Elias Rubin

Received _____; accepted _____

1. Scientific Motivation

Observations of high redshift ($z \geq 6$) quasars hosted by supermassive black holes (SMBHs) of masses in excess of a billion solar masses present a major theoretical challenge. To illustrate, as an example, a seed BH of mass $10^4\ M_\odot$ would need a duty cycle of 100% accreting at 113% Eddington rate from redshfit $z = 15$ in order to reach the observed mass of $2 \times 10^9\ M_\odot$ of quasar ULAS J1120+0641 at $z = 7.1$ (Mortlock et al. 2011). Similarly demanding accretion arrangements are required for some other cases, including the observed quasar SDSS J0100+2802 of SMBH mass of $1.2 \times 10^{10}\ M_\odot$ at $z = 6.3$ (Wu et al. 2015) and quasar ULAS J1342+0928 of SMBH mass of $0.8 \times 10^9\ M_\odot$ at $z = 7.1$ (Bañados et al. 2017).

Clearly, black holes of masses $\sim 10 - 100\ M_\odot$ formed by normal massive stars, when they end their lives, provide inadequate seeds for these observed SMBHs at the epoch of reionization (Bromm & Loeb 2003; Hosokawa et al. 2013). To explain SMBHs with masses over $10^9\ M_\odot$ forming under 1 Gyr, BH seeds of masses around $> 10^4\ M_\odot$ formed at redshift $z \geq 10$ are needed.

In this work we examine the scenario of formation of very massive stars ($10^5 - 10^8\ M_\odot$) through the process of runaway collisions in a subset of dense and massive globular clusters (GCs) formed at very high redshift. Sufficiently dense clusters are highly collisional, and provide for the possibility of runaway growth. As a single star continues to increase in mass and stellar radius, it increases the likelihood of further collisions, and provides an avenue for runaway star growth (Katz et al. 2015). This mechanism for very massive black hole formation is one of two that are commonly considered, the other being the direct collapse of gaseous clouds, as described in Bromm & Loeb (2003). Although direct collapse black holes have been investigated as a promising path to SMBH formation, Latif & Ferrara (2016) and others raise questions about the viability of that pathway because of significant radiation feedback effects which can reduce the rate of mass accretion below the 0.1 $M_\odot$/yr required

to develop sufficiently large seed mass black holes.

We perform calculations of stellar dynamics and stellar collisions in globular clusters with direct N-Body simulations and extend the scenario of runaway stellar collisions to the formation of very massive mass stars, beyond the intermediate mass ($\sim 10^3\ M_\odot$) BHs as previously demonstrated by Portegies Zwart et al. (2004).

An important requirement for a possible runaway formation of very massive stars is to have significantly larger collision cross sections of stars. This requirement then translates to a stringent time constraint. For a normal stellar initial mass function (IMF) it can be shown that the collision cross sections are dominated by the most massive stars. It is thus our expectation, which is verified by our simulations later, that in order to achieve a runaway collision sufficient dense and massive globular clusters are required such that the runaway collisions occur in the first few million years before massive stars end their lives and lose most of their collision cross sections. For a typical initial mass function with a maximum mass of $\sim 100\ M_\odot$, this time is about 3 Myr, afterwards stellar evolution effects such as supernovae and stellar winds can be expected to prevent runaway (Portegies Zwart & McMillan 2002).

Very massive ($\geq 10^4\ M_\odot$) stars, once formed, will ultimately shortly collapse, due to general relativistic instability (Chandrasekhar 1964), to become SMBHs. These SMBHs will anchor the centers of massive galaxies and grow primarily by accreting gas from their surroundings at rates probably fairly close to the Eddington rate to become the observed billion solar mass SMBHs powering the quasars at $z > 6$. Occasionally, mergers of galaxies may also help somewhat grow the SMBHs. **Should observational constraints or other literature be discussed in this section? What about more detail about alternative supermassive black hole (SMBH) pathways like in Latif?**

Cite for relationship between quasar luminosity and black hole mass?

2. Parameter Space of Globular Clusters

Globular clusters are some of the densest stellar systems in the universe. The GCs in the Milky Way come in two distinct flavors, one which is on average more spherically distributed, has low metallicity and low rotation, the other which on average possesses an opposite set of properties (Zinn 1985). The former sub-population has an age of $\sim 12 - 13$Gyr and is thus believed to have formed at the epoch of reionization ($z > 6$). The stellar dynamics in the context of runaway stellar collisions, right after the formation of this old GC population, is the subject of our study.

The overall picture of a runaway collision proceeds approximately in the following way after the formation of a GC. Initially, the stellar distribution is unsegregated in terms of stellar mass. Subsequently, gravitational interactions between stars, a process called dynamical friction, cause more massive stars to lose kinetic energy to gradually towards the center of the GC, with lighter stars moving outwards in compensation. The ability of allowing massive stars to move to the central region is critical, because massive stars have the largest collision cross sections and the central region is densest. Therefore, two of the most important parameters that likely determine whether a runaway collision occurs or not are the dynamical friction time at the half-mass radius, t_{DF}, and the density profile of the GC.

The density profiles of observed GCs are often approximated as a Plummer (1911) or King (1962) profile. For our purpose we choose the King profile due to its ability to easily parameterize the central region of a GC, where most of the collision action occurs. In their investigation of collision in UCDs MGG-9 and MGG-11, Portegies Zwart et al. (2004) identify two factors about collisional clusters that can lead to supermassive star formation: concentration parameter w and dynamical time $t_{\rm df}$. For a GC with a King profile, we have three key parameters: the concentration parameter c, total mass M and t_{df}, with c and t_{df}

defined as

$$c \equiv \log \frac{R_t}{R_c} \tag{2.1a}$$

$$t_{\rm df} \equiv \frac{\langle m \rangle}{M_*} \frac{0.138N}{\ln 0.11M/100\ M_\odot} \left(\frac{R^3}{GM}\right)^{1/2} \tag{2.1b}$$

where M_* is the mass of massive stars that migrate inward via dynamic friction; we choose $M_* = 100\ M_\odot$ to define t_{df}.

King (1966) shows how the parameter c is related to the potential of the cluster. For a system in virial equilibrium and stars of unit mass, the energy of a star is given by

$$E = \frac{1}{2}v^2 + V(r), \tag{2.2}$$

and the velocity is distributed with a Fokker-Planck distribution such that at the center of the cluster, the velocity is given by

$$f(0, v) = k[\exp(-j^2v^2) - \exp(-j^2v_e^2), \tag{2.3}$$

where $1/j$ is the velocity dispersion and v_e is the cluster escape velocity. At the tidal radius, a particle becomes unbound from the cluster even at $v_e = 0$, hence, $V(r_t) = 0$. As particles with positive energy escape the cluster, we see that the escape velocity is given as a function of r, such that

$$v_e^2 = -2V(r). \tag{2.4}$$

We see that the distribution function can be extended over r and so

$$f(r, v) = k[\exp\{-2j^2(V(r) - V(0))\}][\exp[-j^2v^2] - \exp\{-2j^2(V(r))\}. \tag{2.5}$$

The density ρ is given by the integral of $f(r, v)$ over v, that is

$$\rho = \int_0^{v_e} f(r, v) 4\pi v^2 {\rm d}v, \tag{2.6}$$

and so we see ρ is a function of $V(r)$, related by the Poisson equation,

$$\frac{\mathrm{d}^2V}{\mathrm{d}r^2} + \frac{2}{r}\frac{\mathrm{d}V}{\mathrm{d}r} = 4\pi G\rho. \tag{2.7}$$

Letting $W = -2j^2V$ and $R = r/r_c$, we can substitute the above to

$$\frac{\mathrm{d}^2W}{\mathrm{d}R^2} + \frac{2}{R}\frac{\mathrm{d}W}{\mathrm{d}R} = -8\pi G j^2 r_c^2 \rho, \tag{2.8}$$

where r_c is the core radius, which King (1966) defines as

$$r_c^2 = \frac{9}{8\pi G j^2 \rho_0}, \tag{2.9}$$

from an empirical result. Substituting equation 2.6 into equation 2.8 and making the substitions for R and W sets up sets up an integral equation which can be numerically solved for r_t by setting W (and hence V) $= 0$. Because the concentration is given by $c = \log r_t/r_c$, we can see that for a given core radius, higher concentrations will have a greater tidal radius and therefore a deeper potential well.

We stress that all three parameters are important. In particular, the collision rate per time that determines the outcome depends not only on c and t_{df} but also on M.

We survey the parameter space of c, t_{df} and M, in part guided by observations. But we imagine that some of the densest and most massive globular clusters may have disappeared from our sight to have integrated into larger stellar systems that are no longer recognized as GCs today. Observationally, the GCs and ultra compact dwarf galaxies (UCDs), which are essentially more massive cousins of GCs observed beyond the Local Group, span a mass range of $10^5 - 10^{8.5}$ $M_\odot$. Well known GCs 47 Tuc and ω Cen have $c = 2.03$ and 1.30 (Carraro & Lia 2000), respectively, falling in the overall range of $c \sim 1-3$ (e.g., Evstigneeva et al. 2007). We thus sample the range $1-3$ for the parameter c. The parameter t_{DF} is a function of GC mass M, half-mass radius R and the mass of the star in question $< m >$ (see equation above).

3. Simulation Methods

3.1. The N-Body Problem

We are interested in following the evolution of collisional clusters and exploring the range of parameters in which they can lead to SMBH formation. Finding a solution entails solving the equation of motion for each particle specified by the force $\mathbf{F}_i$ and its time derivative $\mathbf{F}_i^{(1)}$ as follows:

$$\mathbf{F}_i = -m_i \sum_{i \neq j} G m_j \frac{\mathbf{R}}{R^3}; \tag{3.1a}$$

$$\mathbf{F}_i^{(1)} = -m_i \sum_{i \neq j} G m_j \left[\frac{\mathbf{V}}{R^3} + \frac{3\mathbf{R}(\mathbf{V} \cdot \mathbf{R})}{R^5} \right], \tag{3.1b}$$

where G is the gravitational constant, m_j is the mass of particle j, and $\mathbf{R}$ and $\mathbf{V}$ are the vector distance and velocity respectively between any two particles i and j. Because it is not possible to analytically solve the dynamical equations for every body in a cluster, systems must be evolved numerically. Developing numerical methods for N-Body simulations is a long tradition, but the two most popular methods are tree-based simulations and direct simulations. The tradeoff between the two is one of computational cost versus accuracy. Notwithstanding implementation details, direct simulations compute pairwise force terms for every pair of bodies at each discrete simulation timestep, which comes with a computational cost per timestep of $O(N^2)$ in the number of particles. Tree-based simulations partition the simulation space and compute pairwise terms for close neighbors of a given particle, but aggregate forces from groups outside of a close radius. Thus, the cost per timestep goes as $O(N \log N)$.

The metric generally used for simulation error is energy conservation across the system. Practicioners focus their efforts on increasing simulation speed while keeping increases to errors within acceptable bounds. The acceptable level of error depends on the problem

domain and desired fidelity of the result. For our cluster simulations we have many bodies in close proximity and care about the accurate treatment of close encounters, binaries, multiple systems, and collisions, so we use `NBody6++GPU` , a direct code. An alternative we considered is the tree-based code `Starlab`, but our initial testing and recommendations from practicioners suggested that `NBody6++GPU` would be better for our use case.

We perform direct N-body simulations over a range of cluster parameters varying in total mass, half-mass radius, and concentration. We use the `McLuster` software of Küpper et al. (2011) to generate initial conditions and the `NBody6++GPU` software of Wang et al. (2015) to evolve the clusters. We also describe some modifications to the `NBody6++GPU` software made in attempts to accelerate simulations.

3.2. Numerical Methods

`NBody6++GPU` is the latest version in a family of direct N-body simulators beginning with `NBody1` of Aarseth (1963). Aarseth (1999) describes in detail the evolution of the `NBody` family of programs and numerical methods therein. The code is written primarily in Fortran but has modules written in CUDA for GPU extensions and C++ for access to AVX/SSE instructions. It is parallelized with OpenMP.

In order to solve equations 3.1, the `NBody6++GPU` integrator uses a direct fourth-order Hermite integration method, as well as a hierarchical block step (Wang et al. 2015, and refs within). This is a kind of adaptive timestep system. Instead of evaluating all particles at every timestep, particles are binned into quantized groups, and only those particles at an integer multiple of their timestep are integrated. This speeds up the integration of slow moving particles, while still allowing for accuracy for more extreme particles. The code uses the neighbor scheme of Ahmad & Cohen (1973) to speed up the integration further by

separating into the more economical regular force (forces on a body generated by bodies outside a neighbor radius) and irregular force (forces generated by bodies within a neighbor radius). In `NBody6++GPU` , the regular force computations are done on the GPU, and irregular force computations are done on the CPU with the AVX/SSE library of Nitadori & Aarseth (2012). The next part of this section explains each of these in more detail.

The Hermite integration method is used to solve equations 3.1. This is a predictor-corrector method that uses a third-order Taylor expansion. The expression for the force is described in Aarseth (1999) and is as follows:

$$\mathbf{F}_{t+1} = \mathbf{F}_t + \mathbf{F}_t^1 \Delta t + \frac{1}{2}\mathbf{F}_t^{(2)} \Delta t^2 + \frac{1}{6}\mathbf{F}_t^{(3)} \Delta t^3, \tag{3.2a}$$

$$\mathbf{F}_{t+1}^{(1)} = \mathbf{F}_t^{(1)} + \mathbf{F}_t^{(2)} \Delta t + \frac{1}{2}\mathbf{F}_t^{(3)} \Delta t^2. \tag{3.2b}$$

$\mathbf{F}$ and $\mathbf{F}^{(1)}$ can be computed at the beginning and end of a timestep by using equations 3.1. They are used to form the higher order terms

$$\mathbf{F}_t^{(2)} = \frac{2}{\Delta t^2}\left[-3(\mathbf{F}_t - \mathbf{F}_{t+1}) - 2(\mathbf{F}_t^{(1)} + \mathbf{F}_{t+1}^{(1)})\Delta t\right], \tag{3.3a}$$

$$\mathbf{F}_t^{(3)} = \frac{6}{\Delta t^3}\left[2(\mathbf{F}_t - \mathbf{F}_{t+1}) + (\mathbf{F}_t^{(1)} + \mathbf{F}_{t+1}^{(1)})\Delta t\right]. \tag{3.3b}$$

Then the predictor and corrector $\mathbf{r}_{p,t+1}, \mathbf{v}_{p,t+1}, \Delta\mathbf{r}$, and $\Delta\mathbf{v}$ are given thusly

$$\mathbf{r}_{p,t+1} = \mathbf{r}_t + \mathbf{v}_t \Delta t + \frac{1}{2}\mathbf{F}_t \Delta t^2 + \frac{1}{6}\mathbf{F}_t^{(1)} \Delta t^3, \tag{3.4a}$$

$$\mathbf{v}_{p,t+1} = \mathbf{v}_t + \mathbf{F} \Delta t + \frac{1}{2}\mathbf{F}_t^{(1)} \Delta t^2, \tag{3.4b}$$

$$\Delta\mathbf{r} = \frac{1}{24}\mathbf{F}_t^{(2)} \Delta t^4 + \frac{1}{120}\mathbf{F}_t^{(3)} \Delta t^5, \tag{3.4c}$$

$$\Delta\mathbf{v} = \frac{1}{6}\mathbf{F}_t^{(2)} \Delta t^3 + \frac{1}{24}\mathbf{F}_t^{(3)} \Delta t^4, \tag{3.4d}$$

with the final state at $t + 1$ given by $\mathbf{r}_{t+1} = \mathbf{r}_{p,t+1} + \Delta\mathbf{r}$ and $\mathbf{v}_{t+1} = \mathbf{v}_{p,t+1} + \Delta\mathbf{v}$.

The Hermite predictor is used for each time step in the hierarchical block step scheme. Block steps are the way the `NBODY` family of codes addresses the large dynamic range in the

velocities and proximities of bodies in clusters. This is necessary for performance because otherwise the time step for the entire system would be determined by the separation of the closest bodies. Instead each particle is given its own timestep using the following formula of Khalisi, E. and Wang, L. and Spurzem, R. (2017)

$$\Delta t_i = \sqrt{\eta \frac{|\mathbf{a}_{1,i}||\mathbf{a}^{(2)}_{1,i}| + |\mathbf{a}^{(1)}_{1,i}|^2}{|\mathbf{a}^{(1)}_{1,i}||\mathbf{a}^{(3)}_{i,1}||\mathbf{a}^{(2)}_{1,i}|^2}}. \tag{3.5}$$

We use an initial value of 0.01 for η which we arrived at as it generally allowed our simulations to progress further before energy error becomes too great which forces a restart. This value may be further adjusted if the relative energy error dE (defined as the difference in total simulation energy between checkpoint times) becomes close (within a factor of 5) to the maximum tolerance Q. The correction factor is given by $\sqrt{dE/Q}$. Our simulations use a checkpoint timestep of 0.01 $t_{\rm nb}$, where $t_{\rm nb}$ is the simulation time unit and is equal to $\frac{1}{2\sqrt{2}} t_{\rm df}$ (Khalisi, E. and Wang, L. and Spurzem, R. 2017). We set $Q = 0.05$, which allows for a maximum energy error of 20%, and causes a readjustment of η when the energy error reaches 4%.

Figure 1 shows an example of how the particles are advanced.

The `NBody6++GPU` employs the neighbor scheme of Ahmad & Cohen (1973) in conjunction with the Hermite integrator and block timestep. This is a further optimization to speed up the calculations, this time by identify particles that are spatially close to each other. Thus, a given particle will actually have two timesteps, a regular timestep and (smaller) irregular timestep. Particles have a neighbor list of size at most $N_{\rm nb} \ll N_{\rm tot}$. These are advanced using a similar predictor-corrector as the Hermite scheme above, computing the full force for neighbor particles and the last regular values for non-neighbors. Membership in the neighbor list is determined by the size of the neighbor sphere and the

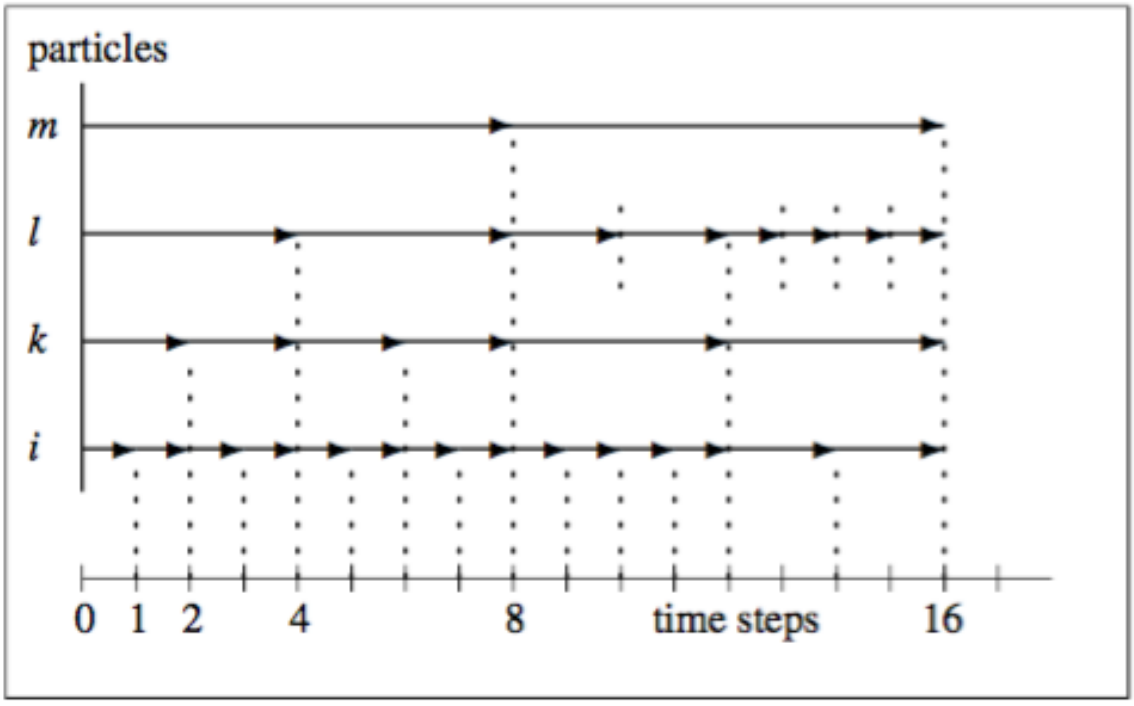


Fig. 1.— Particles i, k, l, and m have separate time steps. A full force computation is done at each arrow, otherwise, the position and velocity are extrapolated from the values already computed. These time steps are reevaluated by equation 3.5 after a full integration cycle. Subsystems which are evaluated independently are replaced by their center of mass in the block scheme. Figure reproduced from Khalisi, E. and Wang, L. and Spurzem, R. (2017).

length of the neighbor list $N_{\rm nb}$. The first $N_{\rm nb}$ particles within a sphere of radius $r_{\rm search}$ are included. We found that too small values for $N_{\rm nb}$ would lead to code crashes and settled on a value of 1024. We use a search sphere of radius 1×10^{-4} pc.

The distinguishing feature of the `NBODY` family is their use of Kustaanheimo & Stiefel (1965) (KS) regularization to generate solutions for binaries, triples, and multiple systems with high levels of accuracy. This treatment for close encounters adds a good deal of complexity to the implementation, but the scheme essentially works by searching for bodies closer to each other than $r_{\rm min}$, replacing them in the next integration step by their center of mass, and integrating them separately at a smaller timescale and in a different reference frame. This is useful to avoid numerical truncation errors caused by bodies being too close in the originally reference frame. The KS regularization is extended to multiple systems and adds perturbing bodies using a variant of the Ahmad & Cohen (1973) neighbor scheme. KS pairs and multiples are terminated if the distance between the bodies exceeds $r_{\rm min}$ or if the bodies collide, in which case they are merged. We discuss the merger criteria and method later in this section.

Although `NBody6++GPU` uses MPI parallelization to scale across multiple compute nodes, there are some sequential bottlenecks including KS regularization. Figure 2 shows the results of our tests scaling to multiple nodes and compares to the theoretical limit. We also performed profiling which indicated that the bulk of the time was spent in KS regularization. Because we didn't achieve gains from horizontal scaling, we were not able to take full advantage of the computing power available to us.

We performed our simulations on the `Tiger2` GPU cluster at Princeton University. Most experiments were performed on one compute node which each use four NVIDIA Tesla P100 GPUs and 2.4 GHz Broadwell processors parallelized with OpenMP.

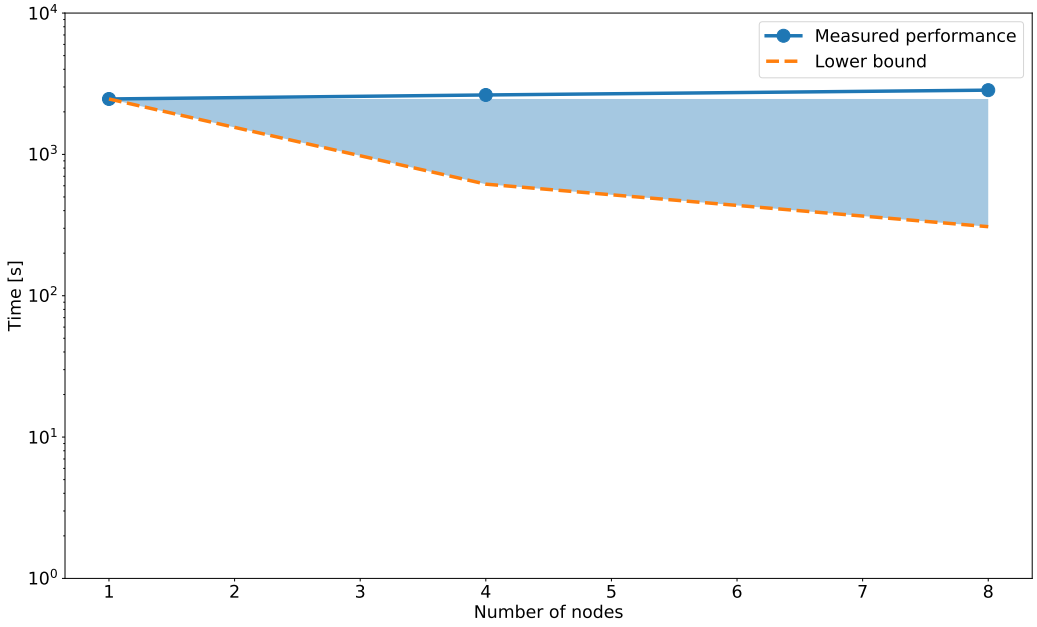


Fig. 2.— This figure shows the results of our parallel test runs and compares to the theoretical limit. The shaded region indicates a range of speedups that we might expect depending on the fraction of parallelizable work following Amdahl's law. That actual performance was even worse is a result of communication overhead. For all of our parallelization test runs we used identical clusters of 4000 bodies with total mass of 13k $M_\odot$ and following a King (1966) profile with concentration 2 and half-mass radius 0.5 pc. Each cluster was evolved to 3 Myr.

3.3. Initial and Boundary Conditions

We generate our initial conditions using the `McLuster` software of Küpper et al. (2011). For the initial distribution of positions and velocities, we use a King (1966) profile, varying half-mass radius and concentration. We use the initial mass function of Kroupa (2001) which degenerates to a Saltpeter IMF above 0.5 $M_\odot$. For our simulations with cluster mass 2×10^5 $M_\odot$, we use a mass range of 1 to 100 $M_\odot$. For cluster mass 6×10^5 $M_\odot$, we increase the lower bound to 3.3 $M_\odot$. For cluster mass 2×10^6 $M_\odot$, we use a lower bound of 13 $M_\odot$ and an upper bound of 120 $M_\odot$. We do not apply an external tidal field, initialize primordial binaries, or apply primordial mass segregation. Figure 3 shows the initial density profiles for each run.

We use an open boundary condition so particles that do escape do not return to the simulation. Our escape condition is a distance from center of greater than twice the tidal radius. Both of these conditions remain uniform over all of our simulations.

3.4. Merger Scheme

The default merger criteria in `NBody6++GPU` follows that of Kochanek (1992). For a two-body system with pericenter R_p, the bodies are merged if

$$\frac{R_p}{R_1} \leq 1.3 \times (\frac{M_T}{2M_1})^{1/3}. \tag{3.6}$$

This merger scheme is conservative. In the case of a central mass with $m = 200$ $M_\odot$ (a typical case after a couple of mergers) and an additional mass with $m = 1$ $M_\odot$, and using $r_{\rm central} = (\frac{m}{M_\odot})^{0.55}$, we would not merge the two stars unless they had a pericenter distance smaller than 20 stellar radii, or a deviation of only about 3% from the radius of the central

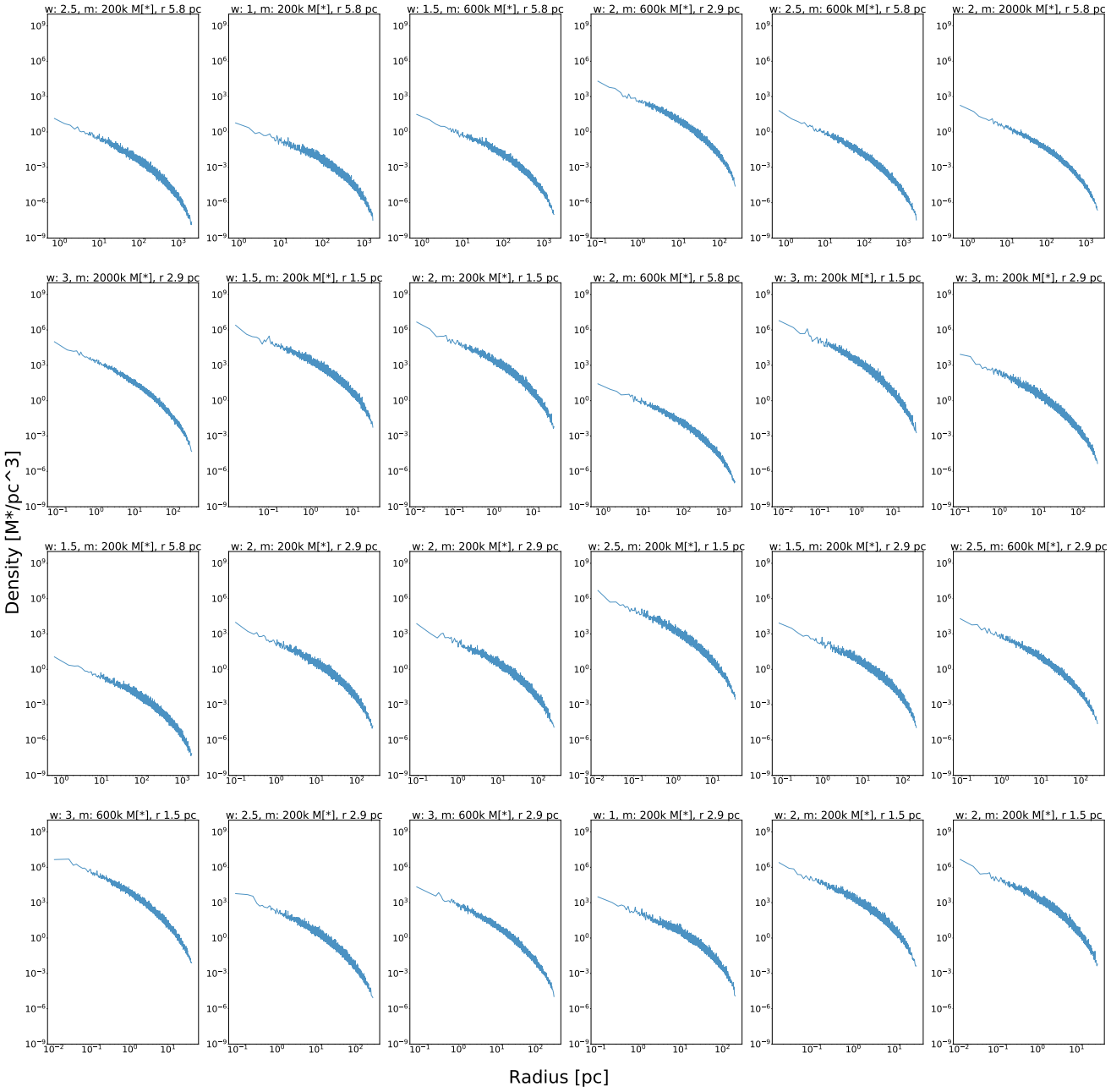


Fig. 3.— Initial density profile for each run. We computed the density profiles by binning stars in groups of 32 and taking the average mass over the spherical volume they occupy, so the noise in the density curves is because without primordial mass segregation more massive

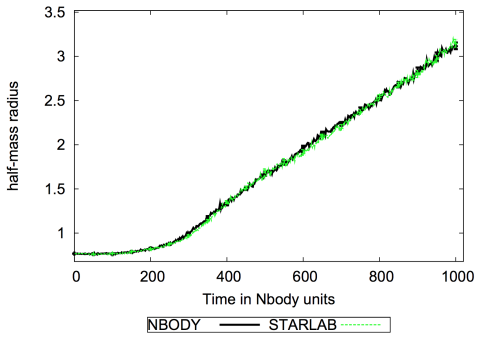
mass.

In an effort to increase the speed of our simulations by spending less time in KS regularization for insignificant bodies, we experimented with short-circuiting this merger scheme for pairs meeting certain conditions. We experimented with merging binaries where one member was 5 or 10 times larger than the other, provided the pericenter distance was smaller than $10r_{\rm large}$ (in the more conservative trial) and $100r_{\rm large}$ in the more liberal one. We avoided merging with the central mass, but did want to reduce the number of KS pairs for peripheral bodies. We found that the overall runtime was not affected significantly by this change to the merger scheme. Simulation results appeared robust to the factor 10 difference, but were affected by the factor 100 difference. We tested changes to our merger scheme at approximately the transition between collisional and non-collisional clusters: a mass of $2 \times 10^5\ M_\odot$, a concentration of 2, and a half-mass radius of 2.9 pc.

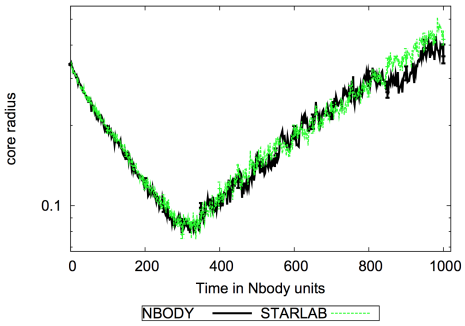
3.5. Verification

Anders et al. (2009, 2012) perform comparisons between `NBody4` of Aarseth (1999) and `Starlab` environment. In order to verify our installation of `NBody6++GPU` we repeated the tests from Anders et al. (2009). We used `McLuster` to generate similar initial conditions to the ones used in their test. Our verification clusters consist of 1024 1 $M_\odot$ stars, distributed in a Plummer profile with initial half-mass radius 0.5 pc. We performed test runs with and without stellar evolution mass loss, as `NBody6++GPU` requires stellar evolution mass loss to perform mergers.

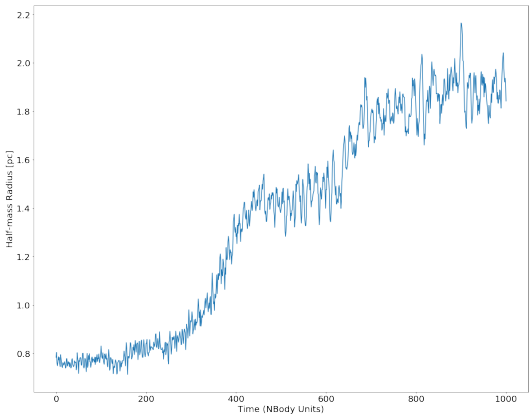
The trends of our verification runs generally agreed with those in Anders et al. (2009). The authors left some uncertainty about the exact parameters used for their runs, and they also sampled many runs to obtain an average. Figures 4 and 5 compare the two.



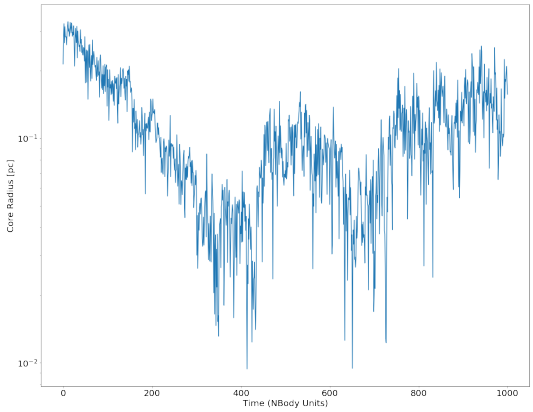
(a) Half-mass radius vs. time reproduced from Anders et al. (2009).



(b) Core radius vs. time reproduced from Anders et al. (2009).

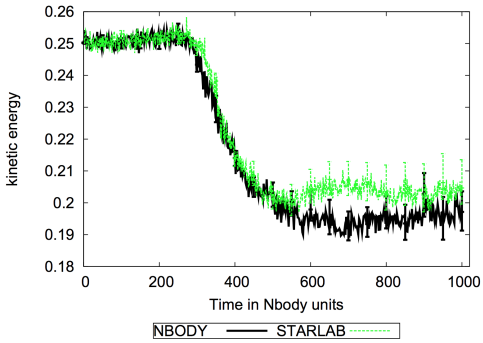


(c) Half-mass radius vs. time from our verification run.

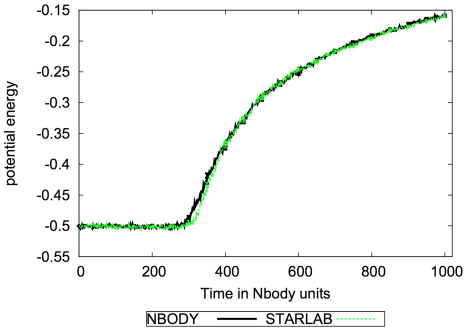


(d) Core radius vs. time from our verification run.

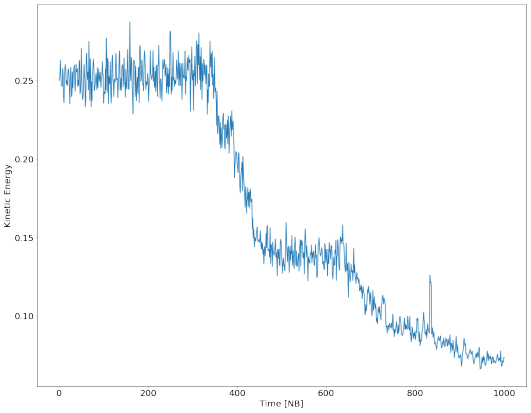
Fig. 4.— Our verification run shows generally good agreement with those of Anders et al. (2009). The half mass radius doesn't start to expand significantly until 200 time steps in both sets, and then follows a general upwards trend. Our core radius shows a similar peak-trough distance of oscillation and a return close to the starting point by the end of the run. Our result appears to oscillate more because we show a single run instead of an entire sample.



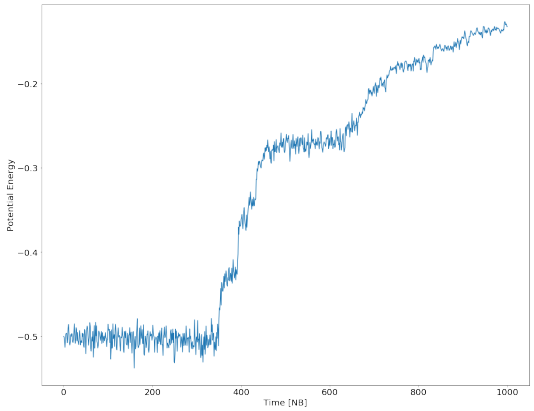
(a) Kinetic energy vs. time reproduced from Anders et al. (2009).



(b) Potential energy vs. time reproduced from Anders et al. (2009).



(c) Kinetic energy vs. time from our verification run.



(d) Potential energy vs. time from our verification run.

Fig. 5.— We see the same general trend with kinetic and potential energy from our verification run, although the kinetic energy from our run seems to end up lower than that of Anders et al. (2009). This is where there is also the greatest divergence between `NBODY4` and `Starlab`, so it's possible that changes in between `NBODY4` and `NBody6++GPU` account for this difference.

3.6. Additional Modifications

We had significant difficulty running more collisional clusters to completion. We found that clusters with many collisions stretched the limits of default `NBody6++GPU` parameters. Among other issues, we encountered a persistent problem where runs would stop progressing during a KS regularization. We settled on a two-part workaround for this problem: first by reducing the KS step size which seemed to forestall these failures at the cost of some additional compute time and second by simply restarting runs once they stopped progressing.

We were unable to get `NBody6++GPU` 's internal restart module to function in our environment, so our restart method required building a tool to parse the checkpoint data and create initial conditions for a fresh simulation. The raw checkpoint data does not perfectly reflect the state of the system because it contains duplicate bodies when binaries, triples, quads, and chains are involved. We implemented an additional heuristic in our restart code to remove spurious binaries and other multiples, leaving only the constituent bodies in the new initial conditions.

`NBody6++GPU` 's default configuration does not always seem to handle collisional clusters well. This led to other failure modes for our simulations which we tried to correct. In dense clusters we found that the neighbor lists would rapidly become full leading to code crashes. Here we tried reducing the size of neighbor search spheres per the suggestion of 201 (????, personal communication), which did not resolve the issue, so we instead doubled the size of the neighbor lists which was enough to alleviate the issue. Another failure mode which we have not yet been able to find a solution to occurs after restarts during GPU force initialization. It appears some initial conditions cause GPU out-of-memory errors which cause `NBody6++GPU` to crash. The resolution we have adopted for that failure mode is to binary search snapshots until finding one that proceeds and begin the run at that point.

We then have to invalidate data after the selected snapshot.

4. Results

4.1. Determining Runaway Candidates

We performed simulations of 24 clusters sampling the parameter space of concentration, cluster mass, and core radius, as detailed in Table 1. We targeted an end time of 3 Myr for each cluster to reach the end of life for the most massive stars in the cluster. Because of the computational difficulty of simulating collisional clusters with many bodies, not all of the runs made it to completion. Figure 6 shows the suite of runs and their end times, as well as the size of their final central masses. As many simulations only make it to $\sim 0.1 - 0.5$ Myr before they become too difficult to simulate, comparing the results of different clusters requires normalizing for time. However, we can still make a guess as to whether a runaway should occur, even if determing the mass of the star at $t_{\rm final}$ is difficult. To do this, we suppose that the early stages of massive star growth can be modeled as an exponential such that

$$M_{\rm max}(t) = M_0 e^{(Kt)}. \tag{4.1}$$

We can solve for K at the end of a given run by isolating it from equation 4.1 as follows:

$$K = \frac{\ln M_{\rm max}(t_{\rm end}) - \ln M_0}{t_{\rm end}} \tag{4.2}$$

We can then use K to make an estimate for the central mass size as a fraction of the cluster at $t = 3$ Myr. If the projected final mass fraction is large, we suggest the cluster is a candidate to have a runaway central mass. Figure 7 shows each run plotted with its projected final mass fraction.

Some of our runs contain central masses that reach $> 1\%$ of the total cluster mass on timescales of only ~ 0.1 Myr. Figure 8 shows the growth of one such case. The central

Run	w	$M_{\rm cluster,0}$	$R_{\rm half}$	$t_{\rm df}$	$t_{\rm end}$	$M_{\rm central,end}$
(units)	-	$M_\odot$	pc	Myr	Myr	$M_\odot$
RUN64K-W2-M200-R1.5-2	2	2.0e+05	1.5	4.37	0.01	268.5
RUN64K-W3-M600-R1.5	3	6.0e+05	1.5	6.29	0.01	829.9
RUN64K-W2-M200-R1.5-3	2	2.0e+05	1.5	4.37	0.04	703.2
RUN64K-W3-M600-R2.9	3	6.0e+05	2.9	16.90	0.05	183.0
RUN64K-W3-M2000-R2.9	3	2.0e+06	2.9	26.03	0.06	1021.0
RUN64K-W2.5-M200-R1.5	2.5	2.0e+05	1.5	4.37	0.11	916.6
RUN64K-W1.5-M200-R1.5	1.5	2.0e+05	1.5	4.37	0.14	2548.8
RUN64K-W2-M200-R1.5	2	2.0e+05	1.5	4.37	0.32	2770.8
RUN64K-W3-M200-R1.5	3	2.0e+05	1.5	4.37	0.38	429.7
RUN64K-W2.5-M600-R2.9	2.5	6.0e+05	2.9	16.90	0.51	163.8
RUN64K-W2-M600-R2.9	2	6.0e+05	2.9	16.90	0.72	770.8
RUN64K-W1.5-M200-R2.9	1.5	2.0e+05	2.9	11.75	0.82	1002.1
RUN64K-W2.5-M200-R2.9	2.5	2.0e+05	2.9	11.75	0.96	151.5
RUN64K-W2-M2000-R5.8	2	2.0e+06	5.8	73.62	1.09	2388.0
RUN64K-W1-M200-R2.9	1	2.0e+05	2.9	11.75	1.20	1082.9
RUN64K-W2-M200-R2.9-3	2	2.0e+05	2.9	11.75	1.26	1398.2
RUN64K-W2-M200-R2.9-2	2	2.0e+05	2.9	11.75	2.96	308.0
RUN64K-W2.5-M200-R5.8	2.5	2.0e+05	5.8	33.22	3.00	121.1
RUN64K-W1-M200-R5.8	1	2.0e+05	5.8	33.22	3.92	844.5
RUN64K-W2.5-M600-R5.8	2.5	6.0e+05	5.8	47.80	4.27	315.8
RUN64K-W1.5-M600-R5.8	1.5	6.0e+05	5.8	47.80	4.35	1320.9
RUN64K-W2-M600-R5.8	2	6.0e+05	5.8	47.80	5.48	3204.3
RUN64K-W3-M200-R2.9	3	2.0e+05	2.9	11.75	5.98	941.0
RUN64K-W1.5-M200-R5.8	1.5	2.0e+05	5.8	33.22	8.68	12.8

Table 1: Table of each full run (64k bodies), where w is the King concentration, $M_{\rm cluster,0}$ is the initial cluster total mass, $R_{\rm half}$ is the initial half-mass radius, $t_{\rm df}$ is the dynamical friction time, $t_{\rm end}$ is the simulation end time, and $M_{\rm central,end}$ is the ending central mass size.

mass reaches over 2700 $M_\odot$ by $t = 0.3$ Myr, or about 1.3% of the cluster mass in 10% of the limiting time. The growth rate starts off starts off superexponential as the stars initially come into contact.

What fraction of the total cluster mass did the runaway achieve? How long didit take?

What was the rate of growth? Did this vary with cluster parameters? What about the shape of the growth curves?

Any clusters that were too expensive to simulate or didn't work for some other reason?

What other observed phenomena? Did any/all clusters experience mass loss? What happened with core radius? Was core collapse correlated with larger central mass?

All clusters had some mass loss, primarily because of escapers. The total number of particles in the system decreases with escapers and mergers, and the decrease in particle number appears to be commensurate with mass loss, suggesting that there's not a big difference between the masses of ejected and retained particles. Some clusters lose large portions of their mass

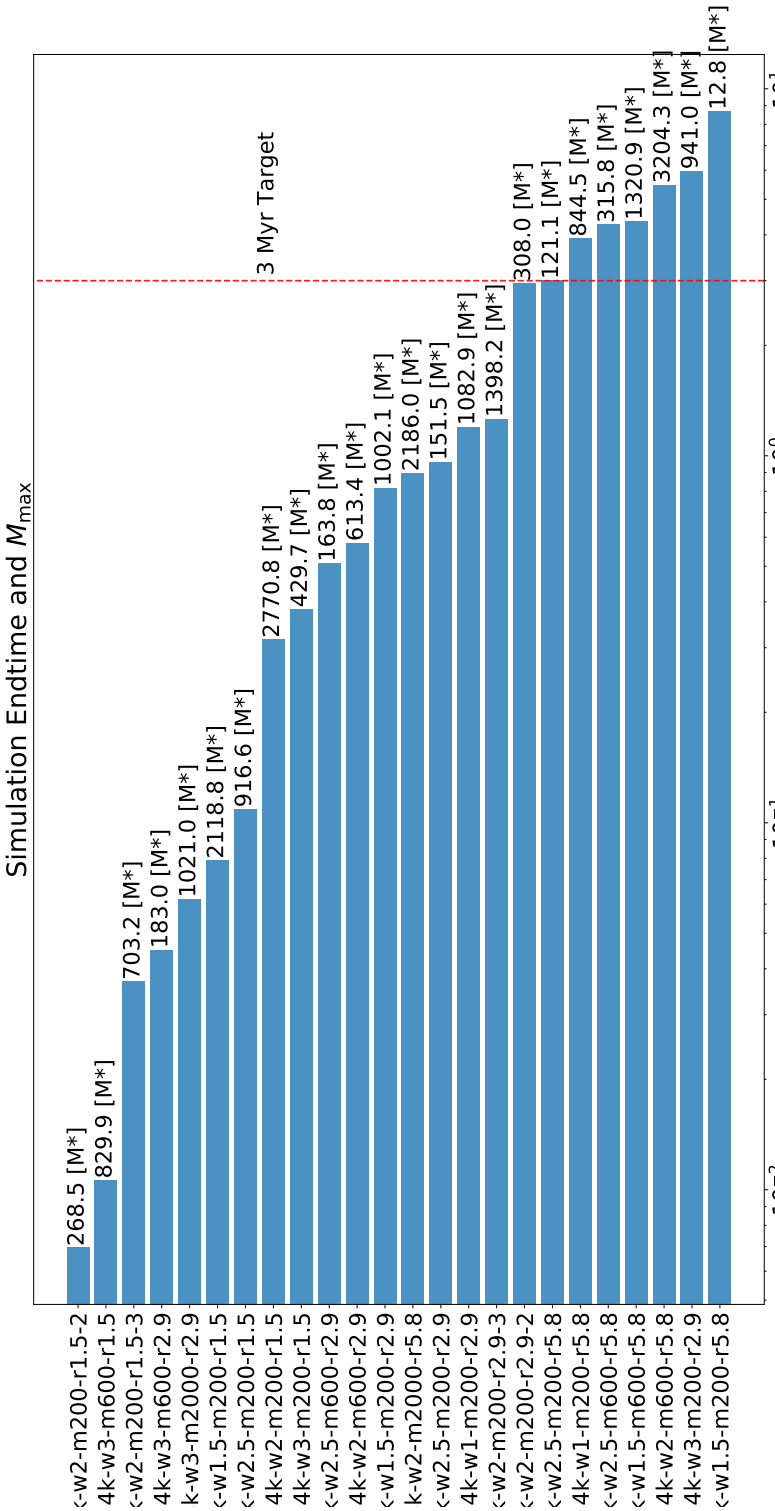


Fig. 6.— Not all jobs ran to completion.

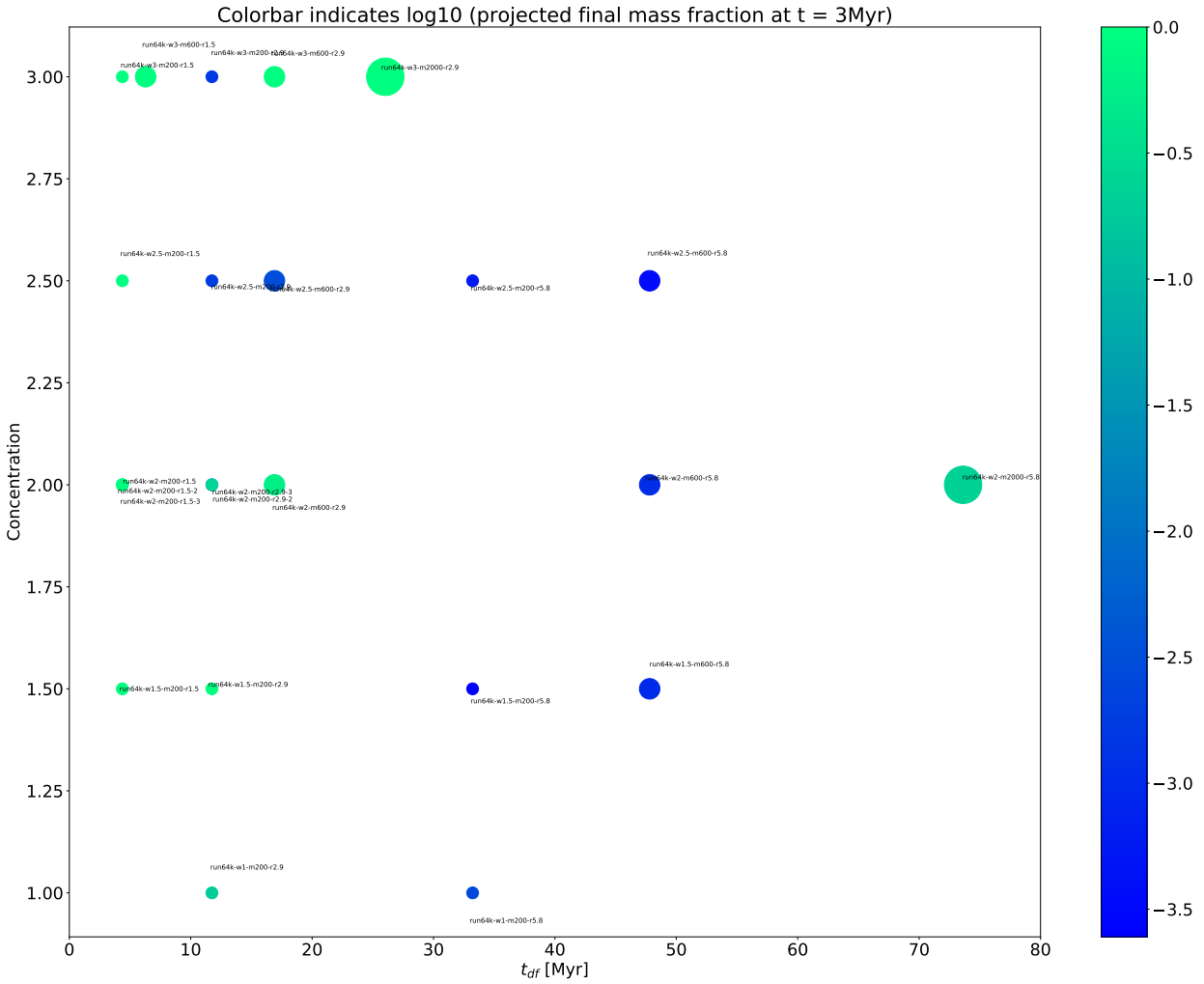


Fig. 7.— The x-axis is the dynamical friction time corresponding to each cluster from equation 2.1. The y-axis is the concentration parameter. The clusters are colored by the log of $\min(M_{\rm max,proj}/M_{\rm total}, 1)$, where $M_{\rm max,proj} = M_{\rm max,0}e^{3\cdot K}$, and the cirlce sizes correspond to the initial mass of the cluster.

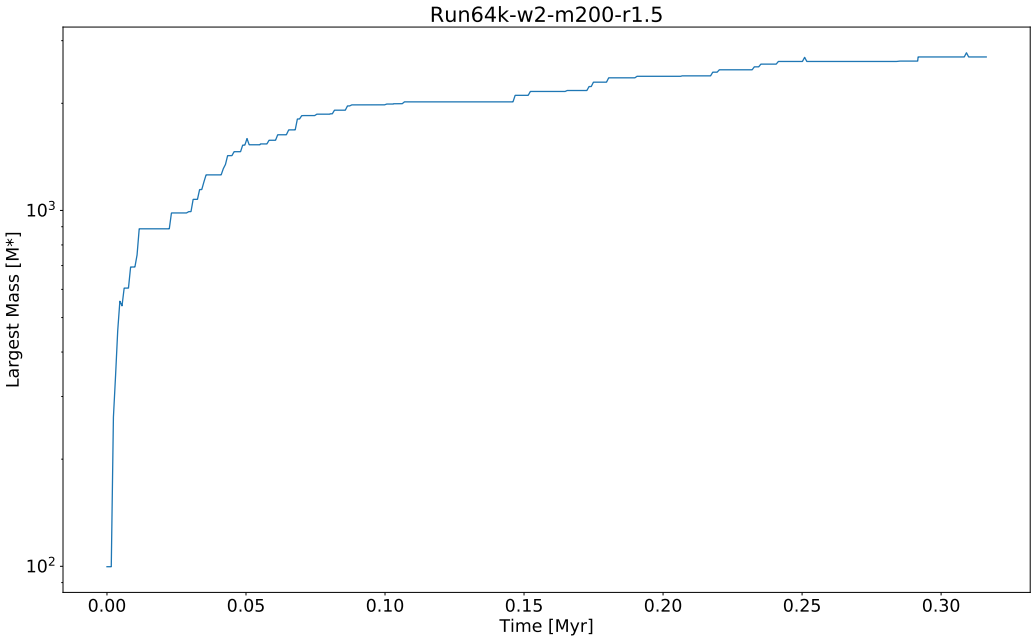

Run64k-w2-m200-r1.5
Largest Mass [M*]
10^3
10^2
0.00
0.05
0.10
0.15
0.20
0.25
0.30
Time [Myr]

Fig. 8.— The

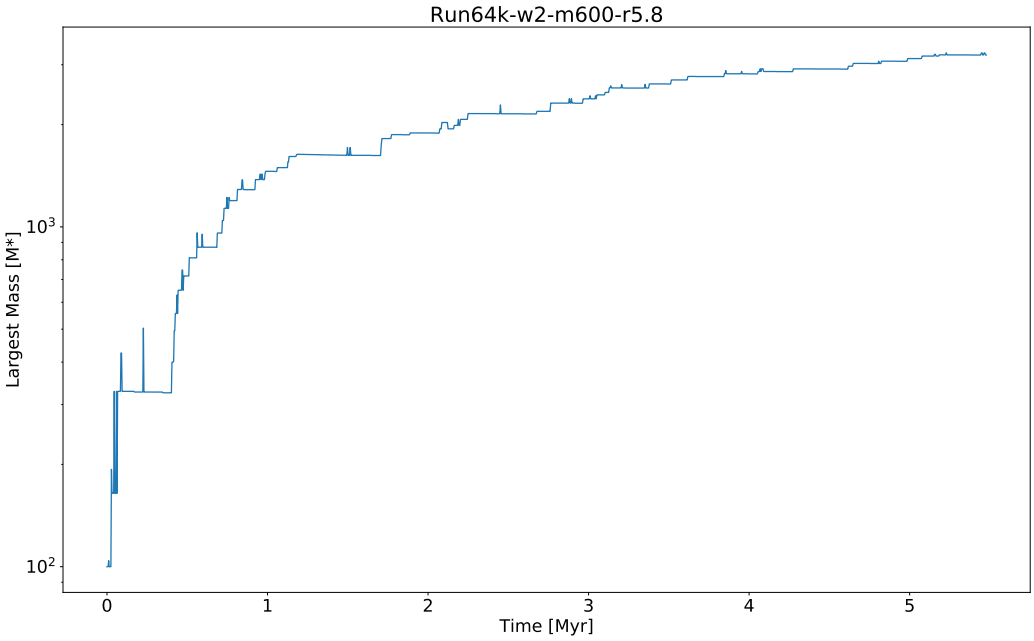


Fig. 9.— placeholder

REFERENCES

????

Aarseth, S. J. 1963, MNRAS, 126, 223

—. 1999, PASP, 111, 1333

Ahmad, A., & Cohen, L. 1973, Journal of Computational Physics, 12, 389

Anders, P., Baumgardt, H., Bissantz, N., & Portegies Zwart, S. 2009, MNRAS, 395, 2304

Anders, P., Baumgardt, H., Gaburov, E., & Portegies Zwart, S. 2012, MNRAS, 421, 3557

Bañados, E., Venemans, B. P., Mazzucchelli, C., et al. 2017, ArXiv e-prints, arXiv:1712.01860

Bromm, V., & Loeb, A. 2003, ApJ, 596, 34

Carraro, G., & Lia, C. 2000, A&A, 357, 977

Chandrasekhar, S. 1964, ApJ, 140, 417

Evstigneeva, E. A., Gregg, M. D., Drinkwater, M. J., & Hilker, M. 2007, AJ, 133, 1722

Hosokawa, T., Yorke, H. W., Inayoshi, K., Omukai, K., & Yoshida, N. 2013, ApJ, 778, 178

Katz, H., Sijacki, D., & Haehnelt, M. G. 2015, MNRAS, 451, 2352

Khalisi, E. and Wang, L. and Spurzem, R. 2017

King, I. 1962, AJ, 67, 471

King, I. R. 1966, AJ, 71, 64

Kochanek, C. S. 1992, ApJ, 385, 604

Kroupa, P. 2001, MNRAS, 322, 231

Küpper, A. H. W., Maschberger, T., Kroupa, P., & Baumgardt, H. 2011, MNRAS, 417, 2300

Kustaanheimo, P., & Stiefel, E. 1965, J. Math. Bd, 218, 27

Latif, M. A., & Ferrara, A. 2016, PASA, 33, e051

Mortlock, D. J., Warren, S. J., Venemans, B. P., et al. 2011, Nature, 474, 616

Nitadori, K., & Aarseth, S. J. 2012, MNRAS, 424, 545

Plummer, H. C. 1911, MNRAS, 71, 460

Portegies Zwart, S. F., Baumgardt, H., Hut, P., Makino, J., & McMillan, S. L. W. 2004, Nature, 428, 724

Portegies Zwart, S. F., & McMillan, S. L. W. 2002, ApJ, 576, 899

Wang, L., Spurzem, R., Aarseth, S., et al. 2015, MNRAS, 450, 4070

Wu, X.-B., Wang, F., Fan, X., et al. 2015, Nature, 518, 512

Zinn, R. 1985, ApJ, 293, 424

This manuscript was prepared with the AAS LaTeX macros v5.2.