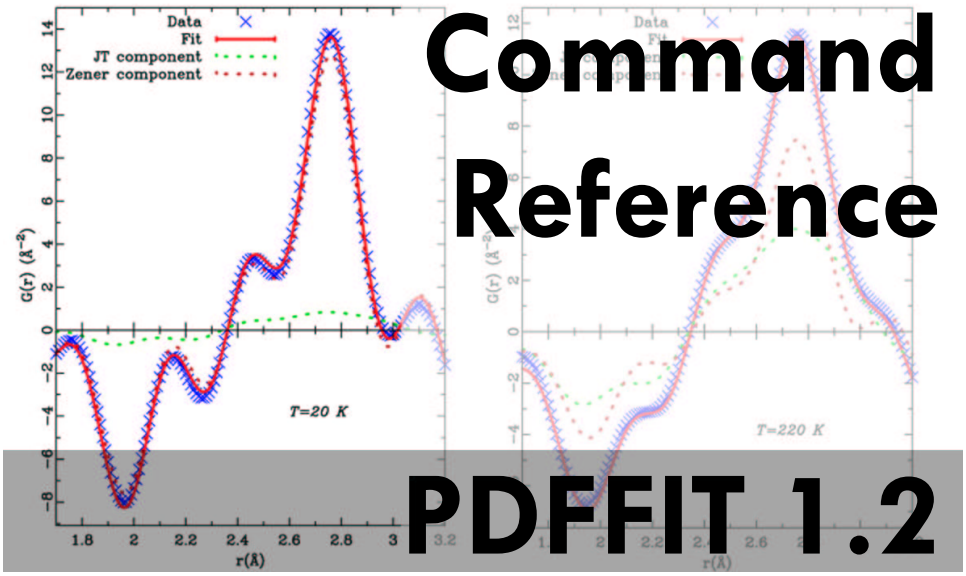

Command Reference

PDFFIT 1.2

written by

Thomas Proffen
Email: tproffen@lanl.gov

Simon Billinge
Email: billinge@pa.msu.edu

Document created: February 27, 2003

Preface

Disclaimer

The *PDFFIT* software described in this guide is provided without warranty of any kind. No liability is taken for any loss or damages, direct or indirect, that may result through the use of *PDFFIT*. No warranty is made with respect to this manual, or the program and functions therein. There are no warranties that the programs are free of error, or that they are consistent with any standard, or that they will meet the requirement for a particular application. The programs and the manual have been thoroughly checked. Nevertheless, it can not be guaranteed that the manual is correct and up to date in every detail. This manual and the *PDFFIT* program may be changed without notice.
PDFFIT is intended as a public domain program. It may be used free of charge. Any commercial use is, however, not allowed without permission of the authors.

Acknowledgments

The basic concept of the full profile refinement of the pair distribution function (PDF) was adapted from the program *RESPAR* written by Simon Billinge. The routines for the processing of the command language and the FORTRAN style interpreter are taken from the program package *DISCUS* by Reinhard Neder and Thomas Proffen. *PDFFIT* is distributed as part of that package as well as a stand-alone program. The routines for the line editing were taken from the program GNUPLOT.

Using PDFFIT

Publications of results totally or partially obtained using the program *PDFFIT* should state that *PDFFIT* was used and contain the following reference:

> PROFFEN, TH. & BILLINGE, S.J.L. (1999) "PDFFIT, a Program for Full Profile Structural Refinement of the Atomic Pair Distribution Function" *J. Appl. Cryst.*, **32**, 572-575

Contents

Getting started

PDFFIT is a program for the full profile structural refinement of the atomic pair distribution function (PDF). For details refer to the users guide for this program or to the article by S. Billinge in "Local Structure from Diffraction", Plenum Press, New York, 1998.
Further help topics are:

Chapter 1

PDFFIT commands

1.1 Summary

Here is a short summary of the PDFFIT specific commands currently available:

```
alloc   : Allocates space for calculating PDFs without reading data
bang    : Calculates bond angles with standard deviations
blen    : Calculates bond lengths with standard deviations
calc    : Calculates PDF from current structure
cycle   : Sets the maximum number of refinement cycles
i/jdese : Selects atoms used for calculating the PDF
i/jsele : Deselects atoms used for calculating the PDF
occ     : Sets occupancy for a given site
par     : Sets a parameter definition
phase   : Switches active structural phase
psel    : Associates structural phases with data sets
range   : Sets r-range used for refinement
read    : Reads structure and PDF data files
reset   : Resets PDFFIT
run     : Starts a refinement
save    : Saves resulting structure, PDF data and fit results
scat    : Overwrites internal scattering factor for given element
show    : Shows various information
temp    : Sets anisotropic temperature factor
urf     : Sets URF (magic refinement number)
xray    : Specifies Q-value for calculating X-ray form factors
```

1.2 alloc

alloc {**"x"**|**"n"**},<**qmax**>,<**qsig**>,<**rmin**>,<**rmax**>,<**np**>

This command allows the user to allocate space when just calculating a PDF. The first three parameters are similar to ’read data’ and the next three parameters specify the range in r to be from <rmin> to <rmax> and the number of points <np>. Note that this command allocates the space for one data set without actually reading any data ! Use the command ’reset’ before reading "real" data for a refinement.

1.3 bang

bang <**a1**>,<**a2**>,<**a3**>

This command calculates the bond angle and its standard deviation for the atoms numbered <a1>, <a2> and <a3>. The angle is measured at <a2>. Note that no periodic boundaries are applied and it might be necessary to add +-1 to the coordinates of some of the atoms to get the desired angle (-> variables).

1.4 blen

blen <**a1**>,<**a2**>
blen {**"all"**|<**name**>|<**number**>},{**"all"**|<**name**>|<**number**>},<**b1**>,<**b2**>

The simple form of this command calculates the bond length and its standard deviation between the atoms number <a1> and <a2>. Note that the 'real' distance within the crystal is given and *no* periodic boundary conditions are applied.
The second command calculates the bond lengths between the atoms selected by the first two parameters that fall in the range specified by <b1> and <b2>. As usual atoms can be selected by name or type number or simply use "all" to select all atoms. Here periodic boundaries are applied like for the PDF calculation itself. Thus the results of the two versions of 'blen' might be different for the same atom number pair !

1.5 calc

calc

This command starts the calculation of the PDF for the current settings. Note that the current refinement parameter settings are applied before the PDFs corresponding to all currently loaded data sets are calculated. Note that the command 'save' must be used to save the result(s).

1.6 cycle

cycle <**n**>

This command sets the maximum number of refinement cycles to <n>. If a minimum is found in less cycles, the fit will stop before <n> cycles are finished.

1.7 functions

The following PDFFIT specific functions exist. For a listing of general intrinsic functions see help entry 'functions' in the 'command language' section of the online help.

```
bang(u1,u2,u3,v1,v2,v3[,w1,w2,w3]) :
    Returns the bond angle at the site v. If only vectors u and v
    are given, the angle between u and v is returned.

blen(u1,u2,u3[,v1,v2,v3]) :
    Returns the length of vector v-u. Vector v defaults to zero.
```

```
dstar(h1,h2,h2[,k1,k2,k3]) :
    Returns the length of reciprocal vector k-h. Vector k defaults
    to zero.

rang(h1,h2,h3,k1,k2,k3[,l1,l2,l3]) :
    Returns the angle between vectors k-h and k-l at reciprocal site k.
    If l is omitted, the angle between the reciprocal vectors h and k
    is returned.

gran(val,typ) :
    Returns Gaussian distributed pseudo random number with mean zero
    an a width given by parameter <val>. If <typ> is "s" <val> is taken
    as sigma, if <typ> is "f", <val> is taken as FWHM.

gbox(r1,r2,r3) :
    Returns pseudo random number with distribution given by a box
    centered at 0 with a width of <r2> and two half Gaussian
    distributions with individual sigmas of <r1> and <r3> to the
    left and right, respectively.
rval(s [,rmin,rmax])
    Returns the weighted R-value corresponding to data set <s>.
    The optional parameters allow the user to calculate the R-value
    for a specific region in r.
```

1.8 i/jdese

idese <**is**>**,** { **"all"** | <**name**> | <**number**> }**, [...]**
jdese <**is**>**,** { **"all"** | <**name**> | <**number**> }**, [...]**

This command deselects atom types given either by <name> or <number> for the PDF calculation. The two commands allow one to deselect atom types for each atom in a pair 'ij' contributing to the PDF calculation. The first parameters <is> specifies the number of the corresponding data set.

1.9 i/jsele

isele <**is**>**,** { **"all"** | <**name**> | <**number**> }**, [...]**
jsele <**is**>**,** { **"all"** | <**name**> | <**number**> }**, [...]**

This command selects atom types given either by <name> or <number> for the PDF calculation. All other atoms are ignored. This allows the calculation of differential or partial PDFs. The two commands allow one to select atom types for each atom in a pair 'ij' contributing to the PDF calculation. The first parameters <is> specifies the number of the corresponding data set.

1.10 occ

occ {**"all"** | <**name**> | <**number**>}**,**<**o**>

This command allows the user to set the occupancy of a given site. The parameter "all" will result in the assignment of the occupancy <o> to all atoms of the currently active phase (-> phase). Alternatively a specific atom name <name>

or even more specific the number <number> of a single atom type can be used. Note that for DISCUS compatibility the occupancies are not part of the structure file and must be set separately. The default value is 1.0.

1.11 par

par <x=f(p(i))>,<dx/dp1>,<dx/dp2>,...
par "reset"

This command defines the relation between fit parameters (p[i]) and experimental and structural parameters, e.g. x[i]. There is no predefined sequence or order. For each parameter p[i] that is used in the definition, the corresponding partial derivative must be given. The user has also to take care of proper starting values for the parameters which can be assigned with p[i]=<expr>. The parameter settings can be deleted using the argument "reset". Note that all expressions within arguments ([]) are evaluated after the command is entered, thus parameter definitions can be made using loops. However, other variables are saves as variables and may change during the fit. For a detailed discussion refer to the PDFFIT users guide. Here are some examples.

```
Examples

par x[1]=p[1],1.0        : x of atom 1 is refined using p[1]
par o[1]=r[1]*p[5],r[1]  : occupancy of atom 1 is r[1]*p[5] and
                           the derivative is r[1] (simple isn't it)
```

1.12 phase

phase [<ip>]

PDFFIT allows the usage of multiple phases. All structural commands and setting will affect the currently active phase. The command 'phase' allows the user to set the active phase to <ip>. Alternatively the command can be used without a parameter in which case the current setting will be displayed.

1.13 psel

psel <is>,"all"
psel <is>,<ip1> [,<ip2>, ...]

This command allows the user to associate certain phases <ip1> to <ipn> with a given data set <is>. One extreme would be to have e.g. two data sets and one phase each completely separate which is not really different from running the refinements separately. However, by using identical refinement parameters one can constrain any structural parameters between the phases. An example is given in the Users Guide. The default is that all structural phases are associated with all loaded data sets.

1.14 range

range {**"all"** | <**is**>},<**rmin**>,<**rmax**>

This command sets the range in R that is used for fitting either for "all" read data sets or for just the specified data set <is>. The limits <rmin> and <rmax> are given in A.

1.15 read

This command reads various information from a specified file. The following formats are currently supported:

1.15.1 stru

read "stru",<**file**>

The command 'read stru' allows the user to read a DISCUS type structure from the file <file>. Note that some DISCUS keywords and the use of molecules are not supported by PDFFIT. To read structural information for multiple phases simply repeat the 'read' command. In order to discard the current structure use the command -> 'reset'. Note that DISCUS only supports an isotropic temperature factor B. When reading a DISCUS structure the temperature factor is converted according to: <u(ii)> = Bglm-1/(8pi**2al*am*). PDFFIT automatically recognizes the structure file format of the input file !

1.15.2 data

read "data",{**"n"** | **"x"**}**,**<**qmax**>**,**<**qsig**>**,**<**file**>

The command 'read data' reads the observed PDF. The file format is ASCII and contains 'r G(r) dummy dG(r)' in each line. The value of 'r' is in A, G(r) is the reduced PDF. The third column is ignored (needed for KUPLOT) and the last value 'dG' is the error of the PDF used to calculate the weight (w=1/dg**2) for this point to be used for the refinement. Alternative formats are 'r G(r) dG(r)' in each line or simple 'r G(r)'. In the later case, the weights are set to unity. This is also done in case the value of dG(r) is found as zero. Additional to the filename <file> the commands needs the following parameters: First the type of radiation is specified, "n" stands for neutrons and "x" for X-rays. Next the maximum value of Q <qmax> is given followed by the value of <qsig> defining the Q resolution. To read multiple data sets just repeat the 'read' command.
If the data file contains a history part created by PDFgetN, some of the information is returned in the res[n] variables. Currently the following information is available after the 'read data' command:

```
res[1]   : Temperature where the data were collected (in K)
res[2]   : Qmax (only AFTER the data were read !)
```

1.15.3 diff

read "diff",{**"n"** | **"x"**}**,**<**qmax**>**,**<**qsig**>**,**<**file**>**,**<**rfile**>

This command is similar to 'read data' but rather than reading a PDF, a difference PDF is read from file <file>. The format is identical to the PDF file format. The additional parameter <rfile> specifies a reference PDF, so that the model PDF fitted to the read data becomes CALC - REFERENCE. See Users Guide for more details.

1.16 reset

reset

This command resets PDFFIT, e.g. all current structural and PDF information is lost.

1.17 run

run

This command starts the least square refinement ...

1.18 save

save <**subcommand**>

This command allows to write different information to a file. The valid subcommands are:

1.18.1 atoms

save "atoms",<**ip**>**,**<**file**>

This command saves the structure of phase <ip> in a format suitable for import into ATOMS using the 'File - Import - Free Form' function. In case of multiple unit cells, a supercell containing the complete crystal is created. Anisotropic thermal parameters are exported and thermal ellipsoids can be viewed using ATOMS.

1.18.2 const

save "const",<**file**>

This command saves the parameter constraints to a text file. The output is similar to the screen output created by 'show const'.

1.18.3 dif

save "dif",<**is**>**,**<**file**>

This command saves the difference between observed and calculated PDF of data set <is> to a file named <file>.

1.18.4 discus

save "discus",<**ip**>**,**<**file**>

This command saves the structural phase <ip> to the file named <file> in DISCUS format. Note that DISCUS uses only an isotropic B value which is computed according to B=8pi**2/3(u11+u22+u33). The information about anisotropic temperature factors, occupancies and standard deviations is NOT saved.

1.18.5 pdf

save "pdf" ,<**is**>**,**<**file**>

The command 'save pdf' saves the calculated PDF corresponding to the given data set <is> to the file <file>. The format is identical to the input format (-> read).

1.18.6 result

save "result",<**file**>

This command saves all information about the refinement to a text file named <file>. The output is similar to the screen output of the 'show' commands.

1.18.7 stru

save "stru",<**ip**>**,**<**file**>

The command 'save stru' saves the structure of the phase specified by <ip> to the file <file>. The file format is related to the DISCUS format. However it contains additional information like anisotropic temperature factors, occupancies and standard deviations for all values. These files can be read again by PDFFIT. To save a structural phase in DISCUS format use -> 'save discus,...'.

1.19 scat

scat <**s**>**,**{<**name**>|<**number**>}**,**<**a1**>**,**<**b1**>**,**<**a2**>**,**<**b2**>**,**<**a3**>**,**<**b3**>**,**<**a4**>**,**<**b4**>**,**<**c**>
scat <**s**>**,**{<**name**>|<**number**>}**,**<**c**>
scat <**s**>**,**{<**name**>|<**number**>}**,"internal"**

This command allows the user to specify a new scattering factor for data set <s> and atom <name> (or <number>) of the current phase. In case data set <s> are neutron scattering data just a single scattering length <c> is needed, in case of X-rays, the scattering factor is given in exponential form. Finally if the last parameter is "internal", the internal scattering factor are used again.

1.20 show

show <**subcommand**>**,..**

This command allows to obtain various information about the model crystal, PDFFIT settings etc.
Valid subcommands are :

1.20.1 atom

show "atom",<**number**> **[,**<**end**>**]**
show "atom","all"
show "atom","last"

Information about the atom(s) such as position, thermal factors, atom type and occupancy are listed for the current phase (-> phase). If the optional second parameter <end> is given, all atoms in the range <number> to <end> are shown. If the second parameter is "all", all atoms in the crystal are shown. WARNING, this could last a while :-) If the second parameter is "last", only the last atom of the crystal is shown. This is identical to setting the second parameter to "n[1]", which contains the number of atoms in the crystal.

1.20.2 config

show "config"

This commands lists the limits of various arrays like the maximum number of phases, atoms, parameters etc. In case one of the limits is not sufficient, PDFFIT needs to be recompiled with adjusted limits.

1.20.3 const

show "const" [,<ip>]

This command displays all constraint equations and derivatives that contain parameter <ip>. If <ip> is omitted, all definitions are show, which can be quite a long list !

1.20.4 elem

show "elem" [,<ip>]

This command lists information about the elements within the crystal. If the parameters <ip> is omitted, the corresponding information for all phases is shown, otherwise just for the specified phase <ip>.

1.20.5 fit

show "fit"

This command outputs refinement information such as R-values and the correlation matrix.

1.20.6 metric

show "metric"

This command show the current lattice parameters, metric tensor as well as reciprocal lattice parameters and the reciprocal metric tensor for the currently active structural phase (-> phase).

1.20.7 par

show "par"

This command lists the current fit parameters on the screen.

1.20.8 pdf

show "pdf" [,<is>]

This command lists information about PDF related settings. If the parameters <is> is omitted, the corresponding information for all data sets is shown, otherwise just for the specified set <is>.

1.20.9 phase

show "phase"

This command displays all information about the currently active phase on the screen.

1.20.10 scat

show "scat",{ **"all"** | <**name**> | <**number**> }

This command allows the user to display the scattering lengths for all or just a specific atom within the currently selected phase (-> phase).

1.21 temp

temp {**"all"**|<**name**>|<**number**>},<**u11**>
temp {**"all"**|<**name**>|<**number**>},<**u11**>,<**u22**>,<**u33**>
temp {**"all"**|<**name**>|<**number**>},<**u11**>,<**u22**>,<**u33**>,<**u12**>,<**u13**>,<**u23**>

This command sets the anisotropic temperature factor U(ij) for "all" or the atoms specified either by <name> or atom type <number>. If just a single further parameter <u11> is given, an isotropic temperature factor is assumed. Otherwise just the diagonal or all elements of the tensor can be specified. Note that U(i,j)=<u(i)u(j)> and specifically for the diagonal elements U(ii)=<u(i)**2> !

1.22 urf

urf <**u**>

This command sets the URF (some German: Unterer Relaxations Faktor) for the fit. This value determines who 'fast' the fit will move to its minimum or how much the parameter values are changed in each cycle depending on the deviations. A small value (e.g. 0.1) might lead to a fast convergence but might also miss the minimum. A larger value (e.g. 100.0) will give a slow convergence which more certain finds the minimum, but might be caught in local minima rather than in the global one.
Understood ? Well just try different values until your fit converges nicely to the global minimum.

1.23 variables

Variables
PDFFIT allows the usage of variables. The general variables i[n], r[n] and res[n] are explained in the 'command language' section of the online help (-> command language). The variables marked with RO below are READONLY.

```
x[n]      : x-position of atom n of current phase (-> phase)
y[n]      : y-position of atom n of current phase (-> phase)
z[n]      : z-position of atom n of current phase (-> phase)
m[n]      : Number of atom type on site n (current phase)
u[6,n]    : Aniso. thermal factor U(ij) for site n (curr. phase)
o[n]      : Occupancy of site n (current phase)

lat[6]    : Lattice parameters (a,b,c,alpha,beta,gamma) f. phase
delt[n]   : Value of DELTA (sharpening !) for phase 'n'
rcut[n]   : Cutoff r-value for additional sharpening for phase 'n'
srat[n]   : Ratio of peak width below and above 'rcut' for phase 'n'
csca[n]   : Value of scaling factor for phase 'n'
rhoz[n]   : Value of number density RHO0 phase 'n' (RO)
```

```
qsig[s]   : Value of QSIGMA (resolution dampening) for data set 's'
dsca[s]   : Value of scaling factor for data set 's'
bave[s]   : Average scatt. length <b> data set 's' (current phase)

dx[n]     : Standard deviation of x-position
dy[n]     : Standard deviation of y-position
dz[n]     : Standard deviation of z-position
du[6,n]   : Standard deviation of u[6,n]  (RO)
do[n]     : Standard deviation of o[n]    (RO)
dlat[6]   : Standard deviation of lat[6]
ddelt[n]  : Standard deviation of delt[n] (RO)
dsrat[n]  : Standard deviation of srat[n] (RO)
dcsca[n]  : Standard deviation of csca[n] (RO)
drhoz[n]  : Standard deviation of rho0[n] (RO)
ddsca[s]  : Standard deviation of dsca[s] (RO)
dqsig[s]  : Standard deviation of qsig[s] (RO)

n[1]      : Total number of atoms of current phase (RO)
n[2]      : Number of different atoms of current phase (RO)
n[3]      : Number of atoms per unit cell (current phase) (RO)
n[4]      : Number of phases (RO)
n[5]      : Number of current phase (change with -> phase) (RO)
n[6]      : Number of loaded experimental PDFs (RO)
n[7]      : Number of possible parameters (RO)
n[8]      : Number of used refinement parameters (RO)

p[n]      : Value of fit parameter n
dp[n]     : Value of standard deviation for parameter n (RO)
pf[n]     : Refinement flag for parameter n, 1=refine 0=fixed
cl[n,m]   : Correlation matrix element n,m (RO)

rw[1]     : expected R-value (after refinement) (RO)
rw[2]     : achieved R-value (after refinement) (RO)
rw[3]     : achieved weighted R-value (after refinement) (RO)

np[s]     : Number of data points of data set 's' (RO)
pc[n,s]   : Value of calculated PDF point 'n' of data set 's'
po[n,s]   : Value of observed PDF point 'n' of data set 's'
pw[n,s]   : Weight for PDF point 'n' of data set 's'
delr[s]   : Value of DELTAR of data set 's'
qmax[s]   : Value of QMAX for data set 's'
rang[2,s] : Value of minimum (1) and maximum (2) R for refinement
rmin[s]   : Value of maximum R of data set 's'
rmax[s]   : Value of maximum R of data set 's'

fa[i,s,p] : Value of matrix A(i,s,p) (see Users Guide)
fb[i,s,p] : Value of matrix B(i,s,p)
fc[p,dp]  : Value of matrix C(p,dp)
```

1.24 xray

```
xray [<xq>]
```

This command sets the Q-value used to calculate the scattering length used in the PDF calculation. The default value is xq=0 which results in a weight corresponding to the number of electrons of the contributing atoms. Other settings could be the Q value of the first Bragg peak or the average Q value of the data set. Calling the command without parameters prints the current setting on the screen.

1.25 errors

The program has been written such that it should handle almost any typing error when giving commands and hopefully all errors that result from calculation with erroneous data. When an error is found an error message is displayed that should get you back on track. See the manual for a complete list of error messages.
Error messages concerning the command language are explained in the -> command language section of the online help. Application specific commands are described here and are grouped as follows:

```
APPL   Errors specific to PDFFIT structure functions
PDF    Errors specific to PDFFIT PDF calculation and fit func.
```

Each error message is displayed together with the corresponding category <cccc> and the error number <numb> in the form:
******CCCC****message **** numb ******

Type help error <cccc> <numb> to get an explanation for the error message and hints for possible steps to correct the situation.
In the default mode PDFFIT returns the standard prompt and you can continue the execution from this point. You can set the error status to "exit" by the ==>'set' command. In this case PDFFIT terminates if an error is detected. This option is useful to terminate a faulty sequence of commands when running PDFFIT in the batch mode of your operating system.

1.25.1 appl

This category lists error messages that are specific to structure related functions of PDFFIT.

Error : -2 : Improper limits for atom number
Either of the upper or lower limits used on the 'append' command is outside the range of atoms present in the crystal. Check whether the limits are both positive, the upper limit is higher or equal to the lower limit and whether both limits are less or equal to the number of atoms present in the crystal. The number of atoms in the crystal can be checked with the command: "eval n[1]".

Error : -3 : No atoms selected yet
The plot and waves can only be run for selected atoms. Use the 'select' command to select individual atom types or to select all atoms present in the crystal.

Error : -7 : Unknown space group symbol
The crystal file contains an unknown space group. Check the spelling of the space group symbol. The allowed space groups are all 230 space groups in the Int. Tables for Crystallography plus the space groups given for alternative settings and cell choices of the monoclinic space groups.

Error : -10 : Too many Atoms in crystal

The maximum number of atoms that can be stored in the structure is determined by the parameter NMAX in the file "crystal.inc". By inserting new atoms or by defining too large a crystal on the 'read' command, this number was exceeded. If necessary, change the value of the parameter NMAX and recompile the program.

Error : -12 : Number of points must be > zero
The value of the parameter given on the 'abs' or 'ord' command is less than one. This value represents the number of data points calculated along the respective direction. The value must be at least one or higher.

Error : -14 : Invalid space group & lattice constants
While reading a new cell the program checks the space group and the lattice constants for consistency. Either a space group was given that is not included in the program, or the lattice constants do not fulfill the constraints imposed by the space group. Check the space group symbol and the lattice constants given in the input file.

Error : -19 : Atom number outside limits
The number of the atom is either less than one, larger than the current number of atoms in the crystal or even larger than the maximum number of atoms allowed in your implementation. Check the value of the parameter(s) on the 'remove' and 'switch' commands or check the index of the variables "m", "x", "y" or "z". Check the number of atoms present in the crystal by the command: 'eval n[1]'.

Error : -20 : Unknown element, no Fourier calculated
An element was detected in the list of atoms for which there is no scattering curve available. The Fourier transform is not calculated at all. Check the name of all atoms present in the crystal using the 'asym' and 'chem' commands. If a charged ion was given, this valence might not be present in the list of scattering curves. Refer to Appendix b of the manual for a list of internally stored names. If the 'scat' and/or 'delf' commands were used, any name may be used. Check whether the commands were used properly.

Error : -21 : No element present, no Fourier calculated
There are no elements present at all in the crystal. The Fourier transform is not calculated at all. Most likely, the Fourier was called before a structure or unit cell was read, or an error occurred during the reading of the structure or unit cell.

Error : -26 : Too many different atoms in crystal
The maximum number of different atoms allowed in your implementation was exceeded. No more new types of atoms can be inserted into the structure. Check the chemistry of your crystal by the 'asym' and 'chem' commands. All atoms are considered different types that are chemically different, have different charge or a different temperature coefficient. If all types are needed, modify the parameter "maxscat" in the file "param.inc" and recompile the program. See chapter 9.1 of the manual for further information.

Error : -27 : No atom of this type present in crystal
An atom was selected for displacement by a wave or for plotting that does not exist within the crystal. Check the spelling of the atom name, and the chemistry of the crystal by the 'asym' and 'chem' commands.

Error : -28 : Input parameters must be > zero
This function/command requires non-negative parameters. Check the values of the parameters and the explanation for the function or command used for valid ranges of numerical input.

Error : -32 : Length of vector is zero

An attempt was made to calculate the angle between two vectors while one of them is of length zero. Check the parameters given on the 'bang' or 'rang' command for proper numbers.

Error : -35 : Volume of unit cell <= zero
The volume of the unit cell was calculated as zero or a negative value. Check the lattice parameters given in the input file. Are there any accidental "-" signs ? Do the angles form an impossible shape ?

Error : -37 : No filename defined yet
An attempt was made to write output to or read from a file without defining a file name. PDFFIT does not provide default names for the output of the 'plot' command or the input filenames. Check the 'outfile' command at sublevels 'output' and 'plot' or the 'content' command at sublevel 'microdomains'.

Error : -43 : Not enough parameter for filename format
An attempt was made to generate a file name from a string like "text%dtext" without supplying enough numerical parameters. Check that the spelling of the sting within " " is correct. Are there any unwanted %d combinations?. Check the number and type of parameters following the file name.

Error : -44 : Right quotation mark missing in format
An attempt was made to generate a file name from a string like "text%dtext" without supplying the right quotation mark. Check the line and provide the missing ".

Error : -45 : Too many atoms in environment
The indices of all atoms found are stored in the internal variable "res". More atoms were found that fit into the dimensions of "res". Restrict the search for the environment to a smaller shell or change the dimension of "res_para" in file "param.inc".

Error : -46 : Error reading title of structure
An error occurred while reading the title line of a structure or unit cell file. Check the file for any garbage.

Error : -47 : Error reading space group symbol
An error occured while reading the space group line of a structure or unit cell file. Check the file for any garbage.

Error : -48 : Error reading lattice constants
An error occured while reading the lattice constants of a structure or unit cell file. Check the file for any garbage or accidental letters.

Error : -49 : Error reading atom coordinates
An error occured while reading the atom coordinates of the atom listed. Check the file for any garbage. Is the line of the type Name x y z B Are there letters among the numerical values?

Error : -54 : Index outside limits
The value given is outside the proper limits allowed by this command. This usually means that an array element is outside the current dimension of an array, for example a correlation matrix or you are trying to include too many atoms in the crystal. Check the section on dimensions in the manual.

1.25.2 pdf

This category lists error messages that are specific PDF and refinement functions of PDFFIT.

Error : -1 : Invalid structure phase selected
The number of an invalid phase was given. One can only enter phase specific commands when the corresponding structure was actually read via 'read stru'.

Error : -2 : Number of phases outside limits
When trying to read another structural phase, the maximum number of phases was exceeded. You might need to adjust the parameter MAXPHA in 'config.inc' and recompile PDFFIT.

Error : -3 : Invalid occupancy specified
A site occupancy not in the range 0 to 1 was specified for an atomic site. Check your input.

Error : -4 : Maximum number of data sets exceeded
When reading a PDF data set, the maximum number of data sets was exceeded. You might need to adjust the parameter MAXDSET in the file 'config.inc' and recompile PDFFIT.

Error : -5 : Invalid radiation type selected
The only radiation types allowed as parameter for the 'read data' command are 'n' for neutrons and 'x' for X-rays. Check your input.

Error : -6 : Invalid data set specified
The number of an invalid data set was given. One can only enter data set specific commands when the corresponding data file was actually read via 'read data'.

Error : -7 : Invalid parameter constraint specified
The parameter constraint specified using 'par' is invalid. Either it contains no "=" or tries to assign a non refinable variable to a refinement parameter.

Error : -8 : Too many different parameters in constr.
You have entered a definition, that depends on too many different refinement parameters. You might need to adjust the variable MAXDPP in the file 'config.inc' and recompile PDFFIT.

Error : -9 : Inconsistency between NCELL and # atoms
When reading a structure file, the number of atoms actually found does not match the number of atoms expected from the parameters of the 'ncell' keyword. For 'ncell nx,ny,nz,n' you expect to read nx.ny.nz.n atoms.

Error : -10 : Invalid parameter selected
An invalid refinement parameter was selected. Use the command 'show config' to see what the maximum number of refinement parameters is you can use.

Error : -11 : Too many data points in experimental PDF
The PDF data file to be read contains too many data points. You might need to adjust the parameter MAXDAT in the file 'config.inc' and recompile PDFFIT.

Error : -12 : Cannot extend r-range for convolution
In order to calculate the PDF, the calculated function is convoluted with a SINC function with a width determined by Qmax. In oder to carry out the convolution a finite number of data points outside the refinement range are needed. You have to use a lower RMAX for the refinement or reprocess the PDF up to a higher R value.

Error : -13 : R-value out of range

You have specified a value of R that is outside the range for the specified data set. Check your input.

Error : -14 : Parameter index in p[i] out of range
You have used a parameter index 'i' that is out of range. Note that the maximum allowed value of 'i' is determined by the maximum number of parameters that can be used for a given set of data and structural phases.

Chapter 2

General commands

All commands consist of a command verb, optionally followed by one or more parameters. All parameters must be separated from one another by a comma ",". There is no predefined need for any specific sequence of commands.
Commands are case sensitive, all commands and alphabetic parameters MUST be typed in lower case letters.
Only the first four letters of any command verb are significant, all commands may be abbreviated even further. At least a single space is needed between the command verb and the first parameter. No comma is to precede the first parameter. The command verb itself MAY NOT be preceded by any blanks.
The word PROG in this part of the help section is to be replaced by the name of the program you are using (e.g. DISCUS).
Further help topics are:

2.1 Options

program [-debug] [-remote] [macro.mac]

All programs allow the following command line parameters. The flag "-debug" starts the program in debug mode. This is the same as using the command "set debug,on". The switch "-remote" starts the program in remote control mode. In this case the commands are send to the program through a socket. See file "remote.f" for an example how to remote control the applications from another program. Note that the program will not accept input from the keyboard when in remote control mode. All other command line arguments are interpreted as macro files and will be executed at startup.

2.2

#<comment>

Any line beginning with a "#" is regarded as comment.

2.3 @

@<filename> [<argument> ...]

Any list of valid commands can be written to an ASCII file and and indirectly by the command:

```
prompt > @<name>
```

The commands must start in the left most column of the file and are otherwise executed as typed. Macro files may call other macro files up to a level defined at installation. This is not a call in the sense of calling a function. All variables are identical at all levels of macro file nesting.
Macro files can be written by any editor on your system or be generated by the ==> 'learn' command. 'learn' starts to remember all the commands that follow and saves them into the file given on the 'learn' command. The learn sequence is terminated by the 'lend' command. The default extension is ".mac"
Optionally arguments can be listed on the command line. These arguments will replace the formal parameters inside the macro. The formal parameters must be given as "$1", "$2" ... The string <argument> will replace the string "$1". "$1" is the first argument on the command line, "$2" the second and so on. If there are not enough command line arguments, an error message is displayed. The parameter "$0" contains the number of parameters listed on the line that called the macro. If no parameters were given this value will be zero.

2.4 =

<**variable**> = <**expression**>

The expression on the right of the equal sign is evaluated and its result stored in variable <variable>.

2.5 input

Input editing functions

If the program was compiled with -DREADLINE, the following basic editing functions are available at the program prompt:

```
^A               : moves to the beginning of the line
^B               : moves back a single character
^E               : moves to the end of the line
^F               : moves forward a single character
^K               : kills from current position to the end of line
^P or arrow up   : moves back through history
^N or arrow down : moves forward through history
^H and DEL       : delete the previous character
^D               : deletes the current character
^L/^R            : redraw line in case it gets trashed
^U               : kills the entire line
^W               : kills last word
```

Furthermore you can move within the line using the arrow keys.
NOTE:

If you redirect the input for executed PROG using 'prog < infile' you MUST use the command 'set prompt,off' or 'set prompt,redirect' in the first line to avoid that the program 'hangs' at the end of the file. (-> set prompt)

2.6 break

break <**levels**>

The 'break' command stops the execution of the current block structure and advances to the next command following the block structure. With <levels> equal to 1 only the current block structure is interrupted, with any higher number the <levels> innermost block structures are interrupted. The 'break' command can be used only inside a block structure.

2.7 cd

cd [<**directory**>**]**

This command allows one to change the current working directory (may not be available everywhere). If the command is called with no parameters, the current working directory is shown.

2.8 continue

continue ["prog"]

This command is effective only while PROG is in the interupted macro mode or inside interrupted do-loop or if-statements, which serves as a debug mode for lengthy macros or block structures
Make sure you have returned to the same sub menue before you continue!

Without parameters PROG resumes the execution of a macro or block structure in the line following the 'stop' command. If you had started another macro while debugging a macro, and this new macro contained a 'stop' command as well, the 'continue' command will run the remaining lines in the new macro and then stop again at the position of the 'stop' command in the outer macro.
By providing the 'prog' parameter, PROG immediately interupts all macros and returns to the normal prompt.

2.9 do

Loops can be programmed with the 'do' command. The command may take the following forms:

```
a)
do <variable> = <start>,<end> [,<increment>]
  <commands to be repeated>
enddo

b)
do while (<logical expression>)
  <commands to be repeated>
enddo

c)
do
  <commands to be repeated>
enddo until (<logical expression>)
```

Type a) loops may contain constants or arithmetic expressions for <start>, <end>, and <increment>. The internal type of the variables is real. The loop counter is evaluated from (<end> - <start>)/<increment> =1 . If this is negative, the loop is not executed at all.
Type b) loops are executed while <logical expression> is true. Thus, they may not be executed at all.
Type c) loops, however, are always executed once, and repeated until <logical expression> is true.
If an error occurres during execution of the loop, the loop is interrupted.

2.10 echo

echo [<string>]
echo ["string%dstring",<integer expression>]
echo ["string%Dstring",<integer expression>]
echo ["string%fstring",<float expression>]
echo ["string%Fstring",<float expression>]

The string <string> is echoed to the default output device as typed. This command serves as a marker inside long macro files. It gives the user a chance to include easy to find messages in order to follow lengthy or nested structures.
The alternative command format allows to echo formatted strings to the screen. Each "%d" is replaced by the value of the corresponding parameter. The sequence of "%d" corresponds to the sequence of the integer parameters, "%f" stands for parameters of the type real.
The value of a numerical expression between the "%" and the "d" determines the width of the integer field that is printed. In the case of a floating variable two expressions seperated by a decimal point specify the width and the number of decimal digits that are printed.
The capital forms "%D" and "%F" will fill leading spaces with zeros.
Examples echo ">%3d<",44 produces : > 44< echo ">%1+2d<",44 produces : > 44< echo ">%3D<",44 produces : >044< echo ">%5.1f<",44.1 produces : > 44.1< echo ">%2**2+1.1f<",44.1 produces : > 44.1<

2.11 eval

eval <expr>

Evaluates the expression and displays the result. The result is not stored, this command is for interactive display only.

2.12 exit

exit

Terminates the program and gets you back to your shell.

2.13 expressions

Arithmetic expressions can be evaluated in a FORTRAN style. Five basic operators are defined:

```
"+"  Addition
"-"  Subtraction
"*"  Multiplication
```

```
"/"  Division
"**" Exponentiation
```

The usual hierarchy of operators holds. The parts of the expression can be grouped with parentheses "(",")" in order to circumvent the standard hierarchy
Several intrinsic functions have been defined, see "functions" for a full listing.
Examples of valid expressions are:

```
1
1+3*(sin(3.14*r[1]))
x[1]*0.155
asind(0.5)
```

2.14 filenames

Usually, file names are understood as typed, including capital letters. Unix operating systems distinguish between upper and lower case typing !
Additionally (integer) numerical input can be written into the filename. The syntax for this is: "string%dstring",<integer expression>
The file format MUST be enclosed in quotation marks. The position of each integer must be characterised by a "%d". The sequence of strings and "%d"'s can be mixed at will. The corresponding integer expressions must follow after the closing quotation mark. If the command line requires further parameter (like "addfile" for example) they must be given after the format-parameters. The interpretation of the "%d"'s follows the C syntax. Up to 10 numbers can be written into a filename.
Examples:

```
1)
i[5]=1
outfile a1.1
outfile "a%d.%d",1,1
outfile "a%d.%d",4-3,i[5]
```

All the above examples will result in the file name "a1.1".

```
2)
do i[1]=1,11
...
outfile "data%d.calc",i[1]
...
enddo
```

The output is written to the files "data1.calc" through "data11.calc"

2.15 fclose

fclose {<**number**>|**"all"**}

This command closes a file that was opened with 'fopen <number>' or closes all open files. If this command is not used before exiting the program, data might be lost !

2.16 fend

fend <**number**>,{**'continue'**|**'error'**}

This command determines the reaction to an unexpected end of file while reading data from input file <number>. If the parameter is set to "continue", the program will set the variable res[0] to -1 and continue the macro. If you repeat the ==> 'fget' command, the program will again set res[0] to -1 and will not result in an error. In order to catch and EOF, you have to evaluate the value of res[0] each time the 'fget' command is executed.
If the parameter is set to "error", the program will stop reading data from the input file and terminate the macro with an error message. The value of res[0] remains undefined.
The default condition at program start is "error"

2.17 fexist

fexist <**file**>

This command checks the existance of the specified file <file>. The result is written on the screen and returned via the res[] variables. If the file exists, res[1] is 1, otherwise it is 0. The variable res[0] returns the number of parameters, here 1.

2.18 fformat

fformat <**nc**>,<**format**>

This command allows one to specify a FORTRAN style format string <format> to be used for column <nc>. The default is free format, which can be selected using the character * as format string. If the command is called with no parameters, the current settings are displayed on the screen. Note that an unsuitable format might result in a conversion error and *** being written to the file !
Example: fform 1,F7.3

2.19 fget

fget <**number**>,<**p1**>,<**p2**>,..

This command allows the user to read data from a file that had been opened with 'fopen <number>. If no parameters are given, a line is read, yet its content is ignored and the line gets skipped. Otherwise the read numbers will overwrite the contents of the specified variables.
Note that a 'fget' command that does not run into an unexpected end of file sets the value of res[0] to zero!
Example: fget 1, r[2],i[2]

2.20 fopen

fopen <**number**>,<**file**> **[,**{**"append"** | **"overwrite"**}**]**

This command allows the user to open a file for reading and writing using the commands 'fget' and 'fput'. iThe first argument is the number of the io_stream. You can open several files at once, the exact value depends on the value of the

variable MAC_MAX_IO in file "macro.inc". The second argument is the file name. The default is that existing files will be overwritten if 'fput' is used. Alternatively one can append data to a file by specifying the optional parameter "append".

2.21 fput

fput <number>,<p1>,<p2>, ..

This command allows one to write data to the file that had been opened by 'fopen <number>'. The parameters <pi> can either be variables and expressions or simple text enclosed in single quotes. If no parameters are given, an empty line is written.

```
Examples:  fput 1, i[1],sqrt(1.0+i[1]*0.01)
           fput 1, 'Current value of i[1] : ',i[1]
```

2.22 fsub

fsub <number>,[<left>,<right>]
The command allows you to limit the string from which 'fget' reads the data from file <number>. Data will only be read collumns <left> to <right>. If both parameters are missing, the full input string is read. If the parameter <right> is set to "-1", the string is read from <left> all the way to the end of the input string, independent of its length.
The default values at program start are 1,-1 for all input channels.

```
Examples:
  Input line: "A text string 20.0  30.0"
  fsub 14,24
  fget r[1],r[2]
```

2.23 functions

The following intrinsic functions exist:

```
asin(<arg>)                    !
acos(<arg>)                    !
atan(<arg>)                    !
asind(<arg>)                   ! Result in degrees
acosd(<arg>)                   ! Result in degrees
atand(<arg>)                   ! Result in degrees

sin(<arg>)                     !
cos(<arg>)                     !
tan(<arg>)                     !
sind(<arg>)                    ! Argument in degrees
cosd(<arg>)                    ! Argument in degrees
tand(<arg>)                    ! Argument in degrees

sinh(<arg>)                    ! Hyperbolic functions
cosh(<arg>)                    !
tanh(<arg>)                    !
```

```
sqrt(<arg>)                    ! Square root of <arg>
exp(<arg>)                     ! exponential (base e)
ln(<arg>)                      ! natural logarithm of <arg>
abs(<arg>)                     ! Absolute value of <arg>

mod(<arg1>,<arg2>)             ! Modulo <arg1> of <arg2>, real arguments
max(<arg1>,<arg2>)             ! Maximum of <arg1> and <arg2>
min(<arg1>,<arg2>)             ! Minimum of <arg1> and <arg2>

int(<arg>)                     ! Convert argument to integer
nint(<arg>)                    ! Convert argument to nearest integer
frac(<arg>)                    ! Returns fractional part of <arg>

ran(<arg>)                     ! Returns uniformly distributed pseudo
                                 random value 0<= r < 1.
```

The arguments to any of these functions are any arithmetic expression.

2.24 help

help [<command> [, <subcommand>]]

The 'help' command is used to display on-line help messages. They are short notes on the command <command>. The command may be abbreviated. If the abbreviation is not unique, only the first help topic that matches the command is listed.
The first line of the help text gives the syntax of the command that is explained in the following lines. For a few commands the syntax line is repeated for different set of possible parameters.
After the text is displayed, you are in the HELP sublevel of PROG and there are the following commands possible:

```
<command> : Display help for <command> of current help level.
".."      : Go up one help level.
"?"       : Prints list of help entries of the current level.
<RETURN>  : Exit help sublevel.
```

2.25 if

The if-control structure takes the following form:

```
if ( <logical expression> ) then
  <conditional commands>
[elseif ( <logical expression>) then
  <conditional commands>]
[else
  <conditional commands>]
endif
```

The logical expressions may contain numerical comparisons with syntax:

```
<arithmetic expression> <operator> <arithmetic comparison>
```

The following operators are allowed:

```
.lt.    ! less than
.le.    ! less or equal
.gt.    ! greater than
.ge.    ! greater or equal
.eq.    ! equal
.ne.    ! not equal
```

The logical expressions may also contain string comparisons with syntax:

```
'<string1>' <operator> '<string2>'
```

Both strings MUST be enclosed by single apostrophes '. The operators are the same as those for the numeric expressions, lexical comparisons are used to evaluate the comparisons "less" and "greater".
Logical expressions can be combined by logical operators:

```
.not.   ! negation of the following expression
.and.   ! logical and
.xor.   ! logical exclusive or
.or.    ! logical or
```

Logical operations may be nested and grouped by brackets "(" and ")".

2.26 learn

learn [<name>]

Starts a learn sequence. All following commands are saved as typed in file <name>. defaults to "<prog>.mac". ==> lend finishes the learn sequence.

2.27 lend

lend

Finishes the learn sequence started by ==> learn.

2.28 set

set <command>,..

This command allows to alter various program independent setting. Allowed values for <command> are:

2.28.1 prompt

set prompt, {**"on"**|**"off"**|**"redirect"**}**,[**{**"on"**|**"off"**|**"file"**}**],["save"]**
set prompt, "old"

First parameter sets the status of the PROG prompt. The default is "on", i.e. PROG prompts for the next command by writing "discus > " (in case you run DISCUS ..). You can turn this prompt off. This is useful, if you are running a long macro and do not want to get all the prompts written into the output. By using this option you can considerably shorten the output written by PROG into a redirected log file. If you are using PROG on a UNIX platform, you can start the program with redirected input by the command:

```
"prog < inputfile"
```

By default, PROG will write the prompt "discus >" into the outputfile, expecting a RETURN from the keyboard. Very long lines in the output file will result. To avoid this situation insert the line "set output,redirect" as first line in the inputfile to force discus to echo the lines from file inputfile.
This second parameter allows the user to assign where the text output of the program should go: "on" prints on the screen, "off" will result in no output and "file" will save the output to a file progname.log (e.g. discus.log in DISCUS). Note that the output of the commands 'echo' and 'eval' will always appear on the screen. The last parameter allows on to save the current prompt and output settings.
The parameter "old" allow the user to and restore the setting of the prompt and output. This can e.g. be used to turn the prompt off in a macro and then restore the original setting after the macro is executed.

2.28.2 error

set error , {**"cont"** | **"exit"** }

Sets the error status.

```
"cont"  PROG returns the normal prompt after the display of the error
        message. You can continue the input of commands.
"exit"  PROG terminates after the display of the error message.
        This option is useful if you run PROG in the batch mode of your
        operating system. Instead of continuing with a faulty calculation
        PROG stops and you can immediately check the error.
```

2.28.3 debug

set debug, {**"on"** | **"off"**}

This command allws the user to enable various DEBUG outputs ...

2.29 stop

stop

This command is active only while reading from a macro file or in interactive mode inside a block structure (do-loops and/or if's).
The current macro file is interupted and you can type commands as in the normal input mode. You can use the whole range of PROG commands, including the '@' macro command. The 'stop' commands provides a convenient mode

to debug a macro by setting a break point at which you can check the value of variables or set new values, run an additional macro etc.
To continue execution of the macro or to continue with the normal PROG mode, use the ==> 'continue' command.
If included in a block structure statement (do-loops and/or if's) in both, macro and interactive mode, the program continues reading all statements that belong to the block structure. During execution of the structure, PROG interupts this execution if it encounters a 'stop' command. You can issue any PROG command except further do or if commands. To continue execution of the structure or to continue with the normal PROG mode, use the ==> 'continue' command.

2.30 system

system <**com**>
system ["string%dstring",<**integer expression**>**]**
system ["string%fstring",<**float expression**>**]**

Executes the single shell command <com>. If the command string is enclosed in "", you can place integer and real format specifiers "%d" "%f" which are then substituted by the corresponding values.
Example i[0]=10 system "ls %d.*"
This would list all files called 10.*

2.31 wait

wait [{**"return"** | **"input" [,**<**prompt**>**]** }**]**

This command waits for user input. Without a parameter or with "return", the program waits for a simple <RETURN>. If the first parameter is "input", the program expects the user to enter one or more real numbers or expressions. The optional <prompt> can be used to ask the user to input the expressions. This is especially helpful if the prompt has been turned of by ==> set prompt,off. The number of expressions entered by the user is stored in the variable res[0] and the results of the expressions in res[i].
This command allows to write interactive macros, demo macros and tutorials.

2.32 errors

The program has been written such that it should handle almost any typing error when giving commands and hopefully all errors that result from calculation with erroneous data. When an error is found an error message is displayed that should get you back on track. See the manual for a complete list of error messages. In this part we refer to the program you are using as DISCUS for convenience.
The error messages concerning the use of the command language are grouped in the following categories:

```
COMM   Command language errors
FORT   Fortran interpreter errors
I/O    Errors regarding input/output
MACR   Errors related to macros
MATH   General mathematical errors
```

Each error message is displayed together with the corresponding category <cccc> and the error number <numb> in the form:

******CCCC****message **** numb ******

In the default mode DISCUS returns the standard prompt and you can continue the execution from this point. You can set the error status to "exit" by the ==>'set' command. In this case DISCUS terminates if an error is detected. This option is useful to terminate a faulty sequence of commands when running DISCUS in the batch mode of your operating system.

2.32.1 comm

Command language errors These messages describe illegal usage of the command language, such as unknown commands, improper numbers of parameters.

Error : -1 : DISCUS directory not defined
The environment variable DISCUS_DIR was not defined. Check the chapter on installation for your platform for the appropriate definition.

Error : -5 : Error in operating system command
The operating system/shell returned an error message. Check the appropriate system manuals for details.

Error : -6 : Missing or wrong parameters for command
Either the command needs more parameters than were provided, or the parameters are incorrect. Check the number and type of parameters. Is the sequence of numerical and character parameters correct?

Error : -8 : Unknown command
The command interpreter read an unknown command. Check the spelling of the command or check, whether this command is allowed at the current sublevel.

Error : -11 : Error in subroutine
More or less a system error message, ignore this message.

Error : -17 : Too many parameters
More parameters have been provided than are required by the command. Check the number, and type of parameters supplied, or the occurrence of additional ','.

2.32.2 fort

Fortran interpreter errors These messages describe erroneous mathematical calculations and improper usage of control structures (do,if, ...).

Error : -1 : Nonnumerical Parameters in expression
The interpreter found a nonnumerical string where a number is expected. If an intrinsic function or a variable was intended, check for missing parentheses.

Error : -2 : Unknown Variable
The expression contains a reference to an unknown variable. Check the spelling of the variable. Chapter 3.7.1 of the manual and the help entry "variables" contains a list of allowed variables. Check whether the variable is a read-only variable and was used on the left side of an expression. Some of the variables associated with microdomains are read-only depending of the circumstances!

Error : -3 : Unknown intrinsic function
The expression contains a reference to an unknown intrinsic function. Check the spelling of the function. Chapter 3.7.4 of the general part in the manual and the help entry "functions' contain a complete list of the allowed intrinsic functions.

Error : -4 : Division by zero'
An attempt was made to divide by zero. Check the value of the argument and correct the algorithm that calculates the argument.

Error : -5 : Square root of negative number
An attempt was made to calculate the square root of a negative argument. Check the value of the argument and correct the algorithm that calculates the argument.

Error : -6 : Missing or wrong Parameters for command
Either the function or variable referenced needs more parameters than were provided, or the parameters are incorrect. Check the number and type of parameters. Is the sequence of numerical and character parameters correct?

Error : -7 : Argument for asin,acos greater 1
An attempt was made to calculate asin or acos with an argument greater than 1. Check the value of the argument and correct the algorithm that calculates the argument.

Error : -8 : Index outside array limits
The index supplied for the variable is outside the limits of this variable. Check the general part for the dimensions of the variables.

Error : -9 : Number of brackets is not matching
The number of opening and closing brackets "[" and "]" does not match or is illegally nested with parentheses "(", ")" or other operators. Check the string used in the expression and correct it following the FORTRAN rules.

Error : -10 : Index for array element is missing
You have used a string like "i[]", where the opening and closing brackets do not contain any expression. Check the string used in the expression and correct it following the FORTRAN rules.

Error : -11 : Number of parentheses is not matching
The number of opening and closing parentheses "(" and ")" does not match or is illegally nested with brackets "[", "]" or other operators. Check the string used in the expression and correct it following the FORTRAN rules.

Error : -12 : Expression between () is missing
You have used a string like "()", where the opening and closing parentheses "(" and ")" do not contain any expression. Check the string used in the expression and correct it following the FORTRAN rules.

Error : -13 : Wrong number of indizes for array
The number of indices given for the entered parameter is wrong. Check the help entry 'variables' for the proper number of indices.

Error : -14 : Index of DO-loop counter is missing
Here the index for the loop counter of a do-loop is missing. Check the online help for the correct syntax of such loops.

Error : -15 : Too many commands
The program stores all commands within a control block in an array. The maximum number of commands that can be stored in this array is given by the parameter MAXCOM in file "doloop.inc". The macro or run used more commands than currently allowed by this parameter. Rewrite the macro or list of commands such that less commands are sufficient, or change the value of the parameter and recompile the program.

Error : -16 : Too deeply leveled (do,if) construction
The program stores all commands within a control block in an array. The maximum number of levels for this array is given by the parameter MAXLEV in file "doloop.inc". The macro or run used more levels than currently allowed by this parameter. Rewrite the macro or list of commands such that less levels are sufficient, or change the value of the parameter and recompile the program.

Error : -18 : Unresolvable condition
An error occurred while trying to calculate the value of an arithmetic or logical expression. Check that there is no illegal operation /(division by zero .../ no typing errors, all parentheses are properly matched.

Error : -19 : Illegal nesting of control commands
Do loops and/or if constructions have been nested with overlapping segments, missing enddo or endif statements or similar causes. Check for spelling errors on the control statements, and that each control statement is properly terminated by a corresponding enddo or endif statement that is not enclosed within another control block.

Error : -20 : Illigal argument for ln(x) function
The argument for the ln must be positiv, larger than zero. Check the value of the argument or the value of the expression that serves as argument

Error : -28 : Too deeply leveled break command
Illegal use of the break command. The parameter on the break command signals how many block structure levels are to be exited. Check the value of this parameter with regard to the nesting of do-loops and if-blocks.

2.32.3 i/o

Errors related to input / output An error occured while attempting to read/write from a file

Error : -1 : File does not exist
DISCUS could not find the file. Check the spelling and the path.

Error : -2 : Error opening file
DISCUS could not open a file. The file might be in use by another process.

Error : -3 : Error reading file
An error occurred while DISCUS was reading a file. Check whether the contents of the file is correct.

Error : -4 : File already exists
An attempt was made to overwrite an existing file. Rename or delete the file in question.

Error : -5 : No such entry in online help
You have tried to obtain help for a string that does not have a matching entry in the help file.
Check the spelling of the string. Are you at the right sublevel? Use the '?' command to get a listing of available help entries.

Error : -6 : Unexpected end of file
DISCUS has encountered the end of a file, but is still expecting data. Check the file(s) involved, to see whether the data are complete or whether erroneous data are present.

Error : -7 : Learning sequence already in progress
You have tried to start a learning sequence by ==>'learn' without closing the active learning sequence. Close the current learning sequence by ==> 'lend' before starting to record a new macro.

Error : -8 : Nothing learned - no macro written
You did not type any commands since the ==>'learn' command. No commands are written to the macro file. You need to give at least one command before closing a learn sequence.

Error : -9 : Error reading user input
An error occured while reading the last input. Does the string contain any characters where a number is expected, or any control or escape sequences.

Error : -10 : IO stream already open
The command 'fopen' was issued while there was already a file open. Close the currently open file with 'fclose'.

Error : -11 : No IO stream open to close
The command 'fclose' was issued, but there is no open file.

Error : -12 : Error writing to file
An error occured when reading a file with 'fget'. Check the file for nonnumerical values and check that the number of columns is equal or larger than the number of arguments of 'fget'.

2.32.4 macro

Errors related to macro These messages describe situations that result from missing macrofiles, missing macro parameters ...

Error : -1 : Too many macro parameters given
The number of parameters given on the macro command line is higher than allowed in your installation. The maximum number of parameters allowed is defined by the parameter MAC_MAX_PARA in the file macro.inc. Check the macro command line for any additional "," or rewrite the macro to use less parameters. If necessary adjust the value of the parameter MAC_MAX_PARA and recompile the program.

Error : -12 : Macro not found
The file given on the @<name> command does not exist. Check the spelling of <name> and the path.

Error : -13 : Macro filename is missing on the command line
The command '@' to execute a macro was called without any macro file name. The file name must start immediatly after the "@". Check the '@' command for completeness and blanks after the "@".

Error : -35 : Too deeply leveled macros
The maximum level at which macros may be nested is defined in the file macro.inc in the parameter MAC_MAX_LEVEL. Check the nesting of macro file for the level of nesting or possible recursive nesting without proper termination. Rewrite the macros to use less nesting, or change the value of the parameter and recompile the program.

Error : -36 : Unexpected EOF in macro file
When DISCUS finds a '@' command inside a macro, it stores the current macro name, the line number inside the current macro and closes the current macro file. After completion of the new macro, the previous macro is read again up to the position stored. The error message is displayed when an end of file is found before the position is reached. Check whether the macro file was damaged, or accidentally deleted during execution of the nested macro.

Error : -41 : Not enough macro parameters given
DISCUS read a parameter number inside a macro file that is higher than the number of parameters given on the command line of the macro. Check the parameters inside the macro for correct numbering and spelling. Check the number of parameters supplied on the command and check whether any "," is missing between parameters.

2.32.5 math

Errors from general usage of mathematical treatments

Singular Matrix

DISCUS/KUPLOT tried to calculate the inverse of a matrix that is singular. Reading a cell file: Are your lattice constants in error, i.e. a length of zero, an angle of 0 or 180 degrees ? Microdomains Is the matrix of the base vectors correct ?