SpinUp Documentation

Nicole Mulyk

August 2022

1 Introduction

SpinUp is based on the NSSS code[1]. The main purpose of this code is to compute a neutron star in several stages as mass, and angular momentum are accreted onto it by a donor. The code has options to run forward (with mass accreting onto the NS) or backward (the reverse process). But, by default, the code runs backward to avoid complications with the spin frequency exceeding the Kepler frequency. This default can be changed by setting **direction** in line 283 of **spinup.c** to zero (backward) or one (forwards). The explanation in Section 3 explains the details of the backward option.

2 Preparation

2.1 Compilation

Before compiling the code, check that the definition of the compiler in the "Makefile" is correct for your computer. In this same file, we can modify the size of the grid, which is, by default, MDIV $\times$ SDIV = 151 $\times$ 301.

To compile, at the command line, type:

```
> make spinup
```

Note that after modifying the grid size in the "makefile," you need to delete the object files (with '.o' extension).

Another important thing to have in mind before running SpinUp is that the EOS files have to be in the same directory as the rest of the files for SpinUp. Otherwise, there will be a "Segmentation fault" error message.

2.2 Command Line Flags

The parameters that the code accepts to compute either of the previous tasks are:

- **f** name of the file with the equation of state (EOS)
- **e** initial central energy density (in g/cm^3)

[1]https://github.com/rns-alberta/NSSS

- **n** number of sequences produced
- **m** central energy density for the maximum mass neutron star for a given equation of state
- **s** spin frequency (in Hz)
- **r** initial ratio of r_p/r_e (between 0 and 1)
- **t** maximum spin frequency

An example of the command line would be

```
./spinup -f eos/eosL -e 0.7e15 -r 1.0 -n 20
```

This command line uses the equation of state L located in the folder "eos" the NS will have a central energy density of $\epsilon_c = 0.7 \times 10^{15}/g/cm^3$, and it will have a ratio of the polar and the equatorial radii of $r_p/r_e = 1.0$, which tells us that the neutron star is spherical and non-rotating.

A few things should be considered when choosing command line options. First, the chosen EOS should allow for a maximum mass larger than the star's initial target mass (see Section 3.1). The initial central energy density and ratio of r_p/r_e need to be chosen such that the initial star has a lower mass than the star's initial target mass described in Section 3.1.

3 Main Structure of the Code

This code has four main parts. First, the initial target NS mass and spin are found and interpolated. Next, the equilibrium radius is found. Thirdly, the step size for NS rest mass and angular momentum are calculated, determining the goal rest mass and angular momentum for the next stage. Finally, the goal rest mass and angular momentum for that stage are found and interpolated. The third and fourth steps may be repeated, modeling multiple stages of accretion until the interpolated NS has a ratio of $r_p/r_e > 0.998$.

3.1 Finding the Initial Mass and Spin

The initial target NS mass (**M_star**) and spin (**Omega_star**) are declared in line 171 of **spinup.c**. Following this, SpinUp computes several NS with the same central energy density, searching for **M_star**. Once the NS mass is larger than the goal, the loop stops, and the exact point associated with the target is interpolated. Next, several NS with the same mass are computed to find **Omega_star**. Once the goal is found, the exact value of spin is interpolated. This results in a starting point with the chosen NS mass and spin.

3.2 Calculation of the Equilibrium Radius

In the next step, the equilibrium radius is calculated. First, it calculates the orbital frequency at several radii and then searches for the radius which has an orbital frequency closest to the initial spin frequency of the NS star.

3.3 Calculation of the Step Sizes of NS Rest Mass and Angular Momentum

There are two choices for the accretion radius. In the function **find_l**, located in **spinup_func.c**, indicate **r_type** = 0 on line 425 of to use the innermost stable circular radius as the accretion radius. Alternatively, indicate **r_type** = 1 to use the equilibrium radius (found in Section 3.2) as the accretion radius. The accretion radius is used to calculate angular momentum l, which is needed to find the step sizes of NS rest mass and angular momentum.

The step size for NS rest mass is Δm, and the step size in angular momentum is $l \cdot \Delta m$. Therefore, Δm and l are used to find the next target NS with rest mass **M0_star** and angular momentum **J_star**. The angular momentum l is calculated using Equation 7 from Cook, Shapiro, and Teukolsky [1]. **delta_m** (Δm) should be carefully chosen by the user in line 255 of **spinup.c**. A choice of **delta_m** that is too large may result in the code overshooting a non-rotating NS and creating unphysical stars with $r_p/r_e > 1.0$. If **delta_m** is too small, then SpinUp is not accurate enough to notice a change in NS rest mass and may take too long to run. One difficulty with choosing **delta_m** is that it affects both **M0_star** and **J_star**. Since a small amount of accreted mass can carry a lot of angular momentum, small changes in **M0_star** can make drastic changes to **J_star**. See Section 4 for more details. Although it varies depending on the circumstances, a choice of **delta_m** = 10^{-4} is usually a good choice.

There is an option to decrease **delta_m** when **J_star** decreases below a given threshold (see lines 265-271 of **spinup.c**). This avoids problems with overshooting a non-rotating NS. See Section 4.1 for more information.

3.4 Finding the Target Rest Mass and Angular Momentum

This step finds a NS with **M0_star** and **J_star**. Several NS with the same central energy density are computed until **M0_star** is reached. Then, the loop stops, and the exact point associated with **M0_star** is interpolated. Next, several NS with the same rest mass are computed to find **J_star**. Once the goal is found, the exact value with **J_star** is interpolated. Resulting in a star with **M0_star** and **J_star**. If $r_p/r_e < 0.998$, then the steps in Sections 3.3 and 3.4 repeat.

This section loops through multiple NS at two different stages, firstly to find **M0_star** (function **M0_loop** in **spinup_func.c**) and secondly to find **J_star** (function **J_loop**). During both loops the step size in r_p/r_e is set by **rstep** which is calculated in line 287-292 of **spinup.c**. There is an option in both loops to make **rstep** smaller as r_p/r_e approaches one. This option is in line 489-490 of **M0_loop** and in multiple locations of **J_loop**.

Finally, SpinUp calculates the NS mass and rest mass accreted by subtracting the current NS's mass from the previous mass. Both accreted masses are included since the accreted rest mass does not always increase reliably. It is expected that both masses will increase with each additional step. However, if

delta_m is smaller than the accuracy of the code, then the rest mass may not increase noticeably; see Section 4.4 for more details.

4 Debugging Common Issues

There are a few common problems that can occur, including:

1. Computing an NS with $r_p/r_e > 1$
2. Getting a large percent error
3. The code running for a long time (30 min or longer)
4. Computing a negative mass accreted

Each issue is discussed below, along with possible solutions. Most problems are caused by a poor choice in step size. Usually, adjusting the values of **rstep** or **delta_m** can resolve most issues.

4.1 Computing an NS with $r_p/r_e > 1$

A NS with $r_p/r_e > 1$ is unphysical and indicates an error in the code. In most cases, a NS with $r_p/r_e > 1$ is produced during the loop searching for **M0_star** or **J_star**. This indicates that the step size in rest mass and angular momentum is too large. This can be fixed by choosing a smaller value of **delta_m**. Be careful not to choose a **delta_m** which is smaller than the accuracy of the code; see Section 4.4 for more details.

There is an option described in Section 3.3 that decreases **delta_m** when **J_star** falls below a threshold. This option can be a useful tool in fixing this problem by adjusting the threshold of **J_star** and the new value of **delta_m**.

4.2 Getting a large percent error

The percent error is calculated and printed after every interpolation. A large percent error is typically caused by the loop not passing through the target value or a large gap between the steps in the loop. The former can be fixed by checking the initial condition indicated with command line flags (see Section 2.2) to make sure they created an initial NS with a lower mass than the target. The former can be solved by decreasing **rstep**.

There is an option described in Section 3.4, which decreases **rstep** when r_p/r_e is at a threshold close to one. It may be helpful to adjust the threshold of r_p/r_e or the new value of **rstep** to fix the percent error.

4.3 The code running for a long time (30 min or longer)

If the code is running for more than 30 minutes, this is an indication that the step sizes are too small. Increasing **delta_m** and **rstep** can help solve this problem.

An unreasonably small choice of **rstep** can cause the run time to increase, but a poor choice of **delta_m** can have bigger consequences, as explained in Section 4.4.

4.4 Computing a negative mass accreted

By using inappropriately small **delta_m**, the code is expected to keep track of changes in rest mass that are smaller than the error of the interpolation (on the order of $10^{-5} - 10^{-6}$). This results in the interpolated answer being further from the correct answer than the step size, causing the code to wander aimlessly. This can often result in the SpinUp subtracting mass from the NS rather than adding mass, causing a negative value for accreted mass. This can usually be fixed by choosing a larger **delta_m**.

4.5 How do you know if the code is working properly?

There are a few different checks that can be done to make sure SpinUp is running smoothly. Take a look at the printouts and check if

- The percent error is reasonable (around $10^{-3} - 10^{-4}$)
- The total accreted mass should be roughly equal to the initial mass minus the final mass. Similarly, the total accreted rest mass should also agree with the initial rest mass and final rest mass.
- The NS makes sense physically. Does the mass, spin, radius, and angular momentum make sense? Is $r_p/r_e < 1$?

5 References

[1] G. B. Cook, S. L. Shapiro, and S. A. Teukolsky, "Rapidly rotating neutron stars in general relativity: Realistic equations of state," Astrophys. J., vol. 424, p. 823, Apr. 1994, doi: 10.1086/173934.