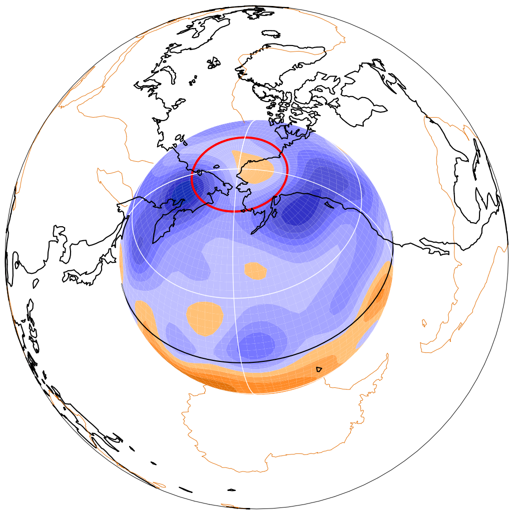
LEOPACK



blscnlsic_evecs

Boundary **L**ocked **S**teady **C**onvection **N**on **L**inear
Solutions and **I**nstability **C**alculate with **E**igen**VEC**tor**S**

Steven J. Gibbons, Oslo

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

1 blscnlsic_evecs

Boundary **L**ocked **S**teady **C**onvection **N**on **L**inear **S**olutions and **I**nstability **C**alculate with **E**igen**VEC**tor**S**

This program performs exactly the same task as blscnlsic and uses exactly the same kind of input file. The only difference is in terms of the output produced: far fewer files(!) but in addition, files in standard form for the instability eigenvectors.

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 (m^{2}s^{-1}), q the rate of local heating (Jm^{-3}s^{-1}), C_p the specific heat capacity (Jkg^{-1}K^{-1}) and ρ the density (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

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

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_{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:

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

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

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

and

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

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

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

We assume that the laterally varying heat-flux, $T_{\rm a}$, locks the flow (i.e. $\partial\Theta/\partial t = 0$ and $\partial \boldsymbol{u}/\partial t = 0$). `blscnlsic` 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

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

This solution is done with Newton-Raphson iteration.

Once the solution $(\boldsymbol{u}_0, \Theta_0)$ is obtained, `blscnlsic` tests the stability of the boundary-locked solution by solving the eigenproblem specified by

$$\begin{aligned} c_a \sigma \tilde{\Theta} &= c_d \nabla^2 \tilde{\Theta} + b_1 \tilde{u}_r r + b_2 \frac{\tilde{u}_r}{r^2} \\ &\quad - c_c \boldsymbol{u}_0.\nabla \tilde{\Theta} - c_c \tilde{\boldsymbol{u}}.\nabla(\Theta_0 + \varepsilon T_{\rm a}) \end{aligned} \tag{25}$$

and

$$\begin{aligned} c_e \sigma \tilde{\boldsymbol{\omega}} &= -c_f \nabla \times (\tilde{\boldsymbol{u}}.\nabla \boldsymbol{u}_0) - c_f \nabla \times (\boldsymbol{u}_0.\nabla \tilde{\boldsymbol{u}}) - c_g \nabla \times (\boldsymbol{k} \times \tilde{\boldsymbol{u}}) \\ &\quad + c_h \nabla \times \left[\tilde{\Theta} \boldsymbol{r} \right] + c_i \nabla^2 \tilde{\boldsymbol{\omega}}. \end{aligned} \tag{26}$$

Here, σ is the complex growth rate of the perturbation variables $\tilde{\boldsymbol{\omega}} = \nabla \times \tilde{\boldsymbol{u}}$ and $\tilde{\Theta}$. If the steady solution contains only wavenumbers 0 and multiples of the non-zero m_0, then sub-classes of the perturbations $\tilde{\boldsymbol{u}}$ and $\tilde{\Theta}$ are defined by the Floquet parameter, M. For example, if $m_0 = 6$ then $\tilde{\boldsymbol{u}}$ and $\tilde{\Theta}$ will have classes defined by $M = 0$, $M = 1$, $M = 2$ and $M = 3$. The $M = 0$ sub-class will contain wavenumbers $\{0, 6, 12, \cdots\}$, the second $\{1, 5, 7, 11, 13, \cdots\}$, the third $\{2, 4, 8, 10, 14, 16, \cdots\}$ and the last $\{3, 9, 15, \cdots\}$. The highest M the program investigates is specified by the integer `MFLOQ`.

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

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

or

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

at the inner boundary, and either

$$T_{\mathrm{a}}(r_{\mathrm{o}},\theta,\phi) = g_{\mathrm{o}}(\theta,\phi) \tag{29}$$

or

$$\left.\frac{\partial T_{\mathrm{a}}(r,\theta,\phi)}{\partial r}\right|_{r=r_{\mathrm{o}}} = g_{\mathrm{o}}(\theta,\phi) \tag{30}$$

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

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

$$g_{\mathrm{i}}(\theta,\phi) = \frac{\sqrt{\varepsilon_{\mathrm{i}}}}{N_{\mathrm{i}}}\sum_{l=0}^{l=L}\sum_{m=0}^{m=l}\left[g_{\mathrm{i},l}^{mc}\cos m\phi + g_{\mathrm{i},l}^{ms}\sin m\phi\right]P_l^m(\cos\theta) \tag{31}$$

and

$$g_{\mathrm{o}}(\theta,\phi) = \frac{\sqrt{\varepsilon_{\mathrm{o}}}}{N_{\mathrm{o}}}\sum_{l=0}^{l=L}\sum_{m=0}^{m=l}\left[g_{\mathrm{o},l}^{mc}\cos m\phi + g_{\mathrm{o},l}^{ms}\sin m\phi\right]P_l^m(\cos\theta), \tag{32}$$

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

$$\int_0^\pi \left[P_l^m(\cos\theta)\right]^2 \sin\theta d\theta = \frac{2(2-\delta_{m0})}{2l+1}. \tag{33}$$

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

$$\int_0^{2\pi}\int_0^\pi \left[g_{\mathrm{i}}(\theta,\phi)\right]^2 \sin\theta d\theta d\phi = \varepsilon_{\mathrm{i}} \tag{34}$$

and

$$\int_0^{2\pi}\int_0^\pi \left[g_{\mathrm{o}}(\theta,\phi)\right]^2 \sin\theta d\theta d\phi = \varepsilon_{\mathrm{o}}. \tag{35}$$

Therefore, if a user specified that the only non-zero coefficients in the series (31) and (32) were to be $g_{\mathrm{o},2}^{2c} = 3.0$ and $g_{\mathrm{o},3}^{2s} = -3.0$ with $\varepsilon_{\mathrm{o}} = 1.0$, the result would be exactly the same had the user specified the coefficients $g_{\mathrm{o},2}^{2c} = 0.012$ and $g_{\mathrm{o},3}^{2s} = -0.012$ with $\varepsilon_{\mathrm{o}} = 1.0$. This gives the user a global control on the normalisation of the "strength" of the functions g_{i} and g_{o} without having to worry about the actually coefficients themselves, other than their relative sizes.

A file specifies all the coefficients $g_{\mathrm{i},l}^{mc}$, $g_{\mathrm{i},l}^{ms}$, $g_{\mathrm{o},l}^{mc}$ and $g_{\mathrm{o},l}^{ms}$. 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. $g_{\mathrm{i},l}^{mc}$ or $g_{\mathrm{o},l}^{mc}$) and `2` to indicate a $\sin m\phi$ harmonic (i.e. $g_{\mathrm{i},l}^{ms}$ or $g_{\mathrm{o},l}^{ms}$). `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 $g_{\mathrm{o},2}^{2c}$ 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 $g_{\mathrm{i},2}^{2c} = 1.0$, $g_{\mathrm{o},2}^{2c} = 1.0$ and $g_{\mathrm{o},3}^{2s} = 1.0$. This fixes the contribution from $g_{\mathrm{o},2}^{2c}$ relative to that from $g_{\mathrm{o},3}^{2s}$, whereas the contribution from $g_{\mathrm{i},2}^{2c}$ 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 blscnlsic
```

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

```
blscnlsic < inputfile
```

The inputs file must have the following format.

```
* input file for blscnlsic
*
boundary_coeffs                         : INHOMOG TEMP. FILE
example_aOUTPUT                         : ROOT
0.666666666   1.666666666  2   1   45   : RI, RO, IVELBC, ITHEBC, LU
  12   0.001   10.0   4    14           : MXATT CTOL DRSV, NEV, NCV
*------------------------------------------------------------------------
*NR  ISP  LH  SYM  MLOW   MINC  MMAX  IOF   CC   CB1   CB2    CA   CE
*  CD   CF     CG     CH       CI   EPS_in  EPS_out  DCH  NCH  MFLOQ
*------------------------------------------------------------------------
*
* In Zhang and Gubbins (1996), Phys Fluids, v8 p1141-1148,
* epsilon is set to 0.001 for the calculation which is performed
* in Table 1.
*
* Their spherical harmonics have normalisation \int_S (Y_l^m)^2 = 4 \pi
* Our function is normalised so that \int_S g(theta,phi)^2 = 1
* Our epsilon must be 2.0 * sqrt( 4.0 * pi ) multiplied by the epsilon
* quoted by Zhang and Gubbins.
*
* Hence eps = 0.001 --> eps = 0.0070898154
*
*-----------------------------------------------------------------------
 30   1   14   1    0      6    12      0   7.0   1.0   0.0   7.0  1.0
   1.0  1.0  447.2136   7500.0   1.0   0.0  0.0070898154     500.0  1   3
 40   1   14   1    0      6    12      0   7.0   1.0   0.0   7.0  1.0
   1.0  1.0  447.2136   7500.0   1.0   0.0  0.0070898154     500.0  1   3
 50   1   14   1    0      6    12      0   7.0   1.0   0.0   7.0  1.0
   1.0  1.0  447.2136   7500.0   1.0   0.0  0.0070898154     500.0  1   3
 50   2   14   1    0      6    12      0   7.0   1.0   0.0   7.0  1.0
   1.0  1.0  447.2136   7500.0   1.0   0.0  0.0070898154     500.0  1   3
*-----------------------------------------------------------------------
 30   1   16   1    0      6    12      1   7.0   1.0   0.0   7.0  1.0
   1.0  1.0  447.2136   7500.0   1.0   0.0  0.0070898154     500.0  1   3
 40   1   16   1    0      6    12      2   7.0   1.0   0.0   7.0  1.0
   1.0  1.0  447.2136   7500.0   1.0   0.0  0.0070898154     500.0  1   3
 50   1   16   1    0      6    12      3   7.0   1.0   0.0   7.0  1.0
   1.0  1.0  447.2136   7500.0   1.0   0.0  0.0070898154     500.0  1   3
 50   2   16   1    0      6    12      4   7.0   1.0   0.0   7.0  1.0
   1.0  1.0  447.2136   7500.0   1.0   0.0  0.0070898154     500.0  1   3
*-----------------------------------------------------------------------
```

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.)

- `DRSV`. The real shift for ARPACK eigensolver.

- `NEV`. The number of eigenvalues requested.

- `NCV`. The length of the Arnoldi factorisation. See [LSY98]) for details.

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` $\rightarrow$ no file output.
  `IOF = 1` $\rightarrow$ standard output for homogeneous part of locked solution only. (This is recommended.)
  `IOF = 2` $\rightarrow$ radial function output.
  `IOF = 3` $\rightarrow$ 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` $\rightarrow$ standard output for both homogeneous part and total locked solution.

- `CC`: Scaling parameter c_c in equations (23) and (25).
- `CB1`: Scaling parameter b_1 in equations (23) and (25).
- `CB2`: Scaling parameter b_2 in equations (23) and (25).
- `CA`: Scaling parameter c_a in Equation (25).
- `CE`: Scaling parameter c_e in Equation (26).

The second of these lines contains the numbers

- `CD`: Scaling parameter c_d in equations (23) and (25).
- `CF`: Scaling parameter c_f in equations (24) and (26).
- `CG`: Scaling parameter c_g in equations (24) and (26).
- `CH`: Scaling parameter c_h in equations (24) and (26).
- `CI`: Scaling parameter c_i in equations (24) and (26).
- `EPS_in`: Strength of heating heterogeneity at inner boundary, $\varepsilon_{\rm i}$, as defined in Equation (34).
- `EPS_out`: Strength of heating heterogeneity at outer boundary, $\varepsilon_{\rm o}$, as defined in Equation (35).
- `DCH`: Increment in parameter c_h: Δc_h. See `NCH`.
- `NCH`: Number of increments in c_h to be made. If `NCH = 0` then only a single solution will be calculated for that given run. Otherwise, solutions and instability calculations will be performed for $c_h =$ `CH`, $c_h =$ `CH + DCH`, `...` , $c_h =$ `CH + NCH*DCH`. This way, a search of parameter space is much more effective to administer.
- `MFLOQ`. The maximum value of Floquet integer, M.

An alternative code to run is `blscnlsic_evecs`. This performs an identical calculation to `blscnlsic` but outputs files differently (produces solution vector files for all of the instability eigenmodes).

1.1 Subprograms required for blscnlsic_evecs

SUBS subroutines

```
fopen.f vecop.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 shkeer.f hmfwt.f svfwt.f xarrwt.f
svprnt.f vtfhsr.f itslsr.f fclose.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 asvdr.f radvlf.f sbrrfc.f
vmeps.f evalas.f evecex.f matop.f shvect.f vfdp.f
forsst.f vfcp.f  vf2qst.f itfa.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 dvecz.f asvta.f asvcl.f
vesr.f  cubeop.f fftrlv.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 sqrll1.f dl.f    dldl.f
```

SUBS integer function

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

BLAS double precision function

```
dnrm2.f ddot.f  dasum.f
```

BLAS integer function

```
idamax.f
```

BLAS subroutines

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

ARPACK subroutines

```
dnaupd.f dneupd.f dnaup2.f dvout.f ivout.f second.f
dstatn.f dmout.f dgetv0.f dnaitr.f dnconv.f dneigh.f
dngets.f dnapps.f dlaqrb.f dsortc.f
```

LAPACK subroutines

```
dgetrf.f dgetri.f dgbtrf.f dgbtrs.f dgetf2.f dlaswp.f
xerbla.f dtrtri.f dgbtf2.f dlahqr.f dgeqr2.f dlacpy.f
dlaset.f dorm2r.f dtrevc.f dtrsen.f dtrti2.f dlabad.f
dlanv2.f dlarfg.f dlarf.f dlaln2.f dlacon.f dtrexc.f
dtrsyl.f dlarnv.f dlascl.f dlartg.f dlassq.f dladiv.f
dlaexc.f dlasy2.f dlaruv.f dlarfx.f
```

LAPACK double precision function

```
dlapy2.f dlamch.f dlanhs.f dlange.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, NCVM
 PARAMETER ( NRMAX = 50, LHMAX = 62, NHMAX = 200, 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, NCVM = 20 )
```

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.

- NCVM is the maximum value of NCV.

- 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 BLSCNLSIC_EVECS

If the filename stem "root" was specified in the input file, the files root.res and root.log will be created.

For each run, files in standard format (i.e. .ints, .vecs and .xarr files) will be output. For the case of argument, we give the names for the first run (i.e. run001).

The indices and radial spacings files for the boundary locked flow are output with the names

```
root.run001.main.ints
root.run001.main.xarr
```

respectively. The subsequent vectors for the homogeneous part of the solution at increasing Rayleigh numbers are output with the names

```
root.run001.ch000.vecs
root.run001.ch001.vecs
root.run001.ch002.vecs
etc ...
```

These are all entirely specified with the `.xarr` and `.ints` files above.

If, in addition, the total vector (i.e. including the inhomogeneous part) is required, then the files

```
root.run001.ch000.inh.ints
root.run001.ch000.inhom
etc ...
```

are output. (Note: the new `.ints` file is necessary because of the boundary conditions for the temperature functions - the `.xarr` file is the same. Note also that files `root.run001.ch000.inh.ints`, `root.run001.ch001.inh.ints` etc. will all be identical: this was a programming oversight and easily corrected if required.)

Our instability vectors have spherical harmonic representations defined by the files

```
root.run001.M00.inst.ints
root.run001.M01.inst.ints
etc ...
```

where M are defined by the input parameter `MFLOQ`. The vectors themselves are given in files

```
root.run001.ch001.M00.inst01.sv
root.run001.ch001.M00.inst02.sv
root.run001.ch001.M00.inst03.sv
etc ...
```

This program can fill disk space very quickly with what I believe to be not very useful information!

1.4 Sample runs of blscnlsic_evecs

The directory

```
$LEOPACK_DIR/SAMPLERUNS/BLSCNLSIC_EVECS
```

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

This is a scaled down version of the calculation performed by `blscnlsic`.

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.

[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.

[LSY98] R. B. Lehoucq, D. C. Sorensen, and C. Yang. *Arpack users guide: Solution of large scale eigenvalue problems by implicitly restarted Arnoldi methods.* SIAM, 1998.