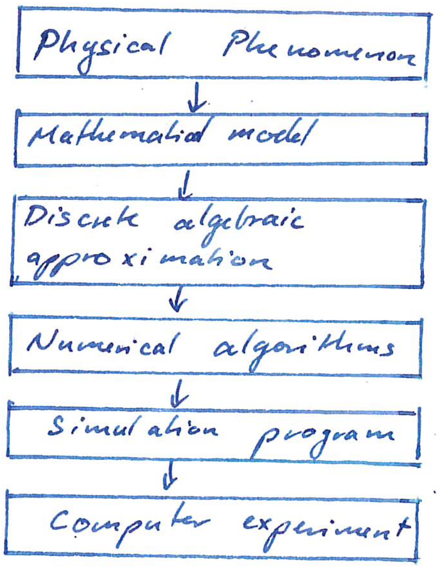
Computational Astrophysics

1. Computational Experiment



A physical phenomenon is described by some mathematical model. It is a model because such equations are often based on some approximation. Mathematical models usually are a set of ordinary or partial differential equations, that describe the physical system.

The mathematical model is discretized to yield a simpler form of the equations that can be efficiently treated numerically.

The simulation program is a collection of many numerical algorithms, each of them designed to solve the discretized equations describing the mathematical model.

The computer experiment (simulation) is a product of the simulation program, which normally can be capable of generating many simulations of many different systems.

2. Collisionless N-body Dynamics

In some cases (stars in a galaxy, dark matter), the number of parti
interactions is simply too great to compute individually. We have to
discretise, creating "superparticles" or "mesh points", each of whic
represents a cluster of stars etc. of $\sim 10^4 - 10^5$ typical size.
The discretized model has to be based on a robust mathematic
model to be physically meaningful.

2.1 The hierarchy of particle distribution function

Any system of particles (or superparticles) can be described by a distribution function f, which expresses the probability density to have a particle with position between $\vec{x}$ and $d\vec{x}$ and velocity between $\vec{v}$ and $d\vec{v}$ at the time t.

The state of an N-particle ensemble at time t can be specified by the exact distribution function:

$$F(\vec{r},\vec{v},t) = \sum_{i=1}^{N} \delta(\vec{r}-\vec{r}_i(t)) \cdot \delta(\vec{v}-\vec{v}_i\,(t))$$

This effectively gives the number density of particles in phase space point $(\vec{r},\vec{v})$.

Let

$$\rho(\vec{r}_1,\vec{r}_2,\dots,\vec{r}_N,\vec{v}_1,\vec{v}_2,\dots,\vec{v}_N)\,d\vec{r}_1\,d\vec{r}_2\dots d\vec{v}_1\,d\vec{v}_2\dots d\vec{v}_N$$

be the probability that the system is in the given state at time t. Then a reduced statistical description is given by ensemble averaging:

$$f_1(\vec{r},\vec{v},t) = \langle F(\vec{r},\vec{v},t)\rangle = \int F\cdot\rho\cdot d\vec{r}_1\,d\vec{r}_2\dots d\vec{r}_N\,d\vec{v}_1\dots d\vec{v}_N$$

$$= N\int \rho(\vec{r}_1=\vec{r},\vec{r}_2,\dots,\vec{v}_1,\vec{v}_2\dots)\,d\vec{r}_2\,d\vec{r}_3\dots d\vec{v}_2\dots d\vec{v}_n$$

$\int\delta(\vec{r}_1)\delta($
$\swarrow$ assuming yo
can interchange
$\vec{r}_i, d\vec{r}_i \rightleftarrows \vec{r}_j,$

assuming all particles are identical.

$f_1(\vec{r},\vec{v},t)\,d\vec{r}\,d\vec{v}$ gives now the mean number of particles in a phase space volume around $(\vec{r},\vec{v})$

Similarly, the ensemble-averaged two-particle distribution is given by

$$f_2(\vec{r}, \vec{v}, \vec{r}', \vec{v}', t) = \langle F(\vec{r}, \vec{v}, t)\, F(\vec{r}', \vec{v}', t) \rangle$$
$$= N(N-1) \int \rho(\vec{r}, \vec{r}', \vec{r}_3, \dots, \vec{r}_N, \vec{v}, \vec{v}', \vec{v}_3 \dots \vec{v}_N)$$

This gives the mean product of numbers of particles at $(\vec{r}, \vec{v})$ and $(\vec{r}', \vec{v}')$.

Likewise, one may define f_3, f_4 etc. This yields the BBGKY chain (Bogoliubov-Born-Green-Kirkwood-Yvon).

Formally, the statistical mechanics description of correlated systems is given by Liouville's conservation equation for probability density ρ in Γ-space:

$$\frac{\partial \rho}{\partial t} + \sum_i v_i \cdot \frac{\partial \rho}{\partial x_i} + \sum_i \frac{\mathcal{F}_i}{m_i} \cdot \frac{\partial \rho}{\partial v_i} = 0$$

where $\rho(\vec{x}, \vec{v}, t)\, d^{3N}x\, d^{3N}\vec{v}$ is the probability that the system is in the phase space volume $d^{3N}x\, d^{3N}v$ of Γ space at ~~time~~ t and $\mathcal{F}_i$ is the force acting on particle i: $\mathcal{F}_i = \frac{d}{dt}(m_i v_i)$

Theoretically, a series of experiments should be performed to provide an ensemble to define ρ. In practice, time averaging generally replaces ensemble averaging in computing expectation values for near equilibrium configuration. In nonequilibrium calculations, the importance of initial conditions cannot be overlooked.

Analytically, the Liouville equation is no more managable than the complete set of particle orbits. TO make progress, more information must be discarded. This is done by defining a hierarchy of distribution functions and obtaining a hierarchy of kinetic equations by integrating the Liouville equation over successively fewer particle coordinates (BBGYK-chain), as was done for f_1, f_2 ...

A differential equation describing the evolution of distribution function f_s is obtained from Liouville's equation by integrating over variables associated with $N-s$ particles. E.g. the equation for f_1 is

$$\frac{\partial f_1}{\partial t} + \vec{v}_1 \cdot \frac{\partial f_1}{\partial \vec{x}_1} + \underbrace{\frac{\vec{F}_1^{ext}}{m_1} \cdot \frac{\partial f_1}{\partial \vec{v}_1}}_{\text{external force term}} + \underbrace{\int \frac{\vec{F}_{12}}{m_1} \frac{\partial f_2}{\partial \vec{v}_1} d\vec{x}_2 \, d\vec{v}_2}_{\text{interparticle force term}} = 0$$

The evolution equation for f_s involves a term in f_{s+1}. Some approximations are required to close the chain of equations.

Uncorrelated Systems

The simplest closure for the BBGKY-chain is to assume that particles are uncorrelated, i.e. that we have

$$f_2(\vec{r}, \vec{v}, \vec{r}', \vec{v}', t) = f_1(\vec{r}, \vec{v}, t)\, f_1(\vec{r}', \vec{v}', t)$$

Physically, this means that a particle at $(\vec{r}, \vec{v})$ is completely unaffected by one at $(\vec{r}', \vec{v}')$ [or that the potential is negligible compared to the kinetic energy of the particles]

The dynamical equations for the follow from the notion of conservation of Probability in phase space:

$$P = \int_V d^6\vec{w} \; (\vec{w}) = \text{const} \Rightarrow \frac{dP}{dt} = 0 \Rightarrow \quad 0 = \frac{\partial}{\partial t} + \frac{\partial}{\partial \vec{w}}(\dot{\vec{w}})$$

$$\frac{\partial \rho}{\partial t} + \frac{\partial}{\partial \vec{w}}(\rho \dot{\vec{w}}) = 0 = \frac{\partial \rho}{\partial t} + \left(\frac{\partial \rho}{\partial \vec{w}} \dot{\vec{w}} + \rho \frac{\partial \dot{\vec{w}}}{\partial \vec{w}} \right)$$

$$= \frac{\partial \rho}{\partial t} + \sum_i \left(\frac{\partial \rho}{\partial \vec{r}_i} \dot{\vec{r}}_i + \rho \frac{\partial \dot{\vec{r}}_i}{\partial \vec{r}_i} + \frac{\partial \rho}{\partial \vec{v}_i} \dot{\vec{v}}_i + \rho \frac{\partial \dot{\vec{v}}_i}{\partial \vec{v}_i} \right)$$

With $\dot{\vec{r}} = \frac{\partial H}{\partial \vec{p}}$ and $\dot{\vec{p}} = -\frac{\partial H}{\partial \vec{r}}$:

$$\frac{\partial \dot{\vec{r}}}{\partial \vec{r}} = \frac{\partial^2 H}{\partial \vec{r}\, \partial \vec{p}} = -\left(-\frac{\partial^2 H}{\partial \vec{p}\, \partial \vec{r}} \right) = -\frac{\partial \dot{\vec{p}}}{\partial \vec{p}}$$

$$\Rightarrow \frac{\partial \rho}{\partial t} + \frac{\partial}{\partial \vec{w}}(\rho \dot{\vec{w}}) = \frac{\partial \rho}{\partial t} + \sum_i \left(\frac{\partial \rho}{\partial r_i} \dot{r}_i + \rho \frac{\partial \dot{r}_i}{\partial r_i} + \frac{\partial \rho}{\partial v_i} \dot{v}_i + \rho \frac{\partial \dot{v}_i}{\partial v_i} \right)$$

$$= \frac{\partial \rho}{\partial t} + \sum_i \left(\vec{v}_i \frac{\partial \rho}{\partial \vec{r}_i} + a_i \frac{\partial \rho}{\partial \vec{v}_i} \right)$$

When $\vec{a}_i = \dot{\vec{v}}_i = \frac{\vec{F}_i}{m_i}$ is the particle acceleration and m_i the particle mass. This is Liouville's theorem.

In the collisionless/uncorrelated limit, this directly carries over to the one-point distribution function $f = f_1$ if we integrate over all particle coordinates except one, yielding the Vlasov equation / Collisionless Boltzmann equation:

$$\frac{df}{dt} = \frac{\partial f}{\partial t} + \vec{v}\,\frac{\partial f}{\partial \vec{r}} + \vec{a}\,\frac{\partial f}{\partial \vec{v}} = 0$$

About the acceleration

The acceleration can't be due to another single particle. However, collective effects, for example due to the gravitational field produced by the whole system, are still allowed.

The number density of the system is

$$n = \int f(\vec{r}, \vec{v}, t)\, d\vec{v} = n(\vec{r}, t)$$

and the mass density is

$$\rho = mn = m \int f(\vec{r}, \vec{v}, t)\, d\vec{v} \quad , \quad m = \text{particle mass}$$

The gravitational field of the mass density is given by the Poisson equation:

$$\nabla^2 \phi(\vec{r}) = 4\pi G \rho(\vec{r}, t)$$

while the acceleration is given by

$$\vec{a} = -\nabla \phi = -\frac{\partial \phi}{\partial \vec{r}}$$

2.2 The relaxation time

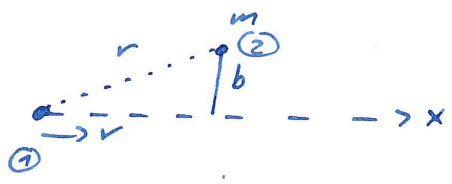
When can a system be considered collisionless?

Consider a system of size R containing N particles. The time for one crossing of a particle through the system is of order $t_{cross} \sim \frac{R}{v}$, where v is the typical particle velocity. For a self-gravitating system of size R we expect

$$v^2 \sim \frac{GNm}{R} = \frac{GM}{R}$$

Now estimate the rate at which a particle experiences weak deflections by other particles. This is the process that violates perfect collisionless behaviour and induces relaxation.

Assume particle ① has a straight path past some particle ② with impact parameter b. Assuming the perturber's influence is small, we take the particle's path to be and remain straight.



Then the force acting on the particle is given by

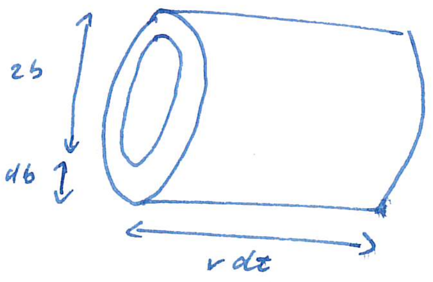
$$F = F_\perp + \underbrace{F_\parallel}_{\approx 0} = F_\perp = -\frac{Gm^2}{r^2}\frac{r_\perp}{r} = -\frac{Gm^2}{(x^2+b^2)^{3/2}}\cdot b$$

$$= -\frac{Gm^2}{(v^2t^2+b^2)^{3/2}}\cdot b$$

The change in momentum for the particle is

$$\Delta p = \int F_\perp\, dt = -\int \frac{Gm^2}{(v^2t^2+b^2)^{3/2}}\cdot b\,dt = -\frac{2Gm^2}{bv} = m\Delta v$$

This is valid for one encounter, but during one transversation of the system, we expect many encounters. These numerous encounters can be described as a "mass flux": $\dot{m} = m\frac{dN}{dt}$



Describe the problem as a particle flying through a cylinder. The number of particles in the cylindrical shell is

$$dN = 2\pi\, db\, v dt \cdot n \implies \frac{dN}{dt} = 2\pi\, db \cdot n \cdot v$$

We define the relaxation time as follows:

$$\frac{1}{t_{relax}} \equiv \frac{1}{E_{kin,loss}} \frac{dE_{kin,loss}}{dt}$$

Where $E_{kin,loss}$ is the lost kinetic energy of a particle due to collisions. (In future: $E_{kin} \equiv E_{kin,loss}$)

The typical kinetic energy is given by $E_{kin} = \frac{1}{2} m v^2 \sim \frac{GmM}{R}$

The loss of kinetic energy may be described as

$$\frac{dE_{kin}}{dt} = \frac{1}{2} \int (\Delta v)^2 \dot{m} = \frac{1}{2} \int_0^\infty (\Delta v)^2 \, 2\pi m n v b \, db$$

$$= \int_0^\infty \frac{4 G^2 m^2}{b^2 v^2} \pi m n v b \, db = \int_0^\infty \frac{4\pi G^2 m^3}{b v} n \, db$$

Integrating from 0 to ∞ is physical nonsense. Instead, we adapt integration limits b_{min} and b_{max}, giving us

$$\frac{dE_{kin}}{dt} = \frac{4\pi G^2 m^3 n}{v} \ln\left(\frac{b_{max}}{b_{min}}\right)$$

For b_{max}, we take the system size R.
For b_{min}, we take the impact parameter necessary for a 90° scatter.
After long enough time, we may assume that all particles with $b < b_{90}$ have been scattered away and don't contribute any further.
With $\frac{1}{2} m v^2 = \frac{G m^2}{b_{90}} \rightarrow b_{90} = b_{min} = \frac{2Gm}{v^2} = \frac{2Gm}{\frac{GmN}{R}} = \frac{2R}{N}$

$$\Rightarrow \frac{dE_{kin}}{dt} = \frac{4\pi G^2 m^3 n}{v} \ln(N/2)$$

This gives us:

$$\frac{1}{t_{relax}} = \frac{1}{E_{kin}} \frac{dE_{kin}}{dt} = \frac{1}{\frac{1}{2} m v^2} \frac{4\pi G^2 m^3 n}{v} \ln(N/2)$$

$$= \frac{8\pi G^2 m^2 n}{v^3} \ln(N/2) = \underbrace{\frac{1}{t_{cross}} \cdot \frac{R}{v}}_{=1} \cdot \frac{8\pi G^2 m^2 n}{v^3} \ln(N/2)$$

$$= \frac{1}{t_{cross}} \frac{8\pi G^2 m^2 R}{v^4} n \ln(N/2) = \frac{1}{t_{cross}} \frac{8\pi G^2 m^2 R}{M^2 G^2 / R^2} n \ln(N/2)$$

$$= \frac{1}{t_{cross}} 8\pi \cdot \underbrace{\frac{m^2}{M^2}}_{\frac{M}{m} \approx N.} \cdot \underbrace{n R^3}_{nR^3 \approx N} \ln(N/2) = \frac{1}{t_{cross}} \frac{8\pi}{N} \ln(N/2)$$

$$\Rightarrow \boxed{t_{relax} \sim \frac{N\, t_{cross}}{8\pi \ln(N/2)}}$$

A system can be viewed as collisionless if $t_{relax} \gg t_{age}$, where t_{age} is the time of interest. t_{cross} doesn't depend on the particle number N, but only on the system size.

$\Rightarrow$ The primary requirement for the collisionless system is a large N.

Examples:

- Globular star clusters:

  $N \sim 10^5$, $t_{cross} \sim 0.5\,\mathrm{Myr} \sim \frac{1}{H_0} \cdot 10^{-4}$

  These systems are strongly affected by collisions over the age of the Universe.

- Galaxy

  $N \sim 10^{11}$, $t_{cross} \sim 1/100\, \frac{1}{H_0}$

  Large galaxies are collisionless over the age of the Universe to good approximation.

- Dark matter

  Assuming WIMP with $m \sim 100\,\mathrm{GeV}$: $N \sim 10^{77}$, $t_{cross} \sim \frac{1}{10} \frac{1}{H_0}$

  Mother of all collisionless systems

2.3 Weakly Correlated and Strongly Correlated Systems

What if the system is weakly correlated, i.e. particle-particle interactions can't be neglected anymore?

Then
$$f(\vec{r}, \vec{v}, \vec{r}', \vec{v}'; t) = f_1'(\vec{r}, \vec{v}, t) f_2'(\vec{r}', \vec{v}'; t) + g(\vec{r}, \vec{v}, \vec{r}', \vec{v}'; t)$$

The Vlasov equation then becomes the generalized Boltzmann equation:
$$\frac{df}{dt} = -\int \frac{\vec{F}}{m} \frac{\partial}{\partial \vec{v}} g \, d\vec{x}' d\vec{v}' = \left(\frac{\partial f}{\partial t}\right)_c$$

If the interactions are happening on very short timescales or the wavelength associated with them is much smaller than the size of the system, then one can move to a moment-based approximation to make the calculation computationally feasible. This is the <u>fluid approximation:</u> The small interaction wavelength leads to establish and restore a thermal equilibrium rapidly. The Boltzmann equation is then satisfied by a Maxwellian distribution.

The fluid equations are obtained by taking moments of the Boltzmann equation. In its simplest form, one neglects viscosity, thermal conduction and shear stresses (ideal fluid) and only considers isotropic pressure (perfect fluid).

Most astrophysical systems that cannot be described by the collisionless or (weakly) collisional approximation can still be well treated as ideal perfect fluids.

2.4 N-body models and gravitational softening

For collisionless systems, rather than solving the Vlasov-Poisson system in the 'mean field' regime, we forcefully recover the particle nature of the system by discretising using superparticles, provided we have a high enough number of superparticles: the relaxation timescale in the discretized representation must still allow the system to be collisionless!

The equations of motion for superparticles are:

$$\ddot{\vec{x}}_i = -\nabla_i \phi(\vec{r}_i)$$

$$\phi(\vec{r}) = -G \sum_{j=1}^{N} \frac{m_j}{[(\vec{r}-\vec{r}_j) + \varepsilon^2]^{1/2}}$$

Provided there are enough particles to describe the gravitational potential accurately, the orbits of superparticles will be just as valid as the orbits of the real physical particles.

The N-body model gives only one, albeit quite noisy, representation of the one-point function; It doesn't ensemble-average directly. This would require multiple simulations.

The equations of motion contain a softening length ε. The purpose of the force softening is:

- computational efficiency
- avoid large angle scatterings
- avoid expense to calculate orbits in singular potential
- prevent possibility of the formation of bound particle pairs (that would violate collisionless behaviour)

Condition to avoid bound pairs:

$$\langle v^2 \rangle \gg \frac{Gm}{\varepsilon}$$

2.5. N-body Equations in Cosmology

In cosmological simulations, it is customary to use comoving coordinates $\vec{x}$ instead of physical coordinates $\vec{r}$. They are related by $\vec{r} = a(t)\,\vec{x}$ with $a = \frac{1}{1+z}$

The evolution of a is governed by

$$\frac{\dot{a}}{a} = H(a) = \left[\Omega_0 a^{-3} + (1-\Omega_0-\Omega_\Lambda)\,a^{2} + \Omega_\Lambda\right]^{1/2}$$

The infinite expanding space can be modelled through period replication of a box of size L. The Newtonian equation of motions can be written as

$$\frac{d}{dt}\left(a^2 \dot{\vec{x}}\right) = -\frac{1}{a}\,\nabla_i\,\phi(\vec{x}_i)$$

$$\nabla^2 \phi(\vec{x}) = 4\pi G \sum_i m_i \left[-\frac{1}{L^3} + \sum_{\vec{n}} \delta(\vec{x}-\vec{x}_i-\vec{n}L)\right]$$

where the sum over i extends over N particles in the box. The sum over all particles extends also over all their period images (duplicates) with $\vec{n} = (n_1, n_2, n_3)$ being a triplet of integers. The term $-1/L^3$ is needed to ensure that the mean density sourcing the poisson equation vanishes; Otherwise, there would be no solution for an infinite space.

3. Calculating the dynamics of an N-body System

The calculation of the dynamics of an N-body system consists of two steps:

1) Compute the right-hand side of the equation of motion, i.e. compute the gravitational forces
2) Integrate the equation of motion to obtain updated velocities and positions of particles.

In this chapter, we will first look at the first step.
It seems straightforward at first to directly compute the force:

$$\ddot{\vec{r}}_i = -G \sum_{j=1}^{N} \frac{m_j}{[(\vec{r}_i - \vec{r}_j)^2 + \varepsilon^2]^{3/2}} (\vec{r}_i - \vec{r}_j)$$

But for a system of N particles, this means that for each of N particles we have to compute N sums, which is as expensive as $\mathcal{O}(N^2)$. This quickly becomes prohibitive for large N, and causes conflict with our need for large N.

We therefore need faster, approximative force calculation schemes. These usually reduce to direct summation at short distances, but "smooth out" the force using some coarser representation of the density field or potential at large enough distance.

In practice, the choice is dictated by a compromise between accuracy of the computation and the efficiency. The choice will be application dependent.

Note that accuracy shouldn't be pushed too hard, since there are inevitable truncation errors (due to discretisation errors and a finite integration time step) and round-off errors (truncation in floating point number representation).

Among the methods are:
1) Particle Mesh (PM)
2) Fourier-transform based Poisson solvers
3) Iterative Poisson solves on a grid (multigri
4) Tree based methods (hierarchical multipole meth
5) Hybrids, such as TreePM or P^3M

3.1 Particle Mesh Technique

The particle mesh technique relies on the use of an auxiliary mesh. The technique involves four steps:

1) Construction of a density field ρ on a suitable mesh
2) Computation of the potential on the mesh by solving the Poisson equation using ρ
3) Calculation of the force field from the potential on the mesh
4) Calculation of the forces at the original particle position (interpolation step)

Mass assignment

We want to put N particles onto a mesh with uniform spacing $h = L/N_g$, when L is the box size and N_g the number of grid cells points per dimension.

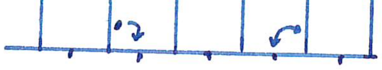
We associate/give each particle a normalized shape to "spread" the mass of particles across cells. To each mesh cell then the fraction of the particle mass is assigned in proportion to the overlap of the particle shape and the mesh cell.

Different choices of the shape function lead to different mappings between the particle distribution and density of the mesh.

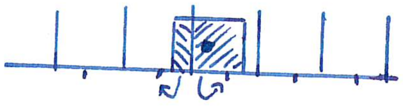
A bigger "spread" of particles over more cells gives smoother density functions as measured by the continuity of first and second derivatives.

Common choices for the shape function:

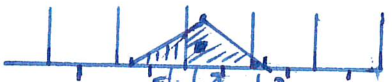
- Nearest grid point assignment
  $S(\vec{x}) = \delta(\vec{x})$; noisy and discontinuous



- Clouds-in-cell assignment
  $S(\vec{x}) = \frac{1}{h^3} \Pi\left(\frac{\vec{x}}{h}\right)$
  $\Pi(x) = \begin{cases} 1 & \text{for } |x| \le 1/2 \\ 0 & \text{else} \end{cases}$
  which is the same cubical 'cloud' shape as that of a mesh cell



- Triangular shaped Cloud
  TSC is smoother than CIC, and the first derivative is continuous (which is not the



Solving the Poisson Equation

With the found density field ρ, we now solve the Poisson equation

$$\nabla^2 \Phi = 4\pi G \rho$$

There are primarily two main methods in use to solve this:

1) Fourier-transform based methods
2) Iterative methods

We will look at these methods in detail later.

Calculation of the forces

Assuming we obtained the potential Φ, we now can calculate the acceleration field from $\vec{a} = -\nabla \Phi$

This can be achieved by calculating a numerical derivative of the potential by finite differencing:

$$a_x^{(i,j,k)} = -\frac{1}{2h}\left(\Phi^{(i+1,j,k)} - \Phi^{(i-1,j,k)}\right)$$

The truncation error of this expression is $\mathcal{O}(h^2)$, hence the estimate of the derivative is second-order accurate.

For a more accurate determination, one can account for more distant cells in the finite difference estimate of Φ.

For example:

$$a_x^{(i,j,k)} = -\frac{1}{2h}\left\{\frac{4}{3}\left[\Phi^{(i+1,j,k)} - \Phi^{(i-1,j,k)}\right] - \frac{1}{6}\left[\Phi^{(i+2,j,k)} - \Phi^{(i-2,j,k)}\right]\right\}$$

has a truncation error of $\mathcal{O}(h^4)$.

The same procedure can be used for y and z direction by varying the j, k.

The choice for a second or fourth order is dictated by the compromise between accuracy and computation speed.

Interpolating from the mesh to the particles

We now have the force field on the mesh, but not on the particle coordinates yet. We need to interpolate the forces from the mesh to the particle coordinate.

It is very important to use the same assignment kernel as used in the density construction also for the force interpolation. This requirement results from the desire to have vanishing self-force, as well as pairwise antisymmetric forces between every particle pair.

3.2 Fourier Techniques

Fourier techniques are powerful tool to solve PDEs whose solutions can be expressed as convolutions of more than one function in real space. Then one maps to k-space and uses mathematical properties of the Fourier transform.

For non-periodic space, the potential is given by

$$\phi(\vec{x}) = -\int G \frac{\rho(x')\,dx'}{|\vec{x}-\vec{x}'|} = \int g(\vec{x}-\vec{x}')\,\rho(\vec{x}')\,d\vec{x}'$$

with $g(\vec{x}) = \frac{-G}{|\vec{x}|}$ the Green's function of Newtonian gravity.

We may also write it as a convolution:

$$\phi = g * \rho$$

Using the convolution theorem for Fourier transforms:

$$\mathcal{F}[f*g] = \mathcal{F}[f]\cdot\mathcal{F}[g]$$, a simple multiplication.

We can exploit this:

$$\phi = \mathcal{F}^{-1}[\mathcal{F}[g]\cdot\mathcal{F}[\rho]]$$

or

$$\hat{\phi} = \mathcal{F}[\phi] = \mathcal{F}[g]\cdot\mathcal{F}[\rho] = \hat{g}\,\hat{\rho}$$

In practice, consider a box of size L and periodic boundary conditions.
A continuous (not discretized in a mesh) density field ρ can be written as a Fourier series of the form:

$$\rho(\vec{x}) = \sum_{\vec{k}} \rho_{\vec{k}}\, e^{i\vec{k}\vec{x}} \qquad \text{with} \qquad \rho_{\vec{k}} = \frac{1}{L^3}\int_V \rho(\vec{x})\, e^{-i\vec{k}\vec{x}}\, d\vec{x}$$

the integration for $\rho_{\vec{k}}$ is over one instance of the periodic box.
The sum over $\vec{k}$-vectors extends over a discrete spectrum of wave vectors with

$$\vec{k} \in \frac{2\pi}{L}\begin{pmatrix} n_1 \\ n_2 \\ n_3 \end{pmatrix} \qquad \text{and } n_1, n_2, n_3 \text{ being integers.}$$

The allowed modes in $\vec{k}$ hence form an infinitely extended Cartesian grid. (mode: allowed values for $\vec{k}$). Reality constraints imply that $\rho_{\vec{k}} = \rho^*_{-\vec{k}}$, hence the modes are not all independent.

The Fourier coefficients $\rho_{\vec{k}}$ features the following orthogonality and closure relationships:

$$\frac{1}{L^3}\int d\vec{x}\, e^{i(\vec{k}-\vec{k}')} = \delta_{\vec{k}\vec{k}'}$$

$$\frac{1}{L^3}\sum_{\vec{k}} e^{i\vec{k}\vec{x}} = \delta(\vec{x})$$

Now back to the Poisson equation:

$$\nabla^2 \Phi = 4\pi G \rho$$

$$\hookrightarrow \nabla^2\left(\sum_{\vec{k}} \Phi_{\vec{k}}\, e^{i\vec{k}\vec{x}}\right) = 4\pi G\left(\sum_{\vec{k}} \rho_{\vec{k}}\, e^{i\vec{k}\vec{x}}\right)$$

$$= \sum_{\vec{k}} \left(-\vec{k}^2 \phi_{\vec{k}}\right) e^{i\vec{k}\vec{x}}$$

The equation must hold for every mode separately, therefore:

$$\phi_{\vec{k}} = -\frac{4\pi G}{\vec{k}^2}\rho_{\vec{k}}$$

Comparing with $\vec{\Phi} = \vec{g}\cdot\vec{\rho}$, we see that

$$g_{\vec{k}} = -\frac{4\pi G}{\vec{k}^2}$$

The Discrete Fourier Transform (DFT)

On the mesh, ρ is not continuous, but is rather sampled by a discrete set of points. In this case, we have to consider discrete series running on the set of N mesh points rather than integrals to represent the function in k-space.

Assuming we have N equally spaced points per dimension, the $\vec{x}$ positions of the mesh points are

$$\vec{x}_p = \frac{L}{N}\begin{pmatrix} p_1 \\ p_2 \\ p_3 \end{pmatrix} \quad \text{where } p_1, p_2, p_3 \in \{0, 1, \dots, N-1\}$$

We replace $d^3\vec{x}$ with $(L/N)^3$ to get a discrete sum:

$$\rho_{\vec{k}} = \frac{1}{N^3} \sum_{\vec{p}} \rho_{\vec{p}}\, e^{-i\vec{k}\vec{x}_p}$$

The series is finite, with N integers for every dimension. Shifting $\vec{k}$ in any of the dimensions by N times the fundamental mode $2\pi/L$ gives again the same result because of the periodicity and the finite numbers of mesh points. We can thus set as reference set of modes $\vec{k}_l$ such that

$$\vec{k}_l = \frac{2\pi}{L}\begin{pmatrix} l_1 \\ l_2 \\ l_3 \end{pmatrix}, \quad l_1, l_2, l_3 \in \{0, 1, \dots, N-1\}$$

The construction of ρ through the Fourier series becomes a finite sum over these N^3 modes:

$$\hat{\rho}_{\vec{l}} = \frac{1}{N^3} \sum_{\vec{p}} e^{-i\frac{2\pi}{N}\vec{l}\vec{p}}$$

$$\rho_{\vec{p}} = \sum_{\vec{l}} \hat{\rho}_l \cdot e^{i\frac{2\pi}{N}\vec{l}\vec{p}}$$

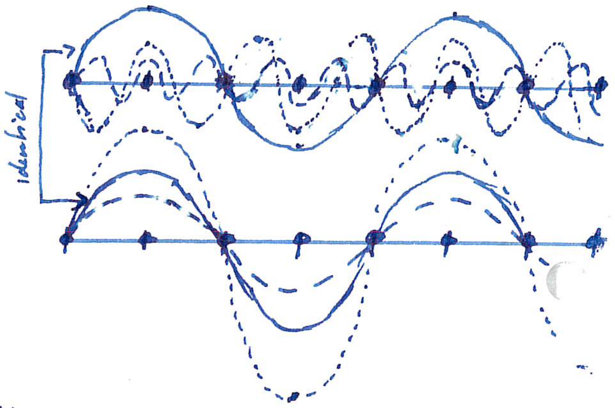
[ρ at mesh point $\vec{p}$ in real space]

The two transformations are an invertible linear mapping of a set of N^3 complex values $\rho_{\vec{p}}$ to N^3 complex values $\hat{\rho}_{\vec{l}}$ and vice versa.

Conventionally one often uses the set $l \in \{-N/2, \dots, 0, \dots, N/2 - 1\}$ instead of $\{0, \dots, N-1\}$. The occurance of both positive and negative frequencies is made more explicit, and they are arranged quasi-symmetrically in a box in $\vec{k}$-space centred around $\vec{k} = (0,0,0)$. The box extends out to $k_{max} = \frac{N}{2}\frac{2\pi}{L}$, which is the so-called <u>Nyquist-frequency</u>.

Adding waves beyond the Nyquist frequency in a reconstruction of S on a given grid would introduce redundant information that could not be unambiguously recovered from the discretized density field. Instead, the power in these waves would be erroneously mapped to lower frequencies (= aliasing).

The sampling theorem states that, in order to represent a function with characteristic frequency fc the sampling rate has to be at least double of such frequency. In practice, this means that the mesh has to have a resolution of at least half the wavelength of the function that one wants to represent (= Nyquist frequency).



Otherwise, we get aliasing problems: The effect of unsampled waves present in the DFT that represents the functions, whose amplitudes are transferred to lower resolved frequencies and compromise the result of the DFT itself.

If mesh points correspond to maxima, minima or zeros for both low-order (solid line) and higher frequency aliases (dotted and dashed line in upper plot), then amplitudes are increased (dotted line in lower plot). Otherwise, the amplitudes are decreased (dashed line in lower plot).

Fast Fourier Transform

Computing the DFT requires to compute N terms in the Fourier series (and sum them up) for each of the numbers $\hat{s}_k$
$\Rightarrow$ The computations go as $O(N^2)$.

Fortunately, there is a method to reduce the computational cost: The fast Fourier Transform method. One splits the individual sums recursively so that one has to compute less terms for each of the N numbers. If the calculation can be expressed as power in base 2, the computation reduces to $O(N \log_2 N)$
Furthermore, the FFT algorithm also reduces the numerical floating point error (round-off) that would otherwise be incurred.

Let $\quad \mathfrak{F}[f] = \hat{f}(n) = \sum_{k=0}^{N-1} f[k] e^{-i\frac{2\pi}{N}nk} \equiv \sum_{k=0}^{N-1} f[k] W_N^{nk}$

The same values of W_N^{nk} are calculated many times as the computation proceeds:

- the integer product nk repeats for different combinations of n and k
- W_N^{nk} is a periodic function with only N distinct values

Now let's split the single summation over N samples into 2 summations, each with $N/2$ samples; One for even k and one for odd k.

$$\Rightarrow \hat{f}(n) = \sum_{k=0}^{N-1} f[k] e^{-i\frac{2\pi}{N}nk} = \sum_{k=0}^{N-1} f[k] W_N^{nk} =$$

$$= \sum_{m=0}^{N/2-1} f[2m] W_N^{2mn} + \sum_{m=0}^{N/2-1} f[2m+1] W_N^{(2m+1)n}$$

Using $W_N^{2mn} = e^{-i\frac{2\pi}{N}(2mn)} = e^{-i\frac{2\pi}{N/2}mn} = W_{N/2}^{mn}$

$$\sum_m f[2m+1] W_N^{(2m+1)n} = \sum_m f[2m+1] W_N^{2mn} W_N^n = W_N^n \sum_m f[2m+1] W_{N/2}^{mn}$$

$$\Rightarrow \hat{f}(n) = \sum_{m=0}^{N/2-1} f[2m] W_{N/2}^{mn} + W_N^n \sum_{m=0}^{N/2-1} f[2m+1] W_{N/2}^{mn}$$

$$\equiv G[n] + W_N^n H[n]$$

Also consider:

$$W_N^{n+N/2} = e^{-i\frac{2\pi}{N}(n+N/2)} = e^{-i\frac{2\pi}{N}\frac{N}{2}} e^{-i\frac{2\pi}{N}n} = e^{-i\pi} e^{-i\frac{2\pi}{N}n} = -e^{-i\frac{2\pi}{N}n} = -W_N^n$$

$$W_{N/2}^{n+N/2} = e^{-i\frac{2\pi}{N/2}(n+N/2)} = e^{-i\frac{2\pi}{N/2}n} e^{-i2\pi} = e^{-i\frac{2\pi}{N/2}} = W_{N/2}^n$$

Then for example for $N=8$:

$\hat{f}[0] = G[0] + W_8^0 H[0]$ $\quad\quad$ $\hat{f}[4] = G[4] + W_8^4 H[4] = G[0] - W_8^0 H[0]$

$\hat{f}[1] = G[1] + W_8^1 H[1]$ $\quad\quad$ $\hat{f}[5] = G[5] + W_8^5 H[4] = G[1] - W_8^1 H[1]$

$\hat{f}[2] = G[2] + W_8^2 H[2]$ $\quad\quad$ $\hat{f}[6] = G[6] + W_8^6 H[6] = G[2] - W_8^2 H[2]$

$\hat{f}[3] = G[3] + W_8^3 H[3]$ $\quad\quad$ $\hat{f}[7] = G[7] + W_8^7 H[7] = G[3] - W_8^3 H[3]$

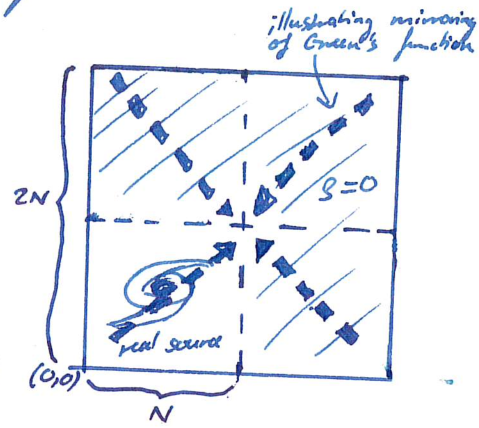
Overall, only 4 terms needed to be computed.
Assuming N is a power of 2, we can repeat the process on the two $N/2$-point transforms (G and H), breaking them down to $N/4$-points transforms etc until we come down to 2-point transforms.

If N is not a power of 2, one can use alternate symmetries or add zeros to adjust the number of terms to a power of 2.

FFT method for non-periodic problems

The DFT is intrinsically periodic. Can we use these techniques for non-periodic problems? This is possible with the zero-padding trick.

We need to obtain a periodic potential function despite the non-periodicity of the density, exploiting the fact that the potential is a convolution of the density and the Green's function:



- Arrange our mesh such that the source distribution lives only in one quarter of the mesh, let's say the bottom left, the rest of the density field needs to be zeroed out.

- Set up real-space Green's function $g_{ij} = \frac{-G}{|\vec{x}_i + \vec{x}_j|}$, $i,j \le N$ (~~the~~ response of a mass at origin). The Green's function for the whole mesh then is set up as $g_{ij} = g_{2N-i,j} = g_{i,2N-j}$ $= g_{2N-i,\,2N-j}$ i.e. duplicated and mirrored.

- Carry out real-space convolution $\phi = g * \rho$ by using the definition of the discrete, periodic convolution:

  $$\phi_{\vec{p}} = \sum_{\vec{n}} g_{\vec{p}-\vec{n}}\, \rho_{\vec{n}}$$; $\vec{p}$ = gridpoint, $\vec{n}$ gridpoint to sum over; both vectors of integers $< 2N$

  both g and ρ are treated as periodic fields.
  This gives the correct solution on the domain of the real source (as $\rho=0$ elsewhere). The zero-padded region is big enough to make sure the "fake periodicity" is not changing the value in the real domain.

- Since the convolution can be expressed as a periodic one this way, we may use FFT methods again to solve the convolution quickly, thus obtaining ϕ and being able to calculate the forces later.

- Downside: Enlarged cost of CPU and memory usage by factor 2^{ndim}

3.3 Iterative and Multigrid Methods

Central idea: turn non-linear PDE Poisson equation into a set of linear equation that can be solved on the mesh using methods from calculus. Consider the 1D problem:

$$\frac{\partial^2 \phi}{\partial x^2} = 4\pi G \rho(x)$$

with $\left(\frac{\partial^2 \phi}{\partial x^2}\right)_i \simeq \frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{h^2} \quad \left[+ \mathcal{O}(h^2)\right]$

This gives us N equations for N ϕ_i-s:

$$\frac{\phi_{i+1} - 2\phi_i + \phi_{i-1}}{h^2} = 4\pi G \rho_i$$

In principle, we should be able to solve this algebraically. The system can be rewritten as

$$A\vec{x} = \vec{b} \quad \text{with} \quad \vec{x} = (\phi_i), \quad \vec{b} = h^2 4\pi G \rho$$

$$A = \begin{pmatrix} -2 & 1 & & & 1 \\ 1 & -2 & 1 & & \\ & 1 & -2 & 1 & \\ & & & \vdots & \\ 1 & & & 1 & -2 \end{pmatrix}$$

Solving $A\vec{x} = \vec{b}$ directly constitutes a matrix inversion that can in principle be carried out, but is $\mathcal{O}(N^3)$.

To reduce the cost, we instead turn to approximate iterative methods. The main idea is to speed up the time by splitting up the matrix A into smaller matrices and invert the result to get the solution for $x^n = (\phi_i^n)$ using repeated iterations after an initial "guess" until convergence is reached.

Convergence is achieved until either the

- absolute error $\max_i |x_i^{(n)} - x_i^{(n-1)}| < \delta$

or

or

- relative error $\max_i |x_i^{(n)} - x_i^{(n-1)}| < \max_i |x_i^{(n)}| \, \varepsilon$

for given $\delta, \varepsilon > 0$.

Jacobi - Iteration

Decompose the matrix A as $A = D - (L+U)$

D: diagonal part

L: Lower diagonal part } negative

U: Upper diagonal part

$\Rightarrow \quad [D - (L+U)]\vec{x} = \vec{b}$

$\Rightarrow \quad D\vec{x} - (L+U)\vec{x} = \vec{b}.$

$\Rightarrow \quad \vec{x} = D^{-1}\vec{b} + D^{-1}(L+U)\vec{x}$

We use this to define an iterative sequence of vectors $\vec{x}^{n}$:

$$\vec{x}^{(n+1)} = D^{-1}\vec{b} + D^{-1}(L+U)\vec{x}^{(n)} \qquad \text{(Jacobi iteration)}$$

D^{-1} is trivially obtained because it is diagonal: $D_{ii}^{-1} = 1/A_{ii}$

The scheme converges if and only if the so-called convergence matrix $M = D^{-1}(L+U)$ has only eigenvalues < 1 $\Leftrightarrow$ $\rho_S(M) \equiv \max_i |\lambda_i| < 1$ [ρ_S: Spectral radius]

This condition can be derived by considering the error vector of the iteration:

$$\vec{e}^{(n)} \equiv \vec{x}_{exact} - \vec{x}^{(n)}$$

Then $\vec{e}^{(n+1)} = \vec{x}_{exact} - \vec{x}^{(n+1)} = \vec{x}_{exact} - D^{-1}\vec{b} - D^{-1}(L+U)x^{(n)}$

$= M\vec{x}_{exact} - M\vec{x}^{(n)} = M\vec{e}^{(n)}$

where I used $\vec{x} = D^{-1}\vec{b} + D^{-1}(L+U)\vec{x}$

$\Rightarrow \vec{x}_{exact} - D^{-1}\vec{b} = D^{-1}(L+U)\vec{x}_{exact} = M\vec{x}_{exact}$

We find: $\vec{e}^{(n)} = M^{n}\vec{e}^{(0)}$

This implies $|\vec{e}^{(n)}| \leq [\rho_S(M)^{n}]^{n}|\vec{e}^{(0)}|$ and hence convergence if the spectral radius < 1.

Gauss-Seidel Iteration Method

For any iteration step of the Jacobi iteration, the entire step is completed for all cells in the mesh (all mesh points) and then the next iteration step is performed.

The central idea of Gauss-Seidel iteration is to use the updated values as soon as they become available for computing further updated values, rather than first computing the iteration step for all mesh points.

As before, we can write:

$$A = D - (L + U) \; ; \qquad A\vec{x} = \vec{b}$$

$$\hookrightarrow \; (D-L)\,\vec{x} = U\vec{x} + \vec{b}$$

$$\Rightarrow \; \vec{x} = (D-L)^{-1}U\vec{x} + (D-L)^{-1}\vec{b}$$

suggesting the iteration rule

$$\vec{x}^{(n+1)} = (D-L)^{-1}Ux^{(n)} + (D-L)^{-1}\vec{b}$$

modify the equation:

$$(D-L)\,\vec{x}^{(n+1)} = U\vec{x}^{(n)} + \vec{b}$$

$$\Rightarrow \; D\vec{x}^{(n+1)} = U\vec{x}^{(n)} + L\vec{x}^{(n+1)} + \vec{b}$$

$$\Rightarrow \; \vec{x}^{(n+1)} = D^{-1}U\vec{x}^{n} + D^{-1}L\vec{x}^{(n+1)} + D^{-1}\vec{b}$$

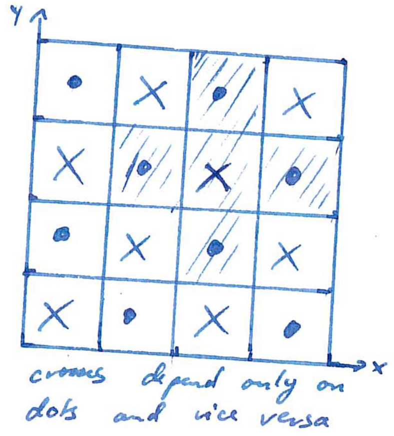
If we start computing the new elements in the first (highest) row $i=1$ of this matrix equation, we see that no values of $\vec{x}^{(n+1)}$ are actually needed, because L has only elements below the diagonal. Proceeding with further rows, only values of $\vec{x}^{(n+1)}$ given by the rows above will be needed.

A problematic point of Gauss-Seidel is that the equations need to be solved in a specific order, meaning that this part can't be parallelized. Also, the result will in general depend on which element is selected to be first.

Red Black Ordering

To overcome the non-parallisability and dependence of the choice of the first element of Gauss-Seidel iteration, one can sometimes use so-called red-black ordering, which effectively is a compromise between Jacobi and Gauss-Seidel.

Certain update rules, such as that for the Poisson equation, allow a decomposition of the cells into disjoint sets whose update rules depend only on cells from other sets. For example for the Poisson equation, this is the case for a chess board like pattern of 'red' (crosses) and 'black' (dots) cells.



One can then first update all the black points (dot points), followed by an update of all the crosses, which themselves only rely on dots. In the second of this half-update one can already use the updated values from the first half-step, making it intuitively clear that such a scheme can almost double the convergence rate compared to Jacobi.

The Multigrid Technique

Iterative solvers like Jacobi or Gauss-Seidel often converge quite slowly. One also observes that high-frequency errors in the solution are damped out quickly by the iteration but long wavelength errors die out much more slowly. This is intuitively not unexpected, as the information only "travels" by one cell every iteration. For convergence, it needs to propagate back and forth over the whole domain multiple times.

An idea to fix this is to go to a coarser mesh and try to compute an improved initial guess, which may help speed up the convergence on the fine grid. On a coarser mesh, we have 2^{ndim} less cells, the relaxation will be cheaper.

First we need to figure out how to map from a coarse to a finer grid and vice versa.

Coarse - to - fine: Prolongation

Going from a mesh of grid size $2h$ to a mesh of size h

Define prolongation operator:

$$I_{2h}^{h} \vec{x}^{(2h)} = \vec{x}^{(h)}$$

Simple realisation in 2D:

$$I_{2h}^{h} = \begin{cases} x_{2i}^{(h)} = x_i^{(2h)} \\ x_{2i+1}^{(h)} = \frac{1}{2}\left(x_i^{(2h)} + x_{i+1}^{(2h)}\right) \end{cases}$$ extrapolate midpoint

for $0 \leq i < \frac{N}{2}$

Fine - to - coarse: <u>Restriction</u>

Going from (h)-sized mesh to coarser 2h-size mesh

Define restriction operator:

$$I_h^{2h} \vec{x}^{(h)} = \vec{x}^{(2h)}$$

Simple realisation in 2D:

$$I_h^{2h} : \quad x_i^{(2h)} = \frac{x_{2i-1}^{(h)} + 2x_{2i}^{(h)} + x_{2i+1}^{(h)}}{4}$$

with $0 \leq i < N/2$

Example: Assume $h = 1$, such that $x_{i+1}^{(h)} - x_i^{(h)} = h = 1$

$\Rightarrow x_i^{(h)} = i \cdot h$

Then: $x_1^{(2h)} = \dfrac{x_1^{(h)} + 2x_2^{(h)} + x_3^{(h)}}{4} = \dfrac{1+4+3}{4} = 2$

$x_2^{(2h)} = \dfrac{x_3 + 2x_4 + x_5}{4} = \dfrac{3+8+5}{4} = 4$

Remember: x_i are supposed to give us mesh points only, not other values! Here we have exactly as expected:

$$x_{i+1}^{(h)} - x_i^{(h)} = h = 1$$

$$x_{i+1}^{(2h)} - x_i^{(2h)} = 2h = 2$$

Now that we have defined the prolongation and restriction operators, we can return to the computation of the potential. The general idea is to use the fine grid to "correct" the solution obtained on the coarse grid, where it was computed faster.

We define the error vector:

$$\vec{e} \equiv \vec{x}_{exact} - \tilde{\vec{x}}$$, where $\vec{x}_{exact}$ is the exact solution and $\tilde{\vec{x}}$ the (current) approximate solution.

We also define the residual:

$$\vec{r} = \vec{b} - A\tilde{\vec{x}}$$ (Remember, we solve $A\vec{x} = \vec{b}$)

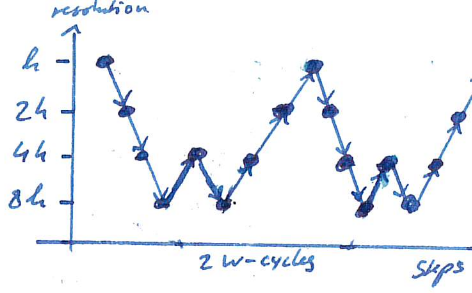
with $A\vec{x}_{exact} = \vec{b}$, we have $A(\vec{x}_{exact} - \tilde{\vec{x}}) = \vec{b} - A\tilde{\vec{x}}$

$$\Rightarrow A\vec{e} = \vec{r}$$

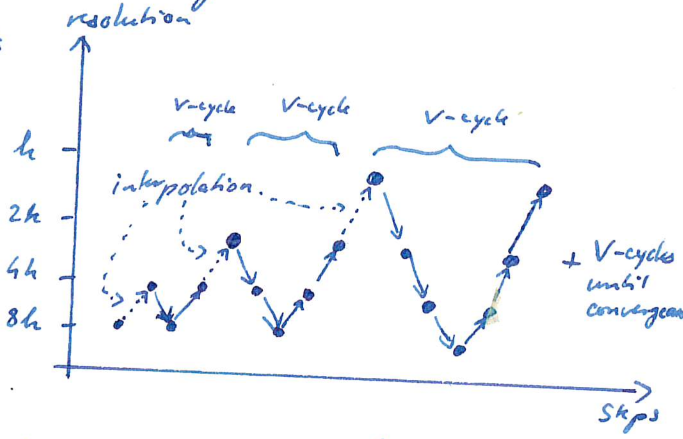
The <u>coarse grid correction scheme</u> is supposed to return an improved solution $\vec{x}^{1(h)}$ at the fine grid level h based on correcting an initial guess $\tilde{\vec{x}}^{(h)}$ by solving the problem on the coarser mesh and interpolating back. The steps are:

1) Carry out one relaxation step on h (for example one Gauss-Seidel or Jacobi iteration)
2) Compute the residual $\vec{r}^{(h)} = \vec{b}^{(h)} - A^{(h)}\tilde{\vec{x}}^{(h)}$
3) Restrict the residual to a coarser mesh: $\vec{r}^{(2h)} = I_h^{2h} r^{(h)}$
4) Solve $A^{(2h)}\vec{e}^{(2h)} = \vec{r}^{(2h)}$ on the coarser mesh with $\tilde{\vec{e}}^{(2h)} = 0$ as initial guess
5) Prolong the obtained error $\vec{e}^{(2h)}$ to the finer mesh, $\vec{e}^{(h)} = I_{2h}^{h}\vec{e}^{(2h)}$, and use it to correct the current solution on the fine grid: $\tilde{\vec{x}}^{1(h)} = \tilde{\vec{x}}^{(h)} + e^{(h)}$
6) Repeat until converged

In the V-Cycle, one performs one iteration per level (step 4). One could perform 2 or more, but more than 2 is typically too expensive. This method is called the "W-cycle".



The V-Cycle scheme discussed thus far relies on an initial guess for the solution. A bad guess will require more V-cycles to reach satisfactory convergence. The full multi-grid method solves this problem of how to initiate the first step naturally: Get a good guess by solving the problem on a coarser grid first, and then interpolate the coarse grid solution to the fine grid as a starting guess. This procedure can be applied recursively as well. One can define a sequence of cycles (restrict, ~~interpolate~~, prolong). The algorithm is the following:



1) initialize the r.h.s. on all grid levels ($\vec{b}^h, \vec{b}^{2h}, \dots, \vec{b}^H$) up to some coarsest level H.

2) Solve the problem exactly on the coarsest level H.

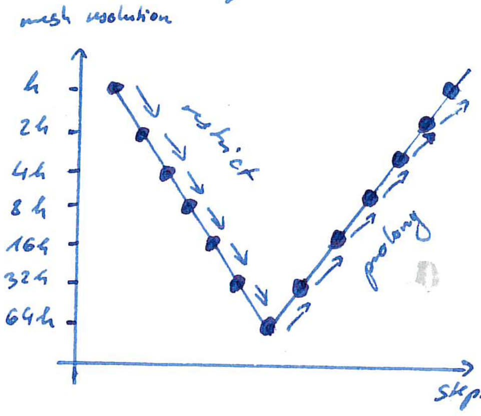
3) Given a solution on level i with spacing $2h$, map it to the next level $i+1$ with spacing h and obtain the initial guess $\hat{\vec{x}}^{(h)} = I_{2h}^{h} \vec{x}^{(2h)}$

4) Use this starting guess to solve the problem on the level $i+1$ with one V-Cycle.

5) Repeat step 3 until the finest level is reached.

The solution for the equation $A\vec{e} = \vec{r}$ in step 4 can be found recursively by keeping going to increasingly coarser grids until we reach a minimum number of grids. That threshold is decided based on how accurate we want to be, often driven by empirical considerations.

At this "maximally coarse grid" one solves the problem exactly or using an iterative method and skips steps 2-5.



This recursive method is called the <u>V-Cycle.</u> The algorithm reduces cost to $\mathcal{O}(N_g \log N_g)$ to reach machine truncation/round off error precision; Each V-Cycle has a cost $\mathcal{O}(N)$, but multiple cycles are needed for convergence. For the Poisson equation, this is the same cost scaling as one gets with FFT-based methods. An interesting advantage of multigrid is that it requires less data communication when parallelised on distributed memory machines.

One remaining technical point is finding $A^{(2h)}$ on the coarse mesh given $A^{(h)}$ on the fine mesh. There are two generic approaches:

a) Direct coarse grid approximation

Discretise equation to be solved (e.g. Poisson) directly on the coarse grid, scaled by the resolution.

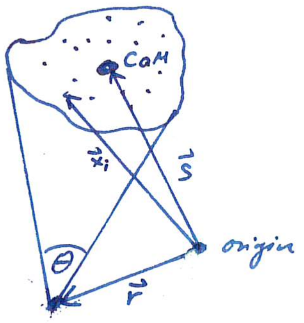
b) Galerkin coarse grid approximation

Define $A^{(2h)} = I_h^{2h} A^{(h)} I_{2h}^{h}$

which is formally the consistent way to define $A^{(2h)}$, but more expensive

3.4 Hierarchical Multipole Methods ("tree codes")

The main idea is to use multipole expansion of a distant group of particles to describe its gravity instead of summing up the forces from all individual particles.



The potential energy of the group is given by

$$\phi(\vec{r}) = -G\sum_i \frac{m_i}{|\vec{r}-\vec{x}_i|} = -G\sum_i \frac{m_i}{|\vec{r}-\vec{s}+\vec{s}-\vec{x}_i|}$$

We now expand the denominator assuming $|\vec{x}_i-\vec{s}| \ll |\vec{r}-\vec{s}|$, which is the case for a sufficiently small opening angle θ.

Let $\vec{y} \equiv \vec{r}-\vec{s}$. Then we can use the Taylor expansion:

$$\frac{1}{|\vec{y}+\vec{s}-\vec{x}_i|} = \frac{1}{|\vec{y}|} - \frac{\vec{y}\cdot(\vec{s}-\vec{x}_i)}{|y|^3} + \frac{1}{2}\frac{\vec{y}^T\left[3(\vec{s}-\vec{x}_i)(\vec{s}-\vec{x}_i)^T - (\vec{s}-\vec{x}_i)^2\right]\vec{y}}{|y|^5} + \dots$$

the expansion could be extended further if desired.

Since we carried out the expansion relative to the centre of mass $\vec{s} = \frac{1}{M}\sum_i m_i x_i$, the dipole term vanishes.

We define:

Monopole: $M = \sum_i m_i$

Quadrupole: $Q_{ij} = \sum_k m_k\left[3(\vec{s}-\vec{x}_k)_i(\vec{s}-\vec{x}_k)_j - \delta_{ij}(\vec{s}-\vec{x}_k)^2\right]$

If we restrict ourselves only up to quadrupole order, we arrive at the expansion

$$\phi(\vec{r}) = -G\left(\frac{M}{|\vec{y}|} + \frac{1}{2}\frac{\vec{y}^T Q \vec{y}}{|\vec{y}|^5}\right), \quad \vec{y} = \vec{r}-\vec{s}$$

We consider the expansion accurate enough if

$$\theta \simeq \frac{\langle |x_i-\vec{s}|\rangle}{|\vec{y}|} \simeq \frac{l}{y} \ll 1 \qquad \text{with } l = \text{radius of the group}$$

Tree algorithms are based on a hierarchical grouping of the particles and for each group, one then pre-computes the multipole moments for later use in approximations of the force due to distant groups. Usually the hierarchy of groups is organized with the help of a tree-like data structure, hence the name "tree algorithm".

Particles are organized in "nodes", namely recursive divisions of the computational volume, until individual particles are reached (buckets). The full domain is called level 0 or root.

There are different strategies for defining the groups. The popular Barnes & Hut algorithm, one starts with a cube that contains all particles. This cube is then subdivided into 8 sub-cubes of half the size in each spatial dimension. One continues this refinement recursively until each subnode contains only a single particle.

An important property of such hierarchical, tree-based groupings is that they are geometrically highly flexible and adjust to any clustering state the particles may have. They are automatically adaptive. Also, there is no significant slow-down when severe clustering starts.

The force calculation with the tree then proceed by <u>walking the tree.</u>* Starting at the root node, one checks for every node whether the opening angle under which it is seen is smaller than a prescribed tolerance angle Θ_c. If this is the case, the multipole expansion of the node can be accepted, and the corresponding partial force is evaluated and added to an accumulation of the total force. The tree walk along this branch of the tree then can be stopped. Otherwise, one must open the tree node and consider all its sub-nodes in turn.

The resulting force is approximate by construction, but the overall size of the error can be conveniently controlled by the tolerance opening angle Θ_c. Smaller Θ_c requires more computation, but is more accurate.

The standard tree-algorithm, the cost is $\mathcal{O}(N \log N)$. A faster algorithm can be obtained if one does a multipole expansion also for the target particles rather than for the sources only, and further exploiting Newton's third law (Fast Multipole Methods) with cost even slightly better than $\mathcal{O}(N)$.

* for each target particle.

3.5. Hybrid Schemes

P^3M

The Force is decomposed in two components:

$$F(r) = F_{PM}(r, h) + F_{PP}(r, \varepsilon)$$

The PP force is a short-range interaction for $r < r_c = 2\text{-}3h$
It is computed with direct N^2 summation. Otherwise: Particle mesh techniques like multigrid or iterative.
The problem with this method is that for a highly clustered configuration, the PP force dominates the CPU time.

AP^3M

Adaptive Particle-Particle, Particle Mesh.
PM is performed on finer grids until relatively small particle separations are reached. The force is decomposed recursively in smaller and smaller scale components and only the coarsest grid used for long range force calculation.
It isn't more accurate than P3M, but faster.

Tree PM

The particle mesh (PM) approach based on Fourier methods is probably the fastest method to calculate the gravitational field on a homogeneous mesh, but the force resolution can't be better than the size of one mesh cell.
Tree PM codes try to combine the hierarchical tree and PM schemes: The gravitational field on large scales is calculated with a PM algorithm, short-range forces by a tree-method.
A clean separation of scales is done in Fourier space:

$$\phi_{\vec{k}} = \phi_{\vec{k}}^{long} + \phi_{\vec{k}}^{short}$$

with
$$\phi_{\vec{k}}^{long} = \phi_{\vec{k}} \exp(-\vec{k}^2 r_s^2)$$
$$\phi_{\vec{k}}^{short} = \phi_{\vec{k}} [1 - \exp(-\vec{k}^2 r_s^2)]$$

r_s is the spatial scale of the force split (exponential cut-off).

Below a few cut-off radii $(\sim 5 r_s)$, a tree calculation is performed. The tree component of the calculation is efficient since it will cover only a limited region of the domain, allowing to consider only a low number of nodes. We transform back to real space:

$$\phi^{short}(\vec{x}) = - G \frac{m}{r} \operatorname{erfc}\left(\frac{r}{2 r_s}\right)$$

where $\vec{x} = \min(|\vec{x} - \vec{r} - \vec{n}L|)$ is defined as the smallest distance of any of the periodic images of the point mass at $\vec{r}$ relative to the point $\vec{x}$. This looks like the newtonian potential modified by a truncation factor that rapidly turns off the force at large distances. In practice, the force is made $\sim 1\%$ of the newtonian value at $r \sim 4.5 r_s$.

One computes the short-range potential with the tree code, namely using the multipole expansion for a whole node.

In addition, this algorithm is ideal for cosmological simulations since it avoids the problem of introducing periodic boundaries in the tree code (Ewald summation). One simply exploits the natural periodicity of the long-range part performed by the PM code using FFT.

4. Time Integration Techniques

We look at some basic methods for the integration of ordinary differential equations (ODE's). These are relations between an unknown scalar or vector-values function $\vec{y}(t)$ and its derivatives with respect to an independent variable t.
Such equations take on the form

$$\frac{d\vec{y}}{dt} = \vec{f}(\vec{y}, t)$$

we seek the solution $\vec{y}(t)$, subject to boundary conditions.

4.1 Explicit Euler Methods

Also called "forward Euler".

$$y_{n+1} = y_n + f(y_n)\,\Delta t$$

Δt is the integration step.

This approach is the simplest of all. It is called "explicit", because y_{n+1} is computed with a right-hand side that only depends on quantities that are already known.

In general, the method is only stable for sufficiently small step size. The stability may be a sensitive function of the step size.

It is recommended to refrain from using this scheme in practice, since there are other methods that offer better accuracy at the same or lower computational cost. The Euler method is only first order accurate:

The truncation error at a single step is of order $O_s(\Delta t^2)$, but we need $N_s = T/\Delta t$ steps, giving $N_s\, O_s(\Delta t^2) = O_T(\Delta t)$

For a method to reach a global error that scales as $O_T(\Delta t^n)$, we need a local truncation error of one order higher: $O_s(\Delta t^{n+1})$

The method is also not time-symmetric, which makes it prone to accumulation of secular integration errors.

4.2 Implicit Euler

Also called "backwards Euler" scheme:

$$y_{n+1} = y_n + f(y_{n+1})\,\Delta t$$

This approach has excellent stability properties, even for large time steps. This makes the implicit Euler useful for stiff equations, where the derivatives (suddenly) can become really large.

This method is still first order accurate and also lacks time-symmetry, just like the explicit Euler scheme.

4.3 Implicit midpoint Euler

$$y_{n+1} = y_n + f\left(\frac{y_n + y_{n+1}}{2}\right)\Delta t$$

The implicit midpoint rule can be viewed as a symmetrized variant of explicit and implicit Euler.

It is second order accurate, but still implicit, thus for difficult to use in practice. It is also time-symmetric, i.e. one can formally integrate backwards and recover exactly the same steps as in a forward integration.

Proof that local truncation is $\mathcal{O}_S(\Delta t^3) \Rightarrow$ global truncation $\mathcal{O}_T(\Delta t^2)$.

Use Taylor expansion:

$$\begin{aligned} y(x) &= y(x_0) + y'(x_0)(x - x_0) + \frac{1}{2} y''(x_0)(x-x_0)^2 + \mathcal{O}((x-x_0)^3) \\ &= y(x_0) + f(x_0)(x-x_0) + \frac{1}{2} f'(x_0)(x-x_0)^2 + \mathcal{O}((x-x_0)^3) \end{aligned}$$

Let x_0 be the midpoint: $x_0 = t + \frac{\Delta t}{2}$, and expand around $x = t + \Delta t$ and $x = t$:

$$y(t+\Delta t) = y(t+\tfrac{\Delta t}{2}) + f(t+\tfrac{\Delta t}{2})\left(\tfrac{\Delta t}{2}\right) + f'(t+\tfrac{\Delta t}{2})\cdot\tfrac{1}{2}\left(\tfrac{\Delta t}{2}\right)^2 + \mathcal{O}(\Delta t^3)$$

$$y(t) = y(t+\tfrac{\Delta t}{2}) - f(t+\tfrac{\Delta t}{2})\,\tfrac{\Delta t}{2} + f'(t+\tfrac{\Delta t}{2})\cdot\tfrac{1}{2}\left(\tfrac{\Delta t}{2}\right)^2 + \mathcal{O}(\Delta t^3)$$

$$\Rightarrow \quad y(t+\Delta t) - y(t) = 2\cdot f(t+\tfrac{\Delta t}{2})\,\tfrac{\Delta t}{2} + \mathcal{O}(\Delta t^3)$$

$$\Rightarrow \quad y(t+\Delta t) = y(t) + f(t+\tfrac{\Delta t}{2})\,\Delta t + \mathcal{O}(\Delta t^3)$$

4.4 Runge-Kutta Methods

Runge-Kutta schemes form a whole class of versatile integration methods. One of the simplest schemes is:

From $y_{n+1} = y_n + \int_{t_n}^{t_{n+1}} f(y(t))\,dt$

approximate the integral with the (implicit) trapezoidal rule:

$$y_{n+1} = y_n + \frac{f(y_n) + f(y_{n+1})}{2}\Delta t$$

Idea: predict unknown y_{n+1} on r.h.s. by an Euler step, yielding a second order accurate Runge-Kutta scheme:

$$k_1 = f(y_n, t_n)$$
$$k_2 = f(y_n + k_1 \Delta t, t_{n+1})$$
$$y_{n+1} = \frac{k_1 + k_2}{2}\Delta t + y_n$$

A variety of further Runge-Kutta schemes of different order can be defined. A commonly used scheme is the 4th order Runge-Kutta:

$$k_1 = f(y_n, t_n)$$
$$k_2 = f\left(y_n + k_1 \frac{\Delta t}{2}, t_n + \frac{\Delta t}{2}\right)$$
$$k_3 = f\left(y_n + k_2 \frac{\Delta t}{2}, t_n + \frac{\Delta t}{2}\right)$$
$$k_4 = f(y_n + k_3 \Delta t, t_n + \Delta t)$$

$$y_{n+1} = y_n + \left(\frac{k_1}{6} + \frac{k_2}{3} + \frac{k_3}{3} + \frac{k_4}{6}\right)\Delta t + O(\Delta t^5)$$

4.5 The Leapfrog

Suppose we have a second order differential equation of the type

$$\ddot{x} = f(x)$$
$$v \equiv \dot{x}$$

The leapfrog integration scheme is the mapping $(x_n, v_n) \rightarrow (x_{n+1}, v_{n+1}$
defined as

$$v_{n+1/2} = v_n + f(x_n)\frac{\Delta t}{2}$$

$$x_{n+1} = x_n + v_{n+\frac{1}{2}}\,\Delta t$$

$$v_{n+1} = v_{n+\frac{1}{2}} + f(x_{n+1})\frac{\Delta t}{2}$$

This is a second-order accurate scheme that is also time-symmetric. It is a symplectic integrator.

4.6 Symplectic Integrators

Symplectic methods are structure-preserving integration methods that observe important special properties of Hamiltonian systems: Such systems have first conserved integrals (such as the energy), they exhibit phase-space conservation as described by the Liouville theorem, and more generally, they preserve Poincaré's integral invariants.

A <u>linear map</u> $F: \mathbb{R}^{2d} \rightarrow \mathbb{R}^{2d}$ is called symplectic if

$$\omega(F\vec{\xi}, F\vec{\eta}) = \omega(\vec{\xi}, \vec{\eta})$$

for all vectors $\vec{\xi}, \vec{\eta} \in \mathbb{R}^{2d}$, where ω gives the area of the parallelogramm spanned by the two vectors.

A <u>differentiable map</u> $g: U \rightarrow \mathbb{R}^{2d}$ with $U \in \mathbb{R}$ is called symplectic if its Jacobian matrix is everywhere symplectic, i.e.

$$\omega(g'\vec{\xi}, g'\vec{\eta}) = \omega(\vec{\xi}, \vec{\eta})$$

<u>Poincaré's theorem</u> states that the time evolution generated by a Hamiltonian in phase-space is a symplectic transformation.

This suggests that there is a close connection between exact solutions of Hamiltonians and symplectic transformations. Furthermore, two consecutive symplectic transformations are again symplectic.

Dynamical problems that are described by the Hamiltonian of the form

$$H(p,q) = \frac{p^2}{2m} + U(q)$$

are quite common. These systems have separable Hamiltonians that can be written as

$$H(p,q) = H_{kin}(p) + H_{pot}(q)$$

For $H = H_{kin} = \frac{p^2}{2m}$, the equations of motion are

$$\begin{aligned} \dot{q} &= \frac{\partial H}{\partial p} = \frac{p}{m} \\ \dot{p} &= -\frac{\partial H}{\partial q} = 0 \end{aligned} \quad \Rightarrow \quad \begin{aligned} q_{n+1} &= q_n + \frac{p_n}{m}\Delta t \\ p_{n+1} &= p_n \end{aligned}$$

This solution is exact for the given Hamiltonian for arbitrarily long time intervals Δt. The solution constitutes a symplectic mapping, given that it's the solution of a Hamiltonian.

The potential part, $H = H_{pot} = U(q)$, leads to the following eqns

$$\begin{aligned} \dot{q} &= \frac{\partial H}{\partial p} = 0 \\ \dot{p} &= -\frac{\partial H}{\partial q} = -\frac{\partial U}{\partial q} \end{aligned} \quad \Rightarrow \quad \begin{aligned} q_{n+1} &= q_n \\ p_{n+1} &= p_n - \frac{\partial U}{\partial q}\Delta t \end{aligned}$$

This is again an exact solution independent of Δt, therefore a symplectic transformation.

Let $\varphi_{\Delta t}(H)$ be an operator that describes the mapping of phase-space under a Hamiltonian H that is evolved over a time interval Δt.

$\Rightarrow$ The Leapfrog is given by

$$\varphi_{\Delta t, LF} = \varphi_{\frac{\Delta t}{2}}(H_{pot}) \circ \varphi_{\Delta t}(H_{kin}) \circ \varphi_{\frac{\Delta t}{2}}(H_{pot})$$

for a separable Hamiltonian $H = H_{kin} + H_{pot}$.

Since each individual step of the leapfrog is symplectic, the concatenation of $\varphi_{\Delta t, LF}$ is also symplectic. The leapfrog generates the exact solution of a modified Hamiltonian $H_{leap} = H + H_{err}$, where

$$H_{err} \propto \frac{\Delta t^2}{12} \left\{ \{ H_{kin}, H_{pot} \}, H_{kin} + \frac{1}{2} H_{pot} \right\} + \mathcal{O}(\Delta t^3)$$

This explains the superior long-term stability of the integration of conservative systems with the Leapfrog:
Because it respects phase-space conservation, secular trends are largely absent, and the long term energy error stays bounded and reasonably small.

5. Numerical Hydrodynamics

5.1 Euler and Navier-Stokes equations

Fluid equations (for a perfect, ideal fluid) are the most common type of PDE's to be solved in astrophysics, even more common than the Poisson equation. Gravity can be treated as a fixed external force in many problems.

There are two approaches to fluid dynamics:

In the eulerian approach, one seeks the solution of the flow from a static frame of reference from which the flow is observed: $\vec{v} = \vec{0}$

In the lagrangian approach, one seeks the solution of the flow in a reference frame co-moving with the flow $(\vec{v} = \vec{v}(\vec{x}, t))$

The Eulerian approach lends itself naturally to the adoption of grid-based codes, in which a fixed mesh introduces the static reference frame.

The Lagrangian approach lends itself naturally to particle-based codes, in which some macroparticle tracers of the fluid model the trajectories of fluid elements analogously to the case of star clusters or dark matter lumps in collisionless systems. Since the fluid equations are moments of the Boltzmann equation, they are perfectly consistent with the idea of using macroparticles to sample the various quantities.

The gas flows in astrophysics are often of extremely low density, making internal friction in the gas extremely small. In the limit of vanishing viscosity, we arrive at the ideal gas dynamics, described by the Euler equations:

mass conservation $\quad \frac{\partial \rho}{\partial t} + \nabla(\rho \vec{v}) = 0 \qquad \frac{d\rho}{dt} + \rho \nabla\cdot\vec{v} = 0$

momentum cons. $\quad \frac{\partial(\rho\vec{v})}{\partial t} + \nabla(\rho\vec{v}\vec{v}^T + P) = 0 \qquad \frac{d\vec{v}}{dt} + \frac{\nabla P}{\rho} = 0$

energy conservation $\quad \frac{\partial(\rho e)}{\partial t} + \nabla[(\rho e + P)\vec{v}] = 0 \qquad \frac{de}{dt} + \frac{P}{\rho}\nabla\cdot\vec{v} = 0$

where $e = u + \vec{v}^2/2$ is the total energy per unit mass, and u is the thermal energy per unit mass.

The equations are not complete yet, we need an equation of state (a relation between pressure, density and internal energy). For an ideal gas, we have

$$P = (\gamma - 1)\rho u \qquad \text{with } \gamma = c_p/c_v = 5/3 \text{ for monoatomic gas}$$

Real fluids have internal stresses due to viscosity which dissipates relative motions of the fluid into heat. The Navier-Stokes equations are given by

$$\frac{\partial \rho}{\partial t} + \nabla(\rho\vec{v}) = 0$$

$$\frac{\partial}{\partial t}(\rho\vec{v}) + \nabla(\rho\vec{v}\vec{v}^T + P) = \nabla\Pi$$

$$\frac{\partial}{\partial t}(\rho e) + \nabla[(\rho e + P)\vec{v}] = \nabla(\Pi\vec{v})$$

$$\Pi = \eta\left[\nabla\vec{v} + (\nabla\vec{v})^T - \frac{2}{3}(\nabla\cdot\vec{v})\mathbb{1}\right] + \xi(\nabla\cdot\vec{v})\mathbb{1}$$

Π is the viscous stress tensor, which is a material property. η scales the traceless part of of the tensor and describes the shear viscosity. ξ gives the strength of the diagonal part, and is the so-called bulk viscosity.

5.2 Smoothed Particle Hydrodynamics (SPH)

SPH is a technique for approximating the continuum dynamics of fluids through the use of particles, which may also be viewed as interpolation points. The principle idea is to treat hydrodynamics in a completely mesh-free fashion in terms of a set of sampling particles. Hydrodynamical equations of motion are then derived for these particles, yielding quite a simple and intuitive formulation of gas dynamics.
Moreover, it turns out that the particle representation of SPH has excellent conservation properties: Energy, linear momentum, angular momentum, mass, and entropy are simultaneously conserved.* In addition, there are no advection errors and the scheme is fully Galilean invariant. Due to its Lagrangian character, the local resolution of SPH follows the mass flow automatically, a property that is convenient in representing the large density contrasts often encountered in astrophysical problems.

Particles are used as interpolation points to obtain the values of the hydrodynamical variables at the location of a particle. The values of the hydro variables are obtained using convolution integrals in order to obtain a smooth estimate.
The convolution kernel is chosen based on a trade-off between interpolating on a volume large enough to have an as smooth as possible function reconstruction (e.g. smooth out Poisson noise due to discretisation) and have enough resolution locally to identify features in the flow (e.g. resolves sudden density/pressure changes)

* The conservation is valid in absence of shocks, in which case artificial viscosity needs to be introduced and energy conservation is lost.

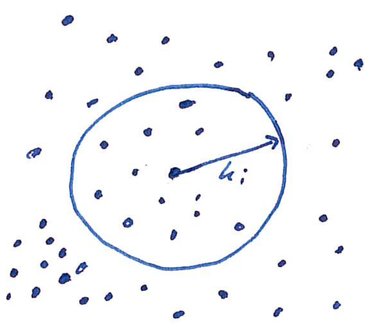
For any fluid field $F(\vec{r})$ one writes the corresponding smoothed interpolated version using a finite set of particles as

$$F_s(\vec{r}) = \int F(\vec{r}\,')\, W(\vec{r}-\vec{r}\,', h)\, d\vec{r}\,' \qquad \text{[interpolated, smoothed version of } F]$$

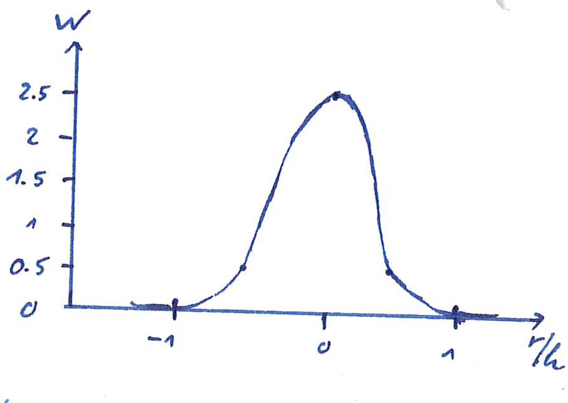
h is the "smoothing length" (characteristic width of the kernel).
The smoothed interpolated version is obtained through a convolution with a kernel $W(\vec{r}, h)$. The kernel $W(\vec{r}, h)$ is normalized to unity, and $W(\vec{r}, h) \xrightarrow{h\to 0} \delta(\vec{r})$. We furthermore require that the kernel is symmetric and sufficiently smooth to make it at least differentiable twice.
A standard kernel choice is the cubic spline with finite support:
The kernel is spherical: $W(\vec{r}, h) = w\left(\frac{r}{2h}\right)$, and

$$w_{3D} = \frac{8}{\pi}\begin{cases} 1 - 6q^2 + 6q^3 & 0 \le q \le \frac{1}{2} \\ 2(1-q)^3 & \frac{1}{2} < q \le 1 \\ 0 & q > 1 \end{cases}$$

This kernel interpolant is second-order accurate for regularly distributed points.
The smoothing length h defines the characteristic size of the interpolation kernel.



By choosing a finite support kernel rather than e.g. a gaussian one a finite volume is defined that has short tails, which means that the interpolation is inherently local. This enables high resolution in principle by weighing only on the region quite close to a particle, which also makes the calculation fast.



A typical choice is to fix the number of neighbours to define h at any given time.
This makes the estimate of any field $F(\vec{r})$ automatically adaptive.
In general, the smoothing length can be variable in space: $h = h(\vec{r}, t)$ to account for variations in the sampling density.

Suppose now we know the field F at a set of point $\vec{r}_i$, i.e. $F_i = F(\vec{r}_i)$. Then integrals have to be replaced by sums. Each particle is given a mass m_i and thus associated density $\rho_i = \frac{m_i}{V_i}$, so that we can write

$$F_s(\vec{r}) \simeq \sum_j \frac{m_j}{\rho_j} F_j \, W(\vec{r} - \vec{r}_j, h)$$

Particles will be in general distibuted in a regular fashion since they will represent physical quantities such as density or pressure profiles of physical systems. This aids the accuracy of the method.

The field functions $F_s(\vec{r})$ are defined everywhere, not just at the underlying points, and are differentiable, since the kernel is differentiable (albeit with a considerably higher interpolation error for the derivative).

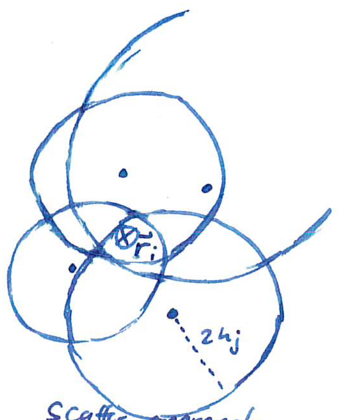
If we set $F(\vec{r}) = \rho(\vec{r})$, we obtain

$$\rho_s(\vec{r}) \simeq \sum_j m_j W(\vec{r} - \vec{r}_j, h)$$

the "SPH density".

The smoothing length can be made adaptive in space, as mentioned before, e.g. by fixing the number of neighbours. There are two basic approaches to introduce adaptive kernels:

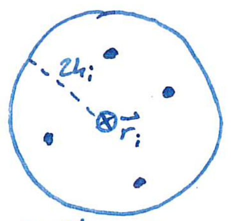
1) "scatter" approach: $W(\vec{r}, h) = W(|\vec{r}_i - \vec{r}_j|, h(\vec{r}_j))$
particle i collects the contributions from all other particles j whose smoothing volume h_j scatter onto location $\vec{r}_i$



Scatter approach

2) "gather" approach: $W(\vec{r}, h) = W(|\vec{r}_i - \vec{r}_j|, h(\vec{r}_i))$
particle i gathers the contributions from all particles which centres fall within the smoothing volume of i

If all particles have the same smoothing length, both approaches are equivalent.



gather approach

The "gather" approach is computationally easier and faster as one uses only the smoothing length of one particle to estimate the density at the interpolation point corresponding to it:

$$\rho_i = \sum_{j=1}^{N} m_j W(\vec{r}_i - \vec{r}_j, h_i)$$

Because we are using a kernel with finite support, we are only summing over the N_i particles contained within a distance $r = 2h$, beyond that, $W = 0$. $\Rightarrow$ The SPH density estimate is of order $N \times O(N_{neigh})$, where N_{neigh} is the (fixed) number of neighbours that defines the kernel. With a gaussian kernel, which does not have finite support, the cost would have been $O(N$

The same mathematical formulation used for the density can be used for all fluid fields and also for differential operators needed to solve the Euler equation:

$$(\nabla \cdot \vec{v})_i = \sum_j \frac{m_j}{\rho_j} \vec{v}_j \cdot \nabla_i W(\vec{r}_i - \vec{r}_j, h)$$

The derivative is applied only to the kernel because v_j is not a function, but rather a scalar value, so it gives zero under differentia
An alternative estimate can be obtained by exploiting the identity

$$\rho \nabla \cdot \vec{v} = \nabla(\rho \vec{v}) - \vec{v} \cdot \nabla \rho$$

$$\Rightarrow (\nabla \cdot \vec{v})_i = \frac{1}{\rho_i} \sum_j m_j (\vec{v}_j - \vec{v}_i) \cdot \nabla_i W(\vec{r}_i - \vec{r}_j, h)$$

This pairwise formulation turns out to be more accurate in practice. In particular, it always provides a vanishing velocity divergence if all particle velocities are equal.

SPH equations of motion

In Lagrangian form, the inviscid fluid equations are:

$$\frac{d\rho}{dt} + \rho\nabla\cdot\vec{v} = 0$$

$$\frac{d\vec{v}}{dt} + \frac{\nabla P}{\rho} = 0$$

$$\frac{du}{dt} + \frac{P}{\rho}\nabla\cdot\vec{v} = 0$$

where $\frac{d}{dt} = \frac{\partial}{\partial t} + \vec{v}\cdot\nabla$ is the convective/comoving/Lagrangian/total derivative

One approach is to recast the equations in SPH form by replacing all individual fields and operators with their SPH versions, but fuck that. A more elegant approach stems from a Lagrangian from which the Euler equations can be obtained:

$$\mathcal{L} = \int \rho\left(\frac{\vec{v}^2}{2} - u\right) dV$$ where u is the internal energy of the gas.

We start by discretising the Lagrangian in terms of fluid particle of mass m_i:

$$L_{SPH} = \sum_i \left(\frac{1}{2} m_i \vec{v}_i^2 - m_i u_i\right) \qquad \left[dV \rightarrow \frac{m_i}{\rho_i}\right]$$

Note that it doesn't depend on time and is invariant under translation and rotation, as it doesn't depend on spatial coordinates explicitly either. This expresses total energy, momentum and angular momentum conservation.

Using the variational principle $\delta L = 0$, we get the Euler-Lagrange equations: $\frac{d}{dt}\frac{\partial L}{\partial \dot{\vec{r}}_i} - \frac{\partial L}{\partial \vec{r}_i} = 0$

Since $\vec{r}$ is an (implicit) function of the smoothing length h for each particle, we need to have an educated definition of h for the derivations. In early SPH works the smoothing length was defined using a variable number of neighbours to cope with the issue that in low density regions, kernels tend to be large and very small in high density regions, giving very different volume sampling at high and low density. This results in energy conservation errors.

In modern SPH, one fixes the number of neighbours to avoid that. We can also afford more particles now in simulations, so we can sample everywhere better. This ensures mass conservation within the kernel if the masses of the particles are equal, i.e. $\rho_i h_i^3 = \text{const}$

This becomes a useful conservation property that we can exploit in deriving the SPH hydro equations from the Euler-Lagrange equations. We use that the pressure of particles is

$$P_i = (\gamma - 1)\rho_i u_i$$ for an inviscid fluid.
γ is the adiabatic index.

$$\Rightarrow u = \frac{P_i}{(\gamma-1)\rho_i} = \frac{P_i}{(\gamma-1)\rho_i(\vec{r})}$$

$$L = \sum_i \left(\frac{1}{2} m_i \vec{v}_i^{\,2} - m_i u_i \right)$$

$$\delta L = 0 \quad \Rightarrow \quad \frac{d}{dt}\frac{\partial L}{\partial \dot{\vec{r}}_i} - \frac{\partial L}{\partial \vec{r}_i} = 0$$

$$\frac{\partial L}{\partial \dot{\vec{r}}_i} = \frac{\partial L}{\partial \vec{v}_i} = \sum_j \frac{1}{2} m_j \frac{\partial (\vec{v}_j^{\,2})}{\partial \vec{v}_i} = m_i \vec{v}_i$$

$$\Rightarrow \frac{d}{dt}\frac{\partial L}{\partial \dot{\vec{r}}_i} = \frac{d}{dt}\frac{\partial L}{\partial \vec{v}_i} = m_i \frac{d\vec{v}_i}{dt}$$

$$\frac{\partial L}{\partial \vec{r}_i} = -\frac{\partial}{\partial \vec{r}_i} \sum_j \frac{m_j P_j}{(\gamma-1)\rho_j} = \sum$$

Deriving SPH Equations of Motion

Euler Equations for an inviscid gas follow from the Lagrangian: $\mathcal{L} = \int \rho \left(\frac{1}{2}\vec{v}^2 - u\right) dV$

For SPH: $\rho = \sum_i m_i \delta(\vec{r} - \vec{r}_i)$

$$\Rightarrow \quad L_{SPH} = \sum_i m_i \left(\frac{1}{2}\vec{v}_i^2 - u_i\right)$$

Using the equations of state: $P = A\rho^\gamma$, $u = \frac{1}{\gamma - 1}\frac{P}{\rho}$

Get equations of motion from

$$\frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \vec{v}} - \frac{\partial \mathcal{L}}{\partial \vec{r}} = 0$$

$$\frac{\partial \mathcal{L}}{\partial \vec{v}_j} = \frac{\partial}{\partial \vec{v}_j}\sum_i m_i\left(\frac{1}{2}\vec{v}_i^2 - u_i\right) = m_j \vec{v}_j$$

$$\Rightarrow \frac{d}{dt}\frac{\partial \mathcal{L}}{\partial \vec{v}_j} = m_j \frac{d\vec{v}_j}{dt}$$

$$\frac{\partial \mathcal{L}}{\partial \vec{r}_j} = -\sum_i m_i \frac{\partial}{\partial \vec{r}_j} u_i = -\sum_i m_i \left[\frac{\partial u_i}{\partial P_i}\frac{dP_i}{d\rho_i} + \frac{\partial u_i}{\partial \rho_i}\right]\frac{d\rho_i}{d\vec{r}_j}$$

$$\frac{\partial u_i}{\partial P_i} = \frac{1}{\gamma - 1}\frac{1}{\rho}; \qquad \frac{\partial u_i}{\partial \rho_i} = \frac{-1}{\gamma - 1}\frac{P}{\rho^2}$$

$$\frac{\partial P}{\partial \rho} = \gamma A \rho^{\gamma - 1} = \frac{\gamma P}{\rho}$$

$$\Rightarrow \left[\frac{\partial u_i}{\partial P_i}\frac{dP_i}{d\rho_i} + \frac{\partial u_i}{\partial \rho_i}\right] = \frac{1}{\gamma - 1}\frac{1}{\rho_i}\frac{\gamma P_i}{\rho_i} - \frac{1}{\gamma - 1}\frac{P_i}{\rho_i^2} = \frac{\gamma - 1}{\gamma - 1}\frac{P_i}{\rho_i^2}$$

$$= \frac{P_i}{\rho_i^2}$$

$$\Rightarrow \quad m_j \frac{d\vec{v}_j}{dt} = -\sum_i m_i \frac{P_i}{\rho_i^2}\frac{d\rho_i}{d\vec{r}_j}$$

Now compute $\frac{d\rho_i}{d\vec{r}_j}$

Keep in mind that $\rho_i h_i^3 = \text{const}$: h_i defined by # neighbours

Also $\rho_i = \sum_{j=1}^{N} m_j W(\vec{r}_i - \vec{r}_j, h_i)$

$$\frac{d\rho_i}{d\vec{r}_j} = \frac{\partial\rho_i}{\partial\vec{r}_j} + \frac{\partial\rho_i}{\partial h_i}\frac{\partial h_i}{\partial\vec{r}_j}$$

Consider $\frac{d}{d\vec{r}_j}(\rho_i h_i^3) = 0 = \left(\frac{\partial\rho_i}{\partial\vec{r}_j} + \frac{\partial\rho_i}{\partial h_i}\frac{\partial h_i}{\partial\vec{r}_j}\right) h_i^3 + \rho_i 3 h_i^2 \frac{\partial h_i}{\partial\vec{r}_j}$

$$= h_i^3\left(\frac{\partial\rho_i}{\partial\vec{r}_j} + \frac{3\rho_i}{h_i}\frac{\partial h_i}{\partial\vec{r}_j} + \frac{\partial\rho_i}{\partial h_i}\frac{\partial h_i}{\partial\vec{r}_j}\right) = h_i^3\left[\frac{\partial\rho_i}{\partial\vec{r}_j} + \frac{\partial\rho_i}{\partial h_i}\frac{\partial h_i}{\partial\vec{r}_j}\left(1 + \frac{3\rho_i}{h_i}\left(\frac{\partial\rho_i}{\partial h_i}\right)^{-1}\right)\right]$$

$$\Rightarrow \frac{\partial\rho_i}{\partial h_i}\frac{\partial h_i}{\partial\vec{r}_j} = -\frac{\partial\rho_i}{\partial\vec{r}_j}\left(1 + \frac{3\rho_i}{h_i}\left(\frac{\partial\rho_i}{\partial h_i}\right)^{-1}\right)^{-1}$$

$$\Rightarrow \frac{d\rho_i}{d\vec{r}_j} = \frac{\partial\rho_i}{\partial\vec{r}_j} + \frac{\partial\rho_i}{\partial h_i}\frac{\partial h_i}{\partial\vec{r}_j} = \frac{\partial\rho_i}{\partial\vec{r}_j} - \frac{\partial\rho_i}{\partial\vec{r}_j}\left(1 + \frac{3\rho_i}{h_i}\left(\frac{\partial\rho_i}{\partial h_i}\right)^{-1}\right)^{-1} = \frac{\partial\rho_i}{\partial\vec{r}_j}\left(1 - \frac{1}{1 + \frac{3\rho_i}{h_i}\left(\frac{\partial\rho_i}{\partial h_i}\right)^{-1}}\right)$$

$$= \frac{\partial\rho_i}{\partial\vec{r}_j}\left(\frac{1 + 3\rho_i/h_i\left(\frac{\partial\rho_i}{\partial h_i}\right)^{-1} - 1}{1 + 3\rho_i/h_i\left(\frac{\partial\rho_i}{\partial h_i}\right)^{-1}}\right) = \frac{\partial\rho_i}{\partial\vec{r}_j}\left(\frac{1}{\frac{h_i}{3\rho_i}\left(\frac{\partial\rho_i}{\partial h_i}\right) + 1}\right)$$

$$= {}^{\partial}\left(1 + \frac{h_i}{3\rho_i}\left(\frac{\partial\rho_i}{\partial h_i}\right)\right)^{-1}\frac{\partial\rho_i}{\partial\vec{r}_j}$$

$$\equiv f_i \nabla_j \rho_i \qquad \text{with} \quad f_i = \left(1 + \frac{h_i}{3\rho_i}\frac{\partial\rho_i}{\partial h_i}\right)^{-1} \quad \text{and} \quad \nabla_j\rho_i = \left.\frac{\partial\rho_i}{\partial\vec{r}_j}\right|_{h=con}$$

Using $\rho_i = \sum_j m_j W(\vec{r}_i - \vec{r}_j, h_i) \equiv \sum_j m_j W_{ij}(h_i)$

$$\Rightarrow \nabla_j \rho_i = \nabla_j\left(\sum_k m_k W_{ik}(h_i)\right) = m_j \nabla_j W_{ji}(h_i) + \delta_{ij}\sum_k m_k \nabla_j W_{kj}(h_j)$$

$$\Rightarrow m_j\frac{d\vec{v}_j}{dt} = -\sum_i m_i \frac{P_i}{\rho_i^2}\frac{d\rho_i}{d\vec{r}_j} = -\sum_i m_i\frac{P_i}{\rho_i^2} f_i\left(m_j\nabla_j W_{ji}(h_i) + \delta_{ij}\sum_k m_k \nabla_j W_{kj}(h_j)\right)$$

$$= -m_j\sum_i m_i \frac{P_i}{\rho_i^2} f_i \nabla_j W_{ji}(h_i) - \sum_i m_i \frac{P_i}{\rho_i^2} f_i \delta_{ij}\sum_k m_k \nabla_j W_{kj}(h_j)$$

| use $\delta_{ij} \to i \Rightarrow j$, first sum disappears; change $k \to i$

$$= -m_j\sum_i m_i\left[\frac{P_i}{\rho_i^2} f_i \nabla_j W_{ji}(h_i) + \frac{P_j}{\rho_j^2} f_j \nabla_j W_{ij}(h_j)\right]$$

$$\Rightarrow \frac{d\vec{v}_j}{dt} = -\sum_i m_i\left[\frac{P_i}{\rho_i^2} f_i \nabla_j W_{ji}(h_i) + \frac{P_j}{\rho_j^2} f_j \nabla_j W_{ij}(h_j)\right]$$

We obtained the equations of motion:

$$\frac{d\vec{v}_i}{dt} = -\sum_{j=1}^{N} m_j \left[f_i \frac{P_i}{\rho_i^2} \nabla_i W_{ij}(h_i) + f_j \frac{P_j}{\rho_j^2} \nabla_i W_{ij}(h_j) \right]$$

$$\text{with} \quad f_i = \left[1 + \frac{h_i}{3\rho_i} \frac{\partial \rho_i}{\partial h_i} \right]^{-1}$$

$$\text{and} \quad W_{ij}(h) = W(|\vec{r}_i - \vec{r}_j|, h)$$

The correction factors f_i can be easily calculated alongside the density estimate, all that is required is an additional summation to get $\partial\rho_i / \partial\vec{r}_i$ for each particle.

This SPH fluid momentum equation is a simple differential equation as opposed to the original PDEs. The mass conservation equation as well as the total energy equation are already taken care of, because the particle masses and their specific entropies stay constant for reversible gas dynamics.
The differential form though will be problematic at discontinuities (e.g. shocks or shearing flows)

Thermodynamics in SPH: Entropy and Energy

To describe the gas thermodynamics one can use the (specific) internal energy u or entropy s as the fundamental variable.
The formulation with entropy is simpler in absence of discontinuities in the flow, because then the system is thermodynamically reversible and entropy is simply conserved.

For the reversible isentropic case:

$$P_i = A_i \rho_i^\gamma = (\gamma - 1)\rho_i u_i \quad \Rightarrow \quad u_i(\rho_i) = A_i \frac{\rho_i^{\gamma-1}}{\gamma - 1}$$

with $A_i(s)$ = constant, depending on specific entropy s.

For the energy approach: (reversible case)

$$u = \frac{P}{(\gamma-1)\rho} = \frac{A\rho^{\gamma}}{(\gamma-1)\rho} = \frac{A}{\Gamma-1}\rho^{\gamma-1}$$

The time evolution is given by

$$\frac{du_i}{dt} = \frac{A}{\gamma-1}\frac{d}{dt}\left(\rho_i^{\gamma-1}\right) = \frac{A}{\Gamma-1}\frac{\partial(\rho_i^{\gamma-1})}{\partial\rho_i}\sum_j \frac{\partial\rho_i}{\partial\vec{r}_j}\frac{d\vec{r}_j}{dt} = A\frac{\Gamma-1}{\gamma-1}\rho_i^{\gamma-2}\sum_j \vec{v}_j\cdot\frac{\partial\rho_i}{\partial\vec{r}_j}$$

$$= \frac{P_i}{\rho_i^2}\sum_j \vec{v}_j\cdot f_i\left[m_j\nabla_j W_{ji}(h_i) + \delta_{ij}\sum_k m_k\nabla_j W_{kj}(h_j)\right]$$

$$= \frac{P_i}{\rho_i^2}\sum_j \vec{v}_j f_i m_j\nabla_j W_{ji}(h_i) + \frac{P_i}{\rho_i^2} f_i\vec{v}_i\sum_k m_k\underbrace{\nabla_i W_{ik}(h_i)}_{\text{Use } W_{ik} = -W_{ki};\ \text{exchange } i\to j}$$

$$= \frac{P_i}{\rho_i^2} f_i\sum_j\left[m_j\vec{v}_j\nabla_j W_{ji}(h_i) - m_j\vec{v}_i\nabla_j W_{ji}(h_i)\right]$$

$$= \frac{P_i}{\rho_i^2} f_i\sum_j m_j(\vec{v}_i - \vec{v}_j)\nabla_j W_{ij}(h_i)$$

$$\frac{du_i}{dt} = f_i\frac{P_i}{\rho_i^2}\sum_j m_j(\vec{v}_i - \vec{v}_j)\cdot\nabla W_{ij}(h_i)$$

which needs to be integrated along the equation of motion if one wants to use the thermal energy as independent thermodynamic variable.

The differential form of the equations used so far breaks down at sharp discontinuities such as shocks. In computational fluid dynamics, there are two general ways to overcome this issue:

1) switch to integral form (conservative form)

2) introduce a dissipation term (artificial viscosity)

Artificial Viscosity

In SPH, all functions are affected by Poisson noise, namely the fact that the fluid is coarsely sampled by particles, so that e.g. positions or velocities deviate from a reference "exact" profile. The error goes $\sim 1/N^{1/2}$, where both the total N and the N_{neigh} contribute to the error.

This alone implies that there cannot be really sharp discontinuities by construction, which means that equations in the differential form would not break down in practice. But if one tries to model a shock front in SPH with the equations used so far, ringing arises post-shock, and the shock profile is never sharp.

In real fluids, viscosity is present both at microscopic scale (Van der Waals forces), while on macroscopic scales there can be other phenomena producing a viscous force, such as magnetic currents or radiation pressure drag, which are not contained in the Euler equations. Moreover in shocks, kinetic energy is dissipated into heat, otherwise the shock will never end. At a shock front, the integral form equations yield the Rankine-Hugoniot jump conditions that relate upstream and downstream states of the fluid. These relations show that the specific entropy of the gas always increases at a shock front, implying that in the shock layer itself the gas dynamics can no longer be described as inviscid. In turn, this also implies that the discretised SPH equations we derived above can not correctly describe a shock simply because they keep the entropy strictly constant.

However, shocks or contact discontinuities do not occur only in the presence of physical effects not contained in the Euler equations. They can arise in a fluid described as inviscid because dissipation of kinetic energy into heat occurs due to (unresolved) molecular viscosity, where the perfect fluid description breaks down anyway.

But if one sticks to SPH, the first issue is how to model a sharp discontinuity in a function in a smooth and accurate manner, namely e.g. capturing the relation between pre-shock and post-shock values in cases where analytical solutions are known, such as in isothermal and adiabatic shocks.

One must thus allow for a modification of the dynamics at shocks and somehow introduce the necessary dissipation. This is usually accomplished in SPH by an artificial viscosity. Its purpose is to dissipate kinetic energy into heat and to produce entropy in the process. The usual approach is to parametrise the artificial viscosity in terms of a friction force that damps the relative motion of particles.
Provided the viscosity is introduced into the dynamics in a conservative fashion, the conservation laws themselves ensure that the right amount of dissipation occurs at a shock front.

One of the main challenges is to formulate viscosity in a way that it turns off or at least becomes sufficiently small away from discontinuities/shocks. Otherwise, the fluid does not obey the Euler equations anymore (and not even the Navier-Stokes for a viscous fluid since the viscosity here is introduced only for numerical reasons).

Standard Monaghan viscosity:

Add force term for each particle's equation of motion:

$$\left.\frac{d\vec{v}_i}{dt}\right|_{visc} = -\sum_{j=1}^{N} m_j \Pi_{ij} \nabla_i \overline{W_{ij}}$$

with $\overline{W_{ij}} = \frac{1}{2}\left[W_{ij}(h_i) + W_{ij}(h_j)\right]$

and Π_{ij} = viscous tensor

The symmetrized kernel is such that, if the viscous tensor Π_{ij} is symmetric, then the viscous force between two particles will be antisymmetric, allowing conservation of linear and angular momentum.

To conserve energy, we need to compensate the work done against the viscous force in the thermal reservoir:

$$\left.\frac{dA_i}{dt}\right|_{visc} = \frac{1}{2}\frac{\gamma-1}{\rho_i^{\gamma-1}}\sum_{j=1}^{N} m_j \Pi_{ij} \vec{v}_{ij}\cdot\nabla_i \overline{W_{ij}} \qquad \text{(entropy form)}$$

$$\left.\frac{du_i}{dt}\right|_{visc} = \frac{1}{2}\sum_{j=1}^{N} m_j \Pi_{ij} \vec{v}_{ij}\cdot\nabla_i \overline{W_{ij}} \qquad \text{(energy formulation)}$$

with $\vec{v}_{ij} = \vec{v}_i - \vec{v}_j$

Commonly: $\Pi_{ij} = \begin{cases} \left[-\alpha c_{ij}\mu_{ij} + \beta\mu_{ij}^2\right]/\rho_{ij} & \text{if } \vec{v}_{ij}\cdot\vec{r}_{ij} < 0 \\ 0 & \text{otherwise} \end{cases}$

with $\mu_{ij} = \frac{h_{ij}\vec{v}_{ij}\cdot\vec{r}_{ij}}{|\vec{r}_{ij}|^2 + \epsilon h_{ij}^2}$; h_{ij}, ρ_{ij} = arithmetic means of the quantities for particles i,j

c_{ij} = mean sound speed; $\vec{r}_{ij} \equiv \vec{r}_i - \vec{r}_j$

The strength of the viscosity is regulated by the parameters α and β, with typical values in the range $\alpha \simeq 0.5-1$, $\beta = 2\alpha$

$\epsilon \simeq 0.01$ is softening to prevent from singularities.

In this form, the artificial viscosity is basically a combination of a bulk and a von-Neumann-Riemann viscosity.

The entropy production is positive definite because viscosity turns only on for particles that approach each other: $\vec{v}_{ij} \cdot \vec{r}_{ij} < 0$

In differentialy rotating flows at low resolution, it may appear that particles are approaching due to the noisyness of the flow sampling. Rather than reducing the coefficients, a cleaner alternative is to add a rot $\vec{v}$ dependent term in Π, so that the viscosity goes to zero in shearing/rotating flows.

5.3 Eulerian Hydrodynamics Methods

In this class of methods, one solves the eulerian form of the fluid equations onto a mesh. The mesh can be cartesian, but also more complex (unstructured, e.g. using tesselation with triangles or tethraedra as in cylindrical or spherical coordinates)

The Euler equations are so-called hyperbolic (non-linear) conservation laws, in the form of partial differential equations.

Unfortunately, for partial differential equations one cannot give a general solution method that works equally well for all types of problems. Rather, each type requires different approaches. Important classes of solution schemes include:

- Finite difference methods

  Differential operators are approximated through finite difference approximations, usually on a regular (cartesian) mesh. An example is Poisson's equation treated with iterative (multigrid) methods.

- Finite volume methods

  The key aspect is to construct fluxes for all quantities at mesh faces and use the integral form of the PDEs to express the conservation of such fluxes over a certain small volume (usually the volume of a cell). But there is a numerical effect, numerical diffusion, that can compromise conservation laws.

- Finite Element Methods

  One solves the PDEs on elements (subvolumes) with a geometry appropriate to the problem and then connects the solutions at different domains using algebraic relations. The elements may have arbitrary shapes. The solution is then represented in terms of simple (usually polynomial) functions on the element, and then the PDE is transformed to an algebraic problem for the coefficients.

- Spectral methods

The solution is represented with a linear combination of functions, allowing the PDE to be transformed to algebraic equations or ordinary differential equations. Often this is done by applying Fourier techniques. But there are flow conditions that are impossible to capture, because they cannot be reduced to a linear expansion.
Example: High compressibility behaviours, shocks, contact discontinuities.

- Method of Lines

Semi-discrete approach: all derivatives except one are approximated with finite differences. The remaining derivative then forms a set of ordinary differential equations. This approach is often used in time-dependent problems.

Example: heat diffusion equation:

$$\frac{\partial u}{\partial t} + \lambda \frac{\partial^2 u}{\partial x^2} = 0 \longrightarrow \frac{du_i}{dt} + \lambda \frac{u_{i+1} + u_{i-1} - 2u_i}{h^2} = 0$$

Now we can integrate the PDE's as a set of time dependent ODES for each cell i.

Numerical instabilities can arise in certain types of time dependent PDEs, namely in hyperbolic PDEs in which a clear direction of propagation of the information on the state of the system exists.

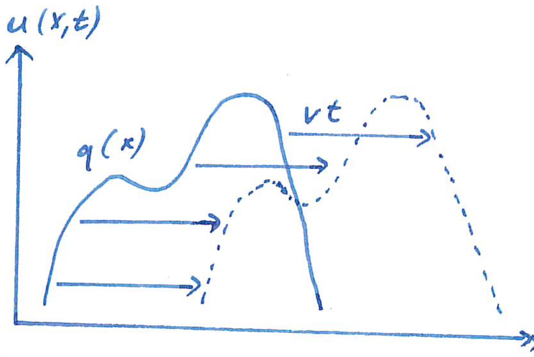
The Advection Problem

Numerical Eulerian hydrodynamics stems from coping with the advection problem. Think of a fluid laid down on a cartesian grid at some time $t=0$: Integrating the fluid equations means essentially to advect the functions defining the fluid $(\rho, \vec{v}, P)$ through the grid. These functions are defined at each point $\vec{x}$ of the grid at time $t=0$.

The simplest equation of this type is the advection equation in 1D:

$$\frac{\partial u}{\partial t} + \vec{v} \cdot \frac{\partial u}{\partial \vec{x}} = 0 = \frac{\partial u}{\partial t} + v \frac{\partial u}{\partial x}$$

where v is a constant parameter.



Any function $q(x)$ is a solution for u if

$$u(x,t) = q(x - vt)$$

then $\frac{du}{dt} = -v \frac{dq}{dx}$; $\frac{du}{dx} = \frac{dq}{dx} \Rightarrow \frac{du}{dt} + v \frac{du}{dx} = 0$

We can interpret $q(x)$ as the initial condition $q(x, t=0)$ for any v. Then the solution at any given time t is simply a copy of $q(x)$ translated by $v \cdot t$.

Points that start at a certain coordinate x_0 are advected to a new location $x_0 + vt$. The values at different times are connected by the characteristics: straight lines that follow the propagation of information (e.g. density) from upstream to downstream. (Downstream is where the flow is going to)

Now let's assume that we don't have an analytic solution, but want to obtain it numerically.

We can approach this straightforwardly with a discretisation of u on a mesh, using for example the method of lines:

$$\frac{du_i}{dt} + v\,\frac{u_{i+1} - u_{i-1}}{2h} = 0$$

and for the time derivative

$$u_i^{(n+1)} = u_i^{(n)} - v\,\frac{u_{i+1}^{(n)} - u_{i-1}^{(n)}}{2h}\,\Delta t \qquad \text{(Euler scheme)}$$

This is a complete update formula which can be readily applied to a given initial state on the grid. Unfortunately, this is violently unstable! If e.g. u is a step function, the returned solution is oscillatory. The reason for the numerical instability is that the flow propagates as the characteristics, i.e. from upstream to downstream. The update formula however includes values from both upstream (u_{i-1}) and downstream (u_{i+1}).* This is inconsistent with how the information flows, namely along the characteristics. The downstream value has not happened yet when we want to estimate u_i; We have a causality problem on our hands.

Instead, we can switch to a one-sided estimate which only involves upstream values:

$$\frac{du_i}{dt} + v\,\frac{u_i - u_{i-1}}{h} = 0$$

With this <u>upwind differencing</u>, numerical instability disappears!

But the discretisation now depends on the sign of v. For negative v, one instead has to use

$$\frac{du_i}{dt} + v\,\frac{u_{i+1} - u_i}{h} = 0$$

The solution is not advected in a perfectly faithful way, instead it is quite significantly smoothed out through numerical diffusion.

* upstream and downstream also refer to spatial coordinates, as the characteristics have a specific direction!

We can figure out where this strong diffusion in the 1st-order upwind scheme comes from. Let us rewrite the finite differencing update as

$$\frac{u_i - u_{i-1}}{h} = \frac{u_{i+1} - u_{i-1}}{2h} - \frac{u_{i+1} - 2u_i + u_{i-1}}{2h}$$

$$\Rightarrow \quad \frac{du_i}{dt} + v\,\frac{u_{i+1} - u_{i-1}}{2h} = \frac{vh}{2}\,\frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}$$

Now remember the diffusion equation:

$$\frac{\partial u}{\partial t} + v \cdot \frac{\partial u}{\partial x} = D\,\frac{\partial^2 u}{\partial x^2} \longrightarrow \left(\frac{\partial^2 u}{\partial x^2}\right)_i \simeq \frac{u_{i+1} - 2u_i + u_{i-1}}{h^2}$$

If we define the Diffusion constant $D = \frac{vh}{2}$, we are effectively solving a diffusion problem, not the original advection problem. With increased resolution $(h \to 0)$ the diffusion decreases $(D \to 0)$.

For the stability of the time integration, one needs to pick the time step in a physically self-consistent way: The timestep cannot be larger than the time it takes to propagate the flow from upstream to downstream, namely $\Delta t_{max} = h/v$. Otherwise, one would have to include information from u_{i-2} to keep the integration stable.
This leads to the Courant-Friedrichs-Levy (CFL) timestep condition:

$$\Delta t \leq h/v$$

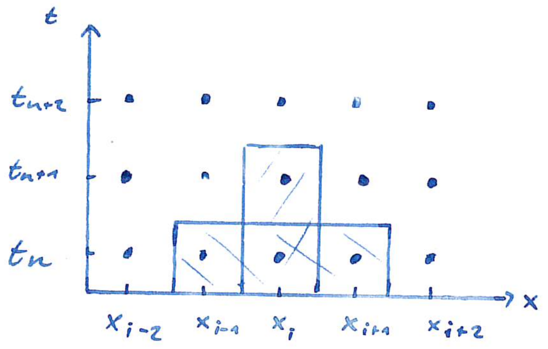
Similar conditions hold for the integration of other types of hyperbolic equations.

Example of hyperbolic conservation laws: Continuity equation

Consider a hyperbolic conservation law, such as the continuity equation for the mass density of a fluid:

$$\frac{\partial \rho}{\partial t} + \nabla (\rho \cdot \vec{v}) = 0$$

This has the form of the advection equation, but with a spatially variable velocity $\vec{v} = \vec{v}(\vec{x})$. Let $\vec{F} = \rho \vec{v}$ be the mass flux.



Let's study the problem in one spatial direction, and consider a discretisation both of the x- and t-axis:

$$\frac{\rho_i^{(n+1)} - \rho_i^{(n)}}{\Delta t} + \frac{F_{i+1}^{(n)} - F_{i-1}^{(n)}}{2\Delta x} = 0$$

giving the update rule $\rho_i^{(n+1)} = \rho_i^{(n)} + \frac{\Delta t}{2\Delta t}\left(F_{i-1}^{(n)} - F_{i+1}^{(n)}\right)$

This is again found to be unstable for the same reasons: it doesn't take into account in which direction the flow goes; it doesn't follow the direction of the characteristics.

Using an upwind scheme we can establish stability once again. In this case, the identification of the characteristics is obvious, but that is not always the case. We need more general schemes: Riemann solvers.

Riemann Problem and Solvers

The Riemann problem is a classic boundary value problem for a hyperbolic system, consisting of two piece-wise constant states (two half-spaces) that meet at a plane at time $t=0$

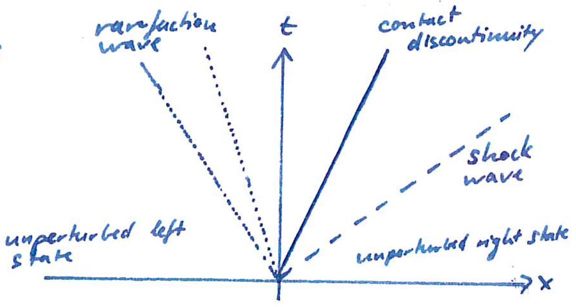
An important case is the Riemann problem for the Euler equations (i.e. for ideal gas dynamics). Here the left and right states of the interface can, for example, be uniquely specified by giving the three "primitive variables":

$$U_L = \begin{pmatrix} \rho_L \\ p_L \\ v_L \end{pmatrix}, \quad U_R = \begin{pmatrix} \rho_R \\ p_R \\ v_R \end{pmatrix}$$

the left and right states are known at $t=0$, the solution at $t>0$ needs to be found.

For an ideal gas, this initial value problem can be solved analytically, but not written explicitly, as it contains an implicit equation that requires numerical root-finding.

The solution always contains characteristics for three self-similar waves:

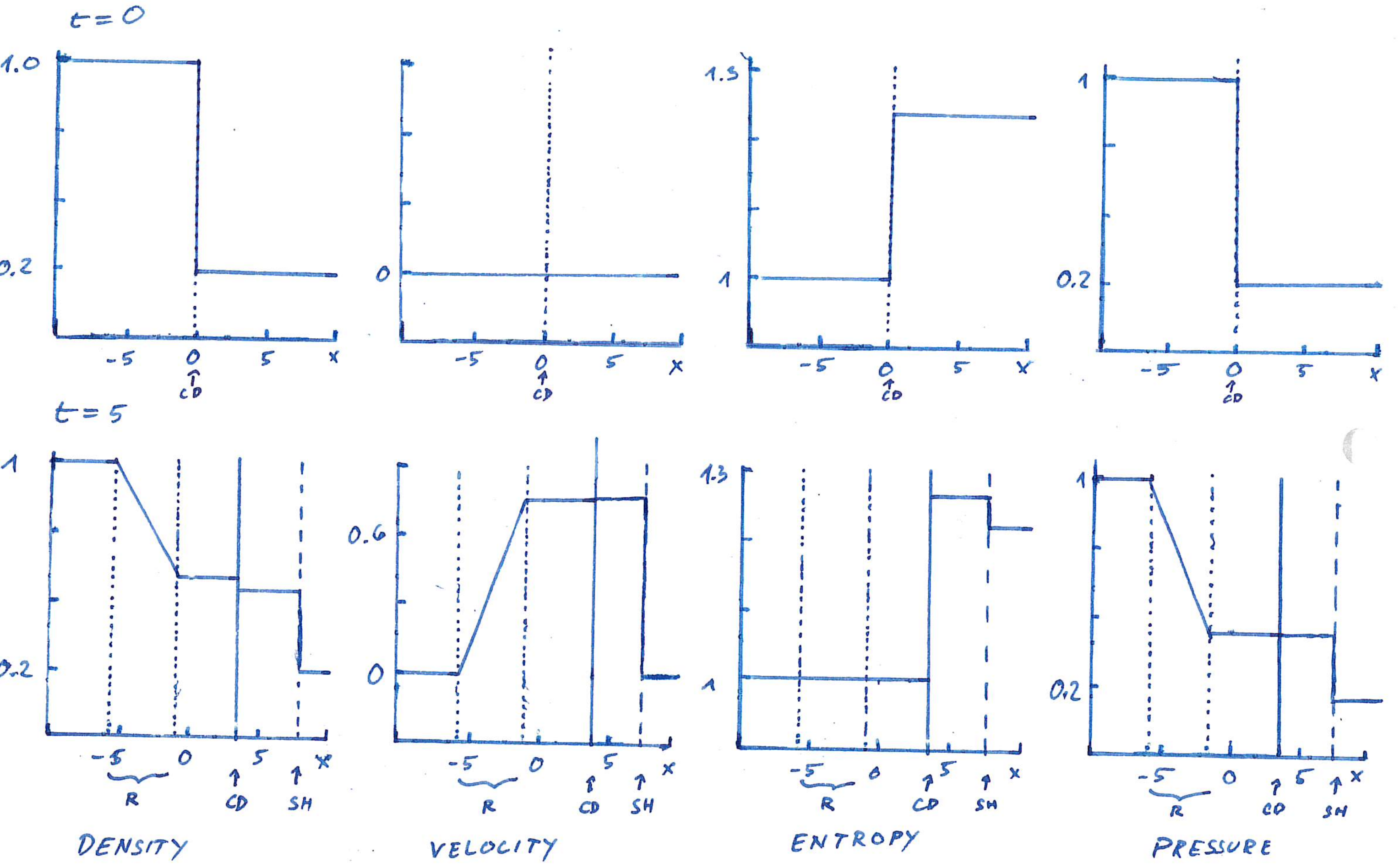


- The middle wave is always present and is a <u>contact wave</u> that marks the boundary between the original fluid phases from the left and right sides.
  The contact wave is sandwiched between a <u>shock</u> or a rarefaction wave on either side (any possible combination)
- The rarefaction wave is not a single characteristic, but rather a rarefaction fan with a beginning and an end.

These waves propagate with constant speed. If the solution is known at some time $t>0$, it can also be obtained at any other time through a suitable scaling transformation.

$\Rightarrow$ at $x=0$, the fluid quantities $(\rho^*, P^*, \vec{v}^*)$ are constant in time for $t>0$.

The corresponding physical situation is that of two gas phases, one cold and dense, one hot and rarefied for examples.
If $\vec{v}_R = \vec{v}_L = 0$, then it is called a "Sod Shock tube" problem.
Let's have a look at $t=0$ and $t=5$ for a problem:



- SHOCK: (SH)
  A sudden compression of the fluid, associated with an irreversible conversion of kinetic energy to heat, producing entropy. ρ, $\vec{v}$, P and ρ all change discontinuously at a shock. In astrophysics, shocks are often associated with supersonic motions of an object through a medium that causes a large disturbance.

- CONTACT DISCONTINUITY (CD)
  Traces the original separating plane between the two fluid phases that have been brought into contact. Pressure and velocity are constant across a contact, but ρ, s, T may jump.

- RAREFACTION WAVE (R)
  Occurs when gas suddenly expands. The rarefaction wave smoothly connects two states over a finite spatial region; there are no discontinuities in any fluid variables.

Finite Volume Methods (Godunov)

In finite volume methods one considers an integral form of the Euler equations. One starts from conservation laws. These are special cases of hyperbolic conservation laws of the type

$$\frac{\partial \vec{U}}{\partial t} + \nabla \cdot \vec{F} = 0$$

Here $\vec{U}$ is a state vector and $\vec{F}$ is the flux vector. The Euler equations for example can be written as

$$\vec{U} = (\rho, \rho\vec{v}, \rho e) \; ; \quad \vec{F} = (\rho\vec{v}, \rho\vec{v}\vec{v}^T + P, (\rho e + P)\vec{v})$$

with the specific energy $e = u + \vec{v}^2/2$ and u being the thermal energy per unit mass. The ideal gas equation $P = (\gamma - 1)\rho u$ gives the pressure and provides closure for the system.

We describe the state of the system as an average over a finite set of cells:

$$U_i = \frac{1}{V_i} \int_{Cell_i} U(\vec{x})\, dV$$

Now we need to find an update scheme for these cell-averaged quantities.

Start by integrating the conservation law over a cell and over a finite time interval:

$$\int_{x_{i-1/2}}^{x_{i+1/2}} dx \int_{t_n}^{t_{n+1}} dt \left(\frac{\partial \vec{U}}{\partial t} + \frac{\partial \vec{F}}{\partial x} \right) = 0$$

$$= \int_{x_{i-1/2}}^{x_{i+1/2}} dx \left[\vec{U}(t_{n+1}) - \vec{U}(t_n) \right] + \int_{t_n}^{t_{n+1}} dt \left[\vec{F}(x_{i+1/2}, t) - \vec{F}(x_{i-1/2}, t) \right)$$

$$= \Delta x \left[\vec{U}(t_{n+1}) - \vec{U}(t_n) \right] + \int_{t_n}^{t_{n+1}} dt \left[\vec{F}(x_{i+1/2}, t) - F(x_{i-1/2}, t) \right] = 0$$

$\vec{F}(x_{i+1/2}, t)$ is given by the solution of the Riemann problem with left state $\vec{U}_i^{(n)}$ and right state $\vec{U}_{i+1}^{(n)}$ for $t > t_n$, because at the interface, this solution is independent of time.

$$\Rightarrow \vec{F}(x_{i+1/2}, t) = \vec{F}^*_{i+1/2} = \vec{F}_{Riemann}(\vec{U}_i^{(n)}, U_{i+1}^{(n)})$$

With time independence, the integral reduces to

$$\Delta x \left[\vec{U}_i^{(n+1)} - \vec{U}_i^{(n)}\right] + \Delta t \left[F^*_{i+1/2} - F^*_{i-1/2}\right] = 0$$

$$\Rightarrow \vec{U}_i^{(n+1)} = \vec{U}_i^{(n)} + \frac{\Delta t}{\Delta x}\left[F^*_{i-\frac{1}{2}} - F^*_{i+1/2}\right]$$

These are called Godunov Schemes. It uses the Riemann Solution in the updating step.

We haven't made any approximations in the update formula. In principle, this should be valid for arbitrary large timesteps since the flux part is time independent.

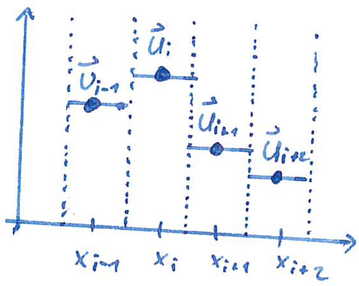
↳ We have implicitly treated the problem as the sum of two independent Riemann problems at all faces. This is reasonable if the cell spacing is such that waves do not propagate faster than the timestep Δt between cell phases. Or in other words, it's fine for timesteps smaller than the Courant timestep.

REA Scheme

The Reconstruct-Evolve-Average scheme describes a scheme based on the Godunov update scheme and is made of three steps:

1) Reconstruct

Construct fluxes using piece-wise constant states at cell interfaces (gives 1st order accurate).
Think of this as a discretisation of the fluid state at a given time.



2) Evolve

Evolve reconstructed state forward in time by Δt. In Godunov's approach, this is done by treating each cell interface as a piece-wise constant initial value problem which is solved with the Riemann solver exactly or approximately. As long as we respect the Courant condition, the solution can be treated as a superposition of the individual solutions.

3) Average

Compute a smooth average of the wave structure resulting from the evolution step to compute the new starting flow states $\vec{U}^{n+1}$. The cycle is then repeated for all timesteps.

The core of the method is the multiple Riemann problem approach. For ideal gas this can be solved exactly using an iterative root-finding method for a non-linear equation. For computational efficiency it is customary to use approximate solvers.

How can we extend the REA scheme to a <u>higher-order accuracy</u>?

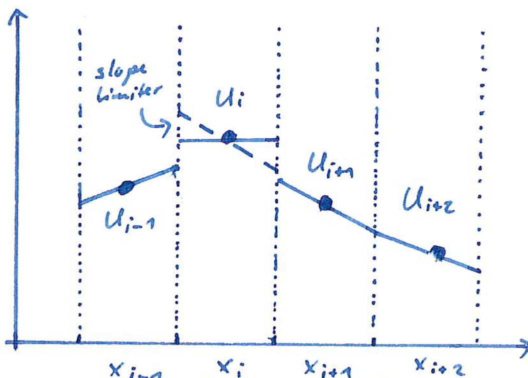
Let's define the error, namely the deviation from an exact solution, for the density of the flow:

$$L1 = \frac{1}{N} \sum_i |\rho_i - \rho(x_i)|$$

provided we know the exact solution $\rho(x)$.
The order of the numerical hydro method is then related to how fast the numerical solution $\underbrace{\text{converges}}_{(=\text{ rate of convergence})}$ to the exact solution with increasing resolution (= increasing N). A first order scheme converges as $1/N$, a second order scheme as $1/N^2$.

Higher order schemes (e.g. 4th order) are possible, but they become progressively more difficult to code and more expensive computationally. In general, going from first to second order is well motivated and computationally not prohibitive, so that should be the target.

The first step in constructing a 2nd order Godunov method is to replace the piece-wise constant with a piece-wise linear reconstruction: interpolate linearly between boundary values of cells, eliminating the "jump" at interfaces.



piece-wise linear reconstruction

This requires that one first estimates gradients for each cell (usually by a finite difference formula). These are then slope-limited if needed such that the linear extrapolations of the cell states to the cell interfaces do not introduce new extrema.
The slope-limiting procedure is quite important: it needs to be done to avoid that real fluid discontinuities introduce large spurious oscillations in the fluid.
Given slope limited gradients, for example $\nabla\rho$ for the density, one can then estimate the left and right states adjacent to an interface $x_{i+1/2}$ by spatial extrapolation from the cells left and right cells from the interface:

$$\rho^L_{i+1/2} = \rho_i + (\nabla\rho)_i \frac{\Delta x}{2} \qquad \rho^R_{i+1/2} = \rho_{i+1} - (\nabla\rho)_{i+1} \frac{\Delta x}{2}$$

The next step would in principle be to use these states in the Riemann solver. In doing this we will ignore the fact that our reconstruction has now a gradient over the cell; instead we still pretend that the fluid state can be taken as piece-wise constant left and right of the interface as far as the Riemann solver is concerned. However, it turns out that the spatial extrapolation needs to be augmented with a temporal extrapolation one half timestep into the future, such that the flux estimate is now effectively done in the middle of the timestep. This is necessary to reach second order accuracy in time and also for stability.

$$\Rightarrow \quad \rho^L_{i+\frac{1}{2}} = \rho_i + (\nabla\rho)_i \frac{\Delta x}{2} + \left(\frac{\partial\rho}{\partial t}\right)_i \frac{\Delta t}{2}$$

$$\rho^R_{i+1/2} = \rho_{i+1} - (\nabla\rho)_{i+1}\frac{\Delta x}{2} + \left(\frac{\partial\rho}{\partial t}\right)_{i+1}\frac{\Delta t}{2}$$

More generally, this has to be done for the whole state vector $\vec{u}$:

$$\Rightarrow \quad \vec{u}^{\,L}_{i+1/2} = \vec{u}_i + (\partial_x\vec{u})_i \frac{\Delta x}{2} + (\partial_t\vec{u})_i \frac{\Delta t}{2}$$

$$\vec{u}^{\,R}_{i+1/2} = \vec{u}_{i+1} - (\partial_x u)_{i+1}\frac{\Delta x}{2} + (\partial_t\vec{u})_{i+1}\frac{\Delta t}{2}$$

Note that the spatial derivatives $(\partial_x u)_i$ are slope limited. The evaluation of the $\overset{\text{time}}{}$ derivatives can be done by exploiting the Jacobian matrix of the Euler equations:

$$\partial_t\vec{u} = -\partial_x\vec{F}(\vec{u}) = -\frac{\partial F}{\partial \vec{u}}\,\partial_x\vec{u} = -\vec{A}(\vec{u})\,\partial_x\vec{u}$$

where $\vec{A}(\vec{u})$ is the Jacobian matrix.

$$\Rightarrow \quad (\partial_t\vec{u})_i = -\vec{A}(\vec{u}_i)\,(\partial_x\vec{u})_i$$

$$\Rightarrow \quad \vec{u}^{\,L}_{i+1/2} = \vec{u}_i + \left[\frac{\Delta x}{2} - \frac{\Delta t}{2}\vec{A}(\vec{u}_i)\right](\partial_x\vec{u})_i$$

$$\vec{u}^{\,R}_{i+1/2} = \vec{u}_{i+1} + \left[-\frac{\Delta x}{2} - \frac{\Delta t}{2}\vec{A}(\vec{u}_{i+1})\right](\partial_x\vec{u})_{i+1}$$

(MUSCL-Hancock scheme, 2nd order accurate extension of Godunov)

Higher order methods such as PPM (piece-wise parabolic method) use reconstruction at cell interfaces with higher order polynomials such as parabolic functions. ENO and WENO schemes are even higher order than PPM. They involve finite differencing between more cells to reconstruct values of cell interfaces (a bit like higher CIC vs. nearest grid point method). All these are conservative in the sense that when integrating over the cell, the values are conserved (e.g. mass as an integral of density).

Multidimensional Godunov Schemes

There are two major types of schemes to solve the 3D Euler equations with Godunov:

- <u>split schemes</u> reduce the 3D problem to sequential 1D problems
- <u>unsplit schemes</u> don't. All flux updates are applied simultaneously to a cell, not sequentially.

The Euler equations will be written as hyperbolic equations:

$$\partial_t \vec{u} + \partial_x \vec{F}(\vec{u}) = 0$$

In 3D, the PDEs describing a fluid become more involved. For an ideal gas:

$$\partial_t \begin{pmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ \rho e \end{pmatrix} + \partial_x \begin{pmatrix} \rho u \\ \rho u^2 + P \\ \rho u v \\ \rho u w \\ \rho u(\rho e + P) \end{pmatrix} + \partial_y \begin{pmatrix} \rho v \\ \rho u v \\ \rho v^2 + P \\ \rho w \\ \rho v(\rho e + P) \end{pmatrix} + \partial_z \begin{pmatrix} \rho w \\ \rho u w \\ \rho v w \\ \rho w^2 + P \\ \rho w(\rho e + P \end{pmatrix} = 0$$

where $e = e_{therm} + (u^2 + v^2 + w^2)/2$ is the total specific energy per unit mass, e_{therm} = thermal energy per unit mass, $P = (\gamma - 1)\rho e_{therm}$ is the pressure.

These equations are commonly written in the following notation:

$$\partial_t \vec{u} + \partial_x \vec{F} + \partial_y \vec{G} + \partial_z \vec{H} = 0$$

$\vec{F}(\vec{u})$, $\vec{G}(\vec{u})$, $\vec{H}(\vec{u})$ give the flux vectors in the x-, y- and z-direction, respectively.

A split scheme splits the problem in each dimension:

$$\partial_t \vec{U} + \partial_x \vec{F} = 0$$
$$\partial_t \vec{U} + \partial_y \vec{G} = 0$$
$$\partial_t \vec{U} + \partial_z \vec{H} = 0$$

The vectors appearing here still have the same dimensionality as in the full equations: They are augmented 1D-problems.
In practice, one solves the Riemann problem for one dimension at a time, updating one flux at a time, and then doing the full time update at the end. Each step to update in one direction is called a sweep.

In an unsplit approach, all flux updates of a cell are applied simultaneously, not sequentially.
In 2D for cartesian meshes:

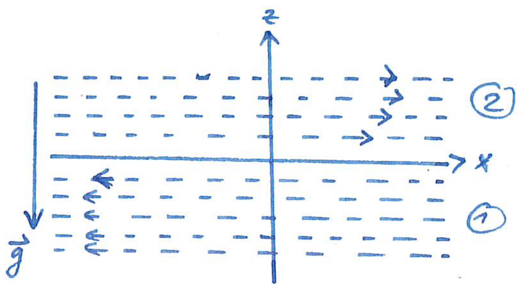
$$U_{i,j}^{(n+1)} = U_{i,j}^{n} + \frac{\Delta t}{\Delta x}\left(\vec{F}_{i-\frac{1}{2},j} - \vec{F}_{i+\frac{1}{2},j}\right) + \frac{\Delta t}{\Delta y}\left(\vec{G}_{i,j-1/2} - G_{i,j+1/2}\right)$$

Unsplit schemes are the choice for irregular meshes. For irregular meshes, one can use the divergence theorem to compute the state vector updates through a cell:

$$\vec{U}^{n+1} = \vec{U}^{n} - \frac{\Delta t}{V}\int \vec{F}\cdot d\vec{S}$$

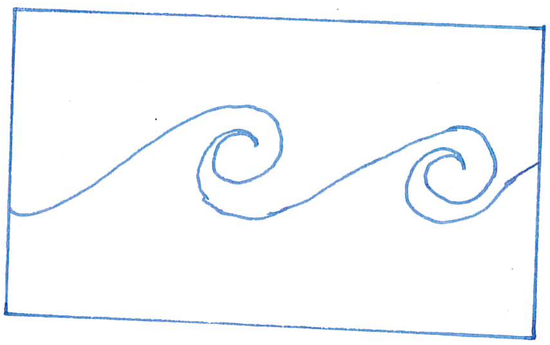
A testbed for numerical hydro: two-fluid instabilities

Consider two fluids shearing one past each other, having different densities. There may be an external force, such as gravity, and the relative velocity also may be null.



Simplest case: $\vec{v}_1 = \vec{v}_2$; $\rho_2 > \rho_1$. Analytically, this configuration is unstable. The fluids rearrange themselves and mix until the lighter fluid is above the heavier fluid. (Rayleigh-Taylor instability). This is a common instability in astrophysics.

If the two fluids have $\Delta\vec{v} > 0$, linear analysis shows that the configuration is unsable independent of the density difference. A sharp velocity gradient always causes the Kelvin-Helmholtz instability. Gravity may be mildly stabilizing, but can't suppress the instability.



Standard SPH fails to capture the KH instability due to large errors in the pressure at the contact discontinuity that prevents fluids from really being in contact.

Miscellaneous Notes

- The particle mesh technique introduces a second discretisation into the system. The first discretisation is by using superparticles, which discretises physics. The further introduction of a mesh discretises the physical model.

- Problems with mesh methods:
  - Meshes are usually rigid structures; They have difficulty to handle strongly dynamical cases
  - One might lose resolution if all particles end up in one cell

- Smoothing out the density field (by e.g. using CIC or TSC) loses some resolution, but adds more stability and is numerically favourable.

- The central difference formula introduces a smaller error ($O(h^4)$) than forward/backward differencing.

  To further increase accuracy: Use larger stencil

Excursus: Intro to Statistical Physics

To describe an N-body system, we need the generalised coordinates and momenta of each particle. They define the phase space Γ: $\quad (q, p) = \{q_1, \dots, q_{3N}, p_1, p_{3N}\} \in \Gamma$

As equal independent variables, the coordinates and momenta can be summarised in a phase or phase vector:

$$\pi = (\pi_1, \pi_2, \dots, \pi_{6N}) \equiv (q_1, q_2, \dots, q_{3N}, p_1, \dots, p_{3N})$$

Every microstate of the system corresponds to a specific phase point π of phase space. The phase trajectory is the set of all points $\pi = (\vec{q}, \vec{p})$ which the system takes on in time. With given initial values $\pi(t=0) = (\vec{q}(0), \vec{p}(0))$, these phase points are given uniquely by the Hamiltonian equations of motion:

$$\dot{p}_i = -\frac{\partial H}{\partial q_i} \, ; \qquad \dot{q}_i = \frac{\partial H}{\partial p_i}$$

Unfortunately, there are no analytical solutions for $N \geq 3$.

For a fixed point in time t, the state of a system is a single point in Γ: $\vec{x} = (q_1, q_2, \dots, q_{3N}, p_1, \dots, p_{3N})$

The system moves through Γ-space with the velocity

$$\begin{aligned} \dot{\vec{x}} &= (\dot{q}_1, \dots, \dot{q}_{3N}, \dot{p}_1, \dots, \dot{p}_{3N}) \\ &= \left(\frac{\partial H}{\partial p_1}, \dots, \frac{\partial H}{\partial p_{3N}}, -\frac{\partial H}{\partial q_1}, \dots, -\frac{\partial H}{\partial q_{3N}}\right) \end{aligned}$$

Using $\vec{\nabla} H = \left(\frac{\partial H}{\partial q_1}, \dots, \frac{\partial H}{\partial q_{3N}}, \frac{\partial H}{\partial p_1}, \dots, \frac{\partial H}{\partial p_{3N}}\right)$

we see that

$$\begin{aligned} \dot{\vec{x}} \cdot \vec{\nabla} H &= \frac{\partial H}{\partial p_1}\frac{\partial H}{\partial q_1} + \dots + \frac{\partial H}{\partial p_{3N}}\frac{\partial H}{\partial q_{3N}} - \frac{\partial H}{\partial p_1}\frac{\partial H}{\partial q_1} - \dots - \frac{\partial H}{\partial p_{3N}}\frac{\partial H}{\partial q_{3N}} \\ &= 0 \end{aligned}$$

The motion of the system is restricted to a hypersurface, which is defined by the gradient of $\mathcal{H}$. This is the surface normal of $\mathcal{H} = $ const. Since the "velocity" $\dot{\vec{x}}$ is always perpendicular to the normal vector ($\dot{\vec{x}} \cdot \vec{\nabla} H = 0$), it will always be tangential to the surface, thus can't escape the surface and move arbitrarily through phase space. This is the formulation of <u>energy conservation</u> in a high-dimensional phase space.

In dissipative systems, the surfaces of constant energy move and merge, but the relation $\dot{\vec{x}} \cdot \vec{\nabla} H = 0$ still holds, even for $H \neq$ const.

In this high-dimensional space we can introduce a <u>density function</u>:

$$\varrho(q, p, t)\, d^{3N}p\, d^{3N}q$$

gives the probability that the system is at time t in the infinitesimal phase space volume $d^{3N}p\, d^{3N}q$ around the point (q, p)

Since the state of the system is uniquely determined by $\pi(t)$, every observable (measurable quantity) of classical mechanics can be expressed as a phase space function:

$$F = F(\vec{q}, \vec{p}) = F(\pi, t)$$

that satisfies the equation of motion:

$$\frac{dF}{dt} = \{F, H\} + \frac{\partial F}{\partial t}$$

where $\{F, H\} = \sum_{j=1}^{3N} \left(\frac{\partial F}{\partial q_j} \frac{\partial H}{\partial p_j} - \frac{\partial F}{\partial p_j} \frac{\partial H}{\partial q_j} \right)$ are the Poisson braces.

We're not necessarily interested in the microscopical state when calculating or measuring a macroscopical state. Think for example about measuring the pressure of a gas, which microscopically comes from particles hitting the wall.
We get the macroscopic value by taking the time average:

$$\bar{F}^{t_0} = \frac{1}{t_0} \int_0^{t_0} F(\vec{q}, \vec{p})\, dt$$

However for a finite t_0, the time average will depend on the initial conditions of the microscopic state, which cannot be known. Therefore we postulate that at least the limit

$$\bar{F} = \lim_{t_0 \to \infty} \bar{F}^{t_0}$$

exists and is independent of the initial conditions.
This is a special formulation of the quasi-ergoden hypothesis:

A phase trajectory bound to the $H(\vec{q}, \vec{p}) = E$ -hypersurface in phase space comes ~~every~~ arbitrarily close to each point of this surface over time.

But an infinitely long duration of measurement is far from practical. Instead, we introduce statistical ensembles.
Statistical ensembles are a family of thought systems, each of which correspond completely to the imagined real system physically. Each member of the ensemble is in a microscopic state conceivable for the real system, that is agreeable with its boundary conditions and evolves through fitting evolution equations.
In every single moment, the entirety of the ensemble-systems embodies the entire evolution of the system in time, but only if the quasi-ergoden hypothesis is valid. In that case, it is possible to replace the time average by the ensemble average, which is instantaneous:

$$\text{time average} \overset{!}{=} \text{ensemble average}$$

Let $\varrho_e(\vec{q}, \vec{p}, t)$ be the probability density to find an ensemble member in the phase $\pi(\vec{q}, \vec{p})$ at time t. Assume ϱ_e is normed: $\int \varrho_e \, d^{3N}q \, d^{3N}p = 1$.

Then we can express the mean of any observable F as

$$\langle F \rangle_t = \int d^{3N}_q \, d^{3N}p \; F(\vec{q}, \vec{p}) \, \varrho_e(\vec{q}, \vec{p}, t)$$ <u>ensemble mean</u>

Similarly, using the density function $\varrho(q, p, t)$, [probability that the system is in the infinitesimal phase space volume $d^{3N}q \, d^{3N}p$ around $(\vec{q}, \vec{p})$ at time t] we can express the time average:

$$\bar{F}^{t_0} = \int d^{3N}q \, d^{3N}p \; \varrho(\vec{q}, \vec{p}, t_0) \, F(\vec{q}, \vec{p})$$ <u>time mean</u>

The time information of the system is given by ϱ here.

<u>Liouville equation</u>

Let $\varrho = \varrho_e$, the ensemble probability density for further calculations.

Using the phase-space velocity $\vec{v} = \dot{\vec{x}} = (\dot{q}_1, \dot{q}_2, \dots, \dot{q}_{3N}, \dot{p}_1, \dots, \dot{p}_{3N})$ we can define a <u>current density</u> of the phase points occupied by the ensemble-systems: $\vec{j} = \varrho \vec{v}$

(analoguely to the electric current density.)

Let G be a region in phase space with the surface $S(G)$.

Then $\int_{S(G)} d\vec{S} \cdot \vec{j} = \int_{S(G)} dS \cdot \vec{n}_S \cdot \vec{j}$ is the number of phase points flowing through the surface S per unit time.

Because there are no sources nor sinks for ensemble systems, this must be equal to the change in number of phase space points in the region G per unit time:

$$\int_{S(G)} d\vec{S} \cdot \vec{j} = -\frac{\partial}{\partial t} \int_G d^{3N}q \, d^{3N}p \; \varrho(\vec{q}, \vec{p}, t)$$

Using the Gauss theorem:

$$\int_{S(G)} d\vec{S}\cdot\vec{j} = \int_G d^{3N}q\, d^{3N}p\ \mathrm{div}(\vec{j}) = -\frac{\partial}{\partial t}\int d^{3N}q\, d^{3N}p\ \rho(\vec{q},\vec{p},t)$$

$$\Rightarrow \int_G d^{3N}q\, d^{3N}p\left[\frac{\partial}{\partial t}\rho(\vec{q},\vec{p},t) + \mathrm{div}(\vec{j})\right] = 0$$

$$\Rightarrow \frac{\partial}{\partial t}\rho(\vec{q},\vec{p},t) + \mathrm{div}(\vec{j}) = 0$$

Now using $\vec{j} = \rho\vec{v} = \rho(\dot{q}_1,\dots,\dot{p}_1,\dots)$

$$\vec{\nabla} = \left(\frac{\partial}{\partial q_1},\dots,\frac{\partial}{\partial p_1},\dots\right)$$

$$\mathrm{div}\,\vec{j} = \vec{\nabla}\cdot\vec{j} = \vec{\nabla}\cdot(\rho\vec{v}) =$$

$$= (\vec{\nabla}\rho)\cdot\vec{v} + \rho(\vec{\nabla}\cdot\vec{v})$$

For $\rho(\vec{\nabla}\cdot\vec{v})$:

$$\rho(\vec{\nabla}\cdot\vec{v}) = \rho\left[\frac{\partial\dot{q}^i}{\partial q_i} + \frac{\partial\dot{p}^j}{\partial p_j}\right] \quad \swarrow \text{Einstein sum convention} \qquad \Big|\ \dot{q}_i = \frac{\partial H}{\partial p_i},\ \dot{p}_i = -\frac{\partial H}{\partial q_i}$$

$$= \rho\left[\frac{\partial}{\partial q_i}\frac{\partial H}{\partial p^i} - \frac{\partial}{\partial p_j}\frac{\partial}{\partial q^j}\right]$$

$$= 0$$

$$\Rightarrow \mathrm{div}\,\vec{j} = (\vec{\nabla}\rho)\cdot\vec{v} + \rho(\vec{\nabla}\cdot\vec{v})$$

$$= (\vec{\nabla}\rho)\cdot\vec{v}$$

$$= \dot{q}_j\frac{\partial}{\partial q^j}\rho + \dot{p}_j\frac{\partial}{\partial p^j}\rho$$

$$= \left(\dot{q}_j\frac{\partial}{\partial q^j} + \dot{p}_j\frac{\partial}{\partial p^j}\right)\rho$$

$$\Rightarrow \boxed{\frac{d\rho}{dt} = \frac{\partial\rho}{\partial t} + \frac{\partial\rho}{\partial\vec{q}}\cdot\dot{\vec{q}} + \frac{\partial\rho}{\partial\vec{p}}\cdot\dot{\vec{p}} = 0}$$

$$\Rightarrow \rho(\vec{q}(t),\vec{p}(t),t) = \rho(\vec{q}(0),\vec{p}(0),0)$$

The "ensemble fluid" moves through phase space like an incompressible fluid.

A comoving observer sees a constant density.

Excursus: Finite Difference

Three forms are commonly considered:

1) Forward difference:

$$\Delta_h [f](x) = f(x+h) - f(x)$$

2) Backward difference:

$$\Delta_h [f](x) = f(x) - f(x-h)$$

3) Central difference:

$$\Delta_h [f](x) = f(x + \tfrac{1}{2}h) - f(x - \tfrac{1}{2}h)$$

The Taylor series expansion is given by

$$f(x) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x_0)}{n!} (x - x_0)^n$$

Let $x = x_0 + h \Rightarrow x - x_0 = h$

$$\Rightarrow f(x_0 + h) = \sum_{n=0}^{\infty} \frac{f^{(n)}(x_0)}{n!} h^n = f(x_0) + \frac{f'(x_0)}{1!} h + \frac{f''(x_0)}{2!} \cdot h^2 + O(h^3)$$

$$\Rightarrow f'(x_0) = \frac{f(x_0 + h) - f(x_0)}{h} + O(h)$$

for the forward difference.

Analogously for the backward difference:

Let $x = x_0 - h \Rightarrow x - x_0 = -h$

$$f(x) = f(x_0 - h) = f(x_0) + \frac{f'(x_0)}{1!}(-h) + O(h^2)$$

$$\Rightarrow \boxed{f'(x_0) = \frac{f(x_0) - f(x_0 - h)}{h} + O(h)}$$

To find the centered difference formula, we use:

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f'''(x_0)h^3 + \mathcal{O}(h^4)$$
$$f(x_0 - h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f'''(x_0)h^3 + \mathcal{O}(h^4)$$

This gives:

$$f(x_0+h) - f(x_0-h) = 2f'(x_0)h + \frac{1}{3}f'''(x_0)h^3 + \mathcal{O}(h^5)$$

$$\Rightarrow f'(x_0) = \frac{1}{2h}\left[f(x_0+h) - f(x_0-h) - \frac{1}{3}f'''(x_0)h^3 + \mathcal{O}(h^5)\right]$$

$$= \frac{f(x_0+h) - f(x_0-h)}{2h} - \frac{1}{6}f'''(x_0)h^2 + \mathcal{O}(h^4)$$

$\Rightarrow$ The centered difference formula introduces a smaller error! ($\mathcal{O}(h^2)$ vs $\mathcal{O}(h)$)

To find the centered difference formula for the second derivative, we again use

$$f(x_0 + h) = f(x_0) + f'(x_0)h + \frac{1}{2}f''(x_0)h^2 + \frac{1}{6}f'''(x_0)h^3 + \mathcal{O}(h^4)$$
$$f(x_0 - h) = f(x_0) - f'(x_0)h + \frac{1}{2}f''(x_0)h^2 - \frac{1}{6}f'''(x_0)h^3 + \mathcal{O}(h^4)$$

$$\Rightarrow f(x_0+h) + f(x_0-h) = 2f(x_0) + f''(x_0)h^2 + \mathcal{O}(h^4)$$

$$\Rightarrow \boxed{f''(x_0) = \frac{1}{h^2}\left[f(x_0+h) - 2f(x_0) + f(x_0-h)\right] + \mathcal{O}(h^2)}$$

Five - Point Stencil

Given a square grid, the five-point stencil of a point in the grid is a stencil made up of the point itself together with its four neighbours. It is used to write finite difference approximations to derivatives at grid points.

To obtain the formula, we again use the Taylor expansion:

$$f(x \pm h) = f(x) \pm h f'(x) + \frac{h^2}{2} f''(x) \pm \frac{h^3}{6} f'''(x) + O(h^4)$$
$$f(x \pm 2h) = f(x) \pm 2h f'(x) + \frac{4h^2}{2} f''(x) \pm \frac{8h^3}{6} f'''(x) + O(h^4)$$

Then:

$O(h^4)$ gets eaten in subtraction

$$f(x+h) - f(x-h) = 2h f'(x) + \frac{h^3}{3} f'''(x) + O(h^5) \qquad [1]$$
$$f(x+2h) - f(x-2h) = 4h f' + \frac{8h^3}{3} f'''(x) + O(h^5) \qquad [2]$$

To make the f''' term disappear, we calculate $8 \cdot [1] - [2]$:

$$8f(x+h) - 8f(x-h) - f(x+2h) + f(x-2h) =$$
$$= 16 h f' + \frac{8h^3}{3} f'''(x) + O(h^5) - 4hf' - \frac{8h^3}{3} f'''(x)$$
$$= 12 h f'$$

$$\Rightarrow \boxed{f' = \frac{1}{h} \left[\frac{2}{3} (f(x+h) - f(x-h)) - \frac{1}{12} (f(x+2h) - f(x-2h)) \right]}$$

$$+ O(h^4)$$