bigi*.m

Island-trapped waves with stratification, topography, mean flow and bottom friction in Matlab

K.H. Brink

May 18, 2018
July 6, 2018

This set of Matlab mfiles (all having names that begin with "bigi") can be used to calculate island-trapped wave modal structures and dispersion curves under very general circumstances for a circular island. A complex frequency is allowed, so that instability and damping can be accounted for directly. Modal structures and energy diagnostics are provided. For most applications, the code is only useful for subinertial wave frequencies (i.e., the real part of wave frequency is smaller than the Coriolis parameter). For interpreting the model results, see Brink (1999), which deals with the case with no mean flow or finite bottom friction. The present code was developed independently of the Fortran code used in that publication.

Although a number of mfiles are involved here, there are only three that you would actually call directly:

bigisetup.m Helps you set up the input data array (which comes out as array "datnew" in the following example). Call the mfile and it will prompt you with questions. Call as:

```
>> datnew = bigisetup;
```

bigiomp.m This is the file that actually calls all the others and so does the calculations. Call as:

```
>> bigiomp(datnew)
```

bigifinch.m This is used to modify a pre-existing input array. It will give you a menu for what you want to change. Call as:

```
>> datnewer = bigiomp(datnew);
```

For examples of usage, see the screen displays copied at the end of this document.

The Problem:

We seek to find the free wave solutions for linearized island-trapped waves at a circular island. The depth $h(r)$ is uniform in the azimuthal (θ) direction. The offshore (radial) direction is r ($r = r_0$ is the coast) and the vertical direction is z. The bottom is at $z = -h(r)$ and the origin is at the island center. Dissipative effects are limited to the infinitesmally thin bottom boundary layer.

The equations of motion in cylindrical coordinates are:

$$v_t + uv_{0r} + v_0 u_\theta / r + uv_0/r + wv_{0z} + fu = -\rho_0^{-1} r^{-1} p_\theta + \rho_0^{-1} \tau^r_z \qquad (1a)$$

$$u_t + v_0 u_\theta / r - 2v_0 v/r - fv = -\rho_0^{-1} p_r + \rho_0^{-1} \tau^\theta_z \qquad (1b)$$

$$0 = - p_z - g\rho_2 \qquad (1c)$$

$$r^{-1}(ru)_r + r^{-1} v_\theta + w_z = 0 \qquad (1d)$$

$$\rho_{2t} + u\rho_{1x} + v_0 \rho_{2\theta} / r + w\rho_{1z} = 0 \qquad (1e)$$

where (u, v, w) are the perturbation velocity components in the (r, θ, z) directions, respectively, p is perturbation pressure, f is the (constant) Coriolis parameter, $v_0(r, z)$ is a steady azimuthal mean flow, density is broken up as: $\rho_0 + \rho_1(r, z) + \rho_2(r, \theta, z, t)$ and $\rho_0 >> \rho_1 >> \rho_2$. The $2v_0 v / r$ term in (1b) comes from expanding $(v_0 + v)^2$. In all cases, subscripted independent variables represent partial differentiation.

The mean azimuthal flow $v_0(r, z)$ obeys

$$-v_0 (f + v_0 / r) = -\rho_0^{-1} p_{0r} \qquad (2a)$$

so that

$$v_{0z} (f + 2v_0 / r) = -g \rho_0^{-1} \rho_{1r} \ . \qquad (2b)$$

Turbulent stresses in the *r, θ* directions are given by τ^r , τ^θ, and the bottom stress ($\tau_B{}^r$, $\tau_B{}^\theta$) is related to the interior bottom velocity by

$$\tau_B{}^r = \rho_0 R\, u_B, \qquad \tau_B{}^\theta = \rho_0 R\, v_B, \tag{3 a, b}$$

where *R(r)* is a bottom resistance coefficient having units of velocity, and a subscript "B" denotes a quantity evaluated at the bottom. We also define

$$N^2 = -g\rho_0^{-1}\, \rho_{1z} \qquad \text{and} \tag{4a}$$

$$M^2 = -g\rho_0^{-1}\, \rho_{1r}\,. \tag{4b}$$

The problem is solved by assuming that

$$p = p'(r, \theta)\, exp[i(\omega t + m\theta)], \text{ etc.,} \tag{5}$$

where the frequency ω can be complex and the azimuthal wavenumber *m* is a real integer. Henceforth, the "primes" will be dropped. Using this form, the problem can be reduced to a single partial differential equation for pressure (analogous to that in Brink, 2006):

$$\begin{aligned} 0 = {} & \omega' p_{rr} + p_{rz}\,(-2 s\omega'^2) + p_{zz}\,\omega'\,(s^2 + G/N^2) \\ & + p_r\,[\omega' \Gamma + \omega'_r - s\,\omega_z + m f_1 / r - m f^*/r] \\ & + p_z\,[s\,\omega' \Gamma - (s\omega')_r - s(s\omega')_z + G\,(\omega'/N^2)_z + m\,s\,(-f_1 + \omega'^2/f_1)/r] \\ & + p\,[m f_1 \Gamma / r + m(f_1/r)_r - ms\,(f_1/r)_z - \omega' m^2 / r^2] \end{aligned} \tag{6}$$

where

$$\omega' = \omega + mv_0/r \quad (7a)$$

$$s = M^2/N^2, \quad (7b)$$

$$f_1 = f + 2v_0/r\ , \quad (7c)$$

$$f_2 = f + v_{0r} + v_0/r\ , \quad (7d)$$

$$f^* = f_2 - sM^2/f_1\ , \quad (7e)$$

$$G = f_1 f^* - \omega'^2\ , \quad (7f)$$

$$\Gamma = (-G_r + s\,G_z)/G\ -s_z - 1/r\ , \quad (7g)$$

This equation and the notation are similar to those of Mooers (1975 a, b), except that these are in cylindrical coordinates and apply to hydrostatic conditions.

The mean azimuthal velocity field is taken to be a Gaussian in r and z, centered at $r = r_v$, $z = -z_v$.

$$v_0(r, z) = \langle v \rangle \exp\{-[(r-r_v)/D^r]^2\} \exp\{-[(z - z_v)/D^z]^2\} \quad (8)$$

where $\langle v \rangle$ is the peak value, and the horizontal scale D^r has different values onshore and offshore of r_v, while the vertical scale D^z is allowed to be different upward and downward of z_v. If you desire to have a different initial velocity form, you can simply change the code in file bigivzcal2.m, around lines 35-78. Velocity gradients and density are computed numerically given v_0, so that no other code need be changed.

As noted by Brink, (1982) or Dale et al. (2001), equation (6) admits a spurious solution at $\omega' = f$ when $v_0 = 0$. The presence of this mode also distorts modal structures and dispersion curves for nearby frequencies. Typically, this spurious pressure mode is vertically uniform and it decays only slowly with distance offshore.

The boundary conditions are as follows:

z = 0:

$$w = \delta\,(g\rho_0)^{-1}[p_t + v_0\, p_{\theta\,/\,r}], \tag{9}$$

where $\delta = 1$ for a free surface condition and $\delta = 0$ for a rigid lid condition. In terms of pressure,

$$0 = \omega'(\,s^2 + G\,/N^2\,)\,p_z \;\; - s\,\omega'\,p_x \;\; + [\omega'\delta\,G\,/g - m\,s\;\; f_1\,/\,r\,]\,p \tag{10}$$

z = - h(r):

$$0 = w + h_r\,u + \;\; r^{-1}(r\;U_E)_r + \;\; r^{-1}V_{E\theta} \;\;, \tag{11}$$

where the Ekman transport is given as

$$U_E = - RG^{-1}\;(f_1\,v + i\omega'u)_B \;\;, \tag{12a}$$

$$V_E = - RG^{-1}\;(i\omega'v - f^*u)_B \;\;, \tag{12b}$$

and the subscript "B" denotes a quantity evaluated at the bottom (along $z = -h(r)$). In terms of pressure,

$$\begin{aligned}
0 = {} & p_{rBr}\;(R\,/G)\;(f_1 f^* + \omega'^2) \;\; + p_{zBr}\;(R\,/G)(-2s\omega'^2\,) \\
& + p_{zB}\;\{i\;\omega'(G + s\,M^2 - M^2\;h_r)\,/N^2 \;\; - s\omega'(G\,f_1)^{-1}[f_1(\omega'R)_r + \omega'(f_1R)_r] \\
& \qquad + (R/G)[4G_r s\omega'^2\,/G \;\; - \omega'(s\omega')_r - f_1(s\omega'^2/f_1)_r \\
& \qquad + (s\omega'/r)(-2\omega' + mf^* + m\;\omega'^2/f_1)]\} \qquad (13) \\
& + prB\;\{-i\omega'(s\text{-}hr) + [-\omega'(\omega'R)r + f^*(f1R)r]\,/\,G \\
& \qquad + (R/G)[-2G_r\;(f_1 f^* + \omega'^2)/G + \omega'\omega'_r + f_1\,f^*_r \;\; + (f_1 f^* + \omega'^2\,\text{-}2m\omega' f^*)/r]\}
\end{aligned}$$

$$+ p_{Br}\,(R/G)\,(2m\omega' f_1 / r)$$

$$+ p_B \{ -imf_1(s-h_r)/r + m(rG)^{-1}[f_1(\omega' R)_r + \omega'(f_1 R)_r]$$

$$+(R/G)\{ -G_r(4mf_1\omega')/(rG) + m\,(\omega' f_{1r} + f_1\omega'_r)/r$$

$$-m^2(f_1 f^* + \omega'^2)/r^2]\}$$

r = r₀:

A closed boundary condition gives no net flux across the coastal boundary:

$$0 = U_E + \int_{-h}^{0} u\,dz \approx U_E + uh \qquad (14)$$

When there is bottom friction ($U_E \neq 0$), this condition really only makes sense if the vertical variations in u are negligible over the depth $h_0 = h(r_0)$. With no bottom friction, this boundary condition is sensible regardless of the coastal wall height.

Stated in terms of pressure, this condition is:

$$0 = - i\,\omega' h_0\, p_r + (i s\omega') h_0\, p_z - (i\,m f_1 h_0 / r)\, p \qquad (15)$$

$$+RG\,G_B^{-2}\{ - (f_1 f^* + \omega'^2)\, p_{rB} + (2 s\omega'^2)\, p_{zB} - (2m\omega' f_1 / r)\, p_B\}$$

r = L:

The offshore boundary condition is an adaptation of the approximate open condition introduced by Brink (1982):

$$0 = (ru)_r \qquad . \qquad (16)$$

This condition seems to work well as long as the outer grid boundary ($r = L$) occurs far offshore of where the modal solutions have substantial structure. The condition, as used,

requires that the bottom is flat at the offshore boundary. The condition in terms of pressure is:

$$0 = -\omega' p_{rr} + s\omega' p_{zr} + p_r (\omega' G_r / G - \omega' / r - \omega'_r - m f_1 / r) \quad (17)$$
$$+ p_z [-s\omega' G_r / G + (s\omega')_r + s\omega' / r]$$
$$+ p [m f_1 G_r / (rG) - m(f_1 / r)_r - m f_1 / r^2]$$

The problem is now expressed in stretched coordinates that map the problem into a rectangular domain (e.g. Wang and Mooers 1976). Specifically,

$$\varphi = z / h(r) \quad (18a)$$

and

$$r' = r. \quad (18b)$$

The problem is then solved numerically via resonance iteration. An arbitrary forcing is applied through the surface boundary condition (essentially, a localized wind stress curl near the grid center), and ω is searched for a maximum response so as to establish a complex frequency corresponding to a given wavenumber m. This search makes use of the Matlab function "fminsearch.m".

Using the software:

Examples of what your Matlab screen will look like as you run the different mfiles are shown at the end of this document. The main routine, "bigiomp", requires an input array that can be generated by "bigisetup" and modified by "bigifinch". When you run "bigisetup" or "bigifinch",

you simply need to answer the questions posed. Do not forget to use square brackets when entering more than one number on a line.

When running "bigiomp", you will first be shown a few simple parameter values. You also get a section plot showing topography, stratification (minus a constant background value), and the mean flow. If f^*/f is negative anywhere, f^* will also be plotted (in red) on the density plot. If you are allowing pauses during the run, you are advised of pauses, and need to hit a key each time to continue. During iteration toward a solution, the azimuthal wavenumber m is displayed, and the program displays complex frequency and the inverse resonance parameter (rrr = inverse of the integral of $r|p|^2$ over r and z) for each ω estimate. When you are at a resonance (free mode), rrr becomes very small. You can thus watch the convergence onto a complex free modal frequency. Convergence to your given tolerance is then announced, and the free mode pressure is the plotted as a section. A pause can also be announced at this point. Hit any key to continue, and you will then also get section plots of u, v, w, ρ, energy diagnostics in arbitrary units (energy pool sizes and transfers: the dissipation integral should be a negative quantity), and line plots of p, v, u at the surface. If you set the program to run more than one frequency/wavenumber, it will then display the next wavenumber and go on from there.

Before any plots are made, the pressure field is normalized (Brink, 1999) so that

$$1 = \int_{-h(r_0)}^{0} |p(r_0, z)|^2 dz + \int_{r_0}^{L} h_r |p(r, -h)|^2 dr \tag{19}$$

This means that, in the following plots, pressure has units of ($cm^{-½}$), velocity has units of ($cm^{-3/2}$ sec gm^{-1}), and density has units of ($cm^{-5/2}$ sec^2). Relations in Brink (1999) can be used to estimate wind coupling coefficients when $R = 0$ and $\langle v \rangle = 0$, given his normalized modal form.

If you have set the program to compute more than one frequency/wavenumber pair, the program will also draw a complex dispersion curve once all pairs have been obtained. In this case, you will be asked at the end of the run whether you want to save the dispersion curve as a .mat file.

Some Notes on using the software:

The $r = r_0$ boundary condition is closed. This causes no issues when $R = 0$, since it just imposes $u = 0$ at each depth, a reasonable condition at a coastal wall. When $R \neq 0$, bottom friction is applied, and the bottom Ekman transport that reaches the coastal wall is balanced by an evenly distributed interior flow. Thus, there is an assumption that the water is shallow enough that $|u_z/u|$ is small. This would be expected if $(f/N)\, L^r$ is large relative to h_0, where L^r is a representative modal scale in the r direction. This means that, with bottom friction, the boundary water depth

should be probably substantially less than 50 m. Further, see Mitchum and Clarke (1986) for aspects of boundary layer theory that affect one's choice of wall height. If there is no bottom friction, then the coastal wall can have any depth.

At the offshore boundary, the bottom should be flat, but the water does not need to be shallow Care should be taken to make sure that the offshore boundary is located far from the main modal structure's location of maximum amplitude.

The numerical grid extends one point outside the *r, z* domain at each end (nn is the number of *r* grid points, and mm the number of *z* (or, equivalently φ) grid points. There is also a grid point beyond each boundary. Thus, for example, nn = 5 gives one interior grid point, 2 on boundaries and 2 outside the domain.

In choosing a grid size, caution is advised. One should experiment with grid sizes in order to be confident of a good answer. I find that nn = 80-120, mm = 20-30 seems to work well in many coastal and island cases. The number of operations or amount of memory needed goes something like nn*mm*mm. There is a test for hydrostatic consistency (the ratio computed is $h_r \Delta r/(h \Delta \varphi)$), and results seem best if the ratio given is less than around 1-5. Further, steep bottom topography can cause problems. It appears that a reasonable criterion is to pick nn such that

$$\Delta r \leq 0.5 \max\left[h \left(\frac{dh}{dr}\right)^{-1} \right] \quad . \qquad (20)$$

The mean azimuthal flow is assumed to be Gaussian (with different onshore, offshore, up and down length scales, as well as a variable center position and depth). This provides a fair bit of flexibility. Once the velocity field is computed, the density structure is computed using a variant of the thermal wind equation (2b). The density field can be left undisturbed at either the offshore or onshore end of the domain. The program will alert you (by blue "+" signs) to any regions where your choices have led to gravitationally unstable vertical density gradients, i.e., when $(\rho_0 + \rho_1)_z > 0$.

There is substantial documentation within the mfile "bigiomp.m" for further detail.

Results for pressure, velocity and density are plotted. The smaller of the real or imaginary is suppressed if it is too small (maximum absolute value less than about 1000 times less than the other's maximum absolute value, as it is set now). This ratio is controlled by the value of "ratnplot" in "bigiuvwr.m" and "bigiomp.m".

The search can be very inefficient if you give it 0 as the starting guess for real or imaginary frequency. Better to give it a very small nonzero value. For example, if you are seeking a mode

with a real frequency (say, when R and v_0 are both zero), then a good starting guess might be $1\times10^{-5} + i\ 1\times10^{-13}$ sec^{-1} rather than simply $1\times10^{-5} + i\ 0$ sec^{-1}. Similarly, if you are seeking a damped wave frequency (e.g., if $R \neq 0$ and $v_0 = 0$), then it would be well to start with a guess like $(1 + i)\times2\times10^{-5}$sec^{-1}, rather than $(2\times10^{-5} + i\ 1\times10^{-10})$ sec^{-1} , for example (i.e., use similar real and imaginary magnitudes, rather than a large and very small value). Similarly, if you are seeking an unstable mode, you might want to start searching in a similar way, e.g., $(1 - i)\times5\times10^{-6}$ sec^{-1}. The magnitude of the numbers here are just for illustration. The relative values of the real and imaginary parts are the real point. The search will usually converge on some value, but it may not be meaningful. The search might also miss meaningful solutions if the initial guess is too far off.

In considering the outputs, you need to judge whether the imaginary value is small enough to be effectively zero, but this is usually pretty obvious. Looking at the energy diagnostics can often tell you if an answer is silly. The nonzero imaginary frequency means, also, that there can be very small energy exchanges in the energy diagnostics when the flow is stable. Further, if the contoured solutions show a concentration of contours near-surface in the center of the grid (where the arbitrary forcing occurs), then the result is likely bogus. Again, you need to decide what is effectively zero.

By the same token, if you give the algorithm an initial estimate for real or complex frequency that has few places of accuracy (say 1e-5), the Matlab search code will look around at fairly large increments (perhaps ± 20%), and the search can miss nearby solutions. If you provide more places of accuracy (say 1.01e-5), the search will be initially confined to more nearby locations (perhaps ± 3%), and it becomes much more likely that nearby solutions will be found.

The search can sometime miss a complex frequency, for example if searching all takes place only near the real axis, but the true result has a substantial imaginary part. If it is a good solution, the inverse resonance parameter in the output (rrr) should be several orders of magnitude lower than neighboring values. Use a smaller accuracy tolerance (e) or start the search again farther from the real axis.

For an accuracy estimate (“e” option in bigifinch.m), I find that 0.0001 works out fairly well. This means a nominal accuracy of 0.01% of the absolute value of the initial frequency guess.

The bottom stress is assumed to be proportional to the bottom velocity. For the errors associated with this assumption in the presence of a mean flow, see Brink (1997).

If more than one frequency is computed, there is an option to save the “dispersion curve” information (wavenumbers and complex frequencies).

When the Ertel potential vorticity $q = -N^2 f^*$ changes sign relative to f, symmetric instability is possible (Hoskins,1974). So, the routine "bigirhocal2.m" tests to see if f^*/f is negative anywhere. If this happens, f^* is contoured in red on the same plot as mean density, and a message appears on your Matlab screen. If f^*/f does not become negative, it is not contoured, and no message about it is given. Experience with the program suggests that modes calculated when f^* changes sign are of questionable merit. Structures are often complex and have short scales near the region of negative f^*. It seems probable that if, in nature, a flow is symmetrically unstable, it will quickly mix to a neutral state with zero or positive f^*/f. So, even a "good" modal solution for a symmetrically unstable case is of questionable value.

Given the presence of the near-inertial spurious mode, and given that the "open" boundary condition is only reasonable for subinertial frequencies, the model results should only be trusted for subinertial frequencies, $|\mathrm{Re}(\omega)| < |f|$. If, in computing a dispersion curve, you start to obtain real frequencies greater than the inertial, the superinertial results should be discarded.

All frequencies are in radians per second, and all wavenumbers in radians per cm. Radians do not have units, so these units could be written as "1/sec" for example.

You can get spurious results if the domain is too large relative to the natural scale of the wave under consideration. For example, with a domain size $L = 70$ km, trying to find a flat-bottom internal Kelvin wave with internal Rossby radius = 3 km yields bad results. Doing the same calculation with $L \leq 40$ km yields reasonable results. This problem arises because the arbitrary forcing used to create the resonance is applied near $L/2$. Trouble can arise when the desired wave has negligible amplitude at this distance offshore.

A file is included with the input information for a realistic Bermuda case (berm0.mat).

Mfiles in this package:

This collection of mfiles includes the following:

Mfiles that you can call directly:

bigiomp.m

The driver for the island-trapped wave calculations. Carries out all input functions, calls the functions for computing mean density and velocity fields, and for searching for resonant frequencies. Also does some output and graphics functions. An example is given below.

bigisetup.m

This mfile sets up an input array for "bigiomp", based on user responses to questions. An example of its use is given below.

bigifinch.m

This mfile changes an existing "bigiomp" input array, without having to generate an all new array. An example of its use is given below.

Mfiles that are called through bigiomp.m (either directly or indirectly):

bigidep.m
Computes a gridded depth profile, along with its derivatives.

bigiconchk.m
Grid consistency check to see if the bottom slope is large relative to vertical spacing.

bigir.m
Computes gridded bottom resistance parameter, *R*.

biginorm.m
Normalizes the pressure field according to Brink (1999), eqn. 9.

biginsq.m
Computes an undisturbed (by mean flow) gridded N^2 *(z)* profile.

bigivzcal2.m
Calculates the mean velocity field and its vertical derivative, from inputs.

bigirhocal2.m
Calculates the modifications to the basic density field caused by the mean flow.

bigidrver.m
Mainly called from fminsearch. This sets up the matrix equations that are equivalent to the field differential equation and its boundary conditions. Also solves the equation, and computes an integral of the response to evaluate convergence.

bigisigtoz.m
Converts a field from sigma coordinates to *z* coordinates.

bigiengdiag.m
Does the energy diagnostics for each solution.

bigiuvwr.m
Called through bigiengdiag.m. Calculates *u, v, w, ρ* fields and plots them.

bigiztosig.m
Converts *z* coordinates to sigma coordinates.

The following are called by bigidrver.m:

bigicc.m
Computes coefficients for the coastal boundary condition.

bigioff.m
Computes coefficients for the offshore boundary condition.

bigibot.m
Computes coefficients for the bottom boundary condition.

bigisbc.m
Computes coefficients for the surface boundary condition.

bigimatco.m
Computes coefficients for interior points on the finite difference grid.

bigiint.m
Computes the integral of the response magnitude.

References:

Brink, K.H., 1982. A comparison of long coastal trapped wave theory with observations off Peru. *Journal of Physical Oceanography*, **12**(8), 897–913.

Brink, K. H., 1997. Time-dependent motions and the nonlinear bottom Ekman layer. *Journal of Marine Research*, **55**(4), 613–631.

Brink, K.H., 1999: Island-trapped waves, with application to observations off Bermuda. *Dyn. Atmos. Oceans*, 29, 93-118.

Brink, K. H., 2006. Coastal-trapped waves with finite bottom friction. *Dynamics of Atmospheres and Oceans*, **41**, 172-190.

Dale, A.C., J.M. Huthnance, and T.J. Sherwin, 2001: Coastal-Trapped Waves and Tides at Near-Inertial Frequencies. , **31**, 2958–2970.

Hoskins, B.J., 1974: The role of potential vorticity in symmetric stability and instability. Quarterly Journal of the Royal Meteorological Society, 100(425), 480-482.

Mitchum G. and A.J. Clarke, 1986: The frictional nearshore response to forcing by synoptic scale winds. *J. Phys. Oceanogr.* 16: 934-946.

Mooers, C.N.K., 1975a: Several effects of a baroclinic current on the cross-stream propagation of inertial-internal waves. *Geophys. Fluid Dyn.,* 6, 245-275.

Mooers, C.N.K., 1975b: Several effects of a baroclinic current on the three-dimensional propagation of inertial-internal waves. *Geophys. Fluid Dyn*., 6, 277-284.

Wang, D.-P. and C.N.K. Mooers, 1976: Coastal-trapped waves in a continuously stratified ocean. *J. Phys. Oceanogr.,* 6: 853-863.

Examples of running the mfiles.

The following are examples of what it will look like in your Matlab window to run these programs.

bigisetup.m

Here, we create the input array for a simple example.

```
>> datnew = bigisetup;


        Island-trapped waves

This mfile will ask you a sequence of questions
    that are used to build the input array.

How many total gridpoints do you want in the cross shelf direction? (nn) 70
How many total gridpoints do you want in the vertical? (mm) 40
First guess at real part of frequency (rad/sec)? 3e-5
First guess at imaginary part of frequency (rad/sec)? 2e-6
Enter 0 for a rigid lid, 1 for a free surface (del) 0
Enter the Coriolis parameter (f) (rad/sec) 1e-4
Enter the outermost radius (rmax) (km) 80
```

```
Enter the nominal fractional accuracy for the solution (eps) 0.0001
Enter the number of frequencies to be computed (npts) 1
Enter the first azimuthal wavenumber to use (m_zero) (integer) 1
How many radius, depth pairs will you provide (ndep >=1) 3
Array of radiuses for depth values, starting at r_zero (rdep in km) (dimension ndep) [15 25 40]
Array of depths corresponding to rdep (depr in m) [10 200 2000]
Number of distance, bottom friction pairs to read (nr) 2
Offshore distances for bottom friction values (rr in km) [15 35]
Array of bottom friction values corresponding to rr (R in cm/sec) [0.05 .02]
Number of Nsquared values to read? (nnsq) 2
Depth increment for input of Nsquared values? (zr in m) 5000
Exponential tail length for Nsquared extrapolation (alph in km) 1
Nsquared values starting at the surface (nsqr in rad^2 /sec^2) (nnsq values) [1 1]*1e-5
Input peak value of mean azimuthal flow (vzero: cm/sec) 0
Enter 0 to skip pauses to see graphics, 1 to see graphics during execution 1
>>
```

bigiomp.m

This example finds an $n = 2$ (i.e., radial mode 2), $m = 1$ (azimuthal mode 1) damped free mode with no mean azimuthal flow.

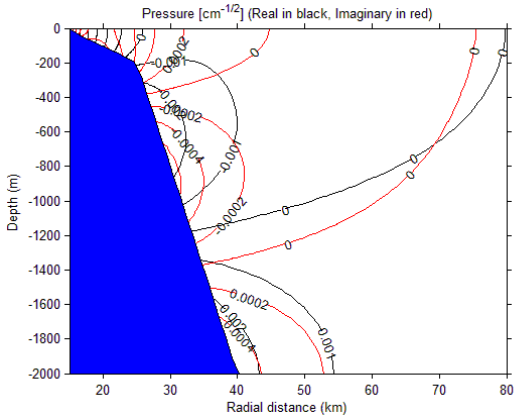
```
>>
>> bigiomp(datnew);

Rigid lid
f = 1.0000e-04 rad/sec
nn, mm = 70  40

 Max consistency ratio = 23.3436 at r = 15 km
       This should be kept less than one, and definitely less than 10


Pause on: hit any key to continue


Wavenumber = 1

wr, wi, rrr = 3.0000e-05 2.0000e-06   1.5811e-33
wr, wi, rrr = 3.1500e-05 2.0000e-06   6.1305e-34
wr, wi, rrr = 3.0000e-05 2.1000e-06   1.5812e-33
wr, wi, rrr = 3.1500e-05 1.9000e-06   6.0630e-34
wr, wi, rrr = 3.2250e-05 1.8000e-06   1.6884e-34
```

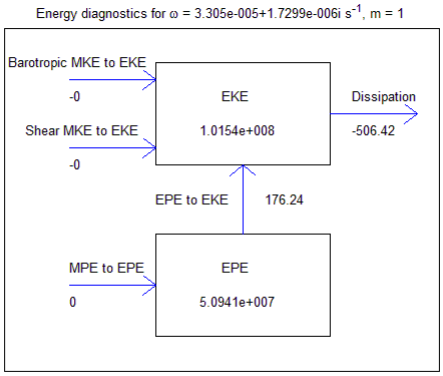
wr, wi, rrr = 3.3750e-05 1.8000e-06 1.0730e-34
wr, wi, rrr = 3.5625e-05 1.7000e-06 9.4253e-34
wr, wi, rrr = 3.4500e-05 1.6000e-06 3.9187e-34
wr, wi, rrr = 3.3750e-05 1.7000e-06 1.0668e-34
wr, wi, rrr = 3.5250e-05 1.7000e-06 7.5083e-34
wr, wi, rrr = 3.3000e-05 1.7750e-06 1.1336e-36
wr, wi, rrr = 3.3000e-05 1.6750e-06 1.3894e-36
wr, wi, rrr = 3.2250e-05 1.7500e-06 1.6799e-34
wr, wi, rrr = 3.3375e-05 1.7125e-06 2.4630e-35
wr, wi, rrr = 3.2625e-05 1.7375e-06 4.6645e-35
wr, wi, rrr = 3.3188e-05 1.7188e-06 4.5340e-36
wr, wi, rrr = 3.2813e-05 1.7313e-06 1.4353e-35
wr, wi, rrr = 3.3094e-05 1.7219e-06 4.6797e-37
wr, wi, rrr = 3.3094e-05 1.8219e-06 2.4995e-36
wr, wi, rrr = 3.3023e-05 1.7117e-06 2.6735e-37
wr, wi, rrr = 3.3117e-05 1.6586e-06 2.3213e-36
wr, wi, rrr = 3.3029e-05 1.7459e-06 1.7467e-37
wr, wi, rrr = 3.2959e-05 1.7357e-06 2.1007e-36
wr, wi, rrr = 3.3060e-05 1.7253e-06 2.6829e-38
wr, wi, rrr = 3.3066e-05 1.7595e-06 2.6815e-37
wr, wi, rrr = 3.3034e-05 1.7237e-06 7.8800e-38
wr, wi, rrr = 3.3065e-05 1.7031e-06 2.2728e-37
wr, wi, rrr = 3.3038e-05 1.7352e-06 4.5257e-38
wr, wi, rrr = 3.3064e-05 1.7369e-06 5.5408e-38
wr, wi, rrr = 3.3057e-05 1.7336e-06 1.1479e-38
wr, wi, rrr = 3.3079e-05 1.7237e-06 1.9881e-37
wr, wi, rrr = 3.3048e-05 1.7323e-06 2.7273e-39
wr, wi, rrr = 3.3045e-05 1.7406e-06 3.5378e-38
wr, wi, rrr = 3.3056e-05 1.7291e-06 7.6714e-39
wr, wi, rrr = 3.3048e-05 1.7279e-06 3.2264e-39
wr, wi, rrr = 3.3040e-05 1.7311e-06 2.9278e-38
wr, wi, rrr = 3.3052e-05 1.7296e-06 5.4696e-40
wr, wi, rrr = 3.3053e-05 1.7341e-06 4.6272e-39
wr, wi, rrr = 3.3049e-05 1.7294e-06 8.2354e-40
wr, wi, rrr = 3.3053e-05 1.7267e-06 3.9621e-39
wr, wi, rrr = 3.3049e-05 1.7309e-06 5.9145e-40
wr, wi, rrr = 3.3053e-05 1.7311e-06 1.0513e-39
wr, wi, rrr = 3.3050e-05 1.7299e-06 1.8329e-40

Converged!

wr, wi, rrr = 3.3050e-05 1.7299e-06 1.8329e-40
>>

Sample figures:





bigifinch.m

This changes the initial guess for complex frequency and leads to finding the $n = 1$, $m = 1$ mode which has $\omega = 5.82\times10^{-5} + i2.24\times10^{-6}$ sec^{-1}.

```
>>
>> datrev = bigifinch(datnew);
```

First you need to select what you want to change.

Options are:
- Grid size: enter "g"
- Initial Frequency guess: w
- Boundary conditions: b
- Coriolis parameter: f
- Domain size: x
- Dispersion curve definition: d
- Nominal accuracy: e
- Depth profile: h
- Bottom friction: R
- Stratification: n
- Mean flow field: v
- Pauses during execution: p

Any arrays are row arrays, not column arrays

Select an option w

```
Old frequency guess (sec^-1) 3e-05       2e-06
Enter new [wreal wimag]   (rad sec^-1) [6e-5 2e-6]
>>
```