PY411 / 506
Computational Physics 2

Salvatore Rappoccio
• Now turn to fluid dynamics
  – Many particles together acting under conservation of mass and momentum
  – Simple cases are incompressible fluids without friction ("Newtonian fluids")

• Consider a volume $V$ inside a fluid. The mass $m$ inside that volume is the integral of the density $\rho$:

$$m = \int \rho \, dV$$
Fluid Dynamics

• The change in mass is determined by the rate of change at the surface

\[ \frac{d}{dt} \int \rho \, dV = - \int d\mathcal{S} \cdot \rho \vec{u} \]

• Then apply the divergence theorem:

\[ \int d\mathcal{S} \cdot \rho \vec{u} = \frac{d}{dt} \int \rho \, dV = \int dV = \vec{\nabla} \cdot \rho \vec{u} \]

• So we then obtain the continuity equation:

\[ \frac{\partial \rho}{\partial t} + \vec{\nabla} \cdot \rho \vec{u} = 0 \]
• Now we can apply Newton’s second law (conservation of momentum):

\[ \rho \frac{d\vec{u}}{dt} = \vec{F} \]

• and then we have:

\[ \frac{d\vec{u}}{dt} = \frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \vec{\nabla}) \vec{u} \]

Change in fluid velocity at a fixed point in space

Change in fluid velocity due to motion of fluid from neighboring points (Advection)
Fluid Dynamics

• What about friction?
  – Add force of gravity \( g \)

• Introduce viscous forces:
  – Dynamic viscosity coefficient \( \mu \)
  – Bulk viscosity coefficient \( \xi \)
  – Pressure \( p \)

• Then total force is:

\[
\vec{F} = \mu \nabla^2 \vec{u} + (\mu + \xi) \nabla (\nabla \cdot \vec{u})
\]

• Then we get the Navier-Stokes equation with dynamic viscosity \( \nu = \mu / \rho \)

\[
\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla) \vec{u} = \vec{g} - \frac{1}{\rho} \nabla p + \nu \nabla^2 \vec{u}
\]
Fluid Dynamics

- If we look at this in 1-d, this is:

\[
\frac{\partial u}{\partial t} + \left( u \frac{\partial u}{\partial x} \right) = g - \frac{1}{\rho} \frac{\partial p}{\partial t} + \nu \frac{\partial^2 u}{\partial x^2}
\]

- If \( g = 0 \) and pressure \( p = 0 \) (or gradient = 0), this is Burgers’ equation!

\[
\frac{\partial u}{\partial t} + \left( u \frac{\partial u}{\partial x} \right) = \nu \frac{\partial^2 u}{\partial x^2}
\]

- So shock propagation is a special case of the Navier-Stokes dynamics
Fluid Dynamics

• What about the full case?
  – It’s a $1M question

  – (No, really, it’s a Millennium Prize question).
  – http://www.claymath.org/millennium-problems
OK, well, what about simple-ish solutions?
– Even then, still pretty hard
– Commercial software in abundance:

– Open source examples:
  • https://people.sc.fsu.edu/~jburkardt/py_src/navier_stokes_2d_exact/navier_stokes_2d_exact.html
  • http://lorenabarba.com/blog/cfd-python-12-steps-to-navier-stokes/
Fluid Dynamics

• Let’s look at one special case of flow through a long tube with constant pressure, like the flow in a river.

Limmat river in Zurich, CH
Let’s play with some Jupyter now.

We will use the example from here:
- [http://lorenabarba.com/blog/cfd-python-12-steps-to-navier-stokes/](http://lorenabarba.com/blog/cfd-python-12-steps-to-navier-stokes/)

We will walk through it.
Fluid Dynamics

• The flow in the center is highest, and the flow at the edges is lowest:
Recall: General Relativity


- Relates gravity to the curvature of space-time!

- Objects with mass or energy distort space-time, and this induces a gravitational field
Recall: General Relativity

- Space-time is a tensor
- So gravity is a tensor
- Einstein’s equations:

\[ R_{\mu\nu} - \frac{1}{2} g_{\mu\nu} R = \frac{8\pi G_N}{c^4} T_{\mu\nu} - g_{\mu\nu} \Lambda \]

- Ricci tensor (gravitational force)
- Curvature scalar
- Stress-energy tensor (energy and momentum density of matter + radiation)
- Metric tensor
- Newton’s constant
- Speed of light
- Cosmological constant
Recall: General Relativity

• That’s a huge set of nonlinear partial differential equations, and can be arbitrarily complicated ($T_{\mu\nu}$ has no constraint to its format)

• A few simple cases can be derived:
  – If spacetime is homogeneous and isotropic, this is the Robertson-Walker metric:

$$ds^2 = \sum_{\mu=0}^{3} \sum_{\nu=0}^{3} g_{\mu\nu} dx^\mu dx^\nu = c^2 dt^2 - R^2(t) \left[ \frac{dr^2}{1 - kr^2} + r^2 (d\theta^2 + \sin^2 \theta d\phi^2) \right],$$

  – Assuming that the matter+radiation behave like a uniform perfect fluid with density $\rho$ and pressure $p$, this is the Friedmann-Lamaitre equations:

$$H^2 = \left( \frac{\dot{R}}{R} \right)^2 = \frac{8\pi G_N \rho}{3} - \frac{k c^2}{R^2} + \frac{\Lambda c^2}{3}, \quad \frac{\dot{R}}{R} = -\frac{4\pi G_N}{3} (\rho + 3p) + \frac{\Lambda c^2}{3}.$$

Hubble parameter: $H(t_0) = 72$ km/s/Mpc at present time
Recall: Hubble’s Law

- Hubble used this equation to determine a linear relationship:
  \[ rK + X \cos \alpha \cos \delta + Y \sin \alpha \cos \delta + Z \sin \delta = v, \]

- Plotting the data:

**Figure 1**: Velocity-Distance Relation among Extra-Galactic Nebulae.
General Relativity

- Now in a position to calculate some GR numerically instead of just analyzing the data

- Some approaches approximate matter as perfect fluid evolving according to GR
  - Morally equivalent equations to Navier-Stokes, so similar techniques can be used for solutions

- Applications we will consider:
  - Inflation
  - Gravitational collapse of supernovae into black holes
  - Greybody radiation from black holes
General Relativity

- Cosmic inflation
Go back to our equations:

\[ H^2 \equiv \left( \frac{\dot{R}}{R} \right)^2 = \frac{8\pi G_N \rho}{3} - \frac{k c^2}{R^2} + \frac{\Lambda c^2}{3}, \]

\[ \frac{\ddot{R}}{R} = -\frac{4\pi G_N}{3} (\rho + 3p) + \frac{\Lambda c^2}{3}. \]

Values of \( k \):
- \( k = 1 \): closed 3-sphere
- \( k = 0 \): flat
- \( k = -1 \): open 3-hyperboloid

Hubble parameter : \( H(t_0) = 72 \text{ km/s/Mpc} \) at present time
• Eliminate Lambda, use conservation of mass+energy, and combine the two equations, you get:

\[ \dot{\rho} = -3H(\rho + 3p) \]

• Further simplify the math by scaling mass density and pressure to include all of the constants in the system:

\[ H^2 = \left( \frac{\ddot{R}}{R} \right)^2 = \frac{8\pi G}{3} \rho - \frac{kc^2}{R^2} \]

\[ \dot{H} + H^2 = \left( \frac{\ddot{R}}{R} \right)^2 = -\frac{4\pi G}{3} \left( \rho + \frac{3p}{c^2} \right) \]
General Relativity

- If we consider a perfect fluid, the pressure is linearly dependent on the density, so you get $p = w \rho$

- If space is completely flat, and we set $c=1$, we have $k = 0$, so

$$H^2 = \left( \frac{\dot{R}}{R} \right)^2 = \frac{8\pi G}{3} \rho$$

$$\dot{H} + H^2 = \frac{\ddot{R}}{R} = -\frac{4\pi G}{3} (1 + 3w) \rho$$

- You will solve this for your homework problem!
N-body simulations

• Consider N interacting particles with long range force acting between them (i.e. gravity)
  – Structure formation of galaxies, stars, superclusters, etc

• Recall our previous attempts at the 3-body problem
  – Will now extend to N bodies

Sloan Digital Sky Survey (NASA)
N-body simulations

• Long-range interaction very inconvenient here
• Force goes like
  \[ F \sim \frac{1}{r^2} \]
• But surface area goes like
  \[ A \sim 4\pi r^2 \]
• Thus the number of particles at distance \( r \) times the strength of the force is basically constant
  –Cannot truncate!
N-body simulations

- If we calculate the forces between N bodies, we therefore need to compute \( N(N-1)/2 \) forces each iteration.

\[
\vec{F}_i = \sum_j \vec{F}_{ij} = - \sum_j \frac{Gm_i m_j}{r^3} \hat{r}
\]
N-body simulations

• Try the next level of sophistication in our ODE solvers
  – Currently evolve position and velocity (according to velocity and acceleration, respectively)

  – How about evolving acceleration!? 

  – Use the JERK!
N-body simulations

• Try the next level of sophistication in our ODE solvers
  – Currently evolve position and velocity (according to velocity and acceleration, respectively)

  – How about evolving acceleration!? 

  – Use the JERK!

\[ \vec{J}_{ij} = \ddot{a}_{ij} = \frac{M_i}{r_{ij}^3} \left[ \vec{v}_{ij} - 3 \frac{\vec{v}_{ij} \cdot \vec{r}_{ij}}{r_{ij}^2} \vec{r}_{ij} \right] \]
N-body simulations

• First let’s look at simple systems where we actually solve the ODEs for all particles
• Going to use third-party software:
  – https://www.ids.ias.edu/~piet/act/comp/algorithms/starter
  – Instructions:
    – https://github.com/rappoccio/PHY410/blob/master/Lecture35/README.md

• Can’t animate and solve the ODEs in real time for any large-ish number of N (like, 30).

Works, but computationally intractable!
N-body simulations

- Code uses a “predictor-corrector” algorithm:
  - Predicts position and velocity at next time steps

\[ \vec{r}_p = \vec{r} + \vec{v}\delta t + \frac{1}{2}\ddot{r}\delta t^2 + \frac{1}{6}\dddot{r}\delta t^3 \]

\[ \vec{v}_p = \vec{v} + \ddot{v}\delta t + \frac{1}{2}\dddot{v}\delta t^2 \]

- Computes acceleration and jerk for those predictions using Taylor series

\[ \vec{k} \equiv \frac{1}{2}\dddot{a}\delta t^2 = 2(\dddot{a} - \dddot{a}_p) + \delta t(\dddot{J} - \dddot{J}_p) \]

\[ \vec{l} \equiv \frac{1}{2}\ddddot{a}\delta t^3 = -3(\dddot{a} - \dddot{a}_p) - \delta t(2\dddot{J} - \dddot{J}_p) \]
N-body simulation

• Then corrects position and velocity with the acceleration and jerk:

\[
\vec{r}_c = \vec{r}_p + \left( \frac{1}{12} \vec{k} + \frac{1}{20} \vec{l} \right) \delta t^2
\]

\[
\vec{v}_c = \vec{v}_p + \left( \frac{1}{3} \vec{k} + \frac{1}{4} \vec{l} \right) \delta t
\]

• Their code also calculates the “least collision time”, i.e. the smallest time between interactions of two objects.
  – Used to adjust the time step, similar to ARK4
N-body simulations

- We need some kind of approximation then:
  - Particle mesh
    - Create a 3-d lattice, approximate forces from them
    - Good for uniform configurations
    - $O(M \log(M))$ ($M = \text{num grid points}$)
  - Trees
    - Partition space into a hierarchy of cubes
    - Compute particle-particle interactions for close interactions
    - Compute particle-cell or cell-cell interactions for far interactions
    - $O(N \log(N))$ ($N = \text{num particles}$)

Figures from
N-body simulations

• Fast multipole methods
  – Expands Green’s function in a multipole
  – $O(N)$ ($N =$ num particles)
• Fluid dynamic approximations
  – Approximate by PDEs, not individual particles
  – Only applicable in certain situations
N-body simulations

• Lots of codes out there!
  – AREPO:
    • http://wwwmpa.mpa-garching.mpg.de/~volker/arepo/
  – GADGET:
    • http://wwwmpa.mpa-garching.mpg.de/gadget/

• Example of simulation from GADGET from a former Comp. Phys. student (Leigh Korbel) for his Master’s project