Running SeBa

This document explains how to run SeBa for stand-alone use (independent from AMUSE).

a)Simulation of one system:
with the parameters primary mass M=2 solar mass, secondary mass m= 1 solar mass, eccentricy e, orbital separation a=200 solar radii, time T=13500 Myrs, metallicity z=0.001.
./SeBa -M 2 -m 1 -e 0.2 -a 200 -T 13500 -z 0.001
default values e=0, T=13500, z =0.02 (solar)

b)Simulation of multiple system - though specific systems
For example
./SeBa -M 2 -m 1 -e 0.2 -a 200 -T 13500 -z 0.001
./SeBa -M 2.5 -m 1.5 -e 0.5 -a 500 -T 500 -z 0.02
Three options:
1) Run code multiple times :-) Not handy for more then 5 systems..
2) Use an input file: ./SeBa -I 'SeBa_input.txt'
SeBa_input.txt containts:
a e M m z
for example
200 0.2 2 1 0.001
500 0.5 2.5 1.5 0.02
3) Use a shell script. Big advantage: this method is applicable to all command line programs. For example a file named run.sh, should contain the lines for the example given above:
./SeBa -M 2 -m 1 -e 0.2 -a 200 -T 13500
./SeBa -M 2.5 -m 1.5 -e 0.5 -a 500 -T 500
Check permissions of run.sh file; it should be executable by the owner. If not: type ' chmod 744 run.sh ' in command line. To run the shell script: ./run.sh

c)Simulation of multiple system - random population
Monte Carlo based approach
./SeBa -R -n 200
./SeBa -R -n 250000 -m 0.96 -M 11 -q 1e-4 -Q 1 -A 1e6 -f 4 -T 13500

-R SeBa generates randomly the initial parameters
-n number of systems simulated
-m -M min/max primary mass
-q -Q min/max mass ratio
-e -E min/max eccentricy
-a -A min/max orbital separation

-T time in Myr in the simulation of the binaries. Same time for all binaries
-z metalicity of binary stars. Same metalicity for all binaries.

initial parameters are drawn from probably distributions

```
////          -x    mass function exponent in case of power law [-2.35]
////          -F/f  mass function option: 0) Equal mass
////                                      1) Power-law [default]
////                                      2) Miller & Scalo
////                                      3) Scalo
////                                      4) Kroupa
////          Option -F requires one of the following strings:
////                    (mf_Power_Law, Miller_Scalo, Scalo, Kroupa)
////                -f requires the appropriate interger (see mkmass.C)

////          -y   exponent for a power-law distribution [0] (flat in log)
////          -G/g Semi major axis option: 0) Equal_sma
////                                       1) Power Law [default]
////                                       2) Duquennoy & Mayor (1987)
////          Option -G requires one of the following strings:
////                    (Equal_sma, sma_Power_Law, Duquennoy_Mayor)
////                -g requires appropriate interger (see double_star.h)

////          -v   exponent for a power-law distribution
////          -U/u eccentricity option: 0) Equal eccentricity
////                                    1) Power Law
////                                    2) Thermal distribution [default]
////          Option -U requires one of the following strings:
////                    (Equal_ecc, ecc_Power_Law, Thermal_Distribution)
////                -u requires appropriate interger (see double_star.h)

////          -w   exponent for a power-law distribution
////          -P/p mass ratio option: 0) constant mass ratio
////                                  1) Flat_q
////                                  2) Power Law
////                                  3) Hogeveen (1992)
////          Option -P requires one of the following strings:
////                    (Equal_q, Flat_q, qf_Power_Law, Hogeveen)
////                -p requires appropriate interger (see double_star.h)
////
```