EPMR: Molecular replacement by evolutionary search

User's Guide for openEPMR, Version 11.02

Last updated: February 8, 2011

Contents

Introduction

EPMR is a general-purpose molecular replacement program. Unlike most molecular replacement programs, it does not divide the problem into separate rotation and translation searches. Instead, it uses an evolutionary search algorithm to simultaneously optimize the orientation and position of a search model (1,2). The program operates as follows:

- An initial set of random solutions (random orientations and positions for the search model) is generated.
- The correlation coefficient is calculated for each trial solution.
- A fraction of the highest scoring solutions are retained and used to regenerate a complete set of new trial solutions. This is done by applying random alterations to the orientation angles and translations for each "surviving" solution.
- The correlation coefficients for the new population are calculated, the population is again regenerated from the top scoring solutions, and this procedure is repeated for a specified number of cycles.

The algorithm provides broad, stochastic, initial sampling of the search space while gradually focusing in on the most promising regions. It allows for efficient searching of the six-dimensional (or higher) space. In general, it is several orders of magnitude faster than a brute-force, systematic, 6-D search. At the end of the evolutionary optimization, a local minimization is performed on the best solution. This is simply a rigid-body refinement of the search molecule.

The program calculates structure factors rapidly by indexing into a molecular transform using the method of Huber and Schneider. A traditional structure factor calculation is done only once - for the search model set at the origin of a P1 cell. Subsequent structure factor calculations are done by transforming reflection indices according to the rotations

and translations applied to the model and the relationship between the P1 and real cells, interpolating into the grid of P1 structure factors and summing over the symmetry operators of the crystal. This is much faster than an FFT calculation. A simple Babinet-type solvent correction is applied to the calculated structure factors. The values of the solvent correction parameters (k, B) are optimized during the search.

Because of the stochastic nature of the evolutionary optimization process, the correct solution will not be obtained on every run, even with a very good search model. The success rate is dependent on the quality of the model (2). By default, 20 optimization attempts are done, and more may be required if you have a difficult problem. For search models that are poor and at the limit of detection, the search efficiency can be quite low. If you have a molecular replacement problem that has not yielded a solution by any other means, a reasonable last resort is to set up EPMR to do as many runs as your patience and computing resources will allow. As long as the true solution represents the global maximum in the correlation coefficient between Fo and Fc, even if by the slimmest of margins, the algorithm will eventually find it.

EPMR includes the following features:

- the ability to automatically search for multiple copies of a molecule in the asymmetric unit, either sequentially or concurrently
- the ability to search with multiple models, either sequentially or simultaneously (i.e., in competition with each other)
- the ability to use multiple coordinate sets as parts of an "assembly" that comprises the complete search model
- an option to search over all related space groups, either sequentially or simultaneously
- rotation-only and translation-only search modes
- an option to provide static, partial structure
- independent optimization of each segment of a search model during the final rigid-body refinement step
- an option to bypass the evolutionary search and do only local, rigid-body optimization of the model

This version of EPMR is free, open-source software, distributed under the terms of the GNU General Public License. New versions of the program will be released on a regular basis. The latest version of EPMR is always available for download from http://www.epmr.info.

Feedback is welcome. E-mail concerning the program can be sent to Chuck Kissinger at ckissinger at epmr . info.

If you publish results obtained using EPMR, please cite Charles R. Kissinger, Daniel K. Gehlhaar & David B. Fogel, "Rapid automated molecular replacement by evolutionary search", Acta Crystallographica, D55, 484-491 (1999).

Usage

The program requires either two or three input files on the command line depending upon whether a CCP4 mtz file is used for both cell constants and reflection data. The first file specified on the command line is used as the source of information on the unit cell and space group. The file can be either an mtz file or a text file specifying the cell constants and space group number in the order:

```
a b c alpha beta gamma space_group_number
```

These are free-format and can be divided between any numbers of lines. An example of appropriate contents for the text file is:

```
40.76 18.49 22.33 90 90.61 90 4
```

All 230 space groups are available in their standard settings. For rhombohedral space groups, the hexagonal setting is expected, and your data needs to be indexed accordingly.

The second file on the command line should be a standard PDB format file containing the search model. Any lines in this input file that are not ATOM or HETATM records are ignored. The program expects correctly formatted PDB ATOM records. The names of atoms with two-letter element symbols (e.g., Fe, Ca, Se) must be appropriately left-shifted to be correctly recognized. If the PDB file contains multiple segment identifiers in columns 73 to 76, these will be used to subdivide the search model during the final rigid-body refinement. Each segment is optimized independently during that step.

If the first file on the command line is an mtz file, and you also want to use it as the source of your reflection data, it is not necessary to provide a third file on the command line. The program will read the Fobs data from any data column in the mtz file labeled "F", "FP" or "FNAT" or from the first column with a label beginning with "F_" or "FP_". If you are not using an mtz file, the third input file is a text file that contains the observed structure factors. The only requirement is that the file has H, K, L, and Fobs as the first four items on each line, separated by spaces.

The command line:

```
epmr example.cell example.pdb example.hkl
```

or:

```
epmr example.mtz example.pdb
```

will run the program in its default mode. In this mode, the program will search for a single copy of the search molecule in the asymmetric unit. It will run the evolutionary search procedure up to twenty times, or until a solution with a correlation coefficient of 0.65 is obtained. Data in the resolution range between 1000 and 3 angstroms will be used in the search. The top solution found will be written to a file called "epmr.best.pdb".

If you wish to provide multiple search models, put the list of file names in a file, and supply that file to the program in place of a coordinate file, with a "@" symbol before it:

```
epmr example.mtz @example.filelist
```

A single search model can be read from the standard input by putting a "-" (dash) in place of the coordinate file name on the command line:

```
epmr example.cell – example.hkl <example.pdb >example.log
```

When the program is started, it will print some information about the input data and program settings, do the initial FFT structure factor calculation, and then start the optimization runs. At the end of each evolutionary search, a local, rigid-body refinement is performed on the result. The final orientation for each run will be reported on a line that begins with "Solution", followed by the run number, rotation (alpha, beta, gamma as defined for the CCP4 programs), translation (in fractional coordinates for the model after it has been centered at the origin) and then the space group, model number, correlation coefficient and R factor. Theta1, theta2, and theta3 in the CNS convention are printed on a subsequent line. (If you intend to make use of the rotation and translation values outside of the program, they must be applied to the search model after it has been centered at the origin). The angular and translational relationships to the best previous solution obtained are also printed. You can use the Linux/UNIX command:

```
grep Solution epmr.log
```

to print just the solutions. On Linux systems, the command:

```
grep Solution epmr.log | sort –n -k 11r,11 -k 12
```

will list the solutions sorted by correlation coefficient, then R-factor.

Options

The operation of the program is controlled by a set of command line options. (If the program is run without any command line arguments, a brief summary of the available options will be listed.) The possible options are:

-A Treat models as an **assembly**, search with all simultaneously

This option is off by default. It causes each input model to be treated as an independently positioned and oriented segment of the complete search model. (By default, each input model is treated as an entirely separate, alternative search model.) When this option is turned on, the program will attempt to optimize the orientation and position of all search models simultaneously.

This is an experimental option and is not yet recommended for general use.

-a Treat models as an **assembly**, search with each sequentially

This option is off by default. It causes each input model to be treated as an independently positioned and oriented segment of the complete search model. (By default, each input model is treated as an entirely separate, alternative search model.) When this option is turned on, the program will find the optimal solution for the first model, keep that as static structure, search for the optimal solution for the next model, store that as static structure, and so on. Both this option and the "-A" option can be combined with "-m" or "-M" to search for multiple copies of an assembly.

-b number The minimum "**bump**" distance - the smallest unpenalized distance between the center of mass of a solution and that of any symmetry mates

The default value is 0.0 (no packing restrictions). This is applied to all trial solutions that are generated during the course of the search. When a solution violates this minimum distance, the correlation coefficient calculated for that solution is scaled down by the ratio of the shortest observed distance over the minimum allowed distance. In the case of searches for multiple molecules in the asymmetric unit, this also sets the minimum distance between a solution and any previously found solutions. (This applies both to previous solutions found during the run and to partial structure entered with the -s option. See the description of the -s option below for instructions on entering multiple fragments of partial structure for use with this option.)

This option imposes a simple penalty on solutions that pack poorly. It can be helpful in some searches, but decreases the efficiency of others, particularly if a large value is used. This option appears to be less useful in single molecule searches than in searches for multiple molecules in the asymmetric unit, but is worth a try if you are having trouble finding a solution that packs well. It is up to you to decide what an appropriate minimum intermolecular distance should be. (Remember that it is the distance between centers of mass.) It is best to be conservative - the program will not search efficiently without some room to move solutions around through positions that pack poorly.

-C Search simultaneously over all space groups in the same point group and **crystal** class as the input space group

This option is off by default. The space group is treated as an additional variable in the evolutionary search (i.e., the alternative space groups compete with each other during the evolution). This option works well for routine molecular replacement calculations with a good search model. For difficult cases, it is safer to use the "–c" (lowercase) option instead, which will search each of the candidate space groups separately.

-c Search sequentially over all space groups in the same point group and **crystal** class as the input space group

This option is off by default. Each space group is tried in turn. The number of optimization runs specified by the "–n" option will be completed for each space group choice, and all input models will be tried in each space group before moving to the next.

-e integer Set the **seed** value for the random number generator to a specific value.

By default (or if the seed value is set to a value of zero using this option), the seed is generated from the system clock at run time. This option is not necessary for normal operation of the program, but can be useful for testing purposes. Two separate runs of the program that use the same seed value will produce identical results.

-g integer The number of "**generations**" (cycles of optimization).

The default value is 50. It is not normally necessary (or recommended) to change this value. However, setting this value to zero (-g0) will bypass the evolutionary search and feed your input model directly to the local optimizer. This allows you to use EPMR as a convenient rigid-body refinement program.

-h number **High**-resolution limit for diffraction data used in the search (in angstroms)

The default value is 3.0 Å. It can be useful to try different high-resolution limits, but the search efficiency is generally higher if data to at least 3.0 are included. It is rarely effective to set this to a value higher than 5.0.

-l number **Low**-resolution limit for diffraction data (angstroms).

The default value is 1000.0 Å (effectively no cutoff). MR calculations can be highly sensitive to the inclusion or exclusion of the lowest resolution data, and also to the accuracy of that data. In some cases, poorly measured low-resolution data can negatively impact the search. If you are having problems, you could try excluding low-resolution data using this option. A low-resolution limit of 15 angstroms is often effective.

-M integer The number of copies of the **molecule** in the asymmetric unit to find simultaneously

The default value is 1. Values greater than one cause multiple orientations and positions to be optimized for the search model.

This option has not yet been optimized in this new version of the program and is not yet recommended for general use.

-m integer The number of copies of the **molecule** in the asymmetric unit to find sequentially

The default value is 1. A value of 2 would cause the program to search for one copy of the molecule, save the solution as partial structure and continue searching for a second solution.

-n integer The **number** of independent optimization attempts

The default value is 20, which is intended for moderately difficult cases. For some very difficult MR problems, values of 100 or more are worth trying. The program will stop before the completion of the number of runs specified here if a solution is obtained that has a correlation coefficient that exceeds a specified threshold (flag –t, below).

-o name The file name prefix for the **output** coordinate files

The default is “epmr”. If you run multiple jobs in the same directory, you will have to use this flag to avoid writing over files from other runs. If you specify “-” (a dash) as the file name prefix, the coordinates will be written to the standard output of the program instead of a separate file.

If you specify the option “-w2”, the program will name the output files for each run as “prefix”.”run_number”.pdb (e.g., epmr.1.pdb). If you are searching sequentially over multiple space groups, search models, and/or

copies in the asymmetric unit, the file name will have additional numeric indicators for those (e.g., epmr.3.4.2.5.pdb would indicate the third space group choice, fourth model, second copy, fifth optimization attempt). The coordinate file containing the top solution from all runs with a specific combination of space group, model and copy-number is named with "best" replacing the run number.

-p integer — The **population** size (number of trial solutions evaluated in each cycle of optimization)

The default value is 300. Increasing this value beyond 300 will increase the search efficiency, but you will get more benefit from increasing the number of optimization attempts (-n) instead.

-R — **Rotation** search only

-S — Try all **search** models simultaneously

This option is off by default. The choice of the model becomes an additional variable in the evolutionary search. This is an experimental option still under development.

-s filename — Read **static** structure from the specified file

If you have partial structure to input, include this flag and follow it with the name of the PDB file containing the correctly positioned partial structure. You can separate the partial structure into as many files as you wish and use this flag multiple times on the command line. It is only necessary to divide the partial structure up this way, however, if you are using the -b flag (see above) and are inputting multiple "pieces" of partial structure (e.g., multiple monomers). The minimum packing distance calculation will treat the partial structure within each separate file as a separate fragment.

-T — **Translation** search only

This option is off by default. It will cause the program to search only translation space, keeping the orientation of the search model unchanged. This could be useful, for instance, when you have a search model that has been pre-oriented by another program or through knowledge of non-crystallographic symmetry. Note that the orientation WILL be optimized during the final rigid-body optimization after the evolutionary search, so the orientation is likely to change slightly and could change significantly during this step.

-t number — The **threshold** value of the correlation coefficient that indicates an acceptable solution, which will stop the program

The default value is 0.65. The program will stop when a solution correlation coefficient exceeds this threshold; if you want the program to continue for a specified number of runs no matter what, set this value to 1.0. It is best to keep this value relatively high to avoid having the program stop on an incompletely converged solution. Unlike previous

versions of EPMR, this value is fixed throughout the search and not adjusted during sequential searches for multiple copies of a molecule.

-v integer — The **verbosity**, which controls the amount of information written to the standard output of the program

The default value is 1. A value of 0 (zero) results in nothing being written (unless the coordinates are written to the standard output using the option, "-o –"). A value of 2 causes more detailed information on the progress of the evolutionary search to be written.

-w integer — The quantity of solutions you want to **write** out to PDB files

The default value is 1. A value of 0 (zero) here means no coordinates will be written out. A value of 1 means only the top solution from all of the runs will be written out. A value of 2 means all solutions will be written out. (The -o flag controls the name of the output PDB files.)

Acknowledgements

The original EPMR program was written by Chuck Kissinger and Dan Gehlhaar at Agouron Pharmaceuticals. David Fogel (Natural Selection, Inc.) contributed numerous ideas that were essential to the original development of the program. Bradley Smith (Pfizer) was responsible for rewriting and improving the original program and testing numerous alternative algorithms and ideas. SGX Pharmaceuticals and Anadys Pharmaceuticals also supported the development of this open-source version of EPMR.

References

1) Kissinger, CR, Gehlhaar, DK & Fogel, DB, "Rapid automated molecular replacement by evolutionary search", Acta Crystallographica, D55, 484-491 (1999).

2) Kissinger, CR, Smith, BA, Gehlhaar, DK & Bouzida, D, "Molecular replacement by evolutionary search", Acta Crystallographica, D57, 1474-1479 (2001).

3) Huber, R. & Schneider, M., "A group refinement procedure in protein crystallography using Fourier transforms", J. Appl. Cryst. 18, 165-169 (1985).

Examples

Example 1

Search for one molecule in the asymmetric unit, perform up to 10 attempts of the evolutionary search procedure (or until a cc above 0.65 is obtained), and write out the best solution to the file "epmr.best.pdb":

```
epmr example.cell example.pdb example.hkl > example.log
```

Example 2

Do a translation search, use static partial structure and do up to 50 runs for each molecule:

```
epmr -T –s example_A.pdb –n 50 example.cell
example_B.pdb example.hkl > example.log
```

Example 3

Search for one molecule in the asymmetric unit (default), use data from 80 to 3.5 Å, do up to twenty runs, write out only the top solution (default) to a file with the prefix "example_solution", and use static structure:

```
epmr –l 80 –h 3.5 -s example_partial.pdb -o
example_solution -n 20 example.cell example.pdb
example.hkl > example.log
```

Example 4

Search for three identical molecules in the asymmetric unit, write out all solutions, up to 10 runs (default) for each molecule. Input two fragments of partial structure. Penalize solutions that pack within 15.0 Å (center-to-center distance) of any symmetry mates or any partial structure that was input or generated during the run:

```
epmr –m3 -w2 -s example_partial1.pdb -s
example_partial2.pdb -b15.0 example.cell example.pdb
example.hkl > example.log
```

Example 5

Use EPMR in a program pipeline. The search model is read from the output of the mythical program "search_model_generator" and the solutions are written to the input of "solution_evaluator". Cell and reflection data are read from an mtz file. Search all related space groups sequentially, search sequentially for two identical molecules in the asymmetric unit, write out all solutions, and do up to 100 optimization runs for each molecule:

```
search_model_generator | epmr —o- —c -m2 -w2 —n 100
example.mtz - | solution_evaluator
```