# EM3DR.DOC

(last update August 24, 2004)

## CONTENTS



## A. INTRODUCTION

EM3DR is a non-parallelized three-dimensional (3D) image reconstruction program. It produces a 3D map from a set of particle images. The program first extracts the data for normal equations from each particle view, then combines the normal matrices and solves them to give big G's, which are temporarily or permanently stored in a disk file (DEFAULT = BG.DAT), then computes little g's from the big G's, and finally computes a Fourier Bessel transformation of the little g's to produce the 3D density map of a particle with defined symmetry (typically icosahedral, but several cyclic and dihedral symmetries are also now supported) in the "standard" 2-fold setting (see EMICO_SYS at web site: http://bilbo.bio.purdue.edu/~baker/programs/programs.html)). Since January 1998, 3D map data have been stored in the Purdue *.PIF format.

EM3DR is normally run after and in combination with EMPFT or EMICOGRAD.

In *THEORY*, EM3DR has no restrictions as to number of particles combined or the resolution limit. Of course, the memory capacity of the computer and disk space available WILL pose ultimate limits if you go overboard.

**Latest changes (Jun 2002):** Option for different CTF correction filters.

## B. PROGRAM INPUT

Em3dr now require a keyword organized input. The input is designed as follows:

```
keyword = value
```

where the "keyword" field denotes the data attribute you would like to set and the "value" field the value to which you would like to set it. For example to set the BIN_FACTOR you would like to use in em3dr to "2" you would enter the following line:

```
BIN_FACTOR = 2
```

Normally one would place all the keyword input in an input file and then pipe that into the program all at one time. For example, if your input file is named "input.dat" you might run as follows:

```
% em3dr < input.dat > em3dr.log &
```

with em3dr.log containing all of the program output during the run.

The END_OF_KEYS keyword signals the program to stop gathering keyword input and continue by first gathering parameter filenames.

> NOTE: The minimum required input to identify the keyword is indicated in CAPITAL LETTERS. For example, the keyword "VERbose" will be recognized if only "VER" appears in the input stream. All other characters are then superfluous.

Input parameters control the operation of EM3DR as follows:

THRee3dr_mode
- = 0 Entire program is run to compute a 3D map from images
- = 1 Stop after calculating BG'S (stored in BG.DAT)
- = 2 Start with existing BG.DAT data and compute 3DR

SYMmetry = 1 if no symmetry
2-31 if n-fold about z-axis
532 if icosahedral
The following dihedral point group symmetries are allowed:
222,32,422,52,622,72,822,92

BIN_factor
- = 1 Binfactor of 1 (DEFAULT; i.e. no data compression)
- = 2 Binfactor of 2 (compresses 3D data eightfold and 2D data fourfold) The current limit on BIN is 2.

**NOTE:** BIN=2 is mainly designed to be used in conjunction with EMPFT when it is used with BIN=2. When BIN is set = 2,

then IMG_MP_SCALEF defaults to 0.5, regardless of any other
value you may enter for IMG_MP_SCALEF.

VERbose
= 0 gives minimal output to terminal or *.LOG file
= 1 gives extra output
= 2 generates lots of output!!!

BG_Out = 1 BG data saved in binary file 'BG.DAT'

LG_Out = 1 LG data saved in binary file 'LG.DAT'

MP_Out = 0 - output whole 3D map in standard 2-fold view
= 1 - use only odd set of images to compute MAP
= 2 - use only even set of images to compute MAP
= 3 – output both even and odd maps
= 5 - output quadrant along 5-fold view

When MP_Out = 1 or = 2 then only EVERY OTHER image in the total input set of images is used for the calculation of the 3D MAP. Note that "odd" or "even" does NOT mean that images are selected according to particle ID, but instead just means that the particle images are selected according to the SEQUENCE in which they are read from the PARAM file(s). For example, if your FIRST PARAM file only lists particles with *even-numbered* IDs (e.g. 22, 34, 36, 58, 100, 268, 982, 1046, etc.) and MP_Out is set = 1, then only the images with ID #s 22, 36, 100, 982, etc. will be used. Also, if in this last example, the PARAM file was *instead* the 2nd PARAM file listed in the EM3DR batch job *AND* the first PARAM file specified an *ODD number* of images, then only images with ID #s 34, 58, 268, 1046, etc. would be selected from this PARAM file. This strategy assures that, no matter how many PARAM files there are and no matter what the particle IDs are, the number of images selected will always be equal to or within one of being half of the total number of images listed in the PARAM files.

When MP_Out = 5, only a top (near) quadrant of the 3D density map in the five-fold view is computed and stored in a disk file. Choosing this option may be very useful for producing preliminary maps where you just need a quick look at a partial map to see if it is correct.

Future enhancements to EM3DR may include options to write out even smaller portions of the standard 3D map (e.g. just a quadrant) which will generate greater disk storage savings and perhaps enhancements in other routines.

2. RESOLUTION_HI, RESOLUTION_OUT, UNITS_OUT, MAP_Pizsiz, MAP_DIM, ICO_SPREAD

(2F,I,F,I,F)
---

RESOLUTION_Hi specifies the highest possible resolution to which the final 3D map can be computed. RES_HI (and RES_OUT) are expressed in units defined by the user (see description of UNITS, below). RES_HI can't be set to a value any SMALLER than twice the size of the IMAGE pixels (the so-called Nyquist limit imposed because the Fourier transform of digitized data only extends to a spatial frequency of 1.0/(2*pixel size). If you believe your IMAGE data are good enough to achieve higher resolution, you need to rescan the IMAGE at smaller pixel resolution. The typical rule of thumb is to digitize your IMAGE data at a pixel resolution that is 3 or 4 times the expected resolution, in which case only information in the Fourier transform out to a radius (IRAD) of FFT_DIM/3 or FFT_DIM/4 is used. If, for example, you expect your IMAGE RESOLUTION to be about 2.0 nm, then you would normally digitize at 0.7 nm intervals (or smaller).

**NOTE:** The program prints a warning and then STOPS if you attempt to enter a value for RES_HI that is too small (i.e. too high a resolution).

RESOLUTION_Out is the resolution to which the final 3D map is computed. RES_OUT must be equal or larger than RES_HI. Normally RES_OUT is set equal to RES_HI (the DEFAULT), but sometimes it is useful to calculate several reconstructed density maps at different resolutions. This can be done efficiently by calculating and saving the BG data (MODE = 1) with RES_HI set to the maximum resolution desired and then running EM3DR (MODE = 2) to reuse the already calculated BG data, but setting RES_OUT to different values (larger than RES_HI) for each new map you calculate.

UNIts_out specifies the units assigned to the variables RES_HI and RES_OUT and also to the PIXSIZ variable obtained from each of the PARAM files (see description of PARAM files below: input line #7). The user may define UNITS according to the following table, but REMEMBER - make sure you are CONSISTENT and use the same UNITS to specify ALL these variables.

0 = pixels (i.e. dimensionless)
1 = Angstroms (A)
2 = nanometers (nm)

MAP_Pixsiz is the size of pixels in the final 3D map and also (if MAP_dim is left = 0) will automatically resize the dimensions of the 3D map. Only positive floating point numbers are valid for MAP_Pixsiz. Using negative values for MAP_Pixsiz results in MAP_Pixsiz being set to 0.0. The DEFAULT (0.0) will make a 3D map with pixels identical to the LARGEST pixelsize in the input IMAGE data (called PIXSIZ_MAX) and the map will have NCOL**3, dimensions,

where NCOL is the LARGEST linear dimension of the input IMAGE data. Set MAP_Pixsiz smaller than PIXSIZ_MAX if you wish to produce a larger than normal reconstruction, one that is sampled with smaller size MAP pixels to achieve much nicer 3D rendering effects for making high quality figures for slides or publication. For example, if MAP_Pixsiz = PIXSIZ_MAX/2.0, the final 3D map will have dimensions (2*NCOL+1). Under these conditions EM3DR will take considerably longer than normal to finish and may 'consume' very large amounts of disk space to store the finely sampled output map. Typical, final reconstructions used to generate nice surface-shaded renderings (e.g. with ROBEM) are computed with MAP_Pixsiz = PIXSIZ_MAX/2.0. If you wish to get a map quickly and don't mind coarse MAP pixel resolution, set MAP_Pixsiz > PIXSIZ_MAX and the 3D map will be correspondingly smaller and take up much less disk space (and be rendered much faster, etc.).

**NOTES:** if BIN=2, then MAP_Pixsiz is set EXACTLY EQUAL to 2.0*PIXSIZ_MAX because BIN=2 is treated as a special case for use of the 3D MAP in EMPFT.

It may be best to use RobEM to render MAPs with smaller size pixels rather than creating MAPs with EM3DR and smaller pixels because this saves disk space and I/O overhead in programs.

MAP_Dim can be used to specifically fix the dimensions of the 3D map and thereby override the effects of any change in map dimensions brought about by using MAP_Pixsiz. Though not likely to be used too often, MAP_DIM adds flexibility to how you create output 3D map data. Setting this parameter can be useful, for example if you are trying to produce a 3D model of the right dimensions for use in EMPFT, which currently requires that the NCOL and NROW dimensions of the 3D model EXACTLY match the NCOL and NROW dimensions of the input IMAGE data. MAP_DIM may also be used to keep the dimensions of the 3D output map constant, while at the same time allowing the pixel size of the 3D map data to be magnified or demagnified according to the value of Map_pixsiz. For example, you may choose to calculate a 3D map with, say Map_pixsiz = PIXSIZ_MAX/0.75, but with MAP_DIM set equal to the size the 3D map WOULD have been IF MAG_Pixsiz were left = 0.0. This would give a 3D map in which the reconstructed particle would appear shrunk by 25% but still inside a 3D box of dimensions equal to the dimensions of the original input IMAGE data.

**WARNING:** If you decide to use MAP_DIM, beware of possible "side effects". For example if you set it too small depending on what MAP_Pixsiz is set to, the map may be cut off beyond a radius that lies within the particle! Soooooooooo, unless you really need to use this option, leave MAG_DIM = 0 and let the program size the 3D output map according to the value of MAP_Pixsiz and the input IMAGE dimensions.

ICO_SPREAD is a seldom used variable that sets the condition on the mean inverse eigenvalue. This is usually (99.99% of the time) set equal to 1.0 (The DEFAULT). Set to a lower value for a less stringent test.

3. CTF_Mode, CTF_Filter_mode, TEMperature_factor, FILter_factor_1 (2I,11F)

---

These parameters are used to affect CTF corrections of the images. The following is a simple description of the nature of the use of these variables. More information will be supplied as we become more adept at using CTF corrections with real data.

CTF_Mode is a switch that signifies the type of CTF correction to make.

| CTFMode | Action taken |
|---|---|
| 0 | No CTF correction made to the FFT data. |
| 1 | Full CTF correction (flip phases and modify amplitudes) |
| 2 | Only flip phases in the FFTs |
| 3 | Only modify amplitudes in the FFTs |
| 4 | Multiply image by CTF (WARNING) blah blah |

FILTER options and formulas

TEMperature_factor is a variable that specifies an estimate of the temperature factor for the IMAGE data. Values in the range of 200 to 500 Angstroms square are typical, however, it is recommended that you NOT enter a value for TEMperature_factor until you are near the end of refining your 3D density map. Hence, TEMPERATURE_FACTOR should normally be set = 0.0. **TSB needs to add more detailed description (and formula???)**

FILT_FAC is the FACTOR TO MAKE SURE NOISE IS NOT MAGNIFIED TOO MUCH NEAR THE NODES OF THE CTF. THE FORMULA USED depends on the value of CTF_FILTER_MODE:

| CTF_FILTER_MODE | FORMULA |
|---|---|
| 1 | CTF_FILTER_MODE 4 (before first peak)+ CTF_FILTER_MODE 2 (after first peak)<br>FFTnew = (FFTold*TEMPERATURE_FACTOR)/(ABS(CTF) + FILTER_FACTOR_1*(1-ABS(CTF)) (Before first peak)<br>FFTnew = (FFTold*ABS(CTF)*TEMPERATURE_FACTOR)/(CTF**2 + FILTER_FACTOR_2*(1-ABS(CTF)) (After first peak) |
| 2 | FFTnew = (FFTold*ABS(CTF)*TEMPERATURE_FACTOR)/(CTF**2 + FILTER_FACTOR_1*(1-ABS(CTF)) |
| 3 | FFTnew = (FFTold*ABS(CTF)*TEMPERATURE_FACTOR)/(CTF**2 + FILTER_FACTOR_1) |
| 4 | FFTnew = (FFTold*TEMPERATURE_FACTOR)/(ABS(CTF) + FILTER_FACTOR_1*(1-ABS(CTF)) |
| 5 | FFTnew = (FFTold*TEMPERATURE_FACTOR)/(ABS(CTF) + FILTER_FACTOR_1) |

FILTer_factor_1 and FILTer_factor_2 are typically set to 0.1 (DEFAULT). Use values less than 0.1 at your OWN RISK. Values greater than 1.0 are also not recommended and are reset on input to default values!

4. BG_Filename - File containing BG data (A; DEFAULT = BG.DAT)

Enter the name of the permanent file in which the big G data are stored (MODE=2). This input line is read but ignored if MODE.LE.1.

5. MAP_Filename - 3D MAP output filename (A; DEFAULT = ICOS2F.PIF)

Specify the name of the PIF format file to store the 3D map. This input line is read but ignored if MODE=1.

6. TITLE - Header for 3D MAP (80A1)

The header is text you can use to identify the 3D map data. This input line is read but ignored if MODE=1.

7. Input PARAMETER filename(s) (A FORMAT)

Enter the names of up to 999 different PARAM files, each one on a new input line. A full description of the contents of PARAM files is given in the documentation for the program EMPFT (see web site: http://bilbo.bio.purdue.edu/~baker/programs/programs.html). The first line of each PARAM file should list the name of the IMAGE file which contains byte-packed raw image data stored in *.PIF format. The second line in each PARAM file contains the values of:

PIXSIZ, UNITS, VOLTAGE, AMPFAC, DELF_MAJ, DELF_MIN, ANG_MAJ, and CS_COEFF. The remaining lines in each PARAM file contain the current set of ID, THETA, PHI, OMEGA, X, Y, MAG_FAC, PFTCC, PRJCC, and CMPCC values for specific particles in the raw image file. Values for PFTCC, PRJCC, and CMPCC if present, are ignored by EM3DR. MAG_FAC defaults to 1.0 if no value or a 0.0 is supplied.

ID is the image number--the storage position of the IMAGE data for byte-packed format data.

THETA,PHI,OMEGA are the current values of the three orientation angles (in degrees) for each particle image (obtained from previous runs of EMPFT or EMICOGRAD or EMICOFV).

FFT_ORIGX,FFT_ORIGY are the pixel coordinates of the particle center (the point 0.0,0.0 corresponding to the lower left corner of the boxed particle image). These values are estimated or refined by EMPFT.

MAG_FAC is a radial scale factor computed in the EMPFT program. It specifies the size of each particle image relative to some standard, such as a 3D reconstruction model. Hence, a MAG_FAC = 1.02 means that the particle image is 2% larger than the model. EM3DR uses this value to assure that all particle transforms are sampled so the data are all combined at a uniform magnification. With images of frozen-hydrated particles boxed from a single micrograph, MAG_FAC = 1.0 (DEFAULT) is normally assumed for all particles.

**NOTE:** Input line 7 and following are ignored if MODE=2.

## C. PROGRAM EXECUTION

Normal operation of EM3DR leads to a 3D that map contains the entire reconstructed particle viewed down a 2-fold axis in the standard orientation: for a particle with icosahedral (532) symmetry, this is where three mutually perpendicular two-fold particle axes are aligned with a Cartesian (XYZ) map coordinate system (NCOL columns in the X direction; NROW rows in the Y direction; NSEC sections in the Z direction). For particles with other symmetry, such as with Cn (cyclic) or Dn (dihedral) point group symmetry, the n-fold axis coincides with the Z-axis.

**NOTE:** On Linux this program is generally run in BATCH mode. The following is an example Linux batch run command file.

## D. EXAMPLE EM3DR BATCH JOB

Create a file like this (Let's call it "em3dr.bch" for this example):

```
!       COMMAND FILE FOR RUNNING EM3DR
!
!       THE INPUT ARE AS FOLLOWS:
!
!    1. MODE, SYM, BIN, IPRINT, BG_OUT, LG_OUT, MP_OUT (7I)
!       MODE   = 0: run entire program
!              = 1: stop after calculating bg's
!              = 2: start with existing BG.DAT data
!       SYM    = 1    if no symmetry (like a ribosome)
!              = 2-31 if n-fold cyclic symmetry (about z-axis)
!              = 532  if icosahedral symmetry
!              = 222  if 222 dihedral symmetry
!              = 32   if  32 dihedral symmetry
!              = 422  if 422 dihedral symmetry
!              = 52   if  52 dihedral symmetry
!              = 622  if 622 dihedral symmetry
!              = 72   if  72 dihedral symmetry
!              = 822  if 822 dihedral symmetry
!              = 92   if  92 dihedral symmetry
!       BIN    = 1: no data compression (DEFAULT)
!              = 2: compresses 3D data eightfold and 2D data
!                   fourfold). The current limit on BIN is 2.
!       IPRINT = 1: extra output to terminal or *.LOG file
!              = 2: even more output to terminal or *.LOG file
!       BG_OUT = 1: BG.DAT saved (otherwise not saved)
!       LG_OUT = 1: LG.DAT saved (otherwise not saved)
!       MP_OUT = 0: output whole 3D map in standard 2F view
!              = 1: use only odd-order images to compute MAP
!              = 2: use only even-order images to compute MAP
!              = 5: output top quadrant in 5-fold view
!
!    2. RES_HI, RES_OUT, UNITS, IMG_MP_SCALEF, MAP_DIM, ICO_SPREAD
!           (2F,I,F,I,F)
!    3. CTFMODE, CTF_FILTER_MODE, TEMPERATURE_FACTOR, FILT_FAC(10)
(2I,11F)
!    4. BG data input/output filename (A) (ignored if MODE = 0)
!    5. 3D map output filename (A; DEFAULT = ICOS2F.MAP)
!    6. Header for 3D map (80A1)
!    7. Parameter filename(s) (A)
!
1, 532, 1, 0, 0, 0, 0
35., 35., 1, 1.0, 0, 1.0
0, 1, 0.0, 0.1, 0.1
HSV_BG.DAT
HSV.PIFMAP
Equine herpes intermediates #1856 n=60 3DR 99**3 (35A resolution)
HSV.DAT
```

Now change your default directory to the appropriate:
% cd ~tsb/v/hsv/test

Run the program and generate log file "em3dr.log":
% em3dr < em3dr.bch > em3dr.log &

## E. PROGRAM NOTES

Several parameters are listed at the terminal (or in the *.LOG file):

BOX_DIAM = Original particle diameter (in densitometer pixels). This also equals 2.0 times the reciprocal of the annuli spacing in the MATBG routine (=2.0/FFT_STEPSIZE).

FFT_STEP_MIN = minimum width of each transform annulus, in transform pixel units (TPU), given by:

$$\frac{\text{FFT\_DIM}}{\text{BOX\_DIAM}*2.0} \quad \text{TPU}$$

MAX_BESSEL_ORDER = The highest Bessel order used in the computation of the layerplane data as computed by:

$$(\text{NBES}-1)*\text{Z\_SYM}$$

LG_NANNULI = Cut-off radius (in ANNULI) in transform space. Usually equals MAX_NANNULI (set in MATBG part of program), but may be made smaller to reduce resolution. LG_NANNULI is computed as:

$$\text{NINT}\left(\frac{\text{FLOAT(FFT\_DIM)}}{(\text{FFT\_STEP\_MIN}*\text{PIX\_RES})}\right)$$

NANN_BG_MAX = Largest number of annuli needed from a transform, as calculated by:

$$\text{NINT}\left(\frac{\text{FLOAT(FFT\_DIM)}}{(\text{FFT\_STEP\_MIN}*\text{PIX\_RES})}\right)$$

The default for NANN_MAX_BG is FFT_DIM/4.0.

NBES = The number of Bessel orders needed to achieve a resolution at LG_NANNULI in the Fourier transform, as given by:

$$1 + \mathrm{NINT}\left(\frac{(\mathrm{PI}*\mathrm{LG\_NANNULI}*\mathrm{BIGR\_STEP}*\mathrm{FLOAT}(\mathrm{BOX\_DIAM}))}{(\mathrm{FLOAT}(\mathrm{Z\_SYM}))}\right)$$

NCOL = Dimension of 3D MAP (# of pixels/row). Also equal to NROW (# of pixels/column) and NSEC (# sections in the MAP).

PIX_RES = Pixel resolution with units specified by UNITS parameter. This is the smallest possible pixel resolution as computed by:

$$\frac{\mathrm{RES\_OUT}}{\mathrm{PIXSIZ\_MAX}}$$

PIXSIZ_MAX = Maximum size pixel of the input images as specified in the PARAM file(s).

RADIUS = Number of radial steps (pixels) in real-space for the reconstruction. The particle diameter = (2*RADIUS + 1).

RES_HI = Maximum resolution (in UNITS) for calculations of the big G's, as specified by the user.

RES_OUT = Reconstruction resolution (in UNITS) as specified by the user. Must be ≥ RES_HI because the reconstruc-tion can't be computed at a resolution any higher than the calculation of the big G's.

STEP_SIZE = Size of each radial step (i.e size of each pixel in the reconstruction relative to the original pixel size in the digitized particle image).

$$\frac{1.0}{\mathrm{IMG\_MP\_SCALEF}}$$

## F. NOTES ON USE OF EM3DR FOR ICOSAHEDRAL PARTICLES

For ICOSAHEDRAL particles, EM3DR first computes one half of the density map, oriented with a five-fold axis coincident with the Z axis and one, perpendicular 2-fold axis coincident with the Y direction, from the little g's. Little g's are only computed for Bessel orders that are multiples of 5. The density map is calculated a section at a time in unique $2\pi/5$ (= 72°) sectors. Each sector is subdivided in cylindrical coordinates (RHO(r,$\phi$))

since the little g's are computed in polar form. The polar section is interpolated to produce a Cartesian section (MAP(x,y)).

After all sections are computed (the number of them = RADIUS+1), the half-particle 3D map is rotated so that the old Y direction (2-fold) becomes the new Z direction and a 3D map of the entire particle is temporarily stored. The map in this orientation has two other mutually perpendicular 2-fold axes in the XY plane of the central section, but not coincident with the X and Y directions. A final transformation rotates the particle into the "standard" orientation so the three mutually perpendicular 2-fold axes coincide with the X, Y, and Z directions (ROWS,COLUMNS,SECTIONS) of the 3D map.

The program produces a 3D map with full 532 symmetry.

## G. CREDITS

The original code (some still mostly intact!) for performing the Fourier Bessel reconstruction of icosahedral particles was written by R. Crowther and L. Amos (MRC Laboratory, Cambridge, England). Many modifications have been made since then (includes contributions from EMBL lab in Heidelberg and NIH lab in Bethesda) and the list of references that follow provide more details about the procedures:


Crowther, R. A., L. A. Amos, J. T. Finch, D. J. DeRosier and A. Klug (1970) Three dimensional reconstructions of spherical viruses by Fourier synthesis from electron micrographs. *Nature* **226:421-425.**

Crowther, R. A. (1971) Procedures for three-dimensional reconstruction of spherical viruses by Fourier synthesis from electron micrographs. *Phil. Trans. R. Soc. Lond.* **261:221-230.**

Fuller, S. D. (1987) The T=4 envelope of sindbis virus is organized by interactions with a complementary T=3 capsid. *Cell* **48:923-934.**

Baker, T. S., J. Drak and M. Bina (1989) The capsid of small papova viruses contains 72 pentameric capsomeres: direct evidence from cryo-electron-microscopy of simian virus 40. *Biophys. J.* **55:243-253.**

Fuller, S. D., S. J. Butcher, R.H.Cheng and T. S. Baker (1996) Three-dimensional reconstruction of icosahedral particles - the uncommon line. *J. Struct. Biol.* **116:48-55.**

Baker, T. S., N. H. Olson, and S. D. Fuller (1999) Adding the third dimension to virus life cycles: Three-Dimensional reconstruction of icosahedral viruses from cryo-electron micrographs. *Microbiol. Molec. Biol. Reviews* **63:862-922.**

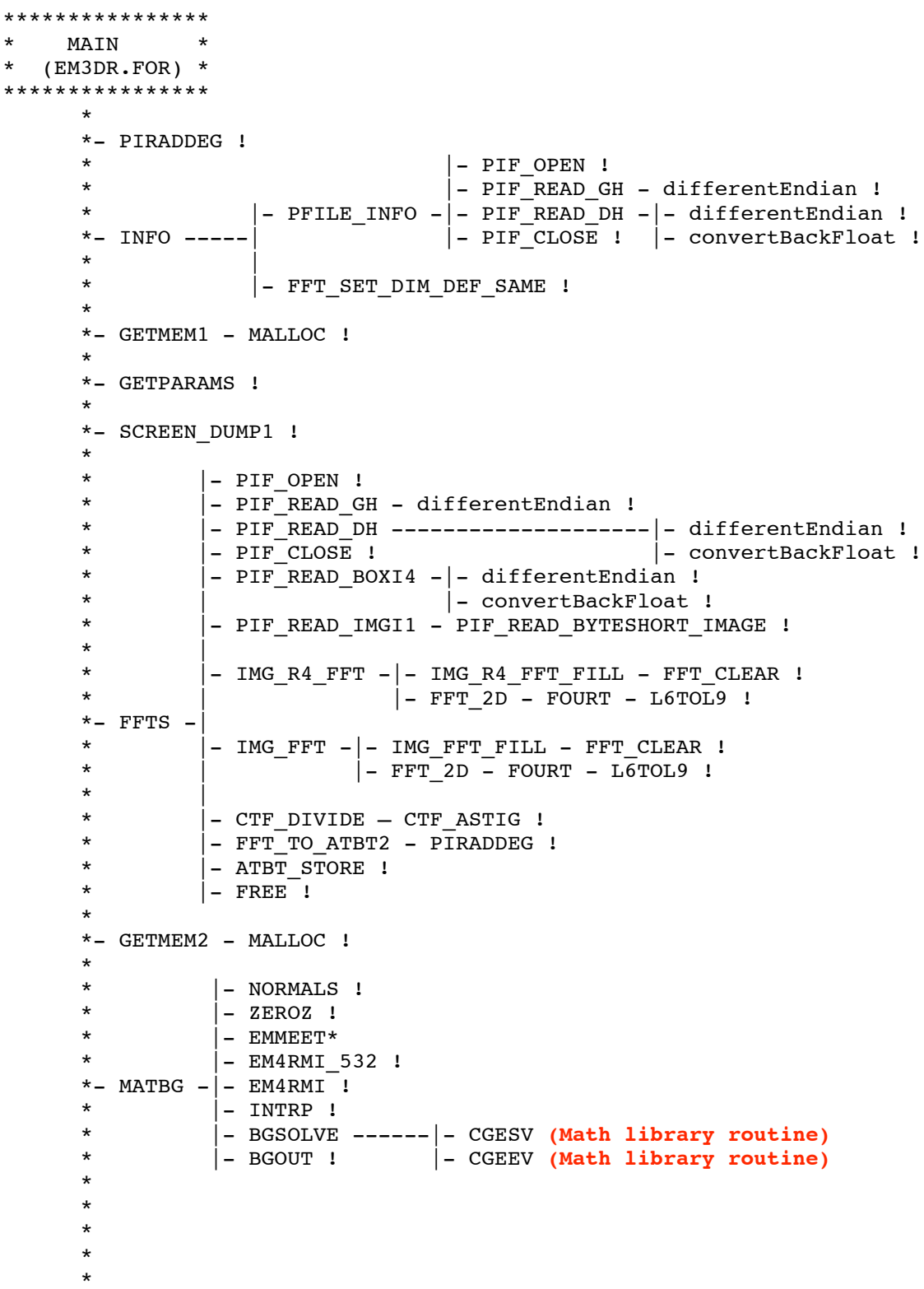

## H. FINAL NOTES

1. Use EMPRJ to project the 3D map in any view orientation as specified in an unedited or edited (to 'fix' orientations as desired) PARAM file.

2. EMPFTPRJ can be used to compute a uniform set of views within the asymmetric unit of the 3D map.

# I. FLOW CHART FOR EM3DR PROGRAM

** means “see below”; # means end of program; ! signifies no subroutine calls



```
      *
      *           |- NORMALS !
      *           |- ZEROZ !
      *           |- EMMEET*
      *           |- EM4RMI_532 !
      *- MATBG2 -|- EM4RMI !
      *           |- INTRP !
      *           |- NMAT -|- NMATA !            |- ICO_LOCATE !
      *           |        |- FREE !             |- ICO_TRIDI !
      *           |- BGSOLVE2 -|- ICO_HOW ------|- ICO_EIGVAL !
      *           |            |- ICO_BGPROD !  |- ICO_EIGVEC !
      *           |- BGOUT2 !                   |- ICO_RELOCT !
      *
      *- FREEMEM1 - FREE !
      *
      *- GETNRECS !
      *
      *- GETMEM3 - MALLOC !
      *
      *- FILLBES -|- BSL0 !
      *           |- BSL1 !
      *
      *- BGS_TO_LGS !
      *
      *- SCREEN_DUMP2 !
      *
      *- GETMEM4 - MALLOC !
      *
      *            |- PIF_INIT_HEAD !
      *            |- PIF_OPEN !
      *- FB_NFOLD -|- PIF_WRITE_GH - differentEndian !
      *            |- PIF_WRITE_DH -|- differentEndian !
      *            |                |- convertBackFloat !
      *            |- PIF_WRITE_SECI2 - PIF_WRITE_MAP_SECTION**
      *            |- PIF_CLOSE !
      *
      *                             |- PIF_INIT_HEAD !
      *           |- MRC2F !         |- PIF_OPEN !
      *           |                  |- PIF_WRITE_GH - differentEndian !
      *- MAP532 -|- XYZ2F ---------|- PIF_WRITE_DH -|- differentEndian !
      #           |                  |                |- convertBackFloat !
                  |- APPLY_3F**      |
                                     |- PIF_WRITE_SECI2 - PIF_WRITE_MAP_SECTION**
                                     |- PIF_CLOSE !

=============================================================================
|                 |- GENROT — DMATMUL !
|                 |- PIF_READ_WMAPI2 - PIF_READ_MAPI2**
|                 |- VECMUL !
|- APPLY_3F -------|- GET_EQUIVALENT_VALUES — INTERP3D !
|                 |- FILL_PERP2 !
|                 |- PIF_WRITE_SECI2 - PIF_WRITE_MAP_SECTION**
|                 |- PIF_WRITE_DH -|- differentEndian !
|                                  |- convertBackFloat !
=============================================================================
|
|- PIF_WRITE_MAP_SECTION - PIF_WRITE_MAP_FLOAT_IMAGE - differentEndian!
|
```

```
================================================================================
|
|- PIF_READ_MAPI2 - PIF_READ_MAP_SHORT_IMAGE - differentEndian !
|
================================================================================
```