technical memorandum Daresbury Laboratory

DL/SCI/TM100T

THE DL_POLY MANUAL

by

T.R. FORESTER and W. SMITH, SERC Daresbury Laboratory

March, 1994

94/175

Science and Engineering Research Council
DARESBURY LABORATORY
Daresbury, Warrington WA4 4AD

ISSN 0144-5677

DL_POLY

CCP5

Molecular Dynamics

DL_POLY USER'S MANUAL

T.R. Forester and W. Smith,
S.E.R.C. Daresbury Laboratory,
Daresbury,
Warrington WA4 4AD,
England

Version I September 1993

Chapter 1

Introduction

1.1 The DL_POLY Package

DL_POLY (Version I) is a package of subroutines, programs and data files, designed to facilitate the molecular dynamics simulation of macromolecules, polymers, ionic systems and solutions on a distributed memory parallel computer. It was written in the eighteen months preceeding September 1993 when the first release appeared. The principal authors at that stage were Bill Smith and Tim Forester at Daresbury Laboratory. For want of a more imaginative acronym the package was named DL_POLY. The entire DL_POLY project was given a major boost by the provision of a grant in January 1993 from the SERC's Computational Science Initiative. This should ensure that DL_POLY will be developed into a significant package over the next two years.

There were many reasons for writing the package. The first reason being a real need to exploit parallel computing for molecular simulation in the UK. There were several worthy packages available to the academic community including GROMOS, AMBER and XPLOR, but none of these were designed (as far as we were able to tell) with parallel computers in mind. Though it is undoubtedly possible to convert these programs into parallel forms, we decided that the effort involved could be better spent devising a new code which had parallelism built into it from the start. This we thought would allow us to adapt the code to new architectures with greater ease. The second reason was logistical; at this laboratory, and generally within the CCP5 community, there was a considerable repository of expertise in parallel molecular dynamics stretching back several years. We therefore sought to tap into this expertise to produce the new package. The third reason was that we thought it would be valuable to produce a package that was entirely free from commercial constraints, by which we mean the source code would be available to academic institutions, for adaptation and extension. There would be no restriction preventing users from examining (and verifying!) the source code, so the dreaded "black box" syndrome would be absent.

Of course, in the period of time available to us before the first release we have not been able to do everything. It is true that the first release contains only a fraction of the capabilities of the aforementioned packages, but we are hopeful that the package will be favourably received. Since this is very much a project endorsed by CCP5, which serves the UK academic community, we are also hopeful that our users will be inclined to give us any extensions of the package they devise and so accelerate its development. As it stands, the package has some nice features we think, while it is designed for a distributed memory parallel machine (particularly the Intel iPSC/860 at Daresbury) we have taken care to ensure that it can, with minimum modification, be run on the ever popular workstations that are now a feature of most simulation laboratories. This ought to encourage novices to take up parallel computing!

Users are reminded that we are interested in hearing what other features could be usefully incorporated. We obviously have ideas of our own and CCP5 has a formed a small group to oversee such developments, but other input would be welcome nevertheless. We also request that our users respect the integrity of DL_POLY source and not pass it on to third parties. We desire that all users of the package register with us, not least because we need to keep everyone abreast of new developments and discovered bugs.

In the following section we outline the capability of DL_POLY as briefly as possible. Some of the features listed are targets for later releases and are not available under Release I. We have indicated these. This is followed by a description of the DL_POLY

directory structure and how to obtain the source code from Daresbury Laboratory.

1.2 Functionality

The following is a list of the features DL_POLY incorporates, or will incorporate in the future. Those not yet available are indicated with a bullet symbol (•). It is worth reminding users that DL_POLY represents a *package* rather than a single program, so it is up to the users to piece together a program with the desired functionality. We will however, supply some ready made examples in the distributed source.

1.2.1 Molecular Systems

DL_POLY will simulate the following molecular species:

1. Simple atomic systems and mixtures e.g. Ne, Ar, Kr, etc.
2. Simple unpolarisable point ions e.g. NaCl, KCl, MgO, etc.
3. Simple rigid molecules e.g. CCl_4, SF_6, Benzene, etc.
4. Rigid molecular ions with point charges e.g. KNO_3, $(NH_4)_2SO_4$, H_2O etc
5. Polymers with rigid bonds e.g. C_nH_{2n+2}
6. Polymers with rigid bonds and point charges e.g. proteins
7. Macromolecules and biological systems
8. Molecules with flexible bonds
9. Mixtures of all the above

1.2.2 Force Field

The DL_POLY default force field is the GROMOS force field which includes:

1. All common forms of non-bonded atom-atom potential.
2. Atom-atom (site-site) Coulombic potential.
3. Valence angle potentials
4. Dihedral angle potentials
5. Improper dihedral angle potentials

Release I also includes the AMBER forcefield. It is intended that DL_POLY will eventually encompass all commonly available force fields.

1.2.3 Boundary Conditions

DL_POLY will permit the following boundary conditions;

1. None. e.g. isolated polymer in space.
2. Cubic periodic boundaries.
3. Orthorhombic periodic boundaries.
4. Parallelepiped periodic boundaries.

5. Truncated octahedral periodic boundaries.
6. Rhombic dodecahedral periodic boundaries.
7. Slab (x,y periodic, z nonperiodic).

1.2.4 Algorithms

1.2.4.1 Parallel Algorithms

The following parallel MD strategies are available:

1. Replicated data ($\mathcal{O}(N^2)$).
2. Systolic loop SLS-G ($\mathcal{O}(N^2)$) •
3. Link-cells (domain decomosition) ($\mathcal{O}(N)$) •

Where (1) and (3) are also suitable for serial computers.

1.2.4.2 Molecular Dynamics Algorithms

The following MD algorithms are available:

1. Verlet leapfrog (NVE).
2. Gear predictor corrector (NVE).
3. Andersen's NPT algorithm with Gear or Verlet •
4. Nosé-Hoover (NVT) with Verlet.
5. Evans Gaussian Thermostat with Verlet.
6. Berendsens' thermostat with Verlet.
7. Multiple timestep algorithm.

1.3 Programming Style

The programming style of DL_POLY is intended to be as uniform as possible. The following stylistic rules apply throughout. Potential contributors of code are requested to note the stylistic convention.

1.3.1 Programming Language

The programming language for DL_POLY is FORTRAN 77. C routines are permissible provided they are FORTRAN callable.

1.3.2 Target Computer

DL_POLY is targeted towards the Intel iPSC/860 parallel computer. However, versions of the program are to be available for other parallel and serial computers. For this reason all machine specific calls are located in dedicated FORTRAN routines, to facilitate substitution by appropriate alternatives.

1.3.3 Internal Data Transfer

There are to be NO COMMON blocks. All nonlocal and large local arrays to be passed as subroutine argument lists. It follows that there should be no block data.

1.3.4 Internal Documentation

All subroutines to be supplied with a header block of FORTRAN comment records giving:

1. The name of the author
2. The version number (if appropriate)
3. A brief description of the function of the subroutine

1.3.5 Subroutine/Function Calling Sequences

The variables in the subroutine argument list should be specified in the order:

1. logical
2. character and character arrays
3. integer
4. real
5. logical arrays
6. integer arrays
7. real arrays

1.3.6 FORTRAN Parameters

All parameters defined by the FORTRAN parameter statements should be specified in the include file: dl_params.inc. which is to be included at compilation time in all subroutines requiring the parameters. All parameters specified in dl_params.inc must be described by one or more comment cards.

1.3.7 Arithmetic Precision

All real variables and parameters must be specified in real*8. i.e. 64-bit precision must be used for all real computations.

1.3.8 Units

Internally all DL_POLY subroutines and functions assume the use of the following defined *molecular units*:

1. The unit of time (t_u) is 1×10^{-12} seconds.
2. The unit of length (ℓ_u) is 1×10^{-10} metres.
3. The unit of mass (m_u) is $1.6605402 \times 10^{-27}$ kilograms.
4. The unit of charge (q_u) is $1.60217733 \times 10^{-19}$ coulombs.

5. The unit of energy ($E_o = m_o(\ell_o/t_o)^2$) is $1.6605402 \times 10^{-23}$.

In addition the following conversion factors are used:
The coulombic conversion factor (γ_o) is:

$$\frac{1}{E_o}\left[\frac{q_o^2}{4\pi\epsilon_o\ell_o}\right] = 138935.4835$$

such that:

$$U_{MKS} = E_o\gamma_o U_{Internal}$$

Where U represents the configuration energy.

1.3.9 Error Messages

All detected errors detected by DL_POLY during run time will initiate a call to a subroutine ERROR, which will print an error message in the standard output file and, if necessary, terminate the program. All terminations of the program will be global (i.e. every node in the parallel computer will be informed of the termination condition and stop executing.

1.4 The DL_POLY Directory Structure

The entire DL_POLY package is stored in a Unix directory structure. The topmost directory is named dl_poly. Beneath this directory are several sub-directories named: source; utility; data; execute; and build. Briefly, the content of each sub-directory is as follows:

sub-directory	contents
source	primary subroutines for the DL_POLY package
utility	all utility subroutines, programs and example data
data	example input and output files for DL_POLY
execute	the DL_POLY run-time directory
build	makefiles to assemble and compile DL_POLY programs

A more detailed description of each sub-directory follows.

1.4.1 The *source* Sub-directory

In this sub-directory all the essential source code for DL_POLY, excluding the utility programs and subroutines, are stored. In keeping with the 'package' concept of DL_POLY, it does not contain any complete programs; these are assembled at compile time using an appropriate makefile. The subroutines present in this sub-directory are documented in Chapter 2 of this document.

1.4.2 The *utility* Sub-directory

This sub-directory stores all the utility subroutines, functions and programs in the DL_POLY package, together with examples of data. The various routines in this sub-directory are documented in Chapter 6 of this document. Users who devise their own utilities are advised to store them in the utility sub-directory.

1.4.3 The *data* Sub-directory

This sub-directory contains examples of input and output files for testing the released version of DL_POLY. The examples of input data are copied into the 'execute' sub-directory when a program is being tested.

1.4.4 The *execute* Sub-directory

In the supplied version of DL_POLY, this sub-directory contains only a few macros for copying and storing data from and to the data sub-directory and for submitting programs for execution. However when a DL_POLY program is assembled from its makefile, it will be placed in this sub-directory and will subsequently be executed from here. The output from the job will also appear here, so users will find it convenient to use this sub-directory if they wish to use DL_POLY as intended. (The experienced user is not absolutely required to use DL_POLY this way however.)

1.4.5 The *build* Sub-directory

This sub-directory contains the standard makefiles for the creation (i.e. compilation and linking) of the DL_POLY simulation programs. The makefiles supplied select the appropriate subroutines from the 'source' sub-directory and deposit the executable program in the 'execute' directory. The user is advised to copy the appropriate makefile into the execute directory, in case any modifications are required. The copy in the 'build' sub-directory will then serve as a backup.

1.5 Obtaining the Source Code

The source code for DL_POLY is located in the CCP5 Program Library at Daresbury Laboratory in the United Kingdom. It is the property of the SERC and all users of DL_POLY must obtain their copy from this source. Copies obtained from elsewhere will be deemed illegal and will not be supported by Daresbury. Potential users should therefore register with the CCP5 Program Librarian (address: Theory and Computational Science Division, Daresbury Laboratory, Daresbury, Warrington WA4 4AD, United Kingdom.) The code is free to all academic institutions, so there is no need to steal it.

DL_POLY may be accessed by electronic mail or by anonymous FTP. However the documentation of the package is *not* available by these routes and can only be obtained as hard copy from the Librarian. A magnetic tape service is also available, but users must first send a magnetic tape to the librarian.

The following sections summarise the methods for obtaining the code.

Scope of Chapter

This chapter describes all the subroutines of DL_POLY to be found in the 'source' subdirectory.

2.1 Subroutine and Function Specifications

2.1.1 DL_POLY Parameters (Include) File

2.1.1.1 Header

```
c
c***********************************************************************
c
c     dl_poly insert file specifying array sizes for the
c     entire package
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c
c     note the following internal units apply everywhere
c
c     unit of time      (to)    =          1 x 10**(-12) seconds
c     unit of length    (lo)    =          1 x 10**(-10) metres
c     unit of mass      (mo)    = 1.6605402  x 10**(-27) kilograms
c     unit of charge    (qo)    = 1.60217733 x 10**(-19) coulombs
c     unit of energy    (Eo)    = 1.6605402  x 10**(-23) joules
c
c***********************************************************************
c
```

2.1.1.2 Function

The include file **dl_params.inc** contains all the parameters defining the arrays dimensions of DL_POLY plus the fundamental constants and conversion factors. It is included within all relevant subroutines at compile time.

2.1.1.3 The Parameters and their Function

parameter	value	function
boltz	8.31451115d-1	boltzmann constant in internal units
kmax	8	max reciprocal space vector index
msatms	1000	max number of atoms in working arrays
msbad	max(mxbond,mxangl, mxdihd)	max size of working arrays for bond, angle and dihedral routines
mxangl	400	max number of bond angles on a node
mxatms	4000	max number of atoms
mxbond	500	max number of chemical bonds on a node
mxbuff	max(6*mxatms, 8*(mxcons+1))	max dimension of atomic transfer buffer
mxcons	2000	max number of constraint bonds on a node
mxdihd	500	max number of torsion angles on a node

mxexcl	50	max number of excluded interactions per atom
mxgrid	2000	max number of grid points in potential arrays
mxlist	800	max number of atoms in verlet list
mxlshp	2000	max dimension of shape routine transfer array
mxmols	1500	max number of molecules
mxnstk	25+((mxsvdw-1)/5)*5	max number of stacked variables
mxproc	64	max number of nodes (used in parallel constraint algorithm)
mxrdf	256	number of tabulation points for radial distribution functions
mxshak	100	max number of iterations in shake algorithm
mxsite	1000	max number of molecular sites
mxstak	100	dimension of stack arrays for rolling averages
mxsvdw	14	max number of different types of sites for pair potentials
mxtang	1500	max number of different bond angle potentials
mxtbnd	1500	max number of chemical bond potentials
mxtcon	4000	max number of specified bondlength constraints
mxtdih	2000	max number of different torsional potentials
mxtmls	3	max number of molecule types
mxvdw	((mxsvdw+1)* mxsvdw)/2	max number of different pair potentials
mxxdf	max(mxlist,msatms)	max number of atoms in xdf,ydf and zdf arrays
nconf	8	configuration file input channel
ndump	500	data dumping interval in event of system crash
nfield	7	force field input channel
nhist	21	trajectory history file channel
nread	5	main input channel
nrest	22	output channel accumulators restart dump file
nrite	6	main output channel
nstats	20	statistical data file output channel
pi	3.141592653589793d0	π
r4pie0	138935.4835d0	electrostatic conversion factor i.e. (unit(charge)**2/(4 pi eps0 unit(length))/ unit(energy)
sqrpi	1.7724538509055159d0	square root of π

2.1.1.4 Comments

Any changes made to the parameters file requires the entire program to be recompiled.

2.1.2 ANGFRC

2.1.2.1 Header records

```
      subroutine angfrc
     x     (idnode,imcon,mxnode,ntangl,engang,virang,keyang,listang,
     x     cell,fxx,fyy,fzz,prmang,xxx,yyy,zzz,xdab,ydab,zdab,xdbc,
     x     ydbc,zdbc)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating bond angle energy and
c     force terms in molecular dynamics.
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith         may 1992
c     modified  - t. forester      feb 1993
c
c*********************************************************************
c
```

2.1.2.2 Function

Calculates valence angle interactions.

2.1.2.3 Dependencies

- images

2.1.2.4 Arguments

integers:		
idnode	input	identity of current processor.
imcon	input	key for periodic boundary conditions.
mxnode	input	total number of processors.
ntangl	input	total number of interactions in the system.
reals:		
engang	output	valence angle interaction energy.
virang	output	valence angle interaction virial.
integer arrays:		
keyang	input	key for angle potential (mxtang).
listang	input	index for potential variables and atomic coordinates (mxangl,4).
real arrays:		
cell	input	MD cell vectors (9).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
prmang	input	potential variables (mxtang,2).
xxx	input	x coordinate (mxatms).

yyy	input	y coordinate (mxatms).
zzz	input	z coordinate (mxatms).
xdab	work	x component of a-b inter atomic vector (msbad).
ydab	work	y component of a-b inter atomic vector (msbad).
zdab	work	z component of a-b inter atomic vector (msbad).
xdbc	work	x component of b-c inter atomic vector (msbad).
ydbc	work	y component of b-c inter atomic vector (msbad).
zdbc	work	z component of b-c inter atomic vector (msbad).

2.1.2.5 Comments

In DL_POLY Release I the array KEYANG is not used - all interactions are assumed to be of the form

$$U = \frac{1}{2}k(\theta - \theta_0)^2. \tag{2.1}$$

k, the force constant, is stored in PRMANG(i,1). θ_0, the equilibrium angle, is stored in PRMANG(i,2).

2.1.3 BNDFRC

2.1.3.1 Header records

```
      subroutine bndfrc
     x     (idnode,imcon,mxnode,ntbndl,engbnd,virbnd,keybnd,listbnd,
     x     cell,fxx,fyy,fzz,prmbnd,xxx,yyy,zzz,xdab,ydab,zdab)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating chemical bond energy and
c     force terms in molecular dynamics.
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith        july 1992
c
c     'lennard-jones' bonds added for GROMOS 1-4 interactions
c               - t forester     march 1993
c
c*********************************************************************
c
```

2.1.3.2 Function

Calculates chemical bond energy and force terms.

2.1.3.3 Dependencies

- images

2.1.3.4 Arguments

integers:		
idnode	input	identity of current processor.
imcon	input	key for periodic boundary conditions.
mxnode	input	total number of processors.
ntbond	input	total number of interactions in the system.
reals:		
engbnd	output	chemical bond interaction energy.
virbnd	output	chemical bond interaction virial.
integer arrays:		
keybnd	input	key for bond potential (mxtbnd).
listbnd	input	index for potential variables and atomic coordinates (mxbond,3).
real arrays:		
cell	input	MD cell vectors (9).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
prmbnd	input	potential variables (mxtbnd,3).

xxx	input	x coordinate (mxatms).
yyy	input	y coordinate (mxatms).
zzz	input	z coordinate (mxatms).
xdab	work	x component of a-b inter atomic vector (msbad).
ydab	work	y component of a-b inter atomic vector (msbad).
zdab	work	z component of a-b inter atomic vector (msbad).

2.1.3.5 Comments

In Release I three type of bond potentials are available:

1. KEYBND(i) = 1 : Harmonic bonds

$$U = \frac{1}{2}k(r - r_0)^2 \tag{2.2}$$

   k, the force constant, is stored in PRMBND(i,1). r_0, the equilibrium distance, is stored in PRMBND(i,2).

2. KEYBND(i) = 2: Morse potential

$$U = A\left[1 - \exp(-C(r - r_0))\right]^2 . \tag{2.3}$$

   A is stored in PRMBND(i,1), r_0 in PRMBND(i,2), and C in PRMBND(i,3).

3. KEYBND(i) = 3 : "Lennard-Jones" bonds. For example, these are used for the GROMOS force field which has modified Lennard-Jones variables for 1-4 type interactions. The appropriate variables are generated from the DL_POLY utility groforce. In effect these terms take out any extra contribution to 1 - 4 interactions that arise from using the normal Lennard -Jones variables in the calculation of non-bonded terms (in the subroutine srfrce).

2.1.4 COUL0

2.1.4.1 Header records

```
      subroutine coul0
     x     (iatm,ik,engcpe,vircpe,rcut,epsq,
     x     ilist,chge,rsqdf,xdf,ydf,zdf,fxx,fyy,fzz)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating coulombic force.
c     1/r potential, no truncation or damping
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1993
c     author    - t. forester february 1993
c
c*********************************************************************
c
```

2.1.4.2 Function

Calculates interaction energy and forces for the Coulombic potential.

2.1.4.3 Dependencies

- none

2.1.4.4 Arguments

integers:		
iatm	input	index of the central site.
ik	input	number of neighbours around the central site.
reals:		
engcpe	output	Coulombic energy.
vircpe	output	Coulombic virial.
rcut	input	potential cut off radius.
epsq	input	relative dielectric constant.
integer arrays:		
ilist	input	indices of neighbours of the central site (mxlist).
real arrays:		
chge	input	electrostatic charge (mxatms).
rsqdf	input	square of intersite distance (mxrdf).
xdf	input	x component of intersite vector (mxxdf).
ydf	input	y component of intersite vector (mxxdf).
zdf	input	z component of intersite vector (mxxdf).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).

2.1.4.5 Comments

2.1.5 COUL2

2.1.5.1 Header records

```
      subroutine coul2
     x     (iatm,ik,engcpe,vircpe,rcut,epsq,ilist,chge,
     x      rsqdf,xdf,ydf,zdf,fxx,fyy,fzz)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating coulombic forces
c     assuming a distant dependent dielectric 'constant'.
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1993
c     author    - t. forester    april 1993
c
c*********************************************************************
c
```

2.1.5.2 Function

Calculates interaction energy and forces for a Coulombic potential with a distant dependent dielectric constant. The form of the potential is

$$E = \frac{1}{4\pi\epsilon_0(\epsilon_r)}\frac{q_i q_j}{r} \tag{2.4}$$

2.1.5.3 Dependencies

- none

2.1.5.4 Arguments

integers:		
iatm	input	index of the central site.
ik	input	number of neighbours around the central site.
reals		
engcpe	output	Coulombic energy.
vircpe	output	Coulombic virial.
rcut	input	potential cut off radius.
epsq	input	relative dielectric constant at r = 1.
integer arrays:		
ilist	input	indices of neighbours of the central site (mxlist).
real arrays:		
chge	input	electrostatic charge (mxatms).
rsqdf	input	square of intersire vector (mxxdf).
xdf	input	x component of intersite vector (mxxdf).
ydf	input	y component of intersite vector (mxxdf).

zdf	input	z component of intersite vector (mxxdf).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).

2.1.5.5 Comments

2.1.6 CYCLE

2.1.6.1 Header records

```
      subroutine cycle
     x     (lfcap,lgofr,loptim,lpgr,ltraj,ltscal,lzeql,atmnam,cfgname,
     x     idnode,imcon,intsta,istraj,keyens,keyfce,keyres,keytrj,
     x     kmax1,kmax2,kmax3,multt,mxnode,natms,nscons,nstack,nstep,
     x     nstbgr,nstbpo,nstbts,nsteql,nstraj,nstrun,ntangl,ntbond,
     x     ntcons,ntdihd,ntpvdw,numacc,numrdf,ntpatm,
     x     alpha,chit,degfre,delr,dlrpot,elrc,engang,engbnd,engcpe,
     x     engdih,engsrp,engunit,epsq,plrc,pmass,press,qmass,rcut,
     x     rprim,temp,timcls,timjob,tolnce,tstep,virang,virbnd,vircon,
     x     vircpe,virdih,virsrp,volm,
     x     ibuff,ilist,irdf,keyang,keybnd,keydih,lashap,lentry,lexatm,
     x     lishap,list,listang,listbnd,listcon,listdih,listme,listin,
     x     listot,lstvdw,ltpvdw,ltype,nexatm,noxatm,numtyp,
     x     buffer,cell,chge,ckc,cks,clm,dens,dxt,dxx,dyt,dyy,dzt,dzz,
     x     elc,els,emc,ems,enc,ens,erc,fer,flx,fly,flz,fpx,fpy,fpz,fxx,
     x     fyy,fzz,ggg,prmang,prmbnd,prmcon,prmdih,prmvdw,ravval,slm,
     x     ssqval,stkval,stpval,sumval,txx,tyy,tzz,uxx,uyy,uzz,vvv,vxx,
     x     vyy,vzz,weight,xdab,xdbc,xdcd,xdf,xxt,xxx,ydab,ydbc,ydcd,
     x     ydf,yyt,yyy,zdab,zdbc,zdcd,zdf,zumval,zzt,zzz,xx0,yy0,zz0)

c
c*********************************************************************
c
c     dl_poly subroutine for controlling the molecular dynamics
c     simulation cycle and generating the periodic output data
c
c     copyright daresbury laboratory 1992
c     author - w. smith         july 1992
c
c*********************************************************************
c
```

2.1.6.2 Function

Controls the MD cycle of evaluating forces, integrating equations of motion and accumulating statistics.

2.1.6.3 Dependencies

- angfrc
- bndfrc
- dihfrc
- dcell

- error
- fcap
- forces
- gdsum
- multiple
- nvt_e0
- nvt_e1
- nvt_h0
- nvt_h1
- parlst
- revive
- rdshake
- static
- timchk
- traject
- verlet
- vertest
- vscale

2.1.6.4 Arguments

logicals:		
lfcap	input	flag for force capping in equilibration mode.
lgofr	input	flag for calculation of r.d.f.'s online.
loptim	input	flag for 0 K structure optimization.
lpgr	input	flag to print radial distribution functions.
ltraj	input	flag for dumping trajectory data.
ltscal	input	flag for temperature scaling.
lzeql	input	flag for equilibration period.
characters:		
atmnam	input	atom labels [a8], (mxatms).
cfgname	input	configuration name [a80].
integers:		
idnode	input	identity of current node.
imcon	input	key for periodic boundary conditions.
intsta	input	interval for accumulating statistics.
istraj	input	interval for dumping trajectories.

keyens	input	key for selecting ensemble.
keyfce	input	key for non-bonded force field.
keyres	input	key for restart of MD run.
keytrj	input	key for trajectory information.
kmax1	input	number of reciprocal space vectors in a direction.
kmax2	input	number of reciprocal space vectors in b direction.
kmax3	input	number of reciprocal space vectors in c direction.
multt	input	size of multiple time-step interval.
mxnode	input	total number of nodes.
natms	input	number of atoms in MD cell.
nscons	input	number of constraints on current node.
nstack	input	size of stack for rolling averages.
nstep	input	running counter of completed MD cycles.
nstbgr	input	interval for collecting rdf statistics.
nstbpo	input	interval for printing out statistical data.
nstbts	input	interval for temperature scaling in equilibration mode.
nsteql	input	number of time-steps in run for equilibation.
nstraj	input	timestep at which dumping of trajectory information begins.
nstrun	input	number of time-steps for MD run.
ntangl	input	total number of valence angles in system.
ntbond	input	total number of bonds in system.
ntcons	input	total number of constraints in system.
ntdihd	input	total number of dihedral angles in the system.
ntpvdw	input	number of types of non-bonded potentials.
numacc	in/out	running counter of the number of steps used for statistics.
numrdf	in/out	number of steps used for rdf statistics.
ntpatm	input	number of unique atom types.
reals:		
alpha	input	Ewald convergence variable.
chit	in/out	Friction coefficient for Hoover thermostat.
delr	input	border width for Verlet list.
dlrpot	input	distance increment for entries in potential interpolation tables.
elrc	in/out	long range correction to energy.
engang	output	valence angle energy.
engbnd	output	bond energy.
engcpe	output	electrostatic energy.
engdih	output	dihedral energy.
engsrp	output	non-bonded potential energy.
engunit	input	conversion factor for energy units.
epsq	input	relative dielectric constant.
plrc	in/out	long range correction to pressure.
pmass	input	piston mass for pressure control (not in use).
press	input	simulation pressure (not in use).
qmass	input	Thermal inertia variable for Hoover thermostat.
rcut	input	potential cut-off radius.
rprim	input	radius of primary cut-off shell.
temp	input	simulation temperature.

timcls	input	time, in seconds, allocated for closing the job.
timjob	input	total time in seconds allocated for the job.
tolnce	input	tolerance for SHAKE algorithm.
tstep	input	simulation time-step, δt.
virang	output	valence angle virial.
virbnd	output	bond virial.
vircon	output	constraint virial.
vircpe	output	electrostatic virial.
virdih	output	dihedral virial.
virsrp	output	non-bonded potential virial.

integer arrays:

ibuff	work	communication routine buffer (mxrdf*mxvdw).
ilist	work	neighbourhood list for central site (mxlist).
irdf	in/out	histograms for rdfs (mxrdf,mxvdw).
keyang	input	valence angle potential key (mxtang).
keybnd	input	chemical bond potential key (mxtbnd).
keydih	input	dihedral potential key (mxtdih).
lashap	work	see **rdshake** (mxproc).
lentry	output	number of entries in Verlet list (msatms).
lexatm	input	excluded atoms list, see **parlst** (msatms,mxexcl).
lishap	work	see **rdshake** (mxlshp).
list	output	the Verlet list (msatms,mxlist).
listang	input	key and indices for valence angle interactions (mxangl,4).
listbnd	input	key and indices for chemical bond interactions (mxbond,3).
listcon	input	indices for constraint bonds (mxcons,3).
listdih	input	key and indices for dihedral interactions (mxdihd,5).
listme	input	see **rdshake** (mxatms).
listin	input	see **rdshake** (mxatms).
listot	input	see **rdshake** (mxatms).
lstvdw	input	index number of non-bonded potential (mxvdw).
ltpvdw	input	key for non-bonded potentials (mxvdw).
ltype	input	list of atom types (mxatms).
nexatm	input	number of excluded atoms, see **parlst** (msatms).
noxatm	work	see **parlst** (msatms).
numtyp	input	number of atoms of a given type (mxsvdw).

real arrays:

buffer	work	see **merge** (mxbuff).
cell	in/out	MD cell vectors (9).
chge	input	partial charges, (mxatms).
ckc	work	see **ewald1** (msatms).
cks	work	see **ewald1** (msatms).
clm	work	see **ewald1** (msatms).
dens	input	number density of atom types (mxsvdw).
dxt	work	see **rdshake** (mxcons).
dxx	work	see **rdshake** (mxcons).
dyt	work	see **rdshake** (mxcons).

dyy	work	see **rdshake** (mxcons).
dzt	work	see **rdshake** (mxcons).
dzz	work	see **rdshake** (mxcons).
elc	work	see **ewald1** (msatms,0:kmax).
els	work	see **ewald1** (msatms,0:kmax).
emc	work	see **ewald1** (msatms,0:kmax).
ems	work	see **ewald1** (msatms,0:kmax).
enc	work	see **ewald1** (msatms,0:kmax).
ens	work	see **ewald1** (msatms,0:kmax).
erc	input	interpolation table for Ewald sum real space energy (mxgrid).
fer	input	interpolation table for Ewald sum real space force (mxgrid).
flx	work	x component of secondary force (mxatms).
fly	work	y component of secondary force (mxatms).
flz	work	z component of secondary force (mxatms).
fpx	work	x component of secondary force (mxatms).
fpy	work	y component of secondary force (mxatms).
fpz	work	z component of secondary force (mxatms).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
ggg	input	interpolation table for non-bonded force (mxgrid,mxvdw).
prmang	input	variables for valence angle potentials (mxtang,2).
prmbnd	input	variables for bond potentials (mxtbnd,3).
prmcon	input	constraint distances (mxtcon).
prmdih	input	variables for dihedral potentials (mxtdih,3).
prmvdw	input	variables for non-bonded potentials (mxvdw,5).
ravval	in/out	statistics array, see **static** (mxnstk).
slm	work	see **ewald1** (msatms).
ssqval	in/out	statistics array, see **static** (mxnstk).
stkval	in/out	statistics array, see **static** (mxstak,mxnstk).
stpval	in/out	statistics array, see **static** (mxnstk).
sumval	in/out	statistics array, see **static** (mxnstk).
txx	work	(mxatms).
tyy	work	(mxatms).
tzz	work	(mxatms).
uxx	work	(mxatms).
uyy	work	(mxatms).
uzz	work	(mxatms).
vvv	input	interpolation table for non-bonded energy (mxgrid,mxvdw).
vxx	in/out	x component of velocity (mxatms).
vyy	in/out	y component of velocity (mxatms).
vzz	in/out	z component of velocity (mxatms).
weight	input	mass (mxatms).
xdab	work	used in **angfrc,bndfrc,dihfrc** (msbad).
xdbc	work	used in **angfrc,dihfrc** (msbad).
xdcd	work	used in **dihfrc** (msbad).
xdf	work	x component of intersite vector (mxxdf).

xxt	work	(mxatms).
xxx	in/out	x coordinates (mxatms).
ydab	work	used in **angfrc,bndfrc,dihfrc** (msbad).
ydbc	work	used in **angfrc,dihfrc** (msbad).
ydcd	work	used in **dihfrc** (msbad).
ydf	work	y component of intersite vector (mxxdf).
yyt	work	(mxatms).
yyy	in/out	y coordinates (mxatms).
zdab	work	used in **angfrc,bndfrc,dihfrc** (msbad).
zdbc	work	used in **angfrc,dihfrc** (msbad).
zdcd	work	used in **dihfrc** (msbad).
zdf	work	z component of intersite vector (mxxdf).
zumval	in/out	statistics array, see **static** (mxnstk).
zzt	work	(mxatms).
zzz	in/out	z coordinates (mxatms).
xx0	in/out	increment in x coordinate (mxatms).
yy0	in/out	increment in y coordinate (mxatms).
zz0	in/out	increment in z coordinate (mxatms).

2.1.6.5 Comments

2.1.7 DCELL

2.1.7.1 Header records

```
      subroutine dcell(aaa,bbb)

c
c***********************************************************************
c
c     dl_poly subroutine to calculate the dimensional properies of
c     a simulation cell specified by the input matrix aaa.
c     the results are returned in the array bbb, with :
c
c     bbb(1 to 3) - lengths of cell vectors
c     bbb(4 to 6) - cosines of cell angles
c     bbb(7 to 9) - perpendicular cell widths
c     bbb(10)     - cell volume
c
c     copyright daresbury laboratory 1992
c     author - w. smith         july 1992
c
c***********************************************************************
c
```

2.1.7.2 Function

Calculates the dimensional properties of the MD cell.

2.1.7.3 Dependencies

- none

2.1.7.4 Arguments

real arrays:

aaa	input	MD cell vectors (9).
bbb	output	cell dimensional properties as per header records (10).

2.1.7.5 Comments

2.1.8 DIFFSN0

2.1.8.1 Header records

```
      subroutine diffsn0
     x     (idnode,natms,mxnode,tstep,vxx,vyy,vzz,xx0,yy0,zz0)

c*********************************************************************
c
c     DL_POLY routine for calculating displacements of sites from
c     t=0 positions
c
c     use diffsn1 for mean squared displacements
c
c     parallel version - replicated data.
c
c     copyright daresbury laboratory 1993
c
c     author - t. forester      june 1993
c
c*********************************************************************
```

2.1.8.2 Function

Calculates increment in atomic positions due to the current time-step.

2.1.8.3 Dependencies

- none

2.1.8.4 Arguments

integers:		
idnode	input	identity of current processor.
natms	input	number of atoms in MD cell.
mxnode	input	total number of processors.
reals:		
tstep	input	time step
real arrays:		
vxx	input	x component of velocity (mxatms).
vyy	input	y component of velocity (mxatms).
vzz	input	z component of velocity (mxatms).
xx0	in/out	running sum of x increments (mxatms).
yy0	in/out	running sum of y increments (mxatms).
zz0	in/out	running sum of z increments (mxatms).

2.1.8.5 Comments

This subroutine should be called every time step. Mean squared displacements are calculated in **diffsn1**.

2.1.9 DIFFSN1

2.1.9.1 Header records

```
      subroutine diffsn1
     x     (idnode,natms,ntpatm,mxnode,ltype,numtyp,amsd,
     x     xx0,yy0,zz0,buffer)

c*********************************************************************
c
c     DL_POLY routine for calculating mean squared displacements
c
c     displacements calculated in diffsn0
c
c     parallel version - replicated data.
c
c     copyright daresbury laboratory 1993
c
c     author - t. forester      june 1993
c
c*********************************************************************
```

2.1.9.2 Function

Calculates mean squared displacements of atomic types.

2.1.9.3 Dependencies

- none

2.1.9.4 Arguments

integers:

idnode	input	identity of current processor.
natms	input	number of atoms in MD cell.
ntpatm	input	number of unique atomic types.
mxnode	input	total number of processors.

integer arrays:

ltype	input	atomic type index (mxatms).
numtyp	input	the number of atoms of each atomic type (mxsvdw)

real arrays:

amsd	output	mean squared displacements (mxsvdw).
xx0	input	running sum of x increments (mxatms).
yy0	input	running sum of y increments (mxatms).
zz0	input	running sum of z increments (mxatms).
buffer	work	work array for global sum (mxbuff).

2.1.9.5 Comments

Use in conjunction with diffsn0

2.1.10 DIHFRC

2.1.10.1 Header records

```
      subroutine dihfrc
     x     (idnode,imcon,mxnode,ntdihd,keyfce,epsq,engcpe,engdih,rcut,
     x     vircpe,virdih,keydih,listdih,
     x     buffer,cell,chge,fxx,fyy,fzz,prmdih,xxx,yyy,zzz,xdab,ydab,
     x     zdab,xdbc,ydbc,zdbc,xdcd,ydcd,zdcd)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating dihedral energy and force
c     terms in molecular dynamics.
c
c     version 2
c     -  reduced 1-4 electrostatics for AMBER force field
c
c     NOTE: assumes the normal electrostatics
c     have already been calculated elsewhere.
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith       march 1992
c
c     version 2  june 1993
c     author    - t. forester.
c
c*********************************************************************
c
```

2.1.10.2 Function

Calculates dihedral angle interactions.

2.1.10.3 Dependencies

- images

2.1.10.4 Arguments

integers:

idnode	input	identity of current processor.
imcon	input	key for periodic boundary conditions.
mxnode	input	total number of processors.
ntdihd	input	total number of interactions in the system.
keyfce	input	key for non-bonded force field.

reals:

epsq	input	relative dielectric constant.
engcpe	in/out	electrostatic energy.
engdih	output	dihedral interaction energy.

rcut	input	cut-off radius.
vircpe	in/out	electrostatic virial.
virdih	output	dihedral interaction virial.

integer arrays:

keydih	input	key for dihedral potential (mxtdih).
listdih	input	index for potential parameters and atomic coordinates (mxdihd,5).

real arrays:

cell	input	MD cell vectors (9).
chge	input	partial charges (mxatms).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
prmdih	input	potential parameters (mxtdih,3).
xxx	input	x coordinate (mxatms).
yyy	input	y coordinate (mxatms).
zzz	input	z coordinate (mxatms).
xdab	work	x component of a-b inter atomic vector (msbad).
ydab	work	y component of a-b inter atomic vector (msbad).
zdab	work	z component of a-b inter atomic vector (msbad).
xdbc	work	x component of b-c inter atomic vector (msbad).
ydbc	work	y component of b-c inter atomic vector (msbad).
zdbc	work	z component of b-c inter atomic vector (msbad).
xdcd	work	x component of c-d inter atomic vector (msbad).
ydcd	work	y component of c-d inter atomic vector (msbad).
zdcd	work	z component of c-d inter atomic vector (msbad).

2.1.10.5 Comments

In DL_POLY version I harmonic and cosine potential terms are available.

1. KEYDIH(i) = 1 : Cosine term

$$U = A\left[1 + \cos(n\phi - \delta)\right] \tag{2.5}$$

   A is stored in PRMDIH(i,1), δ in PRMDIH(i,2) and n in PRMDIH(i,3).

2. KEYDIH(i) = 2 : Harmonic term (suitable for improper dihedral interactions only)

$$U = \frac{1}{2}k(\phi - \phi_0)^2 \tag{2.6}$$

   k, the force constant, is stored in PRMDIH(i,1). ϕ_0, the equilibrium angle, is stored in PRMDIH(i,2).

3. KEYDIH(i) = 3 : Cosine term plus a reduced 1-4 electrostatic interaction. The Cosine term is identical to the KEYDIH(i) = 1 term above. In addition the 1-4 electrostatic interaction is reduced by 50%. This reduction is part of the specification of the AMBER force field.

2.1.11 DLPOLY

2.1.11.1 Header records

```
      program dlpoly
c
c***********************************************************************
c
c     dl_poly is a serc/ccp5 program package for the dynamical
c     simulation of molecular systems.
c
c     dl_poly is the property of the serc daresbury laboratory,
c     daresbury, warrington wa4 4ad. no part of the package may
c     be redistributed to third parties without the consent of
c     daresbury laboratory.
c
c     dl_poly is available free of charge to academic institutions
c     engaged in non-commercial research only. potential users not
c     in this category must consult the ccp5 program librarian at
c     daresbury to negotiate terms of use.
c
c     neither the serc, daresbury laboratory, ccp5 nor the authors
c     of this package claim that it is free from errors and do not
c     accept liability for any loss or damage that may arise from
c     its use. it is the users responsibility to verify that the
c     package dl_poly is fit for the purpose the user intends for
c     it.
c
c     all errors found by the user should be passed on to the
c     ccp5 program librarian without delay, for the benefit of other
c     users.
c
c     users are invited to contribute software to the dl_poly project
c     on the understanding that such software becomes the property of
c     serc and that its re-distribution will comply with the terms
c     outlined above.
c
c***********************************************************************
c
```

2.1.11.2 Function

Overall control of the simulation.

2.1.11.3 Dependencies

- cycle
- gsync

- machine
- result
- simdef
- sysdef
- sysgen

2.1.11.4 Arguments

none

2.1.11.5 Comments

2.1.12 DUNI

2.1.12.1 Header records

```
      function duni()
c
c*********************************************************************
c
c     dl_poly random number generator based on the universal
c     random number generator of marsaglia, zaman and tsang
c     (stats and prob. lett. 8 (1990) 35-39.) it must be
c     called once to initialise parameters u,c,cd,cm
c
c     copyright daresbury laboratory 1992
c     aurhor -  w.smith          july 1992
c
c*********************************************************************
c
```

2.1.12.2 Function

Random number generator. A uniform distribution on the interval [0,1) is produced.

2.1.12.3 Dependencies

- none

2.1.12.4 Comments

DUNI must be called once to initialise internal parameters, thereafter, the function is called once for each random number required.

2.1.13 ERROR

2.1.13.1 Header records

```
      subroutine error(idnode,kode)
c
c*********************************************************************
c
c     dl_poly subroutine for printing error messages and bringing
c     about a controlled termination of the program
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c     warning - this routine terminates the job. user must ensure
c     that all nodes are informed of error condition before this
c     subroutine is called. e.g. using subroutine gstate().
c
c*********************************************************************
c
```

2.1.13.2 Function

Writes error messages and terminates execution.

2.1.13.3 Dependencies

- none

2.1.13.4 Arguments

integers:

idnode	input	identity of current node.
knode	input	key for error type.

2.1.13.5 Comments

This routine should be called by all nodes simultaneously. Failure to do this will result in the program 'hanging'.

2.1.14 EWALD1

2.1.14.1 Header records

```
      subroutine ewald1
     x     (idnode,mxnode,natms,imcon,kmax1,kmax2,kmax3,
     x     engcpe,vircpe,alpha,volm,epsq,
     x     cell,chge,xxx,yyy,zzz,fxx,fyy,fzz,elc,emc,enc,els,ems,
     x     ens,ckc,cks,clm,slm)

c
c*********************************************************************
c
c     dl_poly subroutine for calculating coulombic forces in a
c     periodic system using ewald's method
c
c     parallel replicated data version (part 1)
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c     version 2
c     author    - t. forester april 1993
c
c     part 1 - reciprocal space terms (fourier part)
c
c     note - in loop over all k vectors k=2pi(ll/cl,mm/cl,nn/cl)
c     the values of ll,mm and nn are selected so that the symmetry of
c     reciprocal lattice is taken into account i.e. the following
c     rules apply.
c
c     ll ranges over the values 0 to kmax1 only.
c
c     mm ranges over 0 to kmax2 when ll=0 and over
c     -kmax2 to kmax2 otherwise.
c     nn ranges over 1 to kmax3 when ll=mm=0 and over
c     -kmax3 to kmax3 otherwise.
c
c     hence the result of the summation must be doubled at the end.
c
c*********************************************************************
c
```

2.1.14.2 Function

Calculation of reciprocal space contributions to the coulombic energy and forces for the Ewald sum. Applicable to periodic systems only.

2.1.14.3 Dependencies

- invert
- dcell
- gdsum

2.1.14.4 Arguments

integers:		
imcon	input	periodic boundaries control key. See **images**.
idnode	input	identity of the node on the cube.
mxnode	input	number of nodes on the cube.
natms	input	number of atoms in the MD cell.
kmax1	input	number of **k**-vectors in the a reciprocal space direction.
kmax2	input	number of **k**-vectors in the b reciprocal space direction.
kmax3	input	number of **k**-vectors in the c reciprocal space direction.
reals:		
engcpe	output	Coulombic energy.
vircpe	output	Coulombic virial.
alpha	input	Ewald sum convergence variable.
volm	input	volume of the MD cell.
epsq	input	relative dielectric constant.
real arrays:		
cell	input	cell vectors (9). See **images**.
chge	input	atomic electrostatic charge (mxatms).
xxx	input	x coordinate array (mxatms).
yyy	input	y coordinate array (mxatms).
zzz	input	z coordinate array (mxatms).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
elc	work	x component of the real part of exp(i.**k**.**r**) (msatms,0:kmax).
emc	work	y component of the real part of exp(i.**k**.**r**) (msatms,0:kmax).
enc	work	z component of the real part of exp(i.**k**.**r**) (msatms,0:kmax).
els	work	x component of the imaginery part of exp(i.**k**.**r**) (msatms,0:kmax).
ems	work	y component of the imaginery part of exp(i.**k**.**r**) (msatms,0:kmax).
ens	work	z component of the imaginery part of exp(i.**k**.**r**) (msatms,0:kmax).
ckc	work	work array for reciprocal space sum (msatms).
cks	work	work array for reciprocal space sum (msatms).
clm	work	work array for reciprocal space sum (msatms).
slm	work	work array for reciprocal space sum (msatms).

2.1.14.5 Comments

The amount of cpu time used in this routine is proportional to $kmax^3$. The optimal choice of the variable ALPHA is system dependent.

2.1.15 EWALD2

2.1.15.1 Header records

```
      subroutine ewald2
     x     (iatm,ik,engcpe,vircpe,rcut,alpha,epsq,
     x     ilist,chge,rsqdf,xdf,ydf,zdf,fxx,fyy,fzz,erc,fer)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating coulombic forces in a
c     periodic system using ewald's method
c
c     parallel replicated data version (part 2)
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c     part 2 - real space terms.
c
c     Tabulated potential in r space
c     3pt interpolation
c
c     t. forester March 1993
c
c*********************************************************************
c
```

2.1.15.2 Function

Calculates the Ewald sum real-space contribution to energy, virial and forces. For periodic systems with Coulombic forces only.

2.1.15.3 Dependencies

- none

2.1.15.4 Arguments

integers:		
iatm	input	index of central atom
ik	input	number of atoms in neighbour list array
reals:		
engcpe	output	calculated contribution to electrostatic energy
vircpe	output	calculated contribution to electrostatic virial
rcut	input	cutoff for real space forces calculation
alpha	input	Ewald convergence variable
epsq	input	relative dielectric constant
integer arrays:		
ilist	input	list of secondary atom indices (mxlist)
real arrays:		
chge	input	ionic charges of atoms (mxatms)
rsqdf	input	square of interatomic distance (mxxdf)
xdf	input	x component of interatomic separation vector (mxxdf)
ydf	input	y component of interatomic separation vector (mxxdf)
zdf	input	z component of interatomic separation vector (mxxdf)
fxx	output	x component of atomic forces (mxatms)
fyy	output	y component of atomic forces (mxatms)
fzz	output	z component of atomic forces (mxatms)
erc	input	interpolation array of function erfc(alpha r)/r (mxgrid)
fer	input	first derivative of interpolation array erc (mxgrid)

2.1.15.5 Comments

2.1.16 EWALD2_RSQ

2.1.16.1 Header records

```
      subroutine ewald2
     x     (iatm,ik,engcpe,vircpe,rcut,alpha,epsq,
     x     ilist,chge,rsqdf,xdf,ydf,zdf,fxx,fyy,fzz,erc,fer)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating coulombic forces in a
c     periodic system using ewald's method
c
c     parallel replicated data version (part 2)
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c     part 2 - real space terms.
c
c     Tabulated potential in r-squared space
c     3pt interpolation
c
c     t. forester March 1993
c
c*********************************************************************
c
```

2.1.16.2 Function

Calculates the Ewald sum real-space contribution to energy, virial and forces. For periodic systems with Coulombic forces only. Tabulation takes place in r-squared space.

2.1.16.3 Dependencies

- none

2.1.16.4 Arguments

integers:

iatm	input	index of central atom
ik	input	number of atoms in neighbour list array

reals:

engcpe	output	calculated contribution to electrostatic energy
vircpe	output	calculated contribution to electrostatic virial
rcut	input	cutoff for real space forces calculation
alpha	input	Ewald convergence variable
epsq	input	relative dielectric constant

integer arrays:

ilist	input	list of secondary atom indices (mxlist)

real arrays:

chge	input	ionic charges of atoms (mxatms)
rsqdf	input	square of interatomic distance (mxxdf)
xdf	input	x component of interatomic separation vector (mxxdf)
ydf	input	y component of interatomic separation vector (mxxdf)
zdf	input	z component of interatomic separation vector (mxxdf)
fxx	output	x component of atomic forces (mxatms)
fyy	output	y component of atomic forces (mxatms)
fzz	output	z component of atomic forces (mxatms)
erc	input	interpolation array of function erfc(alpha r)/r (mxgrid)
fer	input	first derivative of interpolation array erc (mxgrid)

2.1.16.5 Comments

This subroutine is designed to be used in conjunction with **forgen_rsq**. See **forgen_rsq** for a discussion on the relative merits of tabulation in r-squared space.

2.1.17 EWALD3

2.1.17.1 Header records

```
      subroutine ewald3
     x     (iatm,ilst,engcpe,vircpe,alpha,epsq,rcut,nexatm,
     x      lexatm,chge,xdf,ydf,zdf,fxx,fyy,fzz,erc,fer)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating coulombic forces in a
c     periodic system using ewald's method
c
c     parallel replicated data version (part 3)
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith dec 1992.
c
c     part 3 - intramolecular correction terms
c
c*********************************************************************
c
```

2.1.17.2 Function

Calculates the intramolecular corrections to the Ewald sum as defined by the excluded atoms list. For periodic systems with Coulombic forces only.

2.1.17.3 Dependencies

- none

2.1.17.4 Arguments

integers:		
iatm	input	index of primary atom
ilst	input	pointer for iatm in excluded atom arrays
reals:		
engcpe	output	calculated contribution to electrostatic energy
vircpe	output	calculated contribution to electrostatic virial
alpha	input	Ewald convergence parameter
epsq	input	relative dielectric constant
rcut	input	cutoff for real space forces calculation
integer arrays:		
nexatm	input	marks last entry in excluded atom list for atom iatm
lexatm	input	list of excluded atoms for atom iatm (msatms,mxexcl)
real arrays:		
chge	input	ionic charges of atoms (mxatms)
xdf	input	x component of interatomic separation vector (mxxdf)
ydf	input	y component of interatomic separation vector (mxxdf)
zdf	input	z component of interatomic separation vector (mxxdf)
fxx	output	x component of atomic forces (mxatms)
fyy	output	y component of atomic forces (mxatms)
fzz	output	z component of atomic forces (mxatms)
erc	input	interpolation array of function erfc(alpha r)/r (mxgrid)
fer	input	first derivative of interpolation array erc (mxgrid)

2.1.17.5 Comments

2.1.18 EXCLUDE

2.1.18.1 Header records

```
      subroutine exclude
     x     (idnode,mxnode,natms,ntpmls,keybnd,lexatm,lexsit,lstang,
     x     lstbnd,lstcon,nexatm,nexsit,numang,numbonds,numcon,nummols,
     x     numsit)
c
c*********************************************************************
c
c     dl_poly subroutine for constructing the excluded pair
c     interaction list of the system to be simulated
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith        june 1992
c
c     Bonds of type 3  (LJ bonds) are not in the exclusion list
c                t forester     march 1993
c
c*********************************************************************
c
```

2.1.18.2 Function

Generates lists of excluded non-bonded interactions. Interactions pertaining to a chemical bond or valence angle are excluded from the non-bonded interaction list.

2.1.18.3 Dependencies

- none

2.1.18.4 Arguments

integers:

idnode	input	identity of current processor.
mxnode	input	total number of processors.
natms	input	number of atoms in MD cell.
ntpmls	input	number of different types of molecule in the system.

integer arrays:

keybnd	input	key for chemical bond potentials (mxtbnd).
lexatm	output	the excluded interaction list (msatms,mxexcl).
lexsit	work	the excluded interaction list before minimisation (mxsite,mxexcl).
lstang	input	indices of sites in valence angles in the system (mxtang,3).
lstbnd	input	indices of sites in chemical bonds in the system (mxtbnd,2).
lstcon	input	indices of sites in constraint bonds in the system (mxtcon,2).
nexatm	output	number of entries in excluded interaction list for the central atom (msatms).
nexsit	work	number of excluded interactions before minimisation (mxsite).
numang	input	number of valence angles in a molecular type (mxtmls).

numbonds	input	number of chemical bondss in a molecular type (mxtmls).
numcon	input	number of constraint bonds in a molecular type (mxtmls).
nummols	input	number of molecules of each type (mxtmls).
numsit	input	number of sites in a molecular type (mxtmls).

2.1.18.5 Comments

2.1.19 FCAP

2.1.19.1 Header records

```
      subroutine fcap
     x     (idnode,mxnode,natms,temp,tstep,fxx,fyy,fzz)

c*********************************************************************
c
c     DLPOLY routinue for limiting the absolute magnitude of
c     forces. Used for equilibrating initial configurations
c
c     parallel version.
c
c     copyright daresbury laboratory 1993
c
c     author -     t. forester march 1993
c
c*********************************************************************
c
```

2.1.19.2 Function

Limits the magnitude of the force on atomic sites.

2.1.19.3 Dependencies

- none

2.1.19.4 Arguments

logicals:		
lfcap	input	flag for force capping.
integers:		
idnode	input	identity of current node.
mxnode	input	total number of nodes.
natms	input	number of sites in MD cell.
reals:		
temp	input	simulation temperature.
tstep	input	simulation time step, δt.
real arrays:		
fxx	in/out	x component of force (mxatms).
fyy	in/out	y component of force (mxatms).
fzz	in/out	z component of force (mxatms).

2.1.19.5 Comments

The maximum magnitude of the force permitted is $20k_b min(T, 1000K)$ in DL_POLY internal units. The forces are capped and then shifted so that momentum remains a

conserved quanity of the simulation. However, energy is not conserved by force capping. The principal use of force capping is to facilitate equilibration from "bad" configurations.

2.1.20 FORCES

2.1.20.1 Header records

```
      subroutine forces
     x     (idnode,imcon,keyfce,kmax1,kmax2,kmax3,mxnode,natms,
     x     alpha,dlrpot,engcpe,engsrp,epsq,rcut,vircpe,virsrp,volm,
     x     ilist,lentry,lexatm,list,lstvdw,ltype,nexatm,
     x     cell,chge,ckc,cks,clm,elc,els,emc,ems,enc,ens,erc,fer,fxx,
     x     fyy,fzz,ggg,slm,vvv,weight,xdf,xxx,ydf,yyy,zdf,zzz)

c
c*********************************************************************
c
c     dl_poly subroutine for calculating interatomic forces
c     using the verlet neighbour list
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c     modified  - t. forester april 1993
c     key:
c
c     keyfce = odd  ---- short range potentials calculated: srfrce
c            = 0,1  ---- no electrostatics
c            = 2,3  ---- Ewald sum                        : ewald1,2,3
c            = 4,5  ---- Distant dependent dielectric     : coul2
c            = 6,7  ---- coulombic                        : coul0
c            = 8,9  ---- truncated and shifted coulombic  : coul1
c
c*********************************************************************
c
```

2.1.20.2 Function

Controls calculation of non-bonded interactions for single time-step simulations.

2.1.20.3 Dependencies

- coul0
- coul1
- coul2
- ewald1
- ewald2
- ewald3
- gdsum
- images
- srfrce

2.1.20.4 Arguments

integers:		
idnode	input	identity of current node.
imcon	input	key for periodic boundary conditions.
keyfce	input	key for non-bonded force field.
kmax1	input	number of reciprocal space vectors in *a* direction.
kmax2	input	number of reciprocal space vectors in *b* direction.
kmax3	input	number of reciprocal space vectors in *c* direction.
mxnode	input	total number of nodes.
natms	input	number of atoms in MD cell.
reals:		
alpha	input	Ewald convergence parameter.
dlrpot	input	distance increment for entries in potential look-up tables.
engcpe	output	electrostatic energy.
engsrp	output	non-bonded potential energy.
epsq	input	relative dielectric constant.
rcut	input	potential cut-off radius.
vircpe	output	electrostatic virial.
virsrp	output	non-bonded potential virial.
volm	input	volume of MD cell.
integer arrays:		
ilist	work	neighbourhood list for central site (mxlist).
lentry	input	number of entries in Verlet list (msatms).
lexatm	input	excluded atoms list (msatms,mxexcl).
list	input	the Verlet list (msatms,mxlist).
lstvdw	input	index number of non-bonded potential (mxvdw).
ltype	input	list of atom types (mxatms).
nexatm	input	number of excluded atoms (msatms).
real arrays:		
cell	input	MD cell vectors (9).
chge	input	partial charges, (mxatms).
ckc	work	see **ewald1** (msatms).
cks	work	see **ewald1** (msatms).
clm	work	see **ewald1** (msatms).
elc	work	see **ewald1** (msatms,0:kmax).
els	work	see **ewald1** (msatms,0:kmax).
emc	work	see **ewald1** (msatms,0:kmax).
ems	work	see **ewald1** (msatms,0:kmax).
enc	work	see **ewald1** (msatms,0:kmax).
ens	work	see **ewald1** (msatms,0:kmax).

erc	input	interpolation table for Ewald sum real space energy.
fer	input	interpolation table for Ewald sum real space force.
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
ggg	input	interpolation table for non-bonded force (mxgrid,mxvdw).
slm	work	see **ewald1** (msatms).
vvv	input	interpolation table for non-bonded energy (mxgrid,mxvdw).
weight	input	mass (mxatms).
xdf	work	x component of intersite vector (mxxdf).
xxx	in/out	x coordinates (mxatms).
ydf	work	y component of intersite vector (mxxdf).
yyy	in/out	y coordinates (mxatms).
zdf	work	z component of intersite vector (mxxdf).
zzz	in/out	z coordinates (mxatms).

2.1.20.5 Comments

2.1.21 FORGEN

2.1.21.1 Header records

```
      subroutine forgen
     x     (idnode,keyfce,ntpvdw,alpha,dlrpot,rcut,
     x      ltpvdw,erc,fer,prmvdw,vvv,ggg)
c
c*********************************************************************
c
c     dl_poly subroutine for generating potential energy and
c     force arrays for van der waals forces only
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith may 1992.
c
c*********************************************************************
c
```

2.1.21.2 Function

Generates interpolation tables for non-bonded potentials and for the real space term of the Ewald sum.

2.1.21.3 Dependencies

- error

2.1.21.4 Arguments

integers:		
idnode	input	identity of current processor.
keyfce	input	key for non-bonded force field.
ntpvdw	input	number of non-bonded potentials in system.
reals:		
dlrpot	output	distance increment for tabulation.
rcut	input	non-bonded potential cut-off.
integer arrays:		
ltpvdw	input	key for non-bonded potentials (mxvdw).
real arrays:		
erc	output	interpolation table for real space Ewald sum potential (mxgrid).
fer	output	interpolation table for real space Ewald sum force (mxgrid).
prmvdw	input	variables for non-bonded potentials (mxvdw,5).
vvv	output	interpolation table for non-bonded potentials (mxgrid,mxvdw).
ggg	output	interpolation table for non-bonded force (mxgrid,mxvdw).

2.1.21.5 Comments

Tabulation takes place in r space. This routine is used in conjunction with **srfrce** and **ewald2** (or their variants). If tabulation in r^2 space is preferred use **forgen_rsq** instead of **forgen** (but see the comments section for **forgen_rsq**). A guide to the *minimum* number of grid points (MXGRID) required is

$$MXGRID \geq 100(RCUT/rmin) \tag{2.7}$$

where $rmin$ is the smallest value of the position minima of the non-bonded potentials in the system. MXGRID is a parameter defined in the **dl_params.inc** file.

2.1.22 FORGEN_RSQ

2.1.22.1 Header records

```
      subroutine forgen
     x     (idnode,keyfce,ntpvdw,alpha,dlrpot,rcut,
     x      ltpvdw,erc,fer,prmvdw,vvv,ggg)
c
c*********************************************************************
c
c     dl_poly subroutine for generating potential energy and
c     force arrays for van der waals forces only
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith may 1992.
c
c     modifed to be tabulated in r2 space not r space
c     author    - t. forester  march 1993
c
c*********************************************************************
c
```

2.1.22.2 Function

Generates look-up tables for non-bonded potentials, tabulated in uniform intervals of r^2.

2.1.22.3 Dependencies

- error

2.1.22.4 Arguments

integers:

idnode	input	identity of current processor.
keyfce	input	non-bonded force-field key.
ntpvdw	input	number of non-bonded potentials in system.

reals:

alpha	input	Ewald convergence variable.
dlrpot	output	distance increment for tabulation.
rcut	input	non-bonded potential cut-off.

integer arrays:

ltpvdw	input	key for non-bonded potentials (mxvdw). **real arrays:**
prmvdw	input	parameters for non-bonded potentials (mxvdw,5).
vvv	output	interpolation table for non-bonded potentials (mxgrid,mxvdw).
ggg	output	interpolation table for non-bonded force (mxgrid,mxvdw).

2.1.22.5 Comments

This subroutine is used in conjunction with the subroutines **srfrce_rsq** and **ewald2_rsq**. If tabulation in r space is preferred use **forgen** in place of **forgen_rsq**. Tabulation in r^2 avoids the use of the square root function in **srfrce_rsq** and **ewald2_rsq** thus enhancing execution time. Note that tabulation in r^2 usually requires more grid points (and hence more memory) than tabulation in r space. This is to ensure sufficient accurracy is retained at small r. A guide to the **minimum** number of grid points required is

$$MXGRID \geq 100(RCUT/rmin)^2 \tag{2.8}$$

where $rmin$ is the smallest value of the position minima of the non-bonded potentials in the system. MXGRID is a parameter in the **dl_params.inc** file.

If you are in doubt as whether to use **forgen** or **forgen_rsq** we recommend **forgen** (and hence **srfrce** and **ewald2** or one of their variants). This is because tabulation in r is less demanding on memory requirements and less prone to inaccuracy should too small a value of MXGRID, or too large a value of RCUT, be used.

2.1.23 GAUSS

2.1.23.1 Header records

```
      subroutine gauss(natms,vxx,vyy,vzz)
c
c*********************************************************************
c
c     dl_poly subroutine for constructing velocity arrays
c     with a gaussian distribution of unit variance.
c
c     based on the method described by Allen and Tildesley in
c     Computer Simulation of Liquids , Clarendon Press 1987 P347
c
c     note - this version uses a universal random number
c     generator, which generates pseudo-random numbers between
c     0 and 1. it is based on the algorithm of marsaglia, zaman
c     and tsang in: stats and prob. lett. 8 (1990) 35-39.
c
c     copyright daresbury laboratory 1992
c     author - w. smith         july 1992
c
c*********************************************************************
c
```

2.1.23.2 Function

Produces a Gaussian distribution of velocities.

2.1.23.3 Dependencies

- duni

2.1.23.4 Arguments

integers:

natms	input	number of atoms in the MD cell.

real arrays:

vxx	output	x component of velocity.
vyy	output	y component of velocity.
vzz	output	z component of velocity.

2.1.23.5 Comments

A call to **vscale** is required to ensure that the systems momentum is zero and to scale the kinetic temperature.

2.1.24 GEARPC

2.1.24.1 Header records

```
      subroutine gearpc
     x     (mode,idnode,mxnode,natms,imcon,tstep,engke,weight,cell,
     x     xxx,yyy,zzz,xx1,yy1,zz1,xx2,yy2,zz2,xx3,yy3,zz3,xx4,yy4,
     x     zz4,xx5,yy5,zz5,fxx,fyy,fzz,buffer)

c
c*********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - gear predictor corrector
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1993
c     author    - w. smith february 1993.
c
c*********************************************************************
c
```

2.1.24.2 Function

The Gear 5'th order predictor corrector for second order linear differential equations, to integrate the Newtonian equations of motion.

2.1.24.3 Dependencies

- gdsum
- images
- merge

2.1.24.4 Arguments

integers:		
mode	input	predictor/corrector selection mode=1 for predictor step mode=2 for corrector step
mxnode	input	total number of nodes on the cube.
natms	input	total number of sites in the simulated system.
imcon	input	periodic boundary condition key. See **images**
reals:		
tstep	input	simulation time step, δt.
engke	output	kinetic energy.
real arrays:		
weight	input	atomic mass array (mxatms).

cell	input	cell vectors (9). See **images**.
xxx	in/out	x coordinate array (mxatms).
yyy	in/out	y coordinate array (mxatms).
zzz	in/out	z coordinate array (mxatms).
xx1	in/out	1st deriv term of xxx (mxatms).
yy1	in/out	1st deriv term of yyy (mxatms).
zz1	in/out	1st deriv term of zzz (mxatms).
xx2	in/out	2nd deriv term of xxx (mxatms).
yy2	in/out	2nd deriv term of yyy (mxatms).
zz2	in/out	2nd deriv term of zzz (mxatms).
xx3	in/out	3rd deriv term of xxx (mxatms).
yy3	in/out	3rd deriv term of yyy (mxatms).
zz3	in/out	3rd deriv term of zzz (mxatms).
xx4	in/out	4th deriv term of xxx (mxatms).
yy4	in/out	4th deriv term of yyy (mxatms).
zz4	in/out	4th deriv term of zzz (mxatms).
xx5	in/out	5th deriv term of xxx (mxatms).
yy5	in/out	5th deriv term of yyy (mxatms).
zz5	in/out	5th deriv term of zzz (mxatms).
fxx	input	x component of force array (mxatms).
fyy	input	y component of force array (mxatms).
fzz	input	z component of force array (mxatms).
buffer	work	buffer array (mxbuff). See **merge**.

2.1.24.5 Comments

Calling this routine with mode=1 results on the predictor step being performed. Calling with mode=2 causes the corrector step to be performed.

2.1.25 GSTATE

2.1.25.1 Header records

```
      subroutine gstate(check)
c
c*********************************************************************
c
c     dl_poly global status subroutine for intel hypercube
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith       march 1992
c
c*********************************************************************
c
```

2.1.25.2 Function

Logical AND of variable across all processors.

2.1.25.3 Dependencies

- crecv
- csend

2.1.25.4 Arguments

logicals:

check	in/out	flag for testing.

2.1.25.5 Comments

If the variable CHECK is .FALSE. on any processor a value of .FALSE. will be obtained. This version is INTEL specific.

2.1.26 IMAGES

2.1.26.1 Header records

```
      subroutine images
     x     (imcon,idnode,mxnode,natms,cell,xxx,yyy,zzz)

c
c*********************************************************************
c
c     dl_poly subroutine for calculating the minimum image
c     of atom pairs within a specified MD cell
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c     for
c     imcon=0 no boundary conditions apply
c     imcon=1 standard cubic boundaries apply
c     imcon=2 orthorhombic boundaries apply
c     imcon=3 parallelepiped boundaries apply
c     imcon=4 truncated octahedron boundaries apply
c     imcon=5 rhombic dodecahedron boundaries apply
c     imcon=6 x-y rectangular boundary conditions : no periodicity in z
c
c     note: in all cases the centre of the cell is at (0,0,0)
c     warning - replicated data version: does not re-merge
c     coordinate arrays
c
c*********************************************************************
c
```

2.1.26.2 Function

Applies periodic boundary conditions as per Header records above.

2.1.26.3 Dependencies

- none

2.1.26.4 Arguments

integers:

imcon	input	key for periodic boundary conditions.
idnode	input	identity of current processor.
mxnode	input	total number of processors.
natms	input	number of vectors.

real arrays:

cell	input	MD cell vectors (9).
xxx	in/out	x coordinate (*).
yyy	in/out	y coordinate (*).
zzz	in/out	z coordinate (*).

2.1.26.5 Comments

2.1.27 INTLIST

2.1.27.1 Header records

```
      subroutine intlist
     x     (idnode,mxnode,natms,nscons,ntangl,ntbond,ntcons,ntdihd,
     x     ntpmls,lashap,lishap,listang,listbnd,listcon,listdih,
     x     listin,listme,listot,lstang,lstbnd,lstcon,lstdih,
     x     numang,numbonds,numcon,numdih,nummols,numsit)

c
c*********************************************************************
c
c     dl_poly subroutine for constructing the interaction lists
c     for the entire simulated system
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith        july 1992
c
c*********************************************************************
c
```

2.1.27.2 Function

Constructs bonded interaction lists for the entire system.

2.1.27.3 Dependencies

- error
- gstate
- passcon

2.1.27.4 Arguments

integers:

idnode	input	identity of the current processor.
mxnode	input	total number of processors.
natms	input	number of atoms in MD cell.
nscons	output	number of constraint bonds on the current processor.
ntangl	output	total number of valence angles in system.
ntbond	output	total number of chemical bonds in system.
ntcons	output	total number of constraint bonds in system.
ntdihd	output	total number of dihedral interactions in system.
ntpmls	input	number of different molecular types.

integer arrays:

lashap	output	see passcon (mxproc).
lishap	output	see passcon (mxlshp).

listang	output	indices for valence angles on current processor (mxangl,4).
listbnd	output	indices for chemical bonds on current processor (mxbond,3).
listcon	output	indices for constraint bonds on current processor (mxcons,3).
listdih	output	indices for dihedral interactions on current processor (mxdihd,5).
listin	output	see passcon (mxatms).
listme	output	see passcon (mxatms).
listot	output	see passcon (mxatms).
lstang	input	indices of sites in valence angles in the system (mxtang,3).
lstbnd	input	indices of sites in chemical bonds in the system (mxtbnd,2).
lstcon	input	indices of sites in constraint bonds in the system (mxtcon,2).
lstdih	input	indices of sites in dihedral interactions in the system (mxtdih,4).
numang	input	number of valence angles in a molecular type (mxtmls).
numbonds	input	number of chemical bondss in a molecular type (mxtmls).
numcon	input	number of constraint bonds in a molecular type (mxtmls).
numdih	input	number of dihedral interactions in each molecular type (mxtmls).
nummols	input	number of molecules of each type (mxtmls).
numsit	input	number of sites in each molecular type (mxtmls).

2.1.27.5 Comments

Need only be called once at the beginning of a job. Each processor constructs it's own, independent, interaction lists.

2.1.28 INVERT

2.1.28.1 Header records

```
      subroutine invert(a,b,d)
c
c*********************************************************************
c
c     dl_poly subroutine to invert a 3 * 3 matrix using cofactors
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith       april 1992
c
c*********************************************************************
c
```

2.1.28.2 Function

Inverts a 3×3 matrix.

2.1.28.3 Dependencies

- none

2.1.28.4 Arguments

reals:

d	output	determinant of original matrix.

real arrays:

a	input	matrix (9).
b	output	inverted matrix (9).

2.1.28.5 Comments

If the matrix is singular, (ie. determinant = 0) the inverse matrix is returned with all entries set to zero.

2.1.29 LOWCASE

2.1.29.1 Header records

```
      subroutine lowcase(string,length)

c*********************************************************************
c
c     DL_POLY routine to lowercase a string of up to 255 characters.
c     Transportable to non-ASCII machines
c
c     copyright daresbury laboratory 1993
c     author    t. forester     july 1993
c
c*********************************************************************
```

2.1.29.2 Function

Converts a string to lowercase characters.

2.1.29.3 Dependencies

- none

2.1.29.4 Arguments

character:

string	in/out	alphanumeric string

integer:

length	input	string length

2.1.29.5 Comments

2.1.30 LRCORRECT

2.1.30.1 Header records

```
      subroutine lrcorrect
     x     (idnode,imcon,keyfce,mxnode,natms,ntpatm,ntpvdw,elrc,engunit,
     x     plrc,rcut,volm,lstvdw,ltpvdw,ltype,numtyp,prmvdw,dens)

c*********************************************************************
c
c     dl_poly subroutine to evaluate long-range corrections to
c     pressure and energy in a periodic system
c
c     copyright daresbury laboratory 1993
c
c     author -       t. forester may 1993
c
c*********************************************************************
```

2.1.30.2 Function

Calculates long range corrections to the energy and pressure in a periodic system.

2.1.30.3 Dependencies

- none

2.1.30.4 Arguments

integers:

idnode	input	identity of current processor.
imcon	input	periodic boundary key.
keyfce	input	force level key.
mxnode	input	total number of processor.
natms	input	number of atoms in MD cell.
ntpatm	output	number of unique atom types.
ntpvdw	input	number of types of non-bonded potentials.

reals:

elrc	output	long range correction to energy.
engunit	input	energy unit for input/output.
plrc	output	long range correction to pressure.
rcut	input	potential cut-off radius.
volm	input	volume of the MD cell.

integer arrays:

lstvdw	input	pointer from potential index to parameter table (mxvdw).
ltpvdw	input	key for non-bonded potentials (mxvdw).
ltype	input	list of atom types (mxatms).
numtyp	output	number of atoms of given type in simulation (mxsvdw).

real arrays:

prmvdw	input	variables for non-bonded potentials (mxvdw,5).
dens	output	number density of atom types (mxsvdw).

2.1.30.5 Comments

2.1.31 MACHINE

2.1.31.1 Header records

```
      subroutine machine(idnode,mxnode)
c
c*********************************************************************
c
c     dl_poly subroutine for obtaining charcteristics of
c     the computer on which the program is being run
c
c     this version is for the intel hypercube
c
c     author - w.smith july 1992
c
c*********************************************************************
c
```

2.1.31.2 Function

Returns the processor identity and the number of processors in use.

2.1.31.3 Dependencies

- mynode
- numnodes

2.1.31.4 Arguments

integers:

idnode	output	identity of current processor.
mxnode	output	total number of processors in use.

2.1.31.5 Comments

The subroutine makes use of the INTEL functions **mynode()** and **numnodes()**. Thus DL_POLY assumes idnode takes a value between 0 and mxnode-1.

For running DL_POLY on single processor non-INTEL machines simply adjust this routine to read as :

```
      subroutine machine(idnode,mxnode)
      idnode = 0
      mxnode = 1
      return
      end
```

2.1.32 MERGE

2.1.32.1 Header records

```
      subroutine merge(idnode,mxnode,natms,nbuff,xxx,yyy,zzz,buffer)
c
c*********************************************************************
c
c     dl_poly subroutine for merging coordinate arrays across
c     a number of processors
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith november 1992.
c
c
c*********************************************************************
c
```

2.1.32.2 Function

Merges arrays across a number of processors. Used, for example, in the Replicated Data integration of the equations of motion where site i is handled on the processor with idnode = (i-1) Modulo mxnode.

2.1.32.3 Dependencies

- csend
- error
- gsync
- irecv
- msgwait

2.1.32.4 Arguments

integers:

idnode	input	identity of the current processor.
mxnode	input	total number of processors.
natms	input	number of elements in arrays to be merged.
nbuff	input	size of buffer array.

real arrays:

x	in/out	first array to be merged.
y	in/out	second array to be merged.
z	in/out	third array to be merged.

2.1.32.5 Comments

2.1.33 MULTIPLE

2.1.33.1 Header records

```
      subroutine multiple
     x     (lgofr,lzeql,newjob,newlst,
     x     idnode,imcon,keyfce,kmax1,kmax2,kmax3,multt,mxnode,
     x     natms,nstep,nstbgr,nsteql,numrdf,
     x     alpha,dlrpot,engcpe,engsrp,epsq,rcut,rprim,vircpe,virsrp,
     x     volm,
     x     ilist,irdf,lentry,lexatm,list,lstvdw,ltype,nexatm,
     x     cell,chge,ckc,cks,clm,elc,els,emc,ems,enc,ens,erc,fer,flx,
     x     fly,flz,fpx,fpy,fpz,fxx,fyy,fzz,ggg,slm,vvv,weight,xdf,xxx,
     x     ydf,yyy,zdf,zzz)


c*********************************************************************
c
c     DL_POLY subroutine for multiple time step algorithm
c     reciprocal space as part of "secondary forces"
c
c
c     copyright daresbury laboratory
c
c     author  t. forester,  may 1993
c
c     keyfce = odd  ------ short range potentials calculated: srfrce
c            = 0,1  ------ no electrostatics
c            = 2,3  ------ Ewald sum                        : ewald1,2,3
c            = 4,5  ------ Distant dependent dielectric     : coul2
c            = 6,7  ------ coulombic                        : coul0
c            = 8,9  ------ truncated and shifted coulombic  : coul1
c
c*********************************************************************
c
```

2.1.33.2 Function

Controls calculation of non-bonded interactions for multiple time-step simulations.

2.1.33.3 Dependencies

- coul0
- coul1
- coul2
- ewald1
- ewald2

- ewald3
- ewald4
- gdsum
- images
- primlst
- srfrce

2.1.33.4 Arguments

logicals:		
newjob	input	flag for beginning of a job.
newlst	input	flag for updated verlet list.
integers:		
idnode	input	identity of current node.
imcon	input	key for periodic boundary conditions.
keyfce	input	key for non-bonded force field.
kmax1	input	number of reciprocal space vectors in a direction.
kmax2	input	number of reciprocal space vectors in b direction.
kmax3	input	number of reciprocal space vectors in c direction.
multt	input	size of multiple time-step interval.
mxnode	input	total number of nodes.
natms	input	number of atoms in MD cell.
nstep	input	step counter.
reals:		
alpha	input	Ewald convergence parameter.
dlrpot	input	distance increment for entries in potential interpolation tables.
engcpe	output	electrostatic energy.
engsrp	output	non-bonded potential energy.
epsq	input	relative dielectric constant.
rcut	input	potential cut-off radius.
rprim	input	radius of primary cut-off shell.
vircpe	output	electrostatic virial.
virsrp	output	non-bonded potential virial.
volm	input	volume of MD cell.
integer arrays:		
ilist	work	neighbourhood list for central site (mxlist).
lentry	input	number of entries in Verlet list (msatms).
lexatm	input	excluded atoms list (msatms,mxexcl).
list	input	the Verlet list (msatms,mxlist).
lstvdw	input	index number of non-bonded potential (mxvdw).
ltype	input	list of atom types (mxatms).
nexatm	input	number of excluded atoms (msatms).
real arrays:		
cell	input	MD cell vectors (9).
chge	input	partial charges, (mxatms).

ckc	work	see **ewald1** (msatms).
cks	work	see **ewald1** (msatms).
clm	work	see **ewald1** (msatms).
elc	work	see **ewald1** (msatms,0:kmax).
els	work	see **ewald1** (msatms,0:kmax).
emc	work	see **ewald1** (msatms,0:kmax).
ems	work	see **ewald1** (msatms,0:kmax).
enc	work	see **ewald1** (msatms,0:kmax).
ens	work	see **ewald1** (msatms,0:kmax).
erc	input	interpolation table for Ewald sum real space energy.
fer	input	interpolation table for Ewald sum real space force.
flx	work	x component of secondary force (mxatms).
fly	work	y component of secondary force (mxatms).
flz	work	z component of secondary force (mxatms).
fpx	work	x component of secondary force (mxatms).
fpy	work	y component of secondary force (mxatms).
fpz	work	z component of secondary force (mxatms).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
ggg	input	interpolation table for non-bonded force (mxgrid,mxvdw).
slm	work	see **ewald1** (msatms).
vvv	input	interpolation table for non-bonded energy (mxgrid,mxvdw).
weight	input	mass (mxatms).
xdf	work	x component of intersite vector (mxxdf).
xxx	input	x coordinates (mxatms).
ydf	work	y component of intersite vector (mxxdf).
yyy	input	y coordinates (mxatms).
zdf	work	z component of intersite vector (mxxdf).
zzz	input	z coordinates (mxatms).

2.1.33.5 Comments

Secondary forcesand enrgies are evaluated on the first and second steps of the multiple timestep cycle. The time derivative of the secondary contributions are then evaluated and used to estimate the value of the secondary contributions on subsequent timesteps. Thus the value of MULTT in the CONTROL file must be at least 3 otherwise a single time step algorithm is reproduced.

2.1.34 NVT_B0

2.1.34.1 Header records

```
      subroutine nvt_b0
     x     (idnode,imcon,natms,mxnode,engke,qmass,sigma,tstep,
     x     buffer,cell,fxx,fyy,fzz,vxx,vyy,vzz,weight,xxx,yyy,zzz)

c
c*********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - verlet leapfrog with Berendsen
c     thermostat.
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory  1993
c     author    - t. forester      july 1993
c
c*********************************************************************
c
```

2.1.34.2 Function

Integrates equations of motion subject to the Berendsen thermostat. Assumes no bond constraints are specified for the system.

2.1.34.3 Dependencies

- gdsum
- images
- merge

2.1.34.4 Arguments

integers:		
idnode	input	identity of node on the cube.
imcon	input	periodic boundary condition key. See **images**
mxnode	input	total number of nodes on the cube.
natms	input	total number of atoms in MD cell.
reals:		
engke	output	kinetic energy.
qmass	input	Berendsen temperature decay variable.
sigma	input	system kinetic energy for specified temperature.
tstep	input	simulation time step, δt.
real arrays:		
buffer	work	buffer array (mxbuff). See **merge**.

cell	input	cell vectors (9). See **images**.
fxx	input	x component of force array (mxatms).
fyy	input	y component of force array (mxatms).
fzz	input	z component of force array (mxatms).
vxx	in/out	x velocity array (mxatms).
vyy	in/out	y velocity array (mxatms).
vzz	in/out	z velocity array (mxatms).
weight	input	atomic mass array (mxatms).
xxx	in/out	x coordinate array (mxatms).
yyy	in/out	y coordinate array (mxatms).
zzz	in/out	z coordinate array (mxatms).

2.1.34.5 Comments

Temperature is controlled by the Berendesen thermostat. The velocities are scaled by χ where this is found from:

$$\chi = \left(1 + \frac{\delta t}{QMASS}\left(\frac{T}{\mathcal{T}} - 1\right)\right)^{1/2} \tag{2.9}$$

where $QMASS$ is the decay variable, $\mathcal{T}$ the instantaneous temperature and T the specified system temperature. Usually a value for $QMASS$ of a few picoseconds works well.

2.1.35 NVT_B1

2.1.35.1 Header records

```
      subroutine nvt_b1
     x     (safe,idnode,imcon,mxnode,natms,nscons,engke,qmass,
     x     sigma,tolnce,tstep,vircon,
     x     lashap,lishap,listcon,listme,listin,listot,
     x     buffer,cell,dxt,dxx,dyt,dyy,dzt,dzz,fxx,fyy,fzz,prmcon,
     x     txx,tyy,tzz,uxx,uyy,uzz,vxx,vyy,vzz,weight,xdf,xxt,xxx,
     x     ydf,yyt,yyy,zdf,zzt,zzz)

c
c*********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - verlet leapfrog with Berendsen
c     thermostat.
c
c     parallel replicated data version
c
c     for systems using bond CONSTRAINTS
c
c     copyright - daresbury laboratory  1993
c     author    - t. forester      july 1993
c
c*********************************************************************
c
```

2.1.35.2 Function

Integrates equations of motion subject to the Berendsen thermostat. Assumes no bond constraints are specified for the system.

2.1.35.3 Dependencies

- gdsum
- images
- merge

2.1.35.4 Arguments

integers:

idnode	input	identity of current processor.
imcon	input	periodic boundary condition key. See **images**
mxnode	input	total number of processors.
natms	input	total number of atoms in MD cell.
nscons	input	number of constraints on current processor.

reals:

engke	output	kinetic energy.
qmass	input	Berendsen temperature decay variable.
sigma	input	system kinetic energy for specified temperature.
tolnce	input	tolerance for SHAKE algorithm.
tstep	input	simulation time step, δt.
vircon	output	constraint virial.

integer arrays:

lashap	work	see **rdshake** (mxproc).
lishap	work	see **rdshake** (mxlshp).
listcon	input	indices for constraint bonds (mxcons,3).
listme	input	see **rdshake** (mxatms).
listin	input	see **rdshake** (mxatms).
listot	input	see **rdshake** (mxatms).

real arrays:

buffer	work	buffer array (mxbuff). See **merge**.
cell	input	cell vectors (9). See **images**.
fxx	input	x component of force array (mxatms).
fyy	input	y component of force array (mxatms).
fzz	input	z component of force array (mxatms).
vxx	in/out	x velocity array (mxatms).
vyy	in/out	y velocity array (mxatms).
vzz	in/out	z velocity array (mxatms).
weight	input	atomic mass array (mxatms).
xxx	in/out	x coordinate array (mxatms).
yyy	in/out	y coordinate array (mxatms).
zzz	in/out	z coordinate array (mxatms).

2.1.35.5 Comments

Temperature is controlled by the Berendesen thermostat. The velocities are scaled by χ where this is found from:

$$\chi = \left(1 + \frac{\delta t}{QMASS}\left(\frac{T}{\mathcal{T}} - 1\right)\right)^{1/2} \tag{2.10}$$

where $QMASS$ is the decay variable, $\mathcal{T}$ the instantaneous temperature and T the specified system temperature.

2.1.36 NVT_E0

2.1.36.1 Header records

```
      subroutine nvt_e0
     x     (idnode,imcon,natms,mxnode,engke,sigma,tstep,
     x     buffer,cell,fxx,fyy,fzz,vxx,vyy,vzz,weight,xxx,yyy,zzz)

c
c*********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - verlet leapfrog with Evans
c     thermostat.
c     Comp. Phys. reports 1, 299, (1984)
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1993
c     author    - t. forester   feb 1993
c
c*********************************************************************
c
```

2.1.36.2 Function

The verlet algorithm for integrating Newton's equations of motion subject to Gaussian constraints on the temperature. The thermostat is due to Evans. This routine is called only when **no bond constraints** are specified in the system.

2.1.36.3 Dependencies:

- images
- merge
- gdsum

2.1.36.4 Arguments:

integers:

idnode	input	identity of node on the cube.
imcon	input	periodic boundary condition key. See **images**
mxnode	input	total number of nodes on the cube.
natms	input	total number of atoms in MD cell.

reals:

engke	output	kinetic energy.
sigma	input	system kinetic energy for specified temperature.
tstep	input	simulation time step, δt.

real arrays:

buffer	work	buffer array (mxbuff). See merge.
cell	input	cell vectors (9). See images.
fxx	input	x component of force array (mxatms).
fyy	input	y component of force array (mxatms).
fzz	input	z component of force array (mxatms).
vxx	in/out	x velocity array (mxatms).
vyy	in/out	y velocity array (mxatms).
vzz	in/out	z velocity array (mxatms).
weight	input	atomic mass array (mxatms).
xxx	in/out	x coordinate array (mxatms).
yyy	in/out	y coordinate array (mxatms).
zzz	in/out	z coordinate array (mxatms).

2.1.36.5 Comments

Simulates a canonical ensemble.

2.1.37 NVT_E1

2.1.37.1 Header records

```
      subroutine nvt_e1
     x     (safe,idnode,imcon,mxnode,natms,nscons,engke,
     x     sigma,tolnce,tstep,vircon,
     x     lashap,lishap,listcon,listme,listin,listot,
     x     buffer,cell,dxt,dxx,dyt,dyy,dzt,dzz,fxx,fyy,fzz,prmcon,
     x     txx,tyy,tzz,uxx,uyy,uzz,vxx,vyy,vzz,weight,xdf,xxt,xxx,
     x     ydf,yyt,yyy,zdf,zzt,zzz)

c*********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - verlet leapfrog with Evans
c     thermostat.
c     Comp. Phys. reports 1, 299, (1984)
c
c     parallel replicated data version
c
c     for systems using bond CONSTRAINTS
c     The constraint virial is calculated.
c
c     copyright - daresbury laboratory 1993
c     author    -    t. forester   july 1993
c
c*********************************************************************
c
```

2.1.37.2 Function

Integrates equations of motion using both bond constraints and Gaussian temperature constraints.

2.1.37.3 Dependencies

- gdsum
- gstate
- images
- merge
- shmove

2.1.37.4 Arguments

logicals:

safe	output	flag for convergence of SHAKE.

integers:		
idnode	input	identity of current node.
imcon	input	key for periodic boundary conditions.
mxnode	input	total number of nodes.
natms	input	number of atoms in MD cell.
nscons	input	number of constraints on current node.
reals:		
engke	output	kinetic energy.
sigma	input	system kinetic energy.
tolnce	input	tolerance for SHAKE algorithm.
tstep	input	simulation time-step, δt.
vircon	output	constraint virial.
integer arrays:		
lashap	work	see **rdshake** (mxproc).
lishap	work	see **rdshake** (mxlshp).
listcon	input	indices for constraint bonds (mxcons,3).
listme	input	see **rdshake** (mxatms).
listin	input	see **rdshake** (mxatms).
listot	input	see **rdshake** (mxatms).
real arrays:		
buffer	work	see **merge** (mxbuff).
cell	in/out	MD cell vectors (9).
chge	input	partial charges, (mxatms).
dxt	work	see **rdshake** (mxcons).
dxx	work	see **rdshake** (mxcons).
dyt	work	see **rdshake** (mxcons).
dyy	work	see **rdshake** (mxcons).
dzt	work	see **rdshake** (mxcons).
dzz	work	see **rdshake** (mxcons).
fxx	in/out	x component of force (mxatms).
fyy	in/out	y component of force (mxatms).
fzz	in/out	z component of force (mxatms).
prmcon	input	constraint distances (mxtcon).
txx	work	(mxatms).
tyy	work	(mxatms).
tzz	work	(mxatms).
uxx	work	(mxatms).
uyy	work	(mxatms).
uzz	work	(mxatms).
vxx	in/out	x component of velocity (mxatms).
vyy	in/out	y component of velocity (mxatms).
vzz	in/out	z component of velocity (mxatms).
weight	input	mass (mxatms).
xdf	work	x component of intersite vector (mxxdf).
xxt	work	(mxatms).
xxx	in/out	x coordinates (mxatms).
ydf	work	y component of intersite vector (mxxdf).

yyt	work	(mxatms).
yyy	in/out	y coordinates (mxatms).
zdf	work	z component of intersite vector (mxxdf).
zzt	work	(mxatms).
zzz	in/out	z coordinates (mxatms).

2.1.37.5 Comments

Bond constraints and temperature constraints are solved iteratively.

2.1.38 NVT_H0

2.1.38.1 Header records

```
      subroutine nvt_h0
     x     (idnode,imcon,natms,mxnode,chit,engke,qmass,sigma,tstep,
     x     buffer,cell,fxx,fyy,fzz,vxx,vyy,vzz,weight,xxx,yyy,zzz)

c
c*********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - verlet leapfrog with Hoover
c     thermostat.
c     Phys. Rev A 31, 1695 (1985)
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1993
c     author    - t. forester      feb 1993
c
c*********************************************************************
c
```

2.1.38.2 Function

Integrates equations of motion subject to the Hoover thermostat. Assumes no bond constraints are specified for the system.

2.1.38.3 Dependencies

- gdsum
- images
- merge

2.1.38.4 Arguments

integers:

idnode	input	identity of node on the cube.
imcon	input	periodic boundary condition key. See **images**
mxnode	input	total number of nodes on the cube.
natms	input	total number of atoms in MD cell.

reals:

chit	in/out	friction coefficient for thermostat.
engke	output	kinetic energy.
qmass	input	thermal inertia variable.
sigma	input	system kinetic energy for specified temperature.
tstep	input	simulation time step, δt.

real arrays:

buffer	work	buffer array (mxbuff). See **merge**.
cell	input	cell vectors (9). See **images**.
fxx	input	x component of force array (mxatms).
fyy	input	y component of force array (mxatms).
fzz	input	z component of force array (mxatms).
vxx	in/out	x velocity array (mxatms).
vyy	in/out	y velocity array (mxatms).
vzz	in/out	z velocity array (mxatms).
weight	input	atomic mass array (mxatms).
xxx	in/out	x coordinate array (mxatms).
yyy	in/out	y coordinate array (mxatms).
zzz	in/out	z coordinate array (mxatms).

2.1.38.5 Comments

Temperature is controlled by the Hoover thermostat. The value of the friction coefficient, χ is found from the first-order differential equation in time:

$$\dot{\chi} = \frac{k_B f}{Q}(T - T) \tag{2.11}$$

where k_B is Boltzmann's constant, f the number of degrees of freedom, Q the thermal inertia variable, T the instantaneous temperature and T the specified system temperature. Usually a value of Q equal to the mass of the most abundant molecule in the system works well.

2.1.39 NVT_H1

2.1.39.1 Header records

```
      subroutine nvt_h1
     x     (safe,idnode,imcon,mxnode,natms,nscons,chit,engke,qmass,
     x     sigma,tolnce,tstep,vircon,
     x     lashap,lishap,listcon,listme,listin,listot,
     x     buffer,cell,dxt,dxx,dyt,dyy,dzt,dzz,fxx,fyy,fzz,prmcon,
     x     txx,tyy,tzz,uxx,uyy,uzz,vxx,vyy,vzz,weight,xdf,xxt,xxx,
     x     ydf,yyt,yyy,zdf,zzt,zzz)

c*********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - verlet leapfrog with Hoover
c     thermostat.
c
c     parallel replicated data version
c
c     for systems using bond CONSTRAINTS
c
c     copyright - daresbury laboratory 1993
c     author    -    t. forester   may 1993
c
c*********************************************************************
c
```

2.1.39.2 Function

Integrates equations of motion using both bond constraints and the Hoover thermostat.

2.1.39.3 Dependencies

- gdsum
- gstate
- images
- merge
- shmove

2.1.39.4 Arguments

logicals:		
safe	output	flag for convergence of SHAKE.
integers:		
idnode	input	identity of current node.
imcon	input	key for periodic boundary conditions.

mxnode	input	total number of nodes.
natms	input	number of atoms in MD cell.
nscons	input	number of constraints on current node.
reals:		
chit	in/out	friction coefficient for heat bath.
engke	output	kinetic energy.
qmass	input	thermal inertia variable.
sigma	input	system kinetic energy.
tolnce	input	tolerance for SHAKE algorithm.
tstep	input	simulation time-step, δt.
vircon	output	constraint virial.
integer arrays:		
lashap	work	see **rdshake** (mxproc).
lishap	work	see **rdshake** (mxlshp).
listcon	input	indices for constraint bonds (mxcons,3).
listme	input	see **rdshake** (mxatms).
listin	input	see **rdshake** (mxatms).
listot	input	see **rdshake** (mxatms).
real arrays:		
buffer	work	see **merge** (mxbuff).
cell	in/out	MD cell vectors (9).
chge	input	partial charges, (mxatms).
dxt	work	see **rdshake** (mxcons).
dxx	work	see **rdshake** (mxcons).
dyt	work	see **rdshake** (mxcons).
dyy	work	see **rdshake** (mxcons).
dzt	work	see **rdshake** (mxcons).
dzz	work	see **rdshake** (mxcons).
fxx	in/out	x component of force (mxatms).
fyy	in/out	y component of force (mxatms).
fzz	in/out	z component of force (mxatms).
prmcon	input	constraint distances (mxtcon).
txx	work	(mxatms).
tyy	work	(mxatms).
tzz	work	(mxatms).
uxx	work	(mxatms).
uyy	work	(mxatms).
uzz	work	(mxatms).
vxx	in/out	x component of velocity (mxatms).
vyy	in/out	y component of velocity (mxatms).
vzz	in/out	z component of velocity (mxatms).
weight	input	mass (mxatms).
xdf	work	x component of intersite vector (mxxdf).
xxt	work	(mxatms).
xxx	in/out	x coordinates (mxatms).
ydf	work	y component of intersite vector (mxxdf).
yyt	work	(mxatms).

yyy	in/out	y coordinates (mxatms).
zdf	work	z component of intersite vector (mxxdf).
zzt	work	(mxatms).
zzz	in/out	z coordinates (mxatms).

2.1.39.5 Comments

See NVT_H0 for comments on the theromstat and choice of QMASS. In this routine bond constraints and thermostat adjustments are solved iteratively.

2.1.40 PARLST

2.1.40.1 Header records

```
      subroutine parlst
     x     (newlst,natms,idnode,mxnode,imcon,rcut,delr,lexatm,nexatm,
     x     noxatm,lentry,list,cell,xxx,yyy,zzz,xdf,ydf,zdf)

c
c*********************************************************************
c
c     dl_poly subroutine for constructing the verlet neighbour
c     list based on the brode-ahlrichs decomposition
c
c     parallel replicated data version
c
c     modified for test of nexatm .ne. noxatm at end of loop
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c
c*********************************************************************
c
```

2.1.40.2 Function

Constructs the verlet neighbourhood list based on the Brode-Ahlrichs decomposition.

2.1.40.3 Dependencies

- images

2.1.40.4 Arguments

logicals:		
newlst	input	flag for construction of a new verlet list.
integers:		
natms	input	number of atoms in the MD cell.
idnode	input	identity of the node on the cube.
mxnode	input	total number of nodes on the cube.
imcon	input	control variable for periodic boundary conditions.
reals:		
rcut	input	potential cut off radius.
delr	input	width of the outer shell of the verlet list.
integer arrays:		
lexatm	input	list of interactions to be excluded from the verlet list (msatms,mxexcl).
nexatm	input	number of excluded atoms for each site (msatms).

noxatm	work	running counter for number of excluded atoms found during the construction of the verlet list (msatms).
lentry	output	number of neighbours in Verlet neighbourhood list (msatms).
list	output	the Verlet neighbourhood list (msatms,mxlist).
real arrays:		
cell	input	MD cell vectors (9).
xxx	input	x coordinate (mxatms).
yyy	input	y coordinate (mxatms).
zzz	input	z coordinate (mxatms).
xdf	work	x component of intersite vector (mxxdf).
ydf	work	y component of intersite vector (mxxdf).
zdf	work	z component of intersite vector (mxxdf).

2.1.40.5 Comments

The sites in the cell are distributed evenly across all nodes. Thus the Verlet list on any particular node will contain pairs of indices unique to the node. The Brode-Ahlrichs decomposition arranges the pairs of interactions so that the number of neighbours for each central site is reasonably uniform.

MSATMS must be larger than NATMS/MXNODE. MSATMS is defined in the file **dl_params.inc**.

The number of neighbours for any particular site increases as $(RCUT + DELR)^3$. The number of neighbours must be less than or equal to MXLIST. MXLIST is defined in the **dl_params.inc** file.

2.1.41 PASSCON

2.1.41.1 Header records

```
      subroutine passcon
     x     (idnode,mxnode,natms,nscons,lashap,lishap,listme,
     x     listin,listot,listcon)

c
c*********************************************************************
c
c     dl_poly subroutine for passing information about bond
c     constraints between nodes
c
c     parallel replicated data version assuming direct node-node
c     connection (i.e. this version may be intel specific)
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith august 1992.
c
c*********************************************************************
c
```

2.1.41.2 Function

Passes information about bond constraints between nodes. This routine is only called during the initalisation of the system for the MD simulation and only when constraint dynamics are required.

2.1.41.3 Dependencies

- csend
- error
- gisum
- gstate
- gsync
- irecv
- msgwait

2.1.41.4 Arguments

integers:		
idnode	input	identity of current processor.
mxnode	input	total number of processors.
natms	input	number of atoms in MD cell.

nscons	input	number of constraints handled on current processor.
integer arrays:		
lashap	output	"last atom" marker (mxproc).
lishap	output	"shared atoms" list (mxlshp).
listme	output	number of constraints involving atom that are processed on this node (m
listin	output	number of constraints involving atom that are processed by communicati
listot	output	number of nodes processing constraints involving atom (mxatms).
listcon	input	bond constraint bookkeeping list (mxcons,3).

2.1.41.5 Comments

You mess with this routine at your own peril.

2.1.42 PRIMLST

2.1.42.1 Header records

```
      subroutine primlst
     x      (idnode,mxnode,natms,imcon,rprim,lentry,list,
     x      cell,xxx,yyy,zzz,xdf,ydf,zdf)

c
c***********************************************************************
c
c     dlpoly routine to split interaction list into primary and
c     secondary neighbours for use with multiple timestep method
c
c     copyright daresbury laborartory 1993
c     author -   t. forester february 1993
c
c***********************************************************************
```

2.1.42.2 Function

Classifies sites in the verlet list as either 'primary' or 'secondary' neighbours of the central site for use with multiple time-step methods. Primary neighbours are flagged by a negative entry in the verlet list.

2.1.42.3 Dependencies

- images

2.1.42.4 Arguments

integers:		
idnode	input	identity of current processor.
mxnode	input	total number of processors.
natms	input	number of atoms in the MD cell.
imcon	input	key for periodic boundary conditions.
reals:		
rprim	input	radius of the primary neighbourhood shell.
integer arrays:		
lentry	input	number of neighbours in the Verlet list (msatms).
list	in/out	the Verlet neighbourhood list (msatms,mxlist).
real arrays:		
cell	input	MD cell vectors (9).
xxx	input	x coordinate array (mxatms).
yyy	input	y coordinate array (mxatms).
zzz	input	z coordinate array (mxatms).
xdf	input	x component of intersite vector (mxxdf).
ydf	input	y component of intersite vector (mxxdf).
zdf	input	z component of intersite vector (mxxdf).

2.1.42.5 Comments

2.1.43 QUENCH

2.1.43.1 Header records

```
      subroutine quench
     x     (imcon,idnode,mxnode,natms,nscons,tolnce,
     x     lashap,lishap,listcon,listin,listme,listot,
     x     buffer,cell,dxt,dyt,dzt,uxx,uyy,uzz,vxx,vyy,vzz,
     x     weight,xxt,xxx,yyt,yyy,zzt,zzz)
c
c*********************************************************************
c
c     dl_poly subroutine for quenching the bond energies in the
c     initial structure of a molecule defined by constraints
c
c     copyright - daresbury laboratory 1992
c     author w.smith november 1992
c
c*********************************************************************
c
```

2.1.43.2 Function

Removes velocity component along direction of constraint bonds.

2.1.43.3 Dependencies

- error
- gstate
- images
- shmove
- splice

2.1.43.4 Arguments

integers:

idnode	input	identity of current node.
imcon	input	key for periodic boundary conditions.
mxnode	input	total number of nodes.
natms	input	number of atoms in MD cell.
nscons	input	number of constraints on current node.

reals:

tolnce	input	tolerance for SHAKE algorithm.

integer arrays:

lashap	work	see **rdshake** (mxproc).
lishap	work	see **rdshake** (mxlshp).
listcon	input	indices for constraint bonds (mxcons,3).

listin	input	see rdshake (mxatms).
listme	input	see rdshake (mxatms).
listot	input	see rdshake (mxatms).
real arrays:		
buffer	work	see merge (mxbuff).
cell	in/out	MD cell vectors (9).
dxt	work	see rdshake (mxcons).
dyt	work	see rdshake (mxcons).
dzt	work	see rdshake (mxcons).
uxx	work	(mxatms).
uyy	work	(mxatms).
uzz	work	(mxatms).
vxx	in/out	x component of velocity (mxatms).
vyy	in/out	y component of velocity (mxatms).
vzz	in/out	z component of velocity (mxatms).
weight	input	mass (mxatms).
xxt	work	(mxatms).
xxx	input	x coordinates (mxatms).
yyt	work	(mxatms).
yyy	input	y coordinates (mxatms).
zzt	work	(mxatms).
zzz	input	z coordinates (mxatms).

2.1.43.5 Comments

2.1.44 RDF0

2.1.44.1 Header records

```
      subroutine rdf0
     x     (iatm,ik,rcut,ilist,ltype,lstvdw,irdf,rsqdf)
c
c*********************************************************************
c
c     dl_poly subroutine for accumulating statistic for radial
c     distribution functions.
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1993
c     author    - t. forester     june 1993
c
c*********************************************************************
c
```

2.1.44.2 Functions

Accumulates histograms for radial distribution functions.

2.1.44.3 Dependencies

- none

2.1.44.4 Arguments

integers:		
iatm	input	index of central site.
ik	input	number of neighbours.
reals:		
rcut	input	non-bonded potential cutoff.
integer arrays:		
ilist	input	index of neighbours around central site (mxlist).
ltype	input	index of atomic types (mxatms).
lstvdw	input	key for type of non-bonded potential (mxvdw).
irdf	in/out	histograms for rdfs (mxrdf,mxvdw).
real arrays:		
rsqdf	input	square of interatomic distances (mxxdf).

2.1.44.5 Comments

The actual rdfs are calculated in **rdf1** from data collected in this routine.

2.1.45 RDF1

2.1.45.1 Header records

```
      subroutine rdf1
     x     (lpgr,unqatm,idnode,ntpvdw,numrdf,rcut,volm,ibuff,ltype,
     x     lstvdw,irdf,dens)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating radial distribution functions
c     from accumulated data.
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1993
c     author    - t. forester     june 1993
c
c*********************************************************************
c
```

2.1.45.2 Functions

Calculates and prints radial distribution functions and the number of neighbours.

2.1.45.3 Dependencies

- none

2.1.45.4 Arguments

logicals:		
lpgr	input	flag for printing radial distribution functions.
characters:		
unqatm	input	atomic type names {a8} (mxsite).
integers:		
idnode	input	identity of current processor.
ntpvdw	input	number of non-bonded potentials.
numrdf	input	number of timesteps used for g(r) statistics.
reals:		
rcut	input	non-bonded potential cutoff.
volm	input	volume of MD cell.
integer arrays:		
ibuff	work	buffer array for global summations (mxrdf*mxvdw).
irdf	input	histograms for g(r) (mxrdf,mxvdw).
real arrays:		
dens	input	number density of atomic sites (mxsvdw).

2.1.45.5 Comments

For non periodic systems only the number of neighbours (n(r)) has meaning since an arbitary value needs to be assigned to the volume in order to compute the radial distribution functions. The average number of atoms of type B within a radius r of atoms of type A is given by

$$n(r) = 4\pi \int_0^r g(r)\rho_B r^2 dr \tag{2.12}$$

2.1.46 RDSHAKE

2.1.46.1 Header records

```
      subroutine rdshake
     x     (safe,idnode,imcon,mxnode,natms,nscons,engke,tolnce,tstep,
     x     vircon,lashap,lishap,listcon,listme,listin,listot,
     x     buffer,cell,dxt,dxx,dyt,dyy,dzt,dzz,fxx,fyy,fzz,prmcon,
     x     txx,tyy,tzz,uxx,uyy,uzz,vxx,vyy,vzz,weight,xdf,xxt,xxx,
     x     ydf,yyt,yyy,zdf,zzt,zzz)

c
c*********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - shake/verlet leapfrog.
c
c     parallel replicated data algorithm
c
c     version 2 : include constraint contribution to the virial
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith august 1992.
c
c*********************************************************************
c
```

2.1.46.2 Function

Controls the MD cycle of evaluating forces, integrating equations of motion and accumulating statistics.

2.1.46.3 Dependencies

- gdsum
- gstate
- images
- merge
- shmove
- splice

2.1.46.4 Arguments

logicals:		
safe	output	flag for convergence of SHAKE.
integers:		
idnode	input	identity of current node.
imcon	input	key for periodic boundary conditions.
mxnode	input	total number of nodes.
natms	input	number of atoms in MD cell.
nscons	input	number of constraints on current node.
reals:		
engke	output	kinetic energy.
tolnce	input	tolerance for SHAKE algorithm.
tstep	input	simulation time-step, δt.
vircon	output	constraint virial.
integer arrays:		
lashap	work	see **rdshake** (mxproc).
lishap	work	see **rdshake** (mxlshp).
listcon	input	indices for constraint bonds (mxcons,3).
listme	input	see **rdshake** (mxatms).
listin	input	see **rdshake** (mxatms).
listot	input	see **rdshake** (mxatms).
real arrays:		
buffer	work	see **merge** (mxbuff).
cell	in/out	MD cell vectors (9).
chge	input	partial charges, (mxatms).
dxt	work	see **rdshake** (mxcons).
dxx	work	see **rdshake** (mxcons).
dyt	work	see **rdshake** (mxcons).
dyy	work	see **rdshake** (mxcons).
dzt	work	see **rdshake** (mxcons).
dzz	work	see **rdshake** (mxcons).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
prmcon	input	constraint distances (mxtcon).
txx	work	(mxatms).
tyy	work	(mxatms).
tzz	work	(mxatms).
uxx	work	(mxatms).
uyy	work	(mxatms).
uzz	work	(mxatms).
vxx	in/out	x component of velocity (mxatms).
vyy	in/out	y component of velocity (mxatms).
vzz	in/out	z component of velocity (mxatms).
weight	input	mass (mxatms).
xdf	work	x component of intersite vector (mxxdf).
xxt	work	(mxatms).
xxx	in/out	x coordinates (mxatms).
ydf	work	y component of intersite vector (mxxdf).
yyt	work	(mxatms).
yyy	in/out	y coordinates (mxatms).

zdf	work	z component of intersite vector (mxxdf).
zzt	work	(mxatms).
zzz	in/out	z coordinates (mxatms).

2.1.46.5 Comments

2.1.47 RESULT

2.1.47.1 Header records

```
      subroutine result
     x     (cfgname,atmnam,unqatm,lgofr,lpgr,idnode,imcon,mxnode,
     x     natms,nstep,ntpatm,numacc,numrdf,chit,rcut,volm,
     x     irdf,ibuff,lstvdw,buffer,
     x     cell,dens,fxx,fyy,fzz,ravval,ssqval,stkval,stpval,
     x     sumval,vxx,vyy,vzz,xxx,yyy,zumval,zzz,xx0,yy0,zz0)

c
c*********************************************************************
c
c     dl_poly subroutine for writing simulation summary and
c     saving the restart data
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith dec 1992.
c
c*********************************************************************
c
```

2.1.47.2 Function

Writes out result file and calls for storing of final configuration.

2.1.47.3 Dependencies

- revive
- timchk

2.1.47.4 Arguments

characters:

cfgname	input	configuration name [a80].
atmnam	input	site labels [a8] (mxatms).
unqatm ¿ inpu	unique atom	labels (mxsite).

logicals:

lgofr	input	flag for construction of rdf histograms.
lpgr	input	flag to print rdfs.

integers:

idnode	input	identity of current processor.
imcon	input	key for periodic boundary conditions.
mxnode	input	total number of processors.
natms	input	number of atoms in MD cell.
nstep	input	number of completed MD steps.
ntpatm	input	number of unique atom types.
numacc	input	number of steps used to accumulate statistics.

numrdf	input	number of configurations used for rdf histograms.
reals:		
chit	input	friction coefficient for Hoover thermostat.
rcut	input	radial cut off.
volm	input	volume of cell.
real arrays:		
buffer	work	work array (mxbuff).
cell	input	MD cell vectors (9).
dens	input	number density of unique atom types (mxsvdw).
fxx	input	x component of force (mxatms).
fyy	input	y component of force (mxatms).
fzz	input	z component of force (mxatms).
ravval	input	rolling averages (mxnstk).
ssqval	in/out	rms fluctuations (mxnstk).
stkval	input	statistics file for rolling averages (mxstak,mxnstk).
stpval	input	statistics file (mxnstk).
sumval	input	statistics file (mxnstk).
vxx	input	x component of velocity (mxatms).
vyy	input	y component of velocity (mxatms).
vzz	input	z component of velocity (mxatms).
xxx	input	x coordinate (mxatms).
yyy	input	y coordinate (mxatms).
zumval	input	statistics file (mxnstk).
zzz	input	z coordinate (mxatms).
xx0	input	x component of atomic displacement (mxatms).
yy0	input	y component of atomic displacement (mxatms).
zz0	input	z component of atomic displacement (mxatms).

2.1.47.5 Comments

2.1.48 REVIVE

2.1.48.1 Header records

```
      subroutine revive
     x     (lgofr,cfgname,atmnam,idnode,imcon,mxnode,natms,nstep,
     x     numacc,numrdf,chit,irdf,ibuff,
     x     buffer,cell,fxx,fyy,fzz,ravval,ssqval,stkval,stpval,
     x     sumval,vxx,vyy,vzz,xxx,yyy,zumval,zzz,xx0,yy0,zz0)

c
c*********************************************************************
c
c     dl_poly subroutine for writing restart files at job termination
c     or at selected intervals in simulation
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith dec 1992.
c
c*********************************************************************
c
```

2.1.48.2 Function

Dumps restart data to disk.

2.1.48.3 Dependencies

- merge

2.1.48.4 Arguments

logicals:		
lgofr	input	flag for radial distribution functions.
characters:		
cfgname	input	configuration name [a80].
atmnam	input	atomic labels [a8] (mxatms)
integers:		
idnode	input	identity of current processor.
imcon	input	periodic boundary key.
mxnode	input	total number of processors.
natms	input	number of atoms in MD cell.
nstep	input	time-step counter.
numacc	input	number of steps used for statistics.
numrdf	input	number of steps used for rdfs.
reals:		
chit	input	friction coefficient for Hoover thermostat.
integer arrays:		

irdf	input	radial distribution function histograms (mxrdf,mxvdw).
ibuff	work	integer buffer array (mxrdf*mxvdw).
real arrays:		
buffer	work	work arrat for merge (mxbuff).
cell	input	cell vectors (9).
fxx	input	x component of force (mxatms).
fyy	input	y component of force (mxatms).
fzz	input	z component of force (mxatms).
ravval	input	statistics array (mxnstk).
ssqval	input	statistics array (mxnstk).
stkval	input	statistics array (mxstak,mxnstk).
stpval	input	statistics array (mxnstk).
sumval	input	statistics array (mxnstk).
vxx	input	x component of velocity (mxatms).
vyy	input	y component of velocity (mxatms).
vzz	input	z component of velocity (mxatms).
xxx	input	x coordinate (mxatms).
yyy	input	y coordinate (mxatms).
zumval	input	statistics array (mxnstk).
zzz	input	z coordinate (mxatms).
xx0	input	increment in x coordinate (mxatms).
yy0	input	increment in y coordinate (mxatms).
zz0	input	increment in z coordinate (mxatms).

2.1.48.5 Comments

2.1.49 SHMOVE

2.1.49.1 Header records

```
      subroutine shmove
     x     (idnode,mxnode,natms,lashap,lishap,xxt,yyt,zzt,
     x      txx,tyy,tzz,buffer)

c
c*********************************************************************
c
c     dl_poly subroutine for passing coordinate updates between
c     nodes during the shake iteration cycle
c
c     parallel replicated data algorithm intel specific version
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith august 1992.
c
c*********************************************************************
c
```

2.1.49.2 Function

Passes coordinate updates between nodes during the shake iteration cycle.

2.1.49.3 Dependencies

- csend
- gsync
- msgwait

2.1.49.4 Arguments

integers:		
idnode	input	identity of current processor.
mxnode	input	total number of processors.
natms	input	number of atoms in MD cell.
integer arrays:		
lashap	input	Outgoing transfer buffer (mxproc).
lishap	input	Incoming transfer buffer (mxlshp).
real arrays:		
xxt	in/out	increment to x coordinate (mxatms).
yyt	in/out	increment to y coordinate (mxatms).
zzt	in/out	increment to z coordinate (mxatms).
txx	work	(mxatms).
tyy	work	(mxatms).
tzz	work	(mxatms).

buffer	work	communications buffer (mxbuff).

2.1.49.5 Comments

Messing with this routine may seriously damage the health of DL_POLY.

2.1.50 SIMDEF

2.1.50.1 Header records

```
      subroutine simdef
     x     (lfcap,lgofr,loptim,lpgr,ltraj,ltscal,lzeql,sysname,
     x     idnode,intsta,istraj,keyens,keyfce,keyres,keytrj,kmax1,
     x     kmax2,kmax3,multt,nstack,nstbgr,nstbpo,nstbts,nsteql,nstraj,
     x     nstrun,alpha,delr,epsq,pmass,press,qmass,rcut,rprim,temp,
     x     timcls,timjob,tolnce,tstep)

c
c*********************************************************************
c
c     dl_poly subroutine for reading in the simulation control
c     parameters
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith july 1992.
c
c     modified
c     author   - t. forester       may 1993
c
c*********************************************************************
c
```

2.1.50.2 Function

Reads in control variables for the job from the CONTROL file.

2.1.50.3 Dependencies

- none

2.1.50.4 Arguments

logicals:

lfcap	output	flag for capping forces during equilibration.
lgofr	output	flag for calculating radial distribution functions.
loptim	output	flag for O K structure optimization.
lpgr	output	flag for printing radial distribution functions.
ltraj	output	flag for dumping trajectory data.
ltscal	output	flag for temperature scaling.
lzeql	output	flag for accumulating statitics during equilibration.

characters:

sysname	output	system name (a80).

integers:

idnode	input	identity of current node.
intsta	output	interval for accumulating statistics.
istraj	output	interval for dumping trajectories.

keyens	output	key for selecting ensemble.
keyfce	output	key for non-bonded force field.
keyres	output	key for restart of MD run.
keytrj	output	key for trajectory information.
kmax1	output	number of reciprocal space vectors in a direction.
kmax2	output	number of reciprocal space vectors in b direction.
kmax3	output	number of reciprocal space vectors in c direction.
multt	output	size of multiple time-step interval.
nstack	output	size of stack for rolling averages.
nstbgr	output	interval for collecting rdf data.
nstbpo	output	interval for printing out statistical data.
nstbts	output	interval for temperature scaling in equiliration mode.
nsteql	output	number of time-steps in run for equilibation.
nstraj	output	timestep at which dumping of trajectory information begins.
nstrun	output	number of time-steps for MD run.
reals:		
alpha	output	Ewald convergence variable.
delr	output	border width for Verlet list.
epsq	output	relative dielectric constant.
pmass	output	piston mass for pressure control.
press	output	simulation pressure.
qmass	output	coupling parameter for heat bath.
rcut	output	potential cut-off radius.
rprim	output	radius of primary cut-off shell.
temp	output	simulation temperature.
timcls	output	time, in seconds, allocated for closing the job.
timjob	output	total time in seconds allocated for the job.
tolnce	output	tolerance for SHAKE algorithm.
tstep	output	simulation time-step, δt.

2.1.50.5 Comments

See the section on the description of the input file CONTROL.

2.1.51 SPLICE

2.1.51.1 Header records

```
      subroutine splice
     x      (idnode,mxnode,natms,listme,listot,xxx,yyy,zzz,buffer)
c
c*********************************************************************
c
c     dl_poly subroutine for splicing together coordinate arrays
c     across a number of processors during shake algorithm
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1993
c     author    - w. smith       march 1993
c
c     second version of splice
c
c
c*********************************************************************
c
```

2.1.51.2 Function

Updates coordinates on all nodes of those sites involved in constraint bonds once the SHAKE cycle has been completed.

2.1.51.3 Dependencies

- gdsum

2.1.51.4 Arguments

integers:		
idnode	input	identity of current processor.
mxnode	input	total number of processors.
natms	input	number of sites involved in constraints.
integer arrays:		
listme	input	number of constraints a site is involved in on the current processor (mxatms).
listot	input	number of processors handling constraints involving a particular site (mxatms)
real arrays:		
xxx	in/out	x coordinate array (natms).
yyy	in/out	y coordinate array (natms).
zzz	in/out	z coordinate array (natms).
buffer	work	work array for global sum (mxbuff).

2.1.51.5 Comments

The arrays 'listme' and 'listot' are created in **passcon**. Only sites involved in bond constraints are updated.

2.1.52 SRFRCE

2.1.52.1 Header records

```
      subroutine srfrce
     x     (iatm,ik,engsrp,virsrp,rcut,dlrpot,ilist,ltype,
     x      lstvdw,rsqdf,xdf,ydf,zdf,fxx,fyy,fzz,vvv,ggg)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating short range force and
c     potential energy terms using verlet neighbour list
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith       march 1992
c
c     version 3
c     author    - t. forester    june  1993
c
c*********************************************************************
c
```

2.1.52.2 Function

Calculates non-bonded forces and potential energies using the Verlet neigbour list. Two variants are available: **srfrce3pt** and **srfrce4pt** which use 3 and 4 point interpolation schemes respectively.

2.1.52.3 Dependencies

- none

2.1.52.4 Arguments

integers:		
iatm	input	index of the central site.
ik	input	number of entries in the verlet list.
reals:		
engsrp	output	the non-bonded potential energy.
virsrp	output	the non-bonded virial.
rcut	input	the non-bonded potential cutoff.
dlrpot	input	tabulation interval for interpolation tables.
integer arrays:		
ilist	input	the Verlet neighbour list (mxlist).
ltype	input	atom type (mxatms).
lstvdw	input	index number of non-bonded potential (mxvdw).
real arrays:		
rsqdf	input	square of inter site vector length (mxxdf).

xdf	input	x coordinate for inter site vector (mxxdf).
ydf	input	y coordinate for inter site vector (mxxdf).
zdf	input	z coordinate for inter site vector (mxxdf).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
vvv	input	interpolation table for potential energy (mxgrid,mxvdw).
ggg	input	interpolation table for force (mxgrid,mxvdw).

2.1.52.5 Comments

The relative merits of 3 and 4 point interpolation are as follows: 4 point interpolation may permit a smaller number of grid points to be used in the interpolation tables thus saving on memory requirements. 3 point interpolation is considerable quicker than 4 point interpolation and in some cases may be more accurate than 4 point interpolation for a sufficently small grid spacing (ie. for a sufficiently large number of grid points).

If in doubt we recommend use of the subroutine **srfrce3pt** .

Alternatively, interpolation may be carried out in r-squared space - see **srfrce_rsq**.

2.1.53 SRFRCE_RSQ

2.1.53.1 Header records

```
      subroutine srfrce
     x     (iatm,ik,engsrp,virsrp,rcut,dlrpot,ilist,ltype,
     x      lstvdw,rsqdf,xdf,ydf,zdf,fxx,fyy,fzz,vvv,ggg)
c
c*********************************************************************
c
c     dl_poly subroutine for calculating short range force and
c     potential energy terms using verlet neighbour list
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith       march 1992
c
c     modifed for tabulation in r2 space not r space
c     author    - t. forester    march 1993
c
c*********************************************************************
c
```

2.1.53.2 Function

Calculates non-bonded forces and potential energies using the Verlet neigbour list. Two variants are available: **srfrce3pt_rsq** and **srfrce4pt_rsq** which use 3 and 4 point interpolation schemes in r-squared space respectively.

2.1.53.3 Dependencies

- none

2.1.53.4 Arguments

integers:		
iatm	input	index of the central site.
ik	input	number of entries in the verlet list.
reals:		
engsrp	output	the non-bonded potential energy.
virsrp	output	the non-bonded virial.
rcut	input	the non-bonded potential cutoff.
dlrpot	input	tabulation interval for interpolation tables.
integer arrays:		
ilist	input	the Verlet neighbour list (mxlist).
ltype	input	atom type (mxatms).
lstvdw	input	index number of non-bonded potential (mxvdw).
real arrays:		
rsqdf	input	square of inter site vector length (mxxdf).

xdf	input	x coordinate for inter site vector (mxxdf).
ydf	input	y coordinate for inter site vector (mxxdf).
zdf	input	z coordinate for inter site vector (mxxdf).
fxx	output	x component of force (mxatms).
fyy	output	y component of force (mxatms).
fzz	output	z component of force (mxatms).
vvv	input	interpolation table for potential energy (mxgrid,mxvdw).
ggg	input	interpolation table for force (mxgrid,mxvdw).

2.1.53.5 Comments

Interpolation in r-squared space has the advantage that it avoids the use of the square root function. See the entries for **forgen** and **forgen_rsq** for further discussion.

2.1.54 STATIC

2.1.54.1 Header records

```
      subroutine static
     x     (lzeql,cfgname,idnode,intsta,imcon,natms,nstack,nstep,nsteql,
     x     ntcons,ntpatm,numacc,mxnode,degfre,engang,engbnd,engcpe,
     x     engdih,engke,engsrp,engunit,stpcfg,stpeng,stpeth,stpprs,
     x     stptmp,stpvir,stpvol,tstep,virbnd,vircon,vircpe,virsrp,
     x     width,ltype,numtyp,buffer,cell,chge,fxx,fyy,fzz,ravval,
     x     ssqval,stkval,stpval,sumval,vxx,vyy,vzz,xxx,yyy,zzz,zumval,
     x     xx0,yy0,zz0)

c
c*********************************************************************
c
c     dl_poly subroutine for accumulating periodic data during the
c     molecular dynamics simulation and computing the rolling averages
c
c     copyright daresbury laboratory 1992
c
c     author - w. smith       august 1992
c
c*********************************************************************
c
```

2.1.54.2 Function

Accumulates periodic data during the simulations and computes rolling averages.

2.1.54.3 Dependencies

- dcell
- diffsn0
- diffsn1

2.1.54.4 Arguments

logicals:		
lzeql	input	flag for accumulating statistics during equilibration.
characters:		
cfgname	input	configuration name
integers:		
idnode	input	identity of current processor.
intsta	input	interval for collection of statistics.
imcon	input	key for periodic boundary conditions.
natms	input	number of sites in the MD cell.
nstack	input	number of stacked variables.

nstep	input	number of MD time steps completed.
nsteql	input	number od equilibration steps required.
ntcons	input	total number of constraints.
numacc	in/out	number of time steps used for statistics.
mxnode	input	total number of processors in use.
reals:		
degfre	input	thermodynamic degrees of freedom.
engang	input	current valence angle energy.
engbnd	input	current chemical bond energy.
engcpe	input	current coulombic energy.
engdih	input	current dihedral interaction energy.
engke	input	current kinetic energy.
engsrp	input	current short-range potential energy.
engunit	input	energy unit.
stpcfg	output	current total configurational energy.
stpeng	output	current total energy.
stpeth	output	current enthalpy.
stpprs	output	current pressure.
stptmp	output	current temperature.
stpvir	output	current virial.
stpvol	output	current cell volume.
tstep	input	MD timestep.
virbnd	input	current chemical bond virial.
vircpe	input	current coulombic virial.
virsrp	input	current short-range potential virial.
width	input	shortest distance across the MD cell.
integer arrays:		
ltype	input	atomic type (mxatms)
numtyp	input	number atoms of given type (mxsvdw)
real arrays:		
buffer	work	(mxbuff).
cell	input	MD cell vectors (9).
chge	input	partial charges.
fxx	input	x component of force (mxatms).
fyy	input	y component of force (mxatms).
fzz	input	z component of force (mxatms).
ravval	in/out	rolling averages (mxnstk).
ssqval	in/out	sum of squares of statistcal data (mxnstk).
stkval	in/out	statistical data (mxstak,mxnstk).
stpval	output	current values of statistical data (mxnstk).
sumval	in/out	sum of statistical data (mxnstk).
vxx	input	x component of velocity (mxatms).
vyy	input	y component of velocity (mxatms).
vzz	input	z component of velocity (mxatms).
zumval	in/out	sum of rolling average statistical data (mxnstk).
xx0	in/out	x displacement from start of simulation (mxatms)
yy0	in/out	y displacement from start of simulation (mxatms)

zz0	in/out	z displacement from start of simulation (mxatms)

2.1.54.5 Comments

Atomic coordinates, velocities, forces and charges are passed into this subroutine for users who may wish to use such information for the evaluation of additional properties in the statistics files.

2.1.55 STRIP

2.1.55.1 Header records

```
      subroutine strip(string,length)

c*********************************************************************
c
c     DL_POLY routine to strip blanks from start of a string
c     maximum length is 255 characters
c
c     copyright daresbury laboratory 1993
c     author   t.forester       july 1993
c
c*********************************************************************
```

2.1.55.2 Function

Strips blanks from the start of an alphanumeric string.

2.1.55.3 Dependencies

- none

2.1.55.4 Arguments

character:

string	in/out	alphanumeric string

integers:

length	input	length of string (max value = 255)

2.1.55.5 Comments

2.1.56 SYSDEF

2.1.56.1 Header records

```
      subroutine sysdef
     x     (molnam,sitnam,unqatm,idnode,keyfce,nangle,natms,
     x     nbonds,nconst,ndihed,nsite,ntpatm,ntpmls,ntpvdw,
     x     alpha,dlrpot,engunit,rcut,keyang,keybnd,keydih,lstang,
     x     lstbnd,lstcon,lstdih,lstvdw,ltpsit,ltpvdw,numang,
     x     numbonds,numcon,numdih,nummols,numsit,chgsit,erc,fer,
     x     ggg,prmang,prmbnd,prmcon,prmdih,prmvdw,vvv,wgtsit)

c
c*********************************************************************
c
c     dl_poly subroutine for reading in the molecular specifications
c     of the system to be simulated
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith may 1992.
c
c*********************************************************************
c
```

2.1.56.2 Function

Reads in molecular specifications of the system from the FIELD file.

2.1.56.3 Dependencies

- error
- lowcase
- strip

2.1.56.4 Arguments

characters:

molnam	output	molecule names [a40] (mxtmls).
sitnam	output	site names [a40] (mxsite).
unqatm	output	unique atom type names (mxsite).

integers:

idnode	input	identity of current processor.
keyfce	input	key for non-bonded potential.
nangle	output	total number of valence angle interactions.
natms	output	total number of atoms in the MD cell.
nbonds	output	total number of chemical bonds.
nconst	output	total number of bond constraints.
ndihed	output	total number of dihedral interactions.

nsite	output	number of defined sites.
ntpatm	output	number of unique types of atoms.
ntpmls	output	number of unique types of molecule.
ntpvdw	output	number of non-bonded potentials.
reals:		
alpha	input	Ewald sum convergence variable.
dlrpot	output	distance increment for tabulation.
engunit	output	unit of energy relative to internal units.
rcut	input	non-bonded potential cut-off.
integer arrays:		
ltpvdw	input	key for non-bonded potentials (mxvdw).
real arrays:		
prmvdw	input	variables for non-bonded potentials (mxvdw,5).
vvv	output	interpolation table for non-bonded potentials (mxgrid,mxvdw).
ggg	output	interpolation table for non-bonded force (mxgrid,mxvdw).

2.1.56.5 Comments

2.1.57 TIMCHK

2.1.57.1 Header records

```
      subroutine timchk(ktim,time)
c
c***********************************************************************
c
c     dl_poly timing routine giving time elapsed in seconds
c
c     note: this routine uses the intel specific MCLOCK, GSYNC
c           and MYNODE routines
c
c     note: TIMCHK must be called concurrently from all nodes
c           otherwise the parallel program will deadlock
c
c     copyright daresbury laboratory 1992.
c     author - w. smith july 1992
c
c***********************************************************************
c
```

2.1.57.2 Function

Gives the elapsed cpu time in seconds.

2.1.57.3 Dependencies

- gsync
- mclock
- mynode

2.1.57.4 Arguments

integers:		
ktim	input	printing key
reals:		
time	output	elapsed cpu time

2.1.57.5 Comments

"ktim" > 0 will result in a line being printed in the OUTPUT file stating the elapsed cpu time.

This version is INTEL specific. The subroutine **cxa.timchk** can be used in place of this routine on CONVEX machines.

2.1.58 TRAJECT

2.1.58.1 Header records

```
      subroutine traject
     x     (ltraj,cfgname,atmnam,idnode,imcon,istraj,levtrj,natms,
     x     nstraj,nstep,cell,xxx,yyy,zzz,vxx,vyy,vzz,fxx,fyy,fzz)

c
c*********************************************************************
c
c     dl_poly subroutine for writing history file at selected
c     intervals in simulation
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith dec 1992.
c
c*********************************************************************
c
```

2.1.58.2 Function

Writes out the HISTORY file.

2.1.58.3 Dependencies

- none

2.1.58.4 Arguments

logicals:		
ltraj	input	flag for writing of HISTORY file.
characters:		
cfgname	input	configuration name [a80].
atmnam	input	atom names [a8], (mxatms).
integers:		
idnode	input	identity of current processor.
imcon	input	periodic boundary key.
istraj	input	interval for writing file.
levtrj	input	trajectory file information key.
natms	input	number of atoms in MD cell.
nstraj	input	time-step at which writing of file commences.
nstep	input	current time-step.
real arrays:		
cell	input	cell dimensions (9).
xxx	input	x coordinates (mxatms).
yyy	input	y coordinates (mxatms).
zzz	input	z coordinates (mxatms).
vxx	input	x component of velocity (mxatms).
vyy	input	y component of velocity (mxatms).

vzz	input	z component of velocity (mxatms).
fxx	input	x component of force (mxatms).
fyy	input	y component of force (mxatms).
fzz	input	z component of force (mxatms).

2.1.58.5 Comments

2.1.59 VERLET

2.1.59.1 Header records

```
      subroutine verlet
     x     (idnode,mxnode,natms,imcon,tstep,engke,weight,cell,
     x     xxx,yyy,zzz,vxx,vyy,vzz,fxx,fyy,fzz,buffer)

c
c***********************************************************************
c
c     dl_poly subroutine for integrating newtonian equations of
c     motion in molecular dynamics - verlet leapfrog.
c
c     parallel replicated data version
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith march 1992.
c
c***********************************************************************
c
```

2.1.59.2 Function

The verlet algorithm for integrating Newton's equations of motion when there are no bond constraints specified in the system.

2.1.59.3 Dependencies:

- images
- merge
- gdsum

2.1.59.4 Arguments:

integers:		
idnode	input	identity of node on the cube.
mxnode	input	total number of nodes on the cube.
natms	input	total number of sites in the simulated system.
imcon	input	periodic boundary condition key. See **images**
reals:		
tstep	input	simulation time step, δt.
engke	output	kinetic energy.
real arrays:		
weight	input	atomic mass array (mxatms).
cell	input	cell vectors (9). See **images**.
xxx	in/out	x coordinate array (mxatms).
yyy	in/out	y coordinate array (mxatms).

zzz	in/out	z coordinate array (mxatms).
vxx	in/out	x velocity array (mxatms).
vyy	in/out	y velocity array (mxatms).
vzz	in/out	z velocity array (mxatms).
fxx	input	x component of force array (mxatms).
fyy	input	y component of force array (mxatms).
fzz	input	z component of force array (mxatms).
buffer	work	buffer array (mxbuff). See **merge**.

2.1.59.5 Comments

On calling this subroutine xxx, yyy, zzz hold values corresponding to $t - \delta t$; vxx, vyy, vzz hold value corresponding to $t - \frac{1}{2}\delta t$; fxx, fyy and fzz hold values corresponding to $t - \delta t$.

On leaving the routine xxx, yyy, zzz hold values corresponding to t, and vxx,vyy and vzz hold values corresponding to $t + \frac{1}{2}\delta t$

2.1.60 VERTEST

2.1.60.1 Header records

```
      subroutine vertest
     x     (newlst,idnode,mxnode,imcon,natms,delr,cell,xxx,yyy,zzz,
     x     xdf,ydf,zdf)

c*********************************************************************
c
c     DL_POLY subroutime to test for updating of Verlet list
c     replicated data version
c
c     copyright daresbury laboratory 1993
c     author -       t. forester may 1993
c
c*********************************************************************
c
```

2.1.60.2 Function

Tests if the Verlet list needs to be (re)constructed. The list is constructed at the beginning of a job and reconstructed whenever any two particles have moved a distance greater than (DELR/2) since the most recent construction of the list.

2.1.60.3 Dependencies

- images
- gstate

2.1.60.4 Arguements

logicals:		
newlst	output	flag for construction of new Verlet list.
integers:		
idnode	input	identity of current processor.
mxnode	input	total number of processors in use.
imcon	input	key for periodic boundary conditions.
natms	input	number of atoms in the MD cell.
reals:		
delr	input	border width for verlet neighbourhood.
real arrays:		
cell	input	components of MD cell vectors (9).
xxx	input	x coordinate array (mxatms).
yyy	input	y coordinate array (mxatms).
zzz	input	z coordinate array (mxatms).
xdf	work	x component of displacement vector (mxxdf).
ydf	work	y component of displacement vector (mxxdf).
zdf	work	z component of displacement vector (mxxdf).

2.1.60.5 Comments

2.1.61 VSCALE

2.1.61.1 Header records

```
      subroutine vscale(idnode,imcon,natms,sigma,weight,vxx,vyy,vzz
     x      ,xxx,yyy,zzz)
c
c*********************************************************************
c
c     dl_poly subroutine for scaling the velocity arrays to the
c     desired temperature
c
c     version 2
c     zeroes angular momentum in non-periodic system.
c
c     copyright daresbury laboratory 1992.
c     author - w.smith july 1992
c
c     version 2 - t. forester  july 1993.
c
c*********************************************************************
c
```

2.1.61.2 Function

Scales velocities to give desired temperature. Also zeroes angular momentum in non-periodic systems.

2.1.61.3 Dependencies

- invert

2.1.61.4 Arguments

integers:		
idnode	input	identity of current processor.
imcon	input	periodic boundary key.
natms	input	number of atoms in MD cell.
reals:		
sigma	input	required kinetic energy.
real arrays:		
weight	input	mass (mxatms).
vxx	in/out	x component of velocity (mxatms).
vyy	in/out	y component of velocity (mxatms).
vzz	in/out	z component of velocity (mxatms).
xxx	input	x coordinate (mxatms).
yyy	input	y coordinate (mxatms).
zzz	input	z coordinate (mxatms).

2.1.61.5 Comments

Chapter 3

DL_POLY Data Files

3.1 Description of the INPUT files

DL_POLY requires four input files named "CONTROL", "CONFIG", "FIELD" and "REVOLD". The first three files are mandatory, while REVOLD is required only if the job represents a restart of a previous job. In the following sections we describe the form and content of these files.

3.1.1 CONTROL

The CONTROL file contains control variables for running a job. It is read in the subroutine **simdef**. The CONTROL file for DL_POLY test case 1 is shown below:

```
DL_POLY PROGRAM TEST CASE 1
     10000      2000         0                    NSTRUN,NSTEQL,KEYRES
         0     .TRUE.        10                    KEYENS,LTSCAL,NSTBTS
    .FALSE.    .FALSE.        0                    LGOFR,LPGR,NSTBGR
     .TRUE.    .FALSE.       100                   LZEQL,LOPTIM,NSTBPO
       100        10                               NSTACK,INTSTA
    .FALSE.      1000        10         0          LTRAJ,NSTRAJ,ISTRAJ,KEYTRJ
         1     .FALSE.                              KEYFCE,LFCAP
         1   1.000E-3     293.0       0.0          MULTT,TSTEP,TEMP,PRESS
       0.0        8.0       0.5                    RPRIM,RCUT,DELR
       0.0          0         0         0          ALPHA,KMAX1,KMAX2,KMAX3
       0.0        0.0       1.0                    QMASS,PMASS,EPSQ
    0.0001                                         TOLNCE
  3.600E03   1.000E02                              TIMJOB,TCLOSE
```

3.1.1.1 Format

The file is formatted, and all variables require 10 characters. Thus integers are formatted as "i10', logicals as "l10", and reals as "f10.0". The only exception is the header record which is formatted as 80 alphanumeric characters.

3.1.1.2 Definitions of variables

record 1

header	a80	alphanumeric string describing the system.
record 2		
nstrun	integer	total number of time-steps for MD run.
nsteql	integer	number of time-steps for equilibration.
keyres	integer	key for restart of MD run.
record 3		
keyens	integer	key for selecting ensemble.
ltscal	logical	flag for temperature scaling during equilibration steps.
nstbts	integer	interval for temperature scaling in equilibration mode.
record 4		
lgofr	logical	flag for constructing histograms for radial distribution functions.
lpgr	logical	flag for printing radial distribution functions.
nstbgr	integer	interval for collection of radial distribution function statistics.
record 5		
lzeql	logical	flag for conducting an equilibration period.
loptim	logical	flag for 0 K structure optimization.
nstbpo	integer	interval for printing out statistical data.
record 6		
nstack	integer	size of stack for rolling averages.
intsta	integer	interval for accumulating statistics.
record 7		
ltraj	logical	flag for dumping trajectory data.
nstraj	integer	time-step at which dumping of trajectory information begins.
istraj	integer	interval at which trajectory information is written.
keytrj	integer	key for trajectory information.
record 8		
keyfce	integer	key for non-bonded force field.
lfcap	logical	flag to cap forces during equilibration.
record 9		
multt	integer	multiple time-step interval. Use 1 for a single time-step.
tstep	real	simulation time-step, δt, in units of picoseconds.
temp	real	simulation temperature (in units of Kelvin).
press	real	simulation pressure in units of 163.9 atm. (Not currently in use.)
record 10		
rprim	real	radius of primary cut-off shell.
rcut	real	potential cut-off radius.
delr	real	border width for Verlet list.
record 11		
alpha	real	Ewald sum convergence variable.
kmax1	integer	number of reciprocal space vectors in a direction.
kmax2	integer	number of reciprocal space vectors in b direction.
kmax3	integer	number of reciprocal space vectors in c direction.
record 12		
qmass	real	coupling variable for heat bath (Hoover thermostat).
pmass	real	piston mass for pressure control (not currently used).
epsq	real	relative dielectric constant.

record 13		
tolnce	real	tolerance for SHAKE algorithm.
record 14		
timjob	real	total time in seconds allocated for the job.
timcls	real	time in seconds allocated for closing the job.

3.1.1.3 Guide to integer keys

Table 3.1: Restart key (record 2)

KEYRES	meaning
0	start new simulation from CONFIG file, and assign velocities from Gaussian distribution.
1	continue current simulation.
2	start new simulation from CONFIG file, and rescale velocities to desired temperature.

Table 3.2: Ensemble key (record 3)

KEYENS	meaning
0	Microcanonical ensemble (NVE)
1	Gaussian temperature constraints
2	Canonical ensemble using Hoover Thermostat (NVT)

Table 3.3: Trajectory file key (record 7)

KEYTRJ	meaning
0	write coordinates
1	write coordinates and velocities
2	write coordinates, velocities and forces

Table 3.4: Non-bonded force key (record 8)

KEYFCE	meaning
odd	evaluate short-range potentials and electrostatics
even	evaluate Electrostatic potential only
Electrostatics are evaluated as follows:	
0†, 1‡	Ignore Electrostatic interactions
2, 3	Ewald summation
4, 5	distant dependent dielectric constant.
6, 7	standard truncated Coulombic potential.
8, 9	truncated and shifted Coulombic potential.

† KEYFCE = 0 means no non-bonded terms are evaluated.
‡ KEYFCE = 1 means only short-range potentials are evaluated.

3.1.1.4 Comments on other variables

NSTBGR (record 4): If a multiple time-step is used, then statistics for radial distribution functions are collected only every NSTBGR updates of the secondary neighbourhood list. In effect this means that statistics are collected at least once every NSTBGR × MULTT time-steps since the secondary neighbourhood list is updated every MULTT steps from the last update of the Verlet list.

LZEQL (record 5): LZEQL set to .TRUE. will cause temperature scaling to be used for

NSTEQL (record 2) time-steps. After which all statistics arrays are reset and ure scaling is switched off. If you wish to collect statistics and use temperature multaneously then set LZEQL to .FALSE.. In this case make sure NSTEQL is or larger than NSTRUN (both on record 2).

M (record 5): If LOPTIM = .TRUE. then at each time-step all velocities are o before the equations of motion are integrated. The result is a crude structure ion.

' (record 9): If MULTT=1 a single time-step algorithm is used. Otherwise / forces are evaluated once every MULTT steps or whenever the verlet list is

record 10): DELR is the width of the Verlet border. The Verlet list is updated two or more atoms have moved a distance of more then DELR/2 from their at the last update of the Verlet list.

3.1.2 CONFIG

The CONFIG file contains the dimensions of the unit cell, the key for periodic boundary conditions and the atomic labels, coordinates, velocities and forces. This file is read in the subroutine sysgen. The first few records of a typical CONFIG file are shown below:

```
IceI structure 6x6x6 unit cells with proton disorder
         2          3
  26.988000000000000   0.000000000000000   0.000000000000000
 -13.494000000000000  23.372293600000000   0.000000000000000
   0.000000000000000   0.000000000000000  44.028000000000000
     OW          1
   -2.505228382        -1.484234330        -7.274585343
   0.5446573999        -1.872177437       -0.7702718106
    3515.939287         13070.74357         4432.030587
     HW          2
   -1.622622646        -1.972916834        -7.340573742
    1.507099154        -1.577400769         4.328786484
    7455.527553        -4806.880540        -1255.814536
     HW          3
   -3.258494716        -2.125627191        -7.491549620
    2.413871957        -4.336956694         2.951142896
   -7896.278327        -8318.045939        -2379.766752
     OW          4
   0.9720599243E-01    -2.503798635        -3.732081894
    1.787340483        -1.021777575        0.5473436377
    9226.455153         9445.662860         5365.202509

etc.
```

3.1.2.1 Format

The file is formatted: integers as "i10", reals as "f20.0". The header record is formatted as 80 alphanumeric characters.

3.1.2.2 Definitions of variables

record 1		
header	a80	title line
record 2		
levcfg	integer	CONFIG file key. See table 3.5 for permitted values.
imcon	integer	Periodic boundary key. See table 3.6 for permitted values
record 3	omitted if IMCON = 0	
cell(1)	real	x component of a cell vector.
cell(2)	real	y component of a cell vector.
cell(3)	real	z component of a cell vector.
record 4	omitted if IMCON = 0	
cell(4)	real	x component of b cell vector.
cell(5)	real	y component of b cell vector.

cell(6)	real	z component of *b* cell vector.
record 5	omitted if IMCON = 0	
cell(7)	real	x component of *c* cell vector.
cell(8)	real	y component of *c* cell vector.
cell(9)	real	z component of *c* cell vector.

Subsequent records consists of blocks of between 2 and 4 records dependent upon the value of LEVCFG. Each block refers to one atom. The atoms must be listed sequentially in order of increasing index. Within each block the data are as follows:

record i		
atmnam	a8	Atom name. Note that the above example includes the site index in the record. The site index is not read by DL_POLY. It is included only to aid browsing and editing the file.
record ii		
xxx	real	x coordinate
yyy	real	y coordinate
zzz	real	z coordinate
record iii	included only if LEVCFG = 1 or 2	
vxx	real	x component of velocity
vyy	real	y component of velocity
vzz	real	x component of velocity
record iv	included only if LEVCFG = 2	
fxx	real	x component of force
fyy	real	y component of force
fzz	real	z component of force

3.1.2.3 Guide to integer keys

Table 3.5: CONFIG file key (record 2)

LEVCFG	meaning
0	Coordinates included in file.
1	Coordinates and velocities included in file.
2	Coordinates, velocities and forces included in file.

Table 3.6: Periodic boundary key (record 2)

IMCON	meaning
0	no periodic boundaries.
1	cubic boundary conditions.
2	orthorhombic boundary conditions.
3	parallelepiped boundary conditions.
4	truncated octahedral boundary conditions.
5	rhombic dodecahedral boundary conditions.
6	x-y parallelogram boundary conditions with no periodicity in the z direction.

3.1.2.4 Further Comments

the CONFIG file has the same format as the output file REVCON. When restarting a previous run of DL_POLY (i.e. KEYRES=1), or starting a new simulation from the final configuration of a previous run (i.e. KEYRES=2), the CONFIG file must be replaced by the REVCON file, which is then renamed as the CONFIG file. The "**copy**" macro in the "execute" directory of DL_POLY does this for you.

3.1.3 FIELD

The FIELD file contains the force field information. It is read in the subroutine sysdef. Excerpts from a force field file are shown below:

```
Valinomycin complex with K+
units internal
   2
Valinomycin
   1  168    6  162  312  376
   N        14.0100     -0.4630        1        1   N      N
   H         1.0080      0.2520        1        2   H      H
   CT       12.0100      0.0350        1        3   CT     CT
 . . .
   HC        1.0080      0.0380        1      168   HC     HC
   1   16   17  1.5345e+05       1.335          1
 . . .
   1  160  161  1.5345e+05       1.335          6
   2    1  0.99996813                    1
   3    1  1.53217867                    2
 . . .
 168  165  1.00002235                  161
 161  163  1.58685275                  162
   1    2    1    3  31798.      118.40          1
   1    2    1  163  29288.      119.80          2
 . . .
   1  166  165  168  29288.      109.50        311
   1  167  165  168  29288.      109.50        312
   1    1    3    5    6  28.033      180.00        3            1
   1    4    3    5    6  28.033      180.00        3            2
 . . .
   1  146  160  148  149 0.18381E+07  180.00      2.0000        375
   1    1  161  163  164 0.18381E+07  180.00      2.0000        376
Potassium
   1    1    0    0    0    0
   K       40.000      1.000
  36
   C       C      2  50.208      3.2963          1
   C       CT     2  35.502      3.2518          2
 . . .
   O       OS     2  72.469      2.8954         35
   OS      OS     2  62.760      2.9400         36
```

3.1.3.1 Format

The file is formatted. Integers are formatted as "i5", reals as "f12.0" and characters as "a8", "a40" or "a80".

3.1.3.2 Definitions of variables

The file divides into three sections: general, molecular details, and non-bonded interactions. These sections follow on from each other in the file.

3.1.3.2.1 General

record 1

header	a80	field file header.

record 2 optional†

engunit	a40	Unit of energy used for input and output.

record 3

NTPMLS	integer	the number of different molecular types in the system.

†Note: This line takes the form

`UNITS units`

where units is one of "eV", "kcal", "kJ" or "INTERNAL", corresponding to energy units of electron-Volts, kcal mol^{-1}, kJ mol^{-1} or DL_POLY internal energy units (10 J mol^{-1}) respectively. units is not case sensitive and may appear anywhere after the keyword UNITS provided it does nor exceed column 40. If this line is omitted DL_POLY internal energy units are used for all input and output. Regardless of the value of units all internal calculations are conducted using DL_POLY internal units - the UNITS option only affects the input and output interfaces.

3.1.3.2.2 Molecular details

For each molecular type a block of data containing site, chemical bond, constraint bond, valence angle and dihedral angle information is required:

next record

molnam	a80	molecule name

next record

nummols	integer	the number of molecules of this type in the M.D. cell
numsit	integer	the number of atomic sites on the molecule.
numbonds	integer	the number of chemical (ie. extensible) bonds in the molecule.
numcon	integer	the number of constraint (ie. rigid) bonds in the molecule.
numang	integer	the number of valence angle interactions for the molecule.
numdih	integer	the number of dihedral interactions in the molecule.

Site details

NUMSIT records giving site names, masses and charges. Each record contains:

next record

sitnam	a8	site name

weight	real	site mass
chge	real	site charge
nrept	integer	repeat counter

The next NREPT - 1 entries in the CONFIG file are ascribed the atomic characteristics given in the current record. If NREPT is omitted it is assumed to be 1. The example given above includes additional information on the end of each record to aid in browsing the file.

Note: DL_POLY (Release I) is an atomistic code. Each site requires a non-zero mass assigned to it to enable the integration of the equations of motion to be performed.

Atomic indices used for chemical bond, constraint, angle and dihedral potentials refer to site indices for the molecule. Permitted values thus range from 1 to NUMSIT.

Chemical bond potentials

NUMBONDS records of chemical bond data. Each record contains:

next record

bond key	integer	see table 3.7
index 1	integer	first atomic site in bond.
index 2	integer	second atomic site in bond.
variable 1	real	see table 3.7
variable 2	real	see table 3.7
variable 3	real	see table 3.7

Constraint bonds

NUMCON records pertaining to constraint bonds. Each record contains:

next record

index 1	integer	first atomic index.
index 2	integer	second atomic index.
bndlng	real	constraint bond length

Valence Angle potentials

NUMANG records pertaining to valence angle interactions. Each record contains

next record

key angle	integer	potential key. See table 3.8
index 1	integer	first atomic index.
index 2	integer	second atomic index (central site).
index 3	integer	third atomic index.
variable 1	real	see table 3.8
variable 2	real	see table 3.8

Table 3.7: Chemical bond potentials

key	potential type	Variables (1-3)			functional form
1	Harmonic	k	r_0		$U = k/2(r - r_0)^2$
2	Morse	D_0	r_0	α	$U = D_0\left([1 - \exp(-\alpha(r - r_0))]^2 - 1\right)$
3	12-6	a	b		$U = a/r^{12} - b/r^6$

Table 3.8: Valence Angle potentials

key	potential type	Variables (1-2)		functional form†
1	Harmonic	k	θ_0	$U = k/2(\theta - \theta_0)^2$

†θ is the a-b-c angle.

Table 3.9: Dihedral Angle potentials

key	potential type	Variables (1-3)			functional form†
1	Cosine	A	δ	n	$U = A(1 + \cos(n\phi - \delta))$
2	Harmonic	k	ϕ_0		$U = k/2(\phi - \phi_0)^2$
3	Modified Cosine¶	A	δ	n	$U = A(1 + \cos(n\phi - \delta))$

†ϕ is the a-b-c-d dihedral angle.

¶This potential also reduces the 1-4 electrostatic interaction by 50 %.

Dihedral potentials

NUMDIH records pertaining to dihedral interactions. Each record contains:

next record

dihedral key	integer	potential key. See table 3.9
index 1	integer	first atomic index.
index 2	integer	second atomic index.
index 3	integer	third atomic index.
index 4	integer	fourth atomic index.
variable 1	real	see table 3.9
variable 2	real	see table 3.9
variable 3	real	see table 3.9

Note that dihedral key = 3 implies a cosine potential with the 1-4 electrostatic interaction reduced by 50% of its standard value. This has not been explicitly stated in the potential functional form but is properly accounted for in the code. This option is a feature inherent in the AMBER force field.

3.1.3.2.3 Non-bonded interactions

Non-bonded interactions are based on groups of atom types as opposed to specific atomic indices. All non-bonded potentials are assumed to be pair-wise additive.

next record

NTPVDW	integer	number of short-ranged potentials

Each of the next NTPVDW records contain:

next record

atmnam 1	a8	first atom type
atmnam 2	a8	second atom type
key	integer	potential key. See table 3.10.
variable 1	real	see table 3.10.
variable 2	real	see table 3.10.
variable 3	real	see table 3.10.
variable 4	real	see table 3.10.
variable 5	real	see table 3.10.

Table 3.10: Definition of potential functions and variables

key	potential type	Variables (1-5)					functional form
1	12-6	a	b				$U = a/r^{12} - b/r^6$
2	Lennard-Jones	ε	σ				$U = 4\varepsilon\,[(\sigma/r)\,r^{12} - (\sigma/r)\,r^6]$
3	m-n	ε	m	n	r_o		$U = \varepsilon/(m-n)\,[n\,(r_o/r)^m - m\,(r_o/r)^n]$
4	Buckingham exp 6	a	ρ	b			$U = a\exp(-r/\rho) - b/r^6$
5	Born Huggins Meyer	a	α	r_0	b	c	$U = a\exp(-\alpha(r - r_0)) - b/r^6 - c/r^8$
6	12-10	a	b				$U = a/r^{12} - b/r^{10}$

3.1.4 REVOLD

This file contains statistics arrays from a previous job. It is not required if the current job is not a continuation of a previous run (ie. if KEYRES in the CONTROL file is 0). The file is unformatted. DL_POLY produces the file REVIVE at the end of a job which contains the statistics files. REVIVE should be copied to REVOLD before a continuation run commences. This may be done by the **copy** command supplied in the "execute" directory of DL_POLY.

3.2 OUTPUT files

DL_POLY produces five output files: HISTORY, OUTPUT, REVCON, REVIVE, and STATIS. These respectively contain: a dump file of atomic coordinates, velocities and forces; a summary of the simulation; a restart configuration; statistics arrays; and a history of statistical output.

3.2.1 HISTORY

The HISTORY file is the dump file of atomic coordinates, velocities and forces. It's principal use is as input for off-line analysis. The file is written from the subroutine **traject**. The control variables for this file are LTRAJ, NSTRAJ, ISTRAJ and LEVTRJ all of which are read from the CONTROL file. The HISTORY file will be created only if LTRAJ is set to .TRUE. in the CONTROL file.

The HISTORY file is of the following form

record 1 (a80)

header	a80	configuration name

record 2 (2i10)

LEVTRJ	integer	trajectory key.
IMCON	integer	periodic boundary key.

For time-steps greater than NSTRAJ, the HISTORY file is appended once every ISTRAJ time-steps with the following information:

record i (a8,i10)

timestep nstep	integer	the current time-step

record ii (3g12.4) for IMCON > 0

CELL(1)	real	x component of a cell vector.
CELL(2)	real	y component of a cell vector.
CELL(3)	real	z component of a cell vector.

record iii (3g12.4) for IMCON > 0

CELL(4)	real	x component of b cell vector.
CELL(5)	real	y component of b cell vector.
CELL(6)	real	z component of b cell vector.

record iv (3g12.4) for IMCON > 0

CELL(7)	real	x component of c cell vector.
CELL(8)	real	y component of c cell vector.
CELL(9)	real	z component of c cell vector.

For each atom in the system the following is then included:

record a (a8)

atmnam	a8	atomic label

record b (3e12.4)

xxx	real	x coordinate.
yyy	real	y coordinate.
zzz	real	z coordinate.

record c (3e12.4) only for LEVTRJ = 1 or 2

vxx	real	x component of velocity.
vyy	real	y component of velocity.
vzz	real	z component of velocity.

record d (3e12.4) only for LEVTRJ = 2

fxx	real	x component of force.
fyy	real	y component of force.
fzz	real	z component of force.

Note that the HISTORY file can become VERY large. For serious simulation work it is recommended that the file be written to a scratch disk capable of accommodating a large data file. Alternatively the file may be written unformatted and using REAL*4 (ie. 32 bit) precision. However, writing an unformatted file has the disadvantage that the file may not be readily readable except by the machine by which it was created. This is particularly important if graphical processing of the data is required.

3.2.2 OUTPUT

The job output consists of 7 sections: Header; Simulation control specifications; Force field specification; Summary of the initial configuration; Simulation progress; Summary of statistical data; Sample of the final configuration; and Radial distribution functions. These sections are written by different subroutines at various stages of a job. Creation of the OUTPUT file *always* results from running DL_POLY. It is meant to be a human readable file, destined for hardcopy output.

3.2.2.1 Header

Gives the version of DL_POLY being used, the number of processors used and a title for the job as given in the header line of the input file CONTROL. This part of the file is written from the subroutine **simdef**

3.2.2.2 Simulation control specifications

Echoes back the input from the CONTROL file. Some variables may be reset if illegitimate values were specified in the CONTROL file. This part of the file is written from the subroutine **simdef**.

3.2.2.3 Force Field Specification

Echoes the FIELD file. A warning line will be printed if the system is not electrically neutral. This warning will appear immediately before the non-bonded short-range potential specifications. This part of the file is written from the subroutine **sysdef**.

3.2.2.4 Summary of the initial configuration

This part of the file is written from the subroutine **sysgen**. It states the periodic boundary key, the cell vectors and volume (if appropriate) and the initial configuration of (a maximum of) 20 atoms in the system. The configuration information given is

based on the value of LEVCFG in the CONFIG file. If LEVCFG is 0 or 1 positions and velocities of the 20 atoms are listed. If LEVCFG is 2 forces are also written out.

For periodic systems this is followed by the long range corrections to the energy and pressure.

3.2.2.5 Simulation progress

This part of the file is written from the subroutine **cycle**. The header line is printed at the top of each page as:

```
-------------------------------------------------------------------------------------------
   step   eng_tot   eng_kin   eng_cfg   eng_vdw   eng_cou   eng_bnd   eng_ang   eng_dih   volume
time(s)   eng_pv      temp   vir_cfg   vir_vdw   vir_cou   vir_bnd   vir_con   vir_dih    press
-------------------------------------------------------------------------------------------
```

The labels refer to :

line 1

step	MD step number.
eng_tot	total internal energy of the system.
eng_kin	kinetic energy of the system.
eng_cfg	configurational energy of the system.
eng_vdw	configurational energy due to short-range (van der Waals type) potentials.
eng_cou	configurational energy due to electrostatic potential.
eng_bnd	configurational energy due to chemical bond potentials.
eng_ang	configurational energy due to valence angle potentials.
eng_dih	configurational energy due to dihedral potentials.
volume	system volume.

line 2

time	elapsed cpu time since the beginning of the job.
eng_pv	enthalpy of system.
temp	temperature
vir_cfg	total configurational contribution to the virial.
vir_vdw	short range potential contribution to the virial.
vir_cou	electrostatic potential contribution to the virial.
vir_bnd	chemical bond contribution to the virial.
vir_con	constraint bond contribution to the virial.
vir_dih	dihedral potential contribution to the virial (zero).
press	pressure.

The interval for printing out this data is determined by the variable NSTBPO in the CONTROL file. At each time-step that printout is requested the instantaneous values of the above statistical variables are given in the appropriate columns. Immediately below these two lines of output the rolling averages of the same variables are also given. The maximum number of time-steps used to calculate the rolling averages is determined by the parameter MXSTAK in the **dl_params.inc** file. The working number of time-steps for rolling averages is controlled by the input variable NSTACK in file CONTROL (Note: NSTACK $\leq$ MXSTAK)

3.2.2.6 Summary of statistical data

This portion of the OUTPUT file is written from the subroutine **result**. The number of time-steps used in the collection of statistics is given. Then the averages over the production portion of the run are given for the variables described in the previous section. The root mean square variation in these variables follow on the next two lines.

3.2.2.7 Sample of final configuration

The positions, velocities and forces of the 20 atoms used for the sample of the initial configuration (section 3.2.2.4) are given. This is written from the subroutine **result**.

3.2.2.8 Radial distribution functions

If both calculation and printing of radial distribution functions have been requested (by selecting LGOFR and LPGR to be .TRUE. in the CONTROL file) radial distribution functions are printed out as the last part of the output. This is wriitten from the subroutine **rdf1**. First the number of time-steps used for the collection of the histograms is stated. Then each function is given in turn. For each function a header line states the atom types ('a' and 'b') used for the function. Then r, g(r) and n(r) are given in tabular form. Output is given from 2 entries before the first non-zero entry in the g(r) histogram. "n(r)" is the average number of atoms of type 'b' within a sphere of radius r around an atom of type 'a'.

3.2.3 REVCON

This file is formatted and written by the subroutine **revive**. REVCON is the restart configuration file. The file is written every NDUMP time steps in case of a system crash during execution and at the termination of the job. A successful run of DL_POLY will always produce a REVCON file, but a failed job may not produce the file if insufficient time steps have been computed. NDUMP is a parameter defined in the "dl_params.inc" file found in the "source" directory of DL_POLY. Changing NDUMP necessitates recompiling DL_POLY. REVCON is identical in format to the input file CONFIG. REVCON should be copied to CONFIG to continue a simulation from one job to the next. This is done for you by the **copy** command supplied in the "execute" directory of DL_POLY.

3.2.4 REVIVE

This file is unformatted and written by the subroutine **revive**. It contains statistics arrays. It is updated whenever the file REVCON is updated (see previous section). REVIVE should be copied to a file named REVOLD to continue a simulation from one job to the next. This is done for you by the **copy** command supplied in the "execute" directory of DL_POLY.

3.2.5 STATIS

The file is formatted, with integers as "i8" and reals as "e14.6". It is written from the subroutine **static**. It consists of a header record followed by many data records of statistical data.

record 1		
cfgname	character	configuration name
record 2		
string	character	Energy units.

Data records

Subsequent lines contain the instantaneous values of statistical variables dumped from the array STPVAL. All entries of STPVAL are written in the format "(1p,5e14.6)". The length of this array is determined by the parameter MXNSTK in the **dl_params.inc** file (in the following we assume MXNSTK is 25 and that there are a maximum of 6 different atomic types in the system). The STATIS file is appended every INTSTA (defined in the CONTROL file) time-steps with this information. The contents of this appended information is:

record i		
nstep	integer	current MD time-step.
record ii	STPVAL(1) - STPVAL(5)	
eng	real	total system energy.
engke	real	kinetic energy.
engcfg	real	configurational energy.
engsrp	real	short range potential energy.
engcpe	real	electrostatic energy.
record iii	STPVAL(6) - STPVAL(10)	
engbnd	real	chemical bond energy.
engang	real	valence angle energy.
engdih	real	dihedral interaction energy.
volume	real	volume.
blank	real	unassigned.
record iv	STPVAL(11) - STPVAL(15)	
eth	real	enthalpy.
temp	real	temperature.
vir	real	total virial.
virsrp	real	short-range virial.
vircpe	real	electrostatic virial.
record v	STPVAL(16) -STPVAL(20)	
virbnd	real	bond virial.
vircon	real	constraint virial.
blank	real	unassigned.
press	real	pressure.
amsd(1)	real	mean squared displacement of first atom types.
record vi	STPVAL(21) -STPVAL(25)	
amsd(2)	real	mean squared displacement of second atom types.
amsd(3)	real	mean squared displacement of third atom types.
amsd(4)	real	mean squared displacement of fourth atom types.
amsd(5)	real	mean squared displacement of fifth atom types.
amsd(6)	real	mean squared displacement of sixth atom types.

Chapter 4

Running DL_POLY

4.1 Guide to preparing input files

The CONFIG file and the FIELD file can be quite large and unwieldly particularly if a polymer or biological molecule is involved in the simulation. This section outlines the paths to follow when trying to construct files for such systems. But first we describe the generalised force field used by DL_POLY.

4.1.1 Description of the force field

The total potential energy of the system is of the form:

$$
\begin{aligned}
V(\underline{r}_1, \underline{r}_2, \ldots, \underline{r}_N) = & \sum_{i_{bond}=1}^{N_{bond}} U_{bond}(i_{bond}, \underline{r}_a, \underline{r}_b) + \sum_{i_{angle}=1}^{N_{angle}} U_{angle}(i_{angle}, \underline{r}_a, \underline{r}_b, \underline{r}_c) \\
& + \sum_{i_{dihed}=1}^{N_{dihed}} U_{dihed}(i_{dihed}, \underline{r}_a, \underline{r}_b, \underline{r}_c, \underline{r}_d) \\
& + \sum_{i=1}^{N-1} \sum_{j>i}^{N} U_{nonbond}(i, j, |\underline{r}_i - \underline{r}_j|) \qquad (4.1)
\end{aligned}
$$

where U_{bond}, U_{angle}, U_{dihed} and $U_{nonbond}$ represent empirical interaction functions for chemical bonds, valence angles, dihedral angles and non-bonded (Van der Waals and Coulombic) forces respectively. The numbers N_{bond}, N_{angle} and N_{dihed} refer to the total numbers of these respective interactions present in the simulated system. The presence of the indices i_{bond}, i_{angle} and i_{dihed} within the defined functions indicates that each individual function may have its own parameters and functional form. The position vectors $\underline{r}_a, \underline{r}_b, \underline{r}_c$ and $\underline{r}_d$ refer to the positions of those atoms specifically involved in a given interaction term. Similarly, the indices i and j appearing in the nonbonded terms indicates that these may also have a functional form dependent on the types of interacting atoms.

4.1.2 "Simple" systems

The utility **genlat** can be used to construct the CONFIG file for relatively simple lattice structures. Input is interactive. The FIELD file for such systems are normally small

and can be constructed by hand. The utility **genlat.to** constructs the CONFIG file for truncated-octahedral boundary conditions.

4.1.3 GROMOS

If you wish to take a *protein* structure from a SEQNET file (eg. from the Brookhaven database) and use this in DL_POLY with the GROMOS force field the following route is suggested:

Use the utility **proseq2** to generate the file PRODATA. This will then function as input for the utility **groforce** which will generate the FIELD and CONFIG files ready for DL_POLY.

4.1.4 AMBER

If you have the "edit.out" file produced by AMBER for your molecule use this as the CONNECT_DAT input file for the utility **ambforce**. **Ambforce** will produce the DL_POLY FIELD and CONFIG files for your molecule.

If you do not have the "edit.out" file things are a little more tricky, particularly in coming up with appropriate partial charges for atomic sites. However there are a series of utilities that will at least produce the CONNECT_DAT file for use with **ambforce**. We now outline these utilities and the order in which they should be used.

If you have a structure from the Cambridge Structural database (CSDB) then use the utility **fraccon** to take fractional coordinate data and produce a CONNECT_DAT and "ambforce.dat" file for use with **ambforce**. Note that you will need to modify **fraccon** to get the AMBER names correct for sites in your molecule. The version of **fraccon** supplied with DL_POLY version I is specific to the valinomycin molecule. Estimates of partial charges can be obtained by using the **chgefnd** utility prior to using **ambforce**. You will need to modify **chgefnd** to suit your needs(!).

If you require an all atom force field and the database file does not contain hydrogen positions then use the utility **fracfill** in place of **fraccon**. **Fraccon** produces an output file HFILL which should then be used as input for the utility **hfill**. The **hfill** utility fills out the structure with the missing hydrogens. The output file (CONNECT_DAT) should then used as input for the utility **chgefnd**. The utility **chgefnd** seeks to assign AMBER partial charges for atomic sites in the CONNECT_DAT file, but note, again, the version of chgfnd supplied with DL_POLY version I is not a general routine - it will need to be modified to suit your requirements. The resultant file, CONNECT_DAT.q should be moved to CONNECT_DAT prior to the **ambforce** utility being run.

Note: with minor modifications the utilities **fracfill** and **fraccon** can be used on structures from databases other than the Cambridge structural database.

4.1.5 Adding solvent to a structure

The utility **wateradd** adds water from an equilibrated configuration of 256 SPC water molecules at 300 K to fill out the MD cell. The utility **solvadd** fills out the MD box with single-site solvent molecules from a f.c.c lattice. The FIELD files will then need to be edited to account for the solvent molecules added to the file.

Hint: to save yourself some work in entering the non-bonded interactions variables involving solvent sites to the FIELD file put two bogus atoms of each solvent type at

the end of the CONNECT_DAT file (for AMBER force-fields) or the PRODATA file (for GROMOS force-fields) the utilities **ambforce** and **groforce** will then evaluated all the non-bonded variables required by DL_POLY. Remember to delete the bogus entries from the CONFIG file before running DL_POLY.

4.2 DL_POLY Error Messages

4.2.1 The DL_POLY Internal Error Facility

DL_POLY contains a number of in-built error checks designed to prevent abuse of the code. These error checks are scattered throughout the package, but in all cases, when an error is detected the subroutine **error** is called, resulting in an appropriate message and termination of the program execution.

Users intending to insert new error checks should ensure that all error checks are performed *concurrently* on *all* nodes, and that in circumstances where a different result may obtain on different nodes, a call to the global status routine **gstate** is made to set the appropriate global error flag on all nodes. Only after this is done, a call to subroutine **error** may be made. An example of such a procedure might be:

```
logical safe
safe=(test_condition)
call gstate(safe)
if(.not.safe) call error(node_id,message_number)
```

In this example it is assumed that the logical operation *test_condition* will result in the answer *.true.* if it is safe for the program to proceed, and *.false.* otherwise. The call to **error** requires the user to state the identity of the calling node (**node_id**), so that only the nominated node in error (i.e. node 0) will print the error message. The variable **message_number** is an integer used to identify the appropriate message to be printed. By convention this number is identical to the FORTRAN format number.

In all cases, if **error** is called, the program execution terminates. A possible modification users may consider is to dump additional data before the call to **error** is made.

Readers should see the description of the **error** subroutine for further details.

4.2.2 Error Messages and User Action

In this section we document the current error messages encoded in DL_POLY (Release I) and the recommended user action. Most of these errors refer to array bound checks and the user is advised to consider what other arrays may need to be altered when making the appropriate change. (If you are not familiar with DL_POLY, the error messages will advise you as each array bound is violated, but this will not happen in one pass.) Readers are warned that reckless increases in the array dimensions are likely to result in the code being too large for any given memory. It should also be remembered that any change in a parameter specified in the include file **dl_params.inc** requires the *whole program* to be recompiled.

4.2.2.1 Message 5: error - unknown energy unit requested

The DL_POLY FIELD file permits a choice of units for input of energy parameters. These may be: electron volts (EV); kilocalories (KCAL); kilojoules (KJ); or the DL_POLY internal units (INTERNAL), with the default being internal units. Failure to specify any of these correctly, or reference to other energy units, will result in this error message. See documentation of the FIELD file.

Action:
Correct energy keyword in FIELD file and resubmit.

4.2.2.2 Message 10: error - too many molecule types specified

DL_POLY has a set limit on the number of kinds of molecules it will handle in any simulation (this is not the same as the number of molecules). If this permitted maximum is exceeded, the program terminates.

Action:
Locate the parameter **mxtmls** in the DL_POLY **dl_params.inc** file and increase the default number. Then recompile the entire program.

4.2.2.3 Message 15: error - duplicate pair potential specified

In processing the FIELD file, DL_POLY keeps a record of the specified short range pair potentials as they are read in. If it detects that a given pair potential has been specified before, no attempt at a resolution of the ambiguity is made and this error message results. See specification of FIELD file.

Action:
Locate the duplication in the FIELD file and rectify.

4.2.2.4 Message 20: error - too many molecule sites specified

DL_POLY has a fixed limit on the number of unique molecular sites in any given simulation. If this limit is exceeded, the program terminates.

Action:
Locate the parameter **mxsite** in the DL_POLY parameter file **dl_params.inc** and increase. Recompile the entire program.

4.2.2.5 Message 25: error - wrong atom type found in CONFIG file

On reading the input file CONFIG, DL_POLY performs a check to ensure that the atoms specified in the configuration provided are compatible with the corresponding FIELD file. This message results if they are not.

Action:
The possibility exists that one or both of the CONFIG or FIELD files has incorrectly specified the atoms in the system. The user must loacte the ambiguity, using the data printed in the OUTPUT file as a guide, and make the appropriate alteration.

4.2.2.6 Message 30: error - too many chemical bonds specified

DL_POLY sets a limit on the number of chemical bond potentials that can be specified in the FIELD file. Termination results if this number is exceeded. See FIELD file documentation. Do not confuse this error with that described by message_id 31 (below).

Action:
Locate the parameter **mxtbnd** in the **dl_params.inc** file and increase accordingly. Recompile the entire program.

4.2.2.7 Message 31: error - too many chemical bonds in system

DL_POLY sets a limit on the number of chemical bond potentials in the simulated system as a whole. (This number is a combination of the number of molecules and the number of bonds per molecule, divided by the number of processing nodes.) Termination results if this number is exceeded. Do not confuse this error with that described by message_id 30 (above).

Action:
Locate the parameter **mxbond** in the **dl_params.inc** file and increase accordingly. Recompile the entire program.

4.2.2.8 Message 40: error - too many bond constraints specified

DL_POLY sets a limit on the number of bond constraints that can be specified in the FIELD file. Termination results if this number is exceeded. See FIELD file documentation. Do not confuse this error with that described by message_id 41 (below).

Action:
Locate the parameter **mxtcon** in the **dl_params.inc** file and increase accordingly. Recompile the entire program.

4.2.2.9 Message 41: error - too many bond constraints in system

DL_POLY sets a limit on the number of bond constraints in the simulated system as a whole. (This number is a combination of the number of molecules and the number of constraints per molecule, divided by the number of processing nodes.) Termination results if this number is exceeded. Do not confuse this error with that described by message_id 40 (above).

Action:
Locate the parameter **mxcons** in the **dl_params.inc** file and increase accordingly. Recompile the entire program.

4.2.2.10 Message 45: error - too many atoms in CONFIG file

DL_POLY limits the number of atoms in the system to be simulated and checks for the violation of this condition when it reads the CONFIG file. Termination will result if the condition is violated.

Action:
Locate the parameter **mxatms** in the **dl_params.inc** file and increase accordingly. Recompile the entire program. Consider the possibility that the wrong CONFIG file is being used (e.g similar system, but larger size.)

4.2.2.11 Message 50: error - too many bond angles specified

DL_POLY limits the number of valence angle potentials that can be specified in the FIELD file and checks for the violation of this. Termination will result if the condition is violated.

Action:
Locate the parameter **mxtang** in the **dl_params.inc** file and increase accordingly. Recompile the entire program.

4.2.2.12 Message 51: error - too many bond angles in system

DL_POLY limits the number of valence angle potentials in the system to be simulated (actually, the number to be processed by each node) and checks for the violation of this. Termination will result if the condition is violated.

Action:
Locate the parameter **mxangl** in the **dl_params.inc** file and increase accordingly. Recompile the entire program. Consider the possibility that the wrong CONFIG file is being used (e.g similar system, but larger size.)

4.2.2.13 Message 55: error - end of CONFIG file encountered

This error arises when DL_POLY attempts to read more data from the CONFIG file than is actually present. The probable cause is an incorrect or absent CONFIG file, but it may be due to the FIELD file being incompatible in some way with the CONFIG file.

Action:
Check contents of CONFIG file. If you are convinced it is correct, check the FIELD file for inconsistencies.

4.2.2.14 Message 60: error - too many dihedral angles specified

DL_POLY will accept only a limited number of dihedral angles in the FIELD file and will terminate if too many are present.

Action:
Locate the parameter **mxtdih** in the **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.15 Message 61: error - too many dihedral angles in system

The number of dihedral angles in the whole simulated system is limited by DL_POLY. Termination results if too many are encountered.

Action:
Locate the parameter **mxdihd** in the file **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.16 Message 65: error - too many excluded pairs specified

This error can arise when DL_POLY is identifying the atom pairs that cannot have a pair potential between them, by virtue of being chemically bonded for example (see subroutine **exclude**). Some of the working arrays used in this operation may be exceeded, resulting in termination of the program

Action:
Locate the parameter **mxexcl** in the file **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.17 Message 70: error - constraint bond quench failure

When a simulation with bond constraints is started, DL_POLY attempts to extract the kinetic energy of the constrained atom-atom bonds arising from the assignment of initial random velocities. If this procedure fails, the program will terminate. The likely cause is a badly generated initial configuration.
Action:
Some help may be gained from increasing the cycle limit, by increasing the parameter **mxshak** in the **dl_params.inc** file and recompiling the entire code. You may also consider reducing the tolerance of the SHAKE iteration, the vairaible **tolnce** in the CONTROL file. However it is probably better to take a good look at the starting conditions!

4.2.2.18 Message 75: error - too many atoms in specified system

DL_POLY places a limit on the number of atoms that can be simulated. Termination results if too many are specified.

Action:
Locate the parameter **mxatms** in the file **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.19 Message 80: error - too many pair potentials specified

DL_POLY places a limit on the number of pair potentials that can be specified in the FIELD file. Exceeding this number results in termination of the program execution.

Action:
Locate the parameters **mxsvdw** and **mxvdw** in the **dl_params.inc** file and increase accordingly. Recompile the entire program.

4.2.2.20 Message 81: error - unidentified atom in pair potential list

DL_POLY checks all the pair potentials specified in the FIELD file and terminates the program if it can't identify any one of them from the atom types specified.

Action:
Correct the erroneous entry in the FIELD file and resubmit.

4.2.2.21 Message 82: error - calculated pair potential index too large

In checking the pair potentials specified in the FIELD file DL_POLY calculates a unique integer index that henceforth identifies the potential within the program. If this index becomes too large, termination of the program results.

Action:
Locate the parameters **mxsvdw** and **mxvdw** in the file **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.22 Message 85: error - required velocities not in CONFIG file

If the user attempts to start up a DL_POLY simulation with the option parameter KEYRES greater than zero (see description of CONTROL file,) the program will expect the CONFIG file to contain atomic velocities as well as positions. Termination results if these are not present.

Action:
Either replace the CONFIG file with one containing the velocities, or if not available, change the KEYRES option to zero in the CONTROL file. Do **not** attempt to fool DL_POLY by altering the variable LEVCFG in the CONFIG file.

4.2.2.23 Message 90: error - system total electric charge nonzero

In DL_POLY a check on the total system charge will result in an error if the net charge of the system is nonzero. (Note: In DL_POLY version I this message has been disabled. The program merely prints a warning stating that the system is not electrically neutral but it does not terminate the program - watch out for this.)

Action:
Check the specified atomic charges and their populations. Make sure they add up to zero. If the system is required to have a net zero charge, you can enable the call to this error message in subroutine **sysdef**.

4.2.2.24 Message 95: error - potential cutoff exceeds cell width

In order for the minimum image convention to work correctly within DL_POLY, it is necessary to ensure that the cutoff applied to the pair potentials does not exceed the perpendicular width of the simulation cell. (The perpendicular width is the shortest distance between opposing cell faces.) Termination results if this is detected.

Action:
Either shorten the cutoff radius, or if this is undesirable, use a larger simulation cell. There are no other cures.

4.2.2.25 Message 100: error - forces working arrays too small

There are a number of arrays in DL_POLY that function as workspace for the forces calculations. Their dimension is equal to the number of atoms in the simulation cell

divided by the number of nodes. If these arrays are likely to be exceeded, DL_POLY will terminate execution.

Action:
Locate the parameter **msatms** in the file **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.26 Message 102: error - parameter mxproc exceeded in shake arrays

The RD-SHAKE algorithm distributes data over all nodes of a parallel computer. Certain arrays in RD-SHAKE have a minimum dimension equal to the maximum nuumber of nodes DL_POLY is likely to encounter. If the actual number of nodes exceeds this, the program terminates.

Action:
Locate the parameter **mxproc** in the file **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.27 Message 103: error - parameter mxlshp exceeded in shake arrays

The RD-SHAKE algorithm requires that information about 'shared' atoms be passed between nodes. If there are too many atoms, the arrays holding the information will be exceeded and DL_POLY will terminate execution.

Action:
Locate the parameter **mxlshp** in the file **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.28 Message 105: error - shake algorithm failed to converge

The RD-SHAKE algorithm for bond constraints is iterative. If the maximum number of permitted iterations is exceeded, the program terminates. Possible causes include: a bad starting configuration; too large a time step used; incorrect force field specification; too high a temperature; inconsistent constraints involving shared atoms etc.

Action:
This is a difficult one. Corrective action depends on the cause. It is unlikely that simply increasing the iteration number will cure the problem, but you can try: increase the parameter **mxshak** in file **dl_params.inc** and recompile. But in truth, the trouble is much more likely to be cured by careful consideration of the physical system being simulated. You are on your own!

4.2.2.29 Message 110: error - neighbour list array too small

DL_POLY constructs a Verlet neighbour list in calculating the nonbonded (pair) force. The size of this array is limited. Any attempt to exceed the array, results in this error and program termination.

Action:
Locate the parameter **mxlist** in the file **dl_params.inc** and increase accordingly. Recompile the entire program.

4.2.2.30 Message 120: error - invalid determinant in matrix inversion

DL_POLY occasionally needs to calculate matrix inverses (usually the inverse of the matrix of cell vectors, which is of size 3×3). For safety's sake a check on the determinant is made, to prevent inadvertent use of a singular matrix.

Action:
Locate the incorrect matrix and fix it.

4.2.2.31 Message 130: error - incorrect octahedral boundary condition

When calculating minimum images DL_POLY checks that the periodic boundary of the simulation cell is compatible with the specifed minimum image algorithm. Program termination results if an inconsistency is found. In this case the error refers to the truncated octahedral minimum image, which is inconsistent with the simulation cell. The most probable cause is the incorrect definition of the simulation cell vectors present in the input file CONFIG, these must equal the vectors of the enscribing cubic cell.

Action:
Check the specified simulation cell vectors and correct accordingly.

4.2.2.32 Message 140: error - incorrect dodecahedral boundary condition

When calculating minimum images DL_POLY checks that the periodic boundary of the simulation cell is compatible with the specifed minimum image algorithm. Program termination results if an inconsistency is found. In this case the error refers to the rhombic dodecahedral minimum image, which is inconsistent with the simulation cell. The most probable cause is the incorrect definition of the simulation cell vectors present in the input file CONFIG, these must equal the vectors of the enscribing tetragonal simulation cell.

Action:
Check the specified simulation cell vectors and correct accordingly.

4.2.2.33 Message 150: error - unknown van der waals potential selected

When constructing the interpolation tables for the short ranged potentials DL_POLY checks that the potential function requested is one which is of a form known to the program. If the requested potential form is unknown, termination of the program results. The most probable cause of this is the incorrect choice of index number (the potential key) in the FIELD file.

Action:
Read the DL_POLY documentation and find the correct index for the potential desired. Insert the correct index in the FIELD file definition. If the correct form is not available,

look at the subroutine **forgen.f** (or its variant) and define the potential for yourself. It is easily done.

4.2.2.34 Message 160: error - unaccounted for atoms in exclude list

This error message means that DL_POLY has been unable to find all the atoms described in the exclusion list within the simulation cell. This should never occur, if it does it means a serious bookkeeping error has occured. The probable cause is corruption of the code somehow.

Action:
If you feel you can tackle it - good luck! Otherwise we recommend you get in touch with the program authors. Keep all relevant data files to help them find the problem.

4.2.2.35 Message 170: error - too many variables for statistic array

This error means the statistics arrays appearing in subroutine **static** are too small. This can happen if the number of unique atom types is too large.

Action:
Locate the parameter **mxnstk** in the file **dl_params.inc** and increase accordingly. **mxnstk** should be at least (19+*number of unique atom types*). Recompile the entire program.

4.2.2.36 Message 180: error - Ewald sum requested in non-periodic system change KEYFCE and/or IMCON

DL_POLY can use either the Ewald method or direct summation to calculate the electrostatic potentials and forces in periodic (or pseudo-periodic) systems. For non-periodic systems only direct summation is possible. If the Ewald summation is requested (KEYFCE=2 or 3) without periodic boundary conditions (IMCON=0), termination of the program results.

Action:
Select periodic boundaries (IMCON>0) or direct Coulombic summation (KEYFCE $\neq$ 2 or 3) in the input file CONTROL and re-run.

4.2.2.37 Message 190: error - buffer array too small in SPLICE

DL_POLY uses a workspace array named BUFFER in several routines. Its declared size is something of a compromise and may sometimes be too small (though in the supplied program, this should happen only very rarely). The only point of failure is in the **splice** routine, which is part of the RD-SHAKE algorithm.

Action:
Locate the parameter **mxbuff** in the file **dl_params.inc** and increase accordingly. Warning - do not set the parameter **mxbuff** to a value less than **6×mxatms** or **8×(mxcons+1)**. Recompile the entire program.

Chapter 5

DL_POLY Examples

5.1 DL_POLY Examples

The following example data sets (both input and output) are stored in the subdirectory *data*. These are proved so that you may check that your version of DL_POLY is working correctly. The output files are stored in compressed format. The test cases can be run by typing

```
select n
```

from the *execute* directory, where n is the number of the test case. The `select` command will copy the appropriate CONTROL, CONFIG, and FIELD files to the *execute* directory ready for execution. The output files obtained may be compared to the files supplied in the *data* directory.

In the following descriptions job execution times are given as a guide to how many resources the test cases will consume. The times refer to the total execution time for an 8 processor job on the INTEL iPSC/860 at Daresbury.

5.1.1 Test case 1

4-acetoamido-3-(1-acetyl-2-(2,6-dichlorobenzylidene))hydrazine-1,2,4-triazole. This molecule is the first entry in the Cambridge structural database. This test case uses the NVE ensemble. There are no electrostatic interactions. Harmonic bond potentials are used. Simulation time: 10000 timesteps = 2 ps. Execution time: 227 seconds.

5.1.2 Test case 2

As for test case 1 except that bond constraints are used in place of harmonic bond potentials and a single timestep algorithm is used with a larger timestep. Simulation time: 10000 timesteps = 5 ps. Execution time: 383 seconds.

5.1.3 Test case 3

NaCl (216 ions). This test case uses the NVE ensemble, a single timestep algorithm, the Ewald sum, and cubic periodic boundary conditions. Radial distribution functions are calculated and printed at the end of the job. Simulation time: 5000 timesteps = 5 ps. Execution time: 2578 seconds.

5.1.4 Test case 4

NaCl (256 ions). This test case uses the NVE ensemble, a single timestep algorithm, the Ewald sum, and truncated octahedral periodic boundary conditions. Radial distribution functions are calculated and printed at the end of the job. Simulation time: 5000 timesteps = 5 ps. Execution time: 1773 seconds.

5.1.5 Test case 5

As for test case 3 but now in the NVT ensemble. Temperature is controlled by the Hoover thermostat. Simulation time: 5000 timesteps = 5 ps. Execution time 2578 seconds.

5.1.6 Test case 6

As for test case 3 but now in the NVT ensemble. Temperature is controlled by Gaussian constraints. Simulation time: 5000 timesteps = 5 ps. Execution time 2554 seconds.

5.1.7 Test case 7

Valinomycin in 146 SPC waters. All chemical bonds involving hydrogen atoms are subject to bond constraints. The remaining bonds on the valinomycin molecule are treated with harmonic potentials. Electrostatic interactions are ignored. The microcanonical ensemble (NVE) is used with a multiple time step algorithm. Cubic periodic boundary conditions are in use. Simulation time: 3000 timesteps = 0.6 ps. Execution time: 955 seconds.

5.1.8 Test case 8

As for test case 7 with electrostatic interactions turned on and a larger primary cut off in use. The electrostatics are handled by a truncated and shifted Coulombic potential. Note, in comparison to test case 7, how large the r.m.s. fluctuation in the total energy has become. This is a consequence of the electrostatic interactions. In particular, it reflects the jump in the force when a neigbour moves across the cutoff boundary. Simulation time: 3000 timesteps = 0.6 ps. Execution time: 1750 seconds.

5.1.9 Test case 9

As for test case 8 but with a single time step algorithm and bond constraints applied to all bonds in the system. Simulation time: 300 timesteps = 0.6 ps. Execution time: 282 seconds.

Chapter 6

DL_POLY Utilities

Scope of Chapter

This chapter describes the utility programs and subroutines in DL_POLY, under the sub-directory 'utility'.

Three kinds of utility are distinguished: Initialisation, Analysis and Miscellaneous.

6.1 Initialisation Utilities

6.1.1 AMBFORCE

6.1.1.1 Header records

```
      program ambforce

c*********************************************************
c
c     program to extract AMBER force field for a macromolecule
c     and interface it to DL_POLY.
c     input file: CONNECT_DAT (AMBER 3.0 edit.out style)
c               : ambforce.dat
c     output    : CONFIG       (coordinates etc)
c               : FIELD.AMBER  (force-field)
c
c     copyright -  daresbury laboratory 1993
c
c     author    -  t forester    April 1993
c
c*********************************************************
```

6.1.1.2 Function

Constructs the DL_POLY CONFIG file and FIELD file using the AMBER force field. The actual files created are "CONFIG" and "FIELD.AMBER". The program requires input files CONNECT_DAT and "ambforce.dat".

6.1.1.3 Dependencies

- none

6.1.1.4 Parameters

integers:

ntyp	80	number of AMBER atom types
mga	464	number of AMBER angle potentials
mgb	239	number of AMBER bond potentials
mgd	285	number of AMBER torsion potentials
mgi	128	number of AMBER improper torsion potentials
mhb	60	number of AMBER hydrogen bond potentials
mlj	80	number of AMBER Lennard-Jones types
mxang	10000	maximun number of valence angles in system
mxatm	10000	number of atoms in FIELD specifications
mxbnd	10000	maximum number of bonds in system

6.1.1.5 Input

Data Files:

filename: CONNECT_DAT

This file should be in the same format as the "edit.out" file produced by AMBER ver 3.0. Excerpts from the example file "CONNECT_DAT.example" supplied in the "utility" directory are given below:

```
Valinomycin from n-octane - ref = 22135

     BOND ARRAY

    1  -99   O     O          4.0674   2.2246   8.7934  -0.400
    2  -99   OS    OS         5.0641   4.1595   9.2145  -0.200
    3  -99   O     O          7.6564   5.1321   7.0402  -0.504
. . .
   22   21   H     H          4.8173   6.0979   7.9033   0.252
   23  -99   N     N          9.0250   5.7322   4.4547  -0.463
   24   23   H     H          8.6759   5.7745   5.3908   0.252
   25  -99   N     N          8.2584   5.4632   0.2435  -0.463
. . .
   39   33   CT    CT         1.2013   3.2283  12.0105  -0.091
   40   39   H     HC         0.5830   3.8465  11.5253   0.031
   41   39   H     HC         1.7359   3.7477  12.6772   0.031
   42   39   H     HC         0.6674   2.5282  12.4848   0.031
. . .
  168   18   C     C          2.4980   6.0633  10.1126   0.616
  168   19   C     C          2.4980   6.0633  10.1126   0.616
  168  162   C     C          2.4980   6.0633  10.1126   0.616
```

the file consists of blocks of the following form:

header lines
BOND ARRAY
blank line
bonding data - one atom per line

The " BOND ARRAY" line signals that bonding data follows. The word BOND must be in uppercase and begin at column 6 in the file.

next record:

ia	i5	atomic index
ib	i5	neighbour index
dummy	3x	
at	a6	atom name
an	a6	AMBER name for atom

dummy	4x	
x	f10.4	x coordinate
y	f10.4	y coordinate
z	f10.4	z coordinate
qq	f8.4	partial charge

6.1.1.6 Comments

The neighbour index need only be given if it is less than the atomic index. If there are no such neighbours then enter "-99" for the neighbour index. If the atom is bonded to more than one neighbour with indices less than the atomic index then all these should be given, (one per line). For example atom 168 is bonded to atoms 18, 19 and 162, this is entered as:

```
168   18   C     C          2.4980   6.0633  10.1126   0.616
168   19   C     C          2.4980   6.0633  10.1126   0.616
168  162   C     C          2.4980   6.0633  10.1126   0.616
```

The program checks that the number of bonds to each atom is consistent with the type of atom indicated. If more bonds than expected are found an error message is written to the screen. If less bonds than expected are found the program enforces a bond between the atom and it's nearest non-bonded neighbour. This means that it is not strictly necessary to indicate all (or indeed any) of the bonded neighbours for a site. However, the locator algorithm is simplistic and the more information you include the better. It should also be said that AMBER only includes one neighbour bond for each site in the "edit.out" file. **Ambforce** was successful in locating all 'missing' interactions in these files for all trial cases including DNA fragments.

Ambforce generates the list of angle and dihedral interactions from the completed bonding data.

6.1.1.7 ambforce.dat

ambforce.dat is a formatted file: integers and reals are in free format. Co-ordinates and partial charges given in the ambforce.dat file are used preferentially over those supplied in the CONNECT_DAT file. If coordinates are not supplied in the ambforce.dat file then data is taken from CONNECT_DAT. In addition the ambforce.dat file contains instructions on whether or not to use constraint bonds, chemical bonds or a combination of both in the force field, and on the energy units to be used. The example file "ambforce.dat.example" supplied in the "utility" directory of DL_POLY is shown below:

```
Valinomycin from n-octane  - ref 22135
CONSTRAIN hydrogen
UNITS kJ
IP            169
0.00    0.00    0.00  1.00
```

The ambforce.dat file is of the form:

record 1		
header	a80	title line
record 2	optional	
constraint	a80	constraint information
record 3	optional	
units	a80	units of energy

record 2 is of the form

`CONSTRAIN string`

where `string` is one of "`all`", "`hydrogen`" or "`none`". This will result in : constraints apply to all bonds ; constraints apply only to bonds involving hydrogen atoms, other bonds are extensible; all bonds are extensible. `string` is not case sensitive and may appear anywhere after the keyword CONSTRAIN provided it does not exceed column 80. The default value of `string` is "none" and this will be used if record 2 is omitted or if the record is incorrectly entered.

Record 3 is also optional. It determines the units of energy used to generate the force-field: It takes the form:

`UNITS units`

where `units` is one of "eV","kcal", "kJ", or "INTERNAL" referring, respectively to units of: elecxtron volts, kcal mol^{-1}, kJ mol^{-1} and the DL_POLY internal energy units (10 J mol−1). If this line is omitted the default value will that for the AMBER force field *viz* kcal mol^{-1}.

The remainder of the file consists blocks of 2 lines each that indicate atomic coordinates and partial charges that are either additional to, or a modification of, those found in the CONNECT_DAT file.

record i		
atmnam	a8	atom name
index	i10	index
record ii		
x	real	x coordinate
y	real	x coordinate
z	real	z coordinate
qq	real	partial charge

6.1.1.8 Comments

6.1.2 CHGEFND

6.1.2.1 Header records

```
      program chgefnd

c***********************************************************************
c
c     DL_POLY routine to assign charges on Valinomycin. Data from
c     CONNECT_DAT file and print out in same format as CONNECT_DAT.q
c
c     copyright darsebury laboratory 1993
c     author -  t. forester     june 1993
c
c***********************************************************************
```

6.1.2.2 Function

Takes the CONNECT_DAT file (e.g. as created by the **fracfill** utility) and assigns partial charges to each atom. The input file is read from the standard input device. The output is placed in the file "CONNECT_DAT.q".

6.1.2.3 Dependencies

- none

6.1.2.4 Parameters

integers:

MXN	10000	maximum number of atoms in system.

6.1.2.5 Input

Interactive:

This file should be in the form of the CONNECT_DAT file required by **ambforce**. See the description of the utility **ambforce** for details on this file.

6.1.2.6 Comments

This routine is just a fairly simple assignment of charges and requires that one knows what the AMBER charges should be. It contains no real "science" but rather a series of questions that elucidates the environment of each atom in the molecule.

6.1.3 FRACCON

6.1.3.1 Header records

```
      program fraccon

c*********************************************************************
c
c     DL_POLY utility to convert fractional coordinates into real space
c     coordinates in form suitable for CONNECT_DAT (see ambforce.f). The
c     ambforce.dat file is also created.
c
c     input a,b,c (cell lengths)
c     input alp,bet,gam  (cell angles)
c     input xi,yi,zi - fractional coordinates
c
c     copyright darsebury laboratory 1993
c     author -  t. forester     june 1993
c
c*********************************************************************
```

6.1.3.2 Function

The program converts fractional coordinates to real space coordinates. The input data is assumed to be compatible with files from the Cambridge structural database. The output file, "CONNECT_DAT", is designed as input for the utility ambforce.

6.1.3.3 Dependencies

- none

6.1.3.4 Parameters

integers:

MXN	10000	maximum number of atoms in the structure.

6.1.3.5 Input

Interactive:

record 1:

TITLE	a40	title line
A0	real	a cell vector length (f8.0)
B0	real	b cell vector length (f8.0)
C0	real	c cell vector length (f8.0)

record 2:

dummy	22x	dummy
ALP	real	crystal angle α.
BET	real	crystal angle β.
GAM	real	crystal angle γ.

record 3:

NATMS	integer	number of atoms in unit cell.

The next NATMS records are of the form

next record:

dummy	5x	
NAME	a5	atom name
XB	real	x fractional coordinate (f10.5).
YB	real	y fractional coordinate (f10.5).
ZB	real	z fractional coordinate (f10.5).
dummy	1x	
ICN(1)	integer	index of first atom bonded to this site (i4).
ICN(2)	integer	index of second atom bonded to this site (i4.)
ICN(3)	integer	index of third atom bonded to this site (i4).
ICN(4)	integer	index of fourth atom bonded to this site (i4).

6.1.3.6 Comments

The assignment of atom types that occurs in the code (lines 123 - 164) is unique for the AMBER force-field applied to the Valinomycin molecule. You will need to edit this portion to suit your own needs.

6.1.4 FRACFILL

6.1.4.1 Header records

```
      program fracfill

c*********************************************************************
c
c     DL_POLY routine to convert fractional coordinates into real space
c     coordinates in form suitable for the hfill.f utility
c
c     input is assumed compatible with Cambridge Structural Database.
c     input a,b,c (cell lengths)
c     input alp,bet,gam  (cell angles)
c     input xi,yi,zi - fractional coordinates
c
c     copyright darsebury laboratory 1993
c     author -  t. forester     june 1993
c
c*********************************************************************
```

6.1.4.2 Function

The program first converts fractional coordinates to real space coordinates. It then assigns the number of neighbouring hydrogens missing from each atom in the crystal structure. The input file (read in by the standard device) is assumed to be in a form compatible with files from the Cambride Structural database. The output file, "HFILL", is designed as input for the utility **hfill**.

6.1.4.3 Dependencies

- none

6.1.4.4 Parameters

integers:

MXN	10000	maximum number of atoms in the structure.

6.1.4.5 Input

Interactive:

record 1:

TITLE	a40	title line
A0	real	a cell vector length (f8.0)
B0	real	b cell vector length (f8.0)
C0	real	c cell vector length (f8.0)

record 2:

dummy	22x	dummy

ALP	real	crystal angle α.
BET	real	crystal angle β.
GAM	real	crystal angle γ.

record 3:

NATMS	integer	number of atoms in unit cell.

The next NATMS records are of the form

next record:

dummy	5x	
NAME	a5	atom name
XB	real	x fractional coordinate (f10.5).
YB	real	y fractional coordinate (f10.5).
ZB	real	z fractional coordinate (f10.5).
dummy	1x	
ICN(1)	integer	index of first atom bonded to this site (i4).
ICN(2)	integer	index of second atom bonded to this site (i4.)
ICN(3)	integer	index of third atom bonded to this site (i4).
ICN(4)	integer	index of fourth atom bonded to this site (i4).

6.1.4.6 Comments

The assignment of atom types that occurs in the code (lines 118 - 164) is unique to the AMBER force-field applied to the Valinomycin molecule. You will need to edit this portion to suit your own needs.

6.1.5 GENLAT

6.1.5.1 Header records

```
      program genlat
c
c*********************************************************************
c
c     program to generate a perfect lattice with general unit cell
c     author - w.smith oct. 1992
c
c*********************************************************************
c
```

6.1.5.2 Function

genlat is used to generate the position coordinates of a simulation cell by replication of a unit cell. i.e. it generates a crystal lattice within the simulation cell. It is not confined to simple cubic lattices. The execution is interactive; the user must supply in turn:

1. a suitable text header;
2. the number of basis atoms;
3. the three components of the unit cell $\underline{a}$ vector;
4. the three components of the unit cell $\underline{b}$ vector;
5. the three components of the unit cell $\underline{c}$ vector;
6. the integer multiples of the $\underline{a}$, $\underline{b}$, $\underline{c}$, in the simulation cell; and
7. the name and fractional coordinates of the basis atoms.

The generated atomic lattice is output in a file called LATTICE, which is directly readable by DL_POLY. (See description of CONFIG file.)

6.1.5.3 Dependencies

- dcell

6.1.5.4 Parameters

integers:

mxatom	1000	maximum number of basis atoms

6.1.5.5 Input

Interactive:

record 1:

title	a80	title for output file

record 2:

natms	integer	number of basis atoms (max 1000)

record 3:

$\underline{a}$	3×real	unit cell vector $\underline{a}$

record 4:

$\underline{b}$	3×real	unit cell vector $\underline{b}$

record 5:

$\underline{c}$	3×real	unit cell vector $\underline{c}$

record 6:

nx,ny,nz	3×integer	multipliers of $\underline{a}$, $\underline{b}$, $\underline{c}$

records 7 to natms+6:

name,x,y,z	a8,3×real	names, and coords of basis atoms

6.1.5.6 Comments

If a large number of atoms are present in the basis, it is recommended that the input data be written into an input file and this fed to the program via standard input.

6.1.6 GENLAT.TO

6.1.6.1 Header records

```
      program genlat2
c
c*********************************************************************
c
c     program to generate a truncated octahedral cell from a perfect
c     lattice
c     author - w.smith oct. 1992
c
c*********************************************************************
c
```

6.1.6.2 Function

genlat is used to generate the position coordinates of a **truncated octahedral** simulation cell by replication of a unit cell. i.e. it generates a crystal lattice within the TO simulation cell. A cubic lattice is assumed. The execution is interactive; the user must supply in turn:

1. a suitable text header;
2. the number of basis atoms;
3. the cubic unit cell width;
4. the required spherical cutoff; and
5. the name and fractional coordinates of the basis atoms.

The generated TO simulation cell is output in a file called LATTICE, which is directly readable by DL_POLY. (See description of CONFIG file.) The cell produced will be the smallest TO compatible with the spherical cutoff provided.

6.1.6.3 Dependencies

- none

6.1.6.4 Parameters

integers:

mxatom	10	maximum number of basis atoms

6.1.6.5 Input

Interactive:

record 1:		
title	a80	title for output file
record 2:		
natms	integer	number of basis atoms (max 10)
record 3:		
wdth	real	unit cell width
record 4:		
radius	real	spherical cutoff
records 5 to natms+4:		
name,x,y,z	a8,3×real	names, and coords of basis atoms

6.1.6.6 Comments

If a large number of atoms are present in the basis, it is recommended that the input data be written into an input file and this fed to the program via standard input.

6.1.7 GROFORCE

6.1.7.1 Header records

```
      program groforce

c***********************************************************************
c
c     program to extract GROMOS force field for a protein
c     sequence and interface it to DL_POLY. The data generated
c     in this routine is read in "sysdef" as the force field
c     file "FIELD".
c
c     copyright -  daresbury laboratory 1993
c     author    -  t forester    feb 1993
c
c***********************************************************************
```

6.1.7.2 Function

Creates the CONFIG and FIELD file (using GROMOS force field) for a protein. Input data from the file "PRODATA" is assumed to be compatible with the output from the utility **proseq2**. **groforce** assumes that no hydrogen atoms are present in the input file and that all non-hydrogen atoms are present. The program inserts all non-carbon hydrogens into the structure and makes the protein a Zwitterion. Note that GROMOS is a united atom force field. In addition to input on the standard device the program also requires the files "AMINODAT" and "GROMOS". AMINODAT contains amino acid partial charges, atom types and connectivity information. "GROMOS_param" contains the remaining information on the GROMOS force field.

6.1.7.3 Dependencies

- none

6.1.7.4 Parameters

integers:

nres	1000	maximum number of residues in the structure.
ngro	37	number of different GROMOS atom types.
naa	25	number of different GROMOS atom types.
mlen	25	maximum number of sites in one amino acid
mbnd	30	maximum number of bonds in one amino acid
mgb	83	number of GROMOS bond types
mga	231	number of GROMOS valence angletypes
mgi	55	number of GROMOS improper dihedrals
mgd	51	number of GROMOS dihedrals

6.1.7.5 Input

Data Files:

PRODATA

AMINODAT:

This file is the data bank of Amino acid residues that **groforce** knows about.

record 1:

variable	type	description etc

record 2:

variable	type	description etc

record 3: etc

6.1.7.6 Comments

6.1.8 H2NADD

6.1.8.1 Header records

```
      subroutine h2nadd(cx,cy,cz,nx,ny,nz,h1x,h1y,h1z,h2x
     $     ,h2y,h2z,bl,angd)

c*********************************************************************
c
c     subroutine to add 2 hydrogens to a primary amine site
c
c     copyright daresbury laboratory 1993
c     author -     t forester    feb 1993
c
c*********************************************************************
```

6.1.8.2 Function

Adds two hydrogens to a primary "amine" with a specified bond length and "C-N-H" bond angle.

6.1.8.3 Dependencies

- none

6.1.8.4 Arguments

reals:

cx	input	x coordinate of first "carbon"
cy	input	y coordinate of first "carbon"
cz	input	z coordinate of first "carbon"
nx	input	x coordinate of "nitrogen"
ny	input	y coordinate of "nitrogen"
ny	input	z coordinate of "nitrogen"
h1x	output	x coordinate of first "hydrogen"
h1y	output	y coordinate of first "hydrogen"
h1z	output	z coordinate of first "hydrogen"
h2x	output	x coordinate of second "hydrogen"
h2y	output	y coordinate of second "hydrogen"
h2z	output	z coordinate of second "hydrogen"
bl	input	bond length
angd	input	bond angle (in degrees)

6.1.8.5 Comments

Both "C-N-H" bond angles are assigned the same value.

6.1.9 HADD

6.1.9.1 Header records

```
      subroutine hadd(cx,cy,cz,ox,oy,oz,hx,hy,hz,bl,angd)

c*********************************************************************
c
c     subroutine to add 1 terminal hydrogen
c
c     copyright daresbury laboratory 1993
c     author -     t forester    feb 1993
c
c*********************************************************************
```

6.1.9.2 Function

Adds one hydrogen to an "oxygen" with a specified bond length and "C-O-H" bond angle.

6.1.9.3 Dependencies

- none

6.1.9.4 Arguments

reals:

cx	input	x coordinate of "carbon"
cy	input	y coordinate of "carbon"
cz	input	z coordinate of "carbon"
ox	input	x coordinate of "oxygen"
oy	input	y coordinate of "oxygen"
oz	input	z coordinate of "oxygen"
hx	output	x coordinate of "hydrogen"
hy	output	y coordinate of "hydrogen"
hz	output	z coordinate of "hydrogen"
bl	input	bond length
angd	input	bond angle (in degrees)

6.1.9.5 Comments

6.1.10 HFILL

6.1.10.1 Header records

```
      program hfill

c*********************************************************************
c
c     DL_POLY utility to add missing hydrogens to a structure.
c
c     required data is
c     atom name, #of H's to add, index of atoms bonding to named atom
c
c     copyright daresbury laboratory 1993
c
c     author t. forester  may 1993
c
c*********************************************************************
```

6.1.10.2 Function

Adds all missing hydrogens to a structure. Input is from the file "HFILL". Output is sent to the file "CONNECT_DAT" and is compatible with the CONNECT_DAT file required by the ambforce utility. All partial charges are set to zero.

6.1.10.3 Dependencies

- hnadd
- hqadd
- h2nadd
- h3nadd

6.1.10.4 Parameters

integers:

mxatm	10000	maximum number of atoms in final structure

6.1.10.5 Input

Data Files:

filename: HFILL

The file consists of blocks of 2 lines of data in the form

record 1

atmnam	a8	atom name
dummy	integer	dummy variable
nh	integer	number of hydrogens to add to this atom.
ibnd(1)	integer	index of first non-H atom bonded to this atom
ibnd(2)	integer	index of second non-H atom bonded to this atom
ibnd(3)	integer	index of third non-H atom bonded to this atom

record 2

xx	real	x coordinate of central site
yy	real	y coordinate of central site
zz	real	z coordinate of central site

6.1.10.6 Comments

The input file can be generated by the utility **fracfill**. An example of input can be found in the file "HFILL.example" in the "utility" directory of DL_POLY".

6.1.11 HNADD

6.1.11.1 Header records

```
      subroutine hnadd(cx,cy,cz,dx,dy,dz,ox,oy,oz,
     x     hx,hy,hz,bl,angd)

c*********************************************************************
c
c     subroutine to add 1 hydrogen onto a secondary amine group
c
c     copyright daresbury laboratory 1993
c     author -    t forester    feb 1993
c
c*********************************************************************
```

6.1.11.2 Function

Adds one hydrogen to a secondary "amine" with a specified bond length and "C-N-H" bond angle.

6.1.11.3 Dependencies

- none

6.1.11.4 Arguments

reals:

cx	input	x coordinate of first "carbon"
cy	input	y coordinate of first "carbon"
cz	input	z coordinate of first "carbon"
dx	input	x coordinate of second "carbon"
dy	input	y coordinate of second "carbon"
dz	input	z coordinate of second "carbon"
ox	input	x coordinate of "nitrogen"
oy	input	y coordinate of "nitrogen"
oz	input	z coordinate of "nitrogen"
hx	output	x coordinate of "hydrogen"
hy	output	y coordinate of "hydrogen"
hz	output	z coordinate of "hydrogen"
bl	input	bond length
angd	input	bond angle (in degrees)

6.1.11.5 Comments

Both "C-N-H" bond angles are assigned the same value.

6.1.12 PROSEQ

6.1.12.1 Header records

```
      program proseq
c
c*********************************************************************
c
c     dl_poly utility program for reading SEQNET data files and
c     extracting the protein structure in a format compatible
c     with the dl_poly CONFIG file
c
c     copyright - daresbury laboratory 1993
c     author    - w. smith jan 1993
c
c*********************************************************************
c
```

6.1.12.2 Function

Extracts a protein structure from a SEQNET file and formats in a way compatible with the CONFIG file require by DL_POLY.

6.1.12.3 Dependencies

- none

6.1.12.4 Parameters

integers:

MXATM	100000	maximum number of atoms in database file
MXSEQ	5000	maximum number of amino acids in sequence

6.1.12.5 Input

The input file must be in SEQNET form.

6.1.12.6 Comments

6.1.13 PROSEQ2

6.1.13.1 Header records

```
      program proseq2
c
c*********************************************************************
c
c     dl_poly utility program for reading SEQNET data files and
c     extracting the protein structure in a format compatible
c     with the groforce PRODATA file
c
c     copyright - daresbury laboratory 1993
c     author    - w. smith jan 1993
c     version 2 - t forester march 1993
c
c*********************************************************************
c
```

6.1.13.2 Function

Extracts a protein structure from a SEQNET file and formats in a way compatible with the input for GROFORCE - the utility for generating the GROMOS force field for a protein. Input is from the standard device.

6.1.13.3 Dependencies

- none

6.1.13.4 Parameters

integers:

MXATM	100000	maximum number of atoms in database file
MXSEQ	5000	maximum number of amino acids in sequence

6.1.13.5 Input

The input file must be in SEQNET form.

6.1.13.6 Comments

The output file is also suitable as a DL_POLY CONFIG file except that some extra lines are included near the top of the file which give the amino acid sequence of the protein.

6.1.14 WATERADD

6.1.14.1 Header records

```
      program wateradd

c*********************************************************************
c
c     DL_POLY utility
c
c     Program to add SPC water molecules to a structure to fill
c     out the MD cell.
c     Assumes atomic postions are in a form compatible
c     with the CONFIG file used in DL_POLY with periodic boundary
c     conditions.
c     Water is added from the 'water.300K' file
c
c     input file = CONFIG
c     output file = CONFIG.plusH2O
c
c     copyright Daresbury laboratory 1993
c     author -     t.  forester  feb 1993
c
c*********************************************************************
```

6.1.14.2 Function

Fills out an MD cell with SPC water molecules. The waters are added from a configuration of SPC waters at 300 K. Input is read from the file "CONFIG" in the "utility" directory. Output is placed in the file "CONFIG.plusH2O" in the "utility" directory.

6.1.14.3 Dependencies

- images - from DL POLY/source
- rotate

6.1.14.4 Parameters

integers:

mxatm	10000	maximum number of atoms in MD cell.

6.1.14.5 Input

Data Files:

filename: CONFIG

See the file CONFIG in the input section for a description of this file.

6.1.14.6 Comments

Cell size, periodic boundary conditions, etc, are set in the CONFIG file.

6.2 Analysis Utilities

6.2.1 ACF3D

6.2.1.1 Header records

```
      program acf3d

c***********************************************************************
c
c     DLPOLY routine for auto correlation function of three
c     dimensional vector -eg. velocities, forces.
c
c     Data is assumed to be formatted and compatible with the
c     HISTORY file written by DL_POLY subroutine traject.f
c
c     parallel version.
c     This program contains dummy routines for machine,gisum,and gdsum
c     to allow running on a single processor.
c
c     copyright daresbury laboratory 1993.
c
c     author t. forester  April 1993.
c
c***********************************************************************
```

6.2.1.2 Function

Calculates the auto-correlation function and its integral for either velocities or forces for one selected atom type. Users will be prompted for:

1. the name fo the trajectory file.
2. the number of atoms in the MD cell.
3. the atom label of interest.
4. the time interval (in ps) at which configurations were written.
5. length of the correlation function
6. output file name
7. vector key (1 for velocities, 2 for forces)

6.2.1.3 Dependencies

- machine - see DL_POLY/source
- gdsum - see DL_POLY/source
- gisum - see DL_POLY/source

6.2.1.4 Parameters

integers:

mxcor	300	maximum length of correlation function.
mxatms	1000	number of atoms in the MD cell.
msatms	500	maximum number of atoms of interest for current processor.

6.2.1.5 Input

Interactive:

record 1:		
dumpfile	a80	name of trajectory file
record 2:		
natms	integer	number of atoms in MD cell
record 3:		
acfme	a8	atom label of interest
record 4:		
tstep	real	time interval (ps) between configurations.
record 5:		
ncor	integer	number of time-steps to correlate over.
record 6:		
outfile	a80	output file name
record 7:		
lvf	integer	function key (1 = velocities, 2 = forces)

data files:

filename: dumpfile (see above)

This file is assummed to be in the formatted form of the DL.POLY output file "HISTORY".

6.2.1.6 Comments

All calculations are done in double precision. This may be changed to single precision if memory requirements pose a problem.

6.2.2 BESTFIT

6.2.2.1 Header records

```
      program bestfit

c***********************************************************************
c
c     Least squares fitting of arbitary functions to data
c     uses Gram-Schmidt Orthogonalisation procedure.
c
c     copyright daresbury laboratory 1993
c     author t. forester may 1993.
c
c***********************************************************************
```

6.2.2.2 Function

The program best-fits a linear combination of M functions to N data points. The program will prompt for the following information from the standard input device:

- number of functions to use
- n data points as : x y

The program will generate a list of the best coefficients for the functions specified and a comparsion for each data point of the input data and the best-fit value.

6.2.2.3 Dependencies

- none

6.2.2.4 Parameters

integers:

M1	20	maximum number of functions
N1	1000	maximum number of data points

6.2.2.5 Input

Interactive:

record 1:		
m	integer	number of functions to use (max value is M1)
record 2:		
x(1)	real	first x data point
y(1)	real	first y data point
record k+1:		
x(k)	real	kth x data point
y(k)	real	kth y data point

last record:

-999	real	end of data list.
-999	real	end of data list.

6.2.2.6 Comments

The functions used are defined starting at line 53 of the source code. Change the definitions to suit and recompile before running.

The number of data points given must equal or exceed the number of functions used.

The program makes use of a version of the matrix inversion utility **syminv**.

6.2.3 CORREL1

6.2.3.1 Header records

```
      program correl1
c
c*********************************************************************
c
c     dl_poly utility program for calculating correlation functions
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith may 1992.
c
c*********************************************************************
c
```

6.2.3.2 Function

correl1 calculates normalised correlation functions by the standard pedestrian method. The data are supplied as columns of numbers in the input file 'input.data', the first column being the time array. Execution is interactive, the user is first required to enter the number of correlation functions desired (max 10) and the number of data columns in the input file (max 15). Next the user must enter the pairs of numbers identifying the columns to be correlated. A column pair being (say) 4 6, meaning the 4'th and 6'th columns are to be correlated (the index does not count the time column). The columns may be up to 500 entries in length.

Each correlation function is output in its own distinct file, with an appropriate name (e.g. x0406 for the above example). Both cross- and auto-correlation functions may be calculated.

6.2.3.3 Dependencies

- None

6.2.3.4 Parameters

integers:

ndiv	500	maximum length of data arrays
mxcols	15	maximum number of data columns in input file (not including time column)
mxcorr	10	maximum number of correlation functions permitted

6.2.3.5 Input

Interactive:

record 1:

npairs	integer	number of correlation functions (max 10)

record 2:

ncols	integer	number of data columns in input file (max 15)

records 3 to npairs + 2:

nc1 nc2	integers	id numbers of columns to be correlated

Data Files:

input.data:

File comprised of N records (N<500) with contents:

time, col_1, col_2, ..., col_{ncols}

where col_i is an entry for the i'th data column. (The data are in free format.)

6.2.3.6 Comments

This method is not recommended for large arrays or large numbers of correlation functions. The related program **correl2** should be used for such cases.

6.2.4 CORREL2

6.2.4.1 Header records

```
      program correl2
c
c*********************************************************************
c
c     dl_poly utility program for calculating correlation functions
c     using the fast fourier transform strategy
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith may 1992.
c
c     note - this program includes a naff fft routine purely for
c     portability and verification reasons. users are strongly
c     recommended to substitute a machine specific fft, which
c     should vastly improve performance.
c
c     note - you will probably need to change some of the array
c     dimensions using the following parameters.
c
c     note - ndiv must be a power of 2. ndiv3 will change according to
c     the fft routine you use. leave ndiv2 as it is.
c
c*********************************************************************
c
```

6.2.4.2 Function

correl2 calculates normalised correlation functions by the Fast Fourier Transform method. The data are supplied as columns of numbers in the input file 'input.data', the first column being the time array. Execution is interactive, the user is first required to enter the number of correlation functions desired (max 10) and the number of data columns in the input file (max 15). Next the user must enter the pairs of numbers identifying the columns to be correlated. A column pair being (say) 4 6, meaning the 4'th and 6'th columns are to be correlated (the index does not count the time column). The columns may be up to 8192 entries in length.

Each correlation function is output in its own distinct file, with an appropriate name (e.g. x0406 for the above example). Both cross- and auto-correlation functions may be calculated.

6.2.4.3 Dependencies

- FFT routine (use machine specific version where available)

6.2.4.4 Parameters

integers:

ndiv	8192	maximum length of data arrays

ndiv2	16384	maximum length of FFT arrays
ndiv3	16834	maximum length of FFT workspace arrays
mxcols	15	maximum number of data columns in input file (not including time column)
mxcorr	10	maximum number of correlation functions permitted

6.2.4.5 Input

Interactive:

record 1:

npairs	integer	number of correlation functions (max 10)

record 2:

ncols	integer	number of data columns in input file (max 15)

records 3 to npairs + 2:

nc1 nc2	integers	id numbers of columns to be correlated

Data Files:

input.data:

File comprised of N records (N<8192) with contents:

time, col_1, col_2, ..., col_{ncols}

where col_i is an entry for the i'th data column. (The data are in free format.)

6.2.4.6 Comments

This method is recommended for large arrays or large numbers of correlation functions. The related program **correl1** is more suitable for smaller cases.

6.2.5 SPECTRUM

6.2.5.1 Header records

```
      program spectrum
c
c*********************************************************************
c
c     dl_poly utility program for spectral analysis using the fast
c     fourier transform
c
c     copyright - daresbury laboratory 1992
c     author    - w. smith june 1992.
c
c     note - this program includes a naff fft routine purely for
c     portability and verification reasons. users are strongly
c     recommended to substitute a machine specific fft, which
c     should vastly improve performance.
c
c     note - you will probably need to change some of the array
c     dimensions using the following parameters.
c
c     note - ndiv must be a power of 2. ndiv2 will change according to
c     the fft routine you use.
c
c*********************************************************************
c
```

6.2.5.2 Function

spectrum calculates the discrete Fourier transform of a number of data columns. The data are supplied as columns of numbers in the input file 'input.data'. Execution is interactive, the user is first required to enter the number of transforms desired (max 10), the number of data columns in the input file (max 15) and the stride index (i.e. the number of records between each data point - not all points in the input need be used). Next the user must enter the numbers identifying the columns to be transformed. The columns may be up to 8192 entries in length.

Each Fourier transform is output in its own distinct file, with an appropriate name (e.g. ft06 for the transform of the 6'th column).

6.2.5.3 Dependencies

- FFT routine (use machine specific version if available)

6.2.5.4 Parameters

integers:

ndiv	8192	maximum length of data arrays
ndiv2	16384	maximum length of FFT arrays
mxcols	15	maximum number of data columns in input file

mxcorr	10	maximum number of transforms permitted

6.2.5.5 Input

Interactive:

record 1:

nspecs	integer	number of Fourier transforms (max 10)

record 2:

ncols	integer	number of data columns in input file (max 15)

record 3:

nstride	integer	integer stride through data columns

records 4 to nspecs+3:

nc1 nc2	integers	id numbers of columns to be transformed

Data Files:

input.data:

File comprised of N records (N<8192) with contents:

$col_1, col_2, \ldots, col_{ncols}$

where col_i is an entry for the i'th data column. (The data are in free format.)

6.2.5.6 Comments

This program uses a Blackman-Harris 4 term window function. If you don't like it, use your own!

6.2.6 STATBLOCK

6.2.6.1 Header records

```
      program statblock
c
c*********************************************************************
c
c     blocking method for estimating standard error of mean
c
c     (reference - Flyvbjerg and Petersen  JCP 91 (1989) 461)
c
c     author - w. smith nov 1992
c
c*********************************************************************
c
```

6.2.6.2 Function

statblock calculates the mean values of a set of data columns and performs a blocking analysis to determine the correct standard error. The method is that of Flyvbjerg and Petersen. Up to 20 columns of data can be processed at once. The data are supplied as columns of numbers in an input file. Execution is interactive, the user must first supply the name of the input file holding the data. Next the number of data columns in the input file (max 20) and the number of columns to be processed (max 20) are entered. Finally the user must enter the numbers identifying the columns selected for processing. The columns may be up to 10000 entries in length. Results are streamed to standard output.

6.2.6.3 Dependencies

- None

6.2.6.4 Parameters

integers:

ndeg	13	max degree of blocking to be performed
mxcols	20	maximum number of data columns in input file
mxrows	10000	maximum number of entries per data column

6.2.6.5 Input

Interactive:

record 1:

statfile	a40	name of data file to be processed

record 2:

mcols	integer	number of data columns in data file

record 3:

ncols	integer	number of columns to be processed

record 4:

$id_1 \ldots id_{ncols}$	integers	id numbers of columns to be processed

Data Files:

statfile:

File comprised of N records (N<10000) with contents:

$col_1, col_2, \ldots, col_{ncols}$

where col_i is an entry for the i'th data column. (The data are in free format.)

6.2.6.6 Comments

It is advisable to plot the calculated standard errors as a function of degree of blocking, to determine the optimal value.

6.3 Miscellaneous Utilities

6.3.1 CXA.TIMCHK

6.3.1.1 Header records

```
      subroutine timchk(ktim,time)
c
c*********************************************************************
c
c     CONVEX specific timing routine (time elapsed in seconds)
c
c*********************************************************************
c
```

6.3.1.2 Function

CONVEX specific timing routing for use with DL_POLY.

6.3.1.3 Dependencies

- none

6.3.1.4 Arguments

integers:

ktim	input	printing key

reals:

time	output	elapsed time in seconds

6.3.1.5 Comments

Use this routine in place of the DL_POLY/source routine "timchk.f" to compile DL_POLY for use on a CONVEX.

6.3.2 ROTATE

6.3.2.1 Header records

```
      subroutine rotate(xo,yo,zo,x1,y1,z1,x2,y2,z2,alp
     x      ,bet,gam)

c*********************************************************************
c
c     DL_POLY utility to rotate water molecule through the
c     Euler angles (alp,bet,gam) in lab fixed axes system.
c     The first data point is taken as the origin for the rotation
c
c     copyright daresbury laboratory 1993
c     author  - t. forester     july 1993
c
c*********************************************************************
```

6.3.2.2 Function

Creates a water molecule with the dipole direction along the x axis then rotates it through the Euler angles (α, β, γ) in the laboratory fixed frame.

6.3.2.3 Dependencies

- none

6.3.2.4 Arguments

reals:

xo	input	x coordinate of oxygen.
yo	input	y coordinate of oxygen.
zo	input	z coordinate of oxygen.
x1	output	x coordinate of first hydrogen.
y1	output	y coordinate of first hydrogen.
z1	output	z coordinate of first hydrogen.
x2	output	x coordinate of second hydrogen.
y2	output	y coordinate of second hydrogen.
z2	output	z coordinate of second hydrogen.
alp	input	α (in degrees).
bet	input	β (in degrees).
gam	input	γ (in degrees).

6.3.2.5 Comments

6.3.3 SORT

6.3.3.1 Header records

```
      subroutine sort(n,index,x)
c
c*********************************************************************
c
c     DL_POLY utility
c     shell sort of index array
c
c     copyright daresbury laboratory 1993
c     author -        t.forester may 1993
c
c*********************************************************************
c
```

6.3.3.2 Function

Sorts n real indexed numbers in to ascending order.

6.3.3.3 Dependencies

- none

6.3.3.4 Arguments

integers:

n	input	number of items to be sorted.

integer arrays:

index	output	index of sorted list.

real arrays:

x	input	numbers to be sorted.

6.3.3.5 Comments

The sort is based on the *index* not the array itself. The array X is unchanged on exit from this routine. Thus X(INDEX(1)) is the smallest number in the array and X(INDEX(N)) the largest.

6.3.4 SYMINV

6.3.4.1 Header records

```
      subroutine syminv(nnn,aaa,bbb)
c
c*********************************************************************
c
c     routine to invert a real symmetric matrix (reference:
c     computing methods v.ii, i.s. berezin and n.p.zhidkov,
c     pergamon press 1965). note that the matrices are
c     packed in the minimum storage mode, (i.e. a(k)=a(i,j),
c     where i>j and k=(i*(i-1))/2+j ).
c     the matrices aaa and bbb may be equivalenced though
c     this will destroy the contents of the original
c     array.
c
c     general version for all real symmetric matrices
c
c     author w.smith (added to dl_poly july 1993)
c
c*********************************************************************
c
```

6.3.4.2 Function

Finds the inverse of a real symmetric matrix.

6.3.4.3 Dependencies

- none

6.3.4.4 Arguments

integers:

nnn	input	dimension of matrix

real arrays:

aaa	input	matrix in triangular form (*)
bbb	output	inverse matrix in triangular form (*)

6.3.4.5 Comments

The program makes use of minimum storage requirements for the matrices. The array AAA has NNN(NNN+1)/2 entries defining the lower triangle of the real symmetric matrix. The inverse matrix (BBB) is given in the same form.

Chapter 7

Appendices

Appendix 1. DL_POLY Arrays

a(9)	work array
aaa(9)	work array
aaa(nnn)	work array
amsd(mxsvdw)	mean squared displacement array
b(9)	work array
bbb(10)	work array
bbb(nnn)	work array
buffer(10)	work array
buffer(4)	work array
buffer(8)	work array
buffer(mxbuff)	main coordinate transfer buffer
cell(9)	simulation cell vectors (1-3)=A vector,
celprp(10)	cell property array
chge(msatms)	atomic charge array
chgsit(mxsite)	atomic site charge array
ckc(msatms)	cosine of Fourier transform of atomic charge density
cks(msatms)	sine of Fourier tranform of atomic charge density
clm(msatms)	partial FT (cosine) of atomic charge density
dens(mxsvdw)	number density of unique atom types
dx0(mxcons)	x component of bond vectors after unconstrained step
dxt(mxcons)	x component of bond vector during SHAKE iteration
dxx(mxcons)	x component of bond vectors (at start of SHAKE)
dy0(mxcons)	y component of bond vectors after unconstrained step
dyt(mxcons)	y component of bond vector during SHAKE iteration
dyy(mxcons)	y component of bond vectors (at start of SHAKE)
dz0(mxcons)	z component of bond vectors after unconstrained step
dzt(mxcons)	z component of bond vector during SHAKE iteration
dzz(mxcons)	z component of bond vectors (at start of SHAKE)
elc(msatms,0:kmax)	cosine of exp(ik.r(x)) terms for Ewald sum
els(msatms,0:kmax)	sine of exp(ik.r(x)) terms for Ewald sum
emc(msatms,0:kmax)	cosine of exp(ik.r(y)) terms for Ewald sum
ems(msatms,0:kmax)	sine of exp(ik.r(y)) terms for Ewald sum
enc(msatms,0:kmax)	cosine of exp(ik.r(z)) terms for Ewald sum
ens(msatms,0:kmax)	sine of exp(ik.r(z)) terms for Ewald sum
erc(mxgrid)	complementary error function (potential) array
fer(mxgrid)	complementary error function (forces) array
flx(mxatms)	x component of secondary forces (mult. timestep)
fly(mxatms)	y component of secondary forces (mult. timestep)
flz(mxatms)	z component of secondary forces (mult. timestep)
fxx(mxatms)	x component of atomic forces array
fyy(mxatms)	y component of atomic forces array
fzz(mxatms)	z component of atomic forces array
ggg(mxgrid,mxvdw)	short-range potential (forces) interpolation table
ibuff(*)	integer buffer for global integer summations
ilist(*)	abbreviated verlet neighbour list for specific atom
irdf(mxrdf,mxvdw)	rdf histogram array (rdf accumulator)

keyang(mxtang)	type key for all valence angle potentials
keybnd(mxtbnd)	type key for all bond potentials
keydih(mxtdih)	type key for all dihedral angle potentials
lashap(mxproc)	'last atom' marker for RD-SHAKE inter-node comms.
lentry(msatms)	'last atom' marker for Verlet neighbour list
lexatm(msatms,mxexcl)	'excluded atom' list for pair force terms
lexsit(mxsite,mxexcl)	'excluded site' list for pair force terms
lishap(mxlshp)	'shared atoms' list for RD-SHAKE comms.
list(msatms,mxlist)	Verlet neighbour list
listang(mxangl,4)	Valence angle potential bookkeeping list
listbnd(mxbond,3)	Bond potential bookkeeping list
listcon(mxcons,3)	Bond constraint bookkeeping list
listdih(mxdihd,5)	Dihedral angle potential bookkeeping list
listin(mxatms)	Incoming 'shared atoms' transfer buffer
listme(mxatms)	Outgoing 'shared atoms' transfer buffer
listot(mxatms)	Global 'shared atoms' register
lstang(mxtang,3)	Valence angle potential site list
lstbnd(mxtbnd,2)	Bond potential site list
lstcon(mxtcon,2)	Bond constraint site list
lstdih(mxtdih,4)	Dihedral angle potential site list
lstvdw(mxvdw)	pointer from potential index to parameter table
ltpsit(mxsite)	atomic type index for molecular sites
ltpvdw(mxvdw)	potential energy type index
ltype(mxatms)	atomic type index for all atoms in simulation
nexatm(msatms)	'last entries' for excluded atom list
noxatm(msatms)	'last atom' counter for excluded atom list in use
numang(mxtmls)	number of valence angles in each molecule
numbonds(mxtmls)	number of chemical bonds in each molecule
numcon(mxtmls)	number of bond constraints in each molecule
numdih(mxtmls)	number of dihedral potentials in given molecule
nummols(mxtmls)	population of each molecule type in simulation
numsit(mxtmls)	number of distinct sites in each molecule
numtyp(mxsvdw)	number of atoms of given type in simulation
parvdw(5)	short range potential parameters input array
prmang(mxtang,2)	valence angle potential parameters array
prmbnd(mxtbnd,3)	chemical bond potential parameters array
prmcon(mxtcon)	bond constraint parameters (bondlengths)
prmdih(mxtdih,3)	dihedral angle potential parameters array
prmvdw(mxvdw,5)	short range potential parameters array
ravval(mxnstk)	rolling averages array
rcell(9)	reciprocal of simulation cell matrix
roti(9)	moment of inertia tensor
rotinv(9)	inverse of moment of inertia tensor
rsqdf(mxxdf)	square of interatomic distance
slm(msatms)	partial FT (sine) of atomic charge density
ssqval(mxnstk)	accumulators of squared system variables
stkval(mxstak,mxnstk)	history stack of system variables

stpval(mxnstk)	array of current system variables
sumval(mxnstk)	accumulators for system variables
txx(mxatms)	x component of coordinate transfer buffer
tyy(mxatms)	y component of coordinate transfer buffer
tzz(mxatms)	z component of coordinate transfer buffer
uxx(mxatms)	x component of velocity work array
uyy(mxatms)	y component of velocity work array
uzz(mxatms)	z component of velocity work array
vvv(mxgrid,mxvdw)	short-range potential (energy) interpolation table
vx1(msatms)	x component of velocity before integration
vxx(mxatms)	x component of atomic velocities
vy1(msatms)	y component of velocity before integration
vyy(mxatms)	y component of atomic velocities
vz1(msatms)	z component of velocity before integration
vzz(mxatms)	z component of atomic velocities
weight(natms)	atomic mass array
xdab(*)	x component of atomic separation (work array)
xdbc(*)	x component of atomic separation (work array)
xdcd(*)	x component of atomic separation (work array)
xdf(*)	x component of interatomic distance vector
xx0(mxatms)	x component of atomic displacement (for MSD)
xold(msatms)	x component of old atomic position (in vertest.f)
xx1(mxatms)	x component of 1st derivative for Gear algorithm
xx2(mxatms)	x component of 2nd derivative for Gear algorithm
xx3(mxatms)	x component of 3d derivative for Gear algorithm
xx4(mxatms)	x component of 4th derivative for Gear algorithm
xx5(mxatms)	x component of 5th derivative for Gear algorithm
xxt(mxatms)	x component of SHAKE increment vector
xxx(mxatms)	x component of atomic position vectors
ydab(*)	y component of atomic separation (work array)
ydbc(*)	y component of atomic separation (work array)
ydcb(*)	y component of atomic separation (work array)
ydf(*)	y component of interatomic distance vector
yy0(mxatms)	y component of atomic displacement (for MSD)
yold(msatms)	y component of old atomic position (in vertest.f)
yy1(mxatms)	y component of 1st derivative for Gear algorithm
yy2(mxatms)	y component of 2nd derivative for Gear algorithm
yy3(mxatms)	y component of 3d derivative for Gear algorithm
yy4(mxatms)	y component of 4th derivative for Gear algorithm
yy5(mxatms)	y component of 5th derivative for Gear algorithm
yyt(mxatms)	y component of SHAKE increment vector
yyy(nxtms)	y component of atomic position vectors
zdab(*)	z component of atomic separation (work array)
zdab(*)	z component of atomic separation (work array)
zdbc(*)	z component of atomic separation (work array)
zdcb(*)	z component of interatomic distance vector
zdf(*)	z component of interatomic distance vector

zumval(mxnstk)	rolling average accumulators of system variables
zz0(mxatms)	z component of atomic displacement (for MSD)
zold(msatms)	z component of old atomic position (in vertest.f)
zz1(mxatms)	z component of 1st derivative for Gear algorithm
zz2(mxatms)	z component of 2nd derivative for Gear algorithm
zz3(mxatms)	z component of 3d derivative for Gear algorithm
zz4(mxatms)	z component of 4th derivative for Gear algorithm
zz5(mxatms)	z component of 5th derivative for Gear algorithm
zzt(mxatms)	z component of SHAKE increment vector
zzz(mxatms)	z component of atomic position vectors