# 1 Day 1 - Introduction to the transport theory

## 1.1 Units

- Speed of light: $c = 2.99762458 \times 10^8\,\mathrm{m}/s$
- Planck constant: $\hbar c = 197.327\,\mathrm{MeV\,fm}$
- Boltzmann constant: $k_B = (1/11604.52)\mathrm{eV/K}$
- Natural unit: $\hbar = c = k_B = 1$
- Measure everything with either MeV or fm
- For rough estimates

$$\begin{aligned} c &\approx 3 \times 10^8\,\mathrm{m}/s \\ 200\,\mathrm{MeV} &\approx 1/\mathrm{fm} \\ 20\,^\circ\mathrm{C} &\approx (1/40)\,\mathrm{eV} \end{aligned} \tag{1}$$

## 1.2 Uncertainty relations

- $\Delta x \Delta p \geq \hbar/2 \to 1/2$
  - Interpretation 1: If a particle is confined within $\Delta x$, the momentum is uncertain by $1/\Delta x$. Same goes for the momentum uncertainty.
  - Interpretation 2: When a particle state is created (via a scatterings, for instance), its momentum is uncertain until enough distance is traversed
- $\Delta t \Delta E \geq 1/2$
  - Interpretation 1: When a particle state is created (vis a scattering, for instance), its energy is uncertain until enough time is passed
  - Interpretation 2: A state (particle) that has a finite life time *cannot* have a well defined energy (mass). If the life time is $\tau$, the energy uncertainty is $1/\tau$.
  - Interpretation 3: An off-shell state with the invariant mass $Q$ lasts about $1/Q$ before it starts to radiate

## 1.3 1 and 2 body mechanics

Transport theory is the theory of many body particles. It is a semi-classical approach to solve many body dynamics. In physics, one can solve one body problem

$$\mathbf{F}(\mathbf{x},t) = \frac{d\mathbf{p}}{dt} \tag{2}$$

to get $\mathbf{x}(t), \mathbf{p}(t)$ or

$$i\partial_t \psi(t,\mathbf{x}) = \hat{H}\psi(t,\mathbf{x}) \tag{3}$$

where

$$\hat{H} = \frac{\hat{\mathbf{p}}^2}{2m} + V(\mathbf{x},t)) \tag{4}$$

to get $\psi(t,\mathbf{x})$. Many such problems admit exact solutions. But most of such problems do not admit an analytic solution. But I just told you that they can be solve. So what do I mean by that? I meant "by any means necessary" – analytical or numerical. When the force depends on on the distance between the two particles, two body problems can be also solved by changing coordinates to the center of mass coordinate and the relative coordinate. For instance, suppose we have the Hamiltonian.

$$H = \frac{\mathbf{p}_1^2}{2m_1} + \frac{\mathbf{p}_2^2}{2m_2} + V(\mathbf{x}_1 - \mathbf{x}_2, t) \tag{5}$$

Let's see how we transform this to the CM frame and how the dynamics is affected. We'll do this here in some detail because we will go to the center of mass system and then come back to the lab frame all the time when we do transport calculations, that is, scatterings.

The total momentum of two interacting particle is

$$\mathbf{P} = \mathbf{p}_1 + \mathbf{p}_2 \tag{6}$$

The total mass is

$$M = m_1 + m_2 \tag{7}$$

Hence, the system as a whole is moving with the speed

$$\mathbf{V}_{\rm cm} = \frac{\mathbf{P}}{M} = \frac{\mathbf{p}_1 + \mathbf{p}_2}{m_1 + m_2} \tag{8}$$

Suppose we move to the system where the system as a whole is at rest, that is $\mathbf{P}_{\rm cm} = 0$. How does the Hamiltonian look like? Well, this is easy to achieve just subtract $\mathbf{V}_{\rm cm}$ from all the velocities:

$$\begin{aligned} \mathbf{p}_1 &= \mathbf{p}_{1,\rm cm} + m_1 \mathbf{V}_{\rm cm} \\ \mathbf{p}_2 &= \mathbf{p}_{2,\rm cm} + m_2 \mathbf{V}_{\rm cm} \end{aligned} \tag{9}$$

Since the total momentum is

$$\mathbf{p}_1 + \mathbf{p}_2 = (m_1 + m_2)\mathbf{V}_{\rm cm} \tag{10}$$

we have an important condition that in the CM

$$\mathbf{p}_{1,\rm cm} + \mathbf{p}_{2,\rm cm} = 0 \tag{11}$$

To transform the kinetic energy, square the momentum to get

$$\begin{aligned} \mathbf{p}_1^2 &= (\mathbf{p}_{1,\rm cm} + m_1 \mathbf{V}_{1,\rm cm})^2 \\ &= \mathbf{p}_{1,\rm cm}^2 + 2m_1 \mathbf{p}_{1,\rm cm} \cdot \mathbf{V}_{\rm cm} + m_1^2 \mathbf{V}_{\rm cm}^2 \end{aligned} \tag{12}$$

Hence the kinetic energy part becomes

$$\begin{aligned} K &= \frac{\mathbf{p}_1^2}{2m_1} + \frac{\mathbf{p}_2^2}{2m_2} + V(\mathbf{x}_1 - \mathbf{x}_2, t) \\ &= \frac{\mathbf{p}_{1,\rm cm}^2}{2m_1} + \frac{\mathbf{p}_{2,\rm cm}^2}{2m_2} + 2(\mathbf{p}_{1,\rm cm} + \mathbf{p}_{2,\rm cm}) \cdot \mathbf{V}_{\rm cm} + \frac{1}{2}(m_1 + m_2)\mathbf{V}_{\rm cm}^2 \end{aligned} \tag{13}$$

The 3rd term vanishes since $\mathbf{p}_{1,\rm cm} + \mathbf{p}_{2,\rm cm} = 0$. The last term is simply

$$\frac{1}{2}(m_1 + m_2)\mathbf{V}_{\rm cm}^2 = \frac{1}{2}M(\mathbf{P}/M)^2 = \frac{\mathbf{P}^2}{2M} \tag{14}$$

is the CM kinetic energy. The first two terms are a bit tricky. Since $\mathbf{p}_{1,\rm cm} + \mathbf{p}_{2,\rm cm} = 0$,

$$\mathbf{p}_{1,\rm cm}^2 = \mathbf{p}_{2,\rm cm}^2 = \mathbf{p}_r^2 \tag{15}$$

Then

$$\begin{aligned} \frac{\mathbf{p}_{1,\rm cm}^2}{2m_1} + \frac{\mathbf{p}_{2,\rm cm}^2}{2m_2} &= \frac{\mathbf{p}_r^2}{2}\left(\frac{1}{m_1} + \frac{1}{m_2}\right) \\ &= \frac{\mathbf{p}_r^2}{2\mu} \end{aligned} \tag{16}$$

where

$$\mu = \frac{m_1 m_2}{m_1 + m_2} \tag{17}$$

So finally,

$$H = \frac{\mathbf{P}^2}{2M} + \frac{\mathbf{p}_r^2}{2\mu} + V(\mathbf{x}_1 - \mathbf{x}_2, t) \tag{18}$$

What about the coordinates? For the total momentum, the speed is

$$\begin{aligned} \mathbf{V}_{\text{cm}} &= \mathbf{P}/M \\ &= \frac{\mathbf{p}_1 + \mathbf{p}_2}{m_1 + m_2} \\ &= \frac{m_1\dot{\mathbf{x}}_1 + m_2\dot{\mathbf{x}}_2}{m_1 + m_2} \\ &= \frac{d}{dt}\mathbf{R} \end{aligned} \tag{19}$$

where

$$\mathbf{R} = \frac{m_1\mathbf{x}_1 + m_2\mathbf{x}_2}{m_1 + m_2} \tag{20}$$

is the CM coordinate. For the second term, the velocity is

$$\begin{aligned} \dot{\mathbf{v}}_r &= \frac{\mathbf{p}_r}{\mu} \\ &= \frac{\mathbf{p}_1 - m_1\mathbf{V}_{\text{cm}}}{\mu} \\ &= \frac{1}{\mu}\left(m_1\dot{\mathbf{x}}_1 - m_1\frac{m_1\dot{\mathbf{x}}_1 + m_2\dot{\mathbf{x}}_2}{m_1 + m_2}\right) \\ &= \frac{1}{\mu}\frac{m_1 m_2(\dot{\mathbf{x}}_1 - \dot{\mathbf{x}}_2)}{m_1 + m_2} \\ &= \frac{d}{dt}\mathbf{r} \end{aligned} \tag{21}$$

where we define $\mathbf{r} = \mathbf{1} - \mathbf{2}$. Now since the potential only depend on the difference $\mathbf{x}_1 - \mathbf{x}_2$ the forces are only between the two particles. Hence, there is no external force acting on the system. Hence $\mathbf{P}$ is a constant of motion.

That is, once we know $\mathbf{P}$, equivalently, $\mathbf{V}_{\rm cm}$, we can go to the CM frame to do an easier calculations and then go back to the lab frame. The equations of motion are

$$\begin{aligned}
\dot{\mathbf{R}} &= \frac{\mathbf{P}}{M} \\
\dot{\mathbf{P}} &= 0 \\
\dot{\mathbf{r}} &= \frac{\mathbf{p}_r}{\mu} \\
\dot{\mathbf{p}}_r &= -\nabla V(\mathbf{r}, t)
\end{aligned} \tag{22}$$

In this way, all 2-body problems with no external force reduce to 1-body problem.

Now go back to the N Newton's equation of motion $\mathbf{F} = m\mathbf{a}$ for the 1-body problem It is a second order differential equation which can be recast in to two first order ones

$$\begin{aligned}
\frac{dx^i}{dt} &= p^i/m \\
\frac{dp^i}{dt} &= F^i
\end{aligned} \tag{23}$$

You have learned how to solve this problem analytically in your Classical mechanics courses. For instance, the Kepler motion or simple harmonic motion. That's when $F^i$ is simple. In general, this needs to be solve numerically.

OK. So let's think about solving this numerically. We'll do it in 1-D, but generalization is immediate. The easiest way is to use the definition of derivative

$$\frac{df}{dt} = \frac{f(t+h) - f(t)}{h} + O(h) \tag{24}$$

So the Newton's equation becomes (we'll just do 1D)

$$\begin{aligned}
\frac{x[n+1] - x[n]}{h} &= p[?]/m + O(h) \\
\frac{p[n+1] - p[n]}{h} &= F[?] + O(h)
\end{aligned} \tag{25}$$

where

$$x[n] = x(t_0 + nh) \tag{26}$$

is the position at time $t_n = t_0 + nh$. But when should right hand side be evaluated? Well, as long as we are making $O(h)$ error, we can use either $p[n], F[n]$ or $p[n+1], F[n+1]$ or any time that is $O(h)$ away from $t_n$. Then ignoring $O(h)$ errors, the simplest algorithm is the **Forward Euler algorithm**

---

$$\begin{aligned} x[n+1] &= x[n] + (p[n]/m)h \\ p[n+1] &= p[n] + F[n]h \end{aligned} \tag{27}$$

---

This is simple because the new value is calculated only with the old values. Unfortunately, this doesn't work well. To see that, consider the SHO.

$$\ddot{x} = -\omega^2 x \tag{28}$$

This becomes

$$\begin{aligned} x[n+1] &= x[n] + h(p[n]/m) \\ p[n+1] &= p[n] - hkx[n] \end{aligned} \tag{29}$$

We know the exact solutions are $e^{\pm i\omega t}$ where $\omega^2 = k/m$. So at $t = t_n$, the exact solution is (choosing $e^{i\omega t}$),

$$x_{\text{exact}}[n] = \left(e^{i\omega h}\right)^n \tag{30}$$

$$p_{\text{exact}}[n] = im\omega \left(e^{i\omega h}\right)^n \tag{31}$$

So let's try

$$x[n] = \xi^n \tag{32}$$

$$p[n] = im\omega\xi^n \tag{33}$$

The discrete equations become

$$\xi = 1 + ih\omega \tag{34}$$

The magnitude of this expression is

$$|\xi| = \sqrt{1 + (h\omega)^2} > 1 \tag{35}$$

Hence as time grows, $x[n]$ will grow indefinitely while the exact solution is bound by 1. This is, of course, bad. This means that the energy is not conserve. It will grow indefinitely.

What if we try the **Backward Euler**?

---

$$\begin{aligned} x[n+1] &= x[n] + (p[n+1]/m)h \\ p[n+1] &= p[n] + F[n+1]h \end{aligned} \tag{36}$$

---

Using the same ansatz for the SHO, one can now show that

$$\xi = \frac{1}{1 - i\omega h} \tag{37}$$

whose magnitude is

$$|\xi| = \frac{1}{\sqrt{1 + (\omega h)^2}} \tag{38}$$

This is good in the sense that this is bound. But this is also bad because the energy is not constant. It will eventually go to zero.

OK. Both are bad. But wait. there is one more trick we can try. Note that

$$\frac{f(t+h) - f(t)}{h} = \dot{f}(t) + O(h) \tag{39}$$

but the same expression is

$$\frac{f(t+h) - f(t)}{h} = \dot{f}(t + h/2) + O(h^2) \tag{40}$$

So if we evaluate the right hand side at $t_{n+1/2} = t_0 + (n + 1/2)h$, then we actually have a second order accurate method! So suppose we use the **Half-point method**

---

$$\begin{aligned} x[n+1] &= x[n] + h\frac{p[n+1/2]}{m} \\ p[n+1] &= p[n] + hF[n+1/2] \end{aligned} \tag{41}$$

---

Again, let $x[n] = \xi^n, p[n] = im\omega\xi^n$ to get

$$\xi = 1 + i\omega h\xi^{1/2} \tag{42}$$

which has the solution

$$\xi^{1/2} = \frac{ih\omega \pm \sqrt{4-(h\omega)^2}}{2} \tag{43}$$

One can easily see that

$$|\xi^{1/2}| = \frac{(h\omega)^2 + (4-(h\omega)^2)}{4} = 1 \tag{44}$$

That's good! The trouble, however, is that we need to evaluate the right hand side at the half-point.

To do better, Recall from the Freshman physics courses that when the acceleration, equivalently the force, is constant, the solution is

$$x(t) = x(0) + v_0 t + \frac{a}{2}t^2 \tag{45}$$

Now consider going from $t_n = t_0 + nh$ to $t_{n+1}$. Then as long as $h$ is small,

$$x[n+1] = x[n] + v[n]h + a[n]\frac{h^2}{2} + O(h^3) \tag{46}$$

For the momentum, the exact expression is

$$p[n+1] = p[n] + \int_{t_n}^{t_{n+1}} dt\, F(t) \tag{47}$$

Using the Trapezoid rule, we can approximate

$$p[n+1] = p[n] + h\frac{F[n]+F[n+1]}{2} + O(h^3) \tag{48}$$

Hence, as long as $F(x,t)$ does not depend on $p$, this completes our algorithm. Namely, the **Velocity Verlet Method**

---

$$\begin{aligned} x[n+1] &= x[n] + h(p[n]/m) + (h^2/2)(F[n]/m) \\ p[n+1] &= p[n] + h\frac{F[n]+F[n+1]}{2} \end{aligned} \tag{49}$$

---

This method is known to conserve energy.

If the force depends on the momentum then things get a bit complicated, but fortunately, we won't have to deal with that.

OK. So far so good. But as long as we have more than 2 particles, trouble starts. The system of 3 particles, for instance, cannot be solve analytically. Furthermore, such system is known to be chaotic meaning that a small error can exponentially grow. Hence, no matter how accurate your numerical method is, it is bound to deviate from the solution if you run it long enough.

## 1.4 The Vlasov equation

Now if you have many particles, however, things get a bit easier again since you can use statistical ideas. The simplest equation of motion for many body system is

$$\begin{aligned} \dot{\mathbf{x}}_i &= \mathbf{p}_i/m_i \\ \dot{\mathbf{p}}_i &= \mathbf{F}_{\text{ext}}(\mathbf{x}_i) \end{aligned} \tag{50}$$

where $\mathbf{F}_{\text{ext}}(\mathbf{x})$ is the external force as opposed to the internal force between the particles. Now define the *phase space density*

$$f(t,\mathbf{x},\mathbf{p}) = \sum_{i=1}^{N} \delta(\mathbf{x}-\mathbf{x}_i(t))(2\pi)^3\delta(\mathbf{p}-\mathbf{p}_i(t)) \tag{51}$$

then

$$\begin{aligned} \partial_t f(t,\mathbf{x},\mathbf{p}) \;=\; & \sum_{i=1}^{N} \left(\partial_t \delta(\mathbf{x}-\mathbf{x}_i(t))\right)(2\pi)^3\delta(\mathbf{p}-\mathbf{p}_i(t)) \\ & + \sum_{i=1}^{N} \delta(\mathbf{x}-\mathbf{x}_i(t))(2\pi)^3 \left(\partial_t \delta(\mathbf{p}-\mathbf{p}_i(t))\right) \end{aligned} \tag{52}$$

Noting

$$\partial_t \delta(\mathbf{x}-\mathbf{x}_i(t)) = (-\dot{\mathbf{x}}_i(t))\cdot\nabla_x \delta(\mathbf{x}-\mathbf{x}_i(t)) \tag{53}$$

and similarly for $\delta(\mathbf{p}-\mathbf{p}_i)$,

$$\begin{aligned} \partial_t f(t,\mathbf{x},\mathbf{p}) \;=\; & \sum_{i=1}^{N} \left((-\dot{\mathbf{x}}_i\cdot\nabla_x \delta(\mathbf{x}-\mathbf{x}_i(t))\right)(2\pi)^3\delta(\mathbf{p}-\mathbf{p}_i(t)) \\ & + \sum_{i=1}^{N} \delta(\mathbf{x}-\mathbf{x}_i(t))(2\pi)^3 \left(-\dot{\mathbf{p}}_i\cdot\nabla_p \delta(\mathbf{p}-\mathbf{p}_i(t))\right) \\ =\; & -(\mathbf{p}/m)\cdot\nabla_x f(t,\mathbf{x},\mathbf{p}) - \mathbf{F}_{\text{ext}}(\mathbf{x})\cdot\nabla_p f(t,\mathbf{x},\mathbf{p}) \end{aligned} \tag{54}$$

the resulting equation

$$\partial_t f(t,\mathbf{x},\mathbf{p}) + (\mathbf{p}/m)\cdot\nabla_x f(t,\mathbf{x},\mathbf{p}) + \mathbf{F}_{\text{ext}}(\mathbf{x})\cdot\nabla_p f(t,\mathbf{x},\mathbf{p}) = 0 \tag{55}$$

is known as the Vlasov equation.

Now what if the system only has internal forces? Can we still apply this formula? Well, we can't apply it exactly since $\dot{\mathbf{p}}_i$'s are not the same any more. Now, trace back why this is so. This is so because when only the internal interactions exist. Consider again the simplest case: The potential between each particle is the same $V(\mathbf{x}_i - \mathbf{x}_j) = V(\mathbf{x}_j - \mathbf{x}_i)$. The equation of motion is

$$\begin{aligned}
\dot{\mathbf{p}}_i &= -\nabla_{\mathbf{x}_i} \sum_{j \neq i} V(\mathbf{x}_i - \mathbf{x}_j) \\
&= -\nabla_{\mathbf{x}_i} \int d^3y \, V(\mathbf{x}_i - \mathbf{y}) \sum_{j \neq i} \delta(\mathbf{y} - \mathbf{x}_j) \\
&= -\nabla_{\mathbf{x}_i} \int d^3y \, V(\mathbf{x}_i - \mathbf{y}) \left( \int \frac{d^3q}{(2\pi)^3} f(t, \mathbf{y}, \mathbf{q}) - \delta(\mathbf{y} - \mathbf{x}_i) \right) \qquad (56)
\end{aligned}$$

Now in the case of many particles, the single $\delta$-function term can be ignored and we can define the average total force

$$\langle \mathbf{F}(\mathbf{x}) \rangle = -\nabla_x \int d^3y \frac{d^3q}{(2\pi)^3} V(\mathbf{x} - \mathbf{y}) f(t, \mathbf{y}, \mathbf{q}) \qquad (57)$$

Using this,

$$\begin{aligned}
\partial_t f &= -(\mathbf{p}/m) \cdot \nabla_x f + \sum_{i=1}^{N} \delta(\mathbf{x} - \mathbf{x}_i(t)) (2\pi)^3 \left( -\dot{\mathbf{p}}_i \cdot \nabla_p \delta(\mathbf{p} - \mathbf{p}_i(t)) \right) \\
&= -(\mathbf{p}/m) \cdot \nabla_x f - \langle \mathbf{F}(\mathbf{x}) \rangle \cdot \nabla_f f \qquad (58)
\end{aligned}$$

to finally get

$$\partial_t f(t, \mathbf{x}, \mathbf{p}) + (\mathbf{p}/m) \cdot \nabla_x f(t, \mathbf{x}, \mathbf{p}) + \langle \mathbf{F}(\mathbf{x}) \rangle \cdot \nabla_p f(t, \mathbf{x}, \mathbf{p}) = 0 \qquad (59)$$

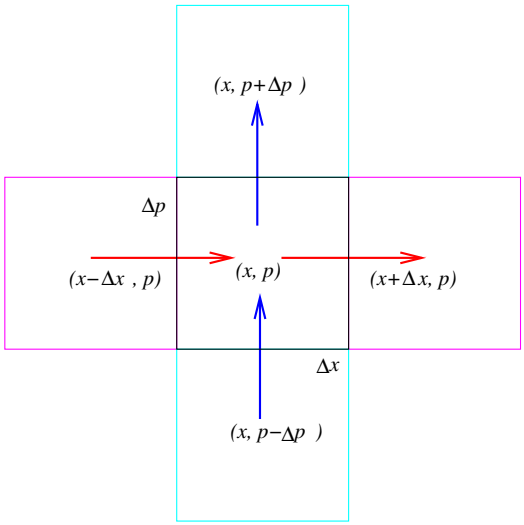
These two equations constitute the "mean-field approximation".

Are we done, then? Not really. Mean-field approximation is nice, but it still assumes some un-physical things. The most important one is our assumption of point particles. Real particles are not point particles. They have a size.

The size of a particle is encoded in the "hard core" part of the potential energy. Think about electrically charge billiard balls. If none of them are

close by, then the system can be regarded as a system of point particles with Coulomb interactions. Then one can use the mean-field approximation. However, as soon as two particles are in contact, the notion of "average force" fails because they must now scatter and that event is independent of what other parts of the system is doing.

The mean-field equation should be then modified. How does one modify it? Well, let's have a look at the concept of the phase space density again.




Since

$$\int d^3x \int \frac{d^3p}{(2\pi)^3} f(t, \mathbf{x}, \mathbf{p}) = N \tag{60}$$

one can say that $f(t, \mathbf{x}, \mathbf{p})$ is the number of particles per unit spatial volume and per unit momentum volume at time $t$.

Now consider a small cell in the phase space shown above. How can the number of particles in the cell change? Well, two ways so far. One, non-zero velocity will move particles from one $\mathbf{x}$ position to its neighbor. Two, non-zero acceleration (force) will move particles from one $\mathbf{p}$ to its neighbor. Now we add the third way: Two particles in a cell can undergo a collision which changes their momenta from $\mathbf{p}_1, \mathbf{p}_2$ to $\mathbf{p}_3, \mathbf{p}_4$. Unlike the acceleration case which can be made arbitrarily smooth by taking smaller and smaller $\Delta t$, the scattering process is not continuous. It doesn't matter how small $\Delta t$ is, once a scattering happens, the two particles *jump* from one momentum

box to another. That's fundamentally different than the picture of a smooth force acting on a particle trajectory.

# 2 Day 2

## 2.1 Quantum mechanical transition rate

So how does on describe this process? Well, we do it probabilistically. The rate with which scattering $\mathbf{p}_1 + \mathbf{p}_2 \to \mathbf{p}_3 + \mathbf{p}_4$ happens depend on the density of particle $\mathbf{p}_1$ and $\mathbf{p}_2$ at $\mathbf{x}$, the conditional probability that given the initial state $\mathbf{p}_1, \mathbf{p}_2$) the final state will be $(\mathbf{p}_3, \mathbf{p}_4)$. To get the total rate, one then needs to integrate over momenta.

Recall Fermi's Golden rule for the transition rate

$$\Gamma_{i\to f} = \left|\langle f|\hat{H}_{\text{int}}|i\rangle\right|^2 (2\pi)\rho(E) \tag{61}$$

where $\rho E$ is the density of state for the final states.

Using this, the rate with which a cell at $(\mathbf{x}, \mathbf{p})$ loses its particles is given by

$$\begin{aligned}\Gamma_{\text{loss}}(t, \mathbf{p}, \mathbf{x}) \;=\; & \frac{S_{p_3p_4}}{2E_p} \int \frac{d^3p_2}{(2\pi)^3 2E_2} \int \frac{d^3p_3}{(2\pi)^3 2E_3} \int \frac{d^3p_4}{(2\pi)^3 2E_4} \left|\mathcal{M}_{pp_2\leftrightarrow p_3p_4}\right|^2 \\ & \times (2\pi)^4 \delta(p + p_2 - p_3 - p_4) f(t, \mathbf{x}, \mathbf{p}) f(t, \mathbf{x}, \mathbf{p}_2)\end{aligned} \tag{62}$$

where $\mathcal{M}_{pp_2\leftrightarrow p_3p_4}$ is the transition matrix element. The gaining rate is

$$\begin{aligned}\Gamma_{\text{gain}}(t, \mathbf{p}, \mathbf{x}) \;=\; & \frac{S_{pp_2}}{2E_p} \int \frac{d^3p_2}{(2\pi)^3 2E_2} \int \frac{d^3p_3}{(2\pi)^3 2E_3} \int \frac{d^3p_4}{(2\pi)^3 2E_4} \left|\mathcal{M}_{pp_2\leftrightarrow p_3p_4}\right|^2 \\ & \times (2\pi)^4 \delta(p + p_2 - p_3 - p_4) f(t, \mathbf{x}, \mathbf{p}_3) f(t, \mathbf{x}, \mathbf{p}_4)\end{aligned} \tag{63}$$

where I've pulled out the energy factors from the scattering matrix element to make it relativity-friendly. Here the $S$ factors take care of identical particles in the final state. Assuming that the particles are all identical, we then have the full Vlasov-Boltzmann equation

$$\langle \mathbf{F}\rangle(\mathbf{x}) = -\nabla_x \int d^3y \frac{d^3q}{(2\pi)^3} V(\mathbf{x} - \mathbf{y}) f(t, \mathbf{y}, \mathbf{q}) \tag{64}$$

$$\partial_t f(t, \mathbf{x}, \mathbf{p}) + (\mathbf{p}/m) \cdot \nabla_x f(t, \mathbf{x}, \mathbf{p}) + \langle \mathbf{F}(\mathbf{x})\rangle \cdot \nabla_p f(t, \mathbf{x}, \mathbf{p}) = \Gamma_{\text{gain}}(t, \mathbf{p}, \mathbf{x}) - \Gamma_{\text{loss}}(t, \mathbf{p}, \mathbf{x}) \tag{65}$$

Our task is to simulate this.

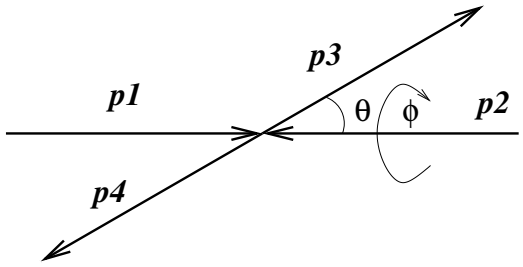
In the formula above, $\mathcal{M}_{pp_2\leftrightarrow p_3p_4|^2}$ is the scattering matrix element. This is right, but it is not the most convenient form. To get the more convenient form, let's look at the definition of scattering cross-section. If you look up the definition of a cross-section, you will find

$$d\sigma = \frac{S_{p_3p_4}}{4E_pE_2|\mathbf{v}_{\rm rel}|}\frac{d^3p_3}{(2\pi)^3 2E_3}\frac{d^3p_4}{(2\pi)^3 2E_3}\left|\mathcal{M}_{pp_2\leftrightarrow p_3p_4}\right|^2(2\pi)^4\delta^{(4)}(p+p_2-p_3-p_4) \quad (66)$$

Comparing with the rate formula, one can easily see

$$\Gamma_{\rm loss} = f(t,\mathbf{x},\mathbf{p})\frac{1}{2E_p}\int\frac{d^3p_2}{(2\pi)^3 2E_2}\int d\sigma\,|\mathbf{v}_{\rm rel}|\,f(t,\mathbf{x},\mathbf{p}_2) \quad (67)$$

Note that $d\sigma$ has 6 integrals to do but has a 4-d energy-momentum conserving delta-function. Hence, 4 of the 6 integrals can be done. Therefore $\int d\sigma$ is really a 2-D integral. We can choose the remaining 2 variables to be the solid angle $\phi, \theta$. In the CM frame,




Hence, we can write

$$\Gamma_{\rm loss} = \frac{f(t,\mathbf{x},\mathbf{p})}{2E_p}\int\frac{d^3p_2}{(2\pi)^3 2E_2}\int d\Omega\frac{d\sigma}{d\Omega}\,|\mathbf{v}_{\rm rel}|\,f(t,\mathbf{x},\mathbf{p}_2) \quad (68)$$

with the understanding that the last integral is best evaluated in the CM frame.

## 2.2 Thermal equilibrium

The Vlasov-Boltzmann equation is in general impossible to solve analytically. That's why we need numerical simulations. There is one solution, however,

that one can obtain. It's the static solution. Suppose that there is no long range forces so that $\mathbf{F} = 0$ and further suppose that the solution does not depend on $(t, \mathbf{x})$. Then the left hand side is automatically zero. To make th right hand side vanish, we need

$$f(\mathbf{p}_3)f(\mathbf{p}_4) - f(\mathbf{p})f(\mathbf{p}_2) = 0 \tag{69}$$

under the condition that the energy-momentum is conserved. The only way this can happen is that

$$f(\mathbf{p}_3)f(\mathbf{p}_4) = F(p_3 + p_4) \tag{70}$$

for some other function $F$. Then since $p + p_2 = p_3 + p_4$, Eq.(**??**) is automatically satisfied.

The only function that does this is exponential:

$$f \propto \exp(-\beta u_\mu p^\mu) \tag{71}$$

where $u^\mu$ represents the CM velocity of the whole system.

So here is a test of any numerical program that is supposed to solve the Boltzmann equation: Start with very non-thermal initial condition and see if they eventually settle to the thermal distribution. In the NR case and in the frame where $\mathbf{u} = 0$,

$$f_{\rm th}(\mathbf{p}) = \mathcal{N}\, e^{-\beta \frac{\mathbf{p}^2}{2m}} \tag{72}$$

Here $\mathcal{N}$ is the normalization factor that fixes the spatial density. Most commonly, this factor is written as

$$\mathcal{N} = e^{\beta\mu} \tag{73}$$

where $\mu$ is the *chemical potential*.

## 2.3 Boltzmann-Uehling-Uhlenbeck model

To make it simple, we will not consider only the scatterings. In that case, the equation to solve is just the Boltzmann equation

$$p^\mu \partial_\mu f(t, \mathbf{x}, \mathbf{p}) = \frac{1}{2} \int \frac{d^3 p_2}{(2\pi)^3 2E_2} \int d\Omega \frac{d\sigma}{d\Omega} \left|\mathbf{v}_{\rm rel}\right| \left(f(t, \mathbf{x}, \mathbf{p}_3) f(t, \mathbf{x}, \mathbf{p}_4) - f(t, \mathbf{x}, \mathbf{p}) f(t, \mathbf{x}, \mathbf{p}_2)\right) \tag{74}$$

using $p^0 = E_p \approx m + \mathbf{p}^2/2m$ and $E_p \mathbf{v} = \mathbf{p}$ and $p^\mu + p_2^\mu = p_3^\mu + p_4^\mu$. So far the only quantum effect included is the symmetry factor for the identical particles. But if the system is composed of identical particles, then the symmetry factor is not the only effect. If the particles are Fermions, there is the Pauli-blocking. If the particles are Bosons, there is the Bose-enhancement. These can be taken into account as follows. When colliding Fermions, each final state gets $(1 - f(t, \mathbf{x}, \mathbf{p})$. Hence

$$f(\mathbf{p}_3)f(\mathbf{p}_4) - f(\mathbf{p})f(\mathbf{p}_2) \tag{75}$$

becomes

$$f(\mathbf{p}_3)f(\mathbf{p}_4)\left(1 - f(\mathbf{p})\right)\left(1 - f(\mathbf{p}_2)\right) - f(\mathbf{p})f(\mathbf{p}_2)\left(1 - f(\mathbf{p}_3)\right)\left(1 - f(\mathbf{p}_4)\right) \tag{76}$$

For Bosons, each final state gets $(1 + f(t, \mathbf{x}, \mathbf{p}))$ so that

$$f(\mathbf{p}_3)f(\mathbf{p}_4) - f(\mathbf{p})f(\mathbf{p}_2) \tag{77}$$

becomes

$$f(\mathbf{p}_3)f(\mathbf{p}_4)\left(1 + f(\mathbf{p})\right)\left(1 + f(\mathbf{p}_2)\right) - f(\mathbf{p})f(\mathbf{p}_2)\left(1 + f(\mathbf{p}_3)\right)\left(1 + f(\mathbf{p}_4)\right) \tag{78}$$

With this modification, the resulting equation

$$\begin{aligned} &p^\mu \partial_\mu f(t, \mathbf{x}, \mathbf{p}) = \\ &\quad \frac{1}{2} \int \frac{d^3 p_2}{(2\pi)^3 2E_2} \int d\Omega \frac{d\sigma}{d\Omega} \left|\mathbf{v}_{\text{rel}}\right| \\ &\quad \Big( f(t, \mathbf{x}, \mathbf{p}_3) f(t, \mathbf{x}, \mathbf{p}_4)(1 \pm f(t, \mathbf{x}, \mathbf{p}))(1 \pm f(t, \mathbf{x}, \mathbf{p}_2)) \\ &\qquad - f(t, \mathbf{x}, \mathbf{p}) f(t, \mathbf{x}, \mathbf{p}_2)(1 \pm f(t, \mathbf{x}, \mathbf{p}_3))(1 \pm f(t, \mathbf{x}, \mathbf{p}_4)) \Big) \end{aligned} \tag{79}$$

is called the Boltzmann-Uehling-Uhlenbeck equation.

Now let's consider equilibrium. For Fermions, we need

$$f(\mathbf{p}_3)f(\mathbf{p}_4)\left(1 - f(\mathbf{p})\right)\left(1 - f(\mathbf{p}_2)\right) = f(\mathbf{p})f(\mathbf{p}_2)\left(1 - f(\mathbf{p}_3)\right)\left(1 - f(\mathbf{p}_4)\right) \tag{80}$$

To solve this, let's set

$$1 - f(\mathbf{p}) = f(\mathbf{p})g(\mathbf{p}) \tag{81}$$

Then the condition becomes

$$g(\mathbf{p})g(\mathbf{p}_2) = g(\mathbf{p}_3)g(\mathbf{p}_4) \tag{82}$$

which we know the solution

$$g(\mathbf{p}) = e^{\pm\beta(u_\mu p^\mu - \mu)} \tag{83}$$

and

$$f(\mathbf{p}) = \frac{1}{g(\mathbf{p}) + 1} \tag{84}$$

If one demands that at low energy $f(\mathbf{p})$ go back to the Boltzmann distribution to get the Fermi-Dirac distribution

$$f(\mathbf{p}) = \frac{1}{1 + e^{\beta(E_p - \mu)}} \tag{85}$$

in the rest frame where $\mathbf{u} = 0$. For Bosons, let

$$1 + f(\mathbf{p}) = f(\mathbf{p})g(\mathbf{p}) \tag{86}$$

then again we have $g(\mathbf{p})g(\mathbf{p}_2) = g(\mathbf{p}_3)g(\mathbf{p}_4)$ with the solution $g(\mathbf{p}) = e^{\pm\beta(u_\mu p^\mu - \mu)}$. Solving for $f(\mathbf{p})$, we get

$$f(\mathbf{p}) = \frac{1}{g(\mathbf{p}) - 1} \tag{87}$$

which yields the Bose-Einstein distirubtion

$$f(\mathbf{p}) = \frac{1}{e^{\beta(E-\mu)} - 1} \tag{88}$$

## 2.4 Numerical Test of thermalization

You should have received a copy of the test programs. If you unzip sim3d.zip, it will create a subdirectory named `3DSim`. In that directory, you will find the source codes in C and the makefile. The makefile is pretty generic and it does not depend on any outside library. Hence, if you have `gcc` just typing `make` will create the executable named `sim3d`.

In the same directory, you will find a subdirectory named `Runs`. CD to `Runs` and you will find some example input files. On the command line, try

```
../sim3d input.typical output.file
```
If your compilation went well, it should run and produce a bunch of output files.

The file named `input.typical` contains the input data which should not change during the course of similation. It currently has these parameters:

```
Time_step_choice_Given_0_Calc_1 0
Final_time_choice_Given_0_Mft_1 0
Initial_Condition_Pfixed_1_Thermal_2_Elliptic_3 1
Time_step_size 1.0
Final_time 200.0
Average_number_of_collisions 2
Ellipse_z_max 5
Ellipse_y_max 3
Ellipse_x_max 5
RandomSeed 123456789
CrossSection_in_mb 10
Mass_in_MeV 140
Pmax 14
Pz_up 10
Pz_down 0
Py_up 10
Py_down 0
Px_up_in_MeV 10
Px_down_in_MeV 0
Z_up 10
Z_down -10
Y_up 10
Y_down -10
X_up 10
X_down -10
Initial_time 0.0
Number_of_particles 1500
EndOfData
```

This a hash-table. The first "key-word" without any white space is the description of the parameter and the second column is the value of the parameter. The read-in function looks for the key-word and takes the value next to it. Hence, the order does not matter. You can move the lines up and

down but don't touch the last line that says `EndOfData`. You can also write a comment by writing `#` at the beginning of a line. The read-in program ignores that.

The program that reads in this file is in `util.c`. In this "utilities" file, you will find this function

```
char *StringFind(char *file_name, char *st)
{
 char *s, *x;
 char line[1000];
 FILE *input, *tmp_file;
 int ind;
 static int flag = 0;

 if(flag == 0)
  {
   if(!IsFile(file_name))
    {
     fprintf(stderr, "The file named %s is absent.\n", file_name);
     if(file_name == NULL)
      {
       fprintf(stderr, "No input file name specified.\n");
       fprintf(stderr, "Please specify the input file name. Exiting.\n");
       exit(1);
      // fprintf(stderr, "Creating a default file named input...\n");
      // file_name = char_malloc(80);
      // strcpy(file_name, "input");
      }
     else
      {
       fprintf(stderr, "Creating %s..\n", file_name);
      }
     tmp_file = fopen(file_name,"w");
     fprintf(tmp_file, "EndOfData\n");
     fclose(tmp_file);
    }/* if isfile */
   flag = 1;
  }/* if flag == 0 */
```

```
 input = fopen(file_name,"r");

// allocate
 s = char_malloc(100);
 x = char_malloc(100);

 ind = 0;
 while(fgets(line, sizeof line, input))
  {
   sscanf(line, "%s  %s", s, x);
   if(line[0] != '#')
    {
     if(strcmp(s, st) == 0)
      {
       free(s);
       return x;
      }
    }
  }// while

 fclose(input);

 if(ind == 0)
  {
   fprintf(stderr, "StringFind: %s not found in %s.\n", st, file_name);
   printf("Enter %s = ", st);
   scanf("%s", x);
   printf("Rewriting %s...\n", file_name);
   ReWriteString(file_name, st, x);
   free(s);
   return x;
  }
 else
  {
   fprintf(stderr, "StringFind: This should not be reached.\n");
   return NULL;
  }
```

```
}/* StringFind */
```

This is the main function and it actually reads in the value of the parameter as a character string. There are few other functions, `DFind, IFind` etc that specifically reads in the double precision value, integer value, etc.

In `main.c`, `ReadInParams`, you will see that these functions are used as follows

```
void ReadInParams(char *input)
{
 double t0, temperature, lmfp, dens, volume;
 long int il_var;
 FILE *input_file;
 FILE *output_file;

 // we do everything in fm

 fprintf(stderr, "ReadInParams: Input file is %s\n", input);

 input_file = fopen(input, "r");
 output_file = fopen("params.dat", "w");

 COM_DATA.num_ptcls = IFind(input, "Number_of_particles");
 fprintf(stderr, "number of particles = %d\n", COM_DATA.num_ptcls);
 .
 .
 .
}
```

To make the input file, you have some choices. You can call the executable with just the name of the non-existent input file. In that case, each call to `DFind, IFind`, etc will ask you the value and add a line to the named file. If you have an incompletely filled out input file, again whenever the `Find` functions cannot locate the asked-for value, it will ask and create a line in the input file. If you have an already filled-out input file, it will just read in the values and do not bother you.

Now, the input data are defined in `data.h`

```
#ifndef DATA_H
#define DATA_H

#ifndef hbarc
#define hbarc (197.3)
#endif

typedef struct init_data
{
 int num_ptcls;

// x ranges
// x_down[0] is the start time
// x_up[0] is the end time
 double x_down[4];
 double x_up[4];

// p ranges
// p_down[0] and p_up[0] are energy slots
 double p_down[4];
 double p_up[4];
 double p_max;

// mass
 double mass;

// cross-section
 double sigma;
 double r0;

// random seed
long int iseed;

// initial condition choices
int init_choice;

// epllipse params
double ax;
```

```
double ay;
double az;

// time step size
double h;
double dt;
int time_step_choice;

// number of time step
int num_steps;

// mean free time
double t_mft;

// maximum speed in 1-d
double v_max;

// temperature
double temperature;

double ave_num_coll;

// final time choice
int final_time_choice;

} InitData;

InitData COM_DATA; // static common data

int COM_IND; // temp

#endif
```

If you want to add another input parameter, all you have to do is add that to the structure `InitData` and put another line in the function `ReadInParams`. For instance, you can specify the name of the output file by putting in an entry in `InitData`

```
typedef struct init_data
```

```
{
 .
 .
 .

 char *output_file_name;
} InitData;
```

and then in the `ReadInParams`

```
void ReadInParams(char *input)
{
.
.
.
 COM_DATA.output_file_name = StringFind(input, "Output_file_name");
.
.
.
}
```

then compile again. Now next time you run `sim3d`, it will ask for the name and add it in to the input file.

Let's take a look at the `main` function

```
int main(int argc, char **argv)
{
 char *input;
 char *output;
 Particle *ptcls;

 if(argc != 3)
  {
   fprintf(stderr, "Must provide input and output file names.\n");
   exit(0);
  }

 input = argv[1];
 output = argv[2];
```

```
 fprintf(stderr, "main: Program name is %s\n", argv[0]);
 fprintf(stderr, "main: Input file is %s\n", input);
 fprintf(stderr, "main: Output file is %s\n", output);
 fprintf(stderr, "main: Removing all previous data files...\n");
 system("rm -f *.dat");

 ReadInParams(input);

 ptcls = (Particle *) malloc(sizeof(Particle)*(COM_DATA.num_ptcls));

 Initialize(ptcls);
 PrintParticles(ptcls, "init_ptcls.dat");
 BinP(ptcls, "initial_p.dat");
 PrintParticlesTwoLayers(ptcls, "init_ptcls_l.dat","init_ptcls_u.dat");

 Evolve(ptcls, output);
 PrintParticles(ptcls, "fin_ptcls.dat");
 BinP(ptcls, "final_p.dat");
 PrintParticlesTwoLayers(ptcls, "fin_ptcls_l.dat","fin_ptcls_u.dat");

 return 1;
}// main
```

Simple, isn't it. This is how you should try to structure your programs. The function `main` is like the list of contents for a book. It only lists "sections". If you go to the one of the "sections", that function should only list the "subsections". Unless the programming task it really simple, it is rare that you actually need to implement anything at the "section" level. The "subsection" level may contain some actual implementations, but most likely a function at this level will contain only the list of "subsubsection" and so on. Actual numerical work should be done functions in the "subsubsection" level and below.

Now, the functions that actually do something should be short and it should *do one thing and one thing only and it must do it as efficiently and thoroughly as possible*. This is the original UNIX programming philosophy - You first create small tools. You then assemble small tools to make a bigger tool, then you assemble the bigger tools to make yet bigger tools, ... There

are good points and bad points to this philosophy. The good points are that your program will be easy to read as long as you give your functions very descriptive names and it will be easy to maintain/modify/improve your code since everything is as modula. The bad things about this approach is that it is not easy to optimize your code in this way. It will be fast if you noticed the "as efficient as possible" part above, but it won't be the fastest possible. However, I find that I make much less coding mistakes following this philosophy.

Now, let's run some test cases. If you run `sim3d` with the provided `input.typical`, it will do 100 time steps. Let's change that. First copy `input.typical` to `input` so that we keep the example intact. In `input`, change `Final_time 100` to `Final_time 20` and run `../sim3d input output.dat`. This produces a file named `final_p.dat` which contains the distribution $dN/p^2dp$ of the simulated particles and also the functional form of the corresponding thermal distribution. You will see that the program report that the average number of collisions is much less than 1 and the calculated distribution is not quite thermal. Now increas `Final_time` to 100 and do it again. It will take a bit longer, but this time the average number of collisions is about 1 and the distribution looks more like the thermal although there still is a strong remnant of the initial distribution. Increase `Final_time` yet again to 200 and run it. You will now see that the number of collisions has increased to about 2 and the distribution looks more or less thermal although the first few bins are off. OK. On to `Final_time 300`. Note the number of collisions. Is that what you would have expected? The distribution will now look much more thermal but not quite. In fact, it is not easy to get the low $p$ part of the thermal spectrum by colliding hard-spheres. This is because the collisions that produces low energy particles is rare furthermore, the to value reported is calculated as

$$\frac{dN}{p^2dp} \approx \frac{\Delta N}{p^2\Delta p} \tag{89}$$

Note the factor of $p^2$ in the denominator. For small $p$ this amplifies the error. If you plot $\frac{dN}{dp}$, the two distribution will look much closer even when all particles suffered on average only 2 scatterings or so. Try it.

## 2.5 How to solve the Boltzmann equation

The gaining rate is

$$\Gamma_{\rm gain}(t,\mathbf{x},\mathbf{p}) = \frac{1}{2E_p}\int \frac{d^3p_2}{(2\pi)^3 2E_2}\int d\Omega \frac{d\sigma}{d\Omega}\,|\mathbf{v}_{\rm rel}|\, f(t,\mathbf{x},\mathbf{p}_3) f(t,\mathbf{x},\mathbf{p}_4) \quad (90)$$

with the understanding that $p^\mu + p_2^\mu = p_3^\mu + p_4^\mu$.

How do you simulate this? Well, we want to use particles. That's where scattering cross-section is useful. To think about it, we first need to know what a cross-section means. The definition of a cross-section is "the effective area where something happens".

Let $\mathbf{b}$ be the displacement vector between the centers of two colliding particles. The effective area can be defined as

$$\sigma = \int d^2b\, P(\mathbf{b}) \quad (91)$$

where $P(\mathbf{b})$ is the probability that *something* happens at $\mathbf{b}$. Let's compare that with the conventional way to define the cross-section. The probability that something happens when a beam of particle with the beam area $A$ is incident on a *single target* is

$$P_{\rm scatt} = \frac{\text{Out flux}}{\text{In flux}} \quad (92)$$

The out-flux is defined as

$$\text{Out flux} = \int d\mathbf{S}\cdot\mathbf{J}_{\rm out} \quad (93)$$

and the in-flux

$$\text{In flux} = AJ_z \quad (94)$$

The cross-section (the effective area) is then

$$\begin{aligned}\sigma &= AP_{\rm scatt}\\ &= \lim_{r\to\infty}\frac{\int d\Omega r^2\, J_{r,\rm out}}{J_z}\end{aligned} \quad (95)$$

using the area element on a sphere

$$d\mathbf{S} = r^s \sin\theta d\theta d\phi\, \hat{\mathbf{r}} \quad (96)$$

and the definition

$$J_{r,\text{out}} = \hat{\mathbf{r}} \cdot \mathbf{J}_{\text{out}} \tag{97}$$

The differential cross-section is

$$\frac{d\sigma}{d\Omega} = \lim_{r \to infty} \frac{r^2 J_{r,\text{out}}}{J_z} \tag{98}$$

The relationship between the out-flux and the cross-section is then

$$\lim_{r\to\infty} r^2 J_{r,\text{out}} = J_z \frac{d\sigma}{d\Omega} \tag{99}$$

The incident flux is defined as the density of projectile times the speed

$$J_z = n_{\text{proj}} v_{\text{rel}} \tag{100}$$

so that

$$\lim_{r\to\infty} r^2 J_{r,\text{out}} = n_{\text{proj}} v_{\text{rel}} \frac{d\sigma}{d\Omega} \tag{101}$$

Recall that this is for a single target particle. If the incident beam is hitting a group of targets, the *total out-flux* is given by

$$\lim_{r\to\infty} r^2 J_{r,\text{out}} = \frac{d\sigma}{d\Omega} v_{\text{rel}} n_{\text{proj}} n_{\text{targ}} A d \tag{102}$$

where $d$ is the thickness of the thin target. Now it looks more or less what is in the loss/gain rate.

This means that to solve the Boltzmann equation, all one has to do is to simulate particle scatterings. That's it. So here is the strategy to simulate this system.

1. We need particle properties such as the mass
2. We need the differential cross-section $d\sigma/d\Omega$
3. Start from an initial configuration of particles
4. Using the total cross-section $\sigma = \int d\Omega (d\sigma/d\Omega)$, determine scattering pairs
5. Determine the final states using $d\sigma/d\Omega$
6. Continue

# 3 Day 3

## 3.1 Relativistic Thomas Fermi Approximation

Our main reference is PRC 46 No 5, 1797 (1992) by Von-Eiff and Weigel For details, read the paper.

**Main idea**
The main idea is simple. The protons and neutrons inside a nucleus are bound because they extert forces on each other. However, dealing with $A(A-1)/2$ pairs of forces individually is too complicated. Here $A$ is the total number of protons and neutrons. Collectively they are called the nucleons. One way to simplify the calculation is to use the idea of the average potential. That is, each nucleon only feels the average force and this average force is common to all.

This technique called the Thomas-Fermi approximation was originally developed for the atomic electron system. When an atom has $N$ electrons, the Hamiltonian operator is given by

$$\hat{H} = \sum_{i=1}^{N} \frac{\hat{\mathbf{p}}_i^2}{2m_e} + \sum_{i>j} V_{\text{Coul.}}(\mathbf{r}_i - \mathbf{r}_j) + \sum_{i=1}^{N} V_{\text{ext}}(\mathbf{r}_i) \tag{103}$$

where $V_{\text{Coul.}}(\mathbf{r}_i - \mathbf{r}_j)$ is the Coulomb potential between the $i$-th electron and the $j$-th electron and $V_{\text{ext}}(\mathbf{r}_i)$ is the Coulomb potential between the $i$-th electron and the nucleus. This is a very complicated problem and solving it exactly is out of question (even numerically) for more than a handful of electrons.

To have some understanding of this complicated system, Fermi developed a method that is a judicious mixture of classical and quantum mechanics. It goes as follows. The charge density of this system is made up of the electron density, which in turn is given by the wavefunctions

$$\rho_C(\mathbf{r}) = -e\rho_N(\mathbf{r}) = -e\sum_{i=1}^{N} e|\psi_i(\mathbf{x}_i)|^2 \tag{104}$$

This then determines the electric field according to the Gauss law

$$\nabla^2 A^0 = -\rho_C = e\rho_N \tag{105}$$

Now we also know that electrons are Fermions. Hence, they obey Pauli exclusion principle and that means that each energy level can hold only up to 2 electrons. Given the number of electrons, then there is a highest quantum energy level which is filled and above it is empty. This energy level is called the Fermi energy $\mu_F$ which is basically the chemical potential. Recall that the chemical potential for a system is defined as the energy you need to put a particle into that system.

For an electron at the position $\mathbf{r}$, the potential energy it feels is then $-eA^0(\mathbf{r})$. Total energy of course is made up of the potential energy and the kinetic energy. For any electron at any position $\mathbf{r}$, the maximum energy it can have is the Fermi energy $\mu_F$ which implies that there is a maximum kinetic energy. For the maximum kinetic energy an electron can have at $\mathbf{r}$, consider a free gas of electrons with the density $\rho_N$. In that case, the maximum kinetic energy is given by the Fermi energy of free electros. One can show that the corresponding maximum momentum $p_F$ is related to the density

$$\rho_N = \frac{p_F^3}{3\pi^2} \tag{106}$$

In Thomas-Fermi approximation, this relationship is promoted to be valid locally

$$p_F(\mathbf{r}) = \left(3\pi^2 \rho_N(\mathbf{r})\right)^{1/3} \tag{107}$$

which can be thought of as a first approximation.

Combined, we then have

$$\begin{aligned} \mu_F &= \frac{p_F^2(\mathbf{r})}{2m_e} - eA^0(\mathbf{r}) + V_{\text{ext}}(\mathbf{r}) \\ &= \frac{(3\pi^2\rho_N(\mathbf{r}))^{2/3}}{2m_e} - eA^0(\mathbf{r}) + V_{\text{ext}}(\mathbf{r}) \end{aligned} \tag{108}$$

which can be re-expressed as

$$p_F(\mathbf{r}) = \sqrt{2m_e(\mu_F + eA^0(\mathbf{r}) - V_{\text{ext}}(\mathbf{r}))} \tag{109}$$

That is, given $A^0(\mathbf{r}), V_{\text{ext}}(\mathbf{r})$ and a global constant $\mu_F$, we can calculate the density as

$$\rho_N(\mathbf{r}) = \frac{p_F^3(\mathbf{r})}{3\pi^2} \tag{110}$$

The electric potential itself is determined by the density

$$\nabla^2 A^0(\mathbf{r}) = e\rho_N(\mathbf{r}) \tag{111}$$

And the chemical potential must satisfy

$$N = \int d^3r\, \rho_N(\mathbf{r}) \tag{112}$$

where the prefactor 2 comes from the spin degeneracy. Solving these three equations self-consistently make up the Thomas-Fermi approximation.

To solve the equations, one can use the following iteration procedure.

1. Start with an initial guess for the density $\rho_N^{\text{now}}(\mathbf{r})$ that satisfies Eq.(112).
2. Calculate the potential $A^0_{\text{now}}(\mathbf{r})$ with $\rho_N^{\text{now}}(\mathbf{r})$ by solving the Gauss law Eq.(111).
3. Use Eq.(108) to calculate the next $\rho_N^{\text{next}}(\mathbf{r})$ with an initial guess of $\mu_F$

   $$\rho_N^{\text{next}}(\mathbf{r}) = \frac{p^3_{F,\text{next}}(\mathbf{r})}{3\pi^2} \tag{113}$$

   where

   $$p_{F,\text{next}} = \sqrt{2m_e(\mu_F + eA^0_{\text{now}}(\mathbf{r}) - V_{\text{ext}}(\mathbf{r}))} \tag{114}$$

4. Adjust $\mu_F$ so that Eq.(112) is satified.
5. Compare $\rho_N^{\text{now}}(\mathbf{r})$ and $\rho_N^{\text{next}}(\mathbf{r})$. If they have converged, then a solution is found. If not, then set $\rho_N^{\text{now}}(\mathbf{r}) = \rho_N^{\text{next}}(\mathbf{r})$ and go back to step 1.

That's for an atomic system. Now consider a nucleus. First, it is made up of two different kinds of particles, protons and neutrons. Second, there is no center providing the external potential. Nevertheless, it still is a quantum system made up of fermions with interactions between them.

What about the force? There certainly is Coulomb interaction between the protons. There also must be nuclear forces that holds the nucleus together. So we postulate 3 additional kind of potentials. There is a scalar potential satisfying

$$(\nabla^2 - m_\sigma^2)\sigma = -g_\sigma \rho_S \tag{115}$$

where $\rho_S(\mathbf{r})$ is the scalar density we will shortly specify. There is also nuclear potential

$$(\nabla^2 - m_\omega^2)\omega = -g_\omega \rho_B \tag{116}$$

where $\rho_B$ is the sum of the proton density and the neutron density (the baryon density). These two forces are isospin (a precise way of distinguishing proton and neutron) blind. The following, however, isospin dependent:

$$(\nabla^2 - m_\rho^2)\rho = -g_\rho \rho_3 \tag{117}$$

where $\rho_3$ is the difference between the proton density and the neutron density. Finally there is the Coulomb potential

$$\nabla^2 A^0 = -e\rho_p \tag{118}$$

which influence only the protons. Here $m_\sigma, m_\omega, m_\rho$ are the masses of the mesons that mediate the forces and $g_\sigma, g_\omega, g_\rho$ are the coupling constants.

Since we are dealing with an atomic nucleus, the relativistic effects cannot be ignored. That means we must use the full relativistic expression for the energy.

$$E = \sqrt{\mathbf{p}^2 + m^2} + V(\mathbf{r}) \tag{119}$$

We know what the potential energy due to the Coulomb interaction is. It's the time-component of the 4-vector potential $eA^0(\mathbf{r})$, but only for the protons. For other forces, the $\omega$ and $\rho$ fields should also be regarded as the time-component of the 4-vector meson fields. Hence,

$$V_p(\mathbf{r}) = eA^0(\mathbf{r}) + g_\omega \omega + g_\rho \rho \tag{120}$$

for the proton and

$$V_n(\mathbf{r}) = g_\omega \omega - g_\rho \rho \tag{121}$$

for the neutron. Note that $\rho$ couples differently to proton and neutron as it can distinguish the two. The positive coupling constant here implies repulsion and the negative one attraction. For instance, the potential for the proton is purely repulsive. So where does the attraction that holds the nucleus

together come from? It comes from the scalar interaction. One can show that the role of the $\sigma$ field is to change the effective mass

$$m^*(\mathbf{r}) = m - g_\sigma \sigma(\mathbf{r}) \tag{122}$$

Again, since they are Fermions, there is the Fermi energy $\mu_p$ and $\mu_n$ for protons and neutrons, respectively. In analogy of the electron case, they are given by

$$\begin{aligned} \mu_p &= \sqrt{p_{F_p}(\mathbf{r})^2 + m_p^*(\mathbf{r})^2} + V_p(\mathbf{r}) \\ &= \sqrt{p_{F_p}(\mathbf{r})^2 + m_p^*(\mathbf{r})^2} + eA^0(\mathbf{r}) + g_\omega \omega + g_\rho \rho \end{aligned} \tag{123}$$

and

$$\begin{aligned} \mu_n &= \sqrt{p_{F_n}(\mathbf{r})^2 + m_n^*(\mathbf{r})^2} + V_n(\mathbf{r}) \\ &= \sqrt{p_{F_n}(\mathbf{r})^2 + m_n^*(\mathbf{r})^2} + g_\omega \omega - g_\rho \rho \end{aligned} \tag{124}$$

The Fermi momenta $p_{F_{p,n}}$ are related as before to the densities

$$\rho_p(\mathbf{r}) = \frac{p_{F_p}(\mathbf{r})^3}{3\pi^2} \tag{125}$$

$$\rho_n(\mathbf{r}) = \frac{p_{F_n}(\mathbf{r})^3}{3\pi^2} \tag{126}$$

and the Fermi energies (chemical potentials) must satisfy the constraints

$$Z = 4\pi \int_0^\infty dr\, r^2\, \rho_p(r) \tag{127}$$

$$(A - Z) = 4\pi \int_0^\infty dr\, r^2\, \rho_n(r) \tag{128}$$

The scalar density is defined to be

$$\begin{aligned} \rho_S(\mathbf{r}) &= 2 \int \frac{d^3 p_p}{(2\pi)^3} \frac{m_p^*(\mathbf{r})}{E_{p_p}} \theta(p_{F_p}(\mathbf{r}) - p_p) + 2 \int \frac{d^3 p_n}{(2\pi)^3} \frac{m_n^*(\mathbf{r})}{E_{p_n}} \theta(p_{F_n}(\mathbf{r}) - p_n) \\ &= \frac{m_p^*(\mathbf{r})^3}{2\pi^2} \left( \frac{p_{F_p}(\mathbf{r}) E_{F_p}(\mathbf{r})}{m_p^*(\mathbf{r})^2} - \ln \left( \frac{p_{F_p}(\mathbf{r}) + E_{F_p}(\mathbf{r})}{m_p^*(\mathbf{r})} \right) \right) \\ &\quad + \frac{m_n^*(\mathbf{r})^3}{2\pi^2} \left( \frac{p_{F_n}(\mathbf{r}) E_{F_n}(\mathbf{r})}{m_n^*(\mathbf{r})^2} - \ln \left( \frac{p_{F_n}(\mathbf{r}) + E_{F_n}(\mathbf{r})}{m_n^*(\mathbf{r})} \right) \right) \end{aligned} \tag{129}$$

where

$$E_{F_{p,n}} = \sqrt{p_{F_{p,n}}(\mathbf{r})^2 + m^*_{p,n}(\mathbf{r})^2} \tag{130}$$

The equation of motion for the $\sigma$ field is

$$(\nabla^2 - m_\sigma^2)\sigma = -g_\sigma \rho_S \tag{131}$$

Hence, obtaining $\sigma$ is not just a matter of inverting the KG operator since the source also contains $\sigma$ itself. This equation needs to be solved self-consistently by iteration.

## 3.2 Iteration to solve for the nuclear densities

So the iteration process should look like the following. We will simplify the problem by looking only for the spherically symmetric solution. That is, all functions are functions of $r = |\mathbf{r}|$ only.

1. Start with initial guesses of $\rho_p(r)$ and $\rho_n(r)$.
2. Calculate the electric potential $A^0$, $\omega$ and $\rho$.
3. Calculate the initial $\sigma(r)$
4. Calculate $p_{F_{p,n}}(r)$ from $\rho_p(r)$ and $\rho_n(r)$. and calculate the scalar density $\rho_S(r)$.
5. Calculate $\sigma(r)$.
6. Get the next $\rho_p(r)$ and $\rho_n(r)$ using Eqs.(123) – (126) and adjusting $\mu_p$ and $\mu_n$ to satisfy Eqs.(158) and (159).
7. Check convergency by comparing the old $\rho_{p,n}$ and the new $\rho_{p,n}$. If the difference is larger than the set tolerance level, then go back to step 2. If the difference is within the set tolerance level, then stop the calculation and output the calculation.

## 3.3 Units

Before we start to build up our program, let's first talk about the units. In the nuclear world, the most natural length unit is $10^{-15}$ m or a femto-meter, or a fermi, denoted as fm. That's the size of a proton or a neutron. The natural mass unit (energy unit) is GeV/$c^2$. But most of times, /$c^2$ is tacitly understood and we just use GeV. For instance, the mass of a proton is almost 1 GeV ($\sim 0.94$ GeV).

In writing our KG equation as

$$(\nabla^2 - m^2)\phi = -\rho \tag{132}$$

without any factors of $\hbar$ and $c$, we are already assuming that we use the "natural unit" where $\hbar = c = 1$ so that mass, momentum, energy all have the same unit and since

$$\hbar c = 0.19732697\,\text{fm}\,\text{GeV} \tag{133}$$

the inverse length also have the same unit. Conceptually this is all good. But in practice, one needs to choose whether everything is going to be measured with the length unit fm or the energy unit GeV. Either one is fine, but one does need to choose one and stick with it. Since our problem is to find proton and neutron densities as a function of $r$, it is perhaps more natural to use the length unit to measure everything. Let's do that.

So when a dimensionful quantity is input, it must be converted to the our unit system. For example, the proton mass in GeV is (We don't need to bother with $c$ just yet. Those will (and must) be put back later.)

$$m_p = 0.93827204\,\text{GeV} \tag{134}$$

As soon as this is read in, it should be converted

$$m_p \leftarrow m_p/\text{hbarc} \tag{135}$$

where hbarc = 0.19732697. That is, in the unit of 1/fm, the proton mass is

$$m_p = 0.93827204/0.19732697 = 4.7549103 \tag{136}$$

in our numerics. All mass, momentum and energy parameters should be treated the same way. All length parameters should be entered in fm. In this way, our densities we calculate will be automatically in the right unit (fm$^{-3}$) without any further conversion.

## 3.4 Input parameters

Input parameters such as the mass of a proton remain unchanged during the whole calculation. It is therefore a bit silly to pass them to any function as arguments. Hence, these static input parameters can be declared as common variables. It is a good practice to make a list of these static parameters rather than declaring them one by one as common. Only these static input parameters should be declared common. I strongly discourage using common variables for any other purposes.

## 3.5 Densities and Fermi momentum

Given the proton density $\rho_p(r)$ and the neutron density $\rho_n(r)$, the local Fermi momentum is defined by

$$\begin{aligned}\rho_{p,n}(r) &= g_p \int \frac{d^3p_{p,n}}{(2\pi)^3}\theta(p_{F_{p,n}} - p_{p,n}) \\ &= \frac{p^3_{F_{p,n}}(r)}{3\pi^2}\end{aligned} \tag{137}$$

where $g_p = 2$ is the spin degeneracy factor. Equivalently,

$$p_{F_{p,n}}(r) = \left(3\pi^2\rho_{p,n}(r)\right)^{1/3} \tag{138}$$

The scalar density is then defined by

$$\rho_S(r) = g_p m_p(r) \int \frac{d^3p_p}{(2\pi)^3}\frac{1}{E_{p_p}}\theta(p_{F_p} - p_p) + g_n m_n(r) \int \frac{d^3p_n}{2(\pi)^3}\frac{1}{E_{p_n}}\theta(p_{F_n} - p_n) \tag{139}$$

where $g_p = g_n = 2$ are the spin degeneracy factor and

$$m_{p,n}(r) = m_{p,n} - g\sigma(r) \tag{140}$$

and

$$E_{p_{p,n}} = \sqrt{\mathbf{p}^2_{p,n} + m_{p,n}(r)^2} \tag{141}$$

The scalar density integrals can be analytically worked out

$$\begin{aligned}2m \int \frac{d^3p}{(2\pi)^3}\frac{1}{E_p}\theta(p_F - p) &= \frac{m}{\pi^2}\int_0^{p_F} dp\,\frac{p^2}{E_p} \\ &= \frac{m^3}{\pi^2}\int_0^{\theta_F} d\theta\,\sinh^2\theta\end{aligned} \tag{142}$$

with $p = m\sinh\theta$ and $E = \sqrt{p^2+m^2} = m\cosh\theta$. Using

$$\begin{aligned}\sinh^2\theta &= \frac{(e^\theta - e^{-\theta})^2}{4} \\ &= \frac{e^{2\theta} + e^{-2\theta} - 2}{4} \\ &= \frac{\cosh 2\theta - 1}{2}\end{aligned} \tag{143}$$

we get

$$\begin{aligned}2m\int \frac{d^3p}{(2\pi)^3}\frac{1}{E_p}\theta(p_F - p) &= \frac{m^3}{\pi^2}\int_0^{\theta_F} d\theta\ \sinh^2\theta \\ &= \frac{m^3}{2\pi^2}\int_0^{\theta_F} d\theta\,(\cosh 2\theta - 1) \\ &= \frac{m^3}{2\pi^2}\left(\frac{\sinh 2\theta_F}{2} - \theta_F\right)\end{aligned} \tag{144}$$

with

$$\theta_F = \sinh^{-1}(p_F/m) \tag{145}$$

Note that

$$\sinh 2\theta = 2\sinh\theta\cosh\theta \tag{146}$$

Hence

$$2m\int \frac{d^3p}{(2\pi)^3}\frac{1}{E_p}\theta(p_F - p) = \frac{m^3}{2\pi^2}\left(\frac{p_F E_F}{m^2} - \theta_F\right) \tag{147}$$

The ArcSine function is not readily available in may computer languages. We can use

$$\begin{aligned}e^\theta &= \sinh\theta + \cosh\theta \\ &= \sinh\theta + \sqrt{1+\sinh^2\theta}\end{aligned} \tag{148}$$

Taking the logarithm and letting $x = \sinh\theta$

$$\theta = \ln(x + \sqrt{1+x^2}) = \sinh^{-1}(x) \tag{149}$$

Hence finally

$$2m\int \frac{d^3p}{(2\pi)^3}\frac{1}{E_p}\theta(p_F - p) = \frac{m^3}{2\pi^2}\left(\frac{p_F E_F}{m^2} - \ln\left(\frac{p_F + E_F}{m}\right)\right) \tag{150}$$

## 3.6 Chemical potential

In the Thomas-Fermi approximation, the highest energy that a particle can have is composed of the kinetic energy $E_{p_F}$ and the potential energies. The potential energies are provided by the scalar field $\sigma$, and the vector fields $A^0$, $\omega^0$ and $\rho_0^0$.

From Von-Eiff and Weigel we follow, this is given by

$$\mu_p = E_{p_{F_p}}(r) + eA^0(r) + g_\rho \rho_0^0(r) + g_\omega \omega^0(r) \tag{151}$$

$$\mu_n = E_{p_{F_n}}(r) - g_\rho \rho_0^0(r) + g_\omega \omega^0(r) \tag{152}$$

The way this is interpreted is this: Given $\mu_{p,n}$, calculate the densities. That is, given $\mu_{p,n}$ we get the Fermi energy

$$E_{p_{F,p}}(r) = \mu_p - eA^0(r) - g_\rho \rho_0^0(r) - g_\omega \omega^0(r) \tag{153}$$

$$E_{p_{F_n}}(r) = \mu_n + g_\rho \rho_0^0(r) - g_\omega \omega^0(r) \tag{154}$$

then we can get the Fermi momentum through

$$E_F = \sqrt{p_F^2 + m^2} \tag{155}$$

and once we know Fermi momentum, we can calculate the density

$$\rho = \frac{p_F^3}{3\pi^2} \tag{156}$$

In all this, the mass used is always the effective mass

$$m_{p,n}(r) = m_{p,n}^{\text{phys}} - g_\sigma \sigma(r) \tag{157}$$

The chemical potentials must satisfy

$$Z = 4\pi \int_0^\infty dr\, r^2\, \rho_p(r) \tag{158}$$

$$(A - Z) = 4\pi \int_0^\infty dr\, r^2\, \rho_n(r) \tag{159}$$

The initial $\sigma$ field is given by

$$\sigma(r) = g_\sigma \rho_S(r)/m_\sigma^2 \tag{160}$$

where the scalar density is defined to be

$$\begin{aligned}\rho_S(\mathbf{r}) &= 2\int \frac{d^3p_p}{(2\pi)^3}\frac{m_p^*(\mathbf{r})}{E_{p_p}}\theta(p_{F_p}(\mathbf{r})-p_p) + 2\int \frac{d^3p_n}{(2\pi)^3}\frac{m_n^*(\mathbf{r})}{E_{p_n}}\theta(p_{F_n}(\mathbf{r})-p_n) \\ &= \frac{m_p^*(\mathbf{r})^3}{2\pi^2}\left(\frac{p_{F_p}(\mathbf{r})E_{F_p}(\mathbf{r})}{m_p^*(\mathbf{r})^2} - \ln\left(\frac{p_{F_p}(\mathbf{r})+E_{F_p}(\mathbf{r})}{m_p^*(\mathbf{r})}\right)\right) \\ &\quad + \frac{m_n^*(\mathbf{r})^3}{2\pi^2}\left(\frac{p_{F_n}(\mathbf{r})E_{F_n}(\mathbf{r})}{m_n^*(\mathbf{r})^2} - \ln\left(\frac{p_{F_n}(\mathbf{r})+E_{F_n}(\mathbf{r})}{m_n^*(\mathbf{r})}\right)\right) \qquad (161)\end{aligned}$$

where

$$p_{F_{p,n}}(r) = (3\pi^2\rho_{p,n}(r))^{1/3} \qquad (162)$$

and

$$E_{F_{p,n}} = \sqrt{p_{F_{p,n}}(\mathbf{r})^2 + m_{p,n}^*(\mathbf{r})^2} \qquad (163)$$

and

$$m_{p,n}^*(r) = m_{p,n} - g_\sigma\sigma(r) \qquad (164)$$

So the idea here is that given $\rho_p(r)$ and $\rho_n(r)$, we use the iteration scheme

$$\sigma^{(n+1)}(r) = (g_\sigma/m_\sigma^2)\rho_S^{(n)}(r) \qquad (165)$$

where $\rho_S^{(n)}(r)$ is calculated with

$$m_{p,n}^*(r) = m_{p,n} - g_\sigma\sigma^{(n)}(r) \qquad (166)$$