Manual for `TriSEPS`

Compiled in LaTeX

on November 29, 2022

Oleg Kozhura

Information present in this manual is partially covered in my thesis:

Kozhura, O (2022).

"Hierarchical triple stars as astrophysical contaminants in planetary transit surveys", viva pending.

Faculty of Science, Technology, Engineering and Mathematics

The Open University

United Kingdom

1 Prerequisites

This manual was created to assist users who want to deploy my Triple Stellar Evolution and Population Synthesis (`TriSEPS`) algorithms in the Binary Stellar Evolution and Population Synthesis (`BiSEPS`) system (Willems and Kolb 2002, Farmer, Kolb, and Norton 2013, Rowden 2019). Specifically, it focuses on understanding the codes, running them, and, if necessary, changing the catalogue, which is at the basis of the code.

Manual assumes beginner knowledge of FORTRAN, Python, bash scripting and Linux. I assume you know how to read error messages and do basic debugging of the code. I also assume readers have a basic understanding of stellar multiplicity, but in any case, it is recommended to read Duchêne and Kraus (2013), Raghavan et al. (2010), Toonen, Hamers, and Portegies Zwart (2016) for basics.

It is **highly** recommended to familiarise yourself with the "`BiSEPS` PLATO Manual" (Bray, 2021) as it contains information crucial for running main `BiSEPS` algorithms (this document is located in the same directory as the current manual).

One could also consult with my thesis (Kozhura, 2022), but, I am assuming, you are more interested in trying to run the code ASAP rather than reading 50k words on distinguishing planets from stars. Nonetheless, it is **highly** recommended to consult with the "Chapter 4: Extending `BiSEPS` to triples", as it discusses biases, assumptions and techniques in more detail. The thesis is also contained in the same directory as the manual under the title "Kozhura_thesis.pdf".

If you have any questions not covered by this and other manuals, contact me at
`oleg.kozhura@open.ac.uk`

2 Editing and running the code

To adapt my codes to your needs, you will only edit two codes: `crossmatching.f90` (in FORTRAN 90) and `pdf.py` (in Python). If you want to change the catalogue at the basis of the code, you would only need to edit `pdf.py`, as `crossmatching.f90` is not affected by any catalogue changes. Therefore, the main focus of this manual is on the `pdf.py`. Nonetheless, in Section 5, I provide a short description of `crossmatching.f90`. Either way, both codes are contained within the "create_triples" directory in the home `BiSEPS` directory:

```
\\stem-linux-homes\Plato\data\BiSEPS_LSF\create_triples
```

There is a variety of text editors available, but personally, I recommend `emacs`, which can be used to open a document using the following command: `emacs document.txt`
To close the document, use: Ctrl+x, followed by Ctrl+c.

You can also use other editors such as `vim`, `vi`, `nano`, Notepad++ (I use screenshots from it in this manual), or even just Notepad! Just make sure you save your edits (and keep the original backups handy).

As I mentioned, this manual assumes basic knowledge of Python and FORTRAN, but I provide examples of running things on the OU CentOS 7 Cluster specifically below.

To run any Python codes, you must first input the following:

```
source /STEM/software/astronomy/anaconda3/etc/profile.d/conda.sh
Conda activate base
```

This enables the Python environment and allows you to run Python3 codes. After that, you can run Python code with: `python3 pdf.py`

Unlike Python, FORTRAN is a compiled language, which means that **if you change the code, it needs to be recompiled for your changes to work**. Luckily, a bash file in the same directory will do it for you. So, if you made changes to the `crossmatching.f90`, just type: `./recompile_crossmatch.sh`

It uses `GFortran` compiler, and as a result, you will get a new executable file `crossmatching.exe`. To run it, use the command: `./crossmatching.exe`

If you are rerunning the `pdf.py`, you **have** to rerun `crossmatching.py` as well to create a new library of triples. The library will automatically be suitable for `BiSEPS`, which will take all necessary libraries by itself. Again, for details on running `BiSEPS` refer to "`BiSEPS` PLATO Manual" (Bray, 2021) in the same directory.

3 Creating triples: overview.

In population synthesis, it is not feasible to have a very detailed stellar evolution, as every added parameter and every decreased time-step increases the runtime for each system, and we need to remember that we require many thousands of systems for population synthesis purposes. For example, one square degree area far from the Galactic plane, e.g. around $l = 228°$and $b = -55°$, would take around 3–4 hours to run on the Open University's CentOS 7 computer cluster. Even if we could sacrifice computing time in favour of the more detailed triple evolution, `BiSEPS` algorithms and Hurley, Tout, and Pols (2002) evolutionary prescriptions are, at their basis, incompatible with triple evolution and, in the end, there is no guarantee that detailed triple evolution would produce results more statistically accurate than an alternative approach.

Therefore, I have decided not to evolve triples from scratch but to have their current properties in the model based on observations and their probability density functions (PDFs). For this, I use the catalogue of triple and quadruple stars (Tokovinin 2008), which has a selection of well-defined triple and quadruple systems with accurate estimates of masses and periods.

4 Probability density function code.

The current PDF code is based on Andrei Tokovinin's catalogue of triple and quadruple stars (Tokovinin 2008), which has 724 triples with reasonably accurate estimates of parameters such as mass and periods. These triples were, in turn, chosen from the Multiple Star Catalogue (MSC) (Tokovinin 1997).

Since we do not want to evolve triples from scratch, the best way to produce them is to create triples based on observed distributions. The whole process is summarised in Figure 1, but let us go into detail.

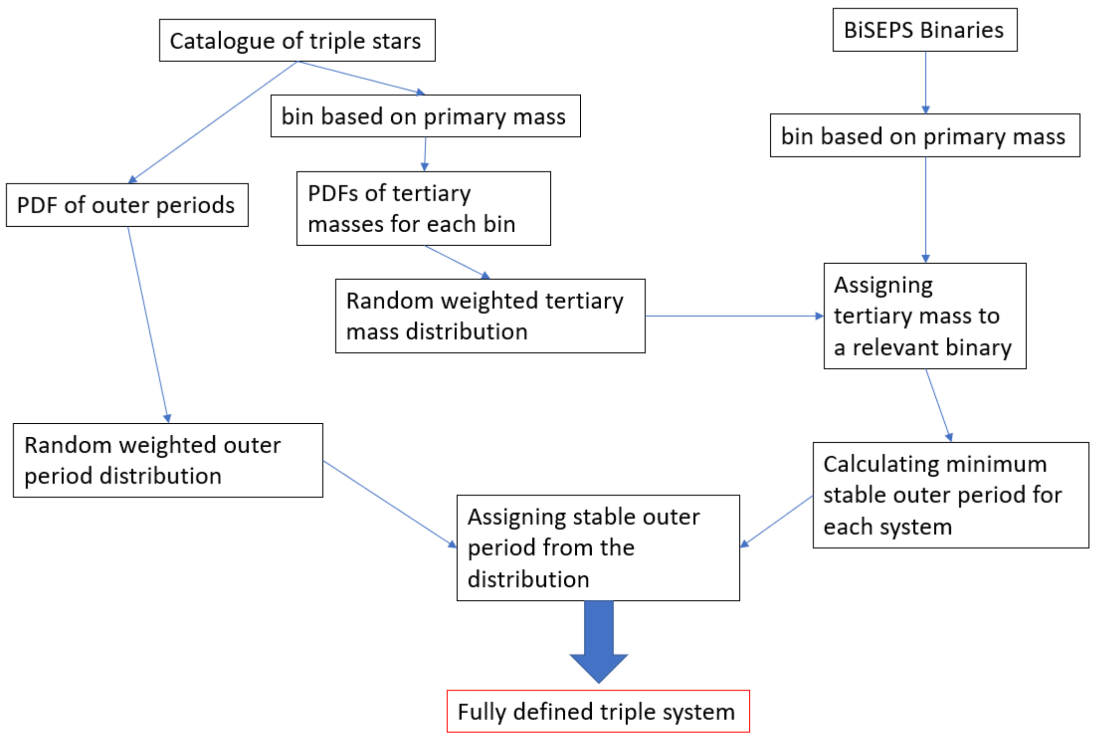

Catalogue of triple stars
bin based on primary mass
PDF of outer periods
PDFs of tertiary masses for each bin
Random weighted tertiary mass distribution
Random weighted outer period distribution
BiSEPS Binaries
bin based on primary mass
Assigning tertiary mass to a relevant binary
Calculating minimum stable outer period for each system
Assigning stable outer period from the distribution
Fully defined triple system

Figure 1: Diagram describing the creation and selection of the expectant tertiary parameters. We merge BiSEPS binaries with expected tertiary parameters from Tokovinin (2008).

I split that catalogue into five groups based on the mass of the primary star: primaries under one $M_\odot$, between 1 and $3M_\odot$, between 3 and $5M_\odot$, 5 and $12M_\odot$ and greater than 12 $M_\odot$. The choice of bins was rather empirical to ensure that all bins have a reasonable number of systems representing the group's statistics.

After that, I take each group separately and create PDFs from their tertiary masses. As a result, I have 10 PDFs that represent the likelihood of different tertiary masses for any inner binary based on the mass of its primary. A similar method is used with outer periods but without separating them into mass bins. It was found that the dependence of outer periods on other parameters is rather weak, so I took the whole sample and built a PDF of outer periods based on all triples. This argument is supported by Tokovinin (1997), who states that there is no dependence of the outer period on the primary mass.

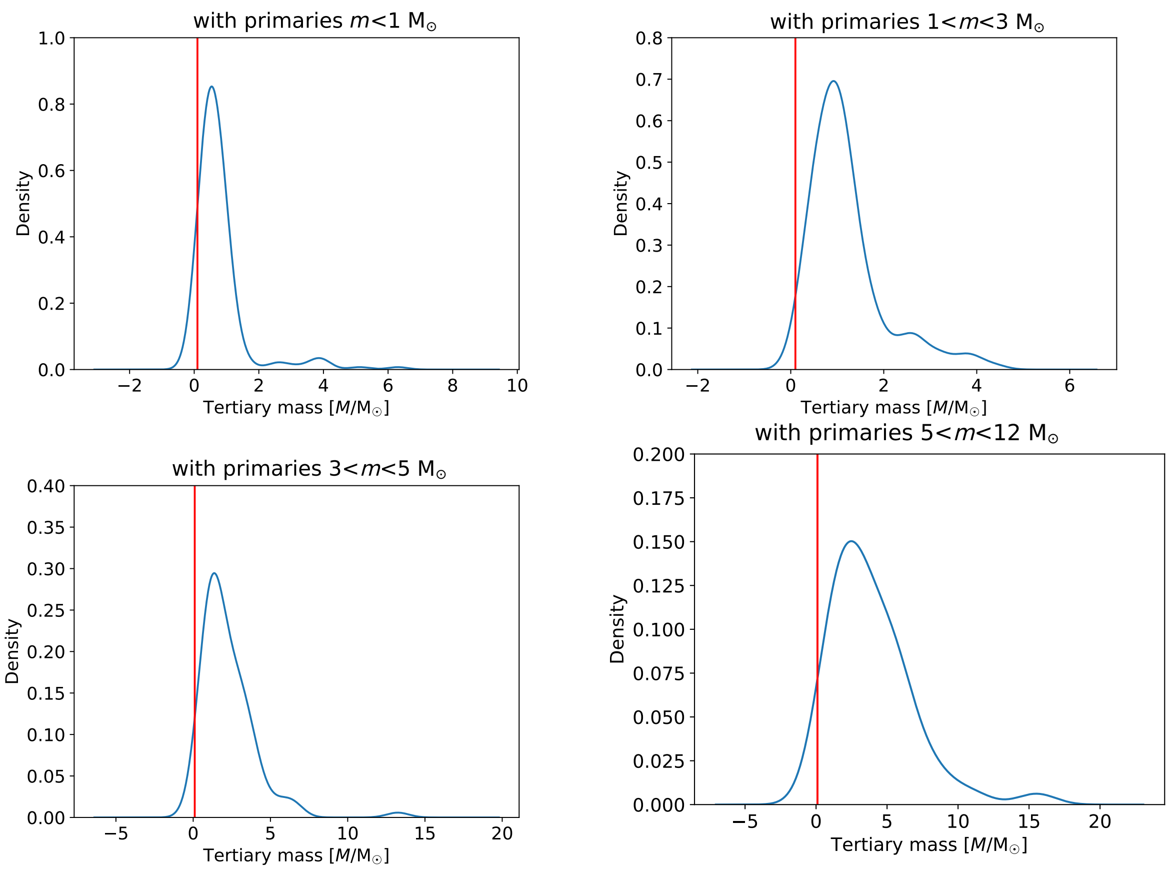


Figure 2: Sample of PDFs based on the catalogue. Default PDFs are not related to any physical properties, which is why they sometimes show negative mass. In the post-processing, we adjust the minimum value for mass to be 0.1 M$_\odot$.

Of course, the raw PDF is a simplification, not related to any physical properties, because sometimes the mass goes to negative values, for example. This is obviously unphysical, but it does not affect the results because we set the minimum mass at 0.1 M$_\odot$, the smallest mass that the `BiSEPS` is coded to evolve.

Once we have all the necessary PDFs, the code takes a set of binary stars previously evolved by `BiSEPS` and splits those binaries into the same five bins based on the mass of the primary component. Next, it takes a relevant mass PDF and assigns a random tertiary mass for each binary.

However, the process is a bit more involved when it comes to periods. At first, we follow the same approach as with masses: take the BiSEPS binary, take the universal PDF with periods, and assign a period randomly. But it can not guarantee that the resultant systems will be stable. Therefore, we need to deploy the stability criteria to check whether the combination of assigned masses and periods results in stable or unstable systems.

The stability criterion, which is most commonly used, is based on chaotic and resonance criteria, as described by Mardling and Aarseth (1999) and Toonen, Hamers, and Portegies Zwart (2016). For my purposes, I take a version of the same criteria from Sterzik and Tokovinin (2002), who rewrote that formula in terms of the inner period ($P_{\rm in}$), using Kepler's third law and assuming coplanar orbits.

I simplify it further, as we assume circular orbits and adapt it to produce the lowest value of the outer period ($P_{\rm out}$). For simplicity, I also rewrite periods in $\log(P/\text{days})$, allowing us to separate the logarithms in the following equation, where q_{out} is the mass of the tertiary component over the sum of masses of inner binary components:

$$P_{out} = P_{in} + 0.1\log(1 + q_{out}) + 0.670737047 \tag{1}$$

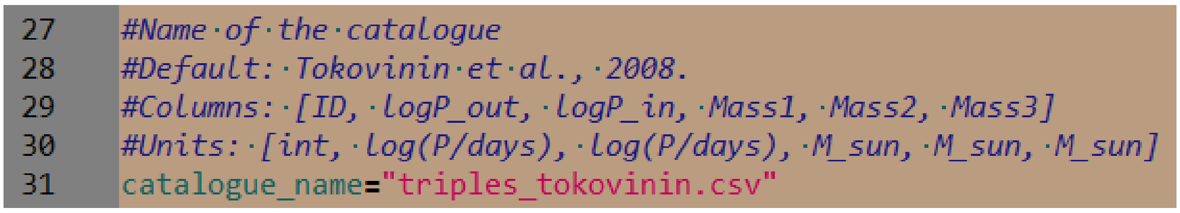
Now, taking the newly assigned mass of the tertiary component, we perform this calculation for every system and obtain a lower limit for the outer period, so when the outer period is chosen randomly from the PDF, it is compared to the minimum value for the system. If it is too small, we select the period again until we find something suitable. We also set a maximum value for the outer period to $\log(P/\text{days})$=10. This is again based on the catalogue and the assumption that the system is effectively isolated beyond that.

We repeat this overall process for every binary system in `BiSEPS`. As a result, we have a list of `BiSEPS` binaries with a realistic tertiary mass and outer period. This alone is not enough for our study as we do not have any other information like temperatures, luminosities etc. However, it works well as a draft for the future library of triples. We can treat these tertiary parameters as “expected” triple parameters.

The advantage of this approach is its relative flexibility. One could easily replace one catalogue of triple stars with another one, and that’s exactly what I am about to describe.

4.1 Changing the catalogue.

The only code we need to edit is `pdf.py`. To make it more accessible, the name of the catalogue is stored in the string, which is right at the beginning of the file at line 31 (Figure 3).



```
27  #Name of the catalogue
28  #Default: Tokovinin et al., 2008.
29  #Columns: [ID, logP_out, logP_in, Mass1, Mass2, Mass3]
30  #Units: [int, log(P/days), log(P/days), M_sun, M_sun, M_sun]
31  catalogue_name="triples_tokovinin.csv"
```

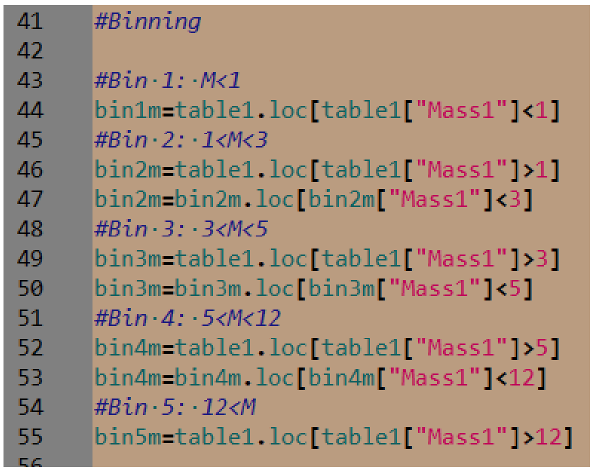
Figure 3: Line of the pdf.py one needs to edit to change the input catalogue. Make sure that the file is .csv and that the columns are in a proper format.

It is important, however, to ensure the following:

- Input file is comma-separated.
- Columns are defined in the first row
- Columns are as follows: [ID, $P_{\rm out}$, $P_{\rm in}$, Mass_1, Mass_2 Mass_3

- Periods are in $\log(P/\text{days})$
- P_{in} is a period of inner binary and P_{out} is a period of outer tertiary
- Masses are in solar masses $M_{\odot}$
- $Mass_1$ and $Mass_2$ are masses of inner binary components and $Mass_3$ is a mass of the outer tertiary component

The final thing one must remember is that splitting the catalogue into five primary mass groups is done automatically, based on primary mass. It is defined between lines 41 and 55 of the `pdf.py`, as shown in Figure 4.



```
41  #Binning
42  
43  #Bin·1:·M<1
44  bin1m=table1.loc[table1["Mass1"]<1]
45  #Bin·2:·1<M<3
46  bin2m=table1.loc[table1["Mass1"]>1]
47  bin2m=bin2m.loc[bin2m["Mass1"]<3]
48  #Bin·3:·3<M<5
49  bin3m=table1.loc[table1["Mass1"]>3]
50  bin3m=bin3m.loc[bin3m["Mass1"]<5]
51  #Bin·4:·5<M<12
52  bin4m=table1.loc[table1["Mass1"]>5]
53  bin4m=bin4m.loc[bin4m["Mass1"]<12]
54  #Bin·5:·12<M
55  bin5m=table1.loc[table1["Mass1"]>12]
```

Figure 4: Lines of the pdf.py one needs to edit to change the binning of the input catalogue. Make sure that all mass ranges are properly represented.

I know that this is not the most efficient way of coding it, but the whole code is very fast, and this way of defining bins is most easily interpretable for a beginner Python user and allows you to easily see how to set it up with more or fewer bins and, if desired, how to introduce binning for periods as well. **The most important thing is ensuring the whole mass range is properly represented.**

If you add or remove bins, ensure that the rest of the code is consistent. To do it, make sure that bins are added or removed from the list on the **line 125**, which feeds them into the main part of the algorithm. The rest of the code will do everything for you. Now, as you run the code, it will produce a list of fully defined triple systems as shown in Figure 1, and save the output in two files: `all_triples.csv` and `all_triples_mag.csv`.

Output is automatically suitable for input into the cross-matching code. Remember that if you rerun the `pdf.py` code, you must also rerun `crossmatching.f90`. As I said before, to run it, type: `./crossmatching.exe`.

5 Cross-matching code

Generally, you do not need to edit this code unless you want to experiment with different assumptions forming the basis of the triple population. For completeness, the section below contains a minimalistic explanation of what the code does. Alongside the comments in the code itself, it provides enough guidance to allow you to understand any part of the code. Nonetheless, you may also consult Section 4.1.3 of my thesis, which contains further details and a discussion of assumptions.

In the previous step, we took a set of binaries, and by assigning expected tertiary parameters to them, we made a skeleton of the library of triple stars. Here we will take that skeleton and a library of single stars and combine them into meaningful triple systems by assigning the physical properties of `BiSEPS` singles to the tertiary component. This will allow us to create meaningful triple populations and perform synthetic observational analysis.

The detailed diagram of this process can be seen in Figure 5, but we will go into some detail here.

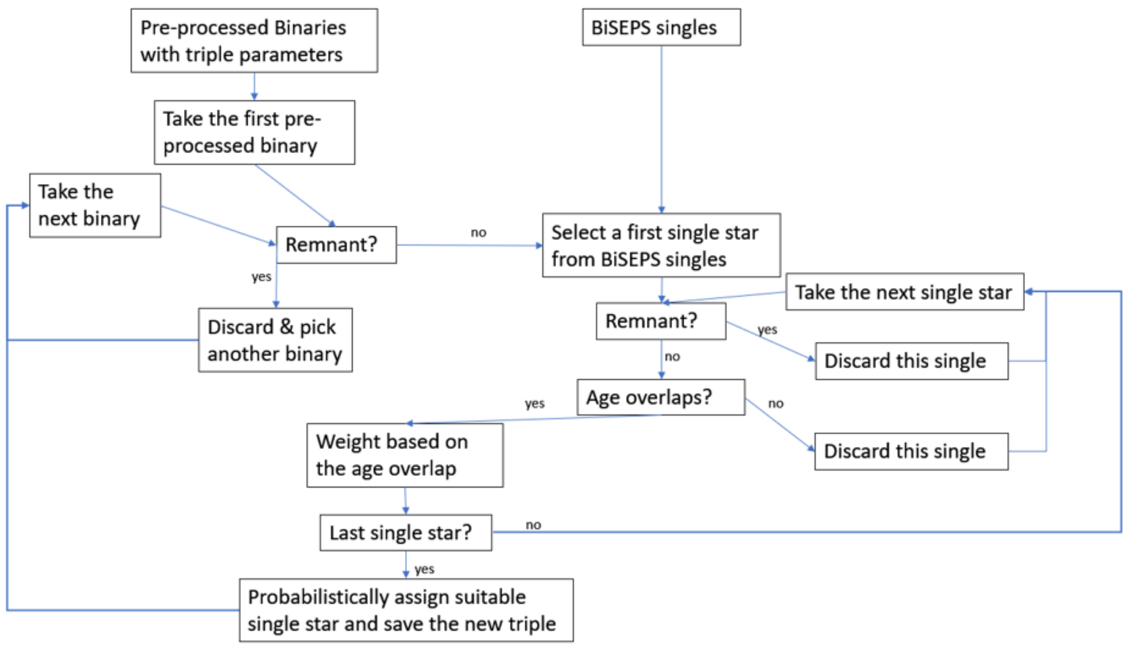

Pre-processed Binaries with triple parameters
BiSEPS singles
Take the first pre-processed binary
Take the next binary
Remnant?
no
Select a first single star from BiSEPS singles
yes
Take the next single star
Discard & pick another binary
Remnant?
yes
Discard this single
no
Age overlaps?
yes
no
Weight based on the age overlap
Discard this single
Last single star?
no
yes
Probabilistically assign suitable single star and save the new triple

Figure 5: The diagram of the cross-matching algorithm. We start with the output of the PDF stage and a library of single stars and, by weighted picking and eliminations, arrive at a library of fully defined triples.

I reject all systems where either star in the inner binary is a remnant, take the expected mass of the tertiary component we obtained during the PDF stage and search the catalogue of single stars for non-remnant stars of the same metallicity within a certain bracket from the expected mass. The choice of the bracket was empirically set at $\pm 10\%$ to ensure that neither the number of failed systems is high nor that the resultant mass is far from the expected mass.

During the next stage, we perform age checks of potential triple components and choose from ageappropriate partners. For this, I assume that all components of the multiple form simultaneously. In `BiSEPS`, we do not have continuous, precise age of stars at any point, but we rather have time elapsed from the formation of the star ($t = 0$) to the beginning and end of the present evolutionary stage. Since the length of these time steps is determined procedurally during each step of stellar evolution, they are not uniform. Therefore, we have to match stars via age overlaps.

The whole process is not very intuitive, but looking at visualisation in Figure 6 should make it clearer. A vertical dashed red line represents the current time. At the bottom of the figure, we see the binary star with its currently observed evolutionary stage highlighted in red. Single stars are shown above the binary. For clarity, I show the full evolutionary cycle of stars, the beginning and end of which are signified by a vertical line, although the code at this point only has access to the one which is happening at the current time.

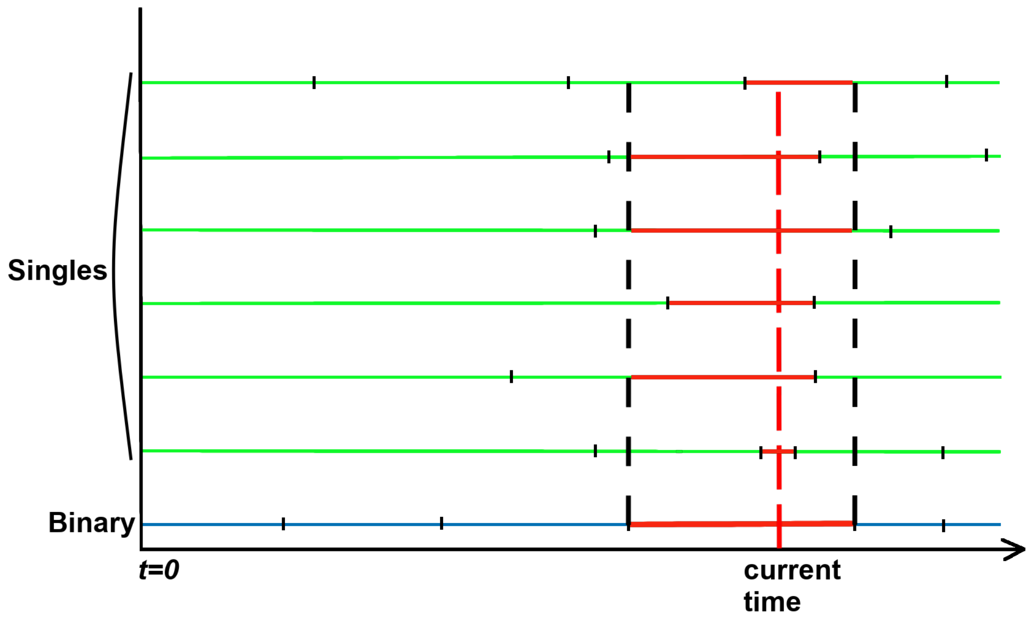


Figure 6: Visualisation of the age-matching and relevant overlaps. We probabilistically match each binary to a suitable single star using their age overlap as a weight. The beginnings and ends of each evolutionary stage are marked with vertical lines.

Since we do not know exactly when we are observing the binary during the evolutionary stage, we take the current stage and look at how it overlaps with the evolutionary sequences of single stars.

The beginning and the end of the current binary evolutionary sequence serve as a lower and upper boundary for sequences of the single stars, so even if the single star sequence begins or finishes earlier or later than that one of the binary, we cut it short using the binary boundary. If the current evolutionary sequence of the single star is fully before or after the current binary sequence, this star is discarded as it can not exist simultaneously with the binary under our assumption that they formed together.

This lets us see how much the current evolutionary sequence of the binary overlaps with the current sequence of the single star. These age overlaps are used as weights – larger overlap increases the likelihood of observing the objects together; hence these are more likely to be paired in a triple. Once the suitable single star is picked, it is permanently attached to the binary, resulting in a hierarchical triple.

To monitor the rate of cross-matching failures, failed triples are also saved into a separate list. The percentage of failures is probabilistic and differs from one run to the other, but the number of failed triples is generally about 10% of the number of successfully created triples, which leaves us with a substantial selection of triples.

This resultant library is a comprehensive list of stellar parameters not only for inner binaries but also for outer tertiary components, and the whole distribution is based on the observed characteristics from the input catalogue. The library is automatically suitable for `BiSEPS`, which will take all necessary libraries automatically.

References

Duchêne, Gaspard and Adam Kraus (2013). "Stellar Multiplicity". *ARAA* 51.1, pp. 269–310. doi: 10.1146/annurev-astro-081710-102602. arXiv: 1303.3028 [astro-ph.SR].

Farmer, R., U. Kolb, and A. J. Norton (2013). "The true stellar parameters of the Kepler target list". *Monthly Notices of the Royal Astronomical Society* 433.2, pp. 1133–1145. issn: 0035-8711. doi: 10.1093/mnras/stt795. eprint: https://academic.oup.com/mnras/article-pdf/433/2/1133/4905798/stt795.pdf. url: https://doi.org/10.1093/mnras/stt795.

Hurley, Jarrod R., Christopher A. Tout, and Onno R. Pols (2002). "Evolution of binary stars and the effect of tides on binary populations". *MNRAS* 329.4, pp. 897–928. doi: 10.1046/j.1365-8711.2002.05038.x. arXiv: astro-ph/0201220 [astro-ph].

Mardling, R. and S. Aarseth (1999). "Dynamics and Stability of Three-Body Systems". *NATO Advanced Science Institutes (ASI) Series C*. Ed. by B. A. Steves and A. E. Roy. Vol. 522. NATO Advanced Science Institutes (ASI) Series C, p. 385.

Raghavan, Deepak et al. (2010). "A Survey of Stellar Families: Multiplicity of Solar-type Stars". *ApJS* 190.1, pp. 1–42. doi: 10.1088/0067-0049/190/1/1. arXiv: 1007.0414 [astro-ph.SR].

Rowden, Pamela M. (2019). "False Positives and Shallow Eclipsing Binaries in Transiting Exoplanet Surveys". url: http://oro.open.ac.uk/60434/.

Sterzik, M. F. and Andrei Tokovinin (2002). "Relative orientation of orbits in triple stars". *AAP* 384, pp. 1030–1037. doi: 10.1051/0004-6361:20020105.

Tokovinin, Andrei (1997). "MSC - a catalogue of physical multiple stars". *AAPS* 124, pp. 75–84. doi: 10.1051/aas:1997181.

— (2008). "Comparative statistics and origin of triple and quadruple stars". *MNRAS* 389.2, pp. 925–938. doi: 10.1111/j.1365-2966.2008.13613.x. arXiv: 0806.3263 [astro-ph].

Toonen, Silvia, Adrian Hamers, and Simon Portegies Zwart (2016). "The evolution of hierarchical triple star-systems". *Computational Astrophysics and Cosmology* 3.1, 6, p. 6. doi: 10.1186/s40668-016-0019-0. arXiv: 1612.06172 [astro-ph.SR].

Willems, B. and U. Kolb (2002). "Population synthesis of wide binary millisecond pulsars". *Monthly Notices of the Royal Astronomical Society* 337.3, pp. 1004–1016. issn: 0035-8711. doi: 10.1046/j.1365-8711.2002.05985.x. eprint: http://oup.prod.sis.lan/mnras/article-pdf/337/3/1004/2810593/337-3-1004.pdf. url: https://doi.org/10.1046/j.1365-8711.2002.05985.x.