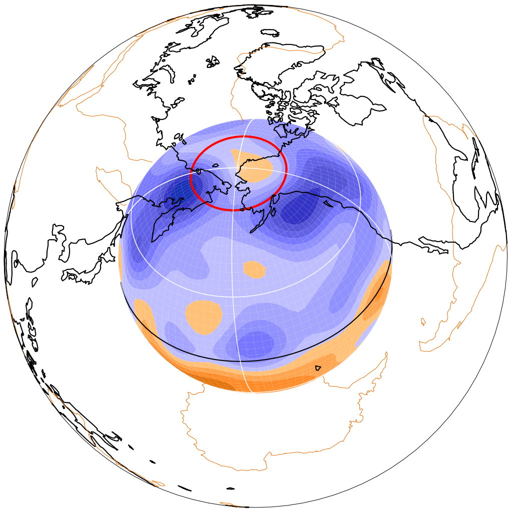
LEOPACK



cicibcdts2

Conducting **I**nner **C**ore **I**nhomogeneous **B**oundary **C**onvective **D**ynamo **T**ime-**S**tep code

Steven J. Gibbons, Oslo

Original document: November 21st, 2001. Updated: October 17th, 2022.

1 cicibcdts2

Conducting **I**nner **C**ore **I**nhomogeneous **B**oundary **C**onvective **D**ynamo **T**ime-**S**tep code

This program time-steps the equations

$$c_a \frac{\partial \Theta}{\partial t} = c_d \nabla^2 (\Theta + \varepsilon T_{\mathrm{a}}) + b_1 u_r r + b_2 \frac{u_r}{r^2} - c_c \boldsymbol{u}.\nabla(\Theta + \varepsilon T_{\mathrm{a}}) \tag{1}$$

and

$$\begin{aligned} c_e \frac{\partial \boldsymbol{\omega}}{\partial t} &= -c_f \nabla \times (\boldsymbol{u}.\nabla \boldsymbol{u}) - c_g \nabla \times (\boldsymbol{k} \times \boldsymbol{u}) \\ &\quad + c_h \nabla \times [(\Theta + \varepsilon T_{\mathrm{a}})\boldsymbol{r}] + c_j \nabla \times [(\nabla \times \boldsymbol{B}) \times \boldsymbol{B}] + c_i \nabla^2 \boldsymbol{\omega}. \end{aligned} \tag{2}$$

and

$$c_k \frac{\partial \boldsymbol{B}}{\partial t} = c_m \nabla \times (\boldsymbol{u} \times \boldsymbol{B}) + c_l \nabla^2 \boldsymbol{B}, \tag{3}$$

where Θ is the part of the temperature distribution which satisfies homogeneous boundary conditions, $\boldsymbol{u}$ is the velocity and $\boldsymbol{\omega}$ is the vorticity (with $\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$). The magnetic field matches to a potential field at the outer boundary. The inner core can be conducting, in which case the velocity must satisfy rigid boundary conditions, or, if there are no grid nodes within the inner core, an insulator.

The inhomogeneous part of the temperature, T_{a}, is provided prior to execution in the standard form of a `.ints`, `.vecs` and `.xarr` file. The standard way of generating these files is using the program `itfvf` although any file containing only temperature spherical harmonic radial functions is valid. The only condition is that the inner and outer boundaries on the temperature function coincide with those for the homogeneous boundary solution. Neither the number of radial grid nodes, nor their pattern of spacing, needs to match exactly as the temperature heterogeneity function is interpolated onto the mesh used by the main solution.

The initial solution is stored in two sets of files. The files

```
stem.intsv
stem.vecsv
stem.xarrv
```

contain all information about the velocity and the temperature distribution. The file `stem.xarrv` contains the locations of the `NRV` grid nodes in the outer core at which the velocity and temperature are stored.

The files

```
stem.intsm
stem.vecsm
stem.xarrm
```

contain all information about the magnetic field. The file `stem.xarrm` contains the locations of the `NRM` grid nodes at which the magnetic field is stored.

The last `NRV` of these `NRM` grid nodes must correspond EXACTLY with the grid nodes for the velocity and temperature distributions. There will remain `NRIC = NRM - NRV` grid nodes within the inner core. It is quite acceptable for `NRIC` to be zero, which allows this code to time-step a solution with an insulating inner core. In the usual case, the first node for the magnetic field will be at radius $r = 0$: in which case the inner core is a conductor with the same conductivity as the outer core. If r_1 is the radius of the first magnetic field node, the case $0 < r_1 < r_\mathrm{i}$ corresponds to an insulating inner core, within a conducting inner core. I have never tried this as I doubt there is a useful application(?), but it is quite permissible from the program's point of view.

The program is compiled by typing

```
make cicibcdts2
```

within this directory. Once the executable is created, begin execution by typing

```
cicibcdts2 < inputfile
```

The inputs file must have the following format.

```
*
* input file for cicibcdts2
*
e0_10_y22_cic_Ra75000                    : filename stem
y22_cic_Ra75000.intsv                    : ints file velocity
y22_cic_Ra75000.vecsv                    : vecs file velocity
y22_cic_Ra75000.xarrv                    : xarr file velocity
y22_cic_Ra75000.intsm                    : ints file magnetic field
y22_cic_Ra75000.vecsm                    : vecs file magnetic field
y22_cic_Ra75000.xarrm                    : xarr file magnetic field
*
* Following three lines are the inhomogeneous boundary
* temperature functions. These are most likely genereated
* by the itfvf program
*
y22_boundary_temp.ints                   : boundary temp. harms
y22_boundary_temp.vecs                   : boundary temp. vecs
y22_boundary_temp.xarr                   : boundary temp. xarrs
  0.10                                   : SCAL
   -2             : NTHP (-1 --> LH+2, -2 --> 3*LH/2 )
 1.0   0.0   0.8284023     1.0   1.0    : CA  CB1  CB2   CC   CD
 1.0     1.0     2000.0   75000.0   1.0 : CE   CF   CG   CH   CI
 200.0   1.0      0.2      1.0          : CJ   CK   CL   CM
 0.4     0.00005   0.0000               : CFAC    DELTAT   STIME
* ioutf is set to 0 for no output, 1 for file and 6 for screen
* nts is the total number of time-steps required
* ntsbb is the number of time-steps between backups
* ntsbs is the number of time-steps between snapshots
   0     25     80  50000               : IOUTF  NTS NTSBB NTSBS
   6     0.1000                         : ITMX   DTOL
   1     1000    1000                   : NTSBSE NTSBLE NTSBME
```

Any line in the input file beginning with an asterisk, `*`, is ignored by the program and can thus be used to enter comments and notes.

The arguments are as follows

- `Filename stem`: First characters in output files to be generated by current run.

- `ints file velocity`: name of already existing indices file describing initial velocity and temperature functions.

- `vecs file velocity`: name of already existing vector file describing initial velocity and temperature functions. Must contain the same number of radial functions as indicated in the `.intsv` file.

- `xarr file velocity`: name of already existing radial spacings file describing initial velocity and temperature functions. Must contain the same number of radial grid nodes as indicated in the `.vecsv` file.

- `ints file magnetic field`: name of already existing indices file describing initial magnetic field functions.

- `vecs file magnetic field`: name of already existing vector file describing initial magnetic field. Must contain the same number of radial functions as indicated in the `.intsm` file.

- `xarr file magnetic field`: name of already existing radial spacings file describing initial magnetic field. Must contain the same number of radial grid nodes as indicated in the `.vecsm` file. See note above on compatibility of `vecsv` and `vecsm` files.

- `SCAL`: Multiplier for the inhomogeneous temperature function.

- `NTHP`: number of θ points for Gaussian quadrature in spherical transforms. Must be greater than `LH` - the highest spherical harmonic degree in the expansions. Is limited by the parameter `NTHMAX`. There are two "special" values of `NTHP` which may be entered and are infact recommended. `NTHP = -1` sets $N_\theta = L_{\max} + 2$ and `NTHP = -2` sets $N_\theta = 3L_{\max}/2$. This higher number of θ points eliminates aliassing effects in the transforms.

- `CA`: Scaling parameter c_a in Equation (1).

- `CB1`: Scaling parameter b_1 in Equation (1).

- `CB2`: Scaling parameter b_2 in Equation (1).

- `CC`: Scaling parameter c_c in Equation (1).
- `CD`: Scaling parameter c_d in Equation (1).
- `CE`: Scaling parameter c_e in Equation (2)
- `CF`: Scaling parameter c_f in Equation (2)
- `CG`: Scaling parameter c_g in Equation (2)
- `CH`: Scaling parameter c_h in Equation (2)
- `CI`: Scaling parameter c_i in Equation (2)
- `CJ`: Scaling parameter c_j in Equation (2)
- `CK`: Scaling parameter c_k in Equation (3)
- `CL`: Scaling parameter c_l in Equation (3)
- `CM`: Scaling parameter c_m in Equation (3)
- `CFAC`: Determines how explicit or implicit the time-stepping scheme is. c must be strictly greater than zero and strictly less than 1. The time derivative of a function f, $\partial_t f$, over a time-step is a linear combination of $\partial_t f$ at the current time-step (i.e. $\partial_t f^i$) and $\partial_t f$ at the next time-step (i.e. $\partial_t f^{i+1}$) such that

  $$\partial_t f = c\partial_t f^i \ + \ (1-c)\partial_t f^{i+1}$$

  If $c = 0.5$, the weighting is equal and we have a Crank-Nicolson scheme. $c < 0.5$ puts greater weight on the value at the next time-step and makes the scheme more implicit (usually more stable, and found to be necessary for many calculations). $c = 0.4$ has been used often. For circumstances where it is stable, $c = 0.5$ is the most accurate.
- `DELTAT`: fixed value of the time-step, Δt.
- `STIME`: starting time. Usually set to zero, but can be useful to set it to higher values for "smooth continuation" of runs which have expired.
- `IOUTF`: output writing flag. Set to 0 to stop all output. Set to 1 to write to a file and set to 6 to write to screen. The amount of output is enormous and so should always be switched off for serious calculations. The option is intended for the initialisation of new runs. If the time-stepping procedure fails, it is

a good idea to switch output on. This may reveal whether the predictor-corrector scheme is not converging (time-step too large?) or whether some other fault has occured.

- `NTS`: the total number of time-steps to be taken.

- `NTSBB`: number of time-steps between each output of “backup solution vectors”. These are vectors which have the same format as the `.vecsv` and `.vecsm` files which define the initial solution and always have the filenames *root.bvv* and *root.bvm*: so in the example here, every 100 time-steps, the solution will be written out to the files `example_bOUTPUT.bvv` and `example_bOUTPUT.bvm`. The solution snapshot after 200 ime-steps will therefore overwrite the snapshot after 100 time-steps. The backup files are basically a safety mechanism, incase the system crashes during a run, but are also useful for monitoring the progress of a solution without generating an enormous volume of files. `.bvv` and `.bvm` files should be output sufficiently often that, should the computer crash, for example, not too much CPU time will be lost, but not so often that time is wasted in outputting.

- `NTSBS`: number of time-steps between each labelled solution vector output. These files are exactly the same as the `.bv` and `.bvm` backup files, apart from that their filename indicates the time-step number at which the output was made. The only problem with files such as these is that they take up huge amounts of disk-space. It is therefore necessary to limit the number made.

- `ITMX`. The maximum number of iterations allowed in the predictor-corrector time-stepping scheme.

- `DTOL` required norm between successive iterations in the predictor-corrector scheme. A large value for `DTOL` essentially says *accept the first iteration of the corrector*. It may be worth setting this to a very small value for a trial run in order to find, by trial and error, how small a time-step is required such that the solution converges to the desired precision with a single iteration.

- `NTSBFE` number of time-steps between each evaluation of kinetic energy, magnetic energy and (rather arbitrarily selected) measurements of the velocity and temperature.

- `NTSBLE` number of time-steps between each breakdown of magnetic and kinetic energy in terms of the spherical harmonic degree, l.

- `NTSBME` number of time-steps between each breakdown of magnetic and kinetic energy in terms of the spherical harmonic order, m.

1.1 Subprograms required for cicibcdts2

SUBS subroutines

```
fopen.f shkeer.f hmfrd.f svfrd.f xarrrd.f svfdcf.f
fdcmbd.f iiasce.f mc2scv.f pvccf.f nphpf.f gauwts.f
schnla.f pvtsmf.f tvtsmf.f tmtsmf.f pftsmf.f tftsmf.f
avbmbr.f vobmbr.f xarrcc.f asvcpl.f svfwt.f pvvcpl.f
vecop.f nsvhst.f nsvbta.f rsdv2c.f sf2vgc.f sf2sdc.f
xsvsdc.f rsdv2d.f rsdv2e.f sf2vgd.f sf2sdd.f xsvsdd.f
xsvsde.f mdcxse.f fclose.f bihfrd.f itvaaf.f fnamer.f
asvdr.f ldgnmf.f gfdcfd.f matop.f amlica.f amsdea.f
amdlt.f vobmar.f nmlica.f dvecz.f fftrlv.f svrint.f
matind.f bmrcop.f powtwo.f
```

SUBS double precision function

```
sqrll1.f emmult.f pmm.f   pmm1.f  plm.f   dpmm.f
dpmm1.f dplm.f  dl.f
```

SUBS integer function

```
indfun.f
```

BLAS double precision function

```
dnrm2.f
```

BLAS integer function

```
idamax.f
```

BLAS subroutines

```
dcopy.f dgbmv.f daxpy.f dgemv.f dger.f  dswap.f
dtbsv.f dgemm.f dscal.f dtrsm.f dtrmm.f dtrmv.f
```

LAPACK subroutines

```
dgbtrs.f dgbtrf.f xerbla.f dgetrf.f dgetri.f dgbtf2.f
dlaswp.f dgetf2.f dtrtri.f dtrti2.f
```

LAPACK integer function

```
ilaenv.f
```

LAPACK logical function

```
lsame.f
```

1.2 Run-time limitations

Several parameters are set at the outset which limit the physical size of the problem.

```
 INTEGER          LHMAX, NTHMAX, NPHMAX, NPMAX
 PARAMETER      ( LHMAX = 52, NTHMAX = 100, NPHMAX = 100,
1                 NPMAX = (LHMAX+1)*(LHMAX+2)/2 )
 INTEGER          NRVMAX, NRMMAX, NH1MAX, NH2MAX, NH3MAX,
1                 NH4MAX, NH5MAX, NHVMAX, NHMMAX, NIVMAX, NIMMAX,
2                 NCFM, NDRVM, NBNM, NNDM, NIV1MX, NIV2MX,
3                 NIV3MX, NIV4MX, NIV5MX, NCMXX,
4                 NIVCMX, NHCMAX, NRMAXC, NDCSV, NDCSM
 PARAMETER      ( NRVMAX = 40, NRMMAX = 60, NH1MAX = 300,
1                 NH2MAX = 300, NH3MAX = 300, NH4MAX = 300,
2                 NH5MAX = 300, NHVMAX = NH1MAX+NH2MAX+NH3MAX,
3                 NHMMAX = NH4MAX+NH5MAX, NCMXX = 22,
4                 NIVMAX = NHVMAX*NRVMAX,
5                 NIMMAX = NHMMAX*NRMMAX )
 PARAMETER      ( NIV1MX = NH1MAX*NRVMAX,
1                 NIV2MX = NH2MAX*NRVMAX,
2                 NIV3MX = NH3MAX*NRVMAX,
3                 NIV4MX = NH4MAX*NRMMAX,
4                 NIV5MX = NH5MAX*NRMMAX )
 PARAMETER      ( NDCSV = 3, NDCSM = 2+LHMAX, NBNM = 3,
1                 NCFM = 2*NBNM+1, NDRVM = 4, NNDM = 4 )
 PARAMETER      ( NRMAXC = 100,
4                 NHCMAX = 300, NIVCMX = NHCMAX*NRMAXC )
```

If the values are insufficient, then change them and recompile. (Note that NCMXX, NDCSV and NDRVM are not size dependent and should not be changed.)

- LHMAX is the highest permitted spherical harmonic degree, l.

- NTHMAX is the maximum permitted number of grid nodes in θ for Gaussian quadrature in the spherical transforms.

- NPHMAX is the maximum permitted number of grid nodes in ϕ for the Fast Fourier Transforms.

- NPHMAX is the maximum permitted number of grid nodes in ϕ for the Fast Fourier Transforms.

- NRVMAX is the maximum permitted number of radial grid nodes for the velocity and temperature functions (i.e. the outer core).

- NRMMAX is the maximum permitted number of radial grid nodes for the magnetic field (inner and outer cores).

- NH1MAX is the maximum permitted number of poloidal velocity spherical harmonic radial functions.

- NH2MAX is the maximum permitted number of toroidal velocity spherical harmonic radial functions.

- NH3MAX is the maximum permitted number of temperature spherical harmonic radial functions.

- NH4MAX is the maximum permitted number of poloidal magnetic field spherical harmonic radial functions.

- NH5MAX is the maximum permitted number of toroidal magnetic field spherical harmonic radial functions.

- NBNM is the number of upper (and lower) diagonals in the banded matrix. NBNM = 3 is recommended value.

- NNDM is the number of nodes used for interpolating radial functions for evaluating components using **RTPFCE**.

- NRMAXC is the maximum permitted number of radial rid nodes for the imposed temperature function.

- NHCMAX is the maximum permitted number of spherical harmonic radial functions for the imposed temperature function.

1.3 Outputs from CICIBCDTS2

In addition to the snapshots of solution vectors, the number and names of which being determined by the integer numbers NTS, NTSBB and NTSBS, there are four files output: root.mnrg, root.nrg, root.comps and root.log.

root.nrg has a line added to it every NTSBFE time-steps. This line contains, in the format (1PD16.7,1PD16.7,1PD16.7,1PD16.7), the four numbers DTIME, DTOTKE, DEATOT and DTORKE.

DTIME is the time elapsed, DTOTKE is the volume integral of the total kinetic energy, DEATOT is the volume integral of the kinetic energy contained in the equatorially anti-symmetric components only, and DTORKE is the volume integral of the kinetic energy contained in the toroidal components only.

root.mnrg has a line added to it every NTSBFE time-steps. This line contains, in the format (1PD16.7,1PD16.7,1PD16.7,1PD16.7), the four numbers DTIME, DTOTME, DEATOM and DTORME.

DTIME is the time elapsed, DTOTME is the volume integral of the total magnetic energy, DEATOM is the volume integral of the magnetic energy contained in the equatorially anti-symmetric components only, and DTORME is the volume integral of the magnetic energy contained in the toroidal components only.

root.comps has a line added to it every NTSBFE time-steps. This line contains, in the format (5(1PD16.7)) the five numbers DTIME, VRAD, VPHI, BTHE and TEMP_TOT.

DTIME is the time elapsed, VRAD, VPHI, BTHE and TEMP_TOT are the values of v_r v_ϕ, B_θ and Θ evaluated at $r = (r_{\rm i} + r_{\rm o})/2$, $\theta = \pi/2$ and $\phi = 0$. (This seemingly arbitrary choice was made because these were the specifications for contributions to the numerical dynamo benchmark project, [CAC+01].)

root.log reports back on the physical size of the problem, the allocation of finite difference schemes and echoes back the physical parameters read in. Then, every NTSBLE time-steps, a breakdown of kinetic and magnetic energy is done as a function of spherical harmonic degree, l. For every l, such that the kinetic energy stored in spherical harmonic radial functions with degree l is not negligible (i.e. less than a specified parameter DLOW), a line is written to root.log in the format ('vL= ',I5,1PD16.7,1PD16.7,1PD16.7,1PD16.7) with the numbers L, DTIME, TOTAL, EA and TOR. L is the spherical harmonic degree, l, DTIME the time elapsed and TOTAL, EA and TOR are the volume integrals of the total kinetic energy, the equatorially anti-symmetric kinetic energy and the toroidal kinetic energy respectively. For example, the lines

```
vL=    31   6.0000000D-05   8.1171900D-03   0.0000000D+00   8.1171900D-03
vL=    32   6.0000000D-05   2.5628502D-03   0.0000000D+00   0.0000000D+00
```

indicate that at $t = 0.00006$, the volume integral of the kinetic energy contributed by velocity harmonics with $l = 31$ was 0.00812, of which none was stored in poloidal or equatorially anti-symmetric terms. The volume integral of the kinetic energy contributed by velocity harmonics with $l = 32$ was 0.00256, of which none was stored in toroidal or equatorially anti-symmetric terms. This is a consequence of an equatorial symmetry restriction on the velocity. There is a similar breakdown of magnetic energy, the output lines being preceded by a lower-case m as opposed to a lower-case v. Specifically, the lines

```
mL=    31   6.0000000D-05   1.5868739D-04   1.5868739D-04   0.0000000D+00
mL=    32   6.0000000D-05   4.0099873D-04   4.0099873D-04   4.0099873D-04
```

indicate that at $t = 0.00006$, the volume integral of the magnetic energy contributed by velocity harmonics with $l = 31$ was 0.000159, of which none was symmetric about the equator or in toroidal components. The volume integral of the magnetic energy contributed by velocity harmonics with $l = 32$ was 0.0004, of which none was stored in poloidal or equatorially symmetric terms. This is a consequence of the equatorial (anti-)symmetry restriction on the magnetic field.

The l-spectra of kinetic and magnetic energies are a very good way of assessing the adequacy of the numerical resolution.

Every NTSBME time-steps, a breakdown of kinetic and magnetic energy is done as a function of spherical harmonic order, m. The lines output are entirely analogous to the L spectrum. The first character is either v or m, indicating velocity or magnetic field. The second character is either L (as we saw above), to indicate spherical harmonic degree, or M, to indicate wavenumber. The next four numbers are once again time, total energy integral, equatorially anti-symmetric energy integral and toroidal energy integral. The following example output,

```
vM=     0   9.0000000D-05   3.7672825D+01   0.0000000D+00   3.6117884D+01
vM=     4   9.0000000D-05   3.6814597D+02   0.0000000D+00   2.2203324D+02
vM=     8   9.0000000D-05   2.5002256D+01   0.0000000D+00   1.0730790D+01
vM=    12   9.0000000D-05   6.4711928D+00   0.0000000D+00   4.8683469D+00
vM=    16   9.0000000D-05   1.5825000D+00   0.0000000D+00   1.0633268D+00
vM=    20   9.0000000D-05   4.4694531D-01   0.0000000D+00   2.9562849D-01
mM=     0   9.0000000D-05   3.3687315D+01   3.3687315D+01   9.9978641D+00
mM=     4   9.0000000D-05   9.4815344D+00   9.4815344D+00   7.1620613D+00
mM=     8   9.0000000D-05   1.5078272D+00   1.5078272D+00   8.4074930D-01
mM=    12   9.0000000D-05   2.6115742D-01   2.6115742D-01   1.1956764D-01
```

indicates that we have imposed a 4-fold azimuthal symmetry with a maximum spherical harmonic wavenumber $m = 20$. The magnetic energy is dominated by the axisymmetric part whereas the kinetic energy is dominated by the $m = 4$ component. The axisymmetric velocity is almost entirely toroidal whereas the kinetic energy in velocity components with $m = 8$ is approximately half poloidal and half toroidal.

The $m-$spectra of kinetic and magnetic energy suggest that, for this particular solution, our wavenumber truncation is justified as there is a steep decline in the energy as the wavenumber increases.

1.4 Sample runs of cicibcdts2

The directory

```
$LEOPACK_DIR/SAMPLERUNS/CICIBCDTS2
```

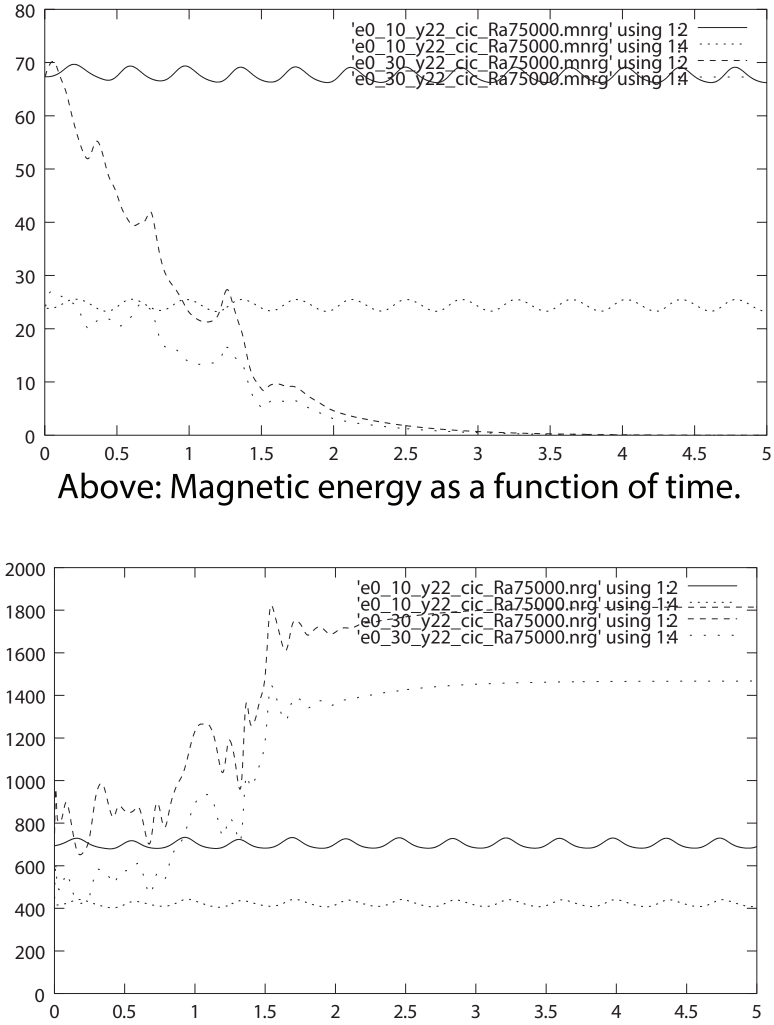
contains example input files, intial conditions and model output. Do not under any circumstances edit these files, as these examples should serve as a control for the correct working of the code. After compiling the program, copy the .input files to another directory, run the code and confirm that the output agrees with that in the directory.

There are two input files,

```
e0_10_y22_cic_Ra75000.input   e0_30_y22_cic_Ra75000.input
```

which perform runs with an inhomogeneous temperature (note NOT heat-flux!!) of 10% and 30% of the mean temperature difference respectively. The dynamo model is essentially the [CAC$^+$01] benchmark model but with a fixed, non-rotating conducting inner core added and a slightly higher Rayleigh number in order to compensate for inner-core diffusion.

Figure (1) shows that for $\varepsilon = 0.1$, the dynamo maintains an approximately oscillating kinetic and magnetic energy. With $\varepsilon = 0.3$, the dynamo dies and the solution becomes non-magnetic. It may be an interesting project to understand why this happens. Is it because the longitudinal propagation of the convection rolls is essential for dynamo action? With the greater strength of the temperature heterogeneity, the convection rolls are unable to propagate past the thermal boundary anomalies, flux therefore being concentrated in very small regions ... Does this suggest that time-dependence is an essential part of the dynamo process? Or, is the temperature heterogeneity simply changing the convection to a form which does not support dynamo action?



Above: Magnetic energy as a function of time.

Above: Kinetic energy as a function of time.

Figure 1: Kinetic and magnetic energies as a function of time for a numerical dynamo, derived from the Case 1 benchmark model, with conducting inner core and a laterally varying temperature at the outer boundary for $\varepsilon = 0.1$ and $\varepsilon = 0.3$.

References

[CAC$^+$01] U. R. Christensen, J. Aubert, P. Cardin, E. Dormy, S. Gibbons, G. A. Glatzmaier, E. Grote, Y. Honkura, C. Jones, M. Kono, M. Matsushima, A. Sakuraba, F. Takahashi, A. Tilgner, J. Wicht, and K. Zhang. A numerical dynamo benchmark. *Phys. Earth Planet. Inter.*, 128:25–34, 2001.