LECTURE 25
CHARMM Course
11-Apr-2006

CHARMM

Chemistry at **HAR**vard **M**olecular **M**echanics

Twentyfifth Lecture:

Graphics

graphx.doc correl.doc

Topics

Exporting data for plotting or molecular graphics
- Time series data; plotting programs
- Extracting data from the output logs
- Coordinates

Molecular graphics with CHARMM
- X11 interactive, PostScript
- POV-Ray file export; using povray
- MPEG animations from trajectories

Rick Venable
rick_venable@nih.gov

Often, one is interested in exporting data in a format suitable for plotting. The time series commands described in **correl.doc** provide a useful basis for this, in a variety of ways. The most obvious is to write a single file per time series via *e.g.*

```
OPEN UNIT 2 WRITE CARD NAME prop.dat
WRITE prp UNIT 2 DUMB TIME
* dummy
*
```

After a review of syntax, we'll consider a more complex example.

Syntax for the CORREL command and subcommands:

```
        { ALL                    }
EDIT    { time-series-name       }  edit-spec
        { CORRelation-function   }

        edit-spec::=  [INDEx int] [VECCod int] [CLASs int] [SECOnd int]
                         [TOTAl int] [SKIP int] [DELTa real]
                         [VALUe real] [NAME new-name] [OFFSet real]
```

Editing a time series

The EDIT command allows the time series specifications to be modified directly.

WARNING:: This command gives the user direct access to most time series specification. There is NO checking to see if what is being done makes sense. As such, this command is versitile and dangerous.

The EDIT command must be followed by a valid time series name. All subsequent keywords will be based on that time series. The series name "ALL" will cause the edit spec to operate on all the time series. The name "CORR" will cause the edit to occur on the correlation function.

The following may be specified for a time series;

INDEx integer - May be specified to modify X,Y, or Z (1,2,3 resp) of a vector timeseries. Otherwise, all are modified. The index number is in fact an offset from the specified time series, where a value of 1 represents the selected time series. A value of 5 will cause the edit operation to modify the fourth time series from the specified.

CLASs integer - May be used to specify a class code (consult source).

TOTAl integer - The total number of valid steps may be altered, but none of the values are modified. By setting this value to zero, the time series is then ready again

for the next TRAJectory command.

SKIP integer - May be specified to reset the SKIP value. This may be useful after reading an external time series.

DELTa real - May be specified to modify the basic time step. The actual time step for a series is (SKIP*DELTA).

OFFSet real - The time of the first element in the time series.

VECCod integer - User may specify a vector code. This may be useful in merging 3 separate time series into a vector time series (or the reverse). In fact any number of time series may be grouped together with this option. For example, if a table with 5 time series is desired, setting VECCOD to 5 for the first one and the writing this time series will output all 5.

VALUe real - This allows the user to modify the series utility value. Its function depends on the Class code. This value is currently used for (USER, GYRAtion, DENSity, MODE, and TIME)

SECOndary int - The secondary class code may be modified (consult source).

The WRITe command

```
        { ALL                    }              { [FILE]          }
WRITe   { time-series-name       }  unit-spec   { CARD            }
        { CORRelation-function   }              { PLOT            }
                                                { DUMB [ TIME ] }
```

The WRITe command will write out time series or a correlation function. All of the write options expect a title to follow this command. There are several file formats; FILE (default), CARD, PLOT, and DUMB. The FILE and CARD options will write out all data regarding the specified time series with the expectation for later retrival by Charmm or another program. The PLOT option will create a BINARY file for plotting by PLT2. The first line of the title is used as the plot title, but this may be reset in PLT2.

The DUMB options will simply write out the values with no title or header to a card file, one value to a line. If the TIME option is specified, the time value will preceed the time series values (as needed for an X-Y plot). If the time series is a vector type, then all component values will be given on each line. Unless LONG (see miscom.doc) is in effect the output is limited to 8 values/line. DUMB option is useful for making plot files, or for feeding the data to other programs.

With the EDIT command, a user may merge 3 separate sequential time series into a vector time series (or the reverse). In fact any number of time series may be grouped together with this option. For example,

if a table with 5 time series is desired, setting VECCOD to 5 for the first one and the writing this time series will output all 5.

The following CHARMM input demonstrates this:

```
* DPPC C27R param, NPAT; P, carbonyl O atom Z, avg and per leaflet
*
bomlev -1

stream rtfprm.str
stream psfcrd.str

define leafa sele type P .and. prop Z .gt. 0.0 end
define leafb sele type P .and. prop Z .lt. 0.0 end
define colfa sele ( type O22 .or. type O32 ) .and. prop Z .gt. 0.0 end
define colfb sele ( type O22 .or. type O32 ) .and. prop Z .lt. 0.0 end
calc mxt = 1000 * @N

correl maxt @MXT maxs 16 maxa ?NATOM noupdate
enter zav zero
enter oav zero
enter lfa atom z sele leafa end
enter lfb atom z sele leafb end
enter coa atom z sele colfa end
enter cob atom z sele colfb end

set k 1
label ulp
calc u = @K + 10
open read unit @U file name /scratch/dyn@k.trj
incr k by 1
if k le @N goto ulp
traj firstu 11 nunit @N

mantim lfb abs
mantim zav copy lfa
mantim zav add  lfb
mantim zav mult 0.5
mantim cob abs
mantim oav copy coa
mantim oav add  cob
mantim oav mult 0.5

edit zav veccod 6
open unit 2 write card name lpdz.dat
write zav unit 2 dumb time
* dummy
*
end
stop
```

Over time I’ve developed a useful script for extracting dynamics property data from the output CHARMM log files; the

script is named ‘getprop.csh’ and is available for use or copy via ~rvenable/bin/getprop.csh on both Biowulf and LoBoS systems; a copy will be posted with the lecture notes in the course forum as well. It will prompt for arguments if they are not supplied, which is a useful way to get a list of the properties available. To extract a potential energy time series from compressed output files, the syntax is simply

```
getprop.csh ENER dyn?.out.gz dyn??.out.gz
```

Two files are produced, one from snapshot (NPRINT) data, ENER.DYNA.DAT, and another from the averages (IPRFRQ) and fluctuations, ENER.AVFL.DAT; the latter is the only source of data from every integration step.

Coordinate export; CHARMM .crd or .pdb format

From io.doc:

```
WRITe { { COORdinate coor-spec } [CARD]  }  UNIT unit-number
      {                                [PDB [MODEL int [FIRSt|LAST]]}

coor-spec:== [COMP]  [OFFS int] [IMAGes]  atom-selection
```

From corman.doc:

```
COORdinates { WRITe io-specification          } [COMP] [atom-selection]
                                                    [IMAGes]
```

The Rasmol and VMD programs can visualize both PDB format and CHARMM ‘card’ format coordinate files; VMD has additional CHARMM compatibility and can read PSF and coordinate trajectory files.

Note that one can include image atoms in the output coordinates, provided they are defined via one of the methods in CHARMM. Normally, only half the images are used for energy evaluation (from Newton’s third law), and of course only atoms within CUTIM of any “real” atom will be defined. To use all the image transformations for the output coords, an UPDATE IMALL is required; CUTIM could also be changed.

```
* extract simulation frame @F from file @N
*

stream rtfprm.str
```

```
stream psfcrd.str

open unit 21 file read name dyn@N.trj
read coor unit 21 file ifile @F
close unit 21

update imall cutim 14
open unit 12 write card name real_img.pdb
write coor pdb unit 12 images
* all real atoms plus image atoms within 14 A
*
stop
```

The above will write out the system coords surrounded by an additional 14 Å "halo" of image atoms. Since the additional atoms are **not** part of the PSF, they would be ignored when the original PSF is used to read the resulting coordinates. The above can be used to easily extract any arbitrary trajectory frame (coord set no. in a file) and export a PDB file. Usually the **images** keyword is not used or wanted.

The VMD and Rasmol programs are easily available, and fairly well documented; their use will not be described here.

CHARMM Element doc/graphx.doc 1.1

File: Graphx, Node: Top, Up: (chmdoc/commands.doc), Next:

GRAPHICS

Graphics is a subparser of charmm, invoked by via the GRAPH command. All of the miscellaneous commands (miscom.doc), coordinate commands (corman.doc), and internal coordinate commands (intcor.doc) are available from the GRAPHX> prompt. Only the 1st three characters are used for primary graphics commands, but many of the options require the 1st four characters.

The graphics facility has been extended to provide general X11 support, and the original Apollo GPR screen display has been dropped; a NODISPLAY version can also be built, which will generate all of the derived files. The other major enhancement is the production of PostScript output files, in either color or grayscale; both X11 and PostScript use the Apollo imaging model. Additional information on X11 usage tips and compiling for X11 are given at the end of this document. Finally, a recent addition is the production of input files for POV-Ray, an excellent freeware ray tracing package for making high quality molecular images. See http://www.povray.org

Option keywords are indicated by the use of upper case; lower

case terms are variable values, generally real numbers, but decimal points are not required. Triplets (x y z) are position dependent; omitted values are assumed to be zero. Items enclosed in square brackets are [optional] but their absence often implies a default choice. Default choices are indicated with an asterisk (*) in syntax listings where apropriate.

* Menu:

File: Graphx, Node: Summary, Up: Top, Previous: Top, Next: Description

GRAPHX COMMAND SUMMARY

Initializing, exiting:

```
GRAPhx  [XXSZ iwidth] [XYSZ iheight]  [ NVERtex integer ]
        [NOWIndow]

OFF                 :: disable all graphics and exit (see END, below).
END                 :: exit from command parser only; preserve window
RESet               :: revert to default view, scale
DEFault-colors      :: revert to default colors

HELp                :: provides a command listing
```

Available within the graphics subcommand parser:

```
miscellaneous-command-spec          ! see *note miscom:(chmdoc/miscom.doc).
IC        ic-subcommand-spec        ! see *note intcor:(chmdoc/intcor.doc).
COOR      coor-subcommand-spec      ! see *note corman:(chmdoc/corman.doc).
TRAJ      traj-subcommand-spec      ! see *note dynamc:(chmdoc/dynamc.doc).
```

These commands affect what is viewed:

```
DRAw [atom-selection]   :: change displayed atoms, force a redraw

DISplay [ON]* [MAIN]* [COMP] [BOND]* [VECT] [ATOM] [TEXT] -
        [OFF] [HBONds] [LABEls] [AXES]    :: change displayed objects

COLor color-name [brightfactor] [COMP] atom-selection
                 [ LEVel izcue-level ]

  color-name =   NONE   RED  BLACk   YELLow GREEn  WHITe
                 BLUE   CYAN MAGEnta GRAY   ORANge BROWn
                 PURPle TURQuoise    CHARtreuse  DKBLue
                    (or an integer from 0 to 15)
```

```
  izcue-level = an integer from 0 to 11; 0 == brightest, 11 == dimmest

HBStyle  [COLOR color-name]  [WIDTH iwidth]  [DASH idash]  :: HBOND style
LBL  label-type  label-atoms  SIZE label-size COLOr label-color
   label-type  = INIT SEGId RESN RESId TYPE ATNUm CHEM CHARge WEIGht
                 USER user-label
   label-atoms = FIRSt and/or atom-selection
   label-size  = VSMAll SMALl MEDIum LARGe          default: SMALL
   label-color = color-name (see COLOR above; default: YELLOW)
   user-label  = up to 8 characters

LINe iwidth         :: bonds or vectors; pixels
RADii [DEFaults] [PARam] atom-scale [bond-scale] atom-sel
AXEs  [XLIM xmin xmax] [YLIM ymin ymax] [ZLIM zmin zmax]
      [DEFAult]  (all limits set to -25, +25 A)
STEreo [ON/OFF] [dist] [angle]

TEXt [text-body]    :: set title text, null entry clears
FONT [ VSMAll | SMALl | MEDIum | LARGe ]  :: text font; default MEDIUM


     These commands change the view only:

SCAle  factor [MOL/LAB] [REP int]
BOXsize size  [MOL/LAB]
CENter [atom-selection]
MAXwindow
POInt  x y z
ROTate rx ry rz  [MOL/LAB] [REP int]
TRAnslate x y z  [MOL/LAB] [REP int]
ZCLip [low] high
ZCUe [ [low] high ]/[AUTO]


     These commands do not affect the display:

AUTo [ON*/OFF]      :: redraw after every command
ERAse [ON*/OFF]     :: screen clear prior to next drawing
EXEcute pathname    :: execute a program; no arguments are passed

     Output Files: the UNIT must be OPENed first...

PLUto UNIT n                                    :: PLUTO FDAT file
MAKE  UNIT n                                    :: LIGHT .atm file

PSC   UNIT n [COLOr] [BWREv] [PORTrait]         :: PostScript file
             [INIT]
             [TERM]

POV   UNIT n  [ INIT ]                          :: POV-Ray file
              [ TERM ]
              [ INCLude n ]  [ UOBJ n ]
              [ OBJect  obj-spec ]
```

(see the description for full syntax details)

File: Graphx, Node: Description, Up: Top, Previous: Summary, Next: Output

DETAILED COMMAND DESCRIPTION

GRAPhx [XXSZ iwidth] [XYSZ iheight] [NVERtex integer] [NOWIndow]

Invoked from the main CHARMM command parser; if already initialized (i.e. GRAPHX ... END) the previous graphic states are retained.

The XXSZ and XYSZ options set the X11 window width and height for the duration of the graphics session; a window resize is NOT passed back to graphics code at this time.

The NVERtex option increases the allocated storage for the vertex array used by the POV-Ray object code, especially for the MESH and RIBBON objects; the default is set to NATOM. but may need to be increased for scripts which create a lot of POV objects, especially RIBBON objects.

The NOWIndow option suppresses the X11 window for batch mode or remote usage; ideal for making PostScript files, etc.

HELp

List the available commands and syntax.

OFF

Disable all graphics and exit the graphics subcommand parser; see also the END command.

END

Exit from command parser only; allows other CHARMM commands to be performed w/o losing the current display. Re-invoking graphics does not re-initialize the graphics settings. Useful for trajectory movies using MERGE DRAW, e.g.

```
end
open unit 12 read file name dyn.trj.10
merge draw firstu 12 nunit 1
graphics
```

Overlays can be created (on screen only) by preceding the END command in the above example with ERAse OFF.

RESet

Restores view settings to the program defaults (scale, translation and rotations).

```
DISplay [ON*]  [MAIN*] [COMP] [VECT] [ATOM] [BOND*] [TEXT] [HBONds] -
        [OFF]  [LABEls] [AXES]
```

Turns the display of various graphic features on or off; the default is DISPLAY ON MAIN BOND TEXT which will show the connectivity of the atoms in main coordinate set, using the default CHARMM title. The options are:

ON*/OFF	enable (default) or disable the display of a feature
MAIN*	the main coordinate set; displayed by default
COMP	the comparison coordinate set; both may be displayed
BOND*	atom connectivity as atom-colored half-bonds; default
ATOM	filled circles using current radius (see RADII)
TEXT*	title display
HBONDS	current HBOND list, using double width lines
LABELS	residue names, atom types, user labels ...
AXES	lab frame axes; + end solid, labeled; - end dashed

Examples:

```
display atom on          ! enable atom display; MAIN assumed
display text             ! toggles title display
display hbonds off       ! disable H-bond display
```

DRAw [atom-selection]

Forces a redraw when AUTO mode is off; also used to to change which set of atoms is currently being displayed. All display modes and output files from PLOT, PSC, PLUTO, XMOLE, and MAKE commands use this atom selection. The initial selection is all atoms and may be restored via:

```
draw sele all end
```

```
COLor color-name [brightfactor] [COMP] atom-selection
                 [LEVel i]
```

Sets the color of individual atoms according to the atom

selection, using one of the color names below:

```
      NONE   RED  BLACk   YELLow GREEn  WHITe
      BLUE   CYAN MAGEnta GRAY   ORANge BROWn
      PURPle TURQuoise    CHARtreuse  DKBLue
```

The color applies to the main coordinate set unless COMP is specified; brightfactor is a relative intensity, 0.0 -- 1.0 Alternatively, the zcue levels may be set directly to one of the indices, an integer in the range 0:11 with 0 being fully bright.

Setting all atoms to WHITE, and using the brightfactor gives the best control over how PostScript grayscale output will look on the printed page; note that NONE is background, BLACK is the basic carbon color is really gray, and WHITE will be black in PostScript grayscale output or in color output with black/white reversal.

Examples:

All carbons for segment s are colored cyan:

```
color cyan sele type c* .and. segid s end
```

Colored based on weighting array:

```
color green lev 11 sele prop wmain .gt. 1.0 end
color green lev 10 sele prop wmain .gt. 2.0 end
               :
             etc.
```

DEFault-colors

Restore the default color assignments, based on element type.

Example:

```
default
```

LINe iwidth (bonds or vectors; pixels)

Set the line width for bonds & vectors, in pixels (integer).

Example:

```
line 2
```

RADii [DEFaults] [PARam] scale [bond] atom-sel

Sets the radius for displaying atoms, and for output files produced by the PLOT, PSC, XMOLE, and MAKE commands. The options are:

scale	required if no other options are specified, and assumed to be 1.0 if omitted; performs a relative scale if used by itself, or scales the radii set by the DEFAULT or PARAM options
DEFAULTS	set radii to a convenient size for display, based on atom type (C, N, O, ...)
PARAM	use VDW radii
bond	value for bond radii for LIGHT program (see MAKE)
atom-sel	atom selection to apply the radii command to

Examples:

```
rad 0.8                       ! reduces radii to 80% of current value
radii param .5                ! set radii to 50% of VDW
radii param .5  .15           ! bond radii to 0.15 A
rad 1.5 sele type H* end      ! enlarge all H atoms by 50%
```

```
LBL  label-type  label-atoms  SIZE label-size  COLOR label-color
   label-type  = INIT SEGID RESN* RESID* TYPE CHEM CHARGE WEIGHT USER user-
label
   label-atoms = FIRST* and/or atom-selection
   label-size  = VSMALL SMALL* MEDIUM LARGE
   label-color = color-name ( see COLOR command above; default: YELLOW )
   user-label  = up to 8 characters
```

The LBL command identifies which atoms are to be labeled, what atom attributes are to be included in the label, and the relative size of the labels; the defaults are marked with an asterisk [*].

The INIT option clears all labels, and any other options are ignored.

One or more the following attributes may be included in the label, by simply including the keyword(s) in the LBL statement:

SEGId	segment name (from GENErate; A4)
RESN	residue name (from the RTF; A4)
RESId	residue ID, a numeral (A4)
TYPE	atom type, e.g. N, CA, CB ... (A4)
CHEM	atom parameter type code (A4)
CHARge	atomic charge (G12.4)
WEIGht	value stored in the weight vector (G12.4)
USER	arbitrary user-specified text (A8)

The label length (24 bytes) is such that all attributes may NOT be displayed simulaneously; in particular, CHARge and WEIGht may not be

displayed at the same time for the same atom.

SIZE is specified by one the keywords VSMALL, SMALL, MEDIUM, LARGE, with a default of SMALL. The COLOR keyword allows setting the label color, using the same color names as the COLor command; the default label color is yellow. Each use of the LBL command can create a group of labels of a different size and color, for atoms which don't overlap with any previous label atom selections.

The blank delimited word following the keyword USER, up to 8 characters, allows the use of any text string as a label for the selected atoms.

The default is to label the first atom of each residue; an atom selection overrides this, unless the FIRSt keyword is present; in this case, the first atom of each selected residue is labeled.

Examples:

```
! the first atom of each residue is labeled by name with normal text
LBL RESN COLOR CYAN

! the first atom of each residue in the segment MAIN is labeled
! by name and number with very small text
LBL RESN RESID FIRST SELE SEGID MAIN END SIZE VSMALL

! all oxygen atoms are labeled by charge with small text
LBL CHARGE SELE TYPE O* END COLOR CHAR

! all alpha-carbons are labeled by the weight vector with medium text
LBL WEIGHT SELE TYPE CA END SIZE MEDIUM

! enter a null label; may be used to selectively "blank" labels
! in this case, all alpha-carbon labels are set to a string of blanks
! for display efficiency, LBL INIT is preferable
LBL SELE TYPE CA END USER

! show the location of formal charges on amino acid side chains
LBL USER - SELE RESN ASP .AND. TYPE CG END SIZE LARGE COLOR GREEN
LBL USER - SELE RESN GLU .AND. TYPE CD END SIZE LARGE COLOR GREEN
LBL USER + SELE RESN ARG .AND. TYPE CZ END SIZE LARGE COLOR GREEN
LBL USER + SELE RESN LYS .AND. TYPE NZ END SIZE LARGE COLOR GREEN
```

```
HBStyle  [COLOR color-name]  [WIDTH iwidth]  [DASH idash]
```

Set the style for representing HBONDS; color-name is as for the COLOR command, and iwidth and idash are integers in pixel units. Specifying HBSTYLE alone resets to the default style, which is equivalent to

```
HBSTYLE COLOR ORANGE WIDTH 4 DASH 4    (N.B. DASH 0 = solid line)
```

If at least one option (COLOR, WIDTH, or DASH) is specified, the

remaining options are unchanged; thus HBSTYLE COLOR WHITE will not reset the WIDTH to 4 pixels, but leave it as it was.

AXEs [XLIM xmin xmax] [YLIM ymin ymax] [ZLIM zmin zmax] [DEFAult]

Changes the length of the displayed axes; the default, which is restored via the DEFAult keyword, is -25 to +25 A for each dimension. Only the axes lines specified are changed, e.g. AXES ZLIM -5 5 only changes the endpoints of the Z axes line.

STEreo [ON/OFF] [dist] [angle]

Invoke side-by-side stereo mode for screen display and for output files produced by the PLOT and MAKE commands, when the ON keyword is used, or when stereo is off. The dist option controls the separation between the two images; the angle option specifies the parallax angle for left and right eye views. The OFF keyword is used to return to mono mode, and is assumed if the command is used while in stereo mode. The default angle is 7 degrees, but the dist parameter defaults to an interval derived from the window width; wider windows are best for stereo.

Example:

stereo on 16.0 7

TEXT [text-body | "text-body"]

Supply the text for the title display; quotes override the conversion to all upper case, and a null entry clears the current title. The quotes are removed, and are not displayed as part of the title; the current CHARMM title is the default graphics title.

FONt [VSMAll | SMALl | MEdium | LArge]

Change the title font to one of four sizes; the initial setting is MEDIUM, as is the default if no size is specified.

SCAle factor [MOL/LAB] [REP int]

Change the Angstrom/pixel scale factor; initially, 1 A = 32 pixels. The repeat factor can provide an ersatz zoom effect. The default is the LAB frame (i.e. scale is done relative to screen center rather

then the system center).

BOXsize size [MOL/LAB]

Set the viewing 'box' to (size)x(size), in Angstroms.

CENter [atom-selection]

Move the selected atoms to the center of display space.

MAXwindow

Scales the molecule to fit in the display window.

POInt x y z

The point specified by x y z becomes the center of display space.

ROTate rx ry rz [MOL/LAB] [REP int]

Apply a rotation to the viewing transform; does not affect the coordinates. The default is the LAB frame (i.e. a Z rotation always spins the screen). If the MOL (molecule) frame is used, then the rotation will be about the origin of the molecular coordinate system (i.e. does not depend on the current view matrix).
(NOTE1: the ROTate command uses a left handed system, multiply rotation angles by -1 as necessary).
(NOTE2: if more than one rotation angle is given as nonzero, then the rotations will occur sequentially, first x, then y, and then z).

Examples:

```
rot 0 90       ! rotate by 90 deg around the y axis
rotate 180     ! rotate by 180 deg around the x axis
rot 90 0 90    ! rotate by 90 deg around x, then 90 deg around z
```

TRAnslate x y z [MOL/LAB] [REP int]

Apply a translation to the viewing transform; does not affect the coordinates.

Examples:

```
tran 2 9.5     ! translate +2 A along x axis, +9.5 along y axis
tra 0 0 4      ! translate +4 axis along the z axis
```

ZCLip [low] high

Set hither and yon clip limits; atoms outside the limits are not displayed, whether selected or not.

Examples:

```
zclip 10       ! atoms outside z = ( -10 .. +10 ) are not displayed
zclip -5 10
```

ZCUe [[low] high]/[AUTO]

Controls the z coordinate range over which depth cueing will be applied. AUTO use the displayed atom coordinates to set the limits.

Examples:

```
zcue 10        ! zcue from -10 to +10
zcue -4 8
zcue auto
```

AUTo [ON/OFF]

Enables or disables automatic redraw after every command; the initial setting is ON, and the command functions as a toggle. AUTO OFF is useful for making multiple changes without the time required for a redraw.

ERAse [ON/OFF]

Enables or disables a screen clear prior to the next drawing; the initial setting is ON, and the command functions as a toggle. ERASE OFF is useful for overlaying trajectory frames or other related collections of structures. Possibly best used with AUTO OFF, and using DRAW for each structure. Also applies to POV and PSC commands; after using the INIT keyword, the file is not closed until one of the following:

(1) PSC/POV command with TERMinate keyword; only correct method
(2) the UNIT is CLOSEd; no title or page eject, etc.
(3) the program terminates; no title or page eject, etc.

Creates an overlay of 50 consecutive trajectory frames (coord traj open on unit 20; .ps file open on unit 12):

```
erase off
psc unit 12 init
set n 1
label frmlp
coor read file unit 20 ifile -1
psc unit 12
let n += 1
if n .lt. 50 goto frmlp
psc unit 12 term
```

EXEcute pathname

Execute a program w/o arguments.

Example:

```
exe /bin/ls         ! list the current directory  (unix)
```

Equivalent to the SYSTem command (miscom.doc)

File: Graphx, Node: Output, Up: Top, Previous: Description, Next: Addendum

Output Facility

PSC UNIT n [COLOr] [PORTrait] [BWREverse] [INIT | TERM]

Writes out a PostScript display file using the current viewing transform and selected atoms; the default is grayscale in landscape mode, and may be changed using the COLOr and/or PORTrait keywords. Colors are direct translations of the RGB color map used for Apollo and X11 displays, or arbitrary conversions to grayscale. Direct control over grayscale may be achieved by setting all atoms to COLOR WHITE, and using the 'brightfactor' option of the COLOR command to set the various gray levels; note that full WHITE is black, since the background (paper) is white in grayscale mode. The default background is black in COLOR mode, but may changed to white
with BWREverse keyword; atoms or text colored white on the screen will be printed in black. As always, the UNIT must be opened first (recommended file extension .ps).

The INIT keyword forces device initialization in ERASE OFF mode, and should only be used in that context; it is not normally required. Likewise, the TERM keyword writes the final part of the PostScript file in ERASE OFF mode, and does not draw the structure; only the graphics title and the final few PostScript commands are written to the file. See the ERAse command for additional details and a usage example.

PLUto UNIT n

Writes out atom coordinates and connectivity based on the current atom selection and view transform, in CSD FDAT format.
WARNING: 999 atom limit!

Stereo mode settings are ignored, as are radii and color. Also, atoms should be renamed to their element types to get proper radii, etc within pluto (e.g. CA is not carbon alpha, it's calcium). As with PLOT, the UNIT must be opened first (recommended file extension .fdat).

Example:

```
rename atom C1 sele type C  end
rename atom C2 sele type CA end
rename atom N1 sele type N  end
rename atom O1 sele type O  end
rename atom H1 sele type H1 end
   ! etc., etc., etc.
graphics
scale .5
rot 0 90
open unit 50 write card name molecule.fdat
pluto unit 50
```

MAKE UNIT n

Writes atom coordinates, radii, and color to a file formatted for the LIGHT program (available from the N.I.H.) which produces nice ray-traced images using current stereo settings and view transform.

A 1280x1024 window is the optimum size for best scaling and centering of the image produced by LIGHT.

The UNIT must be opened first; the LIGHT program requires that an extension of .atm be use for the file name.

POV UNIT int [INIT] [INCLude int]
[TERM]

Write atom-based objects; bonds, atoms, H-bonds, labels

```
  UNIT int    :: final POV file; user must open
                 must also open int+1 for stereo
  INCLude int :: read file w. user override of default camera,
                 lighting, color defns, ...

  INIT        :: intialize POV file in ERASE OFF mode (overlays)
```

```
  TERM        :: close POV file in ERASE OFF mode; other options ignored
```

All units must be opened; in STEreo mode, UNIT numbers are used in pairs as N, N+1 for the left- and right-eye views. The atom, bonds, H-bonds, and labels currently displayed are written to the file; just as for the X11 display and PostScript output, this is determined by the last atom selection used with a DRAw command, and graphics elements enabled via the DISplay command. The current viewpoint is used for transforming coordinates for the POV output file, and the graphics SCAle determines the camera distance. The atom radii are determined from the current graphics RADii, while the bond cylinders are scaled based on the graphics LINe width.

The POV file produced (recommended extension .pov) is a flat text file, and may be manually edited to further customize the image that will be ultimately created by processing the POV file. Note that CHARMM only creates an input file for POV-Ray; it is assumed that the user has access to version 3.1 of the POV-Ray program. When using the povray3 program on Unix systems, the PPM bitmap output format is recommended. For details on obtaining and using the POV-Ray program, see the web site at http://www.povray.org

The INCLude file provides an easy mechanism to add user customization, e.g. changing the light sources or camera position, or adding other POV-Ray objects to be displayed. Comments in the POV input file created indicate the section which is completely replaced by the userfile; using the CHARMM supplied default setup as a template for INCLude files is strongly encouraged.

The INIT keyword forces device initialization in ERASE OFF mode, and should only be used in that context; it is not normally required. Likewise, the TERM keyword writes the final part of the POV file in ERASE OFF mode, and does not output a structure; only the graphics title is written to the file. See the ERAse command for additional details and a PSC (PostScript) usage example.

Example: write a single POV file

```
open unit 21 write card name protein.pov
pov unit 21
```

Example: write two POV files for a stereo image pair

```
open unit 21 write card name protein-l.pov
open unit 22 write card name protein-r.pov
pov unit 21
```

File: Graphx, Node: Addendum, Up: Top, Previous: Output, Next: Top

X11 Usage Tips

On some workstations, the graphics display colors will only be correct when that window has "focus".

The recommended "focus" policy is pointer-has-focus, i.e. the active window is whichever one the mouse pointer is in; this permits typing graphics commands in a mostly obscured terminal window (xterm, winterm,hpterm, aixterm, etc.). The alternative is to "lower" the text command
window via the window manager. Automatic "topping" of the window with focus is not recommended.

Although the graphics display window can be resized, the size change is not currently passed back to the drawing routines.

X11 Compiling Problems

The default assumptions are:

(1) required include files (Xlib.h, Xutil.h) reside in /usr/include/X11

(2) the library libX11.a resides in /usr/lib/X11

These files *must* exist for the X11 graphics to be compiled, and of course the keyword XDISPLAY must be in the pref.dat file. If the files are NOT in the standard places (such as under HP-UX), the compile option -I may be needed to point to the directory with the include files, and the corresponding -L option may be needed to point to the directory which contains libX11.a for the final link step.

Other Useful Programs (all "freeware")

(1) ghostscript -- besides allowing on screen preview of PostScript files prior to printing, this useful utility can convert the .ps files output by the PSC command to other formats (e.g. GIF, PBM) and can also be used to make an "encapsulated" PostScript (EPS) file. Available via FTP from almost any software archive that distributes GNU software (Free Software Foundation).

(2) xfig -- this X11-based technical drawing package can read EPS files, and provides an ideal way to do "pasteup" of PostScript files from diverse sources into a composite figure. Available via FTP from sites carrying X11 software; hardware specific sites may have versions pre-configured for SGI, HP, etc.

(3) pbmplus, netpbm -- this suite of bitmap manipulation programs offers the best looking conversion to GIF files, using a PBM file from ghostscript as an intermediate file type. The names pbmplus and netpbm indicate minor variants of the same suite, with netpbm being more recent, according to UseNet lore. Available at the MIT X11 site, and similar FTP sites.

(4) light -- a ray-tracing program written by BR Brooks at NIH, which uses the file format ouput by the graphics MAK command; used to produce movies and slick-looking single images for presentations. Contact the author.

(5) gnuplot -- 2D and 3D graphics, with X11 and PostScript support (and about 50 other output devices as well); useful for time series plots, etc.,

which can be combined with molecular graphics via xfig (see above). Available at most GNU sites and from the "home" site at Dartmouth.

(6) ImageMagick -- a powerful suite of image manipulation programs, including animation capabilities; available at ftp.x.org

(7) MPEG – ppmtompeg (formerly mpegencode) and mpeg_play produce and replay animations, respectively; multiple images in PPM color image format are combined into a movie.

Compiling for X11 on gnu

Before proceeding to some examples, it should be noted that a small error has been present in most c3?b? versions that causes “unresolved global” errors (for the X11 utility routines) when the ‘X’ option of install.com is used with the ‘gnu’ machine type. There are two solutions:

[1] The quick fix; edit build/gnu/Makefile_gnu, changing

```
GLIB =
```

to

```
GLIB = -L/usr/X11R6/lib –lX11
```

Then re-run install.com, and the final link step should succeed this time.

[2] Permanent version fix; edit build/UNX/Makefile_gnu, changing

```
GLIB =
```

to

```
GLIB = grflib
```

This will fix the ‘template’, such that install.com can (re)create the build/gnu subdir and Makefile_gnu will be correct for linking with the X11 libraries.

Interactive X11 CHARMM usage

For graphics usage, CHARMM is run in foreground, with input and output via the terminal. There can be some variation in how this done, depending on the how the local group of computers is

setup. Also, where the files are stored with respect to the computer being used can affect how things are done; for example, a PC or Mac with X11 capability could be used to visualize structures from data stored on Biowulf, LoBoS or some other group of computers. The usage of SSH for remote connections has some impact as well; the use of tunneling supports remote X11 usage, but each host in the chain must be properly configured.

Biowulf and LoBoS setup examples

From the Biowulf CHARMM documentation—

The CHARMM X11 graphics commands can also be used via mpicharmm; a pre-requisite is to login to biowulf using the ssh command. From some systems, the -X option to ssh is needed to setup use of X11, e.g.

ssh -X biowulf

The next step is to request an "interactive batch" node, via

qsub -l nodes=1 -I -V

Finally, start CHARMM via

mpicharmm proc=single (or proc=0)

At this point, it's useful to have one or two CHARMM stream files that do all the system setup, e.g. read in RTF, PARAM, PSF and COOR file(s). Then one can simply type

```
* a title
*
stream init-prot.str
graphx
draw sele .not. type H* end
center
scale 0.5
... (etc.)
```

Graphics features include trajectory file viewing, and output of coordinate data as a file of POV-Ray objects.

On the LoBoS cluster, X11 enabled SSH connections can be obtained via de23.lobos.nih.gov; at the present time, compute nodes do not support an X11 connection, so Linux desktops or

‘m1’ should probably be used. A pointer to a graphics enabled CHARMM version can be established via

```
alias xcharmm /v/apps/mpicharmm/c32b2/g77-xlg.one-X11
```

This can be conveniently placed in the ~/.cshrc file (or equivalent), so that the command alias ‘xcharmm’ will be understood in all shells.

Startup and graphics initialization

After establishing a proper X11 setup, it is advisable to create a file of CHARMM commands, to be read via the STREAM command. This is key to interactive usage, and allows automation of routine setup tasks. Consider the following:

```
* view 1irk initial structure
*

bomblev -1

stream rtfprm.str
stream psfcrd.str

graphx xxsz 800 xysz 700
center sele resn atp .or. ( resn tyr .and. segid B ) end
sca 0.35
draw sele .not. type H* end
radii param 0.5
display atoms
rad 0.28 sele segid A end
rad 2.5 sele segid M .or. segid CL .or. segid NA end
rad 0.4 sele segid W .or. segid WA+ end

color cyan sele segid W end
color dkbl sele segid WA+ end
color purple sele segid M end
color green sele type P* end
color gray sele atom B * C* end
color oran sele atom B * O* end
color gray sele atom N * C* end
color char sele segid CL end
color mage sele segid NA end

return
```

A predefined view using an initial model structure (read via the **psfcrd.str** stream file) is setup by the above; it can be used via the following keyboard input:

```
xcharmm
```

```
* startup
*
stream vu-init.str
```

(On Biowulf, the command would be ‘mpicharmm proc=0’.) After the STREAM file finishes, further commands can be typed to change the view; the PSC command can be used to create a fairly faithful copy of the screen image at any time; the BWREV option provides a white background, and inverts the grayscale.

Once a view is setup, it can be used to view additional files that use the same PSF, such as formatted COOR files at various points in a process, or even trajectory files. The GRAPHX parser supports all of the “miscellaneous commands” (miscom.doc), as well as the COOR (corman.doc), IC (intcor.doc), and TRAJ (dynamc.doc) commands. This allows the use of CHARMM input loops to view an arbitrary trajectory file:

```
* graphics movie stream for dynamics file
*

open unit 21 read file name dyn@N.trj
traj first 21 nunit 1
auto off

set k 1
label frmlp
calc t = 50. * ( @N - 1 ) + @K * 0.5
text 1irk frame at @T ps
traj read
draw
incr k by 1
if k le ?NFILE goto frmlp

auto on
return
```

Again, this is used via the STREAM command:

```
set N 20
stream vu-mov.str
```

will sequentially show the frames (stored at 0.5 ps intervals) from the file dyn20.trj, as rapidly as possible. Good bandwidth is needed for this; a DSL connection may not be sufficient.

Another useful feature of the X11 and PostScript graphics is the ability to create overlays of multiple structures, possibly with some alignment. The following was used to create an

ensemble overlay of 100 coord sets for the backbone of a small peptide.

```
* overlay of coord sets; expt @S
*

bomblev -1

! READ RTF/PRM/PSF/CRD FILES
stream rtfprm.str
stream init-s2.str
stream list/u@S.str

open unit 2 read card name av@S.crd
read coor card unit 2 comp
close unit 2
define back sele type N .or. type CA .or. type C .or. type O end
define cmmn sele resid 50:57 end
define even sele resid 50 .or. resid 58 end
coor orient comp sele back end
scalar xcomp stat sele atom B 50 CA end
set xca ?SAVE
if xca .gt. 0.0 coor rota ydir 1.0 phi 180. comp
coor orient rms sele back .and. cmmn end

! GRAPHICS SETUP
graphics xxsz 600 xysz 480
!graphics nowin
auto off
draw sele back end
center
scale 0.55
rot 0 0 90
stereo 10.
font large
text
line 1
color dkbl sele type N end

open unit 31 read file name q@S.trj
set frm 1
! LOOP OVER FRAMES IN SUBSET FILE VIA IFILE -1
label frmlp

! FIRST TIME ONLY
if frm .ne. 1 goto skip1
coor read file unit 31 ifile 1
coor orient rms sele back .and. cmmn end
draw
erase off
open unit 12 write card name o@S.ps
psc init unit 12 color bwrev
lbl init
```

```
goto skip2

! ALL OTHER PASSES
label skip1
coor read file unit 31 ifile -1
coor orient rms sele back .and. cmmn end
draw
psc unit 12 color bwrev

label skip2
incr frm by 1
if frm .le. @NFRM goto frmlp
lbl resid sele type CA .and. even end size large color white
draw
psc unit 12 color bwrev
psc term unit 12

off
stop
```

The above script was written before the TRAJ I/O feature was added to GRAPHX, so it uses an alternate method of reading sequential coord sets from a trajectory file (IFILE -1).

Producing files for POV-Ray

For more elegant molecular graphics, a file-based interface to the excellent freeware ray tracing program POV-Ray was developed. Because of some changes in more recent versions, POV-Ray 3.1 is recommended for the best compatibility with the files produced by CHARMM. Another key point to make is that the imaging model is quite a bit different, and isn't all that well correlated to the X11 (or PostScript) images. Also, there a few bugs which further limit the equivalence of the X11 graphics and the output .pov file. Specifically, the ROT, LBL, STEREO, and ERASE OFF commands should not be used (they are broken). The most important workaround is to use COOR ROTA for all rotations when preparing .pov output. Labelling is best done after the fact, using standard image editing programs (PhotoShop, GraphicConvertor, ImageMagick, GIMP, etc.). Currently, stereo is best done by applying a small rotation around the Y axis and writing a second .pov file. Finally, overlays can be done writing separate files, then concatenating files 2-N to the first with their headers removed.

Despite a few flaws (*mea culpa*), the facility is still quite useful; some knowledge of POV-Ray can add to the variety and quality of images produced. At the very least, one must know how to use the 'povray' program to process the .pov file into an

image file with the desired characteristics. Because of the variation in how the program is installed, an alias is useful providing a standard name for invoking the program. For Biowulf, an example is:

```
alias pov3 'povray +L/usr/local/lib/povray31/include +fp'
```

while on LoBoS it would be (the line is wrapped):

```
alias pov3 '/v/apps/povray-3.1/bin/x-povray +fp +L/v/apps/povray-
3.1/lib/povray31/include'
```

This way, 'pov3' can be used to run the program on either system w/o remembering all these gory details, *e.g.*

```
pov3 –iprot5ns.pov –oprot5ns.ppm –w640 –h480 +A +AM2
```

The above produces prot5ns.ppm, a PPM full-color 640x480 bitmap with anti-aliasing. The default output format is Targa, and can be obtained by omitting the '+fp' argument; however, PPM is most useful for making MPEG animations, and PPM can be easily converted to a wide variety of other formats.

The next CHARMM input example produces .pov files corresponding to viewpoints from each of the 6 principal directions, *i.e.* toward the origin from both ends of each Cartesian axis. The system is an aqueous lipid bilayer with 40 sugar molecules. Note the use of GRAPHICS with the NOWIN keyword; this is designed to allow batch usage, and does not require a valid X11 setup.

```
* final frame of file @N
*

stream rtfprm.str
stream psfcrd.str
stream cryst.str

open unit 3 read file name dyn@N.trj
read coor file unit 3 ifile 100
close unit 3

update imgfrq 1
coor copy comp

graphics nowin
center
draw sele .not. type H* end
auto off
```

```
line 1
disp atom
rad param 0.8
rad 0.3 sele segid WAT end
rad 0.15 sele segid L end
rad 1.5 sele type C+ .or. type C1+ end
rad 4.0 sele atom L * O32 .or. atom L * O22 end
rad 4.0 sele type P end
rad 0.1 sele resn aglc end
sca 0.31

color black sele type O* .or. type N end
color cyan sele type OH2 end
color red sele atom L * O32 end
color red sele atom L * O22 end
color gray sele type C+16 end
color green sele type P end
color purpl sele resn dppc .and. ( type N .or. type C13 .or. type C14 -
  .or. type C15 ) end
color orange sele resn aglc end
color yellow sele atom T+# * O1 end

text

! BOTH ENDS
open unit 11 write card name views/xy+z@N.pov
pov unit 11
coor rota xdir 1. phi 180.
open unit 11 write card name views/xy-z@N.pov
pov unit 11

! FOUR SIDE VIEWS
coor rota xdir  1. phi 90.
open unit 11 write card name views/xz+y@N.pov
pov unit 11

coor rota ydir 1. phi 180.
open unit 11 write card name views/xz-y@N.pov
pov unit 11

coor rota ydir 1. phi 90.
open unit 11 write card name views/yz+x@N.pov
pov unit 11

coor rota ydir 1. phi 180.
open unit 11 write card name views/yz-x@N.pov
pov unit 11

off
stop
```

In order to help automate the production of the .pov files and the corresponding images, a shell script can be used. One

additional note for Biowulf users; the following .csh script uses a special parallel version of POV-Ray, which requires an alternate setup using MPICH. (Because the implementation uses the first process as a manager, it's more efficient to run 3 processes per node.)

```
#!/bin/csh
#PBS -j oe
#PBS -N Pov6vu

cd $PBS_O_WORKDIR
set mrun = "/usr/local/mpich/bin/mpirun -machinefile $PBS_NODEFILE"

# MAKE THE .pov FILES; USE qsub –v t= TO PASS VAR $t
mpicharmm T:$t proc=0 size=large < pov6vu.inp >& pov6vu$t.out

# LOOP OVER 6 VIEWPOINTS; MAKE IMAGE FILE, COMPRESS
foreach f ( xy_z xy-z xz+y xz-y yz+x yz-x )
 set p = $f$t
 if ( -e $p.pov ) then
 $mrun -np 3 /usr/local/bin/mpi-x-povray -i$p.pov -o$p.ppm \
   -w800 -h600 +A +AM2 +fp +L/usr/local/pov/include >>& pov6vu$t.out
 gzip -f $p.pov $p.ppm
endif

# LOOP OVER VIEWS
end
```

MPEG animations using CHARMM and POV-Ray

With only a few changes and one additional step, MPEG-1 animation files can be produced. This is a compressed video format, and can be displayed by many multimedia players, and the classic mpeg_play program. (Note: many people are more familiar with the later MPEG-3 format, *i.e.* .mp3 files.) The program ppmtompeg (formerly mpeg_encode) is available on many Linux systems (including Biowulf and LoBoS), and provides a means of combining a consecutive series of PPM files into a single file. The total size of the files is significantly smaller than the sum of the PPM file sizes. Following are three scripts used recently on Biowulf; the CHARMM input that makes .pov files, the .csh script that runs CHARMM and makes the .ppm files with POV-Ray, and a final script to run mpeg_encode on the result. (An advantage of this older version is that can 'gunzip' compressed .ppm files dynamically; ppmtompeg seems to require uncompressed files). First, the CHARMM script (this is for a peptide:micelle complex):

```
* pov files for animation of file @N
```

```
*

stream rtfprm.str
stream imcllwat.str

bomblev -1
graphics nowin
draw sele ( segid M .or. segid B ) .and. .not. type H* end
auto off
center
text
scal 0.30
coor copy comp
coor rota xdir 1.0 phi -90 comp
coor tran xdir -30.
coor tran xdir  30. comp
disp atom comp
disp bond off

color green sele type P .or. type OP+ end
color purp sele atom M * N .or. type C15 .or. type C16 .or. type C17 end
color gray sele atom B * C* end
color dkbl sele atom B * N* end
color turq sele ( resn HSP .and. type N%+ ) .or. –
  ( resid 50 .and. type N ) end
color brown sele ( resn ile .or. resn pro ) .and. -
  ( type CB* .or. type CG* .or. type CD* ) end

color green sele type P .or. type OP+ end comp
color purp sele atom M * N .or. type C15 .or. type C16 .or. type C17 end
comp
color gray sele atom B * C* end comp
color dkbl sele atom B * N* end comp
color turq sele ( resn HSP .and. type N%+ ) .or. -
  ( resid 50 .and. type N ) end comp
color brown sele ( resn ile .or. resn pro ) .and. -
  ( type CB* .or. type CG* .or. type CD* ) end comp

rad param
draw
if n .gt. 1 goto skip0
open unit 12 write card name 0anim/frm0.pov
pov unit 12
label skip0

open unit 9 read file name dyn@N.trj
traj iread 9 nread 1
calc t 1 + 50 * ( @N - 1 )
calc fin @T + 49

label tloop
traj read
coor copy comp
coor rota xdir 1.0 phi -90 comp
```

```
coor tran xdir -30.
coor tran xdir  30. comp
open unit 12 write card name 0anim/frm@T.pov
pov unit 12
incr t by 1
if t .le. @FIN goto tloop

off
stop
```

Most of what's above has been discussed already. Likewise, this .csh script is a lot like the previous one:

```
#!/bin/csh
#PBS -N POV-dyn
#PBS -j oe

cd $PBS_O_WORKDIR

# n passed as arg via qsub -v

mpicharmm vrsn=c28b1 proc=single size=large N:$n PRFX:povdyn \
  < cover.inp >& povdyn$n.out

set mrun = /usr/local/mpich/bin/mpirun
set mf = "-machinefile $PBS_NODEFILE"

@ t = 1 + 50 * ( $n - 1 )
@ e = $t + 49
if ( $n == 1 ) @ t = 0

while ( $t <= $e )
 set fil = 0anim/frm$t
 $mrun $mf -np 2 /usr/local/bin/mpi-x-povray -i$fil.pov -o$fil.ppm \
   -w800 -h600 +A +AM2 +fp +L/usr/local/pov/include >>& povdyn$n.out
 gzip -f $fil.pov $fil.ppm
 @ t += 1
end

gzip povdyn$n.out
```

This script will extract the frames from a single trajectory file and create compressed .ppm image files for each. For making an animation, this allows multiple trajectory to be processed separately; the above script is designed to be submitted to a PBS queue, and expects the file number to be passed as variable, *e.g.*

```
qsub –l nodes=1 –v n=50 povmd.csh
```

Once all of the image files are available, the final step is to combine them into an MPEG file.

```
#!/bin/csh
#
#PBS -N mpeg_encode
#PBS -j oe
#PBS -m ae

cd $PBS_O_WORKDIR

# HEADER INFO; TIMESTAMP, VERIFY NODES, EXECUTABLE, ETC.
date
uname -a
uptime
pwd

set d = p60e

cat > mpg.prm << FINIS
# parameter file with good default values
#
# use this as a guideline for any parameters you don't really understand
# or don't care about
#

PATTERN         IBBPBBPBBPBBPBB
OUTPUT          $d/500ps.mpg

BASE_FILE_FORMAT        PPM
GOP_SIZE        10
SLICES_PER_FRAME        1

PIXEL           HALF
RANGE           10
PSEARCH_ALG     LOGARITHMIC
BSEARCH_ALG     CROSS2
IQSCALE         8
PQSCALE         10
BQSCALE         25

REFERENCE_FRAME ORIGINAL
FORCE_ENCODE_LAST_FRAME

#
# you really need to understand the following
#
#YUV_SIZE       352x240
INPUT_CONVERT   gzip -dc *

INPUT_DIR       $d/0anim

INPUT
```

```
frm*.ppm.gz    [0-500]
END_INPUT

FINIS

mpeg_encode mpg.prm
```

To use the above with ppmtompeg, several changes are needed:

1. The .ppm files must be uncompressed (gunzip)
2. **Change** `INPUT_CONVERT  gzip -dc *` **to** `INPUT_CONVERT  *`
3. **Change** `frm*.ppm.gz` **to** `frm*.ppm`
4. Change the program name from ‘mpeg_encode’ to ‘ppmtompeg’

The scripts presented here are by way of illustration; most will need changes for use for a particular project or computing environment. It is hoped that they illustrate the basic principles enough that the value and power of the CHARMM graphics can be perceived and used effectively.