InversionKit

Version 1.1

Daniel Reese
and
Sergei Zharkov

April 2009

Contents

Acknowledgements

This program was written by Daniel Reese and Sergei Zharkov,whose work is supported by the European Helio- and Asteroseismology Network (HELAS), a major international collaboration funded by the European Commission's Sixth Framework Programme.

1 Getting started

1.1 Running the program

`InversionKit` runs under Java 5.0. 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 HELAS website:

`http://helas.group.shef.ac.uk/science/inversions/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 (rather than the default 64 MB), 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 the `InversionKit` is running, the user has several options:

- calculating theoretical frequency shifts or rotational splittings from target profiles
- invert frequency shifts or rotation splittings to find structural or rotational profiles
- combine the two options above

In order to do the above, the user must first of all:

1. load a stellar model
2. load a set of eigenmodes, which enables the program to calculate a set of kernels

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

- `Rotational Inversion`: this tab does rotational inversions and also allows the user to load a target rotation profile
- `Structural Inversion`: this tab does structural inversions on pairs of structural profiles and allows the user to load target structural profiles.

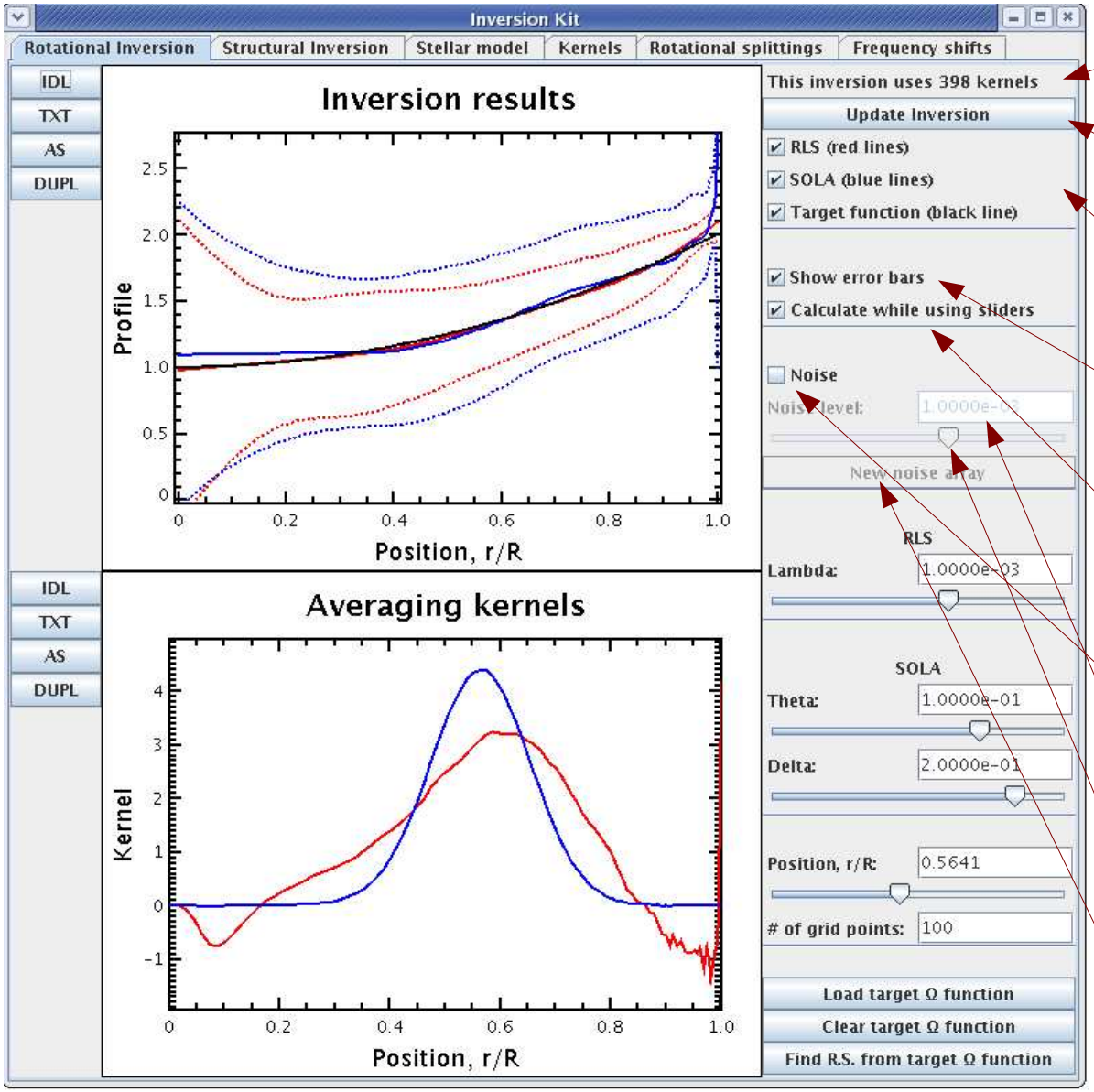
- `Stellar model`: this tab allows the user to load or generate a stellar model
- `Kernels`: this tab allows the user to load eigenfunctions and calculate the corresponding rotational and structural kernels
- `Rotational splittings`: this tab allows the user to load/generate/edit rotational splittings data
- `Frequency shifts`: this tab allows the user to load/generate/edit frequency shifts data

The next few pages gives a brief description of the different buttons and options which appear in these tabs. The `Structural Inversion` tab is similar to the `Rotational Inversion` tab; therefore only extra features are described. The `Rotational Splittings` tab is omitted altogether because it is quite similar to the `Frequency Shifts` tab. Moreover, the actual appearance of these tabs may vary from one platform to another depending on the Java installation.

1.3 Preliminary remarks

Some of the operations can take some time. For example, uploading 800 eigenmodes and calculating the corresponding kernels can take typically 40s. 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.

Also, a number of the check boxes, sliders and text fields and the `Rotation Inversion` and `Structural Inversion` tabs will automatically update the inversion if it is not up to date, whereas the `Update Inversion` button will recalculate the inversion even if it is up to date.



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

this updates/refreshes the inversion

these determine which functions are shown

this determines whether or not to show the error bars in both types of inversion

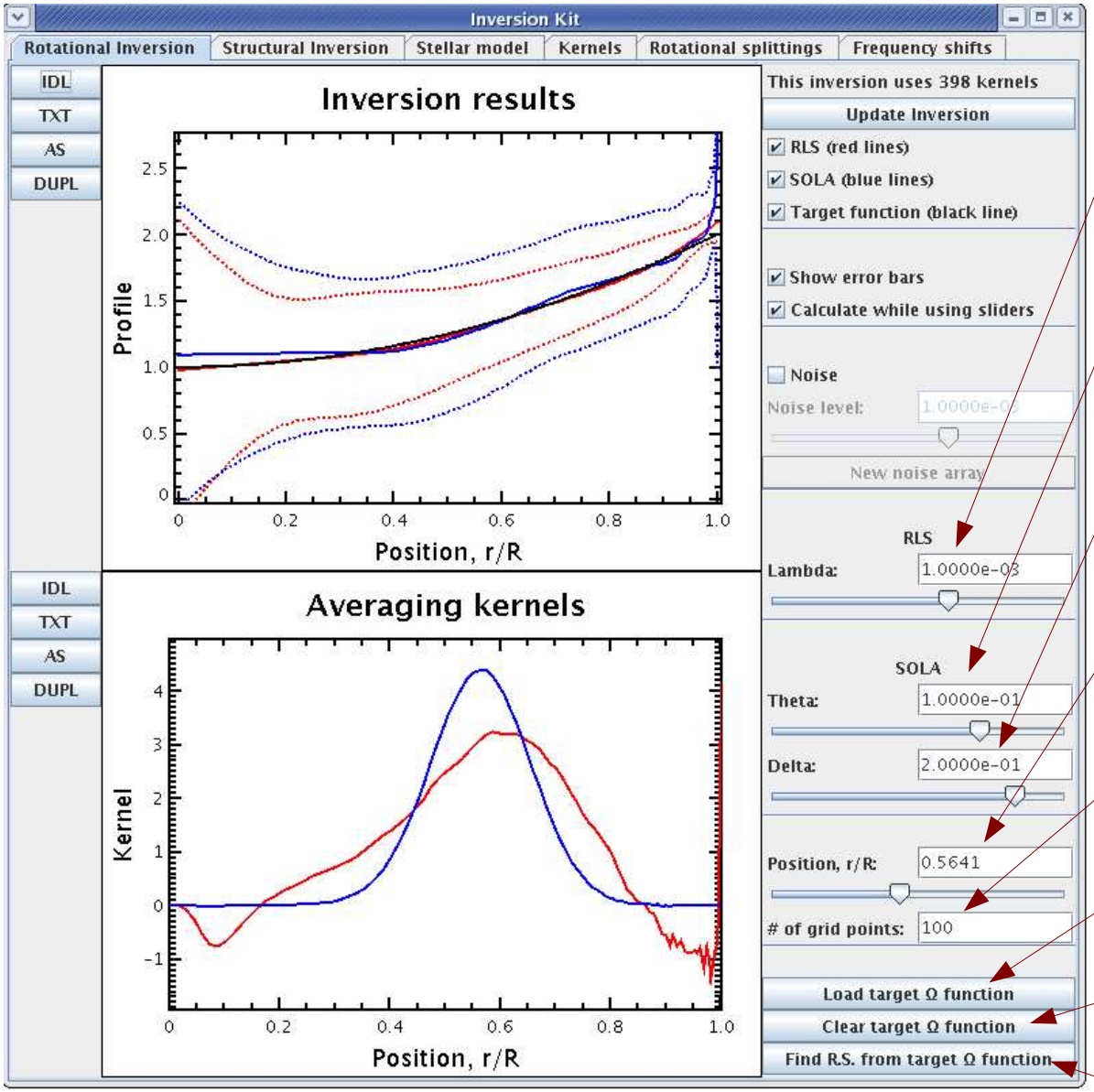
when selected, the program tries to calculate the inversion at each intermediate positions on the sliders below

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 field and slider which determine the amount of regularisation in a SOLA inversion (see Formula section)

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

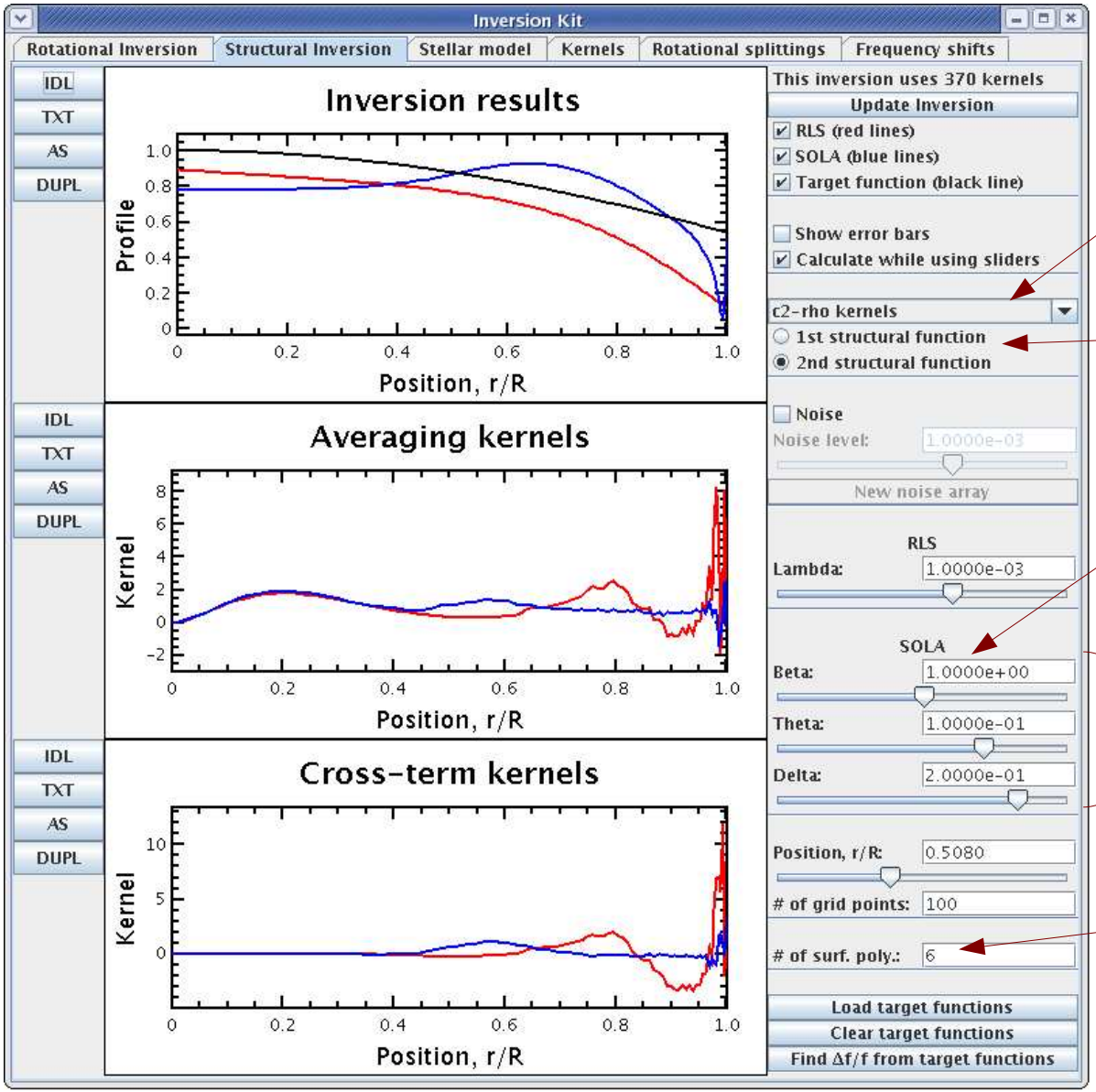
text field and slider which determine the target grid point for which to show the averaging kernels in either type of inversion

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

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

this clears the current target function

this calculates theoretical rotational splittings from the target function



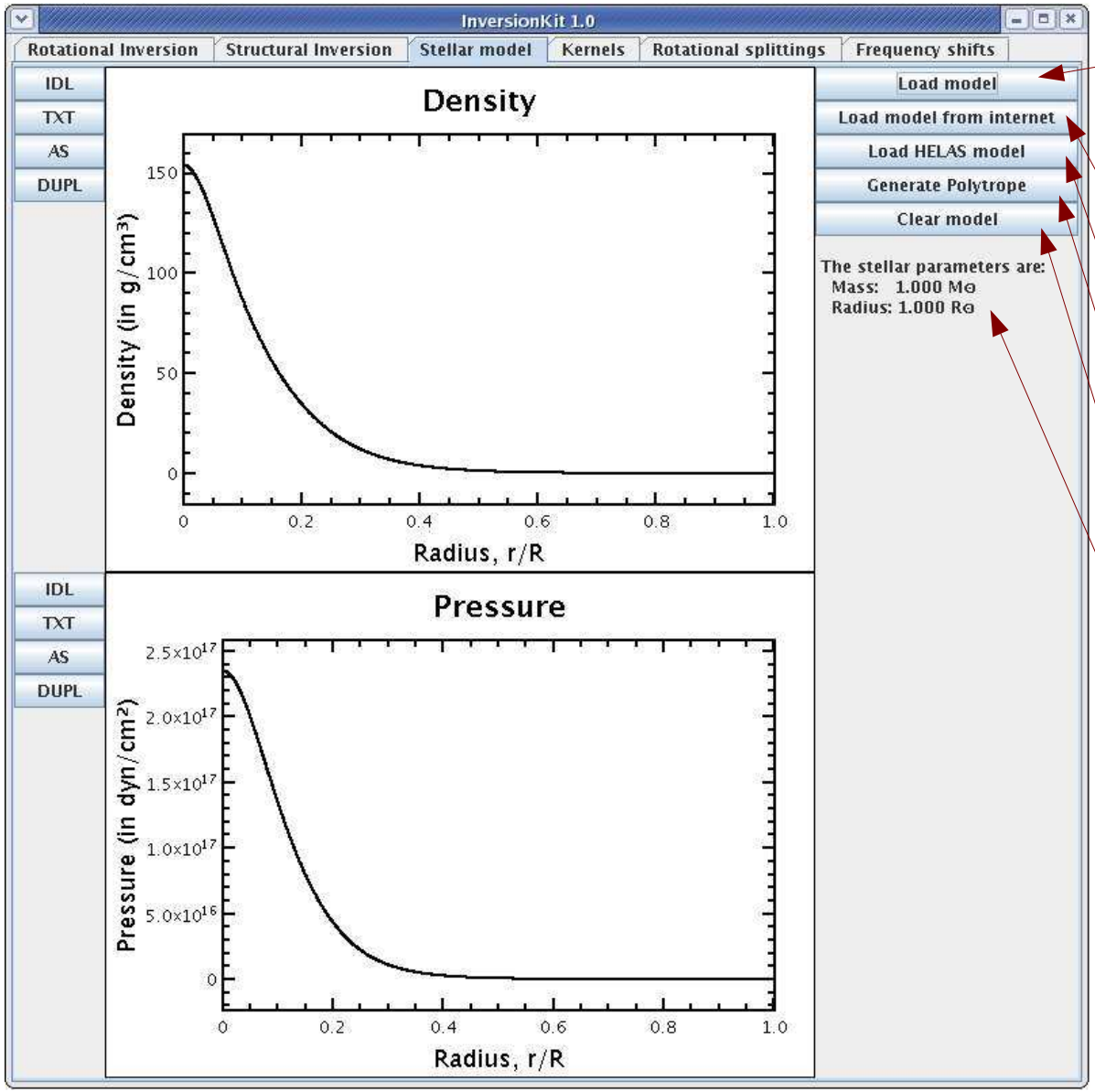
combo box which gives the choice between different types of structural kernels

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 ad-hoc modeling of surface effects (the polynomial degrees go from 0 to this value minus 1)



this loads a stellar model in OSC (CESAM), OSC.gz, FAMDL (ASTEC) or FAMDL.gz format (see File Formats section)

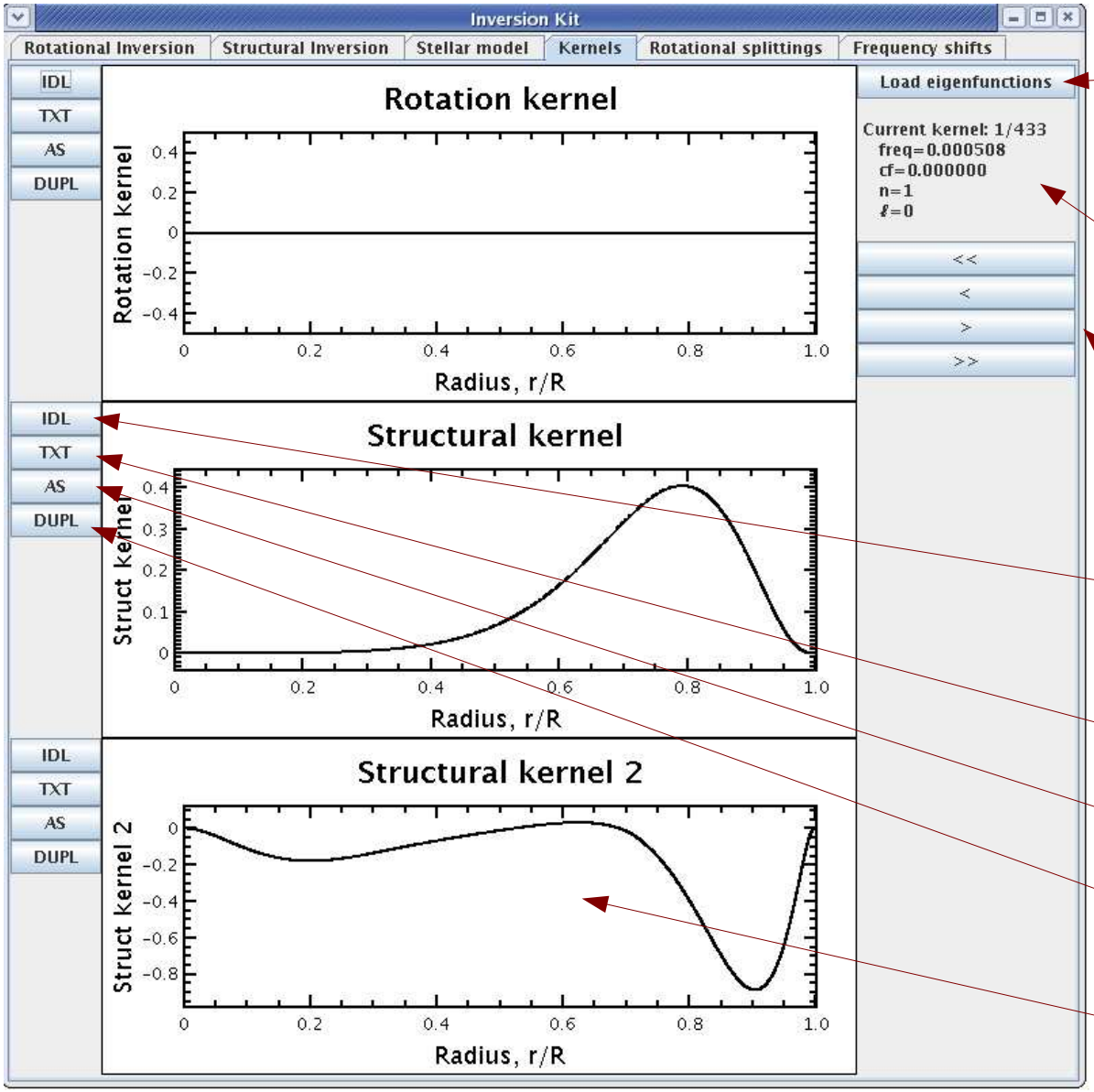
this loads a model in the same formats as above, but from a URL

this loads a model from the HELAS website

this generates a polytropic model

this removes the currently loaded model

this gives some information on the model currently loaded



this loads eigenfunctions in an FAMDE or FAMDE.gz fomat (see File Format section) and calculates all of the kernels from these

this gives information of the different eigenmodes/ kernels

these are navigation buttons so as to look at the different kernels that are loaded

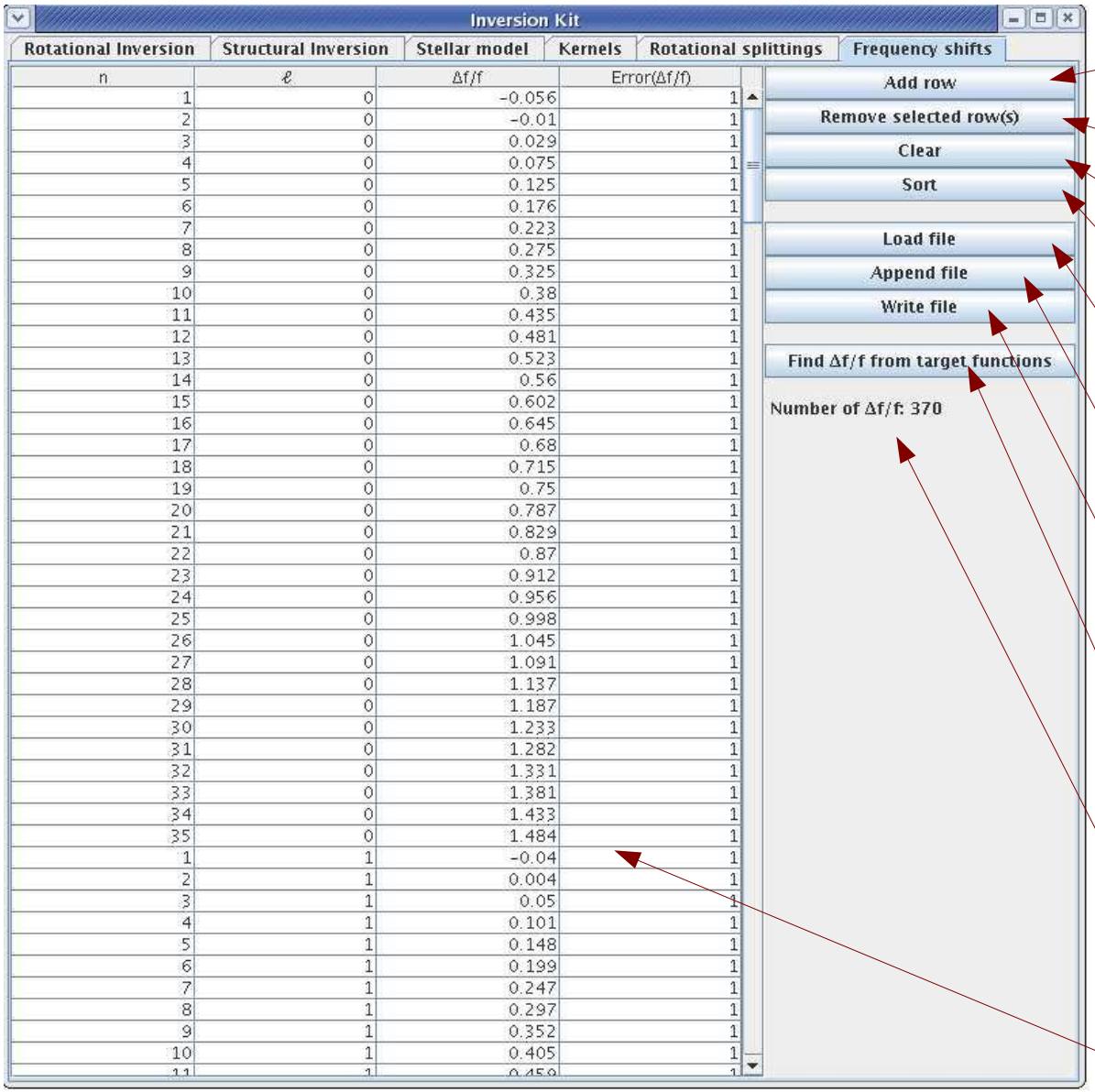
this produces a file with IDL instructions to reproduce the plot

this produces a txt file with the data from the plot

this is the auto-scale button

this duplicates the plot into a new window

click and drag on the plot area to zoom in



this adds a blank row
at the end of the list

this removes the selected rows

this clears all of the data

this sorts the data according to (l,n)

this opens a file and replaces the
current data with the file's data
(see File Format section)

this opens a file and appends its
data to the end of the current data

this writes a file with the
current data

this calculates theoretical
frequency shifts from the target
structural functions loaded
in the structural inversion page

this gives the number of
frequency shift data currently
loaded

the entries in this table can be
edited

2 File formats

2.1 Stellar models

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

1. OSC files generated by CESAM.
2. FAMDL files generated by ASTEC.

A description of these file formats can be found at:

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

`InversionKit` determines automatically which format is being used in the following way: if the word "CESAM" appears on the second line of the file, then the file is in OSC format. Otherwise, it is assumed to be in FAMDL format. If need be, `MODCONV` can convert models from one format to another. This tool is available at:

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

In both cases, `InversionKit` can read either "gzipped" files or uncompressed 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.

2.2 Eigenmodes

`InversionKit` accepts eigenmodes in one of two formats:

- the "FAMDE" format
- the FILOU format

As above, the file can either be "gzipped" or uncompressed

2.2.1 The FAMDE format

FAMDE format is an ASCII version of the AMDE format produced by ADIPLS (with `nfmode=3`) plus an additional 10-line header as described in the 1996 inversion workbench description. `InversionKit` skips the header and reads data which comes after. The data needs to obey the following rules:

- The first section is made up of one entry: the number of grid points.
- The next section contains the grid.
- The next section contains the eigenmodes. For each eigenmode there are two subsections. The first contains 50 global parameters and the second contains the normalised horizontal and vertical displacements.

- Each section and subsection 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.
- Blank lines are not allowed, except in the header.

An auxiliary Fortran tool `amde2famde.f` available with this distribution converts AMDE files to FAMDE files.

2.2.2 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 the keyword "FILOU", which is used to distinguish the file from an FAMDE file, and 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 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). The two first functions y_{01} and y_{02} are used to calculated normalised displacements which can then be used to find the different kernels. 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

`InversionKit` only accepts ASCII files which describe the target profiles. These files 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. Also, when using the target profile(s) to calculate theoretical frequency shifts or rotational splittings, the span covered by the grid needs to be at least as large as the span covered by the eigenmode/kernel grid.
- The next column(s) contain 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).

2.4 Frequency shifts or rotational splittings

`InversionKit` only accepts ASCII files which describe the frequency shifts or rotational splittings. These files obey the following rules:

- On a given line, anything following a "#" is treated like a comment and ignored.
- There are 4 columns. These are:
  1. Integer entry which corresponds to the radial order n.
  2. Integer entry which corresponds to the harmonic degree (or order) ℓ.
  3. Floating number which corresponds to either the frequency shift (Δf) or the rotational splitting (R.S.)
  4. Floating number which corresponds to either error on the frequency shift or the rotational splitting.
- 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).

3 Formulas

RLS and SOLA inversions both involve minimising "cost" functions which will be represented by the letter J in what follows.

3.1 RLS – rotational inversion

J is defined as:

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

where C_l is the rotational splitting, σ_l the corresponding error, K_l the corresponding rotational kernel, and $\left\langle \frac{1}{\sigma^2} \right\rangle = \frac{1}{L} \left(\sum_{l=1}^{L} \frac{1}{\sigma_l^2} \right)$. The function f that minimises J corresponds to the inversion result. Λ is a trade-off parameter between conforming to data and regularising the function f, and can be regulated in the Rotational Inversion tab.

3.2 RLS – structural inversion

J is defined as:

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

where $(\Delta\omega)_l$ is the frequency shifts due changes in the stellar structure, σ_l is the corresponding error, $K_{1,l}$, $K_{2,l}$ 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 $\sum_{n=0}^{N-1} \frac{a_n \psi_n(\omega_l)}{E_l}$ an ad-hoc way of modelling surface effects (where ψ_n 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 function f, 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 is added in some cases to maintain a constant mass for the star.

3.3 SOLA – rotational inversion

The idea in a SOLA inversion is to construct "nice" averaging kernels $K(r_0, r)$ at each grid point r_0. This is done by constructing a target function $T(r_0, r)$ for each r_0 and trying to make $K(r_0, r)$ resemble this function.

For a given point r_0, J is defined as:

$$\begin{aligned} J(c_l(r_0)) &= \int_0^R \{T(r_0, r) - K(r_0, r)\}^2 \, dr + \mu \tan\theta \sum_{l,k} E_{l,k} c_l(r_0) c_k(r_0) \\ &+ \lambda \left\{ 1 - \int_0^R K(r_0, r) dr \right\} \end{aligned} \tag{3}$$

where

$$\begin{aligned} T(r_0, r) &= \text{a target function,} \\ E_{l,k} &= \sigma_l^2 \delta_{l,k} = \text{the variance-covariance matrix on the measurements of } C_l \\ \mu &= \frac{L}{\mathrm{Tr}(E)}, \\ \theta &= \text{a trade off parameter between conforming to data and regularising the solutions,} \\ K(r_0, r) &= \sum_{l=1}^{L} c_l(r_0) K_l(r) = \text{the averaging kernel} \\ \lambda &= \text{Lagrangian multiplier used to insure that } \int_0^R K(r_0, r) dr = 1 \end{aligned}$$

The function $T(r_0, r)$ is defined as follows:

$$T(r_0, r) = \frac{1}{F} \exp\left(- \left(\frac{r - r_0}{c(r_0) \cdot \Delta} \right)^2 \right) \tag{4}$$

where F is a normalisation factor such that $\int_0^R T(r_0, r) dr = 1$ and Δ is a parameter which controls the width of the function. The solution f is then reconstructed as follows:

$$f(r) = \sum_{l=1}^{L} c_l(r) C_l \tag{5}$$

The parameters θ and Δ can both be adjusted in the Rotational Inversion tab.

3.4 SOLA – structural inversion

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

$$\begin{aligned} J_1(c_l(r_0)) &= \int_0^R \{T_1(r_0,r) - K_1(r_0,r)\}^2 \, dr + \beta_1 \int_0^R \{K_2(r_0,r)\}^2 \, dr \\ &+ \mu \tan\theta_1 \sum_{l,k} E_{l,k} c_l(r_0) c_k(r_0) + \lambda \left\{ 1 - \int_0^R K_1(r_0,r) dr \right\} \qquad (6) \\ J_2(c'_l(r_0)) &= \beta_2 \int_0^R \{K'_1(r_0,r)\}^2 \, dr + \int_0^R \{T_2(r_0,r) - K'_2(r_0,r)\}^2 \, dr \\ &+ \mu \tan\theta_2 \sum_{l,k} E_{l,k} c_l(r_0) c_k(r_0) + \lambda' \left\{ 1 - \int_0^R K'_2(r_0,r) dr \right\} \qquad (7) \end{aligned}$$

The functions $T_1(r_0, r)$ and $T_2(r_0, r)$ have there own separate width parameters, Δ_1 and Δ_2, respectively. The solutions are reconstructed as follows:

$$f(r) = \sum_{l=1}^{L} c_l(r)(\Delta\omega)_l \qquad (8)$$

$$g(r) = \sum_{l=1}^{L} c'_l(r)(\Delta\omega)_l \qquad (9)$$

In some cases, keeping the stellar mass constant provides an extra constraint. The parameters β_1, θ_1, Δ_1 and β_2, θ_2, Δ_2 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 is being/can be adjusted.

4 Known bugs

Here are a list of known bugs. If you find any other, please let us know by sending us an email (D.Reese@sheffield.ac.uk).

- excessive zooming on plots can produce irregular behaviour
- some of the structural kernels are not calculated correctly for high ℓ values

5 Copyright notices

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

```
Copyright (c) Daniel Reese, Sergei Zharkov, 2008

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/>.
```

5.1 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