EMPFTPRJ.DOC

(last update Aug 4, 1999)

CONTENTS

A. INTRODUCTION

EMPFTPRJ calculates a database of projection and Polar Fourier transform data for one half of the appropriate asymmetric unit. These data can be displayed by ROBEM or used as input to the EMPFT program (when the value of "MODE" in EMPFT is negative: see EMPFT.DOC). EMPFT computes exactly the same PFT and PRJ data in core memory but will NOT output these data for display with ROBEM or for re-use by EMPFT. Hence, EMPFTPRJ is useful if one wishes to reuse the PFT and PRJ model data over again.

EMPFTPRJ computes two types of data from a 3D model:

1. Projections (PRJs) and
2. Polar Fourier transforms (PFTs) of the projections.

These are used in EMPFT as reference data to determine THETA, PHI, OMEGA, X, and Y values for each 'raw' particle image. The high signal-to-noise ratio of the model density data is what makes the orientation and origin searches work well. For further details about the use of EMPFT, please consult the documentation DEXTRO3:[TSB.FOR.DOC]EMPFT.DOC.

NOTES

EMPFTPRJ currently ONLY works with cubic 3D MAP data (i.e. where NCOL=NROW=NSEC). Also, the program expects ALL particle images to have the same dimensions (NCOL x NROW) as in the 3D model.

EMPFTPRJ PROCEDURE

EMPFTPRJ converts each raw image from Cartesian to polar format by subdividing it into a series of annuli (each usually 1.0 pixels wide). Every annulus is sampled NROT times, with the interpolation performed at a fine enough interval to prevent loss of pixel resolution. Therefore, annuli corresponding to the

lowest radii in the particle image end up significantly over-sampled. Each annulus is then separately Fourier transformed to give what we call a polar Fourier transform (PFT) of the image.

The PFT gives the Fourier transform of the density distribution at each radius of the particle. This provides a particularly useful representation of the data which allows rapid and sensitive determination of the particle orientation.

EMPFTPRJ OUTPUT

EMPFTPRJ outputs two files: the model projections (Default name = PFT.PRJS) and the model PFTs (Default name = PFT.PFTS).

B. PROGRAM INPUT

1. BIN, SYM, DTHE, IRAD_LO, IRAD_HI (2I,F,2I)
2. 3D MAP filename (A)
3. PRJs output filename (A)
4. PFTs output filename (A)

1. BIN, SYM, DTHE, IRAD_LO, IRAD_HI (2I,F,2I)

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

SYM specifies the point group symmetry of the particle you are studying. The program searches one-half of the asymmetric unit. The following symmetries are available currently:

SYM = 1 for no symmetry
= 2-20 for n-fold cyclic symmetry (about z-axis)
= 532 for 532 icosahedral symmetry (DEFAULT)
= 222 for 222 dihedral symmetry
= 52 for 52 dihedral symmetry
= 622 for 622 dihedral symmetry
= 72 for 72 dihedral symmetry
= 822 for 822 dihedral symmetry

SYM determines the size of the asymmetric unit (hence, search window).

DANG is used to specify the step size in the θ and ϕ directions (in degrees: DEFAULT = 1.0). The step size in the θ direction remains constant (= DANG) but it varies in the ϕ from a smallest

value (= DANG) when θ=90° (at the 'equator') and *increases* thereafter for progressively smaller values of θ. Varying the ϕ step size assures uniform sampling of the ASU in regions where θ is <90°. The step size in the ϕ direction is given by the formula:

$$DANG/\sin\theta$$

IRAD_LO and IRAD_HI are variables used to define the annular portion of the projected data that is used in the PFT calculations. The number of annuli in each PFT (NANNULI = [IRAD_HI - IRAD_LO + 1). IRAD_LO is normally left = 1 and IRAD_HI is usually set to a value just beyond the particle boundary (but usually much smaller than NCOL/2 if you boxed the original particle images conservatively). Use IRAD_LO > 1 if you suspect that the projected data at higher radii (i.e. projection only of outer capsid features) will give a more sensitive measure of the orientation parameters (calculated in EMPFT).

IRAD_LO can't be set lower than 1 because the center of the projected view doesn't change with orientation and hence gives no useful information for the PFT calculations.

2. 3D MAP filename (A FORMAT)

Enter the name of the MAP file that contains the 3D model from which new projections and PFTs are generated.

NOTE: EMPFTPRJ currently only works if the dimension of the 3D MAP EXACTLY equals the image dimension.

3. PRJs output filename (A FORMAT)

Enter name of output file for storing model PRJs data (DEFAULT = PFT.PRJS). Here the data are stored as a pseudo 3D map with each 'section' being one of the NCOL by NROW dimension projection images. This file is written in PIF format so the data can be viewed with the ROBEM program.

4. PFTs output filename (A FORMAT)

Enter name of output file for storing model PFTs data (DEFAULT = PFT.PFTS). The data here are also stored as a pseudo 3D map with each 'section' being one of the NROT/4 by NANNULI dimension PFT 'images'. That is, there are NROT/4 values stored per row and NANNULI rows. Each row is one-half of the unique part of the

Fourier transform of a particular annulus. This file is written in PIF format so the data can be viewed with the ROBEM program.

C. EXAMPLE EMPFTPRJ BATCH JOB RUN WITH COMPRESSION (BIN=2)

```
$ SET DEF DEXTRO3:[TSB.HSV]
$ RUN DEXTRO3:[TSB.EXE]EMPFTPRJ
2, 532, 1.0, 1, 46
HSV.PIFMAP
HSV.PRJS
HSV.PFTS
$ EXIT
```

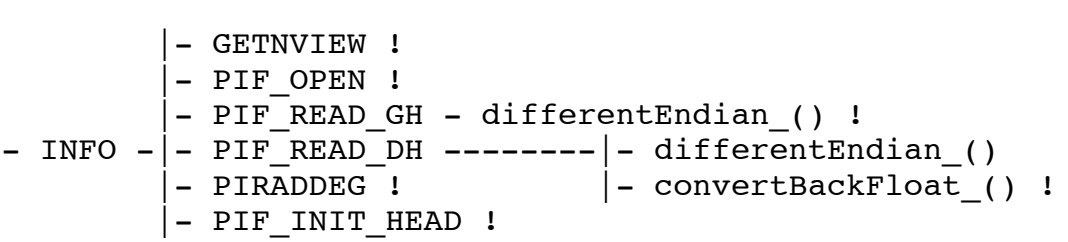
D. FINAL NOTES

1. DEXTRO3:[TSB.FOR]EMPFTPRJ.BCH is an example BATCH command file used to run EMPFTPRJ.

2. Use of BIN=2 is highly recommended, especially in the early stages of data analysis because tests show that EMPFTPRJ runs about 8 times FASTER than when run with BIN=1. Since many people densitometer their data at high pixel resolution (i.e. with a pixel size that is one third or smaller than the expected resolution limit of the data) BIN=2 MAY be useful for all but perhaps the final stages of data analysis (determination and refinement of particle orientations and centers).

3. If you are only interested in displaying the projection or PFT data, you may want to use BIN=1 to get finer sampling of these data. However, be forewarned that when BIN = 1, four times as much disk space will be used to store the data.

E. FLOW CHART FOR EMPFTPRJ PROGRAM



```
      *         |- PIF_WRITE_GH !
      *         |- PIF_WRITE_DH !
      *
      *- GETMEM - MALLOC !
      *
      *- PIF_READ_MAP_SHORT_IMAGE_() - differentEndian_() !
      *
      *- PIF_CLOSE_() !
      *
      *- PFT_CALC_TP !
      *                              |- PIRADDEG !
      *                              |- MAP_CLEAR !
      *             |- MAP_PRJ ----|- MAP_PRJ_XZ --- MAP_CLEAR !
      *             |              |- MAP_PRJ_AXIS - MAP_CLEAR !
      *             |              |- MAP_PRJ_ALL -- MAP_CLEAR !
      *             |
      *             |              |- PIF_WRITE_MAP_FLTINT_() -*
      *             |              |
      *- PRJSPFTS -|- PRJS_STORE -|- MAP_SYM -|- MAP_SYM_CAVG -
MAP_STATS -*
      *             |                          |- MAP_SYM_RAVG -|-
MAP_SYM_GRID
      *             |                          |- COPY_R4       |-
MAP_STATS -*
      *             |                          |- MAP_STATS -
MAP$STATS !
      *             |
      *             |- MAP_POLAR -|- MAP_CLEAR !
      *             |             |- MAP$POLAR - MAP_POLAR_GRID !
      *             |
      *             |- PMAP_FFT - FOURT - L6TO9 !
      *             |
      *             |- PFTS_STORE - PIF_WRITE_MAP_FLTINT_() -*
      *
      *- FREEMEM - FREE !
      *
      #

      |- PIF_WRITE_MAP_FLTINT_() - differentEndian_() !
```