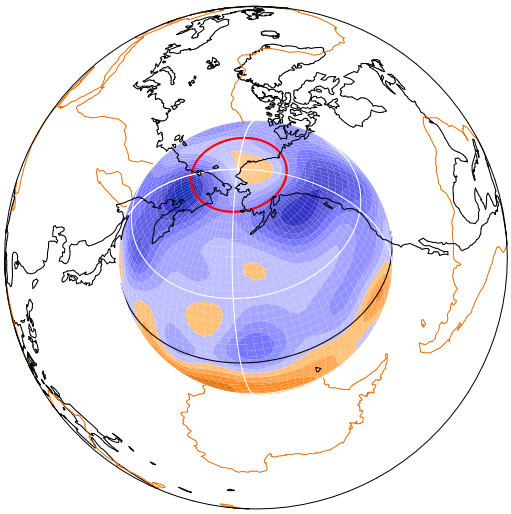
LEOPACK



blscnlsc

Boundary **L**ocked **S**teady **C**onvection **N**on **L**inear **S**olutions **C**alculate

Steven J. Gibbons, Oslo

Original document: November 21st, 2001. Updated: October 15th, 2022.

1 blscnlsc

Boundary **L**ocked **S**teady **C**onvection **N**on **L**inear **S**olutions **C**alculate

The equation defining the advection of heat is (see [GR87])

$$\frac{\partial T}{\partial t} + \boldsymbol{u}.\nabla T = \kappa \nabla^2 T + \frac{q}{C_p \rho} \tag{1}$$

where $\boldsymbol{u}$ is the fluid flow, T the temperature, κ the thermal diffusivity ($\mathrm{m^2 s^{-1}}$), q the rate of local heating ($\mathrm{Jm^{-3}s^{-1}}$), C_p the specific heat capacity ($\mathrm{Jkg^{-1}K^{-1}}$) and ρ the density ($\mathrm{kgm^{-3}}$). The convection codes assume that the temperature, T, is expressed as follows:-

$$T(r,t,\theta,\phi) = T_0(r) + T_1(r,t,\theta,\phi) \tag{2}$$

The steady, basic state temperature distribution, $T_0(r)$, is given the form

$$T_0(r) = -\frac{1}{2} b_1 r^2 + \frac{b_2}{r} + b_3 \tag{3}$$

where b_1, b_2 and b_3 are constants. Its purpose is to define the temperature profile for the sphere or spherical shell, incorporating any internal heating sources. It satisfies

$$\nabla T_0 = -\left(b_1 r + b_2 r^{-2}\right) \boldsymbol{e}_r, \tag{4}$$

where $\boldsymbol{e}_r$ is the unit vector in the radial direction, and

$$\nabla^2 T_0 = -3 b_1. \tag{5}$$

If we substitute the definition (2) into Equation (1) and apply (4) and (5) we derive

$$\frac{\partial T_1}{\partial t} = \kappa \nabla^2 T_1 - 3\kappa b_1 + \frac{q}{C_p \rho} + -\boldsymbol{u}.\nabla T_1 \tag{6}$$

It is now clear that the constant b_1 defines the sources of internal heating with

$$b_1 = \frac{q}{3 C_p \rho \kappa}. \tag{7}$$

If there are no internal heating sources, then $q = 0$ and hence $b_1 = 0$. The constant b_2 is chosen appropriately for systems which have a simple temperature gradient from the inner to the outer boundary.

For numerical simplicity, it is best to solve for temperature functions with homogeneous boundary conditions. We therefore decompose T_1, the perturbation from the basic state temperature,

$$T_1(r,t,\theta,\phi) = \Theta(r,t,\theta,\phi) + \varepsilon T_{\mathrm{a}}(r,\theta,\phi). \tag{8}$$

Θ is the function which is solved for in all of the calculations. $T_{\rm a}$ is an additional temperature which is imposed if, for example, an inhomogeneous heat-flux at the outer boundary is required.

If we denote the radial component of the velocity u_r, then applying Equation (8) to Equation (6) gives us the heat equation as applied in all of the programs:

$$c_a \frac{\partial \Theta}{\partial t} = c_d \nabla^2 (\Theta + \varepsilon T_{\rm a}) + b_1 u_r r + b_2 \frac{u_r}{r^2} - c_c \boldsymbol{u}.\nabla(\Theta + \varepsilon T_{\rm a}) \tag{9}$$

The constants c_a, b_1, b_2, c_c and c_d are arbitrarily named, with no physical meaning attatched to them. Their use simply allows for any scaling to be applied to the equations. In the codes, c_a is stored in the double precision variable `CA`; and similarly with b_1 (`CB1`), b_2 (`CB2`), c_c (`CC`) and c_d (`CD`).

In the Boussinesq approximation, all density variations except those with respect to the buoyancy force are considered to be negligible, and following the analysis of [GR87], the momentum equation is written

$$\frac{\partial \boldsymbol{u}}{\partial t} + \boldsymbol{u}.\nabla \boldsymbol{u} + 2\boldsymbol{\Omega} \times \boldsymbol{u} = -\nabla \tilde{\omega} + \frac{\delta \rho}{\rho_0} \boldsymbol{g} + \frac{\boldsymbol{J} \times \boldsymbol{B}}{\rho_0} + \nu \nabla^2 \boldsymbol{u}, \tag{10}$$

where $\boldsymbol{J}$ and $\boldsymbol{B}$ are respectively the electric current and magnetic field. The scalar function $\tilde{\omega}$ combines the pressure, p, and the centrifugal force such that

$$\tilde{\omega} = \frac{p}{\rho} - \frac{1}{2} |\boldsymbol{\Omega} \times \boldsymbol{r}|^2,$$

and can be removed from the problem by taking the curl of Equation (10). The density variation $\delta\rho$ is expressed in terms of the thermal expansivity, α (K^{-1}), and T, the temperature perturbation from a well mixed state ($\rho = \rho_0$), to give

$$\frac{\delta \rho}{\rho_0} = -\alpha T. \tag{11}$$

The acceleration due to gravity, $\boldsymbol{g}$ is written in terms of the radial vector $\boldsymbol{r}$ as

$$\boldsymbol{g} = -\gamma \boldsymbol{r}, \tag{12}$$

for a constant γ, (s^{-2}). The linear dependence of $\boldsymbol{g}$ on r is a good approximation for the core (see for example [DA81] or [And89]), but would not be appropriate for the mantle. ν is the viscosity ($\mathrm{m}^2\mathrm{s}^{-1}$) and $\boldsymbol{\Omega} = \Omega \boldsymbol{k}$ is the rotation vector, in terms of the unit vector, $\boldsymbol{k}$. The electric current is related to the magnetic field by

$$\nabla \times \boldsymbol{B} = \mu \boldsymbol{J} \tag{13}$$

where μ is the magnetic permeability and assumed to equal μ_0, the magnetic permeability of free space everywhere.

In order to eliminate the pressure gradient from the momentum equation, we take the curl of Equation (10) and apply equations (11) and (12). If we denote the vorticity (the curl of $\boldsymbol{u}$) by $\boldsymbol{\omega}$, then our vorticity equation becomes

$$\begin{aligned}\frac{\partial \boldsymbol{\omega}}{\partial t} &= -\nabla \times (\boldsymbol{u}.\nabla \boldsymbol{u}) - 2\Omega \nabla \times (\boldsymbol{k} \times \boldsymbol{u}) \\ &\quad +\alpha\gamma \nabla \times (T\boldsymbol{r}) + \frac{1}{\rho\mu_0}\nabla \times [(\nabla \times \boldsymbol{B}) \times \boldsymbol{B}] + \nu\nabla^2\boldsymbol{\omega}.\end{aligned} \tag{14}$$

It is assumed here that the kinematic viscosity, ν, is not a function of space. The basic state temperature, T_0, is a function of radius alone and therefore cannot contribute to the buoyancy term in the vorticity equation. Giving arbitrarily defined names to the scalings which multiply the terms in our equation (14), we write the curl of the momentum equation

$$\begin{aligned}c_e\frac{\partial \boldsymbol{\omega}}{\partial t} &= -c_f \nabla \times (\boldsymbol{u}.\nabla \boldsymbol{u}) - c_g \nabla \times (\boldsymbol{k} \times \boldsymbol{u}) \\ &\quad +c_h \nabla \times [(\Theta + \varepsilon T_{\rm a})\boldsymbol{r}] + c_j \nabla \times [(\nabla \times \boldsymbol{B}) \times \boldsymbol{B}] + c_i\nabla^2\boldsymbol{\omega}.\end{aligned} \tag{15}$$

There are two vector quantities in the momentum equation, the velocity $\boldsymbol{u}$ and the magnetic field $\boldsymbol{B}$. $\boldsymbol{B}$ must always satisfy the solenoidal condition

$$\nabla.\boldsymbol{B} = 0 \tag{16}$$

and similarly, for a Boussinesq fluid, $\boldsymbol{u}$ must satisfy

$$\nabla.\boldsymbol{u} = 0. \tag{17}$$

We can therefore express both velocity and magnetic field in poloidal/toroidal decompositions

$$\boldsymbol{B} = \nabla \times \nabla \times \left[\quad {}^{P}B(r,t,\theta,\phi)\ \boldsymbol{r} \quad \right] \quad + \quad \nabla \times \left[\quad {}^{T}B(r,t,\theta,\phi)\ \boldsymbol{r} \quad \right] \tag{18}$$

and

$$\boldsymbol{u} = \nabla \times \nabla \times \left[\quad {}^{P}v(r,t,\theta,\phi)\ \boldsymbol{r} \quad \right] \quad + \quad \nabla \times \left[\quad {}^{T}v(r,t,\theta,\phi)\ \boldsymbol{r} \quad \right]. \tag{19}$$

Note that these definitions are different from those of, for example Bullard and Gellman - [BG54], who use the unit radial vector, $\hat{\boldsymbol{r}}$, instead of $\boldsymbol{r}$.

In this proram, we completely ignore the magnetic field, $\boldsymbol{B}$.

The temperature function T_1, the perturbation from the basic state temperature (see Equation 6), is suject to the decomposition

$$T_1(r,t,\theta,\phi) = \Theta(r,t,\theta,\phi) + \varepsilon T_{\rm a}(r,\theta,\phi). \tag{20}$$

where $T_{\rm a}$ is an additional temperature imposed in order to apply an inhomogeneous heat-flux at the boundary: either inner, outer or both. Our heat and momentum equations are

$$c_a\frac{\partial \Theta}{\partial t} = c_d\nabla^2(\Theta + \varepsilon T_{\rm a}) + b_1 u_r r + b_2\frac{u_r}{r^2} - c_c\boldsymbol{u}.\nabla(\Theta + \varepsilon T_{\rm a}) \tag{21}$$

and

$$c_e\frac{\partial\boldsymbol{\omega}}{\partial t} = -c_f\nabla\times(\boldsymbol{u}.\nabla\boldsymbol{u}) - c_g\nabla\times(\boldsymbol{k}\times\boldsymbol{u}) + c_h\nabla\times[(\Theta+\varepsilon T_{\rm a})\boldsymbol{r}] + c_i\nabla^2\boldsymbol{\omega}. \tag{22}$$

We assume that the laterally varying heat-flux, $T_{\rm a}$, locks the flow (i.e. $\partial\Theta/\partial t = 0$ and $\partial\boldsymbol{u}/\partial t = 0$). `blscnlsc` then solves for the steady solution, $(\boldsymbol{u}_0, \Theta_0)$, to the equations

$$0 = c_d\nabla^2(\Theta+\varepsilon T_{\rm a}) + b_1 u_r r + b_2\frac{u_r}{r^2} - c_c\boldsymbol{u}.\nabla(\Theta+\varepsilon T_{\rm a}) \tag{23}$$

and

$$0 = -c_f\nabla\times(\boldsymbol{u}.\nabla\boldsymbol{u}) - c_g\nabla\times(\boldsymbol{k}\times\boldsymbol{u}) + c_h\nabla\times[(\Theta+\varepsilon T_{\rm a})\boldsymbol{r}] + c_i\nabla^2\boldsymbol{\omega}. \tag{24}$$

This solution is done with Newton-Raphson iteration.

Note that `blscnlsc` does NOT test these solutions for stability! If you require a stability calculation in addition, then you must use either the program `blscnlsic` or the program `blscnlsic_evecs`.

$T_{\rm a}$ has an essentially arbitrary form apart from the conditions that it must satisfy either

$$T_{\rm a}(r_{\rm i},\theta,\phi) = g_{\rm i}(\theta,\phi) \tag{25}$$

or

$$\left.\frac{\partial T_{\rm a}(r,\theta,\phi)}{\partial r}\right|_{r=r_{\rm i}} = g_{\rm i}(\theta,\phi) \tag{26}$$

at the inner boundary, and either

$$T_{\rm a}(r_{\rm o},\theta,\phi) = g_{\rm o}(\theta,\phi) \tag{27}$$

or

$$\left.\frac{\partial T_{\rm a}(r,\theta,\phi)}{\partial r}\right|_{r=r_{\rm o}} = g_{\rm o}(\theta,\phi) \tag{28}$$

at the outer boundary: depending upon whether a constant temperature or constant heat-flux boundary condition is to be satisfied.

The functions $g_{\rm i}$ and $g_{\rm o}$ are both expanded as series of spherical harmonics:

$$g_{\rm i}(\theta,\phi) = \frac{\sqrt{\varepsilon_{\rm i}}}{N_{\rm i}}\sum_{l=0}^{l=L}\sum_{m=0}^{m=l}\left[g_{{\rm i},l}^{mc}\cos m\phi + g_{{\rm i},l}^{ms}\sin m\phi\right]P_l^m(\cos\theta) \tag{29}$$

and

$$g_{\rm o}(\theta,\phi) = \frac{\sqrt{\varepsilon_{\rm o}}}{N_{\rm o}} \sum_{l=0}^{l=L} \sum_{m=0}^{m=l} \left[g_{{\rm o},l}^{mc} \cos m\phi + g_{{\rm o},l}^{ms} \sin m\phi \right] P_l^m(\cos\theta), \tag{30}$$

where the associated Legendre function $P_l^m(\cos\theta)$ satisfies the Schmidt quasi-normalisation condition

$$\int_0^\pi \left[P_l^m(\cos\theta)\right]^2 \sin\theta d\theta = \frac{2(2-\delta_{m0})}{2l+1}. \tag{31}$$

The user specifies the non-zero coefficients $g_{{\rm i},l}^{mc}$, $g_{{\rm i},l}^{ms}$, $g_{{\rm o},l}^{mc}$ and $g_{{\rm o},l}^{ms}$, and **blscnlsc** calculates normalisation factors, $N_{\rm i}$ and $N_{\rm o}$, such that

$$\int_0^{2\pi} \int_0^\pi \left[g_{\rm i}(\theta,\phi)\right]^2 \sin\theta d\theta d\phi = \varepsilon_{\rm i} \tag{32}$$

and

$$\int_0^{2\pi} \int_0^\pi \left[g_{\rm o}(\theta,\phi)\right]^2 \sin\theta d\theta d\phi = \varepsilon_{\rm o}. \tag{33}$$

Therefore, if a user specified that the only non-zero coefficients in the series (29) and (30) were to be $g_{{\rm o},2}^{2c} = 3.0$ and $g_{{\rm o},3}^{2s} = -3.0$ with $\varepsilon_{\rm o} = 1.0$, the result would be exactly the same had the user specified the coefficients $g_{{\rm o},2}^{2c} = 0.012$ and $g_{{\rm o},3}^{2s} = -0.012$ with $\varepsilon_{\rm o} = 1.0$. This gives the user a global control on the normalisation of the "strength" of the functions $g_{\rm i}$ and $g_{\rm o}$ without having to worry about the actually coefficients themselves, other than their relative sizes.

A file specifies all the coefficients $g_{{\rm i},l}^{mc}$, $g_{{\rm i},l}^{ms}$, $g_{{\rm o},l}^{mc}$ and $g_{{\rm o},l}^{ms}$. Coefficients not listed are assumed to be zero.

This file contains one line for each coefficient. The first two characters of each line must contain either `IB` or `OB`: which specify either inner or outer boundary. These two characters must be exactly at the beginning of a line. The remaining part of the line contains, in any format, the numbers `L`, `M`, `ICS` and `COEF`.

`L` and `M` are clearly l and m: `ICS` is `1` to indicate a $\cos m\phi$ harmonic (i.e. $g_{{\rm i},l}^{mc}$ or $g_{{\rm o},l}^{mc}$) and `2` to indicate a $\sin m\phi$ harmonic (i.e. $g_{{\rm i},l}^{ms}$ or $g_{{\rm o},l}^{ms}$). `COEF` is ofcourse just the value of the coefficient.

Examples

The file

```
OB   2   2   1       1.0
```

sets all coefficients to zero except for $g_{{\rm o},2}^{2c}$ which is set to unity.

The file

```
IB   2   2   1       1.0
OB   2   2   1       1.0
OB   3   2   2       1.0
```

sets $g^{2c}_{\mathrm{i},2} = 1.0$, $g^{2c}_{\mathrm{o},2} = 1.0$ and $g^{2s}_{\mathrm{o},3} = 1.0$. This fixes the contribution from $g^{2c}_{\mathrm{o},2}$ relative to that from $g^{2s}_{\mathrm{o},3}$, whereas the contribution from $g^{2c}_{\mathrm{i},2}$ is only fixed relative to the others by the parameters ε_{i} and ε_{o}.

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

```
make sablscnlsc
```

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

```
blscnlsc < inputfile
```

depending upon the executable.

The inputs file must have the following format.

```
* Input file for blscnlsc
*
boundary_coeffs                       : INHOMOG TEMP. FILE
example_aOUTPUT                       : ROOT
0.666666666   1.666666666  2   2   45  : RI, RO, IVELBC, ITHEBC, LU
  12   0.001                          : MXATT CTOL
*-------------------------------------------------------------------------
*NR  ISP  LH  SYM  MLOW   MINC  MMAX  IOF   CC   CB1   CB2
*          CD       CF     CG         CH       CI   EPS_in  EPS_out
*-------------------------------------------------------------------------
 40   2   14   1    0      2    10      1   1.0  -1.0   0.0
          1.0     0.01   1.0         1.0      0.1   0.0      1.00
 40   2   14   1    0      2    10      1   1.0  -1.0   0.0
          1.0    0.001   1.0         1.0     0.01   0.0      1.00
 40   2   14   1    0      2    10      1   1.0  -1.0   0.0
          1.0   0.0001   1.0         1.0    0.001   0.0      1.00
 40   2   14   1    0      2    10      1   1.0  -1.0   0.0
          1.0  0.00001   1.0         1.0   0.0001   0.0      1.00
*-------------------------------------------------------------------------
```

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 following arguments are common to all of the runs carried out by the execution:

- `INHOMOG TEMP. FILE`: Name of file containing the coefficients for boundary temperature as described above.
- `ROOT`: Stem of filename for all output files.
- `RI`. Inner boundary radius.
- `RO`. Outer boundary radius.

- `IVELBC`: Boundary condition for the velocity.
  `IVELBC = 1` $\rightarrow$ rigid boundaries.
  `IVELBC = 2` $\rightarrow$ stress-free boundaries.

- `ITHEBC`: Boundary condition for the temperature.
  `ITHEBC = 1` $\rightarrow$ fixed temperature at both $r = r_{\mathrm{i}}$ and $r = r_{\mathrm{o}}$.
  `ITHEBC = 2` $\rightarrow$ fixed temperature at $r = r_{\mathrm{i}}$ and fixed heat-flux at $r = r_{\mathrm{o}}$.
  `ITHEBC = 3` $\rightarrow$ fixed heat-flux at $r = r_{\mathrm{i}}$ and fixed temperature at $r = r_{\mathrm{o}}$.

- `LU`: Output flag. Set to zero to suppress growth rate information and set to `45` to output growth rate information.

- `MXATT`: The number of attempts allowed for convergence of the Newton-Raphson iteration towards $(\boldsymbol{u}_0, \Theta_0)$.

- `CTOL`: Stopping criterion for Newton-Raphson iteration. The difference between the residual-norms for two consecutive iterations must be less than `CTOL`. (Note that the residual norm itself will not actually go to zero in the program: this is due to the treatment of the boundaries.)

The following (uncommented) lines come in pairs, with two consecutive lines giving the parameters for a single run. The first one of each of these pairs must contain the inputs

- `NR`. Number of radial grid nodes.

- `ISP`. Radial grid node spacings flag.
  `isp = 1` forces evenly spaced grid nodes from `ESNAAS` and `isp = 2` forces Chebyshev zero spaced nodes from `ZCPAAS`.

- `LH`. Highest spherical harmonic degree, l requested.

- `SYM`. Equatorial symmetry flag.
  `SYM = 1` $\rightarrow$ equatorially symmetric modes.
  `SYM = 2` $\rightarrow$ equatorially anti-symmetric modes.
  `SYM = 3` $\rightarrow$ both symmetries.

- `MLOW`: Lowest wavenumber required in the solution. This is (almost?) invariably 0.

- `MINC`: The smallest non-zero wavenumber required.

- `MMAX`: The largest wavenumber allowed. Wavenumbers will then include all multiples of `MINC` up to `MMAX`.

- `IOF`: File output flag.
  `IOF = 0` → no file output.
  `IOF = 1` → standard output for homogeneous part of locked solution only. (This is recommended.)
  `IOF = 2` → radial function output.
  `IOF = 3` → standard output for total locked solution (i.e. including inhomogeneous boundaries). This is useful for easy display of the solution but not very helpful if you wish to use the solution for time-stepping etc. as such solutions should be homogeneous.
  `IOF = 4` → standard output for both homogeneous part and total locked solution.

The second of these lines contains the numbers

- `CD`: Scaling parameter c_d in Equation (23).
- `CF`: Scaling parameter c_f in Equation (24).
- `CG`: Scaling parameter c_g in Equation (24).
- `CH`: Scaling parameter c_h in Equation (24).
- `CI`: Scaling parameter c_i in Equation (24).
- `EPS_in`: Strength of heating heterogeneity at inner boundary, ε_{i}, as defined in Equation (32).
- `EPS_out`: Strength of heating heterogeneity at outer boundary, ε_{o}, as defined in Equation (33).

1.1 Subprograms required for blscnlsc

SUBS subroutines

```
vecop.f fopen.f fclose.f esnaas.f zcpaas.f ontppf.f
gauwts.f schnla.f vthmsr.f cindsw.f svfdcf.f fdcmbd.f
shcanc.f ithcar.f blcnrs.f hmfwt.f svfwt.f xarrwt.f
itfa.f  svprnt.f fnamer.f ldgnmf.f gfdcfd.f lmfind.f
itfcf.f cntric.f vspcc.f vccpcc.f casvdr.f ibtdva.f
ssvlc.f ssvlp.f ssvhst.f ssvta.f sdrqst.f sdvgta.f
rqstcf.f rqstcp.f rqstca.f rqstsv.f avmlta.f rv0cva.f
rvcv0a.f rv0gta.f rvgi0a.f amsdea.f nrcwmf.f bmwdfs.f
asvcpl.f radvlf.f matop.f shvect.f vfdp.f  forsst.f
vfcp.f  vf2qst.f asvdr.f rqstvf.f grinvt.f vfcor.f
vfrqst.f rqstdr.f amlp.f  amlc.f  amccfa.f amta.f
```

```
amcl.f  amhst.f invcvt.f vicexr.f innlca.f invgtt.f
invgit.f bmrcop.f cubeop.f fftrlv.f dvecz.f amdlt.f
amlica.f amccft.f corcof.f amhsar.f matind.f powtwo.f
```

SUBS double precision function

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

SUBS integer function

```
indfun.f indshc.f
```

BLAS double precision function

```
dnrm2.f
```

BLAS integer function

```
idamax.f
```

BLAS subroutines

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

LAPACK subroutines

```
dgetrf.f dgetri.f dgbtrf.f dgbtrs.f dgetf2.f dlaswp.f
xerbla.f dtrtri.f dgbtf2.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, LHMAX, NHMAX, NPHMAX, NTHMAX, KLMAX,
1        NBNDMX, NDCS, NDRVM, ISVMAX, NPMAX, LHLH2M, NCFM,
2        NBN, NRUNM, NITHMX
 PARAMETER ( NRMAX = 40, LHMAX = 62, NHMAX = 162, NBN = 3,
1            NTHMAX = 64, NPHMAX = 128, KLMAX = (NBN+1)*NHMAX-1,
2            NBNDMX = 3*KLMAX+1, NDCS = 4, NDRVM = 4,
3            ISVMAX = NRMAX*NHMAX, NITHMX = 100 )
 PARAMETER ( NPMAX = (LHMAX+1)*(LHMAX+2)/2,
1            LHLH2M = LHMAX*(LHMAX+2), NCFM = 2*NBN + 1,
2            NRUNM = 100 )
```

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

- `NRMAX` is the maximum permitted number of radial grid nodes.
- `LHMAX` is the highest permitted spherical harmonic degree, l.
- `NHMAX` is the maximum permitted number of spherical harmonic radial functions.
- `NBNM` is the number of upper (and lower) diagonals in the banded matrix. `NBNM = 3` is recommended value.
- `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.
- `NRUNM` is the maximum number of independent runs permitted.

IMPORTANT

In addition to the limitations in the top of the program source code file, the program also calls the subroutine `BLCNRS` which also contains a series of parameter declarations. This may return a message of the form

```
Subroutine BLCNRS
NR   =   40 NRMAX  =   200
NH   =   162 NHMAX  =   146
LH   =   14 LHMAX  =   62
N2   =   6480 ISVMAX =   29200
NPHP =   32 NPHMAX =   128
NTHP =   16 NTHMAX =   64
Recompile routine with higher dimensions.
Program aborted.
```

This immediately tells us that the value `NH` is insufficient: change that as necessary, recompile and try again!

1.3 Outputs from BLSCNLSC

If the filename stem "root" was specified in the input file, the files `root.bc`, `root.cb1`, `root.cb2`, `root.cc`, `root.cd`, `root.cf`, `root.cg`, `root.ch`, `root.ci`, `root.epsi`, `root.epso`, `root.isp`, `root.isym`, `root.lh`, `root.log`, `root.m`, `root.nr`, `root.res`, `root.ri` and `root.ro` will all be created. These files just contain the values of various parameters for each run. My advice is to comment out the lines which write these files.

The only files that really matter for this program are the `.ints`, `.vecs` and `.xarr` files which describe the solutions formed.

1.4 Sample runs of blscnlsc

The directory

`$LEOPACK_DIR/SAMPLERUNS/BLSCNLSC`

contains example input files 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.

1.4.1 Example a

We seek to *almost* reproduce the results of [GG00] - almost because those solutions were at infinite Prandtl number and those calculated here are at Prandtl number $P_r = 10$. (Infinite Prandtl number can be achieved by simply setting the parameter `CF` in the input file - `example_a.input` - to zero. `CF` must be set to the Ekman number divided by the Prandtl number as given in Equation (11) of [GG00].

References

[And89] D. L. Anderson. *Theory of the Earth*. Blackwell Scientific Press, Oxford., 1989.

[BG54] E. C. Bullard and H. Gellman. Homogeneous dynamos and terrestrial magnetism. *Phil. Trans. R. Soc. Lond. A*, 247:213–278, 1954.

[DA81] A. M. Dziewonski and D. L. Anderson. Preliminary reference earth model. *Phys. Earth Planet. Inter.*, 25:297–356, 1981.

[GG00] S. J. Gibbons and D. Gubbins. Convection in the earth's core driven by lateral variations in the core-mantle boundary heat flux. *Geophys. J. Int.*, 143:631–642, 2000.

[GR87] D. Gubbins and P. H. Roberts. Magnetohydrodynamics of the earth's core. In J. A. Jacobs, editor, *Geomagnetism Volume* II, pages 1–183. Academic Press, 1987.