InversionKit

Version 2.2

Daniel Reese
Gaël Buldgen
and
Sergei Zharkov

January 2016

Contents

1 Getting started **4**
1.1 Running the program . . . 4
1.2 Using the program . . . 4
1.3 Preliminary remarks . . . 5

2 File formats **24**
2.1 Stellar models . . . 24
2.1.1 MOD (binary/ascii) . . . 24
2.1.2 CLES format . . . 25
2.1.3 LOSC format . . . 26
2.1.4 Comparison of different model formats . . . 26
2.1.5 Useful formulas . . . 26
2.1.6 File format for the list of models for the HR diagram . . . 27
2.2 Eigenmodes . . . 27
2.2.1 The AMDE format . . . 28
2.2.2 The FAMDE format . . . 28
2.2.3 The FILOU format . . . 28
2.3 Target profiles . . . 29
2.4 Observed frequencies and rotational splittings . . . 29
2.5 Output files from Monte Carlo runs for splittings inequalities . . . 30
2.6 GZIP compression . . . 31

3 Treatment of pulsation modes **31**
3.1 Matching observed and theoretical modes . . . 31
3.2 Pulsation calculations . . . 31
3.3 JCD's equations . . . 32
3.3.1 Non-radial modes . . . 32
3.3.2 Radial modes . . . 33
3.4 Lagrangian pressure perturbations . . . 34
3.4.1 Non-radial modes . . . 34
3.4.2 Radial modes . . . 35
3.5 Eulerian pressure and density perturbations . . . 36
3.5.1 Non-radial modes . . . 36
3.5.2 Radial modes . . . 37
3.6 Reconstructing missing variables . . . 38

4 Inversions **39**
4.1 Rotation inversions . . . 39
4.1.1 Description of the problem . . . 39
4.1.2 Expression for the rotational kernel . . . 39
4.1.3 RLS – Regularised Least Squares . . . 39
4.1.4 SOLA – Subtractive Optimally Localised Averages . . . 40
4.1.5 MOLA – Multiplicative Optimally Localised Averages . . . 41
4.1.6 Error bars and averaging kernels . . . 41
4.2 Rotation gradient inversion and angular momentum inversion . . . 42
4.3 Structural inversions . . . 42
4.3.1 Description of the problem . . . 42

4.3.2 c2, rho kernels . . . 43
4.3.3 Gamma1, rho kernels . . . 44
4.3.4 RLS . . . 44
4.3.5 SOLA . . . 45
4.3.6 Various error bars and kernels . . . 46
4.4 Mean density estimates . . . 46
4.4.1 SOLA . . . 47
4.4.2 Delta nu scaling law . . . 47
4.4.3 Kjeldsen et al's approach . . . 47
4.4.4 Various error bars and kernels . . . 48
4.4.5 Displayed quantities . . . 48
4.5 tau and delta f inversions . . . 49
4.6 Integration method . . . 49

5 Known bugs **50**

6 Copyright notices **50**
6.1 Source code for reading fortran binary files . . . 51
6.2 Source code for simplex algorithm . . . 51
6.3 Source code for a subset of the Dierckx package . . . 51
6.4 Supplementary notices . . . 51

Acknowledgements

This program was written by Daniel Reese and Sergei Zharkov, whose work was supported by the European Helio- and Asteroseismology Network (HELAS), a major international collaboration funded by the European Commission's Sixth Framework Programme. Daniel Reese has since then continued development of this code during various postdocs and is currently supported by the SPACEINN network, a major international collaboration funded by the European Commission's Seventh Framework Programme. A number of contributions by Gaël Buldgen have also been incorporated in this version of the code.

1 Getting started

1.1 Running the program

`InversionKit` runs under Java 6.0 or later versions. If Java is not installed on your computer, or is not sufficiently up-to-date, it can be downloaded from:

`http://www.java.com/en/`

JRE (Java Runtime Environment) allows you to run Java programs but not to compile your own. JDK (Java Development Kit) allows you to run and compile Java programs.

To run the program download the file `InversionKit.jar` from the following website:

`http://bison.ph.bham.ac.uk/~dreese/InversionKit/index.html`

then type the following command in a command window, in the directory that contains `InversionKit.jar`:

```
java -jar InversionKit.jar
```

If you are planning to do calculations involving large data and kernels sets, you may need to allocate a larger amount of memory to run the program. To allocate, for example, 500 MB of memory, use the following command:

```
java -Xmx500m -jar InversionKit.jar
```

Note: the option `-Xmx` is nonstandard and may change according to the release installed on your computer.

1.2 Using the program

Once `InversionKit` is running, the user has several options:

- read/write a stellar model and do basic manipulations on it
- read/calculate eigenmodes for a given model
- produce echelle diagrams with observed and theoretical frequencies
- invert observed frequencies or rotational splittings to find structural, rotational profiles, and other related profiles
- calculate "observed frequencies" or rotational splittings from target profiles
- estimate the stellar mean density, the acoustic radius, or an age estimator through SOLA inversions and through scaling laws
- combine the above options

The above operations are done in different tabs within the program. These tabs are:

- `Rotational inversion`, `Rotation gradient inversion`, `Angular momentum inversion`: these tabs allow the user to invert the rotation profile, its gradient, or the angular momentum profile, respectively, and also allows the user to load a target profile
- `Structural inversion`: this tab does structural inversions on pairs of structural profiles and allows the user to load target structural profiles.
- $\underline{\rho}$ `inversion`, τ **inversion**, δf **inversion**: these tabs allow the user to estimate the stellar mean density, acoustic radius, or age indicator (based on the small frequency separation) using a SOLA type structural inversion and different scaling laws
- `Stellar model`: this tab allows the user to load or generate a stellar model
- `Kernels`: this tab allows the user to calculate eigenfunctions, and associated rotational and structural kernels
- `Frequency data`: this tab allows the user to load/generate/edit observed frequencies and rotational splittings
- `Echelle diagram`: this tab produces an echelle diagram of of the observed and theoretical frequencies
- `Physical constants`: this tab allow the user to modify and save the internal values of the gravitational constant, the solar mass, and the solar radius

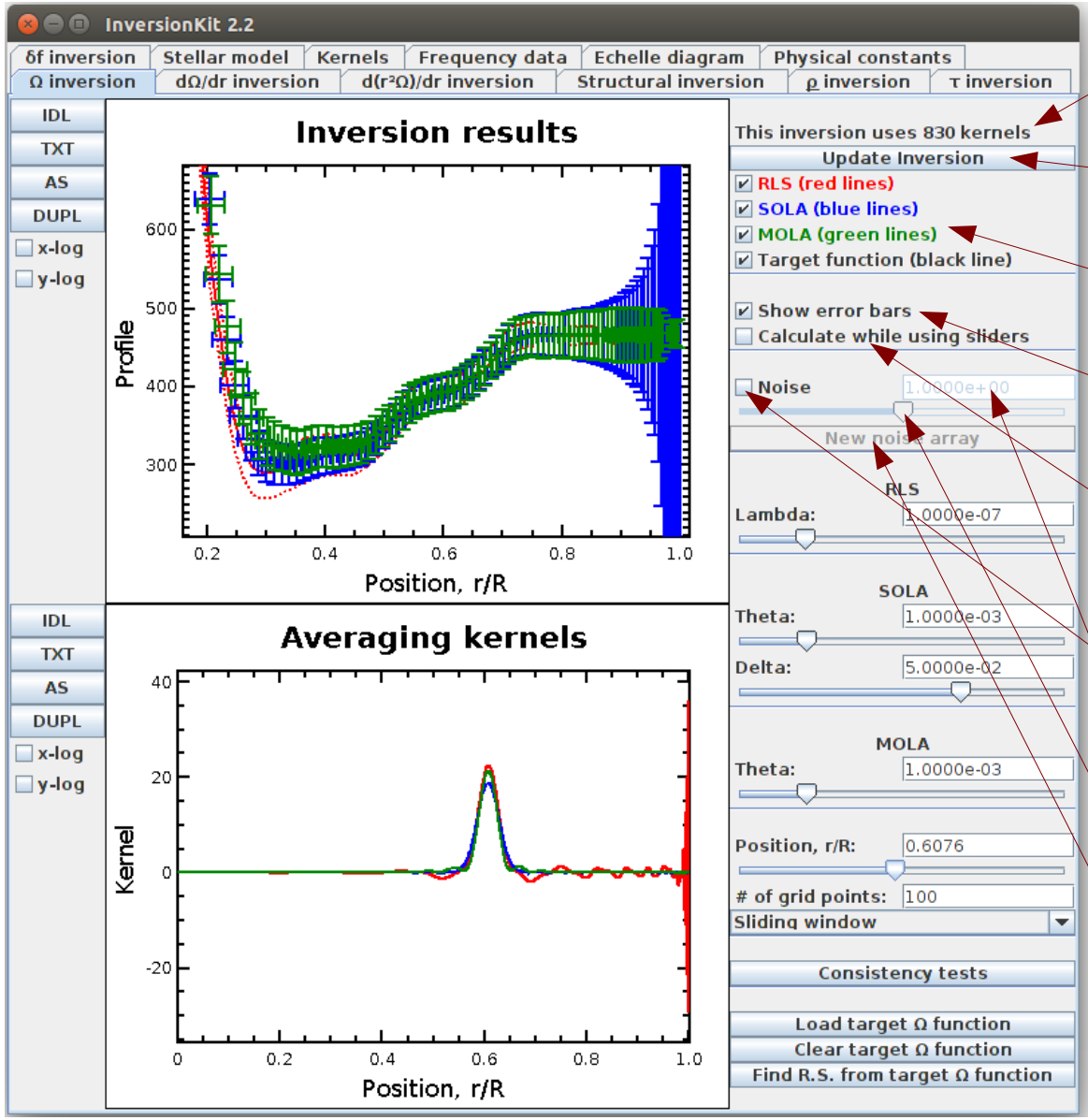
The next few pages show a set of screen captures along with a brief description of the various buttons and options which appear in the different tabs. The `Structural Inversion` tab is similar to the `Rotational Inversion` tab; therefore only extra features are described. Various other inversion tabs are not shown as they are quite similar to the ones shown, and are accordingly, self-explanatory. Some screen captures of pop-up windows, which appear, for instance, when plotting the eigenfunctions and kernels, are included. The actual appearance of these tabs and windows may vary from one platform to another depending on the Java installation. This section can therefore be viewed as a quick and easy guide for `InversionKit`. For more detailed information on the file formats and the different equations which intervene, we refer the reader to the following sections.

1.3 Preliminary remarks

Some of the operations can take some time. For example, calculating 50 eigenmodes and associated kernels can take typically 5s (for 2000 grid points). Doing a rotational inversion with 800 kernels takes typically 7.5s. When the program is calculating, it is best just to wait and let it finish what it is doing. Carrying out simplex calculations (see Frequency data tab) can be quite costly.

Also, unlike previous versions of `InversionKit`, most of the the check boxes, sliders and text fields in the `Rotation inversion`, `Structural inversion` and $\underline{\rho}$ `inversion` tabs will only partially update the inversion, i.e. if the model, the kernels, frequency shift data or rotational splittings have been modified, this will not be taken into account. To take these modifications into account, please use the `Update Inversion` buttons. Besides the `Update Inversion` buttons, the other buttons which cause the inversion to be fully updated are: the `Number of grid points` text field, the `Integration method` combo box (see Sect. 4.6), and the `Kernel type` combo box.

Finally, some of the more experimental features have not been fully documented due to lack of time. Further information may be found in articles such as Buldgen et al. (2015, A&A 574, 42) and Reese (2015, A&A 578, 37).



this says how many kernels are being used in the current inversion

this updates the inversion (using the latest values from the text fields below)

these determine which functions are shown

this determines whether or not to show the error bars from the various inversion

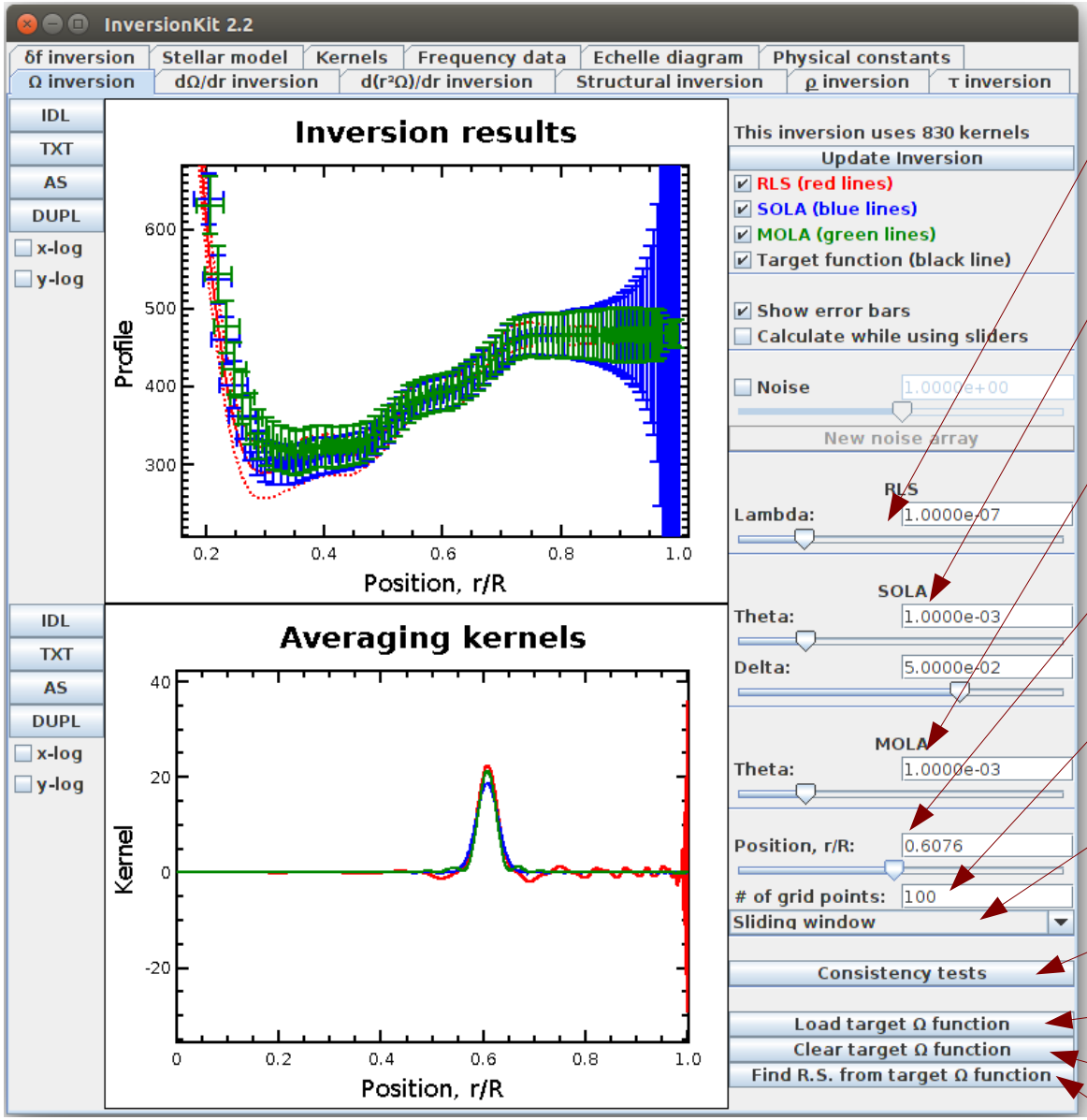
when selected, the program tries to calculate the inversion at each intermediate positions on the sliders below

when selected, artificial noise is added to the data before the inversion

text field which determines the amplitude of the noise

slider which determines the amplitude of the noise

this generates a new realisation of noise



text field and slider which determine the amount of regularisation in an RLS inversion (see Formula section)

text fields and sliders which determine the amount of regularisation and kernels widths in a SOLA inversion (see Formula section)

text field and slider which determine the widths of the averaging kernels in a MOLA inversion (see Formula section)

text field and slider which determine the target grid point for which to show the averaging kernels from the different inversions

text field which determines the number of grid points to be used in the inversions

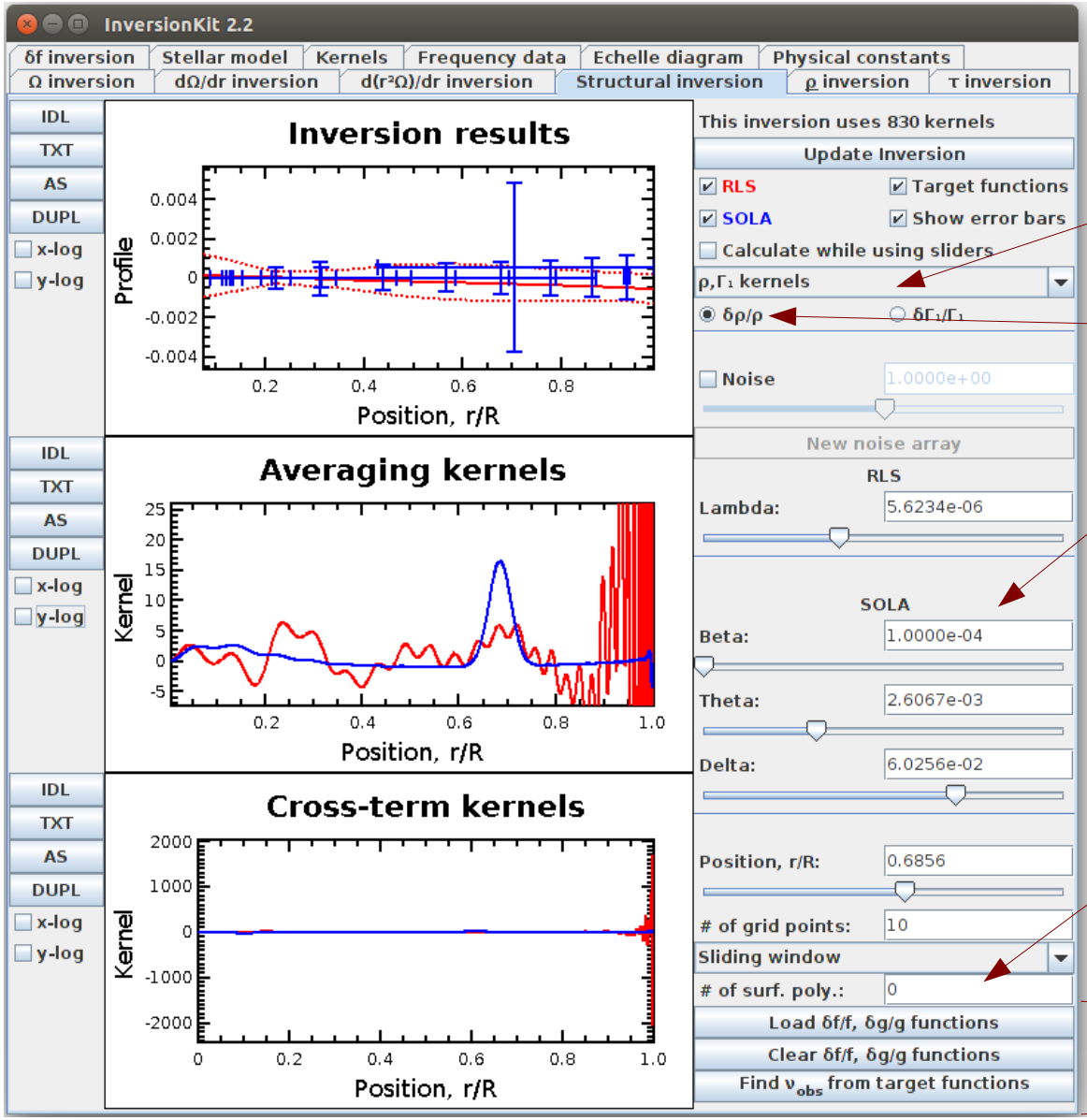
this determines whether to use a sliding window or Chebyshev polynomial approach (see Integration Method section)

(*deprecated*) tests whether the rotation profile increases

this loads a target rotation function (see File format section)

clears the current target function

this calculates rotational splittings from the target function



combo box which gives the choice between different types of structural kernels

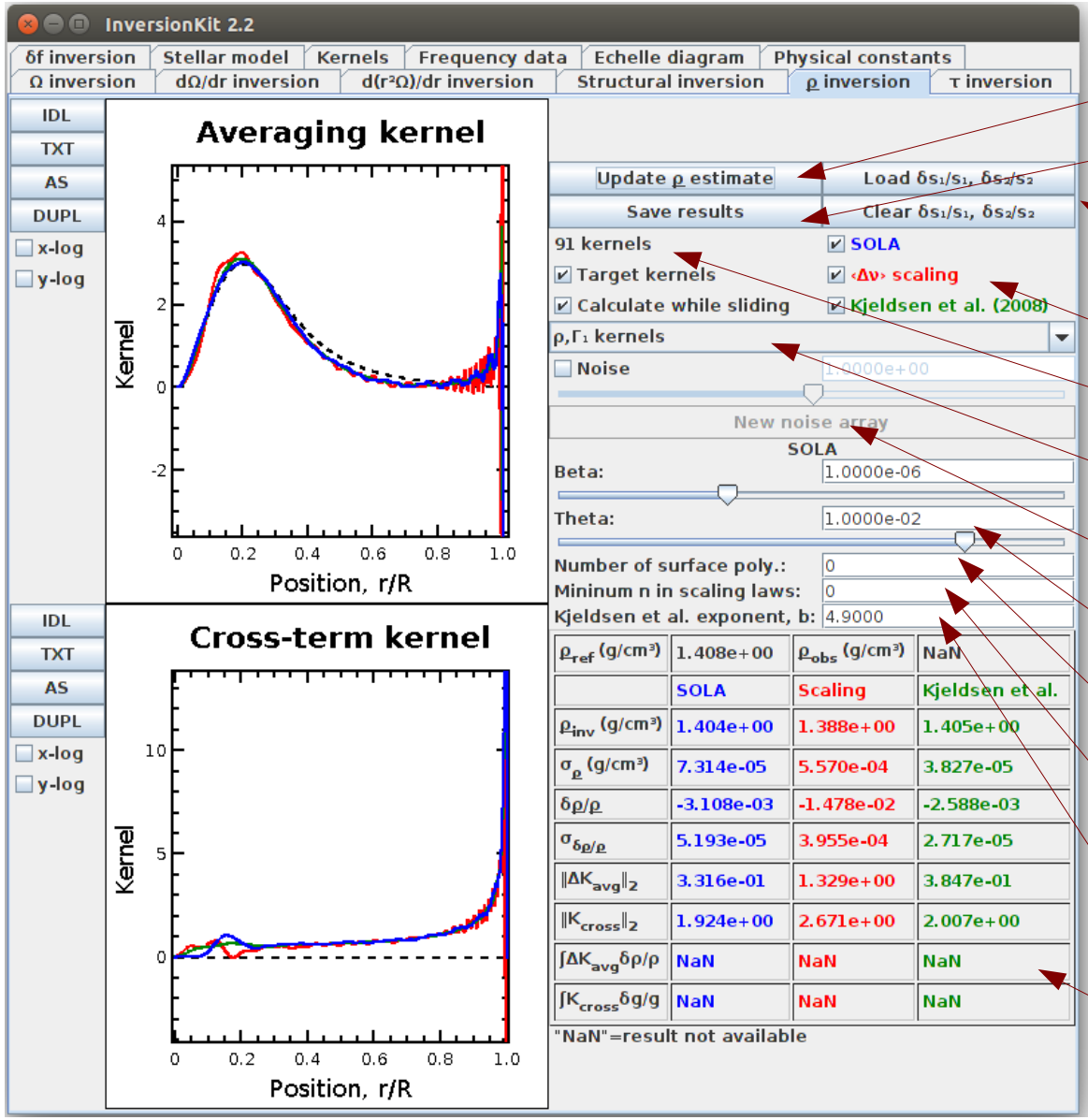
this determines which of two structural functions will be shown

text field and slider which determine the ratio between producing “nice” averaging kernels and reducing cross-term kernels (see Formula section)

these sliders and text fields can be adjusted independently for the two different structural inversions

text field which determines the number of legendre polynomials used in correcting for surface effects (the polynomial degrees go from 0 to this value minus 1)

these load or clear structural target functions, and find “observed” frequencies via the resultant theoretical frequency shifts



this updates the estimates

this saves the results to a file

these load/clear target functions (see last two rows of the table below)

choice of different estimators

number of kernels involved

choice of pair of structural functions

optional noise and controls

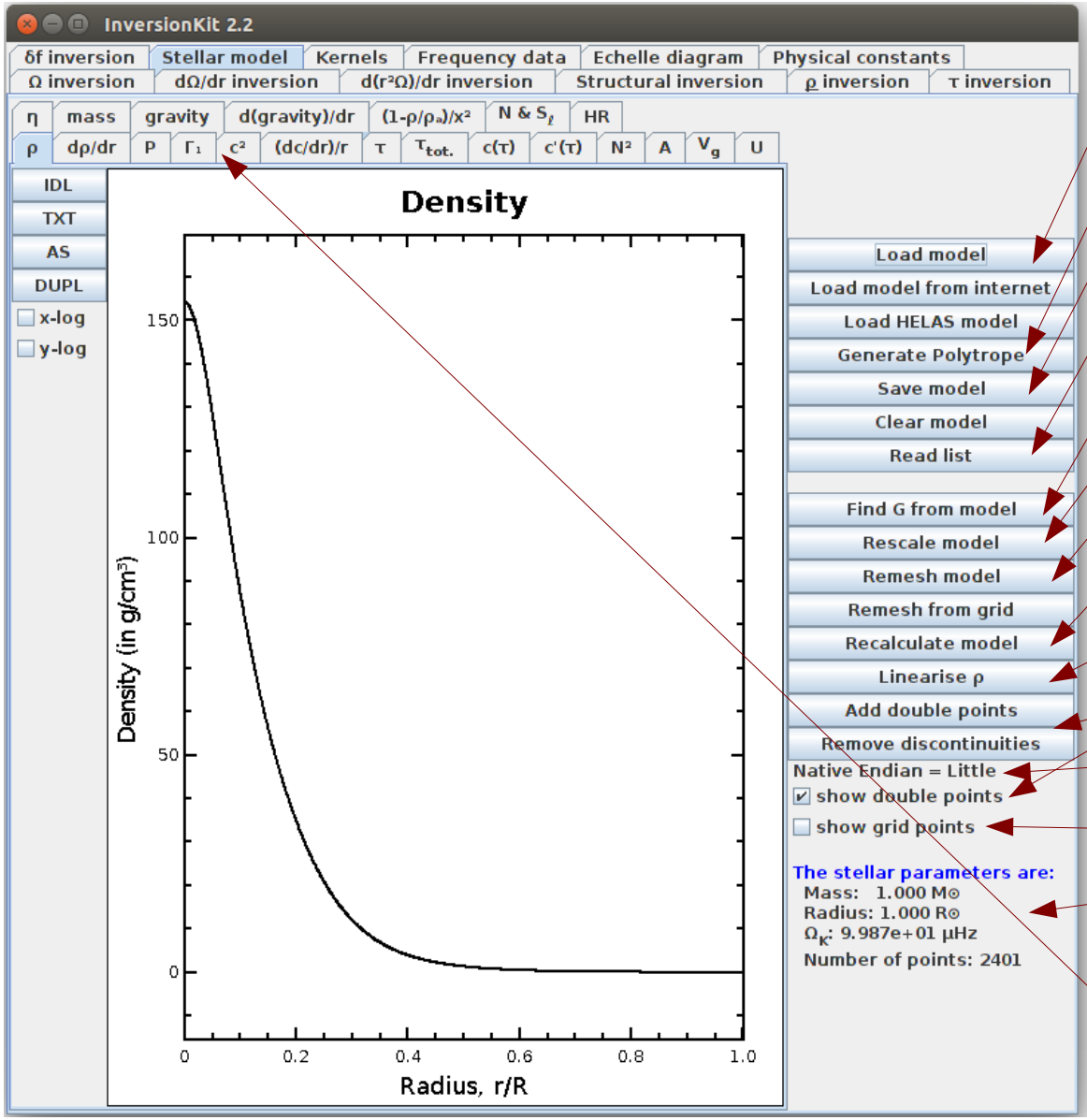
parameters for the SOLA inversion

number of surface terms in SOLA inversion

minimum radial order in scaling laws

exponent in Kjeldsen et al. (2008) recipe

colour coded table with the different results



load a stellar model from a file, a URL, or the CAUP/HELAS website

this generates a polytrope

save/clear the model

reads a list of models (for the HR diagram – cf. Format section)

finds G from hydrostatic equilibrium

homologously rescales model

finds a new mesh for the model

recalculates the model from the ρ and Γ_1 profiles

imposes linear behaviour on part of the ρ profile

add, remove, display double points

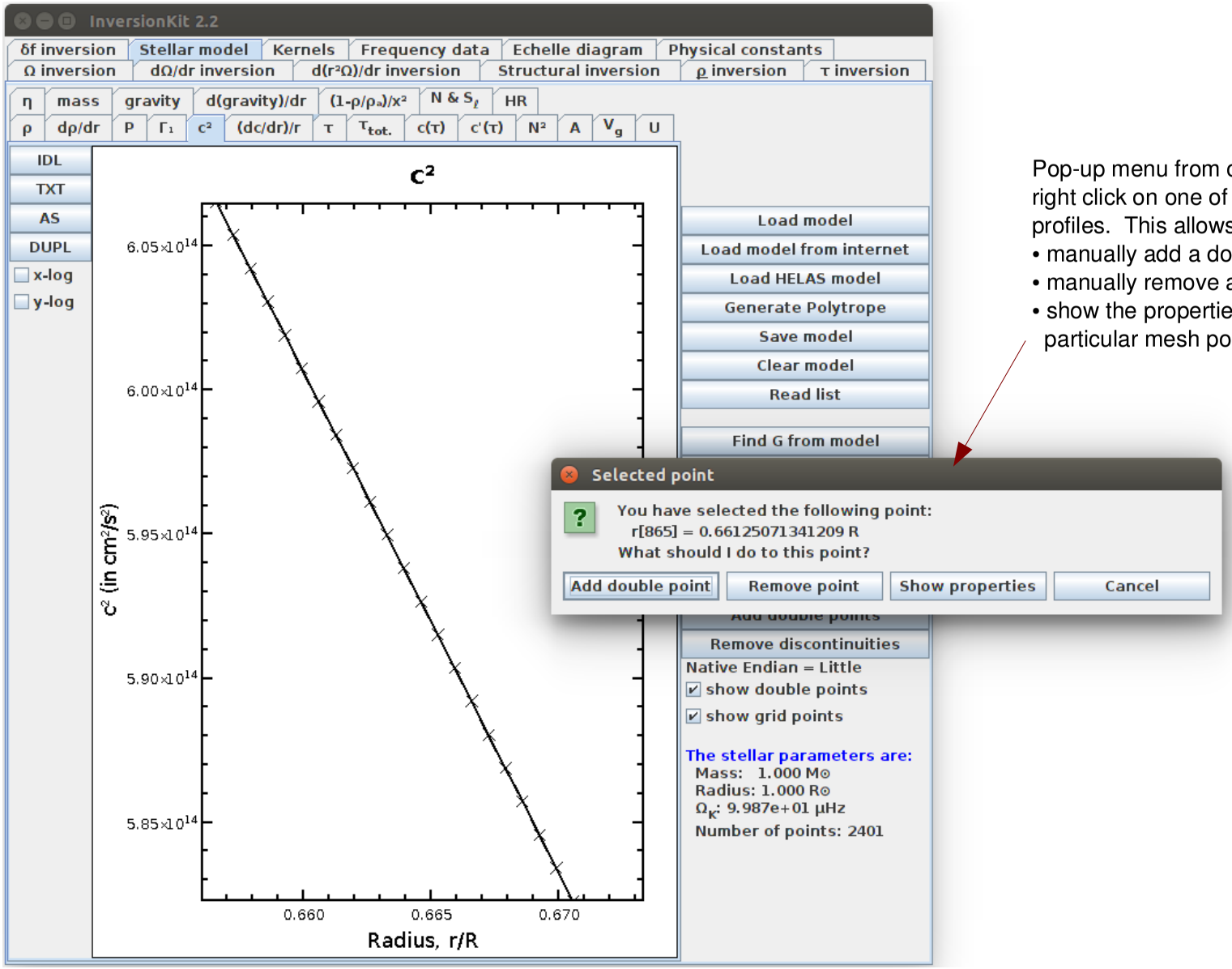
"endianness" of your computer

shows double/grid points

information on current model

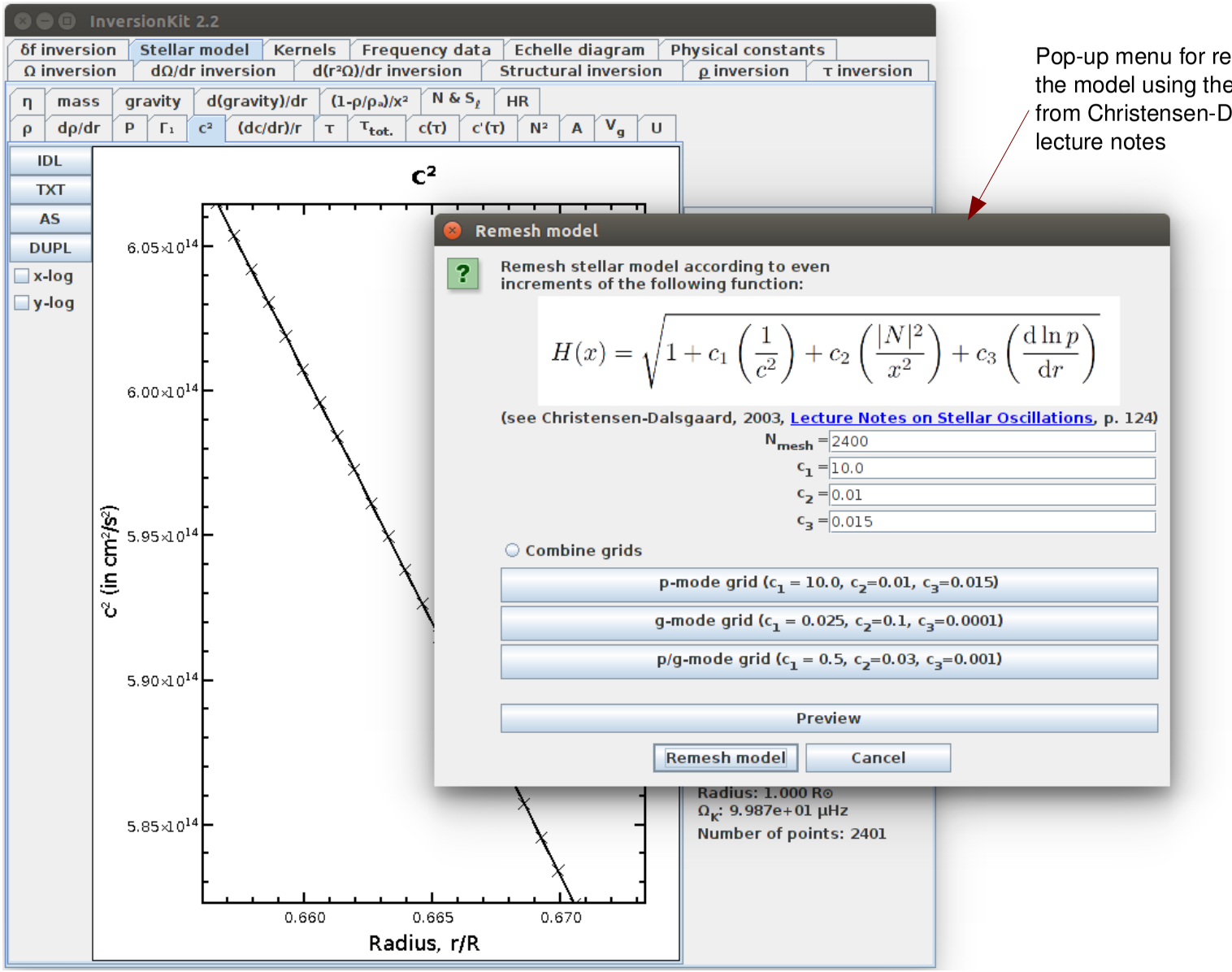
tabbed pane with different stellar structure profiles (N & S = Brunt-Väisälä and Lamb frequencies, HR = HR diagram).

- right click on the various profile and get a pop-up menu
- right click on the HR diagram to load a model from the list

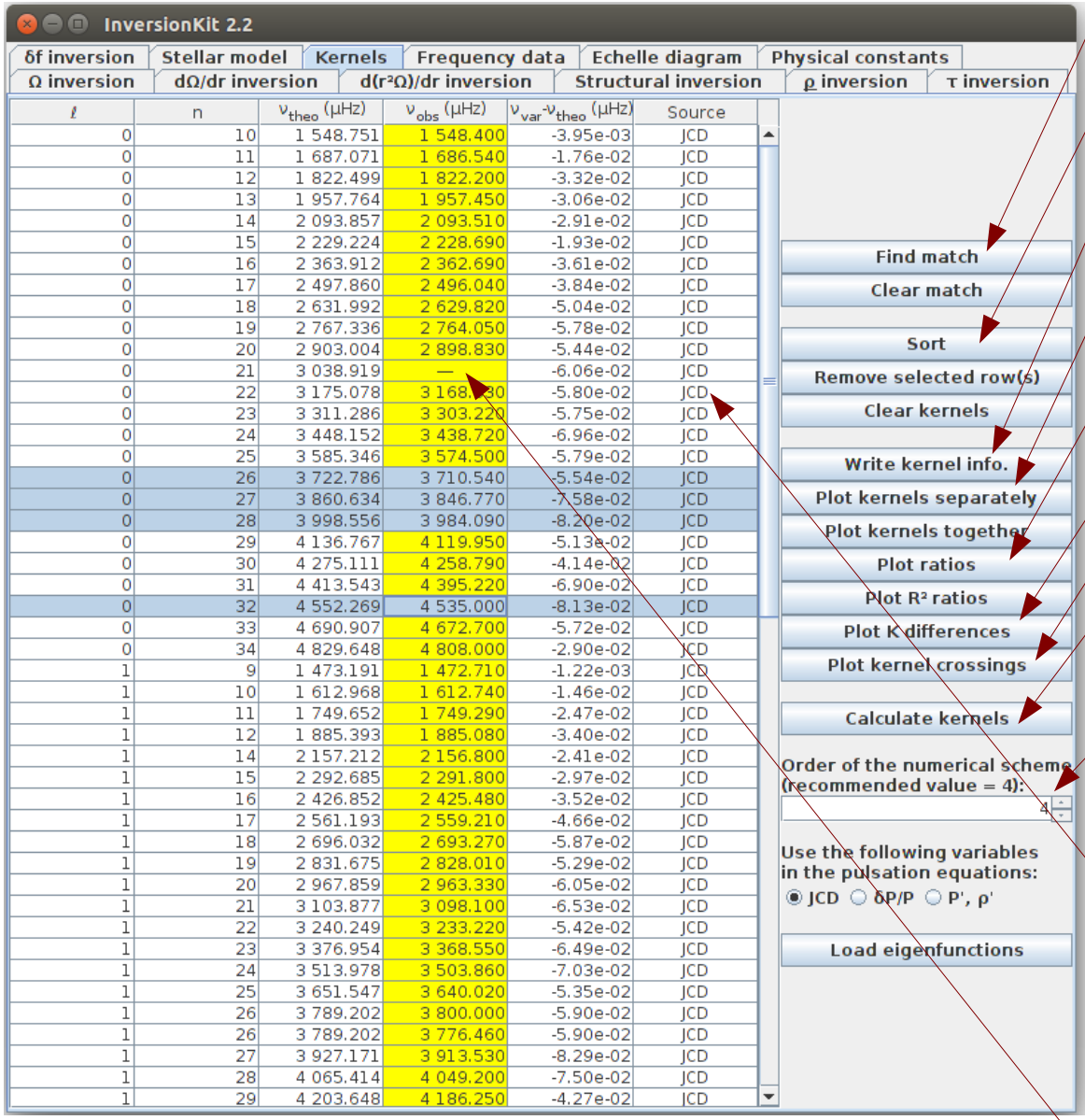


Pop-up menu from obtained via a right click on one of the structural profiles. This allows the user to

- manually add a double point
- manually remove a mesh point
- show the properties at a particular mesh point



Pop-up menu for remeshing the model using the formulas from Christensen-Dalsgaard's lecture notes



ℓ	n	ν_{theo} (μHz)	ν_{obs} (μHz)	$\nu_{var}-\nu_{theo}$ (μHz)	Source
0	10	1 548.751	1 548.400	-3.95e-03	JCD
0	11	1 687.071	1 686.540	-1.76e-02	JCD
0	12	1 822.499	1 822.200	-3.32e-02	JCD
0	13	1 957.764	1 957.450	-3.06e-02	JCD
0	14	2 093.857	2 093.510	-2.91e-02	JCD
0	15	2 229.224	2 228.690	-1.93e-02	JCD
0	16	2 363.912	2 362.690	-3.61e-02	JCD
0	17	2 497.860	2 496.040	-3.84e-02	JCD
0	18	2 631.992	2 629.820	-5.04e-02	JCD
0	19	2 767.336	2 764.050	-5.78e-02	JCD
0	20	2 903.004	2 898.830	-5.44e-02	JCD
0	21	3 038.919	—	-6.06e-02	JCD
0	22	3 175.078	3 168.[illegible]30	-5.80e-02	JCD
0	23	3 311.286	3 303.220	-5.75e-02	JCD
0	24	3 448.152	3 438.720	-6.96e-02	JCD
0	25	3 585.346	3 574.500	-5.79e-02	JCD
0	26	3 722.786	3 710.540	-5.54e-02	JCD
0	27	3 860.634	3 846.770	-7.58e-02	JCD
0	28	3 998.556	3 984.090	-8.20e-02	JCD
0	29	4 136.767	4 119.950	-5.13e-02	JCD
0	30	4 275.111	4 258.790	-4.14e-02	JCD
0	31	4 413.543	4 395.220	-6.90e-02	JCD
0	32	4 552.269	4 535.000	-8.13e-02	JCD
0	33	4 690.907	4 672.700	-5.72e-02	JCD
0	34	4 829.648	4 808.000	-2.90e-02	JCD
1	9	1 473.191	1 472.710	-1.22e-03	JCD
1	10	1 612.968	1 612.740	-1.46e-02	JCD
1	11	1 749.652	1 749.290	-2.47e-02	JCD
1	12	1 885.393	1 885.080	-3.40e-02	JCD
1	14	2 157.212	2 156.800	-2.41e-02	JCD
1	15	2 292.685	2 291.800	-2.97e-02	JCD
1	16	2 426.852	2 425.480	-3.52e-02	JCD
1	17	2 561.193	2 559.210	-4.66e-02	JCD
1	18	2 696.032	2 693.270	-5.87e-02	JCD
1	19	2 831.675	2 828.010	-5.29e-02	JCD
1	20	2 967.859	2 963.330	-6.05e-02	JCD
1	21	3 103.877	3 098.100	-6.53e-02	JCD
1	22	3 240.249	3 233.220	-5.42e-02	JCD
1	23	3 376.954	3 368.550	-6.49e-02	JCD
1	24	3 513.978	3 503.860	-7.03e-02	JCD
1	25	3 651.547	3 640.020	-5.35e-02	JCD
1	26	3 789.202	3 800.000	-5.90e-02	JCD
1	26	3 789.202	3 776.460	-5.90e-02	JCD
1	27	3 927.171	3 913.530	-8.29e-02	JCD
1	28	4 065.414	4 049.200	-7.50e-02	JCD
1	29	4 203.648	4 186.250	-4.27e-02	JCD

find/clear the match between observed and theoretical modes

sort/remove/clear theoretical pulsation modes

write file with information on theoretical modes

plot highlighted theoretical modes together/separately (cf. next pages)

plot ratios of integrated rotation kernels for highlighted theoretical modes (cf. Reese, 2015, A&A 578, 37)

plot integrated kernel weighted differences

plot kernel crossings as a function of r/R

calculate theoretical modes closest to unmatched observed modes

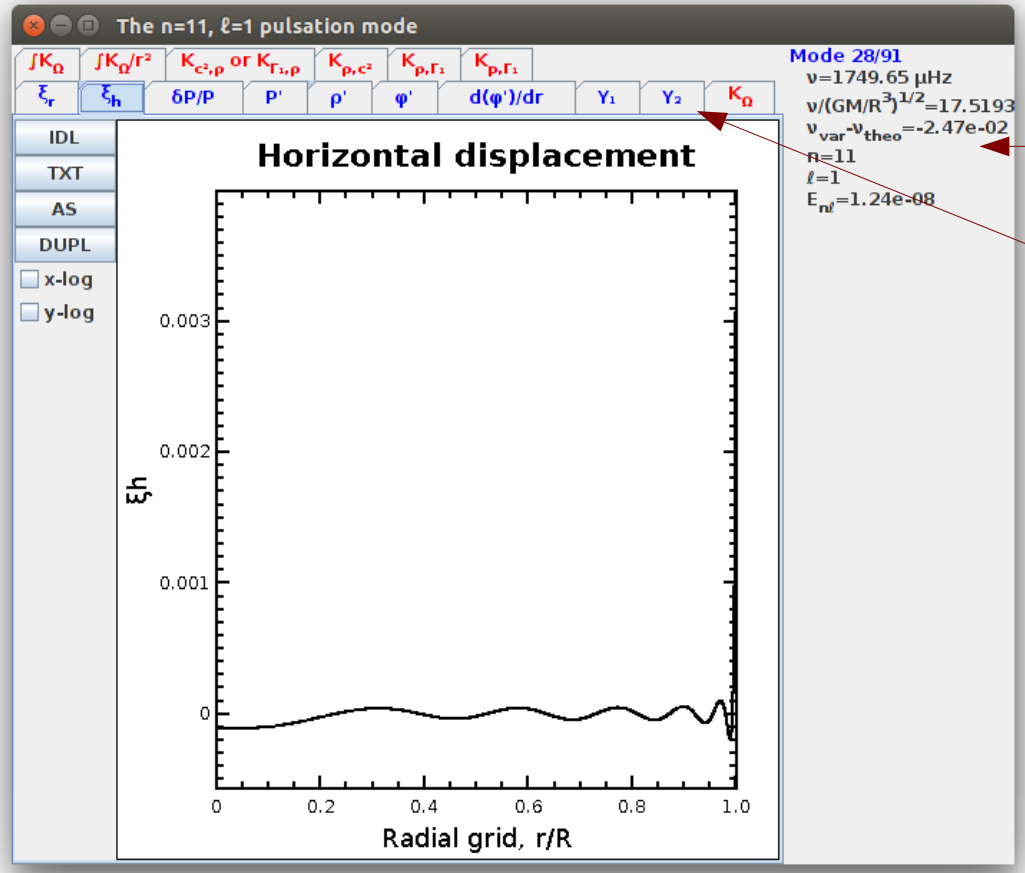
numerical order and system of equations when calculating theoretical pulsation modes

load theoretical modes from a file

table with , n, theoretical and observed frequencies, error on variational frequencies and source (i.e. from a file or calculated from a specified system of equations)

- the rows in this table can be highlighted and don't need to be contiguous
- observed modes are in yellow

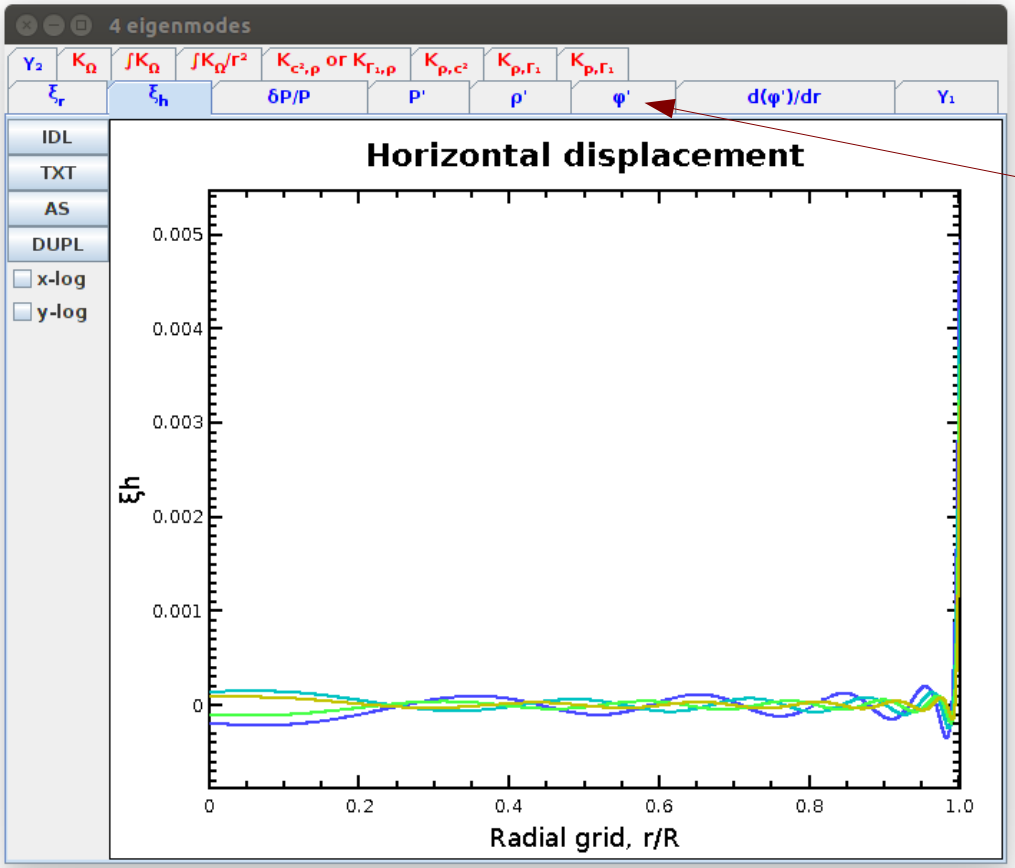
unmatched theoretical mode



A theoretical mode plotted separately

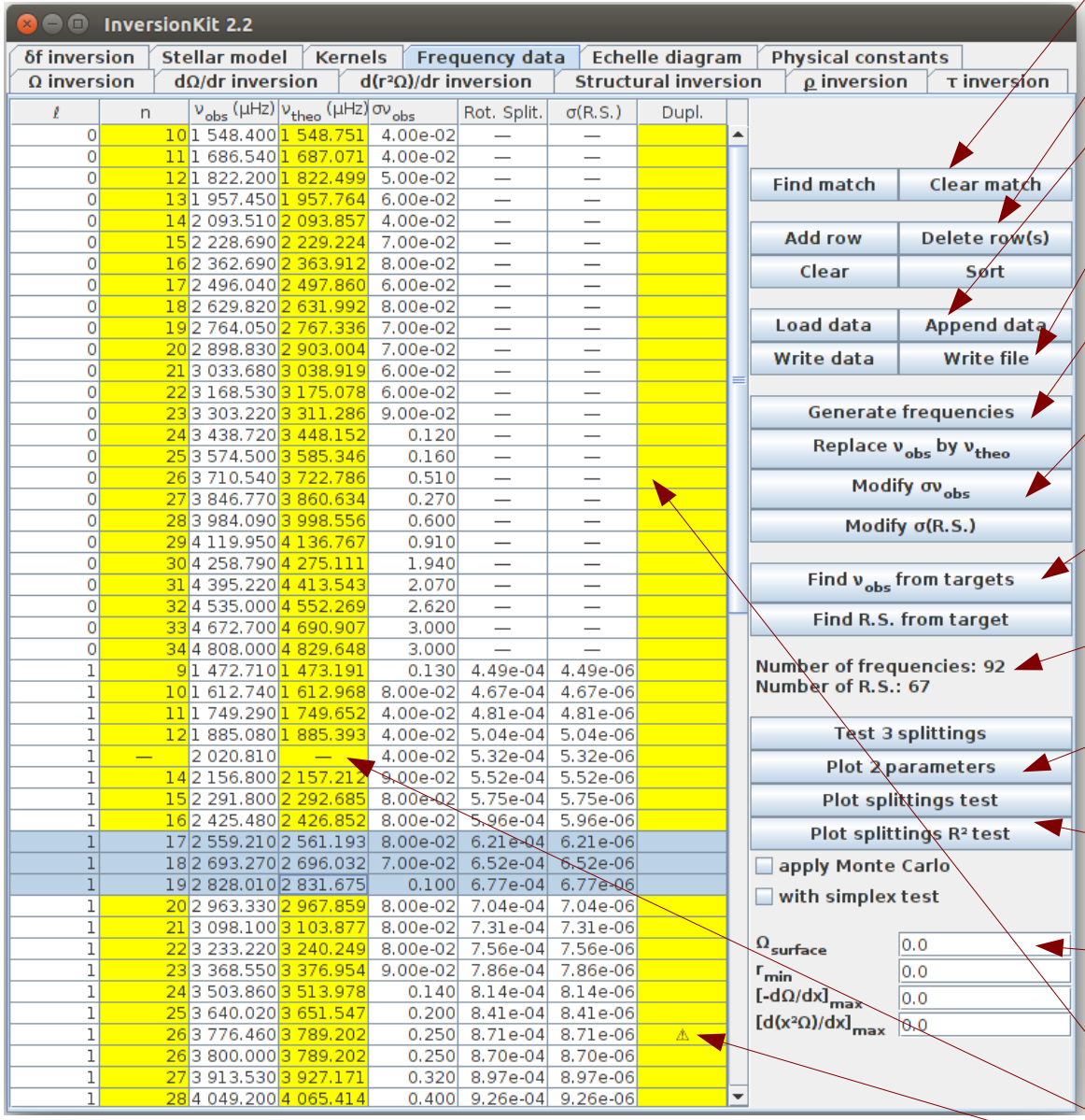
various information on pulsation mode

plots with eigenfunctions (blue tabs) and kernels (red tabs)



Theoretical modes plotted together

plots with eigenfunctions (blue tabs) and kernels (red tabs)



find/clear the match between observed and theoretical modes

add/remove/clear/sort obs. modes

load/append/write data from/to file

write file with observed and theoretical information on modes

generate grid of uniformly spaced frequencies for a set of values

set all error bars on observed frequencies/rotational splittings to specified value

find "observed" frequencies/splittings from target profiles from Structure and Rotation Inversion tabs

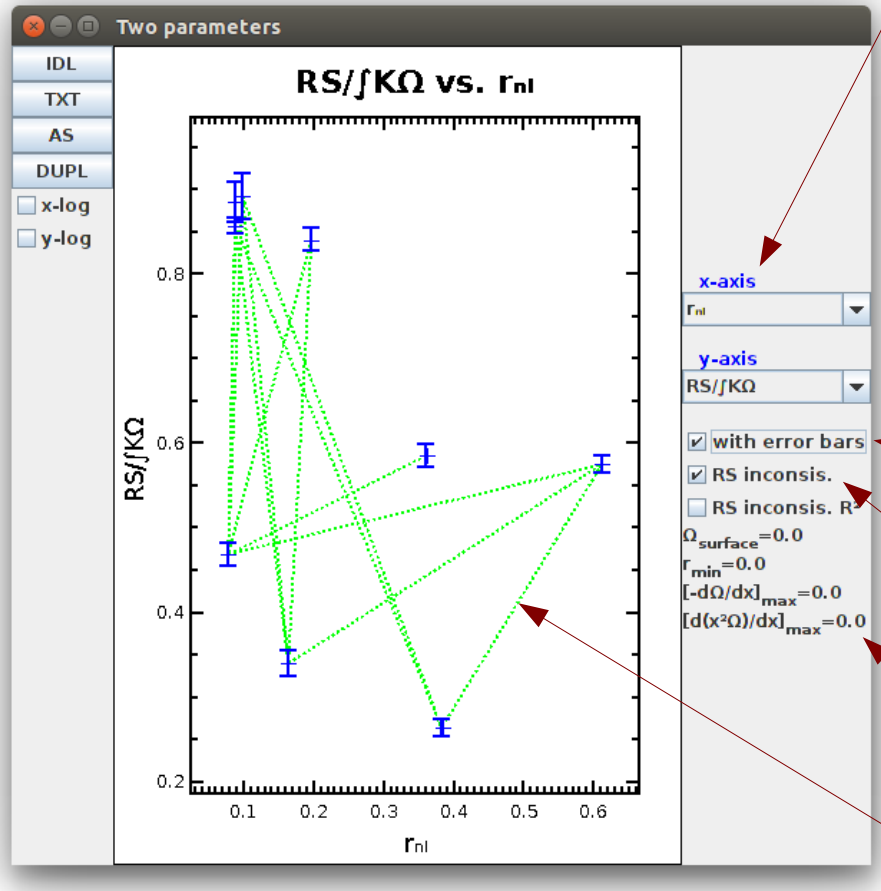
the number of observed frequencies/ rotational splittings with error bars

plots one of the mode parameters as a function of another (cf. next page)

plots lower and upper bounds on the splittings based on assumptions on the rotation profile (Reese, 2015, A&A 578, 37)

parameters which intervene in the above bounds on the splittings

editable table with values, observed frequencies, splittings and error bars
- yellow columns cannot be edited
- unmatched observed mode
- observed mode matched to duplicate theoretical mode



parameters to be used when making the plot:

- ν(obs): observed frequency
- ν(theo): theoretical frequency
- Enl: mode inertia
- RS: rotational splitting
- RS/∫KΩ: splitting divided by integral of rotation kernel
- RS/∫(KΩ/r²): splitting divided by integral of rotation kernel over r²
- rnl: rotation kernel weighted average position
- r'nl: rotation kernel over r² weighted average position

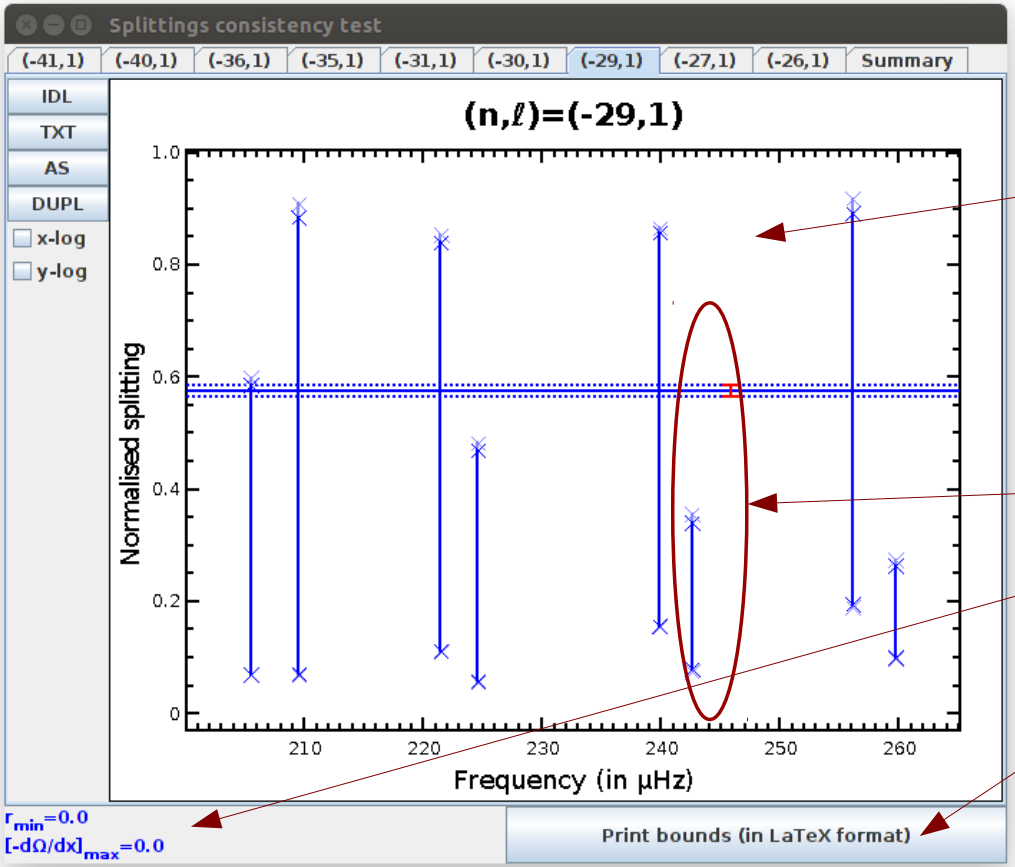
A number of these parameters intervene in the rotational splittings consistency tests described in Reese, submitted

show error bars where applicable

connect modes with “inconsistent” rotational splittings when assuming a decreasing rotation profile, or one which satisfies Rayleigh's criterion

parameters which intervene in splittings consistency tests

modes with inconsistent rotational splittings (green for decreasing rotation profile assumption, red for Rayleigh's criterion)

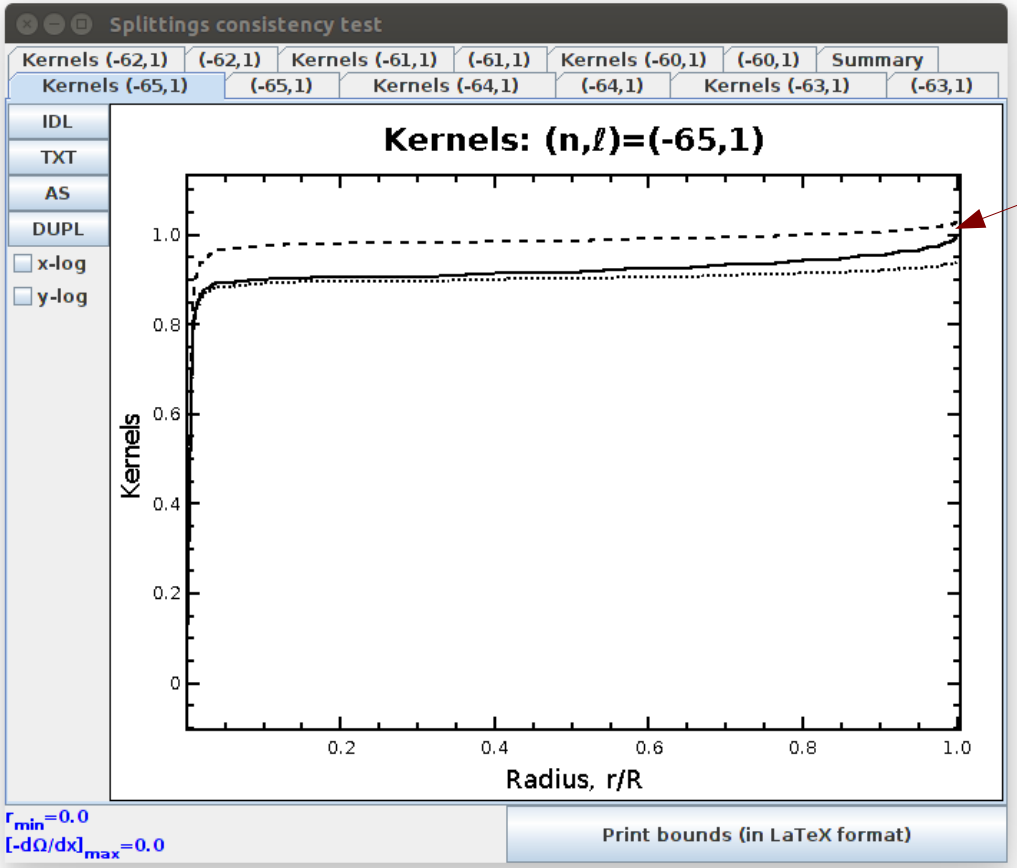


plots which show lower and upper bounds on a given rotational splitting based on assumptions on the rotation profile (in this case, that the rotation profile decreases) – see Reese, 2015, A&A 578, 37. The plots show the bounds and original splittings as a function of the mode frequencies
•blue = bounds from other modes
•red = original mode
Note: the light blue extensions correspond to the 1σ error bars

example of inconsistent bounds

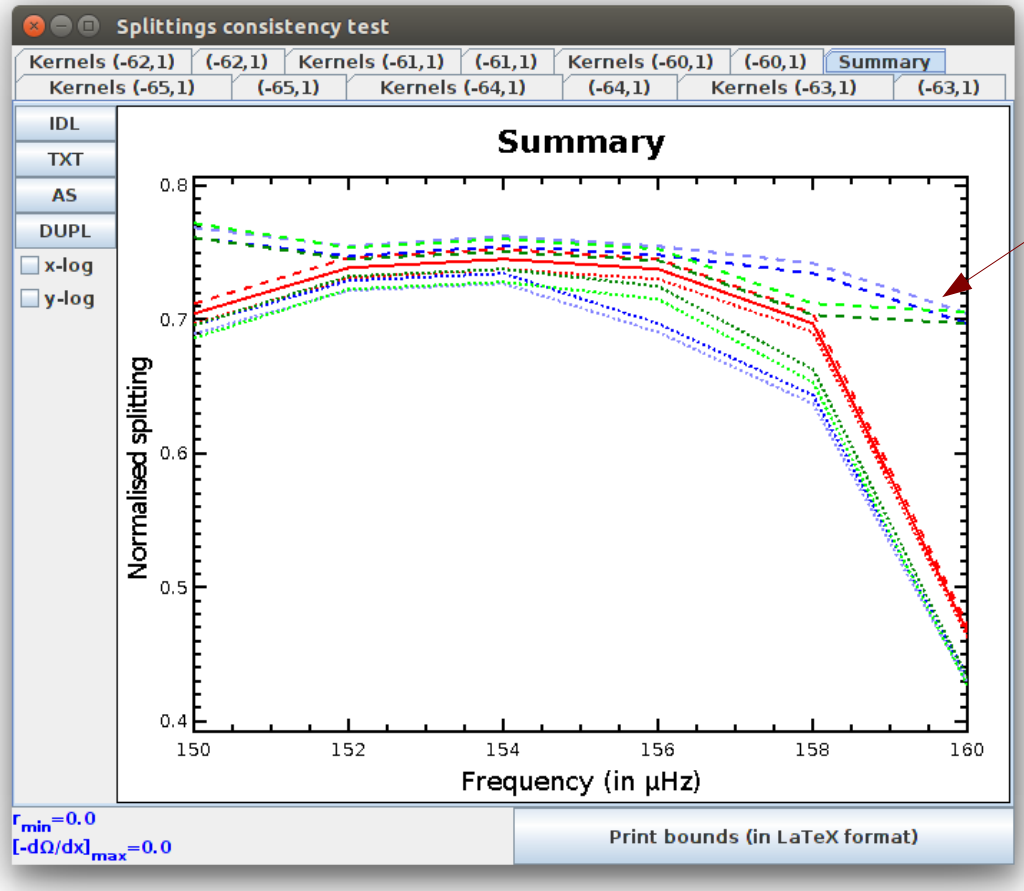
parameters which were used when calculating lower and upper bounds

print a and b coefficients which intervene when finding bounds on rotational splittings (see Reese, 2015, A&A 578, 37)



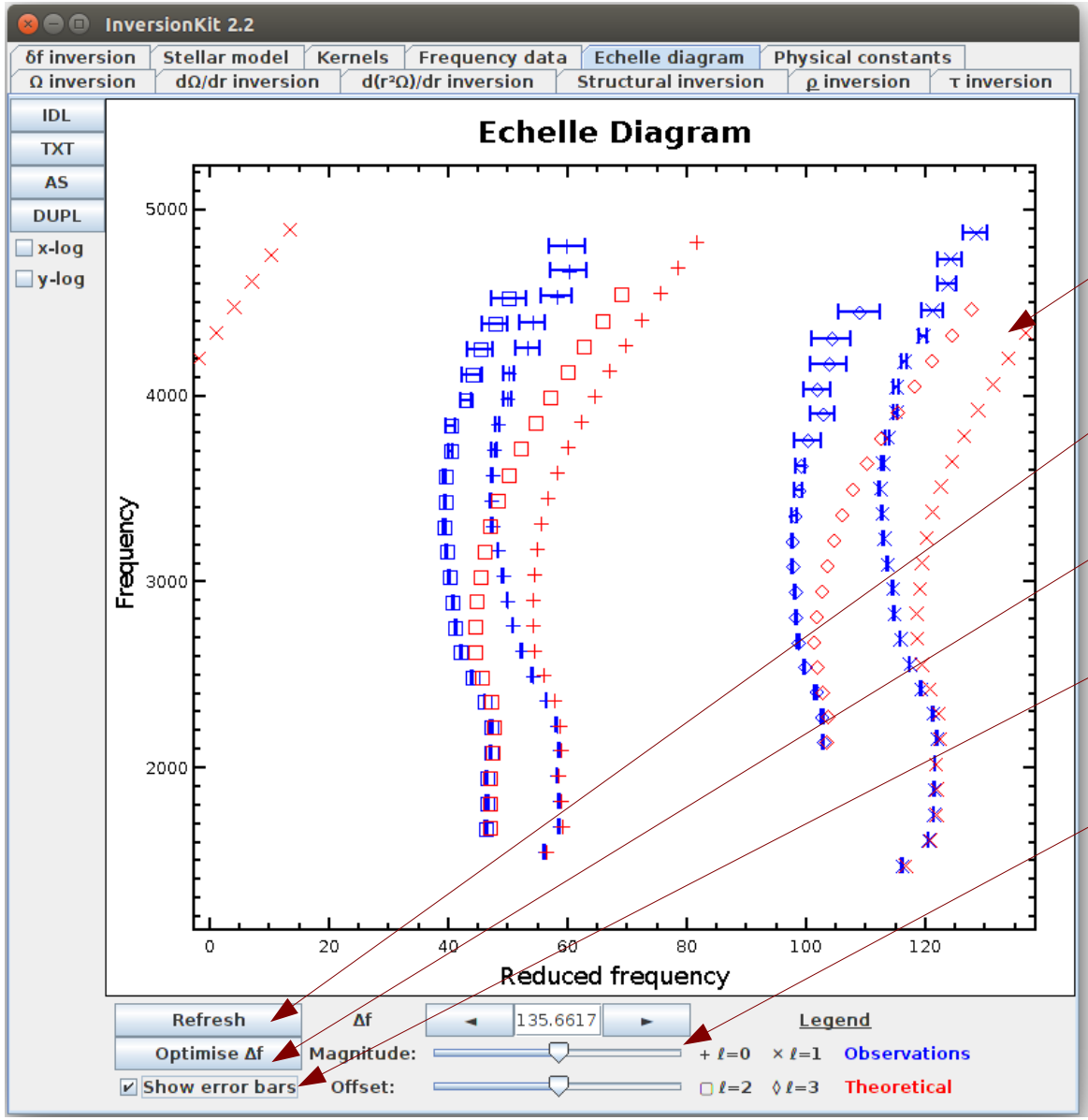
combined kernel based on simplex method

- solid: kernel of mode being investigated
- dashed: combined kernel of upper bound
- dotted: combined kernel of lower bound



summary of upper and lower bounds obtained from different inequalities based on Reese (2015, A&A 578, 37):

- red: original observations
- blue: pairs of modes
- green: simplex combination of modes
- light colours: 1σ error bars
- dashed: upper bounds
- dotted lower bounds



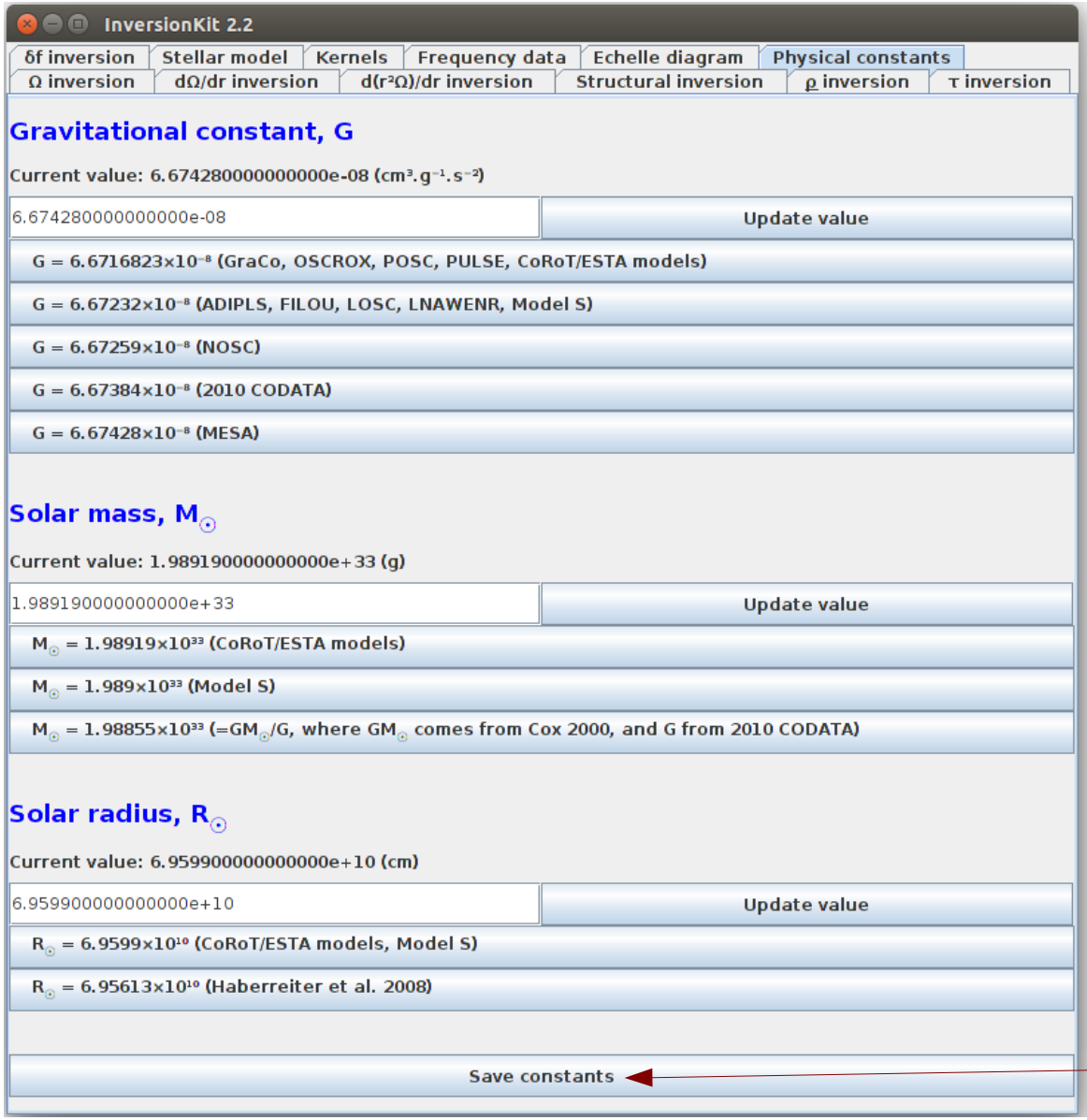
echelle diagram with observed frequencies in blue and theoretical frequencies in red

this updates the echelle diagram with new frequencies

this calculates the large frequency separation through a least-squares approach

shows error bars on observed frequencies

various controls which adjust the large frequency separation and the offset in the echelle diagram



this text field, and these buttons can be used to adjust the value of the gravitational constant in InversionKit (which intervenes in the pulsation calculations)

this text field, and these buttons can be used to adjust the value of the solar mass, in InversionKit

this text field, and these buttons can be used to adjust the value of the solar radius, in InversionKit

this saves the values of the above constants for future sessions with InversionKit

2 File formats

2.1 Stellar models

`InversionKit` accepts the following file formats for stellar models:

1. **AMDL**: FORTRAN binary files generated by ASTEC
2. **FAMDL**: text (or ascii) version of the AMDL files generated by ASTEC
3. **FGONG**: exchange format under the GONG model comparison scheme, and official output format from ASTEC
4. **CESAM**: .osc text files generated by CESAM
5. **MOD (binary)**: FORTRAN binary file from CLES
6. **MOD (ascii)**: text file from CLES
7. **CLES**: another text file format from CLES
8. **LOSC**: a text file from LOSC but in which the model is embedded

`InversionKit` is also able to write files in the AMDL, FAMDL, MOD (binary) and MOD (ascii) formats. The choice of the format (for reading or saving a model) is given by a menu in the `Model Page` tab (see previous section). `InversionKit` will attempt to distinguish between models generated by CESAM2k and earlier versions of CESAM by searching for "CESAM2k" in the header. If need be, `MODCONV` can be used to convert models from one format to another. This tool is available at:

`http://www.astro.up.pt/corot/ntools/modconv/`

A description of the first four file formats can be found at:

`http://www.astro.up.pt/corot/ntools/docs/CoRoT_ESTA_Files.pdf`

and within the instructions to the ADIPLS pulsation code:

`http://www.phys.au.dk/~jcd/adipack.n/notes/adiab_prog.ps.gz`

The last four file formats will be briefly described in the following sections. This will then be followed by a table which summarises the different structural variables available in the various formats, some useful formulas which relate these variables, and finally a brief description of the format of the file with a list of models for the HR diagram.

2.1.1 MOD (binary/ascii)

These contain a 5 line header (made up of 80 characters in the binary version), followed by a line which contains either 3 or 4 parameters. These are:

- the number of grid points
- the stellar radius (in cm)

- the stellar mass (in g)
- if present, the gravitational constant (in cgs)

This is then followed by a section with 6 or 7 columns (in the ascii version) or a list of entries with 5 or 6 values (in the FORTRAN binary version) which contain the following quantities:

- index of grid point (only in the ascii version)
- the radial coordinate (in cm)
- the cumulative mass (in g)
- the pressure (in cgs)
- the density (in g/cm^3)
- the Γ_1 profile
- if present, an unidentified quantity (`InversionKit` does not make use of this column)

2.1.2 CLES format

These contain a header which ends with the expression "`%%beginoscdata`". This is followed by a line which contains the following parameters:

- the stellar radius (in cm)
- the stellar mass (in g)
- the gravitational constant (in cgs)

This is followed by a line with the number of domains (in case there are double points), n_{domains}, as well as a section made up of n_{domains} lines which gives information on these domains. The following line contains the number of grid points. This is then followed by a section with 6 columns which contain the following quantities:

- the radial coordinate (in cm)
- the quantity m/r^3 (in g/cm^3)
- the pressure (in cgs)
- the density (in g/cm^3)
- the Γ_1 profile
- the quantity $-\frac{A}{r^2}$ (see Table 1)

2.1.3 LOSC format

These contain a 5 line header which is skipped. This is followed by a line with the following parameters:

- the number of grid points
- the stellar radius (in cm)
- the stellar mass (in g)
- the gravitational constant (in g)

The next three lines contain a header with column names and a list of various quantities related to the pulsation mode contained with the file. This is followed by a section with many columns. The first six columns are:

- the radial coordinate normalised by the radius
- the quantity $\eta = \frac{R^3 m}{M r^3}$
- the quantity $\frac{R}{GM}\frac{P}{\rho}$
- a normalised density, $\frac{4\pi R^3}{M}\rho$
- the Γ_1 profile
- the quantity $-\frac{A}{r^2}$ (see Table 1)

2.1.4 Comparison of different model formats

Table 1 gives the different variables from the different models and allows an easy comparison between them.

2.1.5 Useful formulas

The following formulas can be useful for finding one structural variable from other variables:

$$c^2 = \frac{\Gamma_1 P}{\rho} \tag{1}$$

$$g = \frac{Gm}{r^2} \tag{2}$$

$$\frac{\mathrm{d}g}{\mathrm{d}r} = 4\pi G\rho - \frac{2Gm}{r^3} \tag{3}$$

$$\frac{R^3}{M}\rho = \frac{\eta U}{4\pi} \tag{4}$$

$$\frac{R^4}{GM}P = \frac{x^2\eta^2 U}{4\pi V_g \Gamma_1} \tag{5}$$

$$A = -r\left(\frac{g}{c^2} + \frac{1}{\rho}\frac{\mathrm{d}\rho}{\mathrm{d}r}\right) \tag{6}$$

$$\frac{R^3}{GM}N^2 = \frac{R^3}{GM}\frac{g}{r}A = \eta A \tag{7}$$

$$\langle\rho\rangle = \frac{m}{\frac{4}{3}\pi r^3} \tag{8}$$

Table 1: Table which lists variables from different model formats. The first column gives the variables which are stored in `InversionKit`.

Variable	(F)AMDL	FGONG	CESAM	MOD	LOSC	CLES
$x = \frac{r}{R}$	$\frac{r}{R}$	r	r	r	$\frac{r}{R}$	r
Γ_1	Γ_1	Γ_1	Γ_1	Γ_1	Γ_1	Γ_1
$\frac{R^3}{M}\rho$	–	ρ	ρ	ρ	$\frac{4\pi R^3}{M}\rho$	ρ
$\frac{R^4}{M}\frac{\mathrm{d}\rho}{\mathrm{d}r}$	–	–	–	–	–	–
$\frac{R^4}{GM^2}P$	–	P	P	P	$\frac{R}{GM}\frac{P}{\rho}$	P
$\frac{R}{GM}c^2$	–	–	–	–	–	–
$\frac{R^3}{GM}N^2$	–	–	–	–	–	–
$\frac{m}{M}$	–	$\ln\left(\frac{m}{M}\right)$	$\ln\left(\frac{m}{M}\right)$	m	–	–
$\frac{R^2}{GM}g$	–	–	–	–	–	–
$\frac{R^2}{GM}\frac{\mathrm{d}g}{\mathrm{d}r}$	–	–	–	–	–	–
$\frac{R^2}{r^2}\left(1 - \frac{\rho}{\langle\rho\rangle}\right)$	–	–	–	–	–	–
$U = \frac{4\pi\rho r^3}{m}$	U	–	–	–	–	–
$A = \frac{1}{\Gamma_1}\frac{\mathrm{d}\ln P}{\mathrm{d}\ln r} - \frac{\mathrm{d}\ln\rho}{\mathrm{d}\ln r} = \frac{rN^2}{g}$	A	A	A	–	$-\frac{A}{r^2}$	$-\frac{A}{r^2}$
$V_g = -\frac{1}{\Gamma_1}\frac{\mathrm{d}\ln P}{\mathrm{d}\ln r} = \frac{Gm\rho}{\Gamma_1 pr}$	V_g	–	–	–	–	–
$\eta = \frac{R^3}{M}\frac{m}{r^3} = \frac{R^3}{GM}\frac{g}{r}$	η	–	–	–	η	$\frac{m}{r^3}$

2.1.6 File format for the list of models for the HR diagram

The file which contains a list models for the HR diagram must contain the following three columns:

- path to the file with the model
- effective temperature of the model
- luminosity of the model

Furthermore, on a given line, anything following the "#" symbol is treated as a comment and ignored.

The HR diagram is useful because if the user clicks with the mouse's right button next to a symbol which represents one of the models, `InversionKit` will propose to read that particular model, using the format specified on the right-hand side menu.

2.2 Eigenmodes

From version 2.1 onwards, `InversionKit` is able to calculate its own eigenmodes as will be described in the following section. However, it can also read eigenfunctions from external files. `InversionKit` accepts eigenmodes in one of the following formats:

- **AMDE**: FORTRAN binary files produced by ADIPLS
- **"FAMDE"**: text version of the AMDE format
- **FILOU**: text files produced by FILOU

2.2.1 The AMDE format

The AMDE format is a FORTRAN binary output file from ADIPLS. `InversionKit` accepts AMDE files produced with the options `nfmode=1`, `nfmode=2` or `nfmode=3`. For more details on this format, please consult the instructions to the ADIPLS code:

`http://www.phys.au.dk/~jcd/adipack.n/notes/adiab_prog.ps.gz`

2.2.2 The FAMDE format

The FAMDE format is a text version of the AMDE format produced by ADIPLS (for all values of `nfmode`) plus an additional header as described in the 1996 inversion workbench description. At the end of the header is a succession of 3 {`CTRL+L, CTRL+J`} in a row, which separates it from the rest of the file. `InversionKit` skips the header and reads data which comes after. The format obeys the following rules:

- Files are divided into different sections. These are given by the following lists, as indicated by the semi-colons:
  - **nfmode = 1**: `gp` array; `nr`; `x` and `y` arrays; `gp` array; `nr`; `x` and `y` arrays; etc.
  - **nfmode = 2**: `nr`; `x` array; `gp` array; `y` array; `gp` array; `y` array; etc.
  - **nfmode = 3**: `nr`; `x` array; `gp` array; `y` array; `gp` array; `y` array; etc.
- where:
  - `nr` = radial resolution
  - `x` = radial grid (size: `nr`)
  - `gp` = global parameters for a given mode (size: 50)
  - `y` = array with eigenfunctions (size: 6×`nr` for `nfmode=1`, and 2×`nr` for `nfmode=2` or `3`)
- Each section starts on a new line. The number of entries per line does not matter, but each entry needs to be separated by at least one space, comma or tab.
- Blank lines are not allowed, except in the header.

2.2.3 The FILOU format

The FILOU format can be described as a series of individual eigenmodes which are defined by a header and a table. The header contains a number of key parameters, preceded by descriptive character strings. The relevant parameters are:

- the harmonic degree ℓ (preceded by "DEGRE DU MODE L :")
- the number of grid points (preceded by "Nombre de points du reseau du modele :")
- the normalised squared frequency (preceded by "Frequence carree normalisee =")
- the radial order (preceded by "noeuds =")
- the frequency in μHz (preceded by "Frequence en micro Hz =")

Three supplementary lines appear between the frequency in μHz and the start of the table. The table contains 5 columns, each of which are 15 characters wide. The first column is the normalised radial position and the remaining 4 give the variables y_{01} .. y_{04} which are defined in Suárez and Goupil, 2008 (Astrophys. Space Sci. 316, 155-161). We note that `InversionKit` has not been thoroughly tested with eigenfunctions in this format. Although the FILOU oscillation code is able to take into account the effects of rotation using perturbation theory, `InversionKit` is only set up to calculate and do inversions using eigenmodes and eigenfrequencies from non-rotating models.

2.3 Target profiles

The files which contain the target profile(s) need to obey the following rules:

- On a given line, anything following a "#" is treated like a comment and ignored.
- There are 2 columns for the target rotation profile and 3 columns for the target structural profiles.
- The first column corresponds to the underlying grid. This grid needs to be in strictly ascending order. If the span covered by the grid is smaller than the span covered by the model grid, then the target profile(s) will be extrapolated to fit the model grid.
- The next column(s) contain(s) the target profile(s).
- A line with the wrong number of entries (after removal of comments) are discarded but provoke a warning message (except if there are no entries).

Note:

- the underlying grid for the target profiles needs to be non-dimensionalised (*i.e.* the grid corresponds to $x = r/R$, where R is the radius of the model)
- the same target profiles are used both for structural inversions and for mean density estimates.

2.4 Observed frequencies and rotational splittings

From version 2.1 onwards, `InversionKit` no longer reads frequency shifts. Instead, it reads directly the observed frequencies from which it calculates the frequency shifts. Furthermore, the observed frequencies need to be combined with the rotational splittings in a single file. This file should obey the following formatting rules:

- On a given line, anything following a "#" is treated like a comment and ignored.
- Lines with fewer than 3 entries are discarded and provoke a warning message.
- Lines with 3 or 4 entries should contain:
  - an integer entry with the harmonic degree (or order) ℓ
  - the observed frequency, $\nu_{\rm obs}$ (in μHz)
  - the error bar on the observed frequency, $\sigma\nu_{\rm obs}$ (in μHz)
  - If a fourth parameter or entry is present, it is discarded

- Lines with at least 5 entries should contain:
  - ℓ, $\nu_{\rm obs}$, and $\sigma\nu_{\rm obs}$ as the first three entries
  - the rotational splitting and its error bar as the next two entries (the choice of units is at the discretion of the user – the same units will also apply to the results from the rotation inversion)
  - further entries are discarded
- for the floating point entries, non-numerical entries can be used instead, such as "null", to indicate an unknown value (this will appear as a blank entry in the table)

2.5 Output files from Monte Carlo runs for splittings inequalities

A typical output from a output file from a Monte Carlo run, to test the significance of a break down of one of the inequalities on rotation splittings, as described in Reese (2015, A&A 578, 37), looks like this:

```
100000 # number of iterations
***********************
Results of inequalities
Max pair dev.:  -1.124106931668793
Max smplx dev.: 4.991075130677703
Max all dev.:   4.991075130677703 (= threshold)
***********************
Results of MCMC run
Number of iterations: 100000
Threshold:      4.991075130677703
P(pairs>smplx): 0.88768
Max pairs:      6.2644197812195
Max simplex:    5.250878571101056
Max all:        6.2644197812195
Mean pairs:     2.837817878579280
Mean simplex:   2.373868294713620
Mean all:       2.858104805242160
Stdv pairs:     0.452432712080448
Stdv simplex    0.463736879337773
Stdv all:       0.452929870248188
P(pairs > s):   1.4E-4
P(simplex>s):   4.0E-5
P(all  >  s):   1.7E-4
```

The last line gives overall false-alarm probability, whereas the lines above give various intermediate steps such as a statistical parameters from the Monte Carlo run, a break down between pair-related inequalities and those coming from simplex calculations, and the threshold deduced from the ε' value (see Reese, 2015, A&A 578, 37, Eq. 29).

2.6 GZIP compression

`InversionKit` can read "gzipped" text files. In order to determine whether or not a file is gzipped, `InversionKit` looks at the filename to see if it ends with ".gz". The same rule also applies when saving a model from the HELAS website `http://www.astro.up.pt/helas/` onto the hard disk – choosing a filename which ends with ".gz" causes `InversionKit` to save a compressed file, whereas any other ending produces an uncompressed file.

3 Treatment of pulsation modes

3.1 Matching observed and theoretical modes

As of version 2.1 onwards, the pulsation modes can either be loaded from a file or calculated directly within `InversionKit`. Furthermore, the observed frequencies are loaded rather than the frequency shifts. Hence, the two need to be matched. Such a matching can be obtained (or updated) through the "Find match" button on either the `Kernels` or `Frequency data` tabs. `InversionKit` sets up matching with the following constraints:

- the observed and theoretical modes must have the same ℓ value
- the observed and theoretical modes must have the closest frequencies possible
- at most one observed frequency corresponds to a calculated mode
- at most one calculated mode corresponds to an observed frequency (if there are multiple copies of a given theoretical mode, then it is possible to have cases where one of them is matched to a given observed frequency and another copy to another observed frequency. Such a situation will then be flagged as a duplicate mode in the "Dupl." column in the table on the `Frequency data` tab.)

Such a mapping is achieved by going through the list of observed frequencies and finding the closest theoretical mode. If this mode is already matched up with another observed frequency then it will replace the matching only if the current observed frequency is closer than the previous one. In most cases, such an approach is perfectly adequate but in some more complex cases, some potential matches may be missed. Of course, such complex situations only occur if there is a significant mismatch between the observed and theoretical frequency spectra, in which case the reference model is likely to be a rather poor choice.

3.2 Pulsation calculations

In order to calculate theoretical modes, the pulsation equations are discretised into matrix form, using the finite-difference scheme described in Reese (2013, A&A 555, A148), and solved using `jlapack` and `jarpack`, a java translation of `lapack` and `arpack`. `InversionKit` uses unmatched observed frequencies as target values for the pulsation calculations and automatically matches the resultant theoretical modes to these observed frequencies. Such a procedure ensures that the nearest theoretical frequencies are obtained.

There are three options for the set of equations. The first option is based on the system of equations used in ADIPLS (Christensen-Dalsgaard, 2008, ApSS 316, 113). The second option uses the Lagrangian pressure perturbation. The third one makes use of

the Eulerian pressure and density perturbations. In all cases, a reduced set of equations is used for radial modes. The associated sets of equations are described in the following sections.

3.3 JCD's equations

3.3.1 Non-radial modes

System of equations The following system of equations is derived from the one used in the ADIPLS code. The variables have been scaled by the appropriate power of r so as to have an $\mathcal{O}(1)$ behaviour at the centre. Furthermore, the differential system has been expressed in terms of r^2 to ensure that the solutions are a power series in r^2 rather than r:

$$2r^2\frac{\mathrm{d}y_1}{\mathrm{d}r^2} = (V_g-\ell-1)y_1+\left(1-\frac{\omega^2V_g}{\ell(\ell+1)\eta}\right)y_2-V_gy_3 \tag{9}$$

$$2r^2\omega^2\frac{\mathrm{d}y_2}{\mathrm{d}r^2} = \ell(\ell+1)\left(\omega^2-\eta A\right)y_1+\omega^2(A-\ell)y_2+\ell(\ell+1)\eta Ay_3 \tag{10}$$

$$2r^2\frac{\mathrm{d}y_3}{\mathrm{d}r^2} = (2-\ell)y_3+y_4 \tag{11}$$

$$2r^2\frac{\mathrm{d}y_4}{\mathrm{d}r^2} = -AUy_1-\frac{\omega^2UV_g}{\ell(\ell+1)\eta}y_2+\left[\ell(\ell+1)+U(A-2)\right]y_3+(3-\ell-2U)y_4 \tag{12}$$

where

$$y_1 = \frac{\xi_r}{r^{\ell-1}} \tag{13}$$

$$y_2 = \ell(\ell+1)\frac{\xi_h}{r^{\ell-1}} = \frac{\ell(\ell+1)}{\omega^2r^\ell}\left(\frac{P}{\rho_0}+\psi\right) \tag{14}$$

$$y_3 = -\frac{\psi}{r^{\ell-1}g_0} \tag{15}$$

$$y_4 = -r^{3-\ell}\frac{\mathrm{d}}{\mathrm{d}r}\left(\frac{\psi}{rg_0}\right) = r^{3-\ell}\frac{\mathrm{d}r^{\ell-2}y_3}{\mathrm{d}r} \tag{16}$$

$$\xi_r = \text{the Lagrangian vertical displacement} \tag{17}$$

$$\xi_h = \text{the Lagrangian horizontal displacement} \tag{18}$$

$$\psi = \text{the Eulerian perturbation to the gravitational potential} \tag{19}$$

$$V_g = -\frac{1}{\Gamma_1}\frac{\mathrm{d}\ln P_0}{\mathrm{d}\ln r} = \frac{m_0\rho_0}{\Gamma_1P_0r} = \frac{m_0}{rc_0^2} \tag{20}$$

$$A = \frac{1}{\Gamma_1}\frac{\mathrm{d}\ln P_0}{\mathrm{d}\ln r}-\frac{\mathrm{d}\ln\rho_0}{\mathrm{d}\ln r} \tag{21}$$

$$U = \frac{4\pi\rho_0r^3}{m_0} \tag{22}$$

$$\eta = \frac{m_0}{r^3}, \tag{23}$$

the above variables being non-dimensionalised with respect to R, M/R^2, and GM^2/R^4 as units of length, density and pressure, respectively. *NOTE*: in JCD's notes for ADIPLS, the convention $\Delta\psi = -4\pi G\rho$ is used, thereby explaining the sign differences with the following sets of equations.

Boundary conditions In the centre, the boundary conditions are:

$$y_2 = (\ell+1)y_1 \tag{24}$$
$$y_4 = (\ell-2)y_3 \tag{25}$$

At the surface, the boundary conditions are:

$$\omega^2 y_2 = \ell(\ell+1)\eta(y_1 - y_3) \tag{26}$$
$$y_4 = -(\ell+U)y_3 + Uy_1 \tag{27}$$

Interface conditions When working with a model with a discontinuity[1], say on the density profile, it is necessary to introduce conditions which relate the variables below to the variables above the discontinuity. These are known as interface conditions. In the present case, the interface conditions are:

$$y_1^- = y_1^+ \tag{28}$$
$$\frac{\omega^2 U^-}{\ell(\ell+1)} y_2^- + \eta U^- \left(y_3^- - y_1^-\right) = \frac{\omega^2 U^+}{\ell(\ell+1)} y_2^+ + \eta U^+ \left(y_3^+ - y_1^+\right) \tag{29}$$
$$y_3^- = y_3^+ \tag{30}$$
$$y_4^- + U^- \left(y_3^- - y_1^-\right) = y_4^+ + U^+ \left(y_3^+ - y_1^+\right) \tag{31}$$

where the "−" and "+" superscripts denote the variable right below and right above the discontinuity. These interface conditions ensure the continuity of the radial Lagrangian displacement, the Lagrangian pressure perturbations, and the Lagrangian perturbations to the gravitational potential and its gradient.

3.3.2 Radial modes

System of equations In the radial case, the above equations are not appropriate, given the $1/[\ell(\ell+1)]$ factor which appears in some of the terms. Furthermore, one can eliminate the perturbations to the gravitational potential by integrating Poisson's equation (see, e.g., Aerts et al. 2010, *Asteroseismology*, p. 195):

$$\frac{\mathrm{d}\psi}{\mathrm{d}r} = -\Lambda \rho_0 \xi_r \tag{32}$$

Hence, the following reduced system is used instead:

$$2r^2 \frac{\mathrm{d}\tilde{y}_1}{\mathrm{d}r^2} = \left(V_g - 3\right)\tilde{y}_1 - \frac{\omega^2}{c_0^2}\tilde{y}_2, \tag{33}$$
$$2\omega^2 \frac{\mathrm{d}\tilde{y}_2}{\mathrm{d}r^2} = \left[\omega^2 - \eta(A-U)\right]\tilde{y}_1 + \omega^2 \frac{A}{r^2}\tilde{y}_2 \tag{34}$$

where

$$\tilde{y}_1 = \frac{\xi_r}{r} \tag{35}$$
$$\tilde{y}_2 = \frac{P}{\omega^2 \rho_0} \tag{36}$$

[1] In `InversionKit`, discontinuities correspond to places with double points, *i.e.* where two mesh points have the exact same numerical value. If three or more mesh points have the exact same value, `InversionKit` will remove the intermediate points so that only a double point is left.

Boundary conditions In the centre, the boundary condition is:

$$0 = 3\tilde{y}_1 + \frac{\omega^2}{c_0^2}\tilde{y}_2 \tag{37}$$

At the surface, the boundary condition is:

$$0 = r^2\eta\tilde{y}_1 - \omega^2\tilde{y}_2 \tag{38}$$

Interface conditions The interface conditions are:

$$\tilde{y}_1^- = \tilde{y}_1^+ \tag{39}$$

$$\omega^2 U^-\tilde{y}_2^- - r^2\eta U^-\tilde{y}_1^- = \omega^2 U^+\tilde{y}_2^+ - r^2\eta U^+\tilde{y}_1^+ \tag{40}$$

3.4 Lagrangian pressure perturbations

The main advantage of using the Lagrangian pressure perturbation is that the Brunt-Väisälä drops out from the equations. Nonetheless, this set of equations is not a good choice when dealing with red giants given that the Lagrangian pressure perturbation is dominated by the $\vec{\xi}\cdot\vec{\nabla}P_0$ term.

3.4.1 Non-radial modes

System of equations As was done above, the variables have been scaled by the appropriate power of r so as to have an $\mathcal{O}(1)$ behaviour at the centre, and the differential system has been expressed in terms of r^2 (the entity $\left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right)$ is treated as a variable in its own right, so is not derived with respect to r^2):

$$0 = \frac{r^2}{\Gamma_1}\frac{\delta\tilde{P}}{P_0} + 2r^2\frac{\mathrm{d}\tilde{\xi}_r}{\mathrm{d}r^2} + (\ell+1)\tilde{\xi}_r - \ell(\ell+1)\tilde{\xi}_h \tag{41}$$

$$\begin{aligned}\omega^2\tilde{\xi}_r = {} & \frac{2r^2P_0}{\rho_0}\frac{\mathrm{d}}{\mathrm{d}r^2}\left(\frac{\delta\tilde{P}}{P_0}\right) + \left[\frac{\ell P_0}{\rho_0} + \left(\frac{1}{\Gamma_1} - 1\right)g_0 r\right]\frac{\delta\tilde{P}}{P_0} \\ & + \left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) + 2rg_0\frac{\mathrm{d}\tilde{\xi}_r}{\mathrm{d}r^2} + \left(\frac{(\ell-1)g_0}{r} + \frac{\mathrm{d}g_0}{\mathrm{d}r}\right)\tilde{\xi}_r\end{aligned} \tag{42}$$

$$\omega^2\tilde{\xi}_h = \frac{P_0}{\rho_0}\frac{\delta\tilde{P}}{P_0} + \tilde{\psi} + \frac{g_0}{r}\tilde{\xi}_r \tag{43}$$

$$0 = 2r^2\frac{\mathrm{d}}{\mathrm{d}r^2}\left(\tilde{\psi}\right) + \ell\tilde{\psi} - \left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) \tag{44}$$

$$0 = 2r^2\frac{\mathrm{d}}{\mathrm{d}r^2}\left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) + (\ell+1)\left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) - \ell(\ell+1)\tilde{\psi} - \Lambda\left(\frac{r^2\rho_0}{\Gamma_1}\frac{\delta\tilde{P}}{P_0} - r\frac{\mathrm{d}\rho_0}{\mathrm{d}r}\tilde{\xi}_r\right) \tag{45}$$

where

$$\begin{aligned}
\delta\tilde{P} &= r^{-\ell}\delta P \\
\tilde{\xi}_r &= r^{1-\ell}\xi_r \\
\tilde{\xi}_h &= r^{1-\ell}\xi_h \\
\tilde{\psi} &= r^{-\ell}\psi \\
\left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) &= r^{1-\ell}\frac{\mathrm{d}\psi}{\mathrm{d}r}
\end{aligned}$$

The above equations are solved in non-dimensional form, where R, M/R^3 and GM^2/R^4 are units of length, density and pressure, respectively. This leads to $\Lambda = 4\pi$ (as opposed to $4\pi G$ in dimensional form). Once the above system is solved, the original non-scaled variables are obtained by multiplying the scaled variables by the appropriate power of r.

Boundary conditions The boundary conditions are as follows. In the centre, the continuity equation and the equation relating $\tilde{\psi}$ and $\left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right)$ become:

$$0 = \tilde{\xi}_r - \ell\tilde{\xi}_h \tag{46}$$

$$0 = \ell\tilde{\psi} - \left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) \tag{47}$$

At the surface, the boundary conditions are:

$$0 = \frac{\delta\tilde{P}}{P_0} \tag{48}$$

$$0 = \left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) + (\ell+1)\tilde{\psi} + \Lambda\rho_0\tilde{\xi}_r \tag{49}$$

Interface conditions The interface conditions are:

$$\tilde{\xi}_r^- = \tilde{\xi}_r^+ \tag{50}$$

$$\frac{\delta\tilde{P}^-}{P_0} = \frac{\delta\tilde{P}^+}{P_0} \tag{51}$$

$$\tilde{\psi}^- = \tilde{\psi}^+ \tag{52}$$

$$\left(\frac{\mathrm{d}\tilde{\psi}^-}{\mathrm{d}r}\right) + \Lambda\rho_0^-\tilde{\xi}_r^- = \left(\frac{\mathrm{d}\tilde{\psi}^+}{\mathrm{d}r}\right) + \Lambda\rho_0^+\tilde{\xi}_r^+ \tag{53}$$

As above, these interface conditions ensure the continuity of the radial Lagrangian displacement, the Lagrangian pressure perturbations, and the Lagrangian perturbations to the gravitational potential and its gradient.

3.4.2 Radial modes

System of equations In the radial case, the above system can be reduced by eliminating the perturbations to the gravitational potential (see Eq. 32) and the variable ξ_h,

which is undefined The final system is, after some manipulations:

$$0 = \frac{1}{\Gamma_1}\frac{\delta P}{P_0} + 2r^2\frac{\mathrm{d}\hat{\xi}_r}{\mathrm{d}r^2} + 3\hat{\xi}_r \tag{54}$$

$$\omega^2\hat{\xi}_r = \frac{2P_0}{\rho_0}\frac{\mathrm{d}}{\mathrm{d}r^2}\left(\frac{\delta P}{P_0}\right) - \frac{g_0}{r}\frac{\delta P}{P_0} - 4\frac{g_0}{r}\hat{\xi}_r \tag{55}$$

where $\xi_r = r\hat{\xi}_r$.

Boundary conditions The boundary conditions are:

$$0 = \frac{1}{\Gamma_1}\frac{\delta P}{P_0} + 3\hat{\xi}_r \qquad \text{at} \qquad r = 0 \tag{56}$$

$$0 = \frac{\delta P}{P} \qquad \text{at} \qquad r = R \tag{57}$$

Interface conditions The interface conditions are:

$$\hat{\xi}_r^- = \hat{\xi}_r^+ \tag{58}$$

$$\frac{\delta P^-}{P_0} = \frac{\delta P^+}{P_0} \tag{59}$$

3.5 Eulerian pressure and density perturbations

This is a straightforward, but somewhat naïve version of the pulsation equations with the Eulerian pressure and density perturbations, P and ρ. As was the case for JCD's equations, the Brunt-Väisälä appears explicitly. However, unlike in JCD's equations, the variable P appears on its own rather than implicitly through the horizontal displacement. Accordingly, this set of equations fails to produce good results in red giants, and is also more expensive since it involves 6 rather than 4 unknowns. It is kept primarily for the purposes of carrying out tests and comparisons.

3.5.1 Non-radial modes

System of equations Once more, the variables have been scaled by the appropriate power of r so as to have an $\mathcal{O}(1)$ behaviour at the centre, and the differential system has been expressed in terms of r^2:

$$0 = r^2\tilde{\rho} + 2r^2\rho_0\frac{\mathrm{d}\tilde{\xi}_r}{\mathrm{d}r^2} + \left[(\ell+1)\rho_0 + r\frac{\mathrm{d}\rho_0}{\mathrm{d}r}\right]\tilde{\xi}_r - \ell(\ell+1)\rho_0\tilde{\xi}_h \tag{60}$$

$$\omega^2\rho_0\tilde{\xi}_r = 2r^2\frac{\mathrm{d}\tilde{P}}{\mathrm{d}r^2} + \ell\tilde{P} + rg_0\tilde{\rho} + \rho_0\left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) \tag{61}$$

$$\omega^2\rho_0\tilde{\xi}_h = \tilde{P} + \rho_0\tilde{\psi} \tag{62}$$

$$0 = \tilde{P} - c_0^2\tilde{\rho} + \frac{\rho_0 N_0^2 c_0^2}{rg_0}\tilde{\xi}_r \tag{63}$$

$$0 = 2r^2\frac{\mathrm{d}}{\mathrm{d}r^2}\left(\tilde{\psi}\right) + \ell\tilde{\psi} - \left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) \tag{64}$$

$$0 = 2r^2\frac{\mathrm{d}}{\mathrm{d}r^2}\left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) + (\ell+1)\left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r}\right) - \ell(\ell+1)\tilde{\psi} - \Lambda r^2\tilde{\rho} \tag{65}$$

where

$$
\begin{aligned}
P &= r^\ell \tilde{P} = \text{the Eulerian pressure perturbation} \\
\rho &= r^\ell \tilde{\rho} = \text{the Eulerian pressure perturbation}
\end{aligned}
$$

The above equations are solved in non-dimensional form, where R, M/R^3 and GM^2/R^4 are units of length, density and pressure, respectively. This leads to $\Lambda = 4\pi$ (as opposed to $4\pi G$ in dimensional form). Once the above system is solved, the original non-scaled variables are obtained by multiplying the scaled variables by the appropriate power of r.

Boundary conditions In the centre, the boundary conditions are:

$$0 = \tilde{\xi}_r - \ell \tilde{\xi}_h \tag{66}$$

$$0 = \ell \tilde{\psi} - \left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r} \right) \tag{67}$$

At the surface, the boundary conditions are:

$$0 = r\tilde{P} - \rho_0 g_0 \tilde{\xi}_r \tag{68}$$

$$0 = \left(\frac{\mathrm{d}\tilde{\psi}}{\mathrm{d}r} \right) + (\ell + 1)\tilde{\psi} + \Lambda \rho_0 \tilde{\xi}_r \tag{69}$$

Interface conditions The interface conditions are:

$$\tilde{\xi}_r^- = \tilde{\xi}_r^+ \tag{70}$$

$$r\tilde{P}^- - \rho_0^- g_0 \tilde{\xi}_r^- = r\tilde{P}^+ - \rho_0^+ g_0 \tilde{\xi}_r^+ \tag{71}$$

$$\tilde{\psi}^- = \tilde{\psi}^+ \tag{72}$$

$$\left(\frac{\mathrm{d}\tilde{\psi}^-}{\mathrm{d}r} \right) + \Lambda \rho_0^- \tilde{\xi}_r^- = \left(\frac{\mathrm{d}\tilde{\psi}^+}{\mathrm{d}r} \right) + \Lambda \rho_0^+ \tilde{\xi}_r^+ \tag{73}$$

As before, these interface conditions ensure the continuity of the radial Lagrangian displacement, the Lagrangian pressure perturbations, and the Lagrangian perturbations to the gravitational potential and its gradient.

3.5.2 Radial modes

System of equations As previously, the perturbations to the gravitational potential are eliminated analytically, thanks to Eq. (32). After some manipulations, the final system is:

$$0 = \frac{1}{\Gamma_1 P_0} P + 2r^2 \frac{\mathrm{d}\hat{\xi}_r}{\mathrm{d}r^2} + \left(3 - \frac{r g_0}{c_0^2} \right) \hat{\xi}_r \tag{74}$$

$$\omega^2 \rho_0 \hat{\xi}_r = 2\frac{\mathrm{d}P}{\mathrm{d}r^2} + \frac{g_0}{r c_0^2} P - \left(\Lambda \rho_0^2 + \frac{\rho_0 g_0^2}{c_0^2} + g_0 \frac{\mathrm{d}\rho_0}{\mathrm{d}r} \right) \hat{\xi}_r \tag{75}$$

where $\xi_r = r\hat{\xi}_r$.

Boundary conditions The boundary conditions are:

$$0 = \frac{1}{\Gamma_1 P_0} P + 3\hat{\xi}_r \qquad \text{at} \qquad r = 0 \tag{76}$$

$$0 = P - r\rho_0 g_0 \hat{\xi}_r \qquad \text{at} \qquad r = R \tag{77}$$

Interface conditions The interface conditions are:

$$\hat{\xi}_r^- = \hat{\xi}_r^+ \tag{78}$$

$$P^- - r\rho_0^- g_0 \hat{\xi}_r^- = P^+ - r\rho_0^+ g_0 \hat{\xi}_r^+ \tag{79}$$

3.6 Reconstructing missing variables

Evidently, the above systems of equations do not use the same variables. When constructing kernels, however, it is helpful to have the same set of variables for all of the pulsation modes. A simple solution consists in having a maximal set of variables which include all of the above variables. This, therefore, involves reconstructing missing variables in the different cases.

We used the following formulas to reconstruct variables when working with the Lagrangian pressure perturbations:

$$P = P_0 \frac{\delta P}{P_0} + \rho_0 g_0 \xi_r \tag{80}$$

$$\rho = \frac{\rho_0}{\Gamma_1} \frac{\delta P}{P_0} - \frac{\mathrm{d}\rho_0}{\mathrm{d}r} \xi_r \tag{81}$$

When working with Eulerian pressure and density perturbations, we applied the following formula:

$$\frac{\delta P}{P_0} = \frac{P - \rho_0 g_0 \xi_r}{P_0} \tag{82}$$

When working with JCD's formulas, the following equations were applied:

$$\psi = -r^{\ell-1} g_0 y_3 \tag{83}$$

$$\frac{\mathrm{d}\psi}{\mathrm{d}r} = -r^{\ell-1} \left[\frac{g_0 y_4}{r} + \left(\frac{g_0}{r} + \frac{\mathrm{d}g_0}{\mathrm{d}r} \right) y_3 \right] \tag{84}$$

$$P = \rho_0 (\omega^2 r \xi_h - \psi) \tag{85}$$

$$\rho = \frac{P}{c_0^2} + \frac{A\rho_0}{r} \xi_r \tag{86}$$

$$\frac{\delta P}{P_0} = \frac{P - \rho_0 g_0 \xi_r}{P_0} \tag{87}$$

When dealing with radial modes, the above formulas were used as needed. The derivative of the gravitational potential perturbations, ψ was obtained thanks to Eq. (32), and ψ was obtained via the following formula which takes into account the boundary condition on ψ and $\frac{\mathrm{d}\psi}{\mathrm{d}r}$:

$$\psi = \int_R^r \frac{\mathrm{d}\psi}{\mathrm{d}r} \mathrm{d}r \tag{88}$$

4 Inversions

4.1 Rotation inversions

4.1.1 Description of the problem

A stellar rotation profile which only depends on depth, $\Omega(r)$, shifts the frequencies by an amount that is proportional to m, the azimuthal order:

$$R_{n\ell} \equiv \frac{\nu_{n\ell m} - \nu_{n\ell 0}}{m} = \int_0^R K_\Omega^{n\ell}(r)\Omega(r)dr \tag{89}$$

where n is the radial order, ℓ the harmonic degree and $K_\Omega^{n\ell}$ the rotational kernel. The quantity $R_{n\ell}$ is the rotational splitting and is deduced from observations for a set of (n, ℓ) pairs. The associated rotational kernels can be calculated from the corresponding pulsation modes in a non-rotating reference model (see Section 4.1.2 for an explicit expression). The goal of an inversion procedure is then to deduce the unknown rotation profile, $\Omega(r)$ from $R_{n\ell}$ and $K_\Omega^{n\ell}$. Two methods for doing this are implemented in `InversionKit`: Regularised Least Squares (RLS) and Subtractive Optimally Localised Averages (SOLA). These are described in Sections 4.1.3 and 4.1.4, respectively.

4.1.2 Expression for the rotational kernel

The rotational kernel can be expressed as follows:

$$K_\Omega^{n\ell} = \frac{\rho_0 r^2}{I}\left(\xi^2 + \ell(\ell+1)\eta^2 - 2\xi\eta - \eta^2\right) \tag{90}$$

where

$$\begin{aligned}
\vec{\xi} &= \xi Y_m^\ell \vec{e}_r + \eta\left(\frac{\partial Y_m^\ell}{\partial\theta}\vec{e}_\theta + \frac{1}{\sin\theta}\frac{\partial Y_m^\ell}{\partial\varphi}\vec{e}_\phi\right) \\
&= \text{the Lagrangian displacement resulting from the pulsation} \\
I &= \int_{r=0}^R \rho_0(r)\left(\xi^2 + \ell(\ell+1)\eta^2\right) r^2 dr \\
\rho_0 &= \text{the equilibrium density}
\end{aligned}$$

It is important to note that the eigenfunctions ξ and η only depend on (n, ℓ) and not m in a non-rotating reference model.

4.1.3 RLS – Regularised Least Squares

Both RLS and SOLA inversions involve minimising "cost" functions which will be represented by the letter J in what follows. In an RLS inversion, J is defined as:

$$J(f) = \sum_{l=1}^L \left\{\frac{R_l - \int_0^R K_\Omega^l(r)f(r)dr}{\sigma_l}\right\}^2 + \Lambda\left\langle\frac{1}{\sigma^2}\right\rangle\int_0^R\left\{\frac{d^2 f}{dr^2}\right\}^2 dr \tag{91}$$

where R_l is the rotational splittings, σ_l the corresponding errors, K_Ω^l the corresponding rotational kernels, and $\left\langle\frac{1}{\sigma^2}\right\rangle = \frac{1}{L}\left(\sum_{l=1}^L \frac{1}{\sigma_l^2}\right)$. The index l is used to represent the pairs (n, ℓ), and L the number of such pairs. Λ is a trade-off parameter between conforming to

data and regularising the solution, and can be regulated in the `Rotational inversion` tab. The function f that minimises J is the inversion result and corresponds to the profile which best reproduces the rotational splittings while satisfying the regularisation constraint.

4.1.4 SOLA – Subtractive Optimally Localised Averages

The SOLA inversion procedure is one of several inversion methods which focuses on constructing "nice" averaging kernels $K_{\text{avg.}}$ (see Section 4.1.6 for an explanation on averaging kernels). It was first introduced by Pijpers & Thompson (1992, A&A 262, L33) and has the advantage of being less computationally expensive than MOLA inversions. In what follows, a description of the SOLA method is given.

For a given grid point, r_0, a target function $T(r_0, r)$ is chosen. The cost function, J, is then set up so as to minimise the difference between the averaging kernel and $T(r_0, r)$ while reducing the effects of the observational errors σ_l.

$$\begin{aligned} J(c_l(r_0)) &= \int_0^R \left\{T(r_0,r) - K_{\text{avg.}}(r_0,r)\right\}^2 dr + \frac{\tan\theta \sum_{l=1}^{L} \left(c_l(r_0)\sigma_l\right)^2}{\langle\sigma^2\rangle} \\ &+ \lambda\left\{1 - \int_0^R K_{\text{avg.}}(r_0,r)dr\right\} \end{aligned} \tag{92}$$

where

$$\begin{aligned} c_l(r_0) &= \text{coefficients from the inversion} \\ \langle\sigma^2\rangle &= \frac{1}{L}\sum_{l=1}^{L}\sigma_l^2 \\ \theta &= \text{a trade-off parameter between optimising } K_{\text{avg.}} \text{ and reducing the error,} \\ &\quad \text{and which can be adjusted by the user} \\ T(r_0,r) &= \text{a target function} \\ K_{\text{avg.}}(r_0,r) &= \sum_{l=1}^{L} c_l(r_0)K_\Omega^l(r) = \text{the averaging kernel} \\ \lambda &= \text{Lagrangian multiplier used to ensure } \int_0^R K_{\text{avg.}}(r_0,r)dr = 1 \end{aligned}$$

In `InversionKit`, the following form has been chosen for the target function:

$$T(r_0,r) = Ar\exp\left\{-\left[\frac{r-r_0}{\Delta\cdot c(r_0)} + \frac{\Delta\cdot c(r_0)}{2r_0}\right]^2\right\} \tag{93}$$

where A is a normalisation factor such that $\int_0^R T(r_0,r)dr = 1$. This type of target function was taken from Rabello-Soares et al. (1999, MNRAS 309, 35), has a maximum at r_0, and behaves like a Gaussian, except that it goes to zero at $r = 0$. The width is $\Delta\cdot c(r_0)$, where $c(r_0)$ is the sound velocity at r_0 (and shouldn't be confused with the inversion coefficients $c_l(r_0)$) and Δ a free parameter which can be adjusted by the user. This form for the

width reflects the resolving power of the kernels (Thompson 1993, ASPCS 42, 141). The solution $\Omega_{\text{inv.}}$ is then constructed as follows:

$$\Omega_{\text{inv.}}(r_0) = \sum_{l=1}^{L} c_l(r_0) R_l \tag{94}$$

for a set of grid points r_0.

4.1.5 MOLA – Multiplicative Optimally Localised Averages

The MOLA inversion procedure is another inversion method which focuses on constructing "nice" averaging kernels $K_{\text{avg.}}$ (see Section 4.1.6 for an explanation on averaging kernels). It was first introduced by Backus & Gilbert (1968, Geophys. J. 16, 169). It is more computationally expensive than SOLA since there is a matrix resolution for each grid point where the inversion is carried out, but involves one less free parameter than a SOLA inversion. In what follows, a description of the MOLA method is given.

For a given grid point, r_0, the following cost function, J, is minimised:

$$\begin{aligned} J(c_l(r_0)) &= \int_0^R 12(r-r_0)^2 K_{\text{avg.}}^2(r_0,r)dr + \frac{\tan\theta \sum_{l=1}^{L} \left(c_l(r_0)\sigma_l\right)^2}{\langle\sigma^2\rangle} \\ &+ \lambda\left\{1 - \int_0^R K_{\text{avg.}}(r_0,r)dr\right\} \end{aligned} \tag{95}$$

where

$$\begin{aligned} c_l(r_0) &= \text{coefficients from the inversion} \\ \langle\sigma^2\rangle &= \frac{1}{L}\sum_{l=1}^{L}\sigma_l^2 \\ \theta &= \text{a trade-off parameter between optimising } K_{\text{avg.}} \text{ and reducing the error,} \\ &\quad \text{and which can be adjusted by the user} \\ K_{\text{avg.}}(r_0,r) &= \sum_{l=1}^{L} c_l(r_0) K_\Omega^l(r) = \text{the averaging kernel} \\ \lambda &= \text{Lagrangian multiplier used to ensure } \int_0^R K_{\text{avg.}}(r_0,r)dr = 1 \end{aligned}$$

Minimising the above cost function ensures that the averaging kernel, $K_{\text{avg.}}$ is well localised around r_0 since it increasingly penalises departures from 0 when r is getting further from r_0. This is somewhat different than a SOLA inversion where the penalisation does not change as a function of distance from r_0.

4.1.6 Error bars and averaging kernels

In linear inversion procedures, including RLS and SOLA, the inverted rotation profile is a linear combination of the rotational splittings:

$$\Omega_{\text{inv.}}(r_0) = \sum_{l=1}^{L} c_l(r_0) R_l \tag{96}$$

where $c_l(r_0)$ represents the inversion coefficients. Applying this linear combination to Eq. (89) yields the following relation:

$$\Omega_{\text{inv.}}(r_0) = \int_0^R \sum_{l=1}^{L} c_l(r_0) K_\Omega^l(r)\Omega(r)dr = \int_0^R K_{\text{avg.}}(r)\Omega(r)dr \tag{97}$$

From this, we see that $\Omega_{\text{inv.}}(r_0)$ is actually an average of the true rotation profile $\Omega(r)$, in which $K_{\text{avg.}}$ plays the role of a weight function, hence the name "averaging kernel". Such functions are useful for assessing the quality of the inversion (see Christensen-Dalsgaard et al., 1990, MNRAS 242, 353). `InversionKit` allows the user to plot the averaging kernel from both types of inversion methods for the different grid points, r_0, used in the inversion.

Additionally, $K_{\text{avg.}}$ is used to calculate the horizontal error bars in SOLA inversions: the left end of the error bar corresponds to the first quartile point, the right end is the third quartile point and the point r_0 is redefined to be the second quartile point. These quartile points are calculated through integration starting from the centre of the star.

From the inversion coefficients, $c_l(r_0)$, and the 1-sigma measurement errors, σ_l, it is possible to define a 1-sigma error on the inversion result at each point, r_0:

$$\mathcal{E}(r_0) = \sqrt{\sum_{l=1}^{L} \left(c_l(r_0)\sigma_l\right)^2} \tag{98}$$

$\mathcal{E}(r_0)$ is then used to draw the vertical error bars on both RLS and SOLA inversions. For RLS inversions, this is represented by dotted red lines below and above the inverted profile, whereas for SOLA inversions, vertical error bars are plotted.

Note: For limited sets of modes, the averaging kernels can be highly oscillatory, meaning that multiple values could qualify as first, second and third quartile points. This makes the above definition for the horizontal error bars essentially meaningless. In such a situation, `InversionKit` arbitrarily uses the innermost quartile points, which of course introduces a bias towards the centre of the star. It is therefore important to look at the averaging kernels, rather than simply trusting the horizontal error bars.

4.2 Rotation gradient inversion and angular momentum inversion

These inversions are quite similar to the rotation inversions. They are based on kernels which can be deduced through an integration by part, as described in Reese (2015, A&A 578, 37). One subtle difference is that a supplementary Lagrangian multiplier is needed to cancel out the surface term which comes from the integration by part. This also has the side-effect of hindering the inversion from getting information from near the surface.

4.3 Structural inversions

4.3.1 Description of the problem

A structural inversion problem typically takes the following form:

$$S_{n\ell} \equiv \frac{\delta\nu_{n\ell}}{\nu_{n\ell}} = \int_0^R K_{a,b}^{n\ell}(r)\frac{\delta a}{a}dr + \int_0^R K_{b,a}^{n\ell}(r)\frac{\delta b}{b}dr + \frac{F_{\text{surf.}}(\nu_{n\ell})}{E_{n\ell}} \tag{99}$$

where $S_{n\ell}$ is the relative frequency difference (or shift) between the observed and calculated frequencies (obtained for $m = 0$), a and b structural profiles (such as c_0^2, ρ_0, Γ_1 etc.), δa and δb the differences between the true structural profiles and the ones from the reference model, $F_{\text{surf.}}$ a slowly varying function which represents unknown surface effects, and $E_{n\ell}$ the mode inertia:

$$E_{n\ell} = \frac{\int_0^R \left[\xi^2 + \ell(\ell+1)\eta^2\right] \rho r^2 dr}{M\left[\xi(R_{\text{phot.}})^2 + \ell(\ell+1)\eta(R_{\text{phot.}})^2\right]} \tag{100}$$

R and $R_{\text{phot.}}$ are the surface and photospheric radii of the model. The goal of a structural inversion procedure is to find the unknown functions $\delta a/a$ and $\delta b/b$ from a set of relative frequency shifts, $S_{n\ell}$, and from the associated structural kernels, $K_{a,b}^{n\ell}$ and $K_{b,a}^{n\ell}$ (which are calculated from the pulsation modes of the model). Although similar to the rotational inversion problem described in Section 4.1.1, there are some noteworthy differences. Firstly, the inversion procedure needs to find two unknown functions – this raises the possibility of cross-talk between the two solutions. Secondly, there is an additional surface term which needs to be suppressed.

In what follows, explicit expressions are given for structural kernels associated with the structural pairs (c_0^2, ρ_0) and (Γ_1, ρ_0). These expressions are obtained by perturbing the variational expression for the frequency, then using the the variational principle to remove terms related to the variation of the eigenfunctions. The following expressions are from Gough & Thompson (1991, in *Solar interior and atmosphere*, p. 519-561).

4.3.2 c^2, ρ kernels

The relative frequency variations are given by

$$\frac{\delta\nu}{\nu} = \int_{r=0}^{R} \left[K_{c^2,\rho}(r)\frac{\delta c_0^2(r)}{c_0^2(r)} + K_{\rho,c^2}(r)\frac{\delta\rho_0(r)}{\rho_0(r)}\right] dr \tag{101}$$

in which the kernels take on the following expressions:

$$K_{c^2,\rho} = \frac{\rho_0 c_0^2 \chi^2 r^2}{2I\omega^2} \tag{102}$$

$$\begin{aligned} K_{\rho,c^2} = \frac{\rho_0 r^2}{2I\omega^2}\Bigg\{ & c_0^2\chi^2 - \omega^2\left(\xi^2 + \ell(\ell+1)\eta^2\right) - 2g_0\xi\chi - 4\pi G\int_{s=r}^{R}\left(2\rho_0\xi\chi + \frac{\mathrm{d}\rho_0}{\mathrm{d}s}\xi^2\right) ds \\ & + 2g_0\xi\frac{\mathrm{d}\xi}{\mathrm{d}r} + 4\pi G\rho_0\xi^2 + 2\left(\xi\frac{\mathrm{d}\psi}{\mathrm{d}r} + \frac{\ell(\ell+1)\eta\psi}{r}\right)\Bigg\} \end{aligned} \tag{103}$$

where

$$
\begin{aligned}
\omega &= 2\pi\nu \\
\chi &= \frac{\vec{\nabla}\cdot\vec{\xi}}{Y_m^\ell} = \frac{\mathrm{d}\xi}{\mathrm{d}r} + \frac{2\xi}{r} - \frac{\ell(\ell+1)\eta}{r} \\
\rho &= -\frac{\mathrm{d}\rho_0}{\mathrm{d}r}\xi - \rho_0\chi \\
\psi &= -\frac{4\pi G}{2\ell+1}\left[\int_{s=0}^{r}\rho(s)\frac{s^{\ell+2}}{r^{\ell+1}}ds + \int_{s=r}^{R}\rho(s)\frac{r^\ell}{s^{\ell-1}}ds\right] \\
\frac{\mathrm{d}\psi}{\mathrm{d}r} &= -\frac{4\pi G}{2\ell+1}\left[-(\ell+1)\int_{s=0}^{r}\rho(s)\frac{s^{\ell+2}}{r^{\ell+2}}ds + \ell\int_{s=r}^{R}\rho(s)\frac{r^{\ell-1}}{s^{\ell-1}}ds\right] \\
m_0 &= 4\pi\int_{s=0}^{r}\rho_0(s)s^2ds \\
g_0 &= \frac{Gm_0}{r^2}
\end{aligned}
$$

and quantities with the subscript "0" refer to the equilibrium model.

4.3.3 Γ_1, ρ kernels

The relative frequency variations are given by

$$
\frac{\delta\nu}{\nu} = \int_0^R \left[K_{\Gamma_1,\rho}(r)\frac{\delta\Gamma_1(r)}{\Gamma_1(r)} + K_{\rho,\Gamma_1}(r)\frac{\delta\rho_0(r)}{\rho_0(r)}\right]dr \tag{104}
$$

Expressions for these kernels can be deduced from the previous set of kernels:

$$
K_{\Gamma_1,\rho} = K_{c^2,\rho} = \frac{\rho_0 c_0^2\chi^2 r^2}{2I\omega^2} \tag{105}
$$

$$
\begin{aligned}
K_{\rho,\Gamma_1} &= K_{\rho,c^2} - K_{c^2,\rho} + \frac{Gm\rho_0}{r^2}\int_{s=0}^{r}\frac{K_{c^2,\rho}(s)}{p_0(s)}ds + \rho_0 r^2\int_{s=r}^{R}\frac{4\pi G\rho_0}{s^2}\left(\int_{t=0}^{s}\frac{K_{c^2,\rho}(t)}{p_0(t)}dt\right)ds \\
&= K_{\rho,c^2} - K_{c^2,\rho} + \frac{Gm\rho_0}{r^2}\int_{s=0}^{r}\frac{\Gamma_1\chi^2 s^2}{2I\omega^2}ds + \rho_0 r^2\int_{s=r}^{R}\frac{4\pi G\rho_0}{s^2}\left(\int_{t=0}^{s}\frac{\Gamma_1\chi^2 t^2}{2I\omega^2}dt\right)ds
\end{aligned} \tag{106}
$$

4.3.4 RLS

The cost function, J, is defined as:

$$
\begin{aligned}
J(f,g,a_n) &= \sum_{l=1}^{L}\left\{\frac{S_l - \int_0^R K_{a,b}^l(r)f(r)dr - \int_0^R K_{b,a}^l(r)g(r)dr - \sum_{n=0}^{N-1}\frac{a_n\psi_n(\nu_l)}{E_l}}{\sigma_l}\right\}^2 \\
&+ \left\langle\frac{1}{\sigma^2}\right\rangle\Lambda\int_0^R\left\{\frac{d^2f}{dr^2}\right\}^2 + \left\{\frac{d^2g}{dr^2}\right\}^2 dr
\end{aligned} \tag{107}
$$

where S_l is the frequency shifts to due changes in the stellar structure, σ_l the corresponding error, $K_{a,b}^l$, $K_{b,a}^l$ the corresponding structural kernels, $\left\langle\frac{1}{\sigma^2}\right\rangle = \frac{1}{L}\left(\sum_{l=1}^{L}\frac{1}{\sigma_l^2}\right)$, and $\sum_{n=0}^{N-1}\frac{a_n\psi_n(\nu_l)}{E_l}$ an ad-hoc way of modelling surface effects (where $(\psi_n)_{n\in[0,N-1]}$ is a set of polynomials). The functions f and g and coefficients a_n that minimise J correspond to

the inversion result. Λ is a trade-off parameter between conforming to data and regularising the functions f and g, and can be regulated in the `Structural inversion` tab. N is the number of polynomials used to model surface effects and can also be modified in the `Structural inversion` tab. An additional Lagrangian constraint can be added to maintain a constant mass for the star when one of the two structural profiles (a or b) is the density profile.

4.3.5 SOLA

There are two separate minimisations, one for each structural profile:

$$\begin{aligned} J(c_l(r_0)) &= \int_0^R \left\{T(r_0,r) - K_{\text{avg.}}(r_0,r)\right\}^2 dr + \beta \int_0^R \left\{K_{\text{cross}}(r_0,r)\right\}^2 w(r)dr \\ &+ \frac{\tan\theta \sum_{l=1}^{L} \left(c_l(r_0)\sigma_l\right)^2}{\langle\sigma^2\rangle} + \lambda\left\{1 - \int_0^R K_{\text{avg.}}(r_0,r)dr\right\} \\ &+ \sum_{n=0}^{N-1} a_n \sum_{l=1}^{L} \frac{c_l(r_0)\psi_n(\nu_l)}{E_l} \end{aligned} \tag{108}$$

$$\begin{aligned} J'(c_l'(r_0)) &= \beta' \int_0^R \left\{K'_{\text{cross}}(r_0,r)\right\}^2 w(r)dr + \int_0^R \left\{T'(r_0,r) - K'_{\text{avg.}}(r_0,r)\right\}^2 dr \\ &+ \frac{\tan\theta' \sum_{l=1}^{L} \left(c_l'(r_0)\sigma_l\right)^2}{\langle\sigma^2\rangle} + \lambda'\left\{1 - \int_0^R K'_{\text{avg.}}(r_0,r)dr\right\} \\ &+ \sum_{n=0}^{N-1} a'_n \sum_{l=1}^{L} \frac{c_l'(r_0)\psi_n(\nu_l)}{E_l} \end{aligned} \tag{109}$$

where

$$\begin{aligned} K_{\text{avg.}}(r_0,r) &= \sum_{l=1}^{L} c_l(r_0) K^l_{a,b}(r), \qquad K_{\text{cross}}(r_0,r) = \sum_{l=1}^{L} c_l(r_0) K^l_{b,a}(r), \\ K'_{\text{avg.}}(r_0,r) &= \sum_{l=1}^{L} c_l'(r_0) K^l_{b,a}(r), \qquad K'_{\text{cross}}(r_0,r) = \sum_{l=1}^{L} c_l'(r_0) K^l_{a,b}(r), \\ w(r) &= (1+r)^4 \\ &= \text{weight function used to suppress near-surface structure in } K_{\text{cross}} \text{ and } K'_{\text{cross}} \end{aligned}$$

The target functions $T(r_0,r)$ and $T'(r_0,r)$ are defined according to Eq. (93), each with their own separate width parameters, Δ and Δ'. The parameters β and β' regulate the balance between optimising the averaging kernels and reducing the cross-term kernels. The parameters θ and θ' regulate the balance between optimising the averaging kernel and reducing the 1-sigma error. The parameters β, θ, Δ and β', θ', Δ' can be adjusted independently in the `Structural inversion` tab – the choice of structural function (given by the structural functions check boxes) determines which set of 3 parameters can be adjusted. The coefficients a_n and a'_n are Lagrangian multipliers used to cancel the unknown surface contributions. The solutions are then constructed as follows:

$$f(r_0) = \sum_{l=1}^{L} c_l(r_0) S_l \tag{110}$$

$$g(r_0) = \sum_{l=1}^{L} c_l'(r_0) S_l \tag{111}$$

Keeping the stellar mass constant can be used as an extra constraint provided one of the two structural profiles (a or b) is the the density profile.

4.3.6 Various error bars and kernels

As was done for the rotational inversions, it is possible to define averaging kernels, and vertical/horizontal error bars. However, since the inversion involves two unknown functions, another kernel called the cross-term kernel also intervenes. To see this, one can start by expressing the first inverted structural function as follows:

$$\frac{\delta a_{\text{inv.}}(r_0)}{a(r_0)} = \sum_{l=1}^{L} c_l(r_0) S_l \tag{112}$$

If inserted into Eq. (99), this yields:

$$\begin{aligned}\frac{\delta a_{\text{inv.}}(r_0)}{a(r_0)} &= \int_0^R \sum_{l=1}^{L} c_l(r_0) K^l_{a,b}(r) \frac{\delta a(r)}{a(r)} dr + \int_0^R \sum_{l=1}^{L} c_l(r_0) K^l_{b,a}(r) \frac{\delta b(r)}{b(r)} dr \\ &= \int_0^R K^l_{\text{avg.}}(r) \frac{\delta a(r)}{a(r)} dr + \int_0^R K^l_{\text{cross}}(r) \frac{\delta b(r)}{b(r)} dr\end{aligned}$$

where we have neglected surface contributions. From this, one can see that K_{cross} gives an idea of the amount of cross-talk coming from the second, unknown structural function $\delta b/b$. An analogous cross-term kernel can also be defined when inverting for the second structural kernel.

4.4 Mean density estimates

The mean density estimates are done in the $\underline{\rho}$ `inversion` tab using 3 different methods: a SOLA inversion, a scaling law based on the average large frequency separation $\langle \Delta\nu \rangle$, and a scaling law based on the surface correcting procedure described in Kjeldsen et al. (2008, ApJ 683, L175). These three methods can be put in the following form:

$$\underline{\rho}_{\text{inv}} = \underline{\rho}_{\text{ref}} s^2 \tag{113}$$

where

$$\underline{\rho} = \frac{M}{V} = \frac{3M}{4\pi R^3} \tag{114}$$

$$s = \frac{1}{2} \sum_{l=1}^{L} c_l \frac{\nu_l^{\text{obs}}}{\nu_l^{\text{ref}}} = 1 + \frac{1}{2} \sum_{\ell=1}^{L} c_l \frac{\delta\nu_l}{\nu_l} = 1 + \frac{1}{2} \frac{\delta \underline{\rho}_{\text{inv}}}{\underline{\rho}} \tag{115}$$

This form represents a non-linear extension to a linear inversion or a linearised scaling law, the c_l being the corresponding linear coefficients. The quantity s is the scale factor by which the reference model needs to be scaled so that linear inversion theory yields no further correction to the mean density. Using this approach has the advantage of fully retrieving the scaling laws (with a slight modification to Kjeldsen et al's approach) while allowing the use of linear inversion coefficients which can then be used to construct averaging and cross-term kernels for the 3 methods (see Reese et al. 2011, A&A, submitted). In what follows, we describe how the c_l coefficients are obtained for each of the 3 methods.

4.4.1 SOLA

The c_l coefficients are obtained by minimising the following cost function:

$$\begin{aligned} J(c_l) &= \int_0^1 \left\{K_{\rm avg}(x) - T(x)\right\}^2 dx + \beta \int_0^1 \left\{K_{\rm cross}(x)\right\}^2 dx + \frac{\tan\theta \sum_{l=1}^{L} \left(c_l \sigma_l\right)^2}{\langle \sigma^2 \rangle} \\ &+ \lambda \left\{ 1 - \frac{1}{2} \sum_{l=1}^{L} c_l \right\} + \sum_{n=0}^{N-1} a_n \sum_{l=1}^{L} \frac{c_l \psi_n(\nu_l)}{E_l} \end{aligned} \tag{116}$$

where $x = r/R$ and where $K_{\rm avg}$, $K_{\rm cross}$ and T are defined in Eqs. (124)-(126). Here, the constraint $1 = \frac{1}{2}\sum_{l=1}^{L} c_l$, imposed using the Lagrangian multiplier λ, ensures that the inversion result is exact for a homologous modification of the model.

4.4.2 $\langle \Delta\nu \rangle$ scaling law

The c_l coefficients are deduced from the following relation:

$$\frac{\delta \underline{\rho}_{\rm inv}}{\underline{\rho}} = 2 \frac{\delta \langle \Delta\nu \rangle}{\langle \Delta\nu \rangle} = \sum_{l=1}^{L} c_l \frac{\delta \nu_l}{\nu_l} \tag{117}$$

where:

$$\langle \Delta\nu \rangle = \frac{\sum_\ell (N_\ell - 1) \langle \Delta\nu(\ell) \rangle}{\sum_\ell N_\ell - 1} \tag{118}$$

$$\langle \Delta\nu(\ell) \rangle = \frac{\sum_{i=1}^{N_\ell} \left[\nu_i(\ell) - \langle \nu(\ell) \rangle\right] \left[n_i(\ell) - \langle n(\ell) \rangle\right]}{\sum_{i=1}^{N_\ell} \left[n_i(\ell) - \langle n(\ell) \rangle\right]^2}, \tag{119}$$

$$\langle \nu(\ell) \rangle = \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} \nu_i(\ell), \tag{120}$$

$$\langle n(\ell) \rangle = \frac{1}{N_\ell} \sum_{i=1}^{N_\ell} n_i(\ell). \tag{121}$$

The above definition uses a weighted average of the least-square estimates of the mean large frequency separation from Kjeldsen et al. (2008, ApJ 683, L175) for each ℓ value. Using a least-squares estimate has the advantage of not requiring consecutive radial orders to estimate the large frequency separation.

4.4.3 Kjeldsen et al's approach

The c_l coefficients are deduced from the following relation:

$$\frac{\delta \underline{\rho}_{\rm inv}}{\underline{\rho}} = 2 \frac{b \frac{\delta \langle \nu \rangle}{\langle \nu \rangle} - \frac{\delta \langle \Delta\nu \rangle}{\langle \Delta\nu \rangle}}{b - 1} = \sum_{l=1}^{L} c_l \frac{\delta \nu_l}{\nu_l} \tag{122}$$

where

$$\langle \nu \rangle = \frac{\sum_\ell (N_\ell - 1) \langle \nu(\ell) \rangle}{\sum_\ell N_\ell - 1}, \tag{123}$$

and where Eqs. (118)-(121) continue to apply. Equation (123) is an average of the frequency, such that the weighting corresponds to what is used in Eq. (118).

4.4.4 Various error bars and kernels

As was done for the structural inversions, we define averaging and cross-term kernels, and a target function, T, as follows:

$$K_{\text{avg.}}(x) = \sum_{l=1}^{L} c_l K^l_{\rho,b}(x) \tag{124}$$

$$K_{\text{cross}}(x) = \sum_{l=1}^{L} c_l K^l_{b,\rho}(x) \tag{125}$$

$$T(x) = \frac{4\pi\rho x^2 R^3}{M} \tag{126}$$

These then enter the following expression for the error, which takes into account the fact that the reference model is scaled by s:

$$\frac{\underline{\rho}_{\text{inv}} - \underline{\rho}_{\text{obs}}}{s^2\underline{\rho}_{\text{ref}}} = \int_0^1 \left(K_{\text{avg}}(x) - T(x)\right) \frac{\rho_{\text{obs}} - s^2\rho_{\text{ref}}}{s^2\rho_{\text{ref}}} dx + \int_0^1 K_{\text{cross}}(x) \frac{b_{\text{obs}} - s^k b_{\text{ref}}}{s^k b_{\text{ref}}} dx \tag{127}$$

where we have neglected surface terms, errors on the frequency shifts and other sources of error such as non-linear effects not accounted for in Eq. (99). The value of the exponent k depends on which structural profile is represented by b. If $b = \Gamma_1$, then $k = 0$, whereas if $b = c^2$, then $k = 2$ (the second case is based on the assumption that the reference model has the same radius as the observed star). An upper bound on the error (neglecting surface terms and other sources of error) can be obtained through the Cauchy-Schwartz inequality:

$$\left|\frac{\underline{\rho}_{\text{inv}} - \underline{\rho}_{\text{obs}}}{s^2\underline{\rho}_{\text{ref}}}\right| \leq \|K_{\text{avg}} - T\|_2 \left\|\frac{\rho_{\text{obs}} - s^2\rho_{\text{ref}}}{s^2\rho_{\text{ref}}}\right\|_2 + \|K_{\text{cross}}\|_2 \left\|\frac{b_{\text{obs}} - s^k b_{\text{ref}}}{s^k b_{\text{ref}}}\right\|_2 \tag{128}$$

where $\|f\|_2 = \sqrt{\int_0^1 f^2}$. The 1-$\sigma$ error bar around the inverted density, based on the measurement errors on the relative frequency shifts, is given by the following formula, which takes into account the non-linear extension:

$$\sigma_{\underline{\rho}_{\text{inv}}} = \underline{\rho}_{\text{ref}} s \sqrt{\sum_{l=1}^{L} c_l^2 \sigma_l^2} \tag{129}$$

4.4.5 Displayed quantities

Various quantities which appear in Eqs. (127) and (128) are displayed in the $\underline{\rho}$ `inversion` tab. Here are some explanations:

- $\underline{\rho}_{\text{ref}}$: density of reference model from which the inversion and scaling relations are applied.
- $\underline{\rho}_{\text{obs}}$: density of the observed "star". This is deduced from target structural profiles if they have been loaded. These are assumed to represent the true difference between the reference model and the observed star.
- $\underline{\rho}_{\text{inv}} = s^2\underline{\rho}_{\text{ref}}$: density estimate from inversion.

- $\sigma_{\underline{\rho}}$, as based on Eq. (129).
- $\delta\underline{\rho}/\underline{\rho} = \left(\underline{\rho}_{\text{inv}} - \underline{\rho}_{\text{ref}}\right)/\underline{\rho}_{\text{ref}} = s^2 - 1$: Relative variation on the mean density, deduced from inversion.
- $\|\Delta K_{\text{avg}}\|_2 = \sqrt{\int_0^1 \left(K_{\text{avg}}(x) - T(x)\right)^2 dx}$.
- $\|K_{\text{cross}}\|_2 = \sqrt{\int_0^1 K_{\text{cross}}^2(x)dx}$.
- $\int \Delta K_{\text{avg}}\delta\rho/\rho = \int_0^1 \left(K_{\text{avg}}(x) - T(x)\right) \frac{\rho_{\text{obs}}(x) - s^2\rho_{\text{ref}}(x)}{s^2\rho_{\text{ref}}(x)} dx$. True error from mismatch on averaging kernel. This formula takes into account the fact that the reference model is scaled by s.
- $\int K_{\text{cross}}\delta b/b = \int_0^1 K_{\text{cross}}(x) \frac{b_{\text{obs}}(x) - s^k b_{\text{ref}}(x)}{s^k b_{\text{ref}}(x)} dx$. True error from cross-talk. This formula takes into account the fact that the reference model is scaled by s.
- $\text{Surf}[i] = \sum_{l=1}^{L} \frac{c_l \psi_i(\nu_l)}{E_l}$. This is the ith surface terms. The number of terms displayed in the window can be adjusted by modifying the parameter `n_surf_disp` in `MeanDensityEstimate.java` and recompiling `InversionKit`.

These quantities as well as the inversion parameters can be saved in a text file using the `Save results` button.

4.5 τ and δf inversions

The τ and δf inversions allow the user to obtain the acoustic radius and an age indicator (based on the small frequency separation), respectively. These inversions are quite similar to the mean density inversions. The theory behind these inversions is described in Buldgen et al. (2015, A&A 574, 42).

4.6 Integration method

`InversionKit` also gives a choice of the integration method which is used in the different inversions. These choices have different effects depending on whether an RLS or a SOLA inversion is applied. The two options are "Sliding window" and "Chebyshev". Their effects are as follows:

- **RLS**
  - *Sliding window*: The kernels are kept on the original grid and the output inverted function(s) expressed on a smaller output grid where the points are distributed according to the inverse of the sound velocity (*i.e.*, there are more grid points where the sound velocity is smaller). Accordingly, integrations are carried out on the original (large) grid, using weights which are based on an interpolation within a sliding window of the function to be integrated (this is somewhat analogous to what is done when using finite differences to calculate derivatives). The inverted function is expressed on a b-spline basis deduced from the output grid and the regularisation matrix is based on the analytical derivatives of the b-splines. The RLS procedure then searches for optimal coefficients over the b-spline basis before calculating the values of the inverted function(s) on the output grid.

- *Chebyshev*: The kernels and the output inverted function(s) are expressed on the Gauss collocation grid associated with Chebyshev polynomials. The integrals and the regularisation matrix are all based on this grid, the coefficients being deduced from a spectral approach. The RLS procedure searches directly for the optimal function values on this (output) grid.

- **SOLA**
  - *Sliding window*: The kernels are kept on the original grid and the output inverted function(s) expressed on a smaller output grid where the points are distributed according to the inverse of the sound velocity. Accordingly, integrations are carried out on the original grid, using weights which are based on an interpolation within a sliding window of the function to be integrated.
  - *Chebyshev*: The kernels and the output inverted function(s) are expressed on the Gauss collocation grid associated with Chebyshev polynomials. The integrals are carried out on this grid using coefficients based on a spectral approach.

Remark: The "Sliding window" approach is slower than the "Chebyshev" approach but yields results which are more accurate and which do not depend on the number of grid points used in the output grid (since the integration grid does not depend on the output grid, as opposed to the "Chebyshev" approach).

5 Known bugs

Here is a list of known bugs. If you find any other, please let us know by sending us an email (daniel.reese@obspm.fr).

- excessive zooming on plots can produce irregular behaviour
- the pulsation calculations becomes unreliable beyond $\ell = 20$ due to the fact that `InversionKit` uses scaled variables in the pulsation equations.

6 Copyright notices

Below is the copyright notice that goes with `InversionKit`.


```
Copyright (c) Daniel Reese, Gael Buldgen, Sergei Zharkov, 2008, 2009, 2011, 2014, 2016

This file is part of InversionKit.

InversionKit is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

InversionKit is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
```

```
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with InversionKit.  If not, see <http://www.gnu.org/licenses/>.
```

6.1 Source code for reading fortran binary files

The source code for reading fortran binary files comes from the following web-pages:
`http://docjar.com/docs/api/org/fudaa/dodico/fortran/NativeBinaryInputStream.html`
`http://docjar.com/docs/api/org/fudaa/dodico/fortran/NativeBinaryOutputStream.html`
`http://docjar.com/docs/api/org/fudaa/dodico/fortran/FortranBinaryInputStream.html`
`http://docjar.com/docs/api/org/fudaa/dodico/fortran/FortranBinaryOutputStream.html`
and are covered by the GNU GPL2 License. They have been corrected and modified so as to meet the needs of `InversionKit`.

6.2 Source code for simplex algorithm

The code for the simplex algorithm comes from the following website:
`http://algs4.cs.princeton.edu/65reductions/Simplex.java.html`
and is covered by the GNU General Public License, version 3. It was written by Robert Sedgewick and Kevin Wayne. The algorithm is described in:

> R. Sedgewick, K. Wayne. 2011. "Algorithms". 4th edition.

Minor alterations have been included in the coded for the purposes of `InversionKit`.

6.3 Source code for a subset of the Dierckx package

The source code for the Dierckx spline-fitting package comes from the following website:
`http://www.netlib.org/dierckx/`
This code is in fortran but has been translated into java for the purposes of `InversionKit` (see notes at the start of Dierckx.java). The mathematical concepts implemented in this library are described in:

> P. Dierckx, 1993, "Curve and Surface Fitting with Splines", Monographs on Numerical Analysis, Oxford University Press.

6.4 Supplementary notices

Some of the code comes from other sources. The corresponding copyright notices are reproduced below:

Notice number 1

```
@(#)OptionPaneDemo.java      1.9 04/07/26

Copyright (c) 2004 Sun Microsystems, Inc. All Rights Reserved.

Redistribution and use in source and binary forms, with or without
```

modification, are permitted provided that the following conditions are met:

-Redistribution of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

-Redistribution in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of Sun Microsystems, Inc. or the names of contributors may be used to endorse or promote products derived from this software without specific prior written permission.

This software is provided "AS IS," without a warranty of any kind. ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE HEREBY EXCLUDED. SUN MIDROSYSTEMS, INC. ("SUN") AND ITS LICENSORS SHALL NOT BE LIABLE FOR ANY DAMAGES SUFFERED BY LICENSEE AS A RESULT OF USING, MODIFYING OR DISTRIBUTING THIS SOFTWARE OR ITS DERIVATIVES. IN NO EVENT WILL SUN OR ITS LICENSORS BE LIABLE FOR ANY LOST REVENUE, PROFIT OR DATA, OR FOR DIRECT, INDIRECT, SPECIAL, CONSEQUENTIAL, INCIDENTAL OR PUNITIVE DAMAGES, HOWEVER CAUSED AND REGARDLESS OF THE THEORY OF LIABILITY, ARISING OUT OF THE USE OF OR INABILITY TO USE THIS SOFTWARE, EVEN IF SUN HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

You acknowledge that this software is not designed, licensed or intended for use in the design, construction, operation or maintenance of any nuclear facility.

Notice number 2

Copyright (c) Ian F. Darwin, http://www.darwinsys.com/, 1996-2002.
All rights reserved. Software written by Ian F. Darwin and others.
$Id: LICENSE,v 1.8 2004/02/09 03:33:38 ian Exp $

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

THIS SOFTWARE IS PROVIDED BY THE AUTHOR AND CONTRIBUTORS ‘‘AS IS’’ AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE AUTHOR OR CONTRIBUTORS

BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Java, the Duke mascot, and all variants of Sun's Java "steaming coffee cup" logo are trademarks of Sun Microsystems. Sun's, and James Gosling's, pioneering role in inventing and promulgating (and standardizing) the Java language and environment is gratefully acknowledged.

The pioneering role of Dennis Ritchie and Bjarne Stroustrup, of AT&T, for inventing predecessor languages C and C++ is also gratefully acknowledged.