

Instruction to the IDL Widget for Time-Distance Computation

The IDL widget, together with this instruction, is prepared specifically for SPD summer school held at Boulder, CO in July 2005. It does not include all computations of time-distance helioseismology, nor inversions. The purpose is just for students to play with helioseismology data, and get some basic ideas of what time-distance helioseismology is.

Although all the IDL widget buttons are not designed to be clicked with the same order as they appear in the widget, it is strongly suggested to do so, by follow instructions below step by step. If the widget crashes, it is recommended to quit from IDL and start over again from the very beginning.

Step by Step Instruction

1. Start the widget:

Start IDL. Type *td_exercise* to pop out the IDL widget. Basically, the left part of the widget window are a few buttons about data, power spectrum and a text window to output messages and error messages; the right part of the window are a few buttons for time-distance helioseismology; and, the middle part is a window to display images or plots.

2. Read in dataset:

Click the button *Open File*, a dialog window pops out. The default file is the one you should use, and click *Accept*. A $128 \times 128 \times 512$ Dopplergram dataset is read in, and the image of the first minute is displayed.

3. Set colors:

Click the button *Set Color*, a dialog window pops out for you to choose your preferred color index. In this case, only linear color index is recommended, such as B-W Linear, Blue/White, Red Temperature, Red-Purple, Green/White Linear ...

4. Play Dopplergram movie or display single images:

Click the button *Dopplergrams*, submenu will pop out. Choose *Movie of Dopplergrams* to play the movie. The movie will play in a separate window, and you can choose the speed of movie playing, or, you can drag the scroll bar to show separate images.

If you want to look at separate Dopplergram images, choose *Display one Dopplergram*. A dialog window pops out for you to enter which image you like to display, and the number you enter is actually corresponding to time series. As a reminder, you only have 512 minutes data, i.e., the number you input should range from 0 to 511.

5. Compute the power spectrum of Dopplergrams:

Click the button *Compute Power Spectrum*. Note that this procedure may be long, depending

on the computer speed. The $k - \omega$ power spectrum will be displayed after computation is over. The horizontal axis, k , ranges from 0 to 2.15 Mm^{-1} ; and the vertical axis, ω , ranges from 0 to 8.3 mHz.

6. Play $k_x - k_y$ power spectra movie or display single images:

Click *kx-ky Power Spectra* button to pop out submenus. Click the *Movie of kx-ky Power Spectra* to play the movie of rings. The sequence of the movie is corresponding to the temporal frequency from -8.3 mHz to 8.3 mHz , and for each image of the movie, the horizontal and vertical axes are k_x and k_y with ranges from -2.15 to 2.15 Mm^{-1} .

If you want to look at separate $k_x - k_y$ power spectrum images, choose *Display one kx-ky Power Spectrum*, and input the number you desire inside the popped window. Suggested input numbers are something near 150.

It is worthwhile to point out that the ring-diagram helioseismology is a technique to analyze such kinds of rings.

7. Play $k_x - \omega$ power spectra movie or display single images:

Click *kx-omega Power Spectra* button to pop out submenus. Click the *Movie of kx-omega Power Spectra* to play the movie. The sequence of the movie is corresponding to the spatial frequency from -2.15 Mm^{-1} to 2.15 Mm^{-1} , and for each image of the movie, the horizontal axis is k_x from -2.15 to 2.15 Mm^{-1} , and the vertical axis is ω from -8.3 to 8.3 mHz .

If you want to look at separate $k_x - k_y$ power spectrum images, choose *Display one kx-omega Power Spectrum*, and input the number you desire inside the popped window. Suggested input numbers are from 30 to 90.

8. Compute the time-distance diagram:

Click the *Compute TD Diagram* button. Note that this procedure may be long depending on the speed of computers. So, wait patiently till a time-distance diagram is displayed in the window. For the displayed diagram, the vertical scale is time in unit of minutes, and the horizontal scale is corresponding to the spatial pixels, which is about 1.46 Mm/pixel .

The computation of time-distance diagram here has avoided the lengthy computation of cross-correlations in space-time domain, but made the computation in the Fourier domain.

9. Display and fit the time-distance curves:

Once time-distance diagram is computed, we can play with the time-distance curves. Click the *Display and Fit TD Curve*, and a submenu appears. Choose the *Display a TD Curve* button to display one curve, and the number you input is corresponding to distances in unit of spatial pixels. Suggested input numbers are from 10 to 55.

To fit a curve, choose the button *Fit TD Curve*. Input the number as in the last step. Then a selected part of the curve will be fitted by the function of

$$A \exp\left(-\frac{\delta\nu^2}{4}(\tau - \tau_g)^2\right) \cos(\nu_0(\tau - \tau_p)),$$

where A is amplitude, $\delta\nu$ is bandwidth, ν_0 is central frequency, τ_p is phase travel time, and τ_g is group travel time. Fitting result is displayed in the display window. White line shows the original curve and red line shows the fitted curve, with all the fitted parameters shown in the text window.

10. Test phase-speed filtering:

Phase-speed filtering is a very important part of time-distance helioseismology when computing time-distance travel time maps of different annuli. This test enables you to input necessary parameters, and to view the corresponding $k - \omega$ power diagram and the time-distance diagram. However, the fitting code of such time-distance curves is not provided.

Click the *Test Phase-Speed Filtering* button to pop out a submenu. Again, click the *Input Parameters and Do Filtering* button to pop out a dialog window. Input the desired phase velocity and FWHM of the filter in units of km/s, and click *Accept*. Filtering starts, and this procedure may also take a while. The suggested input is something around 20 km/s for velocity and 5 km/s for the width.

A filtered Dopplergram will display after the filtering procedure is over. Click the button *Compute and Display Filtered Power* to compute and show you the filtered $k - \omega$ power diagram. Click the button *Compute and Display Filtered TD Diagram* to compute and show you the filtered time-distance diagram. Both procedures may take some computation time.

11. Compute time-distance travel time map:

This is just one example to compute the time-distance travel time map with an annulus of 3 to 6 pixels. Clicking the button *Compute TD Map* will start an external FORTRAN program to compute the time-distance travel time map. The computed travel times actually consist of six different travel times: outgoing, ingoing, east-going, west-going, north-going and south-going travel times.

12. Display time-distance travel time map:

Once the above procedure is done, you can display the travel time maps. This widget only allows you to display travel time differences, including outgoing – ingoing travel times, west-going – eastgoing travel times, and northgoing – southgoing travel times.

13. Quit:

Click the button *Quit* to quit from the IDL widget.

Explanation of Some IDL Codes

Most IDL codes that are included are independent codes that not only can be called by the IDL widget as instructed above, but also can be run independently inside IDL environment. It is recommended to read and run the following IDL codes after playing the IDL widget.

1. `compute_power, doppler_data, k_omega_power, power`

This procedure is to compute the azimuthally integrated $k - \omega$ power diagram `k_omega_power`,

as well as the power `power` with k_x , k_y and ω as its three dimensions. The `doppler_data` is the input, and can be read in by typing

```
doppler_data=readfits('data2.fits').
```

Once this computation is over, you can display the `k_omega_power` by use of an IDL procedure named `tvim`. Type

```
tvim,alog(k_omega_power)
```

to display the $k - \omega$ power diagram, and type

```
tvim,alog(power(*,*,num))
```

to display a ring, or type

```
tvim,alog(power(*,num,*))
```

to display a $k_x - \omega$ power spectrum. If you want to view movies of the rings, type:

```
xmovie,alog(power)
```

2. `compute_td, doppler_data, td_diagram`

This procedure is to compute the time-distance diagram `td_diagram` from the input dataset `doppler_data`. This computation is composed of the following a few procedures: transform the input data into Fourier domain, take a square to get power, and perform inverse Fourier transform. Then integrate azimuthally to obtain the time-distance diagram.

The `td_diagram` is two dimensional, with its first dimension as spatial pixels, and the second dimension as time. To display the time-distance diagram image, type:

```
tvim,td_diagram
```

3. `lmqt, curve, param, yfit`

This IDL procedure is to do fitting of a given time-distance curve by the form of

$$A \exp\left(-\frac{\delta\nu^2}{4}(\tau - \tau_g)^2\right) \cos(\nu_0(\tau - \tau_p)).$$

The `curve` is the curve to be fitted and selected from the time-distance diagram computed in the last step. You should type:

```
curve=reform(td_diagram(num,*))
```

to assign proper values to `curve`. The `num` indicates which curve you plan to do fitting, and should be between 10 and 55.

This fitting code calls a subroutine `lmfit.pro` that is provided by IDL library to perform the fitting. The `lmfit.pro` is a non-linear least squares fitting code employing the Levenberg-Marquardt algorithm, which is given in full details in the book *Numerical Recipe*. The returned values are saved in `param`, with `param[0]` as A , `param[1]` as $\delta\nu$, `param[2]` as τ_g , `param[3]` as ν_0 , and `param[4]` as τ_p . To view the returned values, type:

```
print,param
```

The `yfit` returns the fitted curve. In order to view how good the fitting is, you want to plot the original curve, and overplot the fitted curve to see how well they fit. To do so, type the following:

```
plot,curve
```

```
tv1ct,255,0,0,1
ntemp=where(curve eq min(curve))
xx=findgen(17)+ntemp[0]-8
oplot,xx,yfit,color=1
```

The second line of the above commands is to assign color index 1 to be red.

4. phaspeed, doppler_data, filtered_data, velo, width

This IDL procedure is a very important part of time-distance in order to map travel times at specific spatial locations. The `doppler_data` is the original Dopplergram data as in previous steps, `filtered_data` is the output datasets after performing filtering, `velo` is the desired phase speed ω/k , and `width` is the FWHM of the filter. The later two parameters, here, are in units of ω/l , different with the case when you do the IDL widget, where the unit is km/s.

This procedure is rather complicated, you need to fully understand the details of FFT (Fast Fourier Transform), how IDL performs FFT and saves all components in Fourier domain in order to understand this code. Nevertheless, to run this code, type:

```
phaspeed, doppler_data, filtered_data, 20./4.37, 5.0/4.37
```

to get a filtered dataset with velocity of 20 km/s and FWHM of 5.0km/s. 4.37 in the command line is to convert units from km/s into ω/l .

Once you obtain the filtered dataset, you can run `compute_power` to see the $k - \omega$ diagram after filtering, and run `compute_td` to see the time-distance diagram after filtering. That is, type following commands:

```
compute_power,filtered_data,p1,pow
tvim,p1^0.1
compute_td,filtered_data,td
tvim,td
```

Copyright

All IDL codes that are included in this package were written by Junwei Zhao for the SPD summer school, except that `readfits.pro` is a code in the IDL Astronomy User's Library, `tvim.pro` was written by Paul Ricchiazzi of Earth Space Research Group, UCSB, and `xcolors.pro` was written by David Fanning of Fanning Software Consulting. All rights are reserved for codes by Junwei Zhao. Accuracy and correctness of computations are not guaranteed.