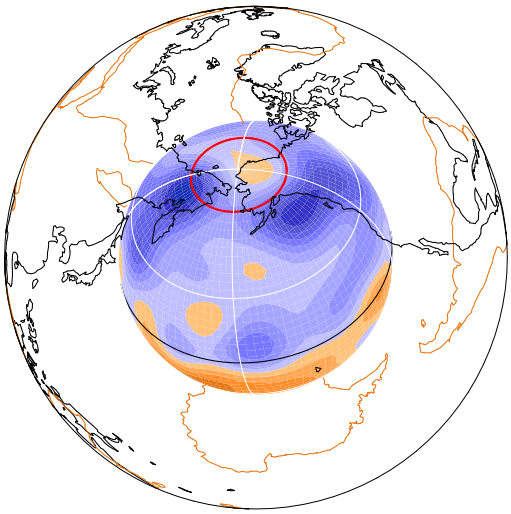
# LEOPACK



## o2ubtctsc2

**U**niform **B**oundary **T**hermal **C**onvection **T**ime-**S**tep **C**ode **2**

Steven J. Gibbons, Oslo

Original document: November $21^{\mathrm{st}}$, 2001. Updated: October $23^{\mathrm{rd}}$, 2022.

# 1 o2ubtctsc2

**U**niform **B**oundary **T**hermal **C**onvection **T**ime-**S**tep **C**ode

This program reads in a solution vector, defined by three files: `stem.ints`, `stem.vecs` and `stem.xarr`, and follows the solution, using fixed $\Delta t$, for a given number (`NTS`) of time-steps, the equations:-

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

and

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

$\Theta$ is the temperature distribution which satisfies homogeneous boundary conditions (with either a zero temperature or a zero heat-flux at the boundaries: inner and outer boundaries need not impose the same boundary condition), $\boldsymbol{u}$ is the velocity and $\boldsymbol{\omega}$ is the vorticity (with $\boldsymbol{\omega} = \nabla \times \boldsymbol{u}$).

The stand-alone source code version of the program is compiled by typing

```
make o2ubtctsc2
```

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

```
o2ubtctsc2 < inputfile
```

The inputs file must have the following format.

---

```
o2ubtctsc2_test                          : filename stem
v36m0.s.4_28.r40.ints                    : name of .ints file
v36m0.s.4_28.r40.vecs                    : name of .vecs file
v36m0.s.4_28.r40.xarr                    : name of .xarr file
   -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   65000.0   1.0 : CE   CF   CG   CH   CI
 0.5     0.0001    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     50      5    25               : IOUTF  NTS NTSBB NTSBS
   6     0.0001                        : ITMX   DTOL
    1                                  : NTSBFE
```

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. Running `o2ubtctsc2` with the above input file will create the files `o2ubtctsc2_test.log` and `o2ubtctsc2_test.eval`, along with all backup and snapshot outputs of the solution.
- `name of .ints file`: name of already existing indices file describing initial solution.
- `name of .vecs file`: name of already existing vector file describing initial solution. Must contain the same number of radial functions as indicated in the `.ints` file.
- `name of .xarr file`: name of already existing radial spacings file describing initial solution. Must contain the same number of radial grid nodes as indicated in the `.vecs` file.
- `NTHP`: number of $\theta$ 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 $\theta$ 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)

- `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, $\Delta t$.

- `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 a "backup solution vector". This is a vector in the same format as the file `v36m0.s.4_28.r40.vecs` which defines the initial solution and always has the filename *root.bv*: so in the example here, every 5 time-steps, the solution will be written out to the file `o2ubtctsc2_test.bv`. The solution snapshot after 10 time-steps will therefore overwrite the snapshot after 5 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. In practice, you would never want to make a backup every 5 time-steps, this would take too long to write to the disk and would be an inefficiency in the program. `.bv` 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 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. To summarise the output which will be made from the above input file, with NTS = 50, NTSBB = 5 and NTSBS = 25, we will have the files o2ubtctsc2_test.ts25.sv (after 25 time-steps), o2ubtctsc2_test.ts50.sv (after 50 time-steps) and o2ubtctsc2_test.bv (after 45 time-steps: this file will have overwritten all of those evaluated at earlier intervals of 5 time-steps).

- ITMX. The maximum number of iterations allowed in the predictor-corrector time-stepping scheme. ITMX is the highest permitted value of $j$.

- 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 and (rather arbitrarily selected) measurements of the velocity and temperature.

## 1.1 Subprograms required for o2ubtctsc2

### SUBS subroutines

```
fopen.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 avbmbr.f vobmbr.f asvcpl.f
svfwt.f pvvcpl.f vecop.f nsvhst.f nsvbta.f rsdv2c.f
sf2vgc.f sf2sdc.f xsvsdc.f rsdv2d.f sf2vgd.f sf2sdd.f
xsvsdd.f nmcxse.f fclose.f shkeer.f fnamer.f ldgnmf.f
gfdcfd.f matop.f amlica.f amsdea.f amdlt.f vobmar.f
nmlica.f asvdr.f dvecz.f fftrlv.f svrint.f matind.f
bmrcop.f powtwo.f
```

### SUBS double precision function

```
rtpfce.f emmult.f pmm.f pmm1.f plm.f dpmm.f
dpmm1.f dplm.f shmplg.f shdplg.f sqrll1.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          NRMAX, NH1MAX, NH2MAX, NH3MAX, NHMAX,
1                 NIVMAX, NDCS, NCFM, NDRVM, NBNM, NIV1MX,
2                 NIV2MX, NIV3MX, NCMXX, NNDM
 PARAMETER       ( NRMAX = 50, NH1MAX =  520, NH2MAX =  520,
1                  NH3MAX =  520, NCMXX = 13, NNDM = 4,
2                  NHMAX = NH1MAX+NH2MAX+NH3MAX,
3                  NIVMAX = NHMAX*NRMAX )
 PARAMETER       ( NIV1MX = NH1MAX*NRMAX,
1                  NIV2MX = NH2MAX*NRMAX,
2                  NIV3MX = NH3MAX*NRMAX )
 PARAMETER       ( NDCS = 3, NBNM = 3, NCFM = 2*NBNM+1,
1                  NDRVM = 4 )
 INTEGER          LHMAX, NTHMAX, NPHMAX, NPMAX
 PARAMETER       ( LHMAX =  36, NTHMAX = 100, NPHMAX =  64,
1                  NPMAX = (LHMAX+1)*(LHMAX+2)/2 )
```

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

- `NRMAX` is the maximum permitted number of radial grid nodes.
- `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.
- `NNDM` is the number of nodes used for interpolating radial functions for evaluating components using **RTPFCE**.
- `NBNM` is the number of upper (and lower) diagonals in the banded matrix. `NBNM = 3` is recommended value.
- `LHMAX` is the highest permitted spherical harmonic degree, $l$.
- `NTHMAX` is the maximum permitted number of grid nodes in $\theta$ for Gaussian quadrature in the spherical transforms.
- `NPHMAX` is the maximum permitted number of grid nodes in $\phi$ for the Fast Fourier Transforms.

## 1.3 Outputs from O2UBTCTSC2

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 two files output: `root.eval` and `root.log`.

`root.eval` has a line added to it every `NTSBFE` time-steps. This line contains, in the format `(6(1PD16.7))` the six numbers `DTIME`, `PVKE`, `TVKE`, `VRAD`, `VPHI` and `TEMP`.

`DTIME` is the time elapsed, `PVKE` is the volume integral of the poloidal kinetic energy, `TVKE` is the volume integral of the toroidal kinetic energy, `VRAD`, `VPHI` and `TEMP` are the values of $v_r$ $v_\phi$ and $\Theta$ evaluated at $r = (r_\mathrm{i} + r_\mathrm{o})/2$, $\theta = \pi/2$ 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` essentially just reports back on the physical size of the problem, the allocation of finite difference schemes and echoes back the physical parameters read in.

## 1.4 Sample runs of o2ubtctsc2

The directory

`$LEOPACK_DIR/SAMPLERUNS/O2UBTCTSC2`

contains example input files, initial solutions 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`, `.ints`, `.vecs` and `.xarr` files to another directory, run the code and confirm that the output agrees with that in the directory.

### 1.4.1 Example a

A spherical fluid shell rotates with a constant angular velocity, $\Omega$, and has inner and outer radii $r_\mathrm{i}$ and $r_\mathrm{o}$ respectively. Gravity within the shell is given by

$$\boldsymbol{g} = -\gamma \boldsymbol{r}, \tag{3}$$

where $\gamma$ is a constant. The thermal expansivity, $\alpha$, thermal diffusivity, $\kappa$, and viscosity, $\nu$, are all constant throughout the shell. A basic state temperature, $T_s$, is maintained by a uniform distribution of heat sources such that

$$\nabla T_s = -\beta \boldsymbol{r}, \tag{4}$$

for constant $\beta$ (see, for example, [Rob68], [Bus70]).

With length scaled by $d = r_\mathrm{o} - r_\mathrm{i}$, time by $d^2/\kappa$ (the thermal diffusion time) and temperature by $d^2\beta$ respectively allows our heat and momentum equations to be written in the dimensionless forms

$$P_r^{-1}\left(\frac{\partial}{\partial t} + \boldsymbol{u}.\nabla\right)\boldsymbol{u} + E^{-1}\boldsymbol{k}\times\boldsymbol{u} = -\nabla\tilde{p} + RT\boldsymbol{r} + \nabla^2\boldsymbol{u} \tag{5}$$

and

$$\frac{\partial T}{\partial t} + \boldsymbol{u}.\nabla T = \nabla^2 T + \boldsymbol{u}.\boldsymbol{r}. \tag{6}$$

The problem is now controlled by the Prandtl number, $P_r$, the Rayleigh number, $R$, and the Ekman number, $E$, given respectively by

$$P_r = \frac{\nu}{\kappa},\ R = \frac{\alpha\beta\gamma d^6}{\nu\kappa},\ \text{and}\ E = \frac{\nu}{2\Omega d^2}. \tag{7}$$

The aspect ratio $\eta = r_\mathrm{i}/r_\mathrm{o}$ is set to 0.4.

The run we demonstrate is for $P_r = 6.0$, $E = 2\times10^{-4}$, $R = 140000.0$. Comparing Equation (5) with Equation (2), and (6) with (1), reveals the coefficients as found in the input file below.

```
* input file for o2ubtctsc2
example_aOUTPUT                : filename stem
example_a.ints            : integers
example_a.vecs            : vector
example_a.xarr            : radial spacings
   -2             : NTHP (-1 --> LH+2, -2 --> 3*LH/2 )
 1.0    1.0   0.0           1.0   1.0          : CA  CB1  CB2   CC   CD
 0.16666666 0.16666666 5000.0  140000.0  1.0   : CE   CF   CG   CH   CI
 0.4     0.000025  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    10      80  25000               : IOUTF  NTS NTSBB NTSBS
  10     1.0                           : ITMX   DTOL
  1                                    : NTSBSE
```

The time-step is set to 0.000025 (set somewhat by trial and error) and it was found necessary to make the time integration rather more implicit than explicit ($c = 0.4$). A total of 10 time-steps is to be performed, with a kinetic energy evaluation at each. A single snapshot will be produced at the end of the run.

There are consequently three output files produced; `example_aOUTPUT.ts10.sv`, `example_aOUTPUT.eval` and `example_aOUTPUT.log`.

An inspection of the `.ints` file reveals that there are `1409` spherical harmonic radial functions in total. All wavenumbers up to the maximum of 20 are featured (i.e. we are not imposing an azimuthal symmetry), although only harmonics with equatorial symmetry are included. All poloidal velocity harmonics have `IIBC` and `IOBC` set to 5 - and therefore have stress-free boundaries. This is confirmed by observing that the toroidal velocity harmonics have `IIBC` and `IOBC` set to 6 (c.f. tables 1 and 2). All temperature harmonics have `IIBC = 2` and `IOBC = 3`: i.e. fixed temperature at the inner boundary and fixed heat-flux at the outer. These are the only combinations of boundary conditions in the solution (as is to be expected for a non-magnetic convection solution) and this is confirmed by the output in the file `example_aOUTPUT.log`:

```
 Number of difference schemes =   3

Scheme(   1) IBC =   5 OBC =   5
Scheme(   2) IBC =   6 OBC =   6
Scheme(   3) IBC =   2 OBC =   3
```

The `.xarr` file confirms that there are 50 non-equally distributed grid nodes between $r_\mathrm{i} = 2/3$ and $r_\mathrm{o} = 5/3$, confirming our intended aspect ratio.

### 1.4.2 Example b

This is a solution to the Case 0 non-magnetic solution of the benchmark study, [CAC+01]. Details of the scaling and parameter regime are found in this publication.

| IIBC | Inner boundary condition. |
|---|---|
| 1 | None imposed |
| 2 | Function must vanish at boundary. |
| 3 | First derivative must vanish at boundary. |
| 4 | Both function and first derivative must vanish. |
| 5 | Both function and second derivative must vanish. |
| 6 | $rdf/dr - f(r) = 0$ |
| 7 | $rdf/dr - lf(r) = 0$ with $l =$L. |

Table 1: Current options for setting boundary conditions at the inner boundary: values of **IIBC**.

| IOBC | Outer boundary condition. |
|---|---|
| 1 | None imposed |
| 2 | Function must vanish at boundary. |
| 3 | First derivative must vanish at boundary. |
| 4 | Both function and first derivative must vanish. |
| 5 | Both function and second derivative must vanish. |
| 6 | $rdf/dr - f(r) = 0$ |
| 7 | $rdf/dr + (l+1)f(r) = 0$ with $l =$L. |

Table 2: Current options for setting boundary conditions at the outer boundary: values of **IOBC**.

Inspection of the output file `example_bOUTPUT.eval` reveals that, to within the precision of the output, the kinetic energy of the solution is not changing.

The poloidal kinetic energy is given as 193.22058 and the toroidal kinetic energy as 655.73037. The total k.e. is therefore 848.95095. The volume of the spherical shell is given by

$$\begin{aligned} V_s &= \frac{4\pi(r_{\rm o}^3 - r_{\rm i}^3)}{3} \\ &= \frac{4\pi(20^3 - 7^3)}{3 \times 13^3} \\ &= 14.598801 \end{aligned} \tag{8}$$

The k.e. normalised by the volume of the spherical shell is therefore $848.95095/V_s = 58.152$.

# References

[Bus70] F. H. Busse. Thermal instabilities in rapidly rotating systems. *J. Fluid Mech.*, 44:441–460, 1970.

[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.

[Rob68] P. H. Roberts. On the thermal instability of a rotating fluid sphere containing heat sources. *Phil. Trans. R. Soc. Lond.*, A263:93–117, 1968.