determine how close atoms must be to be considered the same, and the cutoff is 1 to 2 grid units. If you have a PDB file with your coordinates for one derivative, you can substitute

PDB_XYZ_IN filename instead of the XYZ lines. You still need the DERIVATIVE statement.

For example:

If xyz1.dat looks like:

```
Derivative 1
xyz 0.45 0.93 .33

and xyz2.dat looks like

Derivative
xyz .51 .48 .93
derivative
xyz .44 .351 .31
```

then use a script like this:

```
!----------Script to compare to heavy-atom input files -------------------
infile(1) xyz1.dat
infile(2) xyz2.dat
compare_soln
!------------------------------------------------------------------------
```

FRACT_TO_CART and CART_TO_FRACT: Converting between fractional and Cartesian coordinates

FRACT_TO_CART converts from fractional to Cartesian coordinates. CART_TO_FRACT converts from Cartesian to fractional coordinates. In each case, coordinates are read from the file defined by INFILE and written to the file defined by OUTFILE. Coordinates are read in free-format.

The Cartesian coordinate system is the PDB default: X is along **a** and Z is along **c***; X Y and Z are mutually perpendicular and right-handed.

Use a script like this:

```
!----------Script to convert from fractional to Cartesian coords----------
infile fract.xyz
outfile cart.xyz
fract_to_cart
!------------------------------------------------------------------------
```

MATH

MATH is a routine that is useful for generating test data, for converting I,sig to F,sig, and other simple conversions. In most cases, the routine takes an input file and column numbers for the input data and writes out an output file.

KEYWORDS for routine MATH:

KEYWORD	parameters	description
GENF_PHI	SCALE B	Generates an asymmetric unit of data in this space group based on resolution limits set by DMIN DMAX. The rms F will be SCALE at low resolution and will decrease according to the thermal factor B. Output file has F, PHI. F and Phi are distributed according to Wilson statistics.
I_TO_F	ncolI ncolSIGI	Convert from I, sigma to F, sigma
SIGMA_SCALE	xx	scale all input intensity sigmas * xx
A_B_TO_F_PHI	ncolA ncolB	Convert from (REAL,IMAGINARY) to (F,PHI) (phi in degrees).
F_PHI_TO_A_B	ncolF ncolPHI	Convert from (F,PHI) to (REAL,IMAGINARY)
FOBS_SIG_FROM_F_ERR	ncolF ERR	Add "measurement error" on to values of F. % error in F will be about ERR. Uses error model of: sigma(I)=sqrt(I+0.5*(ERR/100)*I**2), where I=F**2.
SUM	ncol1 ncol2	Output column 1 is sum of data in ncol1 and ncol2 in input file
VECTORSUM	ncol1A ncol1B ncol2A ncol2B	Output column 1 = ncol1A+ncol2A Output column 2 = ncol1B+ncol2B
SEPARATEANO	ncolF ncolsig	Read in hkl, F,Sig from file defined by INFILE and write out the reflection to the file defined by FPLUSFILE if this is F+, or the file defined by FMINUSFILE if this is F-. A reflection is F+ in this definition if it is in the asymmetric unit as defined by HEAVY or it can be rotated into the asymmetric unit with reciprocal lattice symmetry. It is F- if it can be rotated onto -h-k-l of a reflection that is an F+.
FLIP		In separateano, write out the inverse of the input indices for F- (i.e, if h,k,l are read in, write out -h -k -l)

NEQUIV_SEPARATE n	In separateano, map the F- reflections to equivalent reflection "n" before applying FLIP and writing out. This allows you to match up reflections collected on a mirror plane as opposed to those collected with phi+180. If you run SCALE_MIR or SCALE_MAD the value of "n" is chosen for you automatically by maximizing the number of F+/F- pairs related by symmetry operation "n".
TRIM	Delete every line of a reflection file that has "-1" in any column
RESOLUTION xx yy	resolution limits
	If resolution limits and infile and outfile are specified, copies infile to outfile, using only data in the resolution range.
INFILE xxxx	input file name
OUTFILE xxxx	output file name (except separateano)
FPLUSFILE xxxxx	output file for F+ in separateano
FMINUSFILE xxxxx	output file for F- in separateano

Sample script for routine "MATH"

Here is a simple script that will convert from A, B fourier coefficients to F and phi:

```
!-------------------Script for conversion of A B to F, Phi --------------
@solve.setup               !  standard setup for this dataset
infile ab.drg
outfile fphi.drg
a_b_to_f_phi 1 2            ! take columns 1 and 2 of infile and convert to
                           !   F and phi
math
!-------------------------------------------------------------------------
```

GETISO

GETISO is a routine to subtract 2 columns of data in a dorgbn file and to write out a new file with the difference. Reflections with a "-1.0" or "0.0" in either column are ignored as are reflections with F/sigma < ratmin. Here is an example:

```
!------------Script file for getting isomorphous differences ------
@solve.setup                        ! setup script file
nnatf 1                             ! column for Fnat
nnats 2                             ! column for sigma of Fnat
nderf 3                             ! column for Fder
nders 4                             ! column for sigma of Fder
ratmin 2.0                          ! minimum F/sig to include
infile file1.drg                    ! input file
outfile file2.drg                   ! output file
getiso                              ! get isomorphous differences
!--------------------------------------------------------------------
```

GETANOM

GETANOM is a routine to convert from F+, F- to Fbar, DelAno. The routine calculates Fbar = (F+ + F-)/2 and del Ano = (F+ - F-), for the selected reflections and writes it to the output dorgbn file. If F+ or F- are missing (F less than or equal to 0), and the keyword "fp_or_fm" is specified, the one present is written out as Fbar and del Ano and sig of del Ano are set to 0.0. If the keyword "fpfm_only" is set then reflections with F+ or F- missing that are acentric are tossed. Here is an example script file for GETANOM:

```
!------------Script file for converting from F+,F- to Fbar, Delano-------
@solve.setup
ncolfp             ! column for F+
ncolsfp            ! column for sigma of F+
ncolfm             ! column for F-
ncolsfm            ! column for sigma of F-
fpfm_only          ! toss acentric reflections if F+ or F- is missing
infile file1.drg   ! input file
outfile file2.drg  ! output file with 4 columns
getanom
!--------------------------------------------------------------------
```

GETPHASES

GETPHASES is a routine to convert from A and B fourier coefficients to F and Phi. Here is a script file to do this:

```
!------------------Script file to convert from A, B to F, Phi ------------
@solve.setup
ncolfa 1           ! column for F cos(phi) = A
ncolfb 2           ! column for F sin(phi) = B
infile file1.drg   ! input file with A, B
outfile file2.drg  ! output file with F, Phi (2 columns)
getphases
!----------------------------------------------------------------------------
```

Bayesian weighting for atomic refinement

WEIGHTS is a routine to generate weighting factors for atomic refinement. The weighting factors are based on both experimental sigmas and on rms values of (Fobs-Fcalc)**2 in ranges of resolution. The premise for this type of weighting is that the atomic model used to generate Fcalc is incomplete. Note that this leads to an expected difference between Fobs and Fcalc that is larger for centric reflections than for acentrics by a factor of 1.414. The errors in the fit of the model to the data are divided into two parts, one due to errors in measurement and one due to errors in the model. It is assumed that errors in measurement are reasonably well known. They are required for this routine. The input file must contain Fobs, sigma-obs, and Fcalc. It may also contain a flag marking "TEST" reflections for free-R calculations. To generate the input file, you will need to run X-PLOR or another program to get Fcalc values, then IMPORT the Fcalc values and FILEMERGE with your Fobs, sigma-obs data.

The output dataset ("Fobs, sigma, weight") is written in X-PLOR format and can be read in to X-PLOR just as if it were Fobs, sigma, weight. X-PLOR automatically uses the weight as a weighting factor in refinement if it is input in this way in the structure factor file. Note that the sigma here is NOT the experimental error in measurement any more.

You can select reflections with F>n*sigma-obs using RATMIN, but even if you include all reflections, a reasonable weighting factor will be generated for the weak reflections.

The program allows you to keep reflections flagged with RTEST>0 separate from the working set of reflections. The RTEST flag must be in a "column" of data in the input file. RTEST=1 indicates a TEST reflection, 0 indicates a reflection to use in refinement.

The errors in the model are estimated in a shell of resolution as

E**2 = [< (Fobs-Fcalc)**2 > - <Sigma-obs**2>]

where centric and acentric reflections are treated separately. The weighting factor applied to a particular reflection is then:

WEIGHT = 1/(E**2 + Sigma-obs**2)

Reflections where Sigma-obs is not >0 or Fobs is not > ratmin*sigma-obs or Fcalc is not >0 are ignored and not written out.

Script file for WEIGHTS

```
!----------Script file generating Bayesian weighting for refinement-----------
@solve.setup              ! standard information for this dataset
infile fofc.drg           ! input dorgbn file with Fo, sigma, Fc, Rtest
outfile fo.xplor          ! output file with FOBS SIGMA RTEST for X-PLOR
NCOLFOWT  1               !  column # for F of data
NCOLSWT   2               !  column # of sigma-obs
NCOLFC    3               !column # of Fcalc
NCOLRTEST  4              ! column # for RTEST indicator (0 if not present)
weights                   ! get Bayesian weights...
!-----------------------------------------------------------------------------
```

Keywords for WEIGHTS

```
NSHELLS n       number of shells of resolution used to group data (default=10)
INFILE xx       name of file with Fobs,sigma-obs, and Fcalc and optional Rtest
OUTFILE xx      name of output file in X-PLOR format with
                 h,k,l,fobs,sig,weight,rtest
NCOLFOWT n       column # for F of data
NCOLSWT n        column # of sigma-obs
NCOLFC n        column # of Fcalc
NCOLRTEST n     column # for RTEST indicator (0 if not present)
RATMIN xx       minimum ratio of F/sigma to read in at all (default=0)
```

Contents	Index

Bayesian difference refinement

[Script for Fdiff | Keywords for Fdiff]

FDIFF is a routine to generate a pseudo-mutant dataset for difference refinement. The output dataset ("Fdiff, sigma-fdiff, weight") is written in X-PLOR format and can be read in to X-PLOR just as if it were Fobs, sigma, weight. X-PLOR automatically uses the weight as a weighting factor in refinement if it is input in this way in the structure factor file.

Fdiff is used in cases where a "WT" structure has been refined and a "mutant" dataset is available, and where it is the differences between the WT and mutant structures that is of interest. This routine takes Fcalc for the WT and mutant datasets and Fobs for the WT and mutant datasets to create a pseudo-mutant dataset: Fdiff and sigma and weights. These are then used just as if they were Fobs,mutant and sigma and weighting factors in refinement of the mutant structure. You can select reflections with F>n*sigma using RATMIN, but even if you include all reflections, a reasonable weighting factor will be generated for the weak reflections.

The value of Fdiff is given by:

Fdiff = Fc (WT) +Beta* (Fobs, mutant - Fobs, WT)

where the factor Beta is essentially the correlation coefficient between (Fo-Fc),WT and (Fo-Fc),Mutant. Bayesian difference refinement is similar to difference refinement except for the factor Beta (which is 1.0 for difference refinement).

The input file must contain Fobs, sigma-obs, and Fcalc, for BOTH the WT and "MUT" (variant) datasets. It optionally may also contain a flag marking "TEST" reflections for free-R calculations. To generate the input file, you will need to run X-PLOR or another program to get Fcalc values, then IMPORT the Fcalc values and FILEMERGE with your Fobs, sigma-obs data.

The program allows you to keep reflections flagged with RTEST>0 separate from the working set of reflections. The RTEST flag must be in a "column" of data in the input file. RTEST=1 indicates a TEST reflection, 0 indicates a reflection to use in refinement.

Basic script file for FDIFF

```
! ------------Script file to generate data for difference refinement -------
```

```
@solve.setup                 ! standard data for this dataset
infile data.drg              ! input file with WT Fo, sig, Fc and mut Fo, sig
outfile fdiff.drg            ! output file with Fdiff, sigma,weight, RTEST
ncolfowt 1                   ! column # for WT Fobs
ncolswt  2                   ! column # for sigma of WT Fobs
ncolfc   3                   ! column # for WT Fcalc
ncolfomut 4                  ! column # for Mutant Fobs
ncolsfomut 5                 ! column # for sigma of mutant Fobs
ncolrtest 6                  ! column # for RTEST indicator
fdiff                        ! setup up difference refinement
!------------------------------------------------------------------------------
```

A more advanced script file for FDIFF written by Joel Berendzen (joelb@lanl.gov) that works with CNS

The file bayesdiff.ksh contains a script that does Bayesian differencing on two CNS output files and writes the output to a third CNS input file.

Keywords for FDIFF

```
INFILE xx input file with WT Fc, Fo, sig and MUT Fo,sig, and optional
Rtest column
OUTFILE xx output file with FDIFF,sdiff,weight,rtest
NCOLFC n column # for WT Fc (WT Fcalc) in input file
NCOLFOWT n column # for WT Fo (WT Fobs) in input file
NCOLSWT n column # for sigma of WT Fo
NCOLFOMUT n column # for MUT Fo (MUT Fobs) in input file
NCOLSMUT n column # for sigma of MUT Fo
NCOLFCMUT n column # for MUT Fc
NCOLRTEST n column # for RTEST indicator (0 if missing)
RATMIN xx minimum ratio of F/sig to read in data (default=0)
```

New features for versions 1.10-2.10 of SOLVE

- SOLVE now has easy scripts for SAD phasing.
- SOLVE has even better MAD phasing! SOLVE now re-refines scattering factors using the final heavy atom parameters, improving the final phases. In combination with version 1.04 or later of RESOLVE, the final maps are greatly improved over version 1.17.
- Must faster search for solutions: SOLVE now follows only the very best solution by default, greatly speeding up the search in most cases.
- You can start from MR or other input phases.
- You can read CCP4 unmerged intensities directly.
- You can specify a SOLVETMPDIR where SOLVE will write scratch files.
- You can tell SOLVE where some sites are and go on from there easily with ANALYZE_SOLVE and with ADDSOLVE
- There is now a SOLVE FAQS page.

- SOLVE versions starting with 1.16 fix a serious bug that was present in versions 1.10 to 1.15. This bug was that upon conversion from SOLVE's asymmetric unit to the CCP4 asymmetric unit (done in routine "SOLVE" for all data, whether mtz input or not) the phase of FA relative to Fp was not swapped when the asymmetric unit changed. This affects the difference Fouriers calculated by SOLVE for space groups where the CCP4 and SOLVE asymmetric units for reflections are not the same.
- SOLVE version 1.17 fixes some array dimensioning errors in versions up to 1.16 that can cause it to hang when finding very large numbers (50 or so) sites.
- Solve version 1.17 sets the default of "resolution_steps" to 1 (it was 3 in previous versions). This improves MIR results for polar space groups but could slow down Solve in some cases.
- Solve version 1.18 had a minor bug in the way it dealt with SAD phasing, always substituting in the default f-doubleprime values for the heavy atoms from the tables, and including non-zero atomic scattering factors. Version 1.19 allows the user to input f-doubleprime (with fprprv_mad) and sets the form factors other than f-doubleprime to zero. This creates the appropriate scattering factors for a pseudo-derivative that is just like a native but has anomalous differences. This pseudo-derivative is used for phasing with the observed anomalous differences.
- Solve version 1.18 had a dimension error preventing the use of more than 5 derivatives; this is fixed in 1.19.
- Solve version 2.00 fixes a bug in previous versions of combine_all_data where the number of wavelengths was not read in correctly for the second dataset.
- Solve version 2.00 has scattering factors for Br

- Solve version 2.00 fixes bug which prevented solving SAD datasets in space group P1.
- Solve version 2.00 restores compatibility with new d*trek format.
- Solve version 2.01 incorporates Sim weighting on the heavy-atom sites in SAD phasing (this has the effect of replacing the heavy-atom sites in the map, and for cases with a high proportion of anomalous scatterers such as phosphorous phasing of DNA, makes a much better map).
- Solve version 2.03 writes out fractional coordinates to solve.xyz and also the inverse to solve_inverse.xyz.
- Solve version 2.03 fixes a bug in SAD which resulted in no phases written out if refinement of scattering factors failed.
- resolve version 2.03 fixes a bug in the output which caused the HL coefficients from the input file to be copied without changes to the output file whenever resolve terminated without carrying out all requested cycles.
- SOLVE version 2.03 changes the default BMIN (minimum B-value) from 15 to 2.0
- RESOLVE version 2.03 includes iterative model-building and model rebuilding
- RESOLVE version 2.04 will automatically figure out the optimal solvent content and histograms for your structure
- RESOLVE version 2.05 will use local patterns of density to improve the map
- RESOLVE version 2.05 has increased the weighting on map-based information by a factor of 2 relative to earlier versions to match recalculated estimates of the number of degrees of freedom in the map (now 2 * n_refl; was previously n_refl).
- RESOLVE version 2.06 introduces a new resolve_autobuild script that will carry out pattern identification, fragment identification, and model-building, using the superquick_build option, to get a partially refined model from MAD/SAD/MIR data in just a few cycles.
- RESOLVE version 2.08 fixes a bug which caused RESOLVE to fail when NCS was found in the ha sites, but then later rejected due to lack of NCS in the map.
- The RESOLVE_BUILD script for version 2.08 uses the average of the top 5 structures built in the initial stages to start the rebuilding process.
- RESOLVE version 2.08 introduces scoring of electron density maps based on skew, correlation of local rms (as in SOLVE), correlation of map calculated with map-probability with original map, tertiary structure.
- RESOLVE version 2.09 introduces greatly improved automated ligand fitting
- RESOLVE version 2.09 fixes a problem with finding sites with MAD data in trigonal/hexagonal space groups with 2-wavelength data or 4-wavelength data. These searches will have generally failed with previous versions, while single-wavelength or 3-wavelength data from the same datasets would have succeeded. The phasing was fine for cases where the solution was found, but the bug would normally prevent finding the heavy-atom sites. There may be some datasets where version 2.09 can solve 2-wavelength data that could not be solved by previous versions. (This problem was due to accidentally swapping the sign of anomalous differences used in the MADMRG routine during the conversion to the ccp4 asymmetric unit a number of times equal to the number of wavelengths...thereby cancelling out the necessary swapping for even-numbers of wavelengths... The problem did not affect phasing, unless the "nobayes" flag was specified, because phasing is done with all wavelengths, not with the MADMRG data.
- RESOLVE version 2.09 fixes a problem in reading heavy-atom sites and problems with reading

non-protein atoms in PDB files. RESOLVE earlier versions would ignore the atom HG as a heavy-atom in the NCS calculation, because the routines assumed that it was "H" (which is ignored)ignored) This is fixed in version 2.09, which now allows the use of the element symbol specification in the PDB format to define the atom type uniquely. (If no atom type is specified, in SOLVE/RESOLVE the first character of the atom name is now used to define the atom type for ATOM records, and all characters of the atom name are used in HETATM records. This is all because otherwise it is difficult to tell the difference between C-alpha and calcium atoms without the element symbol specification, and many PDB files do not contain element symbol specifications).

- RESOLVE version 2.10 fixes a bug in which a list can be overrun in model-building when the version is just big enough to run. The symptom was the cryptic error message "Please increase size of n_ca_max in iter_frag_place".
- RESOLVE version 2.10 has a very powerful loop fitting algorithm that you can run using a script or by using the new PHENIX software.

Copyright notice and acknowledgements

Acknowledgements

I am deeply thankful for a collaboration with Joel Berendzen that has produced many of the Bayesian approaches used by SOLVE. Many thanks to Richard Fisher who wrote the FFT and file manipulation routines used here and to the many others who write freeware that I have used. Finally, I am indebted to the users of HEAVY versions 1 to 4.5 who have provided me with valuable feedback that has greatly improved this new program.

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
setenv SYMOP /usr/local/lib/solve/symop.lib
#
solve <<EOD > solve.log
!command file to read in raw MAD data, scale, analyze and solve it----
checksolve                    ! compare solution with known h.a. sites
@solve.setup                  ! get our standard information read in
logfile mad.logfile           ! write out most information to this file.
                              ! summary info will be written to solve.prt
readformatted                 ! alternatives are readdenzo, readtrek
premerged                     ! alternative is unmerged
read_intensities              ! alternative is read_amplitudes
refscattfactors               ! alternative is fixscattfactors



mad_atomname se               ! anomalously scattering atom is Se

lambda 1                      ! info on wavelength #1 follows
label Wavelength #  1         ! a label for this wavelength
rawmadfile test_wva.fmt       ! datafile with h k l Intensity sigma or
                              ! h k l I+ sigma+ I- sigma-
wavelength 0.9000             ! wavelength value
fprimv_mad  -1.6              ! f' value at this wavelength
fprprv_mad  3.4               ! f doubleprime value at this wavelength

! input refined h.a. coordinates (used only for comparison in "checksolve")
! offset by 1/2 in y to match gvp.pdb 2-27-01.
atomname se
 XYZ   0.4813319     0.4972169     9.4140753E-02
 XYZ   0.9731338     0.7875228     0.9446641

lambda 2
rawmadfile test_wvb.fmt
wavelength 0.9794
fprimv_mad  -8.5
fprprv_mad  4.8
```

```
lambda 3
rawmadfile test_wvc.fmt
wavelength 0.9797
fprimv_mad  -9.85
fprprv_mad  2.86
premerged
readformatted
nres 100                [approx # of residues in protein molecule]
nanomalous 2              [approx # of anomalously scattering atoms per protein]
SCALE_MAD               ! read in and localscale the data
ANALYZE_MAD              ! run MADMRG and MADBST and analyze all the Pattersons
SOLVE                   ! Solve the structure
EOD
#
# Now run Resolve to do density modification
# (You can download it from http://resolve.lanl.gov
# if you do not have it yet)
#
resolve << EOD > resolve.log
!solvent_content 0.40        !    solvent fraction
seq_file gvp.seq
compare_file coords.pdb
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#
echo 'Here are your SOLVE and resolve files:'
#
ls -l solve.prt solve.mtz solve.ezd resolve.mtz
#
echo 'All done.'
```

```
#!/bin/csh
#
#  set CCP4 and SOLVETMPDIR variables:
#
setenv CCP4_OPEN UNKNOWN
setenv SOLVETMPDIR /var/tmp
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
setenv SYMOP /usr/local/lib/solve/symop.lib
#
#
solve_giant <<EOD > solve.log

!command file to read in raw MAD data, scale, analyze and solve it----
title armadillo repeat of beta catenin 4-wavelength MAD data
logfile mad.logfile          ! write out most information to this file.
                             ! summary info will be written to "solve.prt"
@solve.setup                 ! get our standard information read in
readformatted                ! or: readdenzo, readtrek, readccp4_unmerged
unmerged                     ! or; premerged

mad_atom se

refscattfactors              ! do not refine scattering factors (you can if
                             ! you want though)

       !  Comment out next line if you don't know any sites
checksolve                   ! compare solutions to the one input below

lambda 1                     ! info on wavelength #1 follows
label Wavelength #  1        ! a label for this wavelength
rawmadfile l1.int
wavelength 0.9000            ! wavelength value
fprimv_mad  -1.6             ! f' value at this wavelength
fprprv_mad  3.4              ! f" value at this wavelength

! list of all SE positions in refined beta-catenin
! structure (offset by 0.5 in y from PDB file)

atomname se
xyz  0.2631041     0.6633824     2.8978506E-02
xyz  0.4166300     0.6113137     7.7325497E-03
xyz  0.4765674     0.7249608     2.5320712E-02
xyz  0.4591554     0.7427059     0.4719517
xyz  0.4083922     0.7455686     0.1403100
xyz  0.4416372     0.8393628     7.6342024E-02
xyz  0.1327285     0.4970000     0.4364171
xyz  9.6379094E-02  0.5855882     0.3802352
xyz  7.6066948E-02  0.6245000     0.3974865
xyz  0.1150683     0.7795883     0.3715025
xyz  0.1385160     0.7238529     0.4098982
xyz  9.2073016E-02  0.7063529     0.4022779
xyz  0.2152710     0.8265882     0.3764597
xyz  0.3304202     0.6161765     0.2311389
xyz  0.1806852     0.8512745     0.1618233

lambda 2
rawmadfile l2.int
wavelength 0.9794
fprimv_mad  -11.44
fprprv_mad  8.74

lambda 3
rawmadfile l3.int
wavelength 0.9797
fprimv_mad  -12.83
fprprv_mad  2.56

lambda 4
rawmadfile l4.int
wavelength 0.9897
fprimv_mad -2.42
fprprv_mad 1.13

nres 700                [approx # of residues in protein molecule]
nanomalous 15           [approx # of anomalously scattering atoms per protein]
acceptance 0.10
SCALE_MAD               ! read in and localscale the data
ANALYZE_MAD             ! run MADMRG and MADBST and analyze all the Pattersons
SOLVE                   ! Solve the structure
EOD
#
# Now run Resolve to do density modification
# (You can download it from http://resolve.lanl.gov
# if you do not have it yet)
#
```

```
resolve_giant << EOD > resolve.log
!solvent_content 0.40        !    solvent fraction
seq_file weis.seq
compare_file coords_met.pdb
EOD
#
#  That's it! Now resolve.mtz has your updated phases
#
echo 'Here are your SOLVE and resolve files:'
#
ls -l solve.prt solve.mtz solve.ezd resolve.mtz
#
echo 'All done.'
```

Contents	Index

The routine SOLVE: the core of automated structure determination by SOLVE

The SOLVE routine is an exceptionally powerful feature of this package that can find and evaluate the quality of heavy-atom sites in a MIR, SIR, or MIR-like dataset. The SOLVE routine treats MAD data almost exactly like MIR data, beginning with the output from MADMRG and MADBST.

Ordinarily SOLVE is called after SCALE_MAD and ANALYZE_MAD or SCALE_MIR and ANALYZE_MIR as part of automated structure determination. In this case you don't have to worry about all the keywords because the previous routines figure them out for you and write them to the script file solve_mad.script (or solve_mir.script).

You can, however, control much of what SOLVE does by setting keywords before running it. SOLVE can also be called using the solve_mad.script or solve_mir.script file written out by ANALYZE_MAD or an edited version of this file.

For MAD datasets, SOLVE uses a "compressed" form of MAD data that can be analyzed much more rapidly than the full n-wavelength data. This compressed dataset is generated by MADMRG in ANALYZE_MAD . The compressed dataset essentially consists of the SIR+anomalous scattering equivalent to the full MAD dataset. This dataset can be used to refine heavy atom parameters and generate native phases more quickly than a MAD dataset can. At the conclusion of SOLVE, phases are calculated with full Bayesian correlated MAD phasing.

The SOLVE routine operates by using a new version of HASSP to generate a few or many possible "seed" solutions for the anomalously scattering atoms in the structure. The heavy-atom parameters in each seed are first refined using the very fast refinement procedure in HEAVY (origin-removed patterson refinement). The refined seed is then used in self-difference Fouriers to suggest possible additional sites. A number of solutions are scored based on each seed, each solution being evaluated based on both the difference Patterson and a "free" difference Fourier. Additionally, the non-randomness of the native Fourier is used to judge the quality of a solution and to identify the correct hand of the structure if anomalous data is present. The figure of merit of phasing is the final scoring criterion.

If desired, a solution may be read in and evaluated directly with ANALYZE_SOLVE. Also, a solution may be read in and used as a seed in generating additional sites and a more complete solution with ADDSOLVE.

Using SOLVE is quite easy, particularly since ANALYZE_MAD or ANALYZE_MIR writes out a script file (usually solve_mad.script or solve_mir.script) that has everything you need to run SOLVE.

The only really non-obvious thing you need to know about running SOLVE on MAD data is that it requires 2 input data files. One is the compressed datafile from MADMRG, usually called "solve.data". The other is the full MAD dataset, usually called "mad_fpfm.scl". SOLVE uses "solve.data" for most of its analyses, then switches to the full MAD dataset at the very end.

The way you enter information on scattering factors is a little different in the SOLVE routine from the way it was entered in SCALE_MAD and ANALYZE_MAD . In the SOLVE routine you define atom types for each wavelength and specify the scattering factors for that atom type. Then you tell SOLVE what atom type goes with which wavelength. In SCALE_MAD, in contrast, you specified scattering factors directly for each wavelength. The reason for the difference is that SOLVE has to deal with both MAD and MIR data and defining atom types is a simple way to do that.

The solve_mad.script control file for MAD data

A sample SOLVE script file that will give you an idea of what you need to specify and what other things you can specify follows. This script is an edited version of a script file written out by the ANALYZE_MAD routine.

This script file is written out during automated SOLVE operation. You may wish to edit the one SOLVE has written out for you and use it if:

- you want to change how the core SOLVE routine solves your structure
- you want to use ANALYZE_SOLVE or ADDSOLVE to add sites to a solution you have found or to analyze a solution you have found

Sample MAD script file for SOLVE routine

```
!------------------solve_mad.script: solve a MAD problem---------------------
@solve.setup
 LOGFILE solve.logfile

 INFILE solve.data             !input file with MADMRG-compressed data
 MADFPFMFILE mad_fpfm.scl      !input file with full MAD dataset

 JSTD  1                       ! Lambda 1 is reference wavelength used in MADMRG
 IMADPHASE  1                  ! this is a MAD dataset, reference
                               !   wavelength is #1 (should match jstd)

 NNATF  1                      ! Pseudo-native F is column 1 of solve.data
 NNATS  2                      ! sigma is column 2

 ! Atom definitions with f' and f" values for the 3 wavelengths:
 NEWATOMTYPE LAM1
 AVAL  17.0006 5.8196 3.9731 4.3543
 BVAL  2.4098 .2726 15.2372 43.8163
 CVAL  2.8409
 FPRIMV  -1.6
 FPRPRV  3.4
 NEWATOMTYPE LAM2
 AVAL  17.0006 5.8196 3.9731 4.3543
 BVAL  2.4098 .2726 15.2372 43.8163
 CVAL  2.8409
 FPRIMV  -8.5
 FPRPRV  4.8
 NEWATOMTYPE LAM3
 AVAL  17.0006 5.8196 3.9731 4.3543
 BVAL  2.4098 .2726 15.2372 43.8163
 CVAL  2.8409
 FPRIMV  -9.85
 FPRPRV  2.86


 LAMBDA  1                           !  This is wavelength #1
 LABEL Wavelength  1 from MADMRG     !  label for lambda 1
```

```
NCOLFBAR  3                         ! Ncolfbar...ncolsdelf are column #'s
NCOLSFBAR  4                        ! in solve.data (MADMRG-compressed)
NCOLDELF  5                         ! datafile
NCOLSDELF  6
INPHASE
INANO
NOREFINESCALE                       ! Don't refine overall scale factor
                                    ! because this is MADMRG data

! Information for MADPHASE:
NCOLFPLUS  1                        ! these 4 column numbers refer to the
NCOLSIGPLUS  2                      ! full MAD datafile (mad_fpfm.scl)
NCOLFMINUS  3
NCOLSIGMINUS  4

! Heavy atoms for this wavelength:
ATOMNAME LAM1                       ! "LAM1" tells the program to use
OCCUPANCY  .1                       ! the scattering factors input above for
BVALUE  35.0                        ! LAM1
REFINEALL                           ! the occupancy and b values are guesses

LAMBDA  2
LABEL Wavelength  2 from MADMRG
NCOLFBAR  3
NCOLSFBAR  4
NCOLDELF  5
NCOLSDELF  6
INPHASE
INANO

! Information for MADPHASE:
NCOLFPLUS  5
NCOLSIGPLUS  6
NCOLFMINUS  7
NCOLSIGMINUS  8

! Heavy atoms for this derivative/wavelength:
ATOMNAME LAM2

LAMBDA  3
LABEL Wavelength  3 from MADMRG
NCOLFBAR  3
NCOLSFBAR  4
NCOLDELF  5
NCOLSDELF  6
INPHASE
INANO

! Information for MADPHASE:
NCOLFPLUS  9
NCOLSIGPLUS  10
```

```
 NCOLFMINUS  11
 NCOLSIGMINUS  12

 ! Heavy atoms for this derivative/wavelength:
 ATOMNAME LAM3

 ! Information for HASSP and SOLVE
 NCOLFHCOS  9                       ! column #s for <fh cos theta>
 NCOLFHSIN  10                      ! and <fh sin theta> in solve.data
 PATTFFTFILE patterson.patt         ! name of Bayesian patterson calculated
                                       !  by MADBST

 SOLVE                                 ! run SOLVE
!----------------------------------------------------------------------------
```

The solve_mir.script control file for MIR data

Using SOLVE is quite easy with MIR data too, particularly since ANALYZE_MIR writes out a script file that has everything you need to run SOLVE. A sample SOLVE script file that will give you an idea of what you need to specify and what other things you can specify follows. This script is an edited version of a script file written out by the ANALYZE_MIR routine.

This script file is written out during automated SOLVE operation. You may wish to edit the one SOLVE has written out for you and use it if:

- you want to change how the core SOLVE routine solves your structure
- you want to use ANALYZE_SOLVE or ADDSOLVE to add sites to a solution you have found or to analyze a solution you have found

Sample script file for SOLVE (MIR data)

```
!------------------solve_mir.script: solve an MIR problem----------------------
@solve.setup

LOGFILE solve.logfile

INFILE mir_fbar.scl                  !input file with Fnat,sig, and
                                     !(fbar,sig,delano,sig) for each derivative..

NNATF 1                              ! Native F is column 1 of mir_fbar.scl
NNATS 2                              ! sigma is column 2

Derivative 1                         ! begin information about derivative 1
LABEL deriv 1 HG                     ! label for deriv 1
NCOLFBAR 3                           ! Ncolfbar...ncolsdelf are column #'s
NCOLSFBAR 4                          ! in mir_fbar.scl datafile
NCOLDELF 5
NCOLSDELF 6
```

```
INANO                                ! include anomalous differences

! Heavy atoms for this derivative:

ATOMNAME HG                          ! the atom type is "HG"
OCCUPANCY .1                         ! guess for occupancy
BVALUE 35.0                          ! guess for bvalue
REFINEALL                            ! refine everything that is reasonable

Derivative 2                         ! begin information about derivative 2
LABEL deriv 2 Iodine                 ! label for deriv 2
NCOLFBAR 7                           ! Ncolfbar...ncolsdelf are column #'s
NCOLSFBAR 8                          ! in mir_fbar.scl datafile
NCOLDELF 9
NCOLSDELF 10
INANO                                ! include anomalous differences

ATOMNAME I-                          ! the atom type is "I-"
OCCUPANCY .1                         ! guess for occupancy
BVALUE 35.0                          ! guess for bvalue
REFINEALL                            ! refine everything that is reasonable

SOLVE                                ! run SOLVE
!-----------------------------------------------------------------------------
```

Keywords for the solve_mad.script and solve_mir.script files

There are a lot of keywords that can affect what SOLVE does. Ordinarily you do not have to worry about most of these because they are all set for you in ANALYZE_MAD. The solve_mad.script file written out by ANALYZE_MAD or the solve_mir.script file written by ANALYZE_MIR will have most of these keywords set for you. The keywords are listed here so that you can understand what they do and so that you can set them if you want to.

Most of these keywords can be specified at the beginning of automated data analysis to control what happens when SOLVE is called. For example, typing "ntopsolve 2" in the keywords before running SCALE_MAD and ANALYZE_MAD will affect SOLVE when it is called by restricting the number of solutions analyzed at the end of the routine to 2.

SOLVE treats MAD phasing and MIR phasing in almost exactly the same way except at the very end of the routine. Consequently "derivative" and "lambda" have the same meaning to SOLVE. You can enter information about lambda 1 by typing "lambda 1" or "derivative 1". The keywords that are specific to MAD phasing are listed at the top of the list.

Keywords that have a meaning for MAD data but not for MIR data:

```
INFILE  xxx.data    Principal input dorgbn-style file with compressed MAD data
                    from MADMRG and optional additional columns
                    of data. (usual file name = "solve.data").  This file
                    is usually produced by ANALYZE_MAD.


MADFPFMFILE yyy.scl Additional input file with (F+,sigma,F-
                    ,sigma) for each wavelength will be yyy.scl.
                    This file is used at the very end of SOLVE
                    for Bayesian correlated MAD phasing if the
                    keyword "bayes" is set in ANALYZE_MAD or the
                    keyword "imadphase n" is set in SOLVE.  All the
                    wavelengths have "inphase" specified for this
                    work. (DEFAULT="mad_fpfm.scl")

JSTD n              wavelength to be used as reference (default = lowest wavelength)

IMADPHASE n         This is a MAD dataset, n should match JSTD n

NOREFINESCALE      include this for all wavelengths usually because the
                    refinements in SOLVE are based on MADMRG output which
                    should not be further refined.
                    If xx is not recognized by SOLVE you need to
                    specify instead:



Keywords that apply to both MAD and MIR data:








NNATF n             column # in "infile" for native F (pseudo-native for MAD)
NNATS n             column # in "infile" for sigma of native f




 gotoderiv n        go to derivative (wavelength) n and get ready to read some
                    modifications of the parameters for this wavelength

 gotoatom n         go to the n'th atom in this wavelength/derivative  and get
                     ready to read some modifications of its parameters

 LABEL xxxxxx       label for this wavelength/derivative

 NCOLFBAR  n        column # for Fbar for this wavelength/derivative
                     For MAD data, this and the next three values are only
                       needed for the one wavelength defined by JSTD
                     For MIR data, they are needed for all derivatives
 NCOLSFBAR  n       column # for sigma of Fbar
 NCOLDELF  n        column # for delAno (if INANO is specified)
 NCOLSDELF  n       column # for sig of delAno
```

```
NCOLFHCOS  xx     column # in "infile" for estimated heavy atom structure
                    factor component along native structure
                    factor. (Output from MADBST for MAD data). This will be
                    used in calculation of heavy atom difference
                    Fouriers if ncolfhsin is also specified.
                    For MIR data, you can specify which derivative this applies
                    to by replacing the "1" in "ncolfhcos(1)" with another
                    derivative number. NCOLFHCOS is equivalent to NCOLFHCOS(1)

NCOLFHSIN  xx      column # in "infile" for estimated of heavy atom structure
                    factor perpendicular to native structure
                    factor. See ncolfhcos. NCOLFHSIN is equivalent to
                    NCOLFHSIN(1)

PATTFFTFILE xxxxxx   MAD data: Use previously calculated Patterson FFT
                      xxxxxx as the patterson map for the anomalously
scattering
                      atoms in this MAD structure.
                      MIR data: use patterson FFT xxxxx as patterson map for
                      derivative #1. PATTFFTFILE is equivalent to
                      PATTFFTFILE(1)(For other derivs, change the "1" to the
                      appropriate derivative number).
```

Also see all the commonly-used keywords for SOLVE.

SOLVE scoring

SOLVE uses four scoring criteria to evaluate each solution:

- Analysis of difference Pattersons
- "Free" self-difference Fourier analyses
- Non-randomness test on native fourier
- Figure of merit of phasing

SOLVE uses the 10-30 seeds generated from the analyses of Patterson functions as trial solutions and scores each seed. From this set of scores, an average score and the standard deviation of this average score is obtained for each criteria. A Z-score (number of standard deviations above the mean) is then calculated for each trial solution for each criteria. The overall Z-score for a solution is the sum of the individual Z-scores, corrected for any large deviation among the scores.

Analysis of difference Pattersons

The analysis of the difference Pattersons for a MAD dataset are carried out in two steps. First all Patterson vectors that should result from all the sites in a derivative are calculated. Then the difference Patterson is examined at these positions and the relative occupancies of all the sites are refined so as to match the Patterson as closely as possible. The comparison of observed and calculated peak heights*1000/rms of the map is printed out along with an overall quality of the solution and evaluation of the fit to the difference Patterson.

"Free" self-difference Fourier analyses

The difference fouriers calculated here use all anomalously scattering atoms except just one atom to be examined (and all that are equivalent to it) to calculate native phases. These native phases are used to calculate a difference Fourier; the Fourier is examined at the coordinates of the test site and the peak height/rms of the map is noted. This is repeated for each anomalously-scattering atom in the structure and the heights are printed out.

Non-randomness test on native fourier

Most protein crystals have solvent regions (with low density and low variation in electron density) and protein regions (with high variation of density and moderate average density). This program examines the standard deviation (SD) of the local electron density in various locations around the unit cell and determines if the variation of this SD is high or low. If the phases are random, then you get a low variation of the SD (the whole unit cell is rather uniform and looks like noise). If they are good, you get a high variation of the SD (part of the unit cell is smoother than another part). The program also analyzes the correlation of local r.m.s. density to see whether adjacent regions in the map tend to have similar r.m.s. variation. This is essentially a measure of how contiguous the regions of high and low variation are. This measure is reported back as the "correlation coefficient" (CC) in SOLVE. In versions 1.11 and higher the correlation coefficient is used in the actual scoring procedure (and not the standard deviation).

Figure of merit of phasing

The figure of merit of phasing the native structure proves to be a useful criteria in scoring a solution. Solutions with low figures of merit are unlikely to be correct, while those with a high figure of merit are likely to be right if the other criteria are also favorable.

Correction for uneven scores

SOLVE makes a correction on the score to reflect the discrepancy between the various scores. The effect is to score a solution that has a uniformly good score for all criteria higher than a solution with a very good score on just one criteria.

Back to RESOLVE table of contents

Quick Start

How to use RESOLVE

You can run RESOLVE right after SOLVE (using solve.mtz as input), or you can run it using an mtz-format file written by another program.

Usually if you have MAD/SAD/MIR data, you will want to (1) edit and run one of the standard SOLVE scripts(to solve your structure and calculate an initial map, and (2) edit and run the RESOLVE_BUILD script to density modify the map and build an atomic model. With fast model-building, the whole process only takes a few hours for a small protein.

If you have an MR model and FOBS data, you will want to run the RESOLVE_BUILD script to remove model bias and build a new atomic model.

If you have non-crystallographic symmetry, you will want to include that in your resolve scripts, either by specifying the NCS matrices (see ***Keywords*** for resolve) or by specifying the coordinates of some of the atoms in each monomor with a PDB file "ha.pdb" (this can be the heavy-atom file written by SOLVE for MIR/MAD/SAD data, or a file you create from some of the CA atoms in your starting PDB file for your MR solution).

Running RESOLVE right after SOLVE

It is really easy to run RESOLVE right after SOLVE. Just go to the directory where you ran SOLVE and type (or put in a command file):

```
#!/bin/csh
# Here is a very minimal script to run RESOLVE:
# Set CCP4 variables for symmetry information and
# for file handling:
setenv SYMOP /usr/local/lib/solve/symop.lib
setenv SYMINFO /usr/local/lib/solve/syminfo.lib
setenv CCP4_OPEN UNKNOWN
# Now run RESOLVE:
resolve<<EOD
solvent_content 0.4             ! your solvent content goes here.
!             Next line is protein sequence file
seq_file protein.seq
EOD
# Now "resolve.mtz" has the output amplitudes, weighted F, phases,
# figure of merit and HL-coeffs in columns labelled: FP PHIM FOMM HLAM HLBM HLCM HLDM
# Also resolve.pdb contains your model and the heavy-atom sites from SOLVE.
#
```

That's it! (Sure, there are a few other keywords, but usually you don't need them.) (More sample scripts are available too.)

- RESOLVE will read from "solve.mtz" and write out new structure factor amplitudes and phases to "resolve.mtz."
- You can use "resolve.mtz" in the same way as you use any mtz file in the CCP4 suite.
- The new phases are called PHIM and the figure of merit is FOMM (M is for modified).
- If there is a file called "ha.pdb" in your working directory, RESOLVE will assume it contains heavy-atom sites in PDB format and will try to find NCS in them and to apply it if it exists. It will ignore the NCS if the correlation is very low.
- RESOLVE will build a model of your structure into the density after doing statistical density modification. If you specify seq_file, then it expects a file with the sequence of your protein (1-letter code); if your protein has more than one unique chain, separate them with a line starting with the characters ">>>". No need to enter multiple copies of the same chain.
- Resolve will initially use only data in the resolution range where the starting figure-of-merit is significant, then will carry out phase extension.

Back to RESOLVE table of contents

RESOLVE output

The output from RESOLVE follows. The data are for the molecular replacement example using initiation factor 5A shown in the *example* section of the manual. You can run this data through yourself using the data and command files in the SOLVE/RESOLVE library installation directory, usually located at the place referred to by $SOLVEDIR/examples_resolve/

```
 ------------------------  COPYRIGHT NOTICE  ---------------------------------
                   Los Alamos National Laboratory
 This program was prepared by the Regents of the University of California at
 Los Alamos National Laboratory (the University) under  Contract No.
 W-7405-ENG-36 with the U.S. Department of Energy (DOE).  The University has
 certain rights in the program pursuant to the contract and the program should
 not be copied or distributed outside your organization.  All rights in the
 program are reserved by the DOE and the University.  Neither the U.S.
 Government nor the University makes any warranty, express or implied, or
 assumes any liability or responsibility for the use of this software.

        *******************************************************
        *                --- Resolve---                      *
        *                                                     *
        *    Reciprocal-space solvent flattening and          *
        *    Maximum-likelihood density modification          *
        *                                                     *
        *     Type "resolvehelp" for on-line help             *
        *      or see "http://resolve.lanl.gov"               *
        *******************************************************

              (version 2.00 of 22-Mar-2001)
   Tom Terwilliger, Los Alamos National Laboratory, "terwilliger@LANL.gov"

>hklin sigmaa_2EIF.mtz
 Data to be read from mtz file:
 sigmaa_2EIF.mtz



>labin FP=FP FC=FC PHIC=PHIC FOM=WCMB
 Current value of "LABIN" is:
 labin FP=FP FC=FC PHIC=PHIC FOM=WCMB



>mask_cycles 10
 Number of mask/image cycles:          10

>minor_cycles 10
```

```
 Number of minor cycles per mask/image   cycle:           10

>solvent_content 0.60        !    solvent fraction
 Fraction of unit cell assumed to be solvent is:   0.6000000

>prime_and_switch
 Prime-and-switch phasing will be used (requires FC PHIC FP to be input).

>hklout resolve_ps.mtz
 Data to be written to mtz file:
 resolve_ps.mtz


 All done with inputs

       The date today is 12-apr-01.  Your license is good until 15-jul-07.

 User:   terwill              Logical Name: sigmaa_2EIF.mtz
 Status: READONLY   Filename: sigmaa_2EIF.mtz
 HEADER INFORMATION FROM INPUT MTZ FILE ON INDEX  1

 * Title:

 ...

 * Number of Columns =  11

 * Number of Reflections =  12490

 * Missing value set to NaN in input mtz  file

 * Column Labels :

 H K L FC PHIC FOMM FP SIGFP DELFWT FWT WCMB

 * Column Types :

 H H H F P W F Q F F W

 * Cell Dimensions :

   113.95  113.95   32.47   90.00   90.00   90.00

 *  Resolution Range :

     0.00262     0.22672      (   19.537 -     2.100 A )

 * Sort Order :

     1     2     3     0     0

 * Space group = I4  (number    79)
 * Input Program Labels :

 H K L FP PHIB FOM HLA HLB HLC HLD FC PHIC SIGFP FWT

 * Input File Labels :

 H K L FC PHIC FOMM FP SIGFP DELFWT FWT WCMB

 * Lookup Table : the number indicates the input column no.
 * Array element n corresponds to the nth program label

     0    0    0    7    0   11    0    0    0    0    4    5    0    0

 FP taken from column           7
 FOM taken from column          11
 Reading in FC and PHIC from columns            4           5
 FOM will be applied to PHIC
 Total of        12490 reflections read from file
 Mean FOM of input data =   0.2678500
 Adding F000 term (0.0) to this list
 Closed mtz file
 Space group is          79

 FORMATTED      OLD     file opened on unit  11
 Logical name: SYMOP, Full name: /usr/local/lib/solve/symop.lib

   Centric Zone   1 Reflections of Type  hk0

 Cycles of mask generation:          10
 Sub-cycles per mask cycle:          10


 Warning -- Total of            1 centric phases off by >   0.5000000
  degrees

 Expected I corrected for centering by factor of            2
 Results of wilson scaling:
 Scale on I = 1.89
 B-value = 9.66



 Setting up small boxes for cross-validation
Subdivisions along X Y Z:    6   3   1 for total of   18 boxes.

 Results of wilson scaling of model Fc to Fo :
 Scale on I to apply to Fc = 3.02
 B-value to apply to Fc= .128E-01



 Estimates of overall B and scale for fitting Model map to Observed intensities
 Overall B =     25.6 Corresponding to rmsd of about      0.6 A.
Overall scale, corresponding to fraction of atoms modelled  with this error =   0.28

       CC of Observed, Model Intensities vs resolution
       (CALC is fit to the observed CC with a pseudo-B)
             ACENTRIC                        CENTRIC
           N        CC     CALC          N        CC     CALC           DMIN
          20      0.42    0.27          13     -0.36    0.27          14.00
          53      0.35    0.27          23      0.20    0.27          10.11
         111      0.33    0.26          37      0.16    0.26           7.78
         196      0.10    0.25          44      0.13    0.25           6.22
         211      0.30    0.23          41      0.41    0.23           5.44
         365      0.39    0.22          67      0.01    0.22           4.67
         457      0.39    0.20          63      0.27    0.20           4.12
         511      0.24    0.19          66      0.25    0.19           3.73
         606      0.10    0.17          71      0.29    0.17           3.42
         621      0.13    0.16          64      0.22    0.16           3.19
         817      0.13    0.14          82      0.23    0.14           2.96
         738      0.10    0.13          64      0.14    0.13           2.80
         893      0.06    0.12          83      0.17    0.12           2.64
        1123      0.03    0.11          92     -0.02    0.11           2.49
        1484      0.06    0.09         114     -0.07    0.09           2.33
        1395      0.09    0.08          99      0.07    0.08           2.22
        1751      0.09    0.07         115      0.15    0.07           2.10
Scaling model reflections with overall scale and B

Resolution cutoff recommended based on CC  of FP and FC is:    2.100175


Overall R-factor for FC vs FP:  0.45 for      12491 reflections

Reset-resolution with adjusted_resolution of   2.10 and NREFL =    12491

Total of       0 reflections <  2.10 tossed, leaving a total of   12491

Getting starting map  using current phases


                        Mask cycle           1

big_cycle
get_mask_wang
New Wang averaging radius = 11.57

Mean +/- SD of density in protein region :   0.00 +/-   0.30
Mean +/- SD of density in solvent region :   0.00 +/-   0.13



-------------------------------------------------------------------------------

Plot of probability that a grid point is part of protein region
vs percentiles of grid points
```

```
All points to the left of the "+" signs are in solvent masked region
those to right are in protein masked region.

The values of p(protein) should change from low to high approximately at the value
of the fraction of solvent indicated by the "+" signs.
The sharper the transition the better.

Note: the mask is only used to make an estimate of the p(protein)

The values of p(protein) are used to weight the contribution of each grid
 point to the likelihood of the map:

 p(rho) = p(rho|protein) p(protein) +  p(rho|solvent) (1-p(protein))

This says that the probability that we would observe  the value "rho"  of electron
density at this point is the probability that we would observe "rho" if this were
really protein times the probability that this is  protein, plus the probability
that we would observe "rho" if it were really solvent, times the probability that
it is solvent.

                              Probability that grid points are in protein region


                        1.0  ...........................................xxxxxxx
                             .                             +         xxx      .
                             .                             +       xx         .
                             .                             +     xx           .
                             .                             +   xx             .
                             .                             + xx               .
                             .                             +x                 .
               p(protein)    .                            xx                  .
                        0.5  .                           x +                  .
                             .                          x  +                  .
                             .                         x   +                  .
                             .                        x    +                  .
                             .                      xx     +                  .
                             .                     x       +                  .
                             .                  xxx        +                  .
                        0.0  .xxxxxxxxxxxxxxxxxx...........+...................

                             0       20        40        60        80       100

                                          Percentile of grid points
-------------------------------------------------------------------------------



Range of P(protein)  and percentiles
used for histograms of protein and solvent:

             P(protein)                    Percentile
             Low      High                 Low      High
Protein:    0.85      1.00              80.      100.
Solvent:    0.00      0.12               0.       44.


Minor cycle    1



Plot of Observed (o) and model (x) electron density distributions for protein region
where the model distribution is given by,  p_model(beta*(rho+offset)) = p_ideal(rho)
and then convoluted with a gaussian with width of sigma
where sigma, offset and beta are given below under "Error estimate."


                         0.05..................................................
                             .                   .                            .
                             .                   .                            .
                             .                xxoo                            .
                             .               xooxo                            .
                             .              xo   xo                           .
                             .             xx    .xo                          .
               p(rho)        .             x     . xo                         .
                             .            xo     .  x                         .
                             .            x      .  xx                        .
                             .          ox       .    xx                      .
                             .          x        .     xxx                    .
                             .         ox        .      ooxxxx                .
                             .        ox         .        oooxxxx             .
                             .      oxx          .             ooxxxxooo      .
                        0.0  xxxxxxxx................................xxxxxxxxxx

                            -2        -1         0         1         2        3

                                 normalized rho (0 = mean of solvent region)
-------------------------------------------------------------------------------




Plot of Observed (o) and model (x)
electron density distributions for solvent region


                         0.09..................................................
                             .                   .                            .
                             .                   .                            .
                             .                  oxo                           .
                             .                  x.xx                          .
                             .                  x. x                          .
                             .                 x . o                          .
               p(rho)        .                 x .  x                         .
```

```
                              .               x  .  x                       .
                              .               x  .  x                       .
                              .              oo  .  ox                      .
                              .              x   .   ox                     .
                              .             xx   .    x                     .
                              .             x    .    ox                    .
                              .           xx     .     xx                   .
                         0.0  xxxxxxxxxxxxxx.............xxxxxxxxxxxxxxxxxxxxxx

                             -2         -1         0         1         2        3

                                  normalized rho (0 = mean of solvent region)
 ------------------------------------------------------------------------------




      Error estimate for map on this cycle

 The estimated error in this map is    0.11 based on an analysis of the electron
 density distributions in the protein and solvent regions with rms
 values of    0.31 and    0.13 respectively.
 The value of the scale factor beta relating idealized density distributions
 P_ideal(rho) to observed ones is:

      obs_rho = beta * (ideal_rho + offset) + sigma

 where beta =    0.63, offset is   -0.09   and sigma is a
 random variable with rms value of    0.11



 Mean starting figure of merit this cycle
    Overall    Centric    Acentric
       0.27       0.27       0.27
      12490       1138      11352



Input phase probabilities weighted by factor of   0.00



 Mean normalized structure factor changes this cycle
    Acentric       Centric
       0.24          0.27
      11350          1136




 New figure of merit () of phasing based on:
      (1) experimental phase information, and
      (2) likelihood of the resulting map

 These are the 2 sources of phase information used in resolve.  The phase
```

```
information from them should be correlated and the phase information from
the map should increase during maximum- likelihood density modification.

Correlation between prior and map phase information is measured by
, the mean cosine of the phase difference.

Best estimate of true figure of merit of map-likelihood phasing is
ratio of correlation to  of prior information

Fraction of phase information from prior is estimated from  of prior, map




                 Acentric reflections only:

       --Figure of merit--      CC      Fraction     Total
DMIN   Prior   Map   Total   Prior-Map from Prior  (corrected)      N

ALL:    0.33   0.40   0.40     0.72       0.00         0.40      11352

 6.0    0.45   0.73   0.73     0.76       0.00         0.73        431
 3.8    0.50   0.80   0.80     0.82       0.00         0.80       1473
 3.0    0.36   0.62   0.62     0.75       0.00         0.62       1892
 2.6    0.30   0.36   0.36     0.72       0.00         0.36       1932
 2.3    0.26   0.22   0.22     0.69       0.00         0.22       3459
 2.1    0.31   0.17   0.17     0.65       0.00         0.17       2165

WGT:    0.00   1.00



                 Centric reflections only:

       --Figure of merit--      CC      Fraction     Total
DMIN   Prior   Map   Total   Prior-Map from Prior  (corrected)      N

ALL:    0.27   0.46   0.46     0.56       0.00         0.46       1138

 6.0    0.34   0.65   0.65     0.68       0.00         0.65        130
 3.8    0.38   0.67   0.67     0.54       0.00         0.67        222
 3.0    0.28   0.59   0.59     0.55       0.00         0.59        197
 2.6    0.24   0.40   0.40     0.56       0.00         0.40        179
 2.3    0.16   0.26   0.26     0.52       0.00         0.26        262
 2.1    0.27   0.25   0.25     0.57       0.00         0.25        148

WGT:    0.00   1.00
```

```
                 All reflections:

       --Figure of merit--      CC      Fraction     Total
DMIN   Prior   Map   Total   Prior-Map from Prior  (corrected)      N

ALL:    0.33   0.40   0.40     0.70       0.00        0.40      12490

 6.0    0.42   0.71   0.71     0.74       0.00        0.71        561
 3.8    0.49   0.78   0.78     0.78       0.00        0.78       1695
 3.0    0.35   0.62   0.62     0.73       0.00        0.62       2089
 2.6    0.29   0.36   0.36     0.70       0.00        0.36       2111
 2.3    0.25   0.23   0.23     0.68       0.00        0.23       3721
 2.1    0.31   0.17   0.17     0.65       0.00        0.17       2313

WGT:    0.00   1.00




*********************************************************************
*                                                                   *
*     CORRECTED OVERALL FIGURE OF MERIT OF PHASING:   0.40          *
*     BIAS RATIO:  5.39                                             *
*                                                                   *
*     Bias ratio is given by /(FOMA*FOMB),           *
*     where phiA and FOMA are phase and  from prior (model or     *
*     experiment) and phiB and FOMB are from map.                   *
*     Bias ratio is about 1 if phiA and phiB are independent,       *
*                        >1 if phiB is biased by phiA,              *
*                        <1 if FOMA or FOMB are overestimated       *
*                                                                   *
*     If BIAS RATIO <1, CORRECTED FOM = ESTIMATED FOM * BIAS RATIO   *
*********************************************************************


Cumulative phase change from start to end of this cycle []

DMIN  -----acentric--------  --------centric-------  ---------all---------
           N        N        N
 0.0    0.72    0.82  11352.   0.56    0.74   1138.   0.70    0.81  12490.
 6.0    0.76    0.81    431.   0.67    0.78    130.   0.74    0.80    561.
 3.8    0.82    0.85   1473.   0.54    0.71    222.   0.78    0.84   1695.
 3.0    0.75    0.81   1892.   0.55    0.71    197.   0.73    0.80   2089.
 2.6    0.72    0.81   1932.   0.56    0.73    179.   0.70    0.80   2111.
 2.3    0.69    0.81   3459.   0.52    0.75    262.   0.68    0.80   3721.
 2.1    0.65    0.79   2165.   0.57    0.83    148.   0.65    0.79   2313.


                        End of first cycle
```

```
Minor cycle    2



     Error estimate for map on this cycle

The estimated error in this map is    0.10 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.44 and    0.13 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.00, offset is   -0.07   and sigma is a
random variable with rms value of    0.10


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.40       0.46       0.40
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.17         0.17
     11351         1137


Minor cycle    3



     Error estimate for map on this cycle

The estimated error in this map is    0.11 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.48 and    0.13 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.05, offset is   -0.08   and sigma is a
random variable with rms value of    0.11


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
```

```
     0.45       0.52       0.45
    12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.12          0.11
    11351          1137



Minor cycle    4




     Error estimate for map on this cycle

The estimated error in this map is    0.11 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.48 and    0.13 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.05, offset is   -0.09   and sigma is a
random variable with rms value of    0.11



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.45       0.52       0.45
    12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.10          0.08
    11351          1137



Minor cycle    5




     Error estimate for map on this cycle

The estimated error in this map is    0.11 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.49 and    0.12 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:
     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.06, offset is   -0.10   and sigma is a
random variable with rms value of    0.11



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.46       0.53       0.45
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.08         0.07
     11351         1137



Minor cycle    6




     Error estimate for map on this cycle

The estimated error in this map is    0.11 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.49 and    0.12 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.06, offset is   -0.10   and sigma is a
random variable with rms value of    0.11



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.46       0.53       0.46
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.07         0.07
     11351         1136



Minor cycle    7
```

```
     Error estimate for map on this cycle

The estimated error in this map is    0.11 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.49 and    0.12 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.06, offset is   -0.11   and sigma is a
random variable with rms value of    0.11



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.47       0.54       0.46
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.06         0.06
     11351         1136



Minor cycle    8




     Error estimate for map on this cycle

The estimated error in this map is    0.11 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.49 and    0.12 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.06, offset is   -0.11   and sigma is a
random variable with rms value of    0.11



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.47       0.54       0.46
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
```

```
      0.06         0.06
     11351         1136



Minor cycle    9




     Error estimate for map on this cycle

The estimated error in this map is    0.11 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.50 and    0.11 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.06, offset is   -0.11   and sigma is a
random variable with rms value of    0.11



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.47       0.54       0.46
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.05         0.05
     11351         1136



Minor cycle   10




     Error estimate for map on this cycle

The estimated error in this map is    0.11 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.50 and    0.11 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.06, offset is   -0.11   and sigma is a
random variable with rms value of    0.11
```

```
Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.47       0.54       0.46
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.04         0.05
     11351         1136



                        Mask cycle            2

big_cycle
get_mask_wang
New Wang averaging radius =  6.47

Mean +/- SD of density in protein region :   0.01 +/-   0.50
Mean +/- SD of density in solvent region :   0.00 +/-   0.10



Minor cycle    1




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.50 and    0.10 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.07, offset is   -0.12   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.47       0.55       0.46
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.05         0.05
     11351         1136
```

```
Minor cycle    2



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.50 and    0.10 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.07, offset is   -0.15   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.55       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.04          0.04
     11351          1136


Minor cycle    3



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.50 and    0.09 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.07, offset is   -0.16   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352
```

```
Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.03          0.03
     11351          1136


Minor cycle    4



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.09 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.07, offset is   -0.17   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.55       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.03          0.03
     11351          1136


Minor cycle    5



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.09 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma
```

```
where beta =    1.07, offset is   -0.17   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.03          0.03
     11351          1136


Minor cycle    6



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.09 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.07, offset is   -0.18   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.02          0.03
     11351          1136


Minor cycle    7



     Error estimate for map on this cycle
```

```
The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.09 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.18   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.02          0.03
     11351          1136



Minor cycle    8




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.09 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.18   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.02          0.02
     11351          1136
```

```
Minor cycle    9




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.09 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.19   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.02         0.02
     11351         1136



Minor cycle   10




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.09 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.18   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
```

```
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.02         0.02
     11351         1136



                        Mask cycle            3

big_cycle
get_mask_wang
New Wang averaging radius =  6.37

Mean +/- SD of density in protein region :   0.00 +/-   0.51
Mean +/- SD of density in solvent region :   0.00 +/-   0.08



Minor cycle    1




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.20   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.02         0.02
     11351         1136



Minor cycle    2
```

```
     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.20   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.02         0.02
     11351         1136



Minor cycle    3




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.20   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352
```

```
Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.02         0.02
     11351         1136



Minor cycle    4




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.20   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136



Minor cycle    5




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.20   and sigma is a
random variable with rms value of    0.12
```

```
Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136



Minor cycle    6




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.20   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136



Minor cycle    7




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
```

```
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136



Minor cycle    8




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136
```

```
Minor cycle    9



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136


Minor cycle   10



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352
```

```
Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136


                        Mask cycle            4

big_cycle
get_mask_wang
New Wang averaging radius =  6.35

Mean +/- SD of density in protein region :   0.00 +/-   0.51
Mean +/- SD of density in solvent region :   0.00 +/-   0.08


Minor cycle    1




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1135



Minor cycle    2
```

```
     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136



Minor cycle    3




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
```

```
      0.01         0.02
     11351         1136


Minor cycle    4




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136



Minor cycle    5




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.21   and sigma is a
random variable with rms value of    0.12
```

```
Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.02
     11351         1136



Minor cycle    6




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    7




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
```

```
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.21   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    8



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    9
```

```
     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle   10




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.08 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.21   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.56       0.47
     12490       1138      11352
```

```
Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11351          1136



                        Mask cycle            5

big_cycle
get_mask_wang
New Wang averaging radius =  6.34

Mean +/- SD of density in protein region :   0.00 +/-   0.51
Mean +/- SD of density in solvent region :   0.00 +/-   0.07


Minor cycle    1




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11350          1136



Minor cycle    2




     Error estimate for map on this cycle
```

```
The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    3




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136
```

```
Minor cycle    4



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    5



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
```

```
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    6



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    7



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:
     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    8




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    9
```

```
     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle   10




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
```

```
      0.01         0.01
     11351         1136


                          Mask cycle            6

big_cycle
get_mask_wang
New Wang averaging radius =  6.33

Mean +/- SD of density in protein region :   0.00 +/-   0.51
Mean +/- SD of density in solvent region :   0.00 +/-   0.07


Minor cycle    1




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    2




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
```

```
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11351          1136



Minor cycle    3




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11351          1136
```

```
Minor cycle    4



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11351          1136


Minor cycle    5



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352
```

```
Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11351          1136



Minor cycle    6




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11350          1135



Minor cycle    7




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma
```

```
where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11351          1136


Minor cycle    8



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01          0.01
     11350          1136


Minor cycle    9



     Error estimate for map on this cycle
```

```
The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11350         1136



Minor cycle   10




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11350         1136
```

```
                        Mask cycle            7

big_cycle
get_mask_wang
New Wang averaging radius =  6.32

Mean +/- SD of density in protein region :   0.00 +/-   0.51
Mean +/- SD of density in solvent region :   0.00 +/-   0.07


Minor cycle    1



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    2



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
```

```
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    3



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136


Minor cycle    4
```

```
     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    5




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352
```

```
Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    6




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    7




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12
```

```
Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    8




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.01         0.01
     11351         1136



Minor cycle    9




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
```

```
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.08, offset is   -0.23   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00          0.01
     11351          1136



Minor cycle   10




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00          0.00
     11351          1136
```

```
                         Mask cycle            8

big_cycle
get_mask_wang
New Wang averaging radius =  6.32

Mean +/- SD of density in protein region :   0.00 +/-   0.51
Mean +/- SD of density in solvent region :   0.00 +/-   0.07


Minor cycle    1



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11351         1136


Minor cycle    2



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:
     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11351         1136



Minor cycle    3




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11351         1136



Minor cycle    4
```

```
     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11351         1136



Minor cycle    5




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
```

```
      0.00         0.00
     11350         1136



Minor cycle    6




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



Minor cycle    7




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12
```

```
Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



Minor cycle    8




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



Minor cycle    9




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
```

```
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136


Minor cycle   10



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136


                        Mask cycle            9
```

```
big_cycle
get_mask_wang
New Wang averaging radius =  6.32

Mean +/- SD of density in protein region :   0.00 +/-   0.51
Mean +/- SD of density in solvent region :   0.00 +/-   0.07


Minor cycle    1




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00          0.00
     11350          1136


Minor cycle    2




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma
```

```
where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



Minor cycle    3




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



Minor cycle    4




     Error estimate for map on this cycle
```

```
The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



Minor cycle    5




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136
```

```
Minor cycle    6



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136


Minor cycle    7



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
```

```
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136


Minor cycle    8




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136


Minor cycle    9




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:
     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



Minor cycle   10




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.22   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



                        Mask cycle           10

big_cycle
get_mask_wang
```

```
New Wang averaging radius =  6.31

Mean +/- SD of density in protein region :   0.00 +/-   0.51
Mean +/- SD of density in solvent region :   0.00 +/-   0.07



--------------------------------------------------------------------------------

Plot of probability that a grid point is part of protein region
vs percentiles of grid points

All points to the left of the "+" signs are in solvent masked region
those to right are in protein masked region.

The values of p(protein) should change from low to high approximately at the value
of the fraction of solvent indicated by the "+" signs.
The sharper the transition the better.

Note: the mask is only used to make an estimate of the p(protein)

The values of p(protein) are used to weight the contribution of each grid
 point to the likelihood of the map:

 p(rho) = p(rho|protein) p(protein) +  p(rho|solvent) (1-p(protein))

This says that the probability that we would observe  the value "rho"  of electron
density at this point is the probability that we would observe "rho" if this were
really protein times the probability that this is  protein, plus the probability
that we would observe "rho" if it were really solvent, times the probability that
it is solvent.

                              Probability that grid points are in protein region


                         1.0  ....................................xxxxxxxxxxxxxx
                              .                             +    x             .
                              .                             +   x              .
                              .                             +  x               .
                              .                             + x                .
                              .                             +                  .
                              .                             +x                 .
                p(protein)    .                             +                  .
                         0.5  .                             x                  .
                              .                            x+                  .
                              .                             +                  .
                              .                           x +                  .
                              .                          x  +                  .
                              .                         x   +                  .
                              .                       xx    +                  .
                         0.0  .xxxxxxxxxxxxxxxxxxxxxxxx......+..................
```

```
                              0       20         40         60         80        100

                                           Percentile of grid points
-------------------------------------------------------------------------------



Range of P(protein)  and percentiles
used for histograms of protein and solvent:

              P(protein)                 Percentile
             Low      High              Low      High
Protein:    0.89      1.00              70.      100.
Solvent:    0.00      0.13               0.       52.


Minor cycle    1



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352


Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136


Minor cycle    2



     Error estimate for map on this cycle
```

```
The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00          0.00
     11350          1136



Minor cycle    3




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00          0.00
     11350          1136
```

```
Minor cycle    4



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136


Minor cycle    5



     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12


Mean starting figure of merit this cycle
   Overall    Centric    Acentric
```

```
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11350         1136



Minor cycle    6




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11349         1136



Minor cycle    7




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:
     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11349         1136



Minor cycle    8




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11349         1136



Minor cycle    9
```

```
     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:

     obs_rho = beta * (ideal_rho + offset) + sigma

where beta =    1.09, offset is   -0.23   and sigma is a
random variable with rms value of    0.12



Mean starting figure of merit this cycle
   Overall    Centric    Acentric
      0.48       0.57       0.47
     12490       1138      11352



Mean normalized structure factor changes this cycle
   Acentric       Centric
      0.00         0.00
     11349         1136



Minor cycle   10
```

```
                        Final cycle




Plot of Observed (o) and model (x) electron density distributions for protein region
where the model distribution is given by,  p_model(beta*(rho+offset)) = p_ideal(rho)
and then convoluted with a gaussian with width of sigma
where sigma, offset and beta are given below under "Error estimate."


                           0.03..................................................
                               .                   .                            .
                               .                   .                            .
                               .          xxxxxxxoo .                           .
                               .         ox oooox oo                            .
                               .        ox    o  xx.                            .
                               .       ox         xxoo                          .
               p(rho)          .       x           xxo                          .
                               .      xx           . xo                         .
```

```
                             .    o x            .  xx                        .
                             .    xx             .   oxxx                     .
                             .    x              .    oooxxxxxx               .
                             .   xo              .      ooooooxxxxxxx         .
                             . xx                .               ooooxxxxxxo  .
                             xxo                 .                        xxxxo
                        0.0  o................................................x

                            -2        -1         0         1         2        3

                                 normalized rho (0 = mean of solvent region)
-------------------------------------------------------------------------------




Plot of Observed (o) and model (x)
electron density distributions for solvent region


                         0.18..................................................
                             .                   .                            .
                             .                   .                            .
                             .                   oo                           .
                             .                   .                            .
                             .                   .                            .
                             .                  o.o                           .
               p(rho)        .                   .                            .
                             .                  o. o                          .
                             .                   .                            .
                             .                  xxxo                          .
                             .                 x . xx                         .
                             .                x  .  xx                        .
                             .              xx o .    x                       .
                             .            xxx o  .  o  xx                     .
                        0.0  xxxxxxxxxxxxxoooo.......ooooxxxxxxxxxxxxxxxxxxxxxx

                            -2        -1         0         1         2        3

                                 normalized rho (0 = mean of solvent region)
-------------------------------------------------------------------------------




     Error estimate for map on this cycle

The estimated error in this map is    0.12 based on an analysis of the electron
density distributions in the protein and solvent regions with rms
values of    0.51 and    0.07 respectively.
The value of the scale factor beta relating idealized density distributions
P_ideal(rho) to observed ones is:
      obs_rho = beta * (ideal_rho + offset) + sigma

 where beta =    1.09, offset is   -0.23   and sigma is a
 random variable with rms value of    0.12


 Mean starting figure of merit this cycle
    Overall    Centric    Acentric
       0.48       0.57       0.47
      12490       1138      11352


Input phase probabilities weighted by factor of   0.00


 Mean normalized structure factor changes this cycle
    Acentric       Centric
       0.00          0.00
      11349          1136



 New figure of merit () of phasing based on:
      (1) experimental phase information, and
      (2) likelihood of the resulting map

 These are the 2 sources of phase information used in resolve.  The phase
 information from them should be correlated and the phase information from
 the map should increase during maximum- likelihood density modification.

 Correlation between prior and map phase information is measured by
 , the mean cosine of the phase difference.

 Best estimate of true figure of merit of map-likelihood phasing is
 ratio of correlation to  of prior information

 Fraction of phase information from prior is estimated from  of prior, map





                  Acentric reflections only:

        --Figure of merit--      CC      Fraction     Total
 DMIN   Prior   Map   Total   Prior-Map from Prior  (corrected)      N

 ALL:    0.34   0.47   0.47     0.23       0.00        0.47      11352

  6.0    0.45   0.79   0.79     0.46       0.00        0.79        431
  3.8    0.51   0.83   0.83     0.42       0.00        0.82       1473
 3.0    0.37   0.74   0.74     0.27        0.00        0.74        1892
 2.6    0.30   0.49   0.49     0.21        0.00        0.49        1932
 2.3    0.27   0.31   0.31     0.16        0.00        0.31        3459
 2.1    0.32   0.19   0.19     0.13        0.00        0.19        2165

WGT:    0.00   1.00




                 Centric reflections only:

       --Figure of merit--      CC      Fraction     Total
DMIN   Prior   Map   Total   Prior-Map from Prior  (corrected)      N

ALL:    0.27   0.57   0.57     0.24        0.00        0.57        1138

 6.0    0.34   0.68   0.68     0.38        0.00        0.68         130
 3.8    0.38   0.73   0.73     0.27        0.00        0.70         222
 3.0    0.28   0.69   0.69     0.30        0.00        0.69         197
 2.6    0.24   0.56   0.56     0.16        0.00        0.56         179
 2.3    0.16   0.42   0.42     0.22        0.00        0.42         262
 2.1    0.27   0.34   0.34     0.15        0.00        0.34         148

WGT:    0.00   1.00




                 All reflections:

       --Figure of merit--      CC      Fraction     Total
DMIN   Prior   Map   Total   Prior-Map from Prior  (corrected)      N

ALL:    0.33   0.48   0.48     0.23        0.00        0.48       12490

 6.0    0.42   0.76   0.76     0.44        0.00        0.76         561
 3.8    0.49   0.81   0.81     0.40        0.00        0.81        1695
 3.0    0.36   0.73   0.73     0.27        0.00        0.73        2089
 2.6    0.30   0.50   0.50     0.20        0.00        0.50        2111
 2.3    0.26   0.32   0.32     0.16        0.00        0.32        3721
 2.1    0.31   0.20   0.20     0.13        0.00        0.20        2313

WGT:    0.00   1.00




**********************************************************************
*                                                                    *
*     CORRECTED OVERALL FIGURE OF MERIT OF PHASING:   0.48           *
*     BIAS RATIO:  1.42                                              *
*                                                                      *
*     Bias ratio is given by /(FOMA*FOMB),           *
*     where phiA and FOMA are phase and  from prior (model or     *
*     experiment) and phiB and FOMB are from map.                      *
*     Bias ratio is about 1 if phiA and phiB are independent,          *
*                        >1 if phiB is biased by phiA,                 *
*                        <1 if FOMA or FOMB are overestimated          *
*                                                                      *
*     If BIAS RATIO <1, CORRECTED FOM = ESTIMATED FOM * BIAS RATIO   *
**********************************************************************


Cumulative phase change from start to end of this cycle []

DMIN  -----acentric--------  --------centric-------  ---------all---------
           N        N        N
 0.0    0.25    0.36  11352.   0.25    0.40   1138.   0.25    0.36  12490.
 6.0    0.47    0.50    431.   0.39    0.48    130.   0.45    0.50    561.
 3.8    0.43    0.48   1473.   0.27    0.39    222.   0.41    0.47   1695.
 3.0    0.29    0.34   1892.   0.30    0.43    197.   0.29    0.35   2089.
 2.6    0.23    0.32   1932.   0.16    0.36    179.   0.22    0.32   2111.
 2.3    0.18    0.29   3459.   0.22    0.34    262.   0.18    0.29   3721.
 2.1    0.15    0.23   2165.   0.15    0.43    148.   0.15    0.25   2313.

             All done .. writing out new phases

 (Q)QOPEN allocated #  1
User:   terwill              Logical Name: resolve_ps.mtz
Status: UNKNOWN    Filename: resolve_ps.mtz
HEADER INFORMATION FOR OUTPUT MTZ FILE ON INDEX  3

* Title:



* Number of Columns =  10

* Number of Reflections =  12490

* Missing value set to NaN in output mtz  file

* Column Labels :

H K L FP PHIM FOMM HLAM HLBM HLCM HLDM

* Column Types :

H H H F P W A A A A

* Cell Dimensions :

  113.95  113.95   32.47   90.00   90.00   90.00
```

```
*  Resolution Range :

     0.00262     0.22672      (   19.542 -     2.100 A )

* There is no sort order recorded in the MTZ header

* Space group = I4  (number    79)



**********************************************************************
*                                                                    *
*     NOTE:  HIGH RESOLUTION LIMIT ADJUSTED TO   2.10 A              *
*   (Use the keyword resolution to set a different value)            *
*                                                                    *
**********************************************************************
```

Back to table of contents

RESOLVE FAQ page

Where can I find all the questions and answers that have been sent to the SOLVE/RESOLVE newsgroup?"

- *You can find them archived at Jiansheng Jiang's very nice site at http://asdp.bnl.gov. On that site, click the blue button "BBnML" (Bulletin Boards and Mailing Lists) and then click "solve" under the Archive column.*

What do I do if RESOLVE says:

- *Sorry, the protein region has no volume?* This means that the solvent content is 0 or 1. Try a value in between.
- *Sorry, this is an invalid access code* You need to check out your solve2.access file. If SOLVE runs, so should RESOLVE.
- *Segmentation fault (core dump, program never starts)* There can be several causes. On linux and SGI this can happen if your system does not match the system the binaries were compiled on. Solution: email terwilliger@lanl.gov and I'll send you instructions on how to compile on your own machine. Also on an SGI: This can happen if your machine doesn't have enough memory allocated to you. To get around this (at least on an SGI), use the command: "unlimit" . You can put this as one line in your .cshrc file if you use csh. The same works on an Alpha too. You may also need to increase your swap space in some cases.
- *Segmentation fault (core dump after "Writing peaks to.."), same fix as above: use the command: "unlimit"; may need to increase your swap space or use "small" version.*
- *killed This happens if you run the too big a version of resolve on an SGI or if you're already running a big job. See the above command "unlimit" to fix it or use a smaller version.*
- */sbin/loader: Fatal Error: Program datasize exceeds process datasize limit. This is an error message you get with an Alpha when you don't have enough memory allocated to yourself. To get around this, edit your ".cshrc" or similar file to include the line "unlimit" or use the "small" version.*
- *Sorry, nothing in bins of density? The electron density map you input probably has no reflections in the resolution range or is all zeros.*
- *Sorry, cannot add any more onto the "LABIN" line (max 132 characters). RESOLVE cannot use more than 132 characters in a labin line. Sorry!*
- *Sorry: uninterpretable value or keyword. RESOLVE could not parse the input. It stops when this happens. Usually a keyword it does not recognize was input. Note that RESOLVE does not recognize partial keywords, you have to type the whole word in.*
- *Sorry, need to make n_map_max at least [some number]. The version of RESOLVE you have is*

not big enough for your dataset. Try the "giant" (resolve_giant) or even the "huge" (resolve_huge) version. If that does not work, contact the author for a bigger version.

- *sorry, this map has no density. The electron density map you input probably has no reflections in the resolution range or is all zeros.*
- *Sorry, need to increase n_refl_max The version of RESOLVE you have is not big enough for your dataset. Try the giant or huge versions (resolve_giant or resolve_huge). If that does not work, contact the author for a bigger version.*
- *rms of this map is nan avg = nan This can happen if there are "NaN" (empty) entries in your mtz file. This version of RESOLVE does not always handle these correctly. If you remove them (mtzdump, remove lines with question marks, f2mtz) it should be ok.*
- *resolve: Open failed: File: SYMOP. RESOLVE needs the CCP4 symmetry data file "symop.lib". Usually this is located in /usr/local/lib/solve after you download resolve. If it is not there, then you'll need to find out where it is, then define the environmental variable SYMOP. Similarly, SYMINFO needs to be defined.*

  ```
  setenv SYMOP /usr/local/lib/solve/symop.lib
  setenv SYMINFO/usr/local/lib/solve/syminfo.lib
  ```

  [or wherever they are].
- *Sorry, cannot read segment library needed for automatic model-building. Or Sorry, cannot properly read fragment libraries .. RESOLVE model-building uses a library of helix/strand segments and a library of fragments. These are normally located in /usr/local/lib/solve/segments/ after standard installation of SOLVE/RESOLVE. If you define $SOLVEDIR then RESOLVE will expect to find these libraries in $SOLVEDIR/segments.*

Can RESOLVE..

- *Extend phases or fill in missing reflections? Yes, any reflections with non-zero F in your input file that are in your resolution limits are filled in by RESOLVE (version 1.04 or higher). Additionally with version 2.01 or higher, all reflections that are completely missing are filled in as well if you specify "fill"*
- *Use NC-symmetry? Yes, starting with version 2.0.*
- *Phase from a partial model? Yes, use iterative model-building for this one*
- *Use a MR starting model? Yes, use prime-and-switch phasing or use iterative model rebuilding*
- *Use a mask that I input? Yes, sort of. Make a model that covers the region you want masked. Read the model in with model mask_model.pdb. Then specify use_model_mask and that will be used. You can set the radius around each atom to be masked with rad_mask xxx*

Parameters to consider setting if SOLVE does not find a satisfactory solution

If SOLVE does not find a good solution atomatically, there are a few parameters that you might consider setting to tell it how to go about its search. Some of these are:

- THOROUGH. This keyword tells SOLVE to keep looking through all the seeds even if a good solution (figure of merit greater than 0.5, score greater than 10) has been seen. The default is QUICK, the opposite of THOROUGH.
- RATMIN. This keyword specifies the minimum F/sigma that will be used. The default is 2.0. SOLVE may be tossing too much of your data if it is weak.
- RESOLUTION. This keyword tells SOLVE the resolution limits to use. If they include high resolution but the data is very poor there, this could prevent SOLVE from finding a solution at high resolution.
- RESOLUTION_STEPS. This keyword (default = 1) tells SOLVE to cycle through shells of resolution. Sometimes SOLVE can find a solution at low resolution but not high resolution or vice-versa. Try a value of 3 if 1 does not work.
- CUTOFF_DERIV. This keyword can be used to give SOLVE more restrictive high- and low-resolution limits for a particular derivative, wavelength, or native dataset than the overall resolution limits specified with RESOLUTION.
- RES_PHASE. This keyword can be used to set a different high-resolution limit for phasing in SOLVE than the overall resolution limit.
- SN_MIN and SN_RATIO_MIN. These keywords can be used to automatically set a different high-resolution limit for phasing in SOLVE than the overall resolution limit. Default is signal-to-noise minimum of 0.1, minimum ratio to low-resolution signal-to-noise of 0.1.
- FP_OR_FM. This keyword tells SOLVE to keep reflections in MAD datasets where only F+ or only F- have been measured. If MAD data are very incomplete, SOLVE may be tossing too much data in its effort to only use matched F+/F- pairs. The default is FPFM_ONLY, only use matched pairs.
- FIXSCATTFACTORS vs REFSCATTFACTORS. These keywords tell SOLVE whether to fix or refine scattering factors f' and f". If your MAD data is pretty weak, refining scattering factors can lead to worse estimates than you started with.
- NANOMALOUS and NRES. These define the number of anomalously scattering atoms and protein residue equivalents in the asymmetric unit. NRES defines the overall scale of the data. The ratio of NANOMALOUS to NRES defines how big the expected scattering contribution from the anomalously scattering atoms should be. Additionally NANOMALOUS defines the maximum number of sites for the anomalously-scattering atoms (unless the keyword NSOLSITE is set, in which case the NSOLSITE value is used).
- NSOLSITE and NSOLSITE_DERIV. These keywords define the maximum number of heavy atom sites either for the dataset as a whole (NSOLSITE) or for a particular derivative

(NSOLSITE_DERIV). You can use them to control how much time SOLVE will spend in various derivatives.
- INANO. This keyword tells SOLVE to use anomalous differences for a derivative. If you leave it off it will not include them for MIR data. (For MAD data SOLVE will include anomalous differences unless you tell it not to with noinano).
- NOANOREFINE. This keyword tells SOLVE that even if INANO is specified for a particular derivative, the anomalous differences will not be used in heavy atom refinement. This is the default for MIR data and should be used for MIR or MAD data if the anomalous data are much weaker than isomorphous differences.
- ACCEPTANCE. This keyword tells solve how weak a site can be but still get accepted. Sites that have occupancies of about ACCEPTANCE times the average for other sites are typically accepted (but this is not a simple cutoff, it is a parameter in the overall scoring procedure). If you are not getting as many sites as you would like, try lowering the value of ACCEPTANCE (default =0.2).
- NSEEDSOLVE. This keyword tells solve how many seeds to try (default=5). If SOLVE isn't finding anything, you could try more.
- NTOPSOLVE. This keyword tells SOLVE how many solutions to keep track of at any one time. If SOLVE isn't finding anything, you could increase it (default=5). Note that this will slow things down a lot if you increase it by much.
- ICRMAX. This keyword tells solve how many peaks in the Patterson function to try as potential cross-vectors in a 2-site search. Try increasing it from 30 to a higher number.

```
#!/bin/ksh
#
#  bayesdiff -- does Bayesian differencing on two CNS output files
#              and writes the output to a third CNS input file.
#              The CNS output files should be written in the order
#                 FOBS SIGMA FCALC TEST
#              where FCALC is complex (default for CNS).
#              The output of this script (and the input to the next
#              round of CNS) will be named bayesdiff.hkl unless you
#              specify otherwise.  The output defines the following
#              columns of "mutant" data that you should read in to CNS
#              and use:
#                 FOBS SIGMA WEIGHT TEST
#              A log of the SOLVE run called bayesdiff.log will also
#              be created.
#
#  Usage: bayesdiff ../wt/wildtype.hkl  ../mut/mutant.hkl [output.hkl]
#
#  Author:  JB  25 Sep 99
#

#
# Check arguments
#
if [[ $# < 2 ]]; then
  echo "Usage: bayesdiff ../wt/wildtype.hkl  ../mut/mutant.hkl [output.hkl]"
  exit 0
fi
if [ -r $1 ] ; then
   wtfile=$1
else
   echo "Can't read wild-type input file $1"
   exit 0
fi
if [ -r $2 ] ; then
   intfile=$2
else
   echo "Can't read mutant input file $2"
   echo "Can't read mutant input file $2"
   exit 0
fi
outfile=${3:-bayesdiff.hkl}
```

```
#
# strip header lines at top and get rid of non-numeric parts of
# CNS file before reading it in
#
nskip=7 # number of lines to skip at the top of the file
tail +$nskip $wtfile | tr -d '[=][:alpha:]'| tr -s '[:blank:]' '[:blank:]'  > wtin.dat
tail +$nskip $intfile | tr -d '[=][:alpha:]'| tr -s '[:blank:]' '[:blank:]'  > intin.dat

#
# file up SOLVE, setting it to overwrite any output files that may exist
# from previous rounds
#
CCP4_OPEN=unknown; /usr/local/bin/solve <<EOF

logfile bayesdiff.log
SYMFILE /usr/local/lib/solve/p63.sym
CELL 61.08  61.08  110.40  90.0   90.0  120.0
RESOLUTION 20.0 2.25

IMPORT
wtin.dat
2,
wt.drg
conversion of ground-state data
5
Fobs
Sigma
Fcalc
Phi
Rtest
1.0,
y
n
y
1
4

IMPORT
intin.dat
2,
int.drg
conversion of intermediate data from X-PLOR
```

```
5
Fobs
Sigma
Fcalc
Phi
Rtest
1.0,
y
n
y
1
4

FILEMERGE
2
wt.drg
int.drg
combined.drg
ground state and intermediate data merged into same file
1
1,3
2
1,3
2
5,5
0


infile combined.drg
outfile tmpout.dat
ncolfowt 1
ncolswt 2
ncolfc 3
ncolfomut 4
ncolsmut 5
ncolfcmut 6
ncolrtest 7
fdiff
EOF

#
# fix up header of output file for CNS
#
```

```
set -A nreflection `wc -l tmpout.dat`
echo " NREFlection=    $nreflection">$outfile
cat <<EOF >>$outfile
 ANOMalous=FALSe { equiv. to HERMitian=TRUE}
 DECLare NAME=FOBS                   DOMAin=RECIprocal   TYPE=REAL END
 DECLare NAME=SIGMA                  DOMAin=RECIprocal   TYPE=REAL END
 DECLare NAME=WEIGHT                 DOMAin=RECIprocal   TYPE=REAL END
 DECLare NAME=TEST                   DOMAin=RECIprocal   TYPE=INTE END
EOF
cat tmpout.dat >>$outfile
rm -f wt.dat *.drg tmpout.dat intin.dat wtin.dat


#  END OF bayesdiff.ksh
```