UNIVERSIDAD DE CHILE
FACULTAD DE CIENCIAS FÍSICAS Y MATEMÁTICAS
DEPARTAMENTO DE ASTRONOMÍA

STUDY OF THE ROLE OF GAS IN THE FORMATION OF SMBH SEED, THROUGH SIMULATIONS OF NSC WITH GAS EMBEDDED

TESIS PARA OPTAR AL GRADO DE MAGÍSTER EN CIENCIAS, MENCIÓN ASTRONOMÍA

BORIS CUEVAS GOMEZ

PROFESOR GUÍA:
Andrés Escala

MIEMBROS DE LA COMISIÓN:
PROFESOR 2
PROFESOR 3

Este trabajo ha sido parcialmente financiado por:
NOMBRE INSTITUCIÓN

SANTIAGO DE CHILE
2026

RESUMEN DE LA MEMORIA PARA OPTAR
AL TÍTULO DE MAGÍSTER EN CIENCIAS,
MENCIÓN ASTRONOMÍA
POR: BORIS CUEVAS GOMEZ
FECHA: 2026
PROF. GUÍA: ANDRÉS ESCALA

STUDY OF THE ROLE OF GAS IN THE FORMATION OF SMBH SEED, THROUGH SIMULATIONS OF NSC WITH GAS EMBEDDED

Abstractd

Una frase de dedicatoria,
pueden ser dos líneas.

Saludos

Acknowledgments

Acknowledgments

Table of Contents

List of Figures

Chapter 1

Introduction

1.1. SMBH Formation Channels and the Collisional Paradigm

1.2. The Role of Gas in Dense Stellar Systems

1.3. Research Objectives and Thesis Outline

Chapter 2

Theoretical Framework

2.1. Density criterion for stellar collisions

2.2. Stars and gas in equilibrium

Chapter 3

Numerical Methods

3.1. The AMUSE Framework

In this work, we investigate the effect of gas on the formation seeds of SMBHs. In contrast with other works, we do not take into account gas accretion by stars, since we are interested exclusively in the gravitational interaction between gas and stars.

We require, in principle, three codes: one for the stars, one for the gas and a coupling code that handles gravitational interactions between gas and stars.

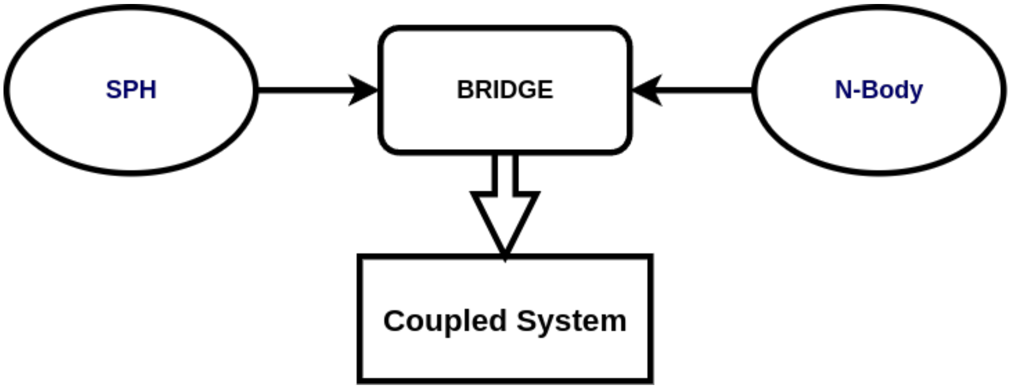

Figure 3.1: Schematic representation of the code coupling framework

The N-Body and SPH codes are accessed through the AMUSE interface, and coupled via an external code.

3.2. N-body code: Ph4

The proposed scenarios presented in (referencia a capitulo) require modeling dense stellar clusters. As we need to consider close encounters and collisions, a high-precision code is necessary.

The PH4 code from the AMUSE interface is used. In this section, we include a description of the code, and some external routines.

3.2.1. Hermite integrator

The Ph4 code uses a fourth-order Hermite integrator (Makino & Aarseth, 1992) for calculating the positions and velocities of stars due to gravity. This is a predictor-corrector method.

The fourth-order Hermite scheme is based on a precise calculation of the individual time step. Considering the particle i with its own time t_i, time step Δt_i, position $\mathbf{x}_i$, velocity $\mathbf{v}_i$, acceleration $\mathbf{a}_i$ and jerk $\dot{\mathbf{a}}_i$, all calculated at time t_i. The integration algorithm proceeds as follows:

1. Select the particle with the minimum $t_i + \Delta t_i$. Set the global time to $t = t_i + \Delta t_i$.
2. Calculate the predicted positions ($\mathbf{x}_p$) and velocities ($\mathbf{v}_p$) for all particles, using the current values for $\mathbf{x}$, $\mathbf{v}$, $\mathbf{a}$, $\dot{\mathbf{a}}$.
3. Calculate the acceleration ($\mathbf{a}_i$) and jerk ($\dot{\mathbf{a}}_i$) for particle i at time $t_i + \Delta t_i$ using the predicted positions and velocities.
4. Calculate the second and third time derivatives of acceleration ($\mathbf{a}_i^{(2)}$ and $\mathbf{a}_i^{(3)}$) using Hermite interpolation.
5. Apply corrections to the position and velocity of particle i.
6. Calculate and update the time step Δt_i.
7. Repeat the algorithm.

A detailed derivation of the algorithm and the full set of equations are provided in Appendix A. From Eqs. (A.2) and (A.3), the predicted positions and velocities are:

$$\mathbf{x}_{p,j} = \mathbf{x}_j + \mathbf{v}_j(t - t_j) + \frac{\mathbf{a}_{0,j}}{2}(t - t_j)^2 + \frac{\dot{\mathbf{a}}_{0,j}}{3!}(t - t_j)^3 \tag{3.1}$$

$$\mathbf{v}_{p,j} = \mathbf{v}_j + \mathbf{a}_{0,j}(t - t_j) + \frac{\dot{\mathbf{a}}_{0,j}}{2!}(t - t_j)^2 \tag{3.2}$$

Here, $\mathbf{x}_j$ and $\mathbf{v}_j$ are the position and velocity of particle j at time t_j. The current acceleration $\mathbf{a}_{0,j}$ and jerk $\dot{\mathbf{a}}_{0,j}$ are computed as:

$$\mathbf{a}_{0,j} = \sum_{k\neq j} Gm_k \frac{\mathbf{r}_{0,jk}}{(\mathbf{r}_{0,jk}^2 + \epsilon^2)^{3/2}} \tag{3.3}$$

$$\dot{\mathbf{a}}_{0,j} = \sum_{k\neq j} Gm_k \left[\frac{\mathbf{v}_{0,jk}}{(\mathbf{r}_{0,jk}^2 + \epsilon^2)^{3/2}} - \frac{3(\mathbf{v}_{0,jk} \cdot \mathbf{r}_{0,jk})\mathbf{r}_{0,jk}}{(\mathbf{r}_{0,jk}^2 + \epsilon^2)^{5/2}} \right] \tag{3.4}$$

$$\mathbf{r}_{0,jk} = \mathbf{r}_k - \mathbf{r}_j \tag{3.5}$$

$$\mathbf{v}_{0,jk} = \mathbf{v}_k - \mathbf{v}_j \tag{3.6}$$

Using Hermite interpolation, the second and third time derivatives of the acceleration can be estimated. This estimation is subsequently used to correct the positions and velocities. The relevant expressions, derived in the appendix, are:

$$\mathbf{a}_{0,i}^{(2)} = \frac{-6(\mathbf{a}_{0,i} - \mathbf{a}_{1,i}) - \Delta t_i(4\dot{\mathbf{a}}_{0,i} + 2\dot{\mathbf{a}}_{1,i})}{\Delta t_i^{\ 2}} \tag{3.7}$$

$$\mathbf{a}_{0,i}^{(3)} = \frac{12(\mathbf{a}_{0,i} - \mathbf{a}_{1,i}) + 6(\dot{\mathbf{a}}_{0,i} + \dot{\mathbf{a}}_{1,i})\Delta t_i}{\Delta t_i^{\ 3}} \tag{3.8}$$

The final corrected position and velocity of particle i are then given by:

$$\mathbf{x}_i(t_i + \Delta t_i) = \mathbf{x}_{p,i} + \frac{\mathbf{a}_{0,i}^{(2)}\Delta t_i^{\ 4}}{24} + \frac{\mathbf{a}_{0,i}^{(3)}\Delta t_i^{\ 5}}{120} \tag{3.9}$$

$$\mathbf{v}_i(t_i + \Delta t_i) = \mathbf{v}_{p,i} + \frac{\mathbf{a}_{0,i}^{(2)}\Delta t_i^{\ 3}}{6} + \frac{\mathbf{a}_{0,i}^{(3)}\Delta t_i^{\ 4}}{24} \tag{3.10}$$

These corrections incorporate terms up to fourth order. The main advantage of this method is that it only requires the explicit calculation of the acceleration and jerk at each step, with higher-order derivatives being obtained through the interpolation scheme.

3.2.2. Block Time Steps

An important part of the Hermite integrator is the time step calculation. The time step controls precision, energy conservation and integration time. While time steps are individual (each particle has its own), they are grouped into blocks for efficiency.

Time steps are set in two steps. First, a dynamic time step is calculated. Then, the time step is assigned using a block scheme. For the dynamical time step, the standard formula (A.35) is used:

$$\Delta t_i = \sqrt{\eta \frac{|\mathbf{a}_{1,i}||\mathbf{a}_{1,i}^{(2)}| + |\dot{\mathbf{a}}_{1,i}|^2}{|\dot{\mathbf{a}}_{1,i}||\mathbf{a}_{1,i}^{(3)}| + |\mathbf{a}_{1,i}^{(2)}|^2}} \tag{3.11}$$

As can be seen, $\mathbf{a}_{1,i}$ and $\dot{\mathbf{a}}_{1,i}$ are calculated in each step, while $\mathbf{a}_{1,i}^{(3)}$ is constant. Only $\mathbf{a}_{1,i}^{(2)}$ needs to be estimated:

$$\mathbf{a}_{1,i}^{(2)} = \mathbf{a}_{0,i}^{(2)} + \Delta t_i \mathbf{a}_{1,i}^{(3)} \tag{3.12}$$

When the simulation begins, the initial time steps are calculated using (A.37):

$$\Delta t = \eta_s \frac{|\mathbf{a}|}{|\dot{\mathbf{a}}|} \tag{3.13}$$

The main algorithm of the Hermite 4th-order scheme evolves the particle with the minimum $t_j + \Delta t_j$. As it is expensive to evolve particles one by one, the goal is to assign particles to discrete time steps. In Ph4, time steps are set to powers of two of the N-body time units.

During initialization, the time step is calculated using Equation (3.13), and then set. Given the initial time step $\Delta t_{0,j}$, the N-body time τ, and an integer n:

$$0.5 < \frac{\tau 2^n}{\Delta t_{0,j}} < 1 \tag{3.14}$$

The time step is then set to $\tau 2^n$, with n being able to be positive or negative.

As the model evolves, the new time step is computed using (3.11). The time step is halved if the calculated time step is smaller than the current time step, and doubled if it is larger than twice the current time step.

The advantage of this time step criterion is that the time step in the code is equal to or lower than that required by the dynamics. Moreover, since particles share the same time steps, they can be integrated simultaneously. Additionally, all particles can be synchronized, as their times are multiples of powers of two of the same time unit.

3.2.3. Collisions and mergers

Stellar collisions occur in two steps. First, the collision is detected by the AMUSE routines in PH4, and then it is handled externally. PH4 has support for stopping conditions, specifically for collision detection.

All particles in AMUSE have different properties: position, velocity, mass, radius, among others. When `collision_detection` is enabled in PH4, the first filter is to check if two particles are closer than the sum of their radii:

$$|\Delta \mathbf{x}| < r_i + r_j \tag{3.15}$$

where $\Delta \mathbf{x}$ is the relative position vector between particles i and j, and r_i, r_j their respective radii. A second filter then verifies close encounters:

$$\epsilon |\Delta \mathbf{x}||\Delta \mathbf{v}| > \Delta \mathbf{x} \cdot \Delta \mathbf{v} \tag{3.16}$$

where $\Delta \mathbf{v}$ is the relative velocity vector and $\epsilon = 0.001$. This avoids cases where particle superposition results from hyperbolic encounters with predominantly tangential relative velocities.

The stopping conditions allow the user to implement custom handling routines. For these simulations, we simply conserve linear momentum. When `collision_detection` is triggered, the code modifies the properties of one particle (preferentially the more massive one), setting its mass to the sum of the masses, its position to the center of mass, its velocity to the center-of-mass velocity, and its radius to conserve mass density:

$$\begin{align}
m_{\text{new}} &= m_i + m_j \tag{3.17}\\
\mathbf{x}_{\text{new}} &= \frac{m_i \mathbf{x}_i + m_j \mathbf{x}_j}{m_i + m_j} \tag{3.18}\\
\mathbf{v}_{\text{new}} &= \frac{m_i \mathbf{v}_i + m_j \mathbf{v}_j}{m_i + m_j} \tag{3.19}\\
r_{\text{new}} &= r_i \left(\frac{m_i + m_j}{m_i} \right)^{1/3} \tag{3.20}\\
& \tag{3.21}
\end{align}$$

Finally, the other particle is removed from the simulation.

3.2.4. Escaping stars

After each analysis time step, potential escapers are identified through a multi-stage detection algorithm. To maintain mass and energy conservation, escaped stars are not removed from the simulation. Instead we add an extra attribute, `escaped` (all stars begin with `particles.escaped = False`), to track bound and unbound stars for subsequent analysis. The process begins with defining an appropriate length scale.

During initialization, the escape scale is defined as the minimum between ten times the virial radius and the Lagrangian radius containing the total mass:

$$r_{0,\mathrm{esc}} = \min(10 \times r_{\mathrm{vir}}, r_{100}) \tag{3.22}$$

where r_{100} represents the Lagrangian radius containing the total stellar mass. In subsequent iterations, the escape distance is updated as:

$$r_{\mathrm{esc}} = \max(r_{\mathrm{esc}}, 10 \times r_{b,\mathrm{vir}}) \tag{3.23}$$

where $r_{b,\mathrm{vir}}$ denotes the virial radius of bound stars.

The escape detection algorithm, executed at each time step, proceeds as follows:

1. Compute the core center and identify stars beyond the escape radius
2. Select gravitationally unbound stars
3. Identify stars with outward radial motion
4. Verify minimal gravitational influence on nearest neighbors

The core center location is determined using a density-weighted scheme following (Sweatman, 1993). For each star, the distance to its 7th nearest neighbor ($r_{i,7}$) is computed, and the density center is calculated as:

$$\mathbf{x}_c = \frac{\sum_{i=1}^{N} \mathbf{x}_i / r_{i,7}^3}{\sum_{i=1}^{N} 1/r_{i,7}^3} \tag{3.24}$$

This formulation resembles a center of mass calculation but uses local stellar density instead of mass. Stars satisfying the condition

$$|\mathbf{x}_i - \mathbf{x}_c| > r_{\mathrm{esc}} \tag{3.25}$$

are identified as potential escapers.

Gravitational unbinding is evaluated using the total energy $E_i = E_{p,i} + E_{k,i}$, where $E_{p,i}$ is the gravitational potential energy (including contributions from both stars and gas) and $E_{k,i}$ is the kinetic energy. Stars with $E_i \leq 0$ are considered bound, while those with $E_i > 0$ are classified as unbound. Among unbound stars, outward-moving candidates are identified through the condition:

$$\Delta\mathbf{x}_i \cdot \Delta\mathbf{v}_i > 0 \tag{3.26}$$

where $\Delta\mathbf{x}_i = \mathbf{x}_i - \mathbf{x}_c$ and $\Delta\mathbf{v}_i = \mathbf{v}_i - \mathbf{v}_c$ represent position and velocity relative to the cluster center. Similar to (3.24), the velocity of the core center is computed as

$$\mathbf{v}_c = \frac{\sum_{i=1}^{N} \mathbf{v}_i m_i / r_{i,7}^3}{\sum_{i=1}^{N} m_i / r_{i,7}^3} \tag{3.27}$$

The final validation step quantifies the gravitational influence on neighboring stars. For each candidate escaper, the relative potential energy ($e_{p,i}$) and kinetic energy ($e_{k,i}$) with respect to its nearest neighbor are computed. The energy ratio is defined as:

$$X_E := \frac{|e_{p,i}|}{e_{k,i}} \tag{3.28}$$

Stars satisfying $X_E < 0.01$ are confirmed as escapers, ensuring minimal gravitational perturbation to neighboring stars. These stars are tagged as escapers (`particles.escaped = True`).

3.3. SPH code: Fi

For the gas dynamics we use the Fi code, which is an MPI-parallel smoothed-particle hydrodynamics (SPH) code implemented in the Amuse interface. Here we describe the most important features, but a more detailed description is given in (Pelupessy, 2005).

3.3.1. Barnes-Hut tree

One important characteristic of the Fi code, is that compute the self gravity of the system. For particles codes, there are bunch of methods for calculating the gravity of the system. The most precise solutions calculates one-vs-all interactions. This kind of solutions has high precision, but are costly, as they need $\mathcal{O}(N^2)$ operations per step in each force evaluation. As this kind of precision is usually required for collisional scenarios, in the case of collisionless simulations approximate methods are used instead, scarifying accuracy for speed.

In the case of Fi, the force is compute using the Barnes-Hut (BH) method (Barnes & Hut, 1986). The basis of this method is the use of an hierarchical organization of particles on oct-tree cells, and a distance criteria for force calculation.

First, an initial cell is defined as a physical cube that contains all particles, and is represented as a node in a tree. If the initial cell (cube) contains more than one particle, is divided into eight new cells, of equal size. Each cell is linked to the progenitor node. Now, each new cell is iteratively divided into eight cells, until contains exactly one particle (leaf cell). The figure 3.2 represent an oct-tree in two dimensions.

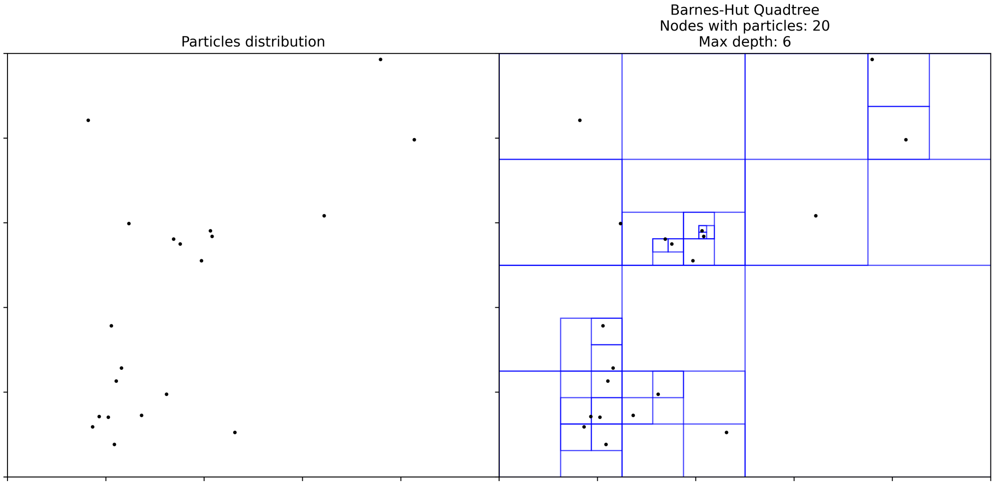

Figure 3.2: Representation of an oct tree

Once the oct-tree is setted, an opening criteria is computed for calculating the gravitational force. Given a particle, the nodes are iteratively opened until a leaf is reached in the tree or the opening criteria is fulfilled:

$$l/d < \theta \tag{3.29}$$

where l is the size of the cell, d is the distance between the particle and the center of mass of the particles into the cell, and θ the parameter of the cell opening criteria that controls the accuracy. If the criteria is fulfilled, the particles in the cell are approximated as one particle, which mass is the total mass of the particle, and with the center of mass as the position. The principle of the criteria is that for a system of particles which size is small enough compared with the distance, can be approximated as a point. One of the advantages of this criteria is that the force calculus scale as $\mathcal{O}(N \log N)$, and can work efficiently with different mass distributions. It has been shown that it is acceptable to use BH tree method for collisionless N-body dynamics (Barnes & Hut, 1986). The implementation of BH method in Fi has a modified cell opening criteria. As the classical BH opening criterion can fail in some circumstances (Salmon & Warren, 1994), the criterion of (Springel & Hernquist, 2002) is used instead:

$$M_n l^4 < \alpha |\mathbf{a}| d^6 \tag{3.30}$$

where M_n is the total mass contained in the node, $\mathbf{a}$ is an estimation of the acceleration (generally the acceleration of the previous step), and α is a parameter to control accuracy.

3.3.2. Smoothed Particle Hydrodynamics

3.4. Coupling strategy: Bridge

Chapter 4

Initial Conditions and Setup

4.1. Stellar Cluster Models

4.2. Gas Configurations

4.3. Numerical Parameters

Chapter 5

Results and analysis

5.1. Gas-Free Baseline Simulations

5.2. Effects of Gas Fraction on Collision Rates

5.3. Mass Spectrum Evolution and Massive Object Formation

5.4. Comparative Analysis Across Gas Fractions

Chapter 6

Discussion and Conclusions

6.1. Interpretation of Gas-Induced Enhancements

6.2. Implications for SMBH Seed Formation

6.3. Limitations and Future Work

References

Aarseth, S. J. (1985). Direct methods for n-body simulations. In J. U. Brackbill & B. I. Cohen (Eds.), *Multiple Time Scales* (pp. 377–418). Academic Press.

Barnes, J. & Hut, P. (1986). A hierarchical o (n log n) force-calculation algorithm. *nature*, 324(6096), 446–449.

Makino, J. & Aarseth, S. J. (1992). On a hermite integrator with ahmad-cohen scheme for gravitational many-body problems. *Publications of the Astronomical Society of Japan*, 44(2), 141–151.

Pelupessy, F. I. (2005). *Numerical studies of the interstellar medium on galactic scales*. PhD thesis, éditeur non identifié.

Salmon, J. K. & Warren, M. S. (1994). Skeletons from the treecode closet. *Journal of Computational Physics*, 111(1), 136–155.

Springel, V. & Hernquist, L. (2002). Cosmological smoothed particle hydrodynamics simulations: the entropy equation. *Monthly Notices of the Royal Astronomical Society*, 333(3), 649–664.

Sweatman, W. L. (1993). A study of Lagrangian radii oscillations and core-wandering using N-body simulations. *MNRAS*, 261(3), 497–512.

Appendices

Appendix A. Fourth-Order Hermite Integration Scheme

The N-body code used in this work, `ph4`, is based on a fourth-order Hermite integrator. The following subsections describe in detail the individual time step scheme (ITS) from (Makino & Aarseth, 1992), including the time step criterion in the N-body code.

A.1. Step 1: Particle selection

The first step is to select the next particle to be advanced. At any given time, each particle j has its own mass m_j, time t_j, time step Δt_j, coordinates $\mathbf{x}_j$ and $\mathbf{v}_j$, acceleration $\mathbf{a}_j$ and jerk $\dot{\mathbf{a}}_j$ (with $\dot{\mathbf{a}}_j = \partial \mathbf{a}_j / \partial t$). The selected particle i is the one satisfying the condition:

$$i = \arg\min_j (t_j + \Delta t_j) \tag{A.1}$$

Note that always $t_j \leq t_i + \Delta t_i$, for all j. The current simulation time is then set to $t = t_i + \Delta t_i$.

A.2. Step 2: Calculate the predicted position and velocities for all particles

The positions and velocities of all particles are predicted at time t, using a Taylor expansion up to third-order (jerk):

$$\mathbf{x}_{p,j} = \mathbf{x}_j + \mathbf{v}_j(t - t_j) + \frac{\mathbf{a}_{0,j}}{2!}(t - t_j)^2 + \frac{\dot{\mathbf{a}}_{0,j}}{3!}(t - t_j)^3 \tag{A.2}$$

$$\mathbf{v}_{p,j} = \mathbf{v}_j + \mathbf{a}_{0,j}(t - t_j) + \frac{\dot{\mathbf{a}}_{0,j}}{2!}(t - t_j)^2 \tag{A.3}$$

The acceleration is due only to gravitational force. The expression for each particle j is:

$$\mathbf{a}_{0,j} = \sum_{k \neq j} G m_k \frac{\mathbf{r}_{0,jk}}{(\mathbf{r}_{0,jk}^2 + \epsilon^2)^{3/2}} \tag{A.4}$$

where ϵ is the softening parameter, and

$$\mathbf{r}_{0,jk} = \mathbf{r}_k - \mathbf{r}_j \tag{A.5}$$

Differentiating Eq. (A.4) with respect to time yields the jerk:

$$\dot{\mathbf{a}}_{0,j} = \frac{\partial \mathbf{a}_j}{\partial t} \tag{A.6}$$

$$= \sum_{k \neq j} G m_k \frac{\partial}{\partial t} \frac{\mathbf{r}_{0,jk}}{(\mathbf{r}_{0,jk}^2 + \epsilon^2)^{3/2}} \tag{A.7}$$

$$\dot{\mathbf{a}}_{0,j} = \sum_{k \neq j} G m_k \left[\frac{\mathbf{v}_{0,jk}}{(\mathbf{r}_{0,jk}^2 + \epsilon^2)^{3/2}} - \frac{3(\mathbf{v}_{0,jk} \cdot \mathbf{r}_{0,jk})\mathbf{r}_{0,jk}}{(\mathbf{r}_{0,jk}^2 + \epsilon^2)^{5/2}} \right] \tag{A.8}$$

where

$$\mathbf{v}_{0,jk} = \mathbf{v}_k - \mathbf{v}_j \tag{A.9}$$

A.3. Step 3: Calculate acceleration and jerk for particle i

Similar to equations (A.4) and (A.8), the acceleration and jerk are calculated using the predicted positions and velocity:

$$\mathbf{a}_{1,i} = \sum_{j \neq i} G m_j \frac{\mathbf{r}_{ij}}{(\mathbf{r}_{ij}^2 + \epsilon^2)^{3/2}} \tag{A.10}$$

$$\dot{\mathbf{a}}_{1,i} = \sum_{j \neq i} G m_j \left[\frac{\mathbf{v}_{ij}}{(\mathbf{r}_{ij}^2 + \epsilon^2)^{3/2}} - \frac{3(\mathbf{v}_{ij} \cdot \mathbf{r}_{ij})\mathbf{r}_{ij}}{(\mathbf{r}_{ij}^2 + \epsilon^2)^{5/2}} \right] \tag{A.11}$$

$$\mathbf{r}_{ij} = \mathbf{r}_{p,j} - \mathbf{r}_{p,i} \tag{A.12}$$

$$\mathbf{v}_{ij} = \mathbf{v}_{p,j} - \mathbf{v}_{p,i} \tag{A.13}$$

In this notation, the subscript 0 denotes quantities evaluated at the beginning of the time step (t_j), while the subscript 1 denotes quantities evaluated at the end of the time step (t), using the predicted position and velocity ($\mathbf{x}_{p,i}$ and $\mathbf{v}_{p_i}$).

A.4. Step 4: Second and third time derivative of acceleration, using Hermite interpolation

To obtain a high-order correction of $\mathbf{x}$ and $\mathbf{v}$, the acceleration of particle i is modeled as a third-order polynomial in time:

$$\mathbf{a}(\tau) = \alpha\tau^3 + \beta\tau^2 + \gamma\tau + \delta \tag{A.14}$$

The successive time derivatives of $\mathbf{a}(\tau)$ are then:

$$\dot{\mathbf{a}}(\tau) = 3\alpha\tau^2 + 2\beta\tau + \gamma \tag{A.15}$$

$$\mathbf{a}^{(2)}(\tau) = 6\alpha\tau + 2\beta \tag{A.16}$$

$$\mathbf{a}^{(3)}(\tau) = 6\alpha \tag{A.17}$$

The coefficients α, β, γ, and δ are determined by using the known values of acceleration and jerk at times t_i and t. This provides a system of four equations:

$$\mathbf{a}(t_i) = \mathbf{a}_{0,i} = \alpha t_i^3 + \beta t_i^2 + \gamma t_i + \delta \tag{A.18}$$

$$\mathbf{a}(t) = \mathbf{a}_{1,i} = \alpha t^3 + \beta t^2 + \gamma t + \delta \tag{A.19}$$

$$\dot{\mathbf{a}}(t_i) = \dot{\mathbf{a}}_{0,i} = 3\alpha t_i^2 + 2\beta t_i + \gamma \tag{A.20}$$

$$\dot{\mathbf{a}}(t) = \dot{\mathbf{a}}_{1,i} = 3\alpha t^2 + 2\beta t + \gamma \tag{A.21}$$

where $t = t_i + \Delta t_i$ is the current time. As the system of equations is linearly independent, the four coefficients can be determined. However, determining α and β is enough to get the second and third time derivative of acceleration. From (A.21):

$$\gamma = \dot{\mathbf{a}}_{1,i} - 3\alpha t^2 - 2\beta t \tag{A.22}$$

Subtracting Eq. (A.20) from (A.21):

$$\beta = \frac{(\dot{\mathbf{a}}_{1,i} - \dot{\mathbf{a}}_{0,i}) - \alpha 3(t^2 - t_i^2)}{2(t - t_i)} \tag{A.23}$$

Subtracting Eq. (A.18) from (A.19), and using results from (A.22) and (A.23) gives an expression for α:

$$\alpha = \frac{(\dot{\mathbf{a}}_{1,i} + \dot{\mathbf{a}}_{0,i})(t - t_i) - 2(\mathbf{a}_{1,i} - \mathbf{a}_{0,i})}{(t - t_i)^3} \tag{A.24}$$

Replacing this result in Eq. (A.23):

$$\beta = \frac{-\dot{\mathbf{a}}_{1,i}(t + 2t_i) - \dot{\mathbf{a}}_{0,i}(2t + t_i)}{(t - t_i)^2} + \frac{3(\mathbf{a}_{1,i} - \mathbf{a}_{0,i})(t + t_i)}{(t - t_i)^3} \tag{A.25}$$

The second and third time derivatives of the acceleration at time t_i can now be evaluated. For the Eq. (A.17), and using $t - t_i = \Delta t_i$:

$$\mathbf{a}_{0,i}^{(3)} = \mathbf{a}^{(3)}(t_i) = 6\alpha \tag{A.26}$$

$$\mathbf{a}_{0,i}^{(3)} = 6\frac{(\dot{\mathbf{a}}_{1,i} + \dot{\mathbf{a}}_{0,i})\Delta t_i - 2(\mathbf{a}_{1,i} - \mathbf{a}_{0,i})}{\Delta t_i{}^3} \tag{A.27}$$

$$\mathbf{a}_{0,i}^{(3)} = \frac{12(\mathbf{a}_{0,i} - \mathbf{a}_{1,i}) + 6(\dot{\mathbf{a}}_{0,i} + \dot{\mathbf{a}}_{1,i})\Delta t_i}{\Delta t_i{}^3} \tag{A.28}$$

Evaluating the second time derivative in t_i, and replacing the obtained for α and β:

$$\mathbf{a}_{0,i}^{(2)} = \mathbf{a}^{(2)}(t_i) = 6\alpha t_i + \beta \tag{A.29}$$

$$\mathbf{a}_{0,i}^{(2)} = \frac{-6(\mathbf{a}_{0,i} - \mathbf{a}_{1,i}) - \Delta t_i(4\dot{\mathbf{a}}_{0,i} + 2\dot{\mathbf{a}}_{1,i})}{\Delta t_i{}^2} \tag{A.30}$$

A.5. Step 5: Add corrections

The next step in the scheme is to correct position and velocity of particle i at time t. Using a higher-order Taylor expansion around t_i:

$$\mathbf{x}_i(t_i+\Delta t_i) = \mathbf{x}_i(t_i) + \dot{\mathbf{x}}_i(t_i)\Delta t_i + \frac{\mathbf{x}_i{}^{(2)}(t_i)\Delta {t_i}^2}{2!} + ... + \frac{\mathbf{x}_i{}^{(5)}(t_i)\Delta {t_i}^5}{5!} \tag{A.31}$$

$$= \mathbf{x}_i + \mathbf{v}_i\Delta t_i + \frac{\mathbf{a}_{0,i}\Delta {t_i}^2}{2} + \frac{\dot{\mathbf{a}}_{0,i}\Delta {t_i}^3}{6} + \frac{\mathbf{a}_{0,i}^{(2)}\Delta {t_i}^4}{24} + \frac{\mathbf{a}_{0,i}^{(3)}\Delta {t_i}^5}{120} \tag{A.32}$$

The first four terms of this expansion correspond to the predicted position $\mathbf{x}_{p,i}$ from Eq. (A.2). Therefore, the corrected position can be written as:

$$\mathbf{x}_i(t_i+\Delta t_i) = \mathbf{x}_{p,i} + \frac{\mathbf{a}_{0,i}^{(2)}\Delta {t_i}^4}{24} + \frac{\mathbf{a}_{0,i}^{(3)}\Delta {t_i}^5}{120} \tag{A.33}$$

In a similar way, the corrected velocity is:

$$\mathbf{v}_i(t_i+\Delta t_i) = \mathbf{v}_{p,i} + \frac{\mathbf{a}_{0,i}^{(2)}\Delta {t_i}^3}{6} + \frac{\mathbf{a}_{0,i}^{(3)}\Delta {t_i}^4}{24} \tag{A.34}$$

The advantage of this scheme is that achieving high integration accuracy only requires the explicit calculation of the acceleration and jerk for all particles, and the predicted acceleration and jerk for the particle being advanced. This results in an integration scheme that is fourth-order accurate in time ($\mathcal{O}(a^{(3)})$).

A.6. Step 6: Time step update

After updating the position and velocity of particle i, a new time step Δt_i must be calculated. This step is crucial for controlling the integration error. A proven and stable criterion is given by the standard formula (Aarseth, 1985):

$$\Delta t_i = \sqrt{\eta \frac{|\mathbf{a}_{1,i}||\mathbf{a}_{1,i}^{(2)}| + |\dot{\mathbf{a}}_{1,i}|^2}{|\dot{\mathbf{a}}_{1,i}||\mathbf{a}_{1,i}^{(3)}| + |\mathbf{a}_{1,i}^{(2)}|^2}} \tag{A.35}$$

The values for $\mathbf{a}_{1,i}$ and $\dot{\mathbf{a}}_{1,i}$ are known from the direct calculation in Step 3. The value of $\mathbf{a}^{(3)}(t)$ is constant, since a third-order polynomial interpolation is used. Only $\mathbf{a}_{1,i}^{(2)}$ should be calculated:

$$\mathbf{a}_{1,i}^{(2)} = \mathbf{a}_{0,i}^{(2)} + \Delta t_i \mathbf{a}_{0,i}^{(3)} \tag{A.36}$$

During the initialization of the algorithm, the higher-order derivatives of the acceleration are not available, so an alternative formula for time step can be used in the startup:

$$\Delta t = \eta_s \frac{|\mathbf{a}|}{|\dot{\mathbf{a}}|} \tag{A.37}$$

The suggested value for the startup parameter is $\eta_s \sim 0.01$ (Makino & Aarseth, 1992).