NSSPlots
(Neutron Star Sequences Plots)

September 30, 2019

1 How it works

This plotting tool written in python computes different types of plots to analyze the different properties of a rotating neutron star, these are a series of constant rest mass tracks of neutron stars with changing spin frequency. It uses the output data from NSSS[1], which are the parameters seen in Table 1, that are worth noting that these are some of the parameters used in [1].

This is one way of running this code, another option is using the Python environment **"Spyder"**, for which there is a guide to do so in the same repository in Github.

2 Plotting packages

If you don't have Python installed, go to https://www.python.org to download and install it.

Before installing needed packages for NSSPlots, you need to have "pip" installed, which is a package manager for Python. If not, you can install it by typing at the command line

```
> curl https://bootstrap.pypa.io/get-pip.py -o get-pip.py
```

and then,

```
> python get-pip.py
```

The following Python packages are needed when using NSSPlots

- numpy
- matplotlib
- sklearn

To install any of the previous packages, at the command line, type:

```
> pip install numpy
```

If you are using Python 3, just change "pip" for "pip3". If this line causes errors you can instead type:

```
> sudo pip install numpy
```

You will need the package "mpl_toolkits", which is an update in the "matplotlib" package. If you have problems when running NSSPlots use the following:

```
> pip install --upgrade matplotlib
```

If this does not work as intended, you could read the guide to run NSSPlots with **"Spyder"**, which comes when downloading "Anaconda", that has the necessary packages included.

[1] It can be obtained from https://github.com/rns-alberta/NSSS

3 Plotting

There are four codes, NSS_Plots, NSS_oneEOS, NSS_BestFitSurface and NSS_Residuals. The first one generates various plots combining the data obtained from ten different equations of state[2] (EOS); the second one generates the same plots as the previous one, but using just one EOS, instead of the ten; the third one creates the best fit surface, and its equating for the sequences using one EOS, which are produced by NSSS; lastly, the fourth code computes the residuals between each set of sequences and the best fit surface of all the ten EOS sequences.

To run either from the Terminal, the following command line is used to run the first one, for example,

```
> python NSS_Plots.py
```

and by typing a number from the list shown in the terminal, and "enter" we get the chosen plot.

To change the data file from NSSS in each of the codes, changing the file name in the variable "filename" or in the function "np.loadtxt()" is necessary. Note that these data files have to be in the same directory as the codes.

Table 1: Physical properties obtained from NSSS for every single NS computed.

Parameter	Description
ε_c	Central energy density
M	Total mass (in $M_\odot$)
M_0	Rest mass, also known as baryonic mass (in $M_\odot$)
M_*	Mass of the first nonrotating NS in a sequence (in $M_\odot$)
M_M	Maximum mass of a nonrotating NS for a given EOS (in $M_\odot$)
R	Equatorial radius of the NS (in km)
r_p/r_e	Ratio of the polar radius and the equatorial radius
R_*	Equatorial radius of the first nonrotating NS in a sequence (in km)
ν	Rotational frequency (in Hz)
ν_K	Kepler limit for rotation (in Hz)
J	Angular momentum (in cm^{2}g/s)
T	Rotational kinetic energy (in g)
U	Gravitational binding energy (in g)
R_M	Radius of the maximum-mass NS for a given EOS (in km)
M_M/R_M	Compactness of the nonrotating maximum-mass NS for a given EOS

When plotting the different variables, the normalization shows the behaviour of the data can be seen better. The angular velocity can be normalized in the following manner

$$\Omega\left(\frac{R^3}{GM}\right)^{1/2},$$

where G is the gravitational constant. The compactness is given by

$$\zeta = \frac{GM}{Rc^2},$$

where c is the speed of light. A way to normalize this quantity is by dividing it by the maximum compactness, which instead of being the total mass, M, divided by the equatorial radius, R, it is M_{max} divided by the corresponding radius, R_M,

$$\frac{\zeta}{\zeta_{max}} = \frac{M/R}{M_{max/R_M}}.$$

[2]They can be obtained from https://github.com/rns-alberta/eos

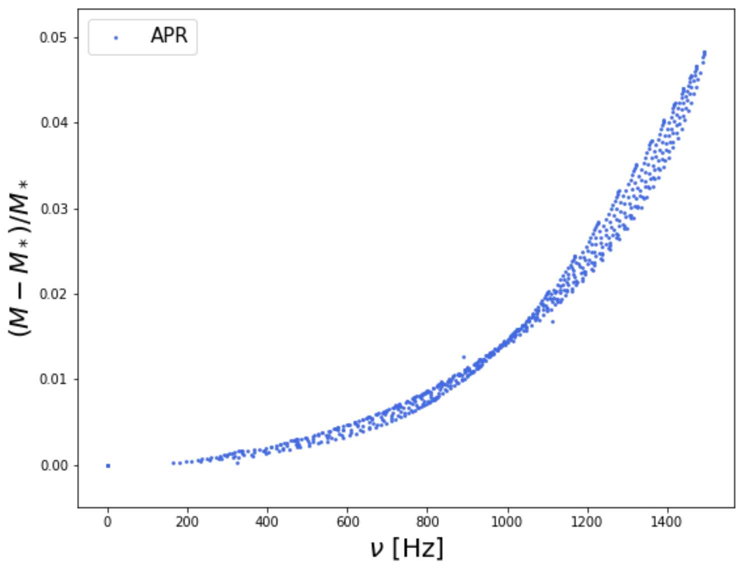

Figure 1: Fraction of mass gain of neutron stars with constant value of M_0 for one EOS, as a function of the normalized frequency

The Kerr spin parameter can be obtained as well,

$$a = \frac{cJ}{GM},$$

and the normalized version of this parameter is

$$\frac{a}{M} = \frac{cJ}{GM^2},$$

The fraction of mass that comes from relativistic sources compared with the baryonic mass is given by

$$\frac{M - M_0}{M_0},$$

and the comparison of the total mass of the neutron star with the mass of the non rotating star with the same properties is

$$\frac{M - M_*}{M_*},$$

and similarly to the last ratio, the comparison of the equatorial radius of the neutron with the radius of the non rotating star with the with the same properties is

$$\frac{R - R_*}{R_*},$$

4 Output

We can see in Figures 1, 2, and 3 some of the types of plots that can be obtained using this code using the normalized quantities previously mentioned. It can output 2D and 3D plots. The figures will be saved automatically to the same directory where the codes are running.

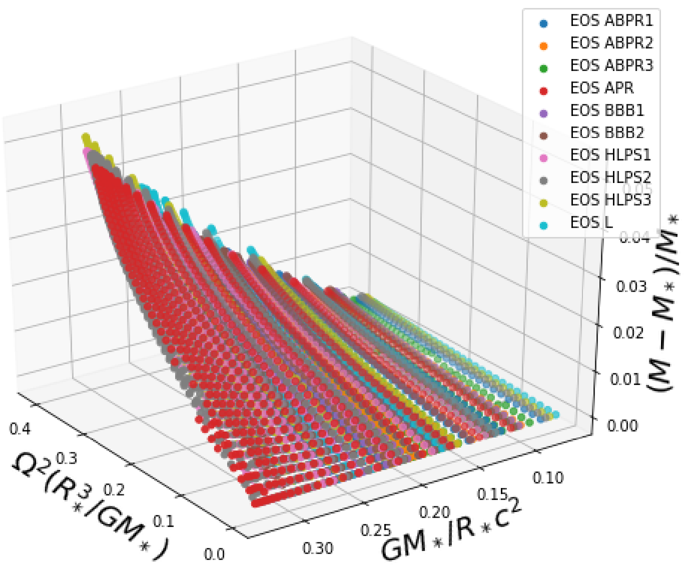

Figure 2: Fraction of total mass gain of neutron stars with constant value of M_0 for ten EOS, as a function of the normalized squared value of angular velocity and the compactness.

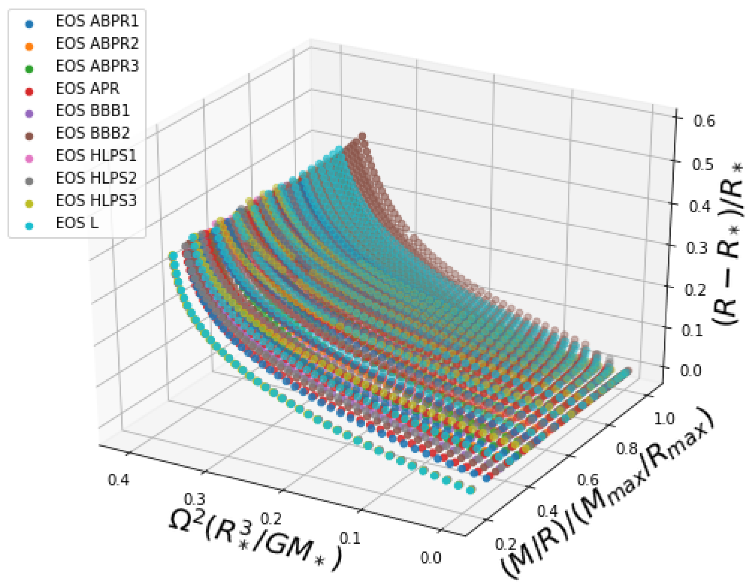

Figure 3: Fraction of total radius gain of neutron stars with constant value of M_0 for ten EOS, as a function of the normalized squared value of angular velocity and the maximum normalized value of M/R

References

[1] Cook, G. B., Shapiro, S. L., & Teukolsky, S. A. 1994, ApJ, 424, 823