A Guide To WD-BASS

James Munday

March 2024

1 Introduction

The White Dwarf Binary And Single Star (WD-BASS) python package specialises in white dwarf (WD) fitting for spectral types DA/DB/DC with any number of time-series spectra. It has the capacity to fit spectra, photometry or both at the same time. Any combination of two WD spectral types are possible, e.g. DA+DA, DA+DB etc. Spectral types currently available are DA, DB, DC, DBA. Likewise, any combination of instrument setup can be simultaneously fit. It should run on the timescale of seconds to minutes. Individual spectral regions are segmented and fit with a unique and user-set normalisation. This manual explains how to obtain atmospheric solutions and fit radial velocities (RVs) for your data. There are many fields required, but hopefully WD-BASS can automate the majority of this for you. The code is expandable to any situation where the continuum is relatively smooth and you can normalise around an absorption line or a set of absorption lines at any wavelength, be it UV, optical or IR.

2 Usage and Citation

WD-BASS was first introduced in Munday et al 2024 for use with the DBL survey and the outline of the most important technical details of the package is stated in Section 4 of this work. If there are updates, I will include these by default into WD-BASS and include a change log at the end of this document.

If you use WD-BASS in your research, you must cite Munday et al 2024 and I ask that you include a footnote pointing readers to the github https://github.com/ at the same time. If you wish to develop your own code and you put lines of code or subroutines from WD-BASS your own scripts, you must cite Munday et al 2024 but there is no need to include a footnote to the github page (with the only exceptions that are freely utilisable being the load_All_Models.py, miscAstro.py and Stilism.py).

As a last request, if you have any suggestions on how to improve the function or efficiency of the code, if you find a bug, or you would like me to include e.g. a different synthetic spectral grid or new cooling sequences into WD-BASS, please contact me via email at james.munday98@gmail or raise an issue on Github. I have invested many hours into this package and I hope that it assists you in your research! I would love to talk about your research or project ideas with this code, so you are more than welcome to contact me about this at any time. I am happy to collaborate and fit your spectra upon request too.

2.1 Additional citing of model grids

- DA WD, 3D-NLTE: Munday et al 2024. With mention of Tremblay et al 2013b, 2015, Kilic et al 2021, Tremblay & Bergeron 2009. See Section 5.1.1 of Munday et al 2024 for details.
- DA WD, 3D: Tremblay et al 2013b, 2015
- DB/DBA/DC (helium-rich atmosphere): Cukanovaite et al 2021
- Gaussian/Lorentzian fitting: Nothing

3 Flowchart & YOUTUBE

The manual is quite long and is meant to be complete with every little function and option that can be input. I view the manual as something to refer to once you have started to understand the format and workflow of

the package. To facilitate use of the program in an 'out of the box' way, I have made a few youtube videos showing the typical things that users will want which are linked to on the main github page. For the general flow of the program, I have made accessible a flowchart also.

4 Installation

Type into terminal the following commands to pip install necessary python packages (pip install NAME). You may need to update some packages, or I may have tested on older packages but I will try to keep WD-BASS regularly updated. To update with pip, do: pip install -U NAME. In place of NAME, insert the following (which can also be copied and pasted from "install_notes.txt":

numpy, matplotlib, astropy, natsort, os, sys, scipy, emcee, corner, schwimmbad, extinction, numba, yaml, lmfit, astroquery, dust-extinction, pillow, shutil, pandas, requests, Python-IO

• Download the latest version of WD-BASS from github.com/ which can be placed anywhere on your local drive
• On LINUX/MAC, add to the bottom of your .bashrc folder (open terminal, enter "cd" then "nano .bashrc") the lines of code found in the file "install_notes.txt" which will make running WD-BASS much easier later.
• Close and reopen your terminal. Now you can execute WD-BASS from anywhere you like. Installation complete!

5 Inputting your data

You must supply your spectra with 2/3 columns. Column 1 is the wavelength (in Angstroms), column 2 is the flux **in erg/cm^2/s/Hz**, column 3 (optional, advised) is the flux error **in erg/cm^2/s/Hz**. If not supplied, a signal to noise ratio of 10 is assumed in the spectral region used for normalistation[1]. If you want to edit things on the backend of WD-BASS because you always have calibrated data in a different unit, see line XX of double_FitSpectrum_MCMC_multfiles_faster.py . The other exception is if you have data from X-shooter, where you can leave the fits files obtained from ESO-REFLEX as is.

I might be able to guess the format of your data already. Open a terminal and type "create_double" or "create_single" for if you want a 2 star fit or a 1 star fit. You will now have a .yaml file. Open it.

If all went smoothly, you will have something like in Fig. 1:

Now, I will go through the function of each parameter in this file. It will help you a lot to open the test .yaml files that come with installation.

spectraHa	Each spectral line for each file has its own entry. On the first line, you have all files within the directory that WD-BASS has found with create_double or create_single.
modelHa	The wavelength region that impacts a χ^2 minimisation
normaliseHa, cut_Ha_max	The wavelength range between **normaliseHa** and **cut_Ha_max** is used to linearly normalise the spectra

These are the most important things for now! In what comes next, you have to follow the same ordering as the file names input at the start. Filenames also need to be repeated in spectraHa if you want to add more lines to fit simulatneously with a common RV. Next we have:

[1]On the case by case basis, you can edit SNR=10 to whatever number you want. Search for 'predicted_SNR_of_normalise_region_per_pixel' in the double/single fitting files in the installation directory.

```yaml
spectraHa: ["/home/james/Desktop/FitDA_properly/grid_3D/dwd_fit_package/test_data_single/J192359+214103 red_3.dat", "/home/james/Desktop/FitDA_properly/grid_3D/dwd_fit_package/test_data_single/J192359+214103 red_2.dat", "/home/james/Desktop/FitDA_properly/grid_3D/dwd_fit_package/test_data_single/J192359+214103 red_1.dat"] # full path desired
spectra_source_type:  # need to INSERT your source type here. Options are 'wl_flux_fluxerr', 'Xshooter'
modelHa: [-130,130] # in angstroms
normaliseHa: [[-150,150], [-150,150], [-150,150]]  # in angstroms
cut_Ha_max: [[-200,200], [-200,200], [-200,200]]  # in angstroms
resolutions: [5000, 5000, 5000] # R=lambda/dlambda
refwave: [6562.81, 6562.81, 6562.81]
share_rv: [-1, -1, -1]  # 0 indexed!  needs all shared rvs to be at the end of the line, otherwise won't work properly!!!!!!
starType: ["DA", "DA"]  # 'DA' or 'DBA'
RV_boundaries1: [[-100,100], [-100,100], [-100,100]] # in kms-1
RV_boundaries2: [[-100,100], [-100,100], [-100,100]] # in kms-1
# please note that, where possible, the performance is about 1.4x when identical refwaves have the same normalisation/model/cut.
HJD_Values: [2458359.5157769877, 2458361.3706962164, 2458361.360167127]
forced_teff: [0, 0]   # number or 0
forced_logg: [0, 0]   # number or 0
forced_HoverHe: [0, 0] # only used if DBA is in starType. number or 0
forced_scaling: [False] # False, 'WD' or float. If false, will be added to the mcmc parameters
plot_corner: True # True or False
forced_ephemeris: [False] # expects [False] or [T0, P0]  (in days)
forced_K1: [False] # in kms-1. Options [False], [float], ['Fit']
K1_boundaries: [-300,300] # in kms-1
forced_Vgamma1: [False] # in kms-1. Options [float], ['Fit']
Vgamma1_boundaries: [-100,100] # in kms-1
forced_K2: [False] # in kms-1. Options [False], [float], ['Fit']
K2_boundaries: [-300,300] # in kms-1
forced_Vgamma2: [False] # False, 'Fit' or number
Vgamma2_boundaries: [-100,100] # in kms-1
plot_fit: [False] # [False] or e.g. [0, 1, 4] for spectra 1 2 5
fit_phot_SED: True
RA: "19:23:59.29"  # only used when fit_phot_SED is True. Hexadesimal string or a float. Set to None if not necessary
Dec: "21:41:02.1" # only used when fit_phot_SED is True. Hexadesimal string or a float. Set to None if not necessary
expected_Gmag: 15 # only used when fit_phot_SED is True. Integer/float/None
nwalkers: 25
nburnin: 10
nsteps: 10
```

Figure 1: An example_config.yaml file to be fed to WD-BASS for execution, found along with the package's installation.

resolutions	The resolution $R = \lambda/\Delta\lambda$ of each absorption line
refwave	The reference wavelength of the absorption feature of interest
share_rv	The array index for which a spectrum shares a RV with another spectrum. If this is the first entry of the file in 'spectraHa', -1 should be entered. For simultaneously fitting more than 1 spectral line with a common RV, you can link these to the initial '-1 line' by specifying the index of the '-1 line' in the array (zero indexing!). If multiple lines are being modelled with a common RV in the file, -1 should correspond to the largest wavelength value for that spectrum in 'refwave'.
NOTE	**ALL -1 values in share_rv should come at the start of the array.** (like in Fig. 1) NOTE: You can maintain a consistent RV between two absorption that originate from unique files, which could particular be useful if data has been simultaneously obtained from two arms, and this is what we did in Munday et al 2024 (see also "Advanced usage notes").

Now we start inserting things specific for each star. **The temperature of star 1 is forced to be larger than the temperature of star 2** at all times to remove degeneracy in the fitting.

starType	The spectral type of the WD. DA/DB/DC allowed. "DC" is a DC with a helium-rich atmosphere (DA includes hydrogen-rich atmosphere DCs) and a DC has a fixed H/He=10^{-5} in WD-BASS.
RV_boundaries1/2	Maximum and minimum RVs for each individual spectrum. If you are fitting multiple lines simultaneously in share_rv, the RV boundary of the absorption line that has share_rv = -1 is applied to all.
HJD_Values	The time of observation. You can choose any scale you like, so long as these are in days if you are fitting for orbital motion (otherwise, this is solely used as a number in the output of the results). I call it HJD values as I assume that HJD will be the most-common input time unit by users. If you are fitting multiple lines simultaneously in share_rv, the RV boundary of the spectrum that has share_rv = -1 is applied to all.
forced_teff	Set Teff to a number if fixed or to 0 if fit
forced_logg	Set logg to a number if fixed or to 0 if fit
forced_HoverHe	Set H/He (for a DB) to a number if fixed or to 0 if fit
forced_scaling	False allows it to vary between supplied limits. A number fixes it. 'WD' computes the radius through WD evolutionary tracks.
plot_corner	Boolean to save a corner plot. If you have many spectra with different RVs, you may not have enough RAM to process the corner plot and the script will crash. My PC has 16gb and it does not like plotting a corner plot with more than ≈ 12 free parameters.

SOME OPTIONS FOR KEPLERIAN MOTION IF DESIRED:

forced_ephemeris	Use ephemeris for Keplerian motion. Set to False (ignored) or [T0,P0] being the reference epoch and the period. Must be in the same time scale as was entered for HJD_Values.
forced_K1	Semi-amplitude of star 1 (in $\mathrm{km\,s^{-1}}$). False = ignore. A float = fixed. Fit = Fit within supplied boundaries
K1_boundaries	Boundaries of K1
forced_Vgamma1	Same details as K1 but for the velocity offset of star 1
forced_K2	Same as forced_K1 but for star 2
K2_boundaries	Boundaries of K2
forced_Vgamma2	Same details as K2 but for the velocity offset of star 2

SOME FINAL ENTRIES:

plot_fit	False or a list of spectral line array indices. If indices are entered, WD-BASS will plot the result of the desired spectrum in an interactive matplotlib figure. For all inputs, a plot of the fit will be saved regardless. If you want an interactive figure, enter the index of the (share_rv=-1) spectrum that you want to use.
fit_phot_SED	Boolean. Set to True to fit a photometric SED (see information in Section 7.2)
RA	Hexadecimal right ascension of target
Dec	Hexadecimal declination of target
Expected_Gmag	Expected Gaia Gmag (so I can try to automatically look up the redenning and parallax also enterable manually)
nwalkers	Number of independent walkers for the MCMC. 2x walkers leads to a 2x runtime
nburnin	Number of iterations until the chains in the MCMC converge. These values will later be discarded from the final results.
nsteps	Number of iterations following the burn-in phase of the MCMC that will then be used to present the results

This is the end of the list. If create_single or create_double do not work well for you, start with the example files that come with the installation of WD-BASS. Once done properly, you should be able to type run_double or run_single and it will start fitting the data. I try to report instructive error messages to fix issues if they arise. The output of the fitting will be found in a newly made directory called "**out**" found in the place where you ran the script and where your input spectra were.

6 Normalising the spectra

I recommend trying to limit the continuum contribution of the WD as much as possible and for the blue side of an absorption feature to have the same width as the red side. The reason is that if you were to model the full continuum you will obtain a photometric fit, which defeats the object of using spectra and you might as well use the photometry alone (I am oversimplifying this to make a point and the advice is solely my recommendation).

You could do a weird normalisation over multiple lines (like would be necessary for DB fitting), but for DAs it is better to split line by line. There is certainly a trade off for the normalisation region that depends on the S/N of the data – if you have low S/N data, I recommend increasing the range of the normalisation region, and vice versa for high S/N to minimise the contribution of the continuum.

Besides this and if you do not have lots of experience in spectral fitting, it is difficult to put into words how to do it well. It is a skill learnt with practice and less so theory and there is no optimal solution. Experiment and make your own opinion.

7 Atmospheric solution

7.1 Converging on a spectroscopic solution

If you know the spectral type and you are fitting a spectrum with a single star model, it is usually quite easy to start with large boundaries on each parameter and narrow in on a solution and/or increasing the number of burn-in iterations, within reason. Start by narrowing in on the temperature first and then the $\log(g)$. I assume that most people will want to fit DAs, so it should be noted that there is a symmetry of the shape of Balmer lines around roughly 15 000 K – a 5 500 K DA looks similar to a 40 000 K DA. You can easily overcome this degeneracy by including photometric fitting when available (see Section 7.2).

You should try to fit simultaneously fit as many absorption features up until something makes the lines untrustworthy, which could be e.g. the normalisation of low S/N data or the wavelength calibration at the edges of a spectrum or the flux calibration. If you are doing two-star fitting, I recommend reading through the methods section of the DBL survey in detail and once again I am happy to collaborate with fitting since two-star fits are trickier.

7.2 Fitting photometry

Initiation of this module requires fit_phot_SED: True to be entered in the yaml file. Extensive details of the photometric fitting is featured in Munday et al, 2024 and please use this is as a point of reference.

First things first, let's input the photometry.

Option 1

How I have done this in the past and what is compatible with WD-BASS is to go to CDS – http://cdsportal.u-strasbg.fr/ – type in the coordinates, scroll down to the table with the closest sources and copy the simbad name of the source into the search field again. Now, you scroll to the bottom to the photometric points and click download. For some reason, immediately clicking download will give you an incomplete file, so please wait 10 seconds before downloading. You now need to change the filename of this file to "photSED.vot". Copy this command to the directory where your spectra are and then a '/out' directory inside.

Option 2

You can manually enter the photometry yourself if you would like to also. I assume that most people will have AB magnitudes, so this is the flux scale to enter your data in.

There are 3 things you now need to include:

External_AB_mag	AB mag in filter
External_AB_mag_err	AB mag error in filter
External_Filter	Name of filter (see naming convention in filter_dict)

WD-BASS comes installed with many filter transmission curves in /Filters. The names inserted into the **External_Filter** field must be in the filter database that is installed in WD-BASS. To add a new filter transmission for a flux source that I do not recognise, you would have to input the file in the same way that is done in the filter_dict space and you would have to include the transmission function of the filter. To know if the profile is for energy/photon counters (which does make a small difference to the integrated flux), this can be found on the SVO database under 'Filter Description'. As a last thing, if photons hit the detector after moving through a vacuum (more than likely when your detector is in space), you need to include reference to the filter in the line:
if "gaia" in filt.lower() or "galex" in filt.lower() or "wise" in filt.lower():
of the function 'process_photometry_in_each_pb' in '/scripts/Fit_Photometric_SED.py'

With the photometric data input, you need the following lines in the yaml file (fields that are not bold are optional):

reddening_Ebv	E(B-V) redenning to apply
parallax	Parallax to the source (in mas)
parallax_uncertainty	Parallax uncertainty (in mas)
plot_phot_spectrum	Boolean. If you want to plot the data that is being input before running
phot_min_val	Minimum flux reject wrong flux points (use plot_phot_spectrum to examine)
want_wise1	Boolean. If WISE:W1 is available in photSED.vot, use it
want_2mass	Boolean. If 2MASS data is available in photSED.vot, use it
want_galex	Boolean. If GALEX:NUV is available in photSED.vot, use it
want_gaiadr3	Boolean. If you want *Gaia* data
ignore_filt	List of filters (featured in photSED.vot) to ignore because of bad photometry

7.3 Output

You will get a file called 'result.out' with all the parameters that you varied. The atmospheric parameters show the 50th percentile of the post-burnin posterior distribution as a first number, the second number is the

16th-quartile (minus the 50th) and the third number is the 84th-quartile (minus the 50th) to give minus/plus errors.

The RVs first show the HJD (or whatever scale you use) of the spectrum which is obtained from the yaml file (HJD_Values). Then, you have the 50th percentile of the post-burnin posterior distribution as a second number. To remove confusion with the output of the RV routine (Section 8.1.1) which should make a user error in analysing the RVs blatantly obvious, the two columns that follow you have the 16th and 84th quartiles again without subtraction of the 50th.

7.4 Advanced spectroscopic fitting notes:

Do you have two spectra of low signal to noise that have a very similar if not exactly the same RV and you wish that you had exposed for longer? Fear not! There is nothing to stop you from fitting the two together even for the same reference wavelength, which is essentially the same as processing a coadd without doing the coadding. You would want to apply this structure in the yaml file:

spectraHa: [File1_HalphaOnly, File2_HalphaOnly]
refwave: [6562.81, 6562.81]
share_rv: [−1, 0]
HJD_Values: [HJD1, HJD2]

And here, as HJD1 applies to the spectrum with share_rv −1, WD-BASS will assign both spectra the HJD values of HJD1. Use this option to your advantage to save poor data! (you may want to manually edit the HJD fields to insert the mean time of the two exposures). A unique normalisation is still applied to each spectrum, so make sure to consider the systematics induced with the normalisation procedure.

Do you want to add your own grid of synthetic spectra?
Check out the file "/scripts/load_All_Models.py" and return 4 or 5 arrays of your free variables. The fields needed are the grids of: 1) wavelength, 2) flux, 3) temperature, 4) surface gravity grid and if you have one 5) another free variable. Each wavelength has a flux, temperature, surface gravity (+extra component) attached to it. I think the best way of explaining what is necessary is to say: if you have a wavelength grid of 5000 wavelengths, 5 temperatures, 10 surface gravities that all corresponding to a flux then every array would be of length 5000*5*10.

You will then need to go into the single/double script and write a function that performs like 'return_DAgrids' (4 fields) or 'return_DBAgrids' (5 fields). Then, in the function used in the MCMC called lnlike() you need to grab the grids which will be fed into return_model_spectrum_DA (if you have 4 fields) or return_model_spectrum_DBA (if you have 5 fields, replace the HoverHe entries with the fifth component that you have). For simplicity, I would leave everything else how it is and note that in the 'result.out' output the headers will correspond to the default style still, but otherwise nothing will have changed and when first doing the .yaml file the HoverHe things all actually correspond to your 5th item.

Do you want to change the cooling sequences that I use because of improvements in physics or because of personal preference?
Take a look at the script '/scripts/importable_MTR_function.py'. Include a cooling sequence in the same format, then supply this to the main single/double_FitSpectrum scripts whenever "get_MTR()" is mentioned. Simple as that! This part can be the bottleneck of the program, so if you have a very high resolution cooling sequence then you may want to chop e.g. every other item.

8 Radial velocities (situation dependent)

RV fitting is a separate subroutine within WD-BASS that depends on everything that came before in this manual and attempts to improve the accuracy of the RVs obtained from the atmospheric solution (which are output to the file output to result.out).

8.1 Improving RVs and modelling to one spectral line

The real point of this extra module for my work has been if you have a double-lined DA+DA WD and you only are able to resolve the two at Hα, and so systematic noise in the normalisation of the other Balmer lines has a negative effect by jeopardising the RV accuracy. Furthermore, if using data from two arms and both have a different wavelength calibration, this introduces more systematics. Strongly consider if you want to use this based on your situation. I assume that if you are using single star fitting that you will benefit from all lines. With this routine, you can also narrow the parameter space for the RV allowed for each spectrum that can be fed back to the atmospheric fitting stage as it is quite a bit faster.

When you obtain a spectroscopic solution, it will be saved to 'out/result.out' and this is needed to do RV fitting of a specific spectral line. As discussed in Munday et al 2024, I try to limit the contribution from the wings of the line and narrow in on $\pm$20Åof the reference wavelength, while the renormalisation of the data still optimises the fit to the whole wavelength region between the minimum and maximum of **model_ha**. This value is changeable if you go inside the scripts 'single_FitSpectrum_MCMC_multfiles_faster2.py' or 'double_FitSpectrum_MCMC_multfiles_faster' and search for 20Å inside.

Then, a series of RV fits and an optimised linear normalisation of the data is performed. Lastly, a bootstrapping algorithm with the optimal normalisation is run to infer the RVs and RV errors (RV errors come from the standard deviation of 1000 bootstrapping iterations) in the spectrum. The upper and lower bounds of the RV search are controlled by the user-input values in the yaml file through RV_boundaries1/2.

The result of the RV fitting to each file is saved to a new folder in the directory where you ran the RV module to 'RVfits'.

8.1.1 The 3 options for RV fitting

There are 3 options that can be executed in terminal. Most of the time you want 1). In all 3 cases, PNG files are saved to the 'RVfits' folder. On the PNGs, the information shown is the RVs, the RV errors, the resolution, the limits surveyed in the bootstrap for the RVs (fed in through the yaml file). In pink are points that were clipped when searching for an optimal normalisation. In red are points that were clipped when fitting the RV.

1) **run_single_RV** or **run_double_RV**: This will take the model fit for a single star or a two star fit and search for an RV solution. This is the conventional thing to do and will suffice for most of your needs.
Output (RVfit_results.dat): The file name, the HJD (or whatever time you entered in the yaml file), RV1, RV1 error (, RV2, RV2 error).

2) **run_single_RV_gauss** or **run_double_RV_gauss**: This is the same as above, except that the synthetic models are taken and an extra Gaussian component fitted to all of the files that are not listed as 'file_ignore' in the yaml file. It will do a quick MCMC fit to get the optimum Gaussian for single star case and the optimum two Gaussian components for both stars in the double case. The purpose of this is to improve the line core for medium/high resolution data but not the wings of the absorption line, hence a maximum standard deviation for the Gaussian, $\sigma < 5$Å, is enforced. The Gaussian is fit pre-convolution, meaning that you can better fit the line core to all data of any resolution simultaneously. WD-BASS will then fit the RVs with this modified input model. Remember that it depends entirely on the atmospheric fit to start with and you should check by eye the validity of the extra Gaussian component and if it has improved the model's fit.
Output (RVfit_results.dat): The file name, the HJD (or whatever time you entered in the yaml file), RV1, RV1 error (, RV2, RV2 error).

3) **run_double_commonRV12**: Only employable if you have done a two-star atmospheric fit. This will take the synthetic spectrum of both models and fit the two of them with a consistent RV. If you have

single-lined variability in a two star source, this might be for you. I used this in routine in Munday et al 2024 for single-lined DWD RVs.
Output (RVfit_results.dat): The file name, the HJD (or whatever time you entered in the yaml file), RV1, RV1 error.

8.2 Improving RVs and modelling to all spectral lines

Really think about how independent the individual spectral lines and how significant the systematics (wavelength calibration, removal of instrument response/flux calibration) are before continuing. If you think these are not at all your limiting factors, proceed.

Above, we went through a process of optimally renormalising spectra based on the model fit. We are not doing that here. Instead, we are doing the exact same procecss as we did for the atmopsheric solution before and fixing the model spectrum instead of varying it, such that only the RVs are fitted in the MCMC. We depend on the same input yaml file as before.

8.2.1 Fitting to Keplerian orbits

This is really easy, all you have to do is fix the temperatures, loggs etc and let the program done again. Done! The output will be written to 'out/result.out'.

8.2.2 Fitting spectra one by one

Here, we are going to fit each spectrum with the same MCMC process as we did for atmospheric fitting, but we are going to fit each spectrum one by one so that no correlation mysteriously arises when fitting spectra taken at unique times. All you need to do is fix the atmospheric parameters, disable SED fitting and execute in terminal the command "run_single_one_by_one" or "run_double_one_by_one". You may also want to decrease the number of walkers, burnin iterations or steps. The result will be written to 'out/resultXXXX.out' where XXXX is replaced by the spectrum that is being fitted. At the end, all will be merged into one single file 'result_one_by_one.out'.