Spectangular - User Manual v1.0

Daniel P. Sablowski

10th February 2019

Abstract

Spectangular is a GUI program written in C++ for spectral disentangling of SB1 and SB2 systems. It is also capable of decomposing static (telluric) features. This manual will provide a description of how to run and install the program.

Contents

Valid for version 1.0 and following until further notice.

1 Introduction

Spectral disentangling is a self-consistent method to decompose spectra of multiple systems without information about the orbital parameters. Spectangular performs this method in the wavelength space and is capable of optimising the orbital parameters such that these parameters are also found from a set of observed spectra. This user manual provides all necessary information to run the program and gives some background information on the method. It is worth noting, that methods like direct or iterative subtraction are not understood as disentangling methods. They rely on a fully analytical approach which is, due to noise, not fulfilled by real data. Hence, the result is very sensitive to the SNR. See Hadrava (2009) for further information on Fourier-disentangling and separation techniques.

There are three principal differences to Fourier-based disentangling. In Fourier-space (1) all spectra need to have the same sampling, (2) each point has the same weight and (3) the output is a periodic function of wavelength. Spectangular is coded without the need of point (3). Hence, the wavelength coverage of the output spectra will be larger compared to the input by a value defined by the radial velocities of the observations. For a more detailed analysis of the code with artifical data as well as real-life data we refer to Sablowski & Weber (2016).

For efficient matrix and vector classes as well as routines for the SVD we use the Armadillo linear algebra library, Sanderson (2010). Furthermore, we use the OpenBLAS[1] library for parallel computing and the Qt[2] library for the GUI classes.

We collect all n observed spectra in

$$\overrightarrow{o} = (\overrightarrow{o_1} \dots \overrightarrow{o_n})^t \tag{1}$$

and the output spectra of the k components in

$$\overrightarrow{x} = (\overrightarrow{x_1} \dots \overrightarrow{x_k})^t. \tag{2}$$

If we assume that the spectral resolution is sufficient (in first-order approximation given by the rotational velocity of the slowest rotating component), we can treat the problem linearly. Hence, we have to solve

$$\underline{M}\,\overrightarrow{x} = \overrightarrow{o} \tag{3}$$

for the solution vector $\overrightarrow{x}$. The transformation matrix $\underline{M}$ represents the radial velocities of each individual observation and images the solution to the observations. This (dense but large) matrix can be further separated to

$$\underline{M} = \begin{pmatrix} \underline{N}_{11} & \cdots & \underline{N}_{k1} \\ \vdots & \ddots & \vdots \\ \underline{N}_{1n} & \cdots & \underline{N}_{kn} \end{pmatrix}, \tag{4}$$

where

$$N_{ij} = \begin{pmatrix} \overbrace{0 \dots 1}^{v_{ij}} & \cdots & 0 \\ \vdots & \ddots & \vdots \\ 0 & \dots & 1 \end{pmatrix} \tag{5}$$

containes the velocity v_{ij} of observation j of component i. Singular value decomposition (SVD) is used to solve this system. SVD yields the solution of smalles residuum

$$r = \|\underline{M}\,\overrightarrow{x} - \overrightarrow{o}\| \tag{6}$$

as long as it is applied to overdetermined an rank-deficient systems. The system is overdetermined, obviously, if $n > k$. If telluric (static) lines are present, another column of matrices $N_{k+1,j}$ is added, hence $n > k+1$ is necessary. We note here, that the code does not apply heliocentric correction. Hence, in case of presense of static lines, no heliocentric correction has to be applied and the optimization (if necessary) needs to be performed on the individual velocities $v_{ij} = RV_{ij} + h_j$, where $RV_{ij}$is the radial velocity of observation j of component i and h_j the heliocentric correction of observation j.

[1] http://www.openblas.net/
[2] https://www.qt.io/

As already mentioned, Spectangular is capable of optimising the orbital parameters to find the best solution to the observations. An initial orbit needs to be set at the beginning as well as the number of iterations of the global optimization algorithm. More details will follow in Sect. 4. The implemented algorithm is the Downhill-Simplex or Nelder and Meade algorithm, Nelder & Meade (1965). This is a multidimensional global optimization algorithm and needs to be initiated first. In the case of a SB2 system, there are 7 free paramters. Hence, the algorithm will create a simplex with 8 points in the 7-dimensional space by variing the initial orbital parameters. The variation of these is not random, i.e., for a given data set and initiation the initiation of the optimization will always be the same and does not need to be recalculated. The initiation is time-consuming since SVD needs to be performed 8 times. As already mentioned it is also possible to optimise on each individual velocity. However, if, e.g., $n = 10$ there are 20 variables in case of a SB2 system, i.e., initiation performes SVD 21 times. Obviously, in most cases optimization on orbital paramters will provide faster convergence.

The procedure is (see Fig. 1): While the optimization is initiating it solves the system multiple times for several sets of variables (as discussed above). The residuum is calculated for each of the solutions and used as the value to be minimised. During each itereation, different transformations of the simplex are used, i.e., SVD is performed multiple times in each iteration. The most time-consuming transformation is the total contraction of the simplex in direction of the best point (current set of variables which gives the smalles residuum).

To find the error of the parameters, we compute the curvature matrix

$$\Gamma_{ij} = \frac{\partial^2 \chi^2}{\partial p_i \partial p_j} \tag{7}$$

according to Phillips & Eyring (1988), where $\chi^2 = \sum w_i(y_i - f_i)$ with observed values y_i, measured values f_i and statistical weights w_i. The variance-covariance matrix is thus given by

$$\epsilon_{ij} = s^2 \left(\Gamma^{-1}\right)_{ij}, \tag{8}$$

where $s^2 = \chi^2/(n - N)$ with observations n fitted to a function f with N unknowns. The error calculation implemented returns the inverse of the curvature matrix and as long as this matrix is dominated by its diagonal elements, the standard deviation of parameter p_i is given by

$$\sigma_{p_i} = \sqrt{\epsilon_{ii}}. \tag{9}$$

We have defined a quality factor which helps to plan observations and to evaluate a given set of measurements. Let $\widehat{RV}$ be the peak-to-peak value of the radial velocities and S_p the two-pixel resolution in the velocity space of the spectum, then $N_{max} = \widehat{RV}/S_p$ is the maximum number of independent spectra. The quality factor is thus defined as

$$Q_d = \frac{N_{IS}}{N_{max}} \left(1 - \frac{N_{RS}}{N_S} - \frac{N_{RS}}{N_{max}}\right), \tag{10}$$

where $N_{RS} = N_S - N_{IS}$ is the number of reproductions, N_{IS} the number of independet spectra and N_S the number of all spectra. This factor is plotted in Fig. 2 for $N_{max} = 27$ in dependence of the number of independent spectra and for different N_S. The factor is unity in case of $N_{IS} = N_{max}$ and redundant spectra are not counted. It is zero in the case that reproductions are compensated by independent spectra, $N_{IS} = N_{RS}$. The definition assumes only a small eccentricity of the system.

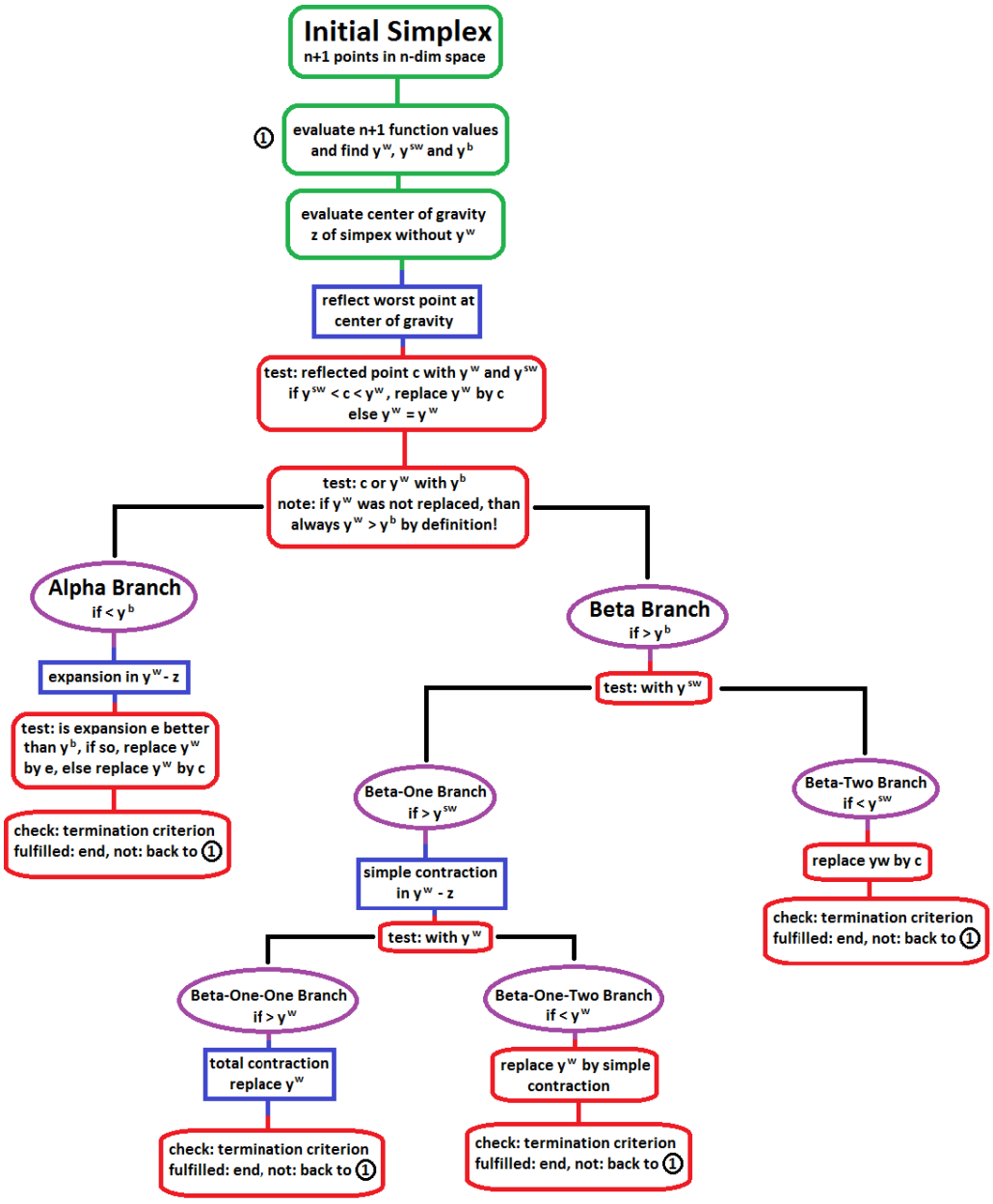

Figure 1: Downhill-Simplex-Algorithm

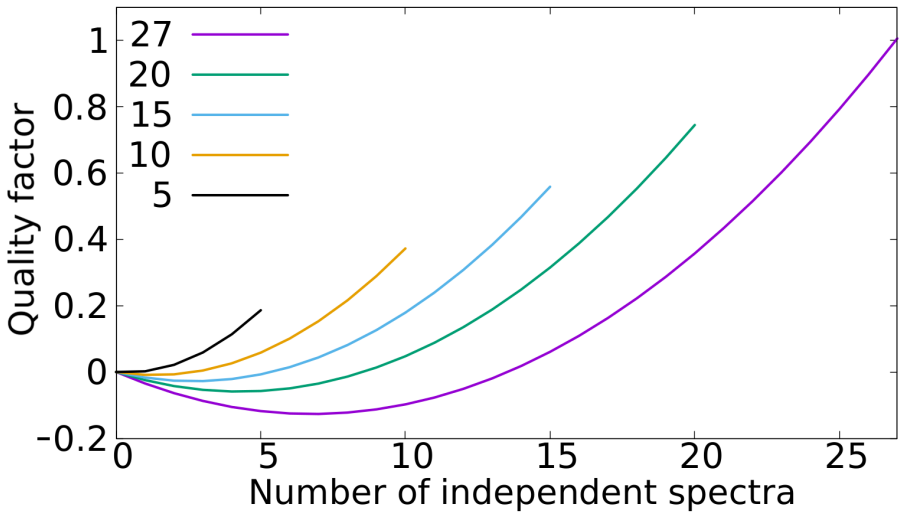

Figure 2: Quality Factor

2 Installation

It is recommended to compile the code with the Qt creator. Download the project folder at github:

Spectangular download

Download the Qt library (5.12 is recommended) and the Qt creator.

Qt download

Since it is non-commercial, everything is free. Install it, you can skip the registration step at the beginning of the installation, see Fig. 3 for more assistance. Make sure you have also installed the packages:

1. Parallel libraries OpenBLAS
2. Linear algebra library Armadillo
3. Further libraries like CCfits, libglu
4. E.g. on Ubuntu, execute
   (a) sudo apt-get install libglu1-mesa-dev* armadillo* openBLAS* CCfits*
5. Copy the spline.h include to your /usr/include directory
   (a) Spline.h
6. Now, open the Qt creator and load the project file SVD.pro from the folder you just downloaded from github. Click yes if a message box appears. Click "configure project" if it asks for it.
7. Bottom left in the main window of the Qt creator, click the big green "Play" button. Hope it compiles otherwise read compile output at bottom of the window and do what ever necessary to solve the problem.
8. Congrats

3 Observations and Input-Data Preparation

As described in the last section, the individual observations should be planned carefully in the sense of orbital sampling. In case that a rough orbital solution is known, the observations can be placed such that the radial velocities of the components differ by S_p from one to the next. As always, SNR should be high as possible and a

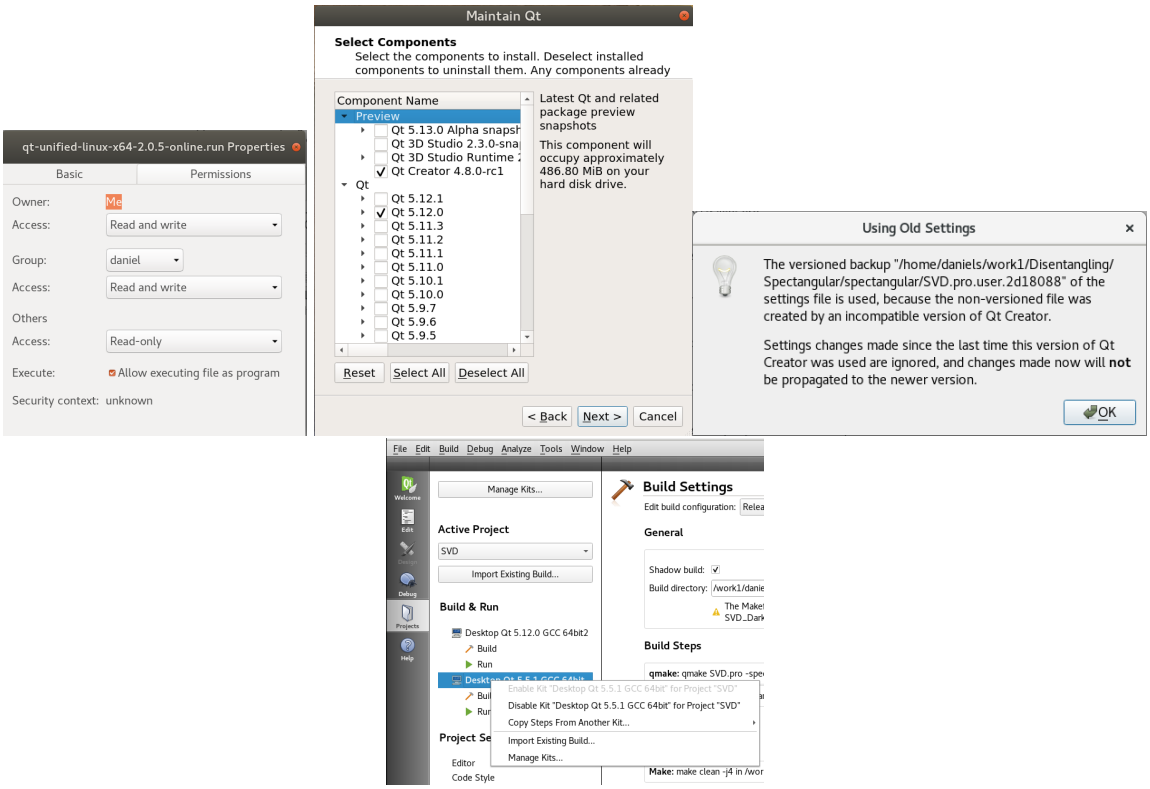

Figure 3: Top Left: Qt installer must be allowed to be executed as a program. Middle: select the newest creator and 5.12 library. Right: If there is a message about version "conflict", click yes (what else?). Bottom: In case you have multiple Qt versions, disable the older one and enable the 5.12.

compromise needs to be found for short period systems between exposure time and SNR to avoid smearing of the radial velocity information. Additionally, it is of great importance that all spectra are of equal quality in the sense of normalisation and SNR.

Care must be taken in data reduction by the wavelength calibration and continuum normalisation. All spectra need to be normalised equal and no differential defomrations of the continuum is allowed. If the reduced spectra are provided by pipeline which does not allow any adjustments, bad spectra should simply be removed. However, of course, there is always the possible of renormalisation of an "special favor" spectrum with standard data reduction packages (IRAF, MIDAS, etc.). There is also a cubic spline fit implemented to renormalise the output spectra. We will demonstrate this in Sect. 5.

An additional tool available is "CroCo", which was written for 2-dimensional cross-correlation and logarithmic resampling of the spectra. This is necessary for all input spectra. Since the shift

$$\Delta\lambda = \lambda_0 \left(\frac{v}{c}\right) \tag{11}$$

on the wavelength axis for a given velocity depends on wavelength, we need to resample to a logarithmic grid. The new wavelength is

$$\lambda' = \lambda_0 \left(1 + \frac{v}{c}\right) \tag{12}$$

and on the logarithmic scale

$$\log(\lambda') = \log(\lambda_0) + \log(1 + v/c) \tag{13}$$

the velocity leads only to an additive constant. CroCo will look for the minimum sampling

$$\delta = \min[\log(\lambda_{i+1}) - \log(\lambda_i)] \tag{14}$$

between all bins in all spectra. The sampling δ' on the logarithmic scale is adjustable by the user, who can set the increment I such that $\delta' = I\delta$.

The main window of CroCo is shown in Fig. 4. To rebin, the spectra need to be in the folder given in the line editor for the work path. All spectra need to be numbered starting at 0, e.g., spectra_0.fits ... spectra_10.fits. Currently allowed data format are fits tables and ASCII files (two columns, first wavelength second intensity). In case of fits tables the user can specify the name of the extension and the name of the columns for wavlength and intensity data. The user sets the number of spectra present in the folder (11 in our example), the common name of all spectra (spectra_) and the common name of the output. Furthermore, the user has two options to set the spectral region to be rebinned. The option, where all spectra will be cropped to equal wavelength range is the default. However, it is also possible defining a wavelength range to crop the data to a samller region of interest which, of course, will be faster and with less memory processable. All other fields and options are for the cross-correlation, which we will not describe here.

4 Run the Program

The main window is shown in Fig. 5. A short description of the numbered lines:

1. Check this box in case the RV data was generated by CroCo and an optimization is done on individual velocities. In any case, the number of spectra to be used must be set in the two spin-boxes.

2. Check this box in case the RVs are listed in two (SB2) or one (SB1) columns in an ASCII file. The values have to be ordered as the spectra are. Hence, first line of values corresponds to the first spectrum.

3. Check this box in case the RVs are computed from provided orbital parameters set in the editor (later). Click the check boxes to fix the corresponding orbital element.

4. Type in the common name and the file extension of all spectra, e.g, "spectra_" *. "fits(txt)" in case the file names are spectra_0.fits(txt) ... spectra_xxx.fits(txt).

5. You can set the flux-ratio of the components here. Constant over wavelength. However, since in most cases renormalization is necessary this correction can also be applied after the disentangling procedure. If the ratios change with orbital phase, set it for each observation via the "Editor". Optimization on these ratios for each observation can be done here as well. Orbital elements are not varied. One can choose between residuals or the slope of linear functions, defining the continuum of the disentangled spectra. Residual is more robust and leads faster to convergence. For more details see Sect. 7.

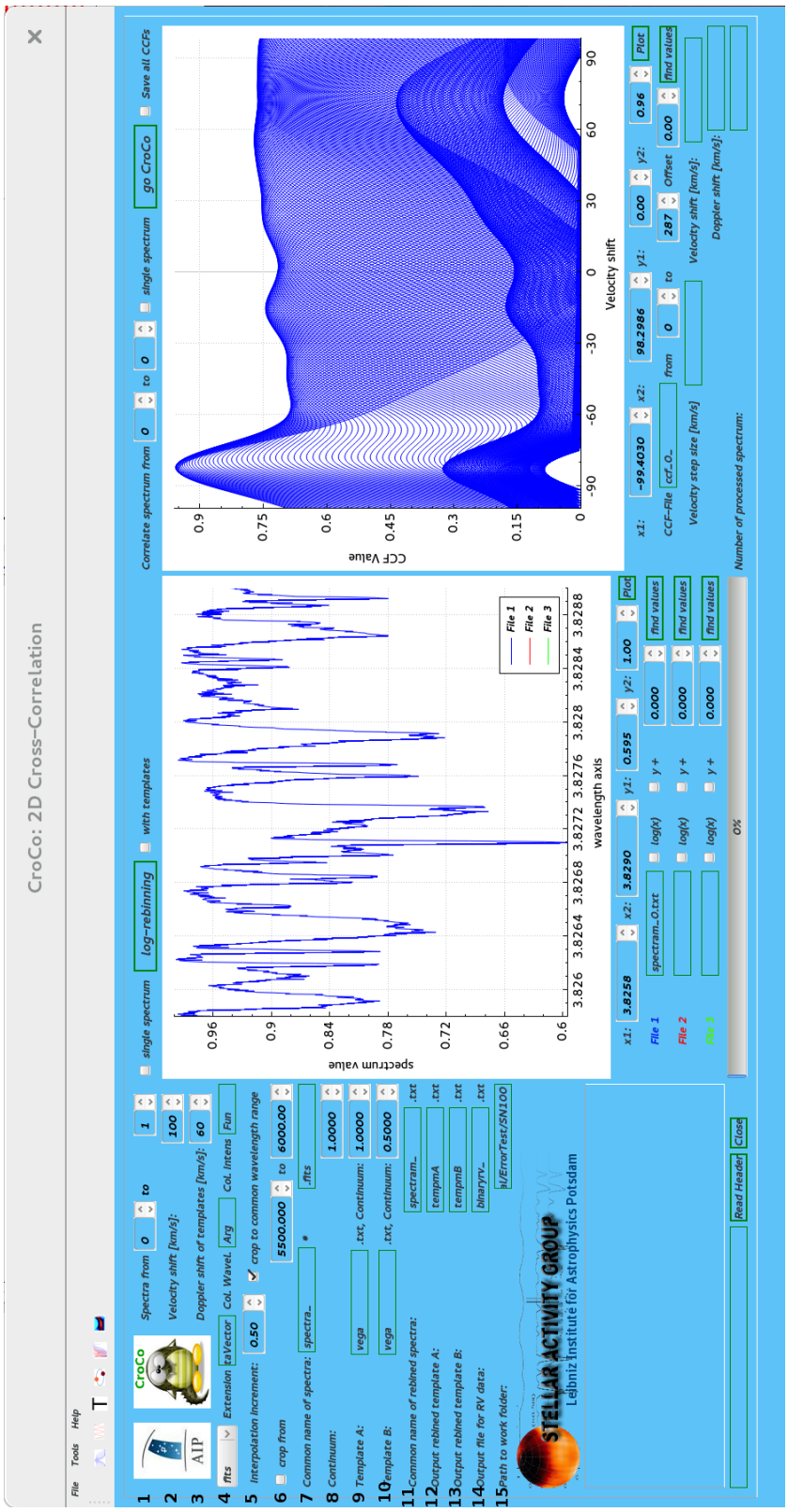

Figure 4: CroCo - main window

6. Choose ASCII if data is stored in ASCII files, choose fits if data is stored in fits tables. Type in the name of the extension in case you are using fits tables.

7. Type in the name of the column for the wavelengths and for the intensity in case that fits tables are used.

8. Type in the path to your work folder.

9. Choose SB1 or SB2, static lines in case there are tellurics or other static features (nebula emission, CCD errors). The "Read Data" button can be used to display the RVs in the text field and check data size before starting the disentangling.mIf there are just a few static features, consider to remove them from all spectra - a third component will increase the computation time drastically.

10. A single SVD can be performed. "SVD" for a full and slow SVD, "ECON" for economical SVD where one can choose between the divide-and-conquer (DaC, fastest, optimization uses this option only) and standard method.

11. Subtraction of the resulting disentangled spectra (either A or B) from the observations. The resulting spectra can be shifted by the RVs of either A or B and multiplied or added with a specific value.

12. The "Optimization" button starts the optimization. The number of interations the algoritm will perform can be set (without the necessary number of calculations for the initiation). If optimization was performed already, the user can start from the "Initial" data or "Continue" from the optimised data. "New" needs to be checked in case that optimization is performed the very first time. The "Reinitiation" is used to create a new initial simplex (from the current best parameters) and restart with optimization automatically. "Auto Stop" stops the optimization in case of no further improvement of the result after an reinitiation (initial values remain as best values).

13. The number of iterations and evaluations performed are shown. The user can choose whether to optimise on the mean residuum or the peak deviation.

14. Type a common name for the files and an extension. "Set File Names" button will apply it to the following rows, continue with points 19.

15. Type in the name of the output file for component one, two and errors.

16. Type in the name of the output file for static component, the differences between result and observations and the name for the log-file.

17. Type in the name for the initial data, first for values of the residuum, second for the values of the variables.

18. Same as 15 for optimised data

19. Computation time, mean residuum and the binning size in velocities is shown.

20. You can save and load the setting given in the window here.

21. Compute errors by fitting a quadratic function to the error surface. This will need additional SVDs. Give the name for the output file in the line edit.

As soon as we have all the spectra rebined, we need an additional ASCII file which lists all times of the indivdual observations. Furthermore, we need to set the strength of the components, f_A and f_B, in each spectrum (see Eqs. (15, 16)) and the initial orbit. This is done with the "Editor" found in the menu bar under "Tools" and is shown in Fig. 6. The tables will be filled with "1" as default when setting the number of spectra. This corresponds to an equal brightness of both components, $f_A = f_B = 0.5$, since $f_A + f_B \equiv 1.0$ for normalised spectrum.

If there are spectra from an eclipse, the relative values between the components need to be given. The data will be written to a local file when clicking the left "Apply" button. The coefficients (central field in the window) for the transformations in the optimization algorithm are changable as well as the steps on the orbital parameters to create the initial simplex. The orbital parameters are set in the right field and written to a local file by clicking the right "Apply" button. Some predefined orbits can be loaded and edited and applied. If there is a set of orbital parameters you want to have in that data base, contact the developer.

The three check-boxes in the upper left corner of the main window are used to set the initial velocities. In case that CroCo was used for cross-correlation, the first box is checked. However, in any case, the number of the first

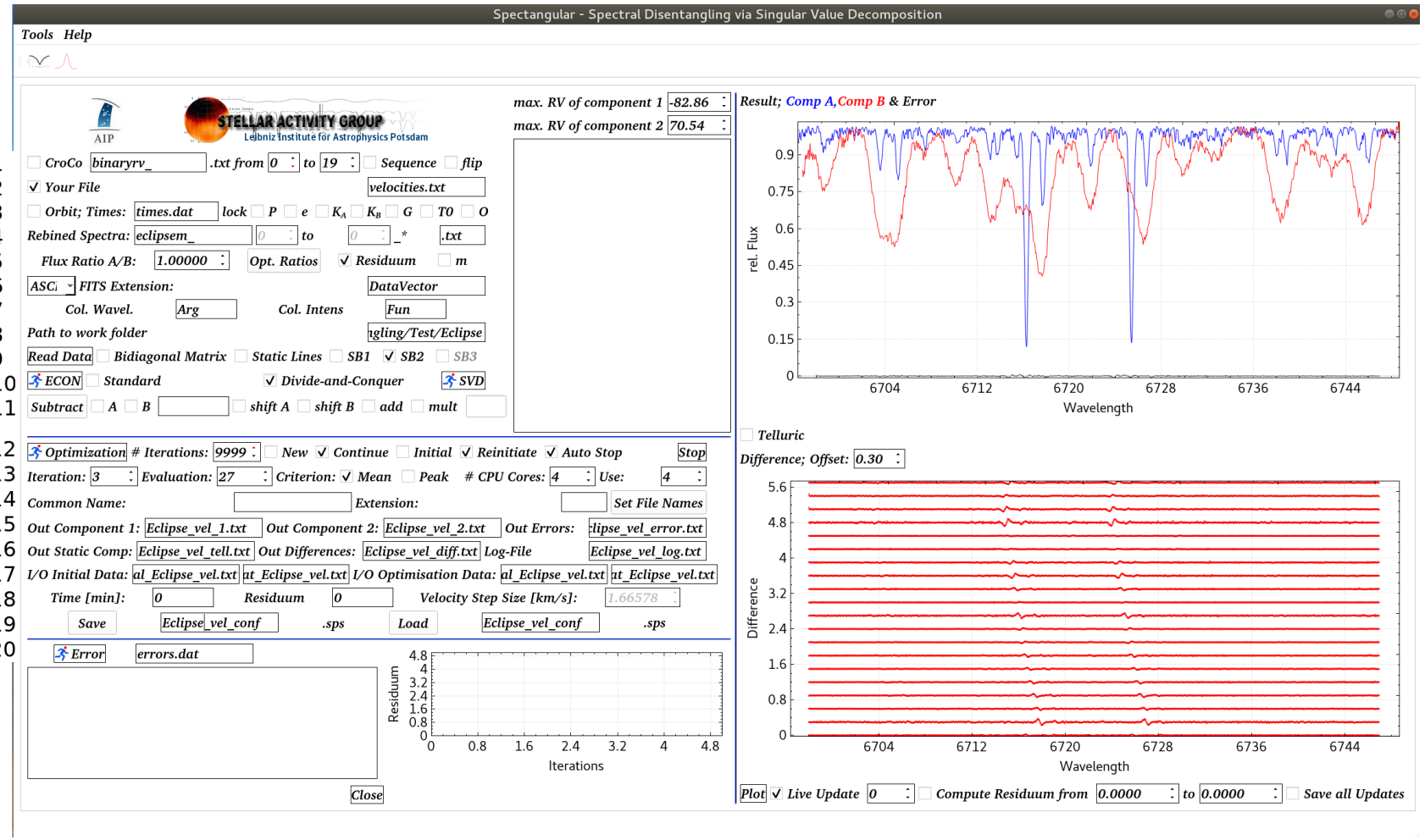

Figure 5: Spectangular - main window

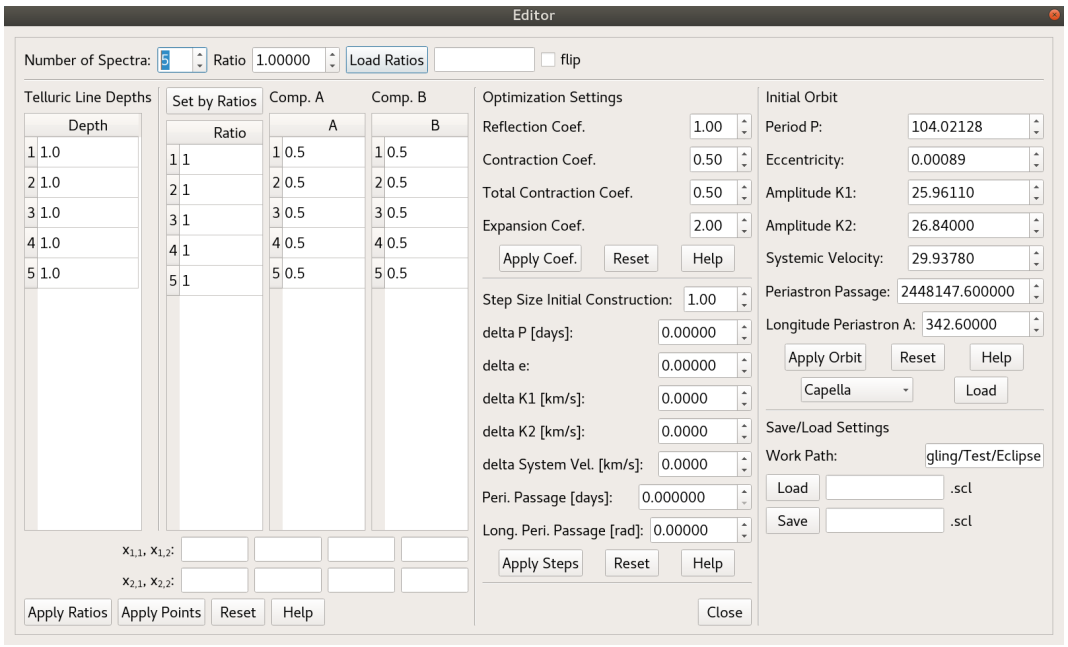

Figure 6: Editor to specify the strengths of the components in each observation, the orbital elements, step sizes for the construction of the initial simplex and the transformation coef. for the Downhill-Simplex optimizer.

spectrum and the last spectrum to be used for the disentangling need to be specified. If there is a custom ASCII file which lists the velocitites (one column in case of SB1 and two columns in case of SB2), the second check-box is used. Note that this file needs to contain as many rows as number of spectra are specified. These two options are used if one wants to optimise on the individual velocities. In case that an initial orbit is specified, the third check box is checked and the file name which contains the HJDs of each observation used is given. This file needs to be placed in the work folder and the path to that folder is given five lines further down. Currently, the user has the option to lock the period, eccentricity or time of periastron passage of the system, since it is sometimes precisely known.

The common name of all rebinned spectra is given in the next lines. Since the code is not able to get the flux ratio of the components, it can be set in the next line. However, it is also possible and sometimes necessary to adjust and renormalise the continuum of the output spectra. We will describe this in Sect. 5. If this ratio is not known "1" is the default. If data is present as txt ASCII file or fits table, set the necessary information in the next two lines. The "Read Data" button can be used to display the velocitites in the text editor and observe the necessary memory (type "top" in a terminal under LINUX), the data addresses. Further check boxes are used to choose between SB1 and SB2 systems and to enable decomposing of static/telluric features. If there are CCD errors (dead pixels etc.) or fringing which appear in the same way in all spectra this option could also help to decompose these. The name for the output (both components, telluric and differences between results and observations) are specified in lines further below.

Full SVD (slowest option) is performed by clicking the "SVD" button. An economical version (standard or applied divide-and-conquer - fastest) can be performed by clicking the "ECON" button. This can be used in case that the orbital solution is presicely known and no optimization is necessary.

If optimization is necessary and done the first time with a given set of data, the check box "New" needs to be used. If there was already an optimization run, the user can start either from the initiation data by checking the "Initial" or from the current optimization status by checking the "Continue" box. Make sure that the correct file names for the optimization data is given in the corresponding line edits further below. However, the code will check first if the file structure corresponds to the number of specified spectra and enabled optimization strategy in case that there is a user-mistake. It will give a self-explaining error message. As already mentioned, the optimization data is stored to files in the work folder and can be used in case that one wants to start from the initial data or wants to continue the optimization. The criterion used for optimization can be set to "Mean" (overall residuum) or "Peak" (largest difference of an individual bin between result and observation). In the last line, the time, overall residuum and the velocity sampling of the spectra are displayed. Since optimization will be a very time-consuming job, only the divide-and-conquer SVD is implemented for that task.

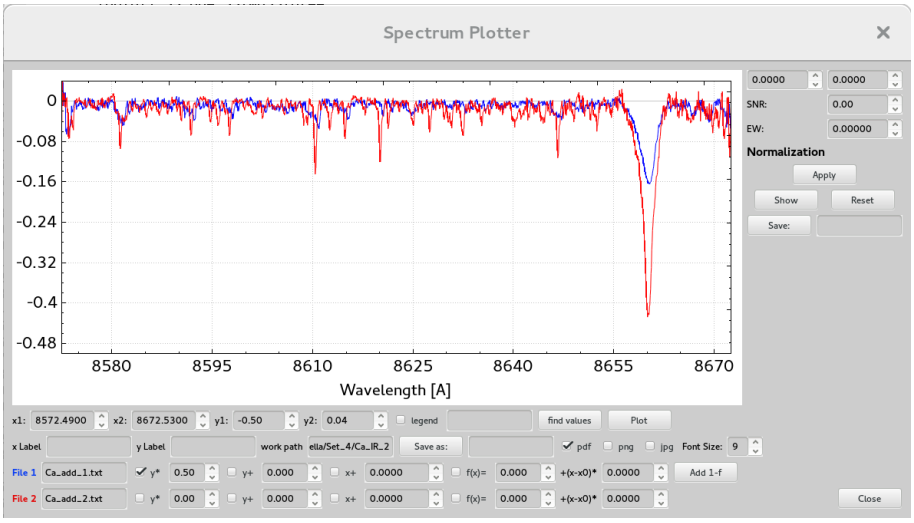

Figure 7: Plot window

Note that the GUI will update from time to time to user input (after each SVD). Hence, if you interact (even accidentally) with the interface, the command will be passed! The optimization can be aborted by clicking the "Abort" button which will happen after the current SVD is finished. The optimization data will be stored in this case and wont be lost. However, if you click (accidentally) another button, this command will be processed after the optimization is finished. Depending upon which button it was, the "Abort" button will not work. Force to quite the program if necessary.

5 Output Data

Since the output spectra are placed on continuum levels with the only constrained given to sum to the input level, they are not determined. Hence, it is not unusual that one component spectra is set to 1 while the other is at 0 in case of a continuum level of the input spectra of 1. To correct this as well as to account for the correct flux ratio between the components, we have implemented two dialog windows which we decribe now.

The first window is to plot the results and to renormalise them with a cubic spline fit[3]. The dialog window is shown in Fig. 7. The two files can be plotted and shifted on both axis for visualisation reasons, font size adjusted, axis labels changed and the graph can be saved as pdf, png or jpg file. The commands for renormalisation is at the right side of the graph window. The function to measure the equivalent width is not implemented yet. The points used for the spline are written to a local file, hence to delete old data the "Reset" button should be pushed. Only the data given in the "File 1" line will be used. A mouse click into the graph window at the continuum position writes this point to the spline file. Note that the points need to be ordered starting at lower wavelength. Otherwise it will cause the program to crash. The "Apply" button divides the current displayed (File 1) by the spline fit if the "divide" check box is checked. The "Show" button can be used to overplot the spline fit. The result is saved as ASCII file by typing the file name into the field right to the "Save:" button. Note that in cases where the continuum of the output reaches values smaller than zero a proper value need to be added to avoid dividing by a small value. This can be done with the second dialog window. The spline data is saved to a temporay file. However, this data can be saved to local file and loaded later again. Note that this will overwrite the temporary spline file.

The "Arithmetic" dialog can be found under "Tools" and is used for basic mathematics on ASCII files with two columns. This dialog window is shown in Fig. 8 and should be self-explainig. It is possible to use the same file name for the output, hence the original file be overwritten.

6 Example with Test-Data

We provide test data located in the folder "Test". These are artificial data generated with CroCo of a SB2 system and will need around 6 GB of memory. The 20 txt files are named spectra_0.txt, ..., spectra_19.txt. We use

[3]We use the include "spline.h" from http://kluge.in-chemnitz.de/opensource/spline/

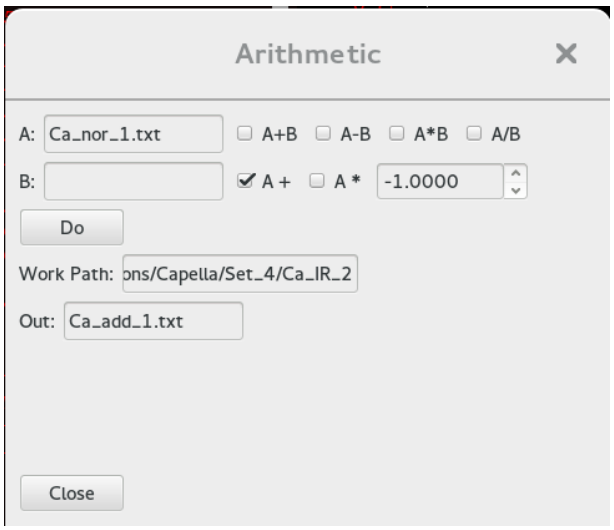

Figure 8: Arithmetic window

CroCo first to rebin the data. To do so, we set the number of spectra to 20, choose ASCII in the combo-box, check the box "crop to common wavelength range" and set a proper name in the field for the input (common name of spectra: spectra_) and output (common name of rebined spectra: spectram_) and the work path to the folder wherever it was saved after download. Pushing the "log-rebinning" button starts the process, the progress bar will show the progress and the processed spectra will be displayed in the left of the two graphs. Do display all spectra the "Sequence Plotter" under "Tools" can be used. Put in the common file name, the number of the first and last file to be plotted, the offset between each file on the y-axis and ASCII in the combo-box. By pushing the "find values" button, the program looks for the largest and smallest value in y and x for all files and writes them to the corresponding boxes. Pushing the "Plot" button will plot the files in the region given in the forementioned boxes. The graph is savable as pdf, png and jpg, font size is adjustable as well as the lables. Further details on CroCo can be found in the corresponding manual.

After resampling we change to Spectangular and set the correct file names and number. Enable the third check-box to use unknown RV option. The file with the times is also in the provided folder and named "otimes.dat". Use again ASCII in the combo box and set the correct work path. We need to check the "SB2" check-box. Set somegit g proper file names for the output and optimization data in the lower lines. Now we have to open the "Editor" window under "Tools" and apply the orbital data for Mizar. Now we can push the "ECON" button for a first run which will solve the system without any optimization.

Optimization can be started by checking the "New" check box. The number of iterations is set to 3 by default. Clicking the "Optimization" button will start the optimization on the orbital parameters. The program will now initiate the optimization. One run will take around 2.7 min on a quad-core machine. After the run is finished the "New" checkbox will get unchecked and the "Continue" box is checked. This will avoid to accidentally overwrite the initial and already optimised data.

7 Optimization on Flux Ratios

The program can also be used to optimize on the flux ratio k of the components in each observed spectrum. The orbital elements (or RVs) are kept fixed. These ratios can be specified in the "Editor" found under "Tools" in the menu bar, see Fig. 6. The component strength is calculated via

$$f_A = k/(1+k) \tag{15}$$

for the primary and

$$f_B = 1/(1+k) \tag{16}$$

for the secondary, respectively. Note that $f_A + f_B = 1$ must be fulfilled for spectra normalized to one.

In case of non-constant f in all observations, the disentangled spectra will show a tilt of the continuum, which is equal but of opposite sign. It may be approximated by two linear functions $y = m\lambda + b$. For the optimization of

the flux ratios, two options are implemented: The minimization of the residuals as usual and the minimization of the mentioned tilt. In the latter case, the code will minimize the quantity $m = \sqrt{m_A^2 + m_B^2}$. However, this criterion may be too insensitive. Furthermore, for this to work, one has to specify two points in the continuum of each of the disentangled spectra. These points are used to calculate $m_{A,B}$. It might therefore be necessary to redefine these points. See also Sablowski et al. (2019).

8 Run program without GUI

To run the program without the GUI one needs to create a configure file. In addition some changes need to be made in the code:

1. In mainwindow.cpp at around line 330, one finds:

```
1       //cout<<"Run optimization now? 0 no, 1 yes: "<<endl;
2       //cin>>runow;
3       //runow=1;
3       if(runow==1){
4               MainWindow::Input();
5       }
6       else{
7               cout<<"Open GUI."<<endl;
8       }
```

   The variable runow has to be set to 0 to run without GUI and 1 to run with GUI. In default mode runow is set to 1 (GUI mode). Two ways are possible: Either line 3 can be activated or first and second line. In the last case, the program has to be started from a terminal and will ask to set the variable runow.

   (a) Line 3 –> runow=1;

   or

   (b) Line 1 –> cout<<"Run optimization now? 0 no, 1 yes: "<<endl;
   Line 2 –> cin>>runow;

2. In main.cpp one finds:

```
1       int main(int argc, char *argv[])
2       {
3               QApplication a(argc, argv);
4               MainWindow w;
5               w.show();
6               return a.exec();
7       }
```

   Comment out line 5: w.show(); –> //w.show();

In the non-GUI mode, the program will ask for the name of the configure file in the terminal.

9 Troubleshooting and Contact

1. **After pushing the "Optimization" button, data is loaded but optimization is not performed.**
   This is a known and unresolved bug. Push the button again.

2. **Program crushes**
   Most possible this happens in case that data storage exceeds available memory or data is in wrong or disordered format. Check all files and check the used RAM of the process (under Linux type "top" in a terminal and observe "RES" as well as "VIRT". If VIRT exceeds the physical + swap, use a machine with more memory).

3. **Normalization leads to crash**
   Push the "Reset" button and/or make sure that data points are ordered with increasing wavelength.

This manual should always be consulted first. It containes all information necessary to run the program and with all references it gives all information to step into the method of spectral disentangling. Please do not contact the developer in case you have questions on the basics. However, if you have tried to solve an issue without any success by reading this manual and the cross-references contact dsablowski@aip.de. In case that you have found another reproducable bug, please send a mail to dsablowski@aip.de.

Furthermore, the developer will be happy to receive suggestions of making Spectangular more convenient.

References

Hadrava P., 2009, Disentangling of spectra - theory and practice
Nelder, J. & Meade, R. 1965, The Computer Journal, 7, 308
Phillips, G. R. & Eyring, E. M., 1988, Anal. Chem, 60, 738
Sablowski, D. P. & Weber, M., 2017, A&A 597, A125
Sablowski, D. P. et al., 2019, A&A, forthcoming
Sanderson, C. 2010, in NICTA, Australia