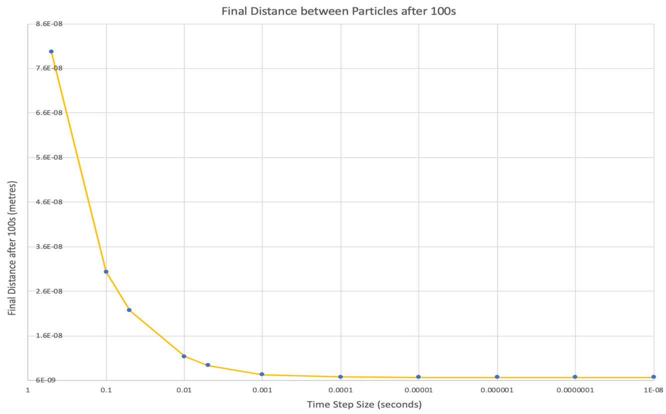
Convergence Plot

The figure to the right shows the relationship between the final distance between the two particles after 100s, and the timestep size used. It is evident from this figure that our algorithm is indeed convergent. As this plot encapsulates both the rounding and discretization errors, and convergence ⇔ consistency (w.r.t discretization errors) and stability (w.r.t rounding errors), we can infer that our scheme is indeed consistent and stable. This is extremely useful, as we cannot easily analytically prove this stability of the Lennard Jones Potential (LJP), because:



1. Our command-line inputs are not exact – due to floating point truncation errors, 1e^{-4} is not read and stored exactly. Therefore, this input error will propagate through the iterations of our algorithm.
2. Our time step is not exact – when using a time step that of the form 10e^{-n} , we introduce further rounding errors, as we cannot store powers of ten exactly (similarly to 1.).
3. It is difficult to prove that taking the derivative is a stable operation – the formula we use to calculate the 'next' position is as follows: $x(t+\Delta t) = x(t) + \Delta t \cdot [v(t)]$ with $v(t) = v(t-\Delta t) + \Delta t \cdot \frac{LJ(x(t-\Delta t))}{mass}$, where $v(t-\Delta t)$ and $LJ(x(t-\Delta t))$ are the velocity and LJP at the previous time step, respectively. Proving that this derivative (the velocity) is stable is difficult using rigorous numerical analysis, as it requires proving the stability of the LJP formula.

Condition Number

As the command-line inputs are non-exact, an analysis of the condition number of the force calculation will be useful. If the force function is ill-posed and the error induced by the non-exact time step propagates unbounded during execution, we will run into significant issues. Here is the calculation of the condition number:

$$cond(x) = \left| x \cdot \frac{f'(x)}{f(x)} \right| = | \frac{x \cdot -4 \cdot \epsilon \cdot \left(-12\left(\frac{\sigma}{x}\right)^{12} + 6\left(\frac{\sigma}{x}\right)^{6}\right) \cdot -\frac{1}{x^2} + x \cdot \epsilon \cdot \left(-12 \cdot -12\frac{\sigma^{12}}{x^{13}} + 6 \cdot -6\frac{\sigma^6}{x^7}\right) \cdot \frac{1}{x}}{-4 \cdot \epsilon \cdot \left(-12\left(\frac{\sigma}{x}\right)^{12} + 6\left(\frac{\sigma}{x}\right)^{6}\right) \cdot -\frac{1}{x}} = \left| -1 + \frac{144 - \frac{36}{\sigma^6}x^6}{-12 + \frac{6}{\sigma^6}x^6} \right|$$

If we let $x \to 0$ we see $cond(x) < 13$, and if we let $x \to \infty$ we see $cond(x) > 7$, and so the $cond(x)$ is bounded at the extremities. The only volatile point of $cond(x)$ is when $x = 2^{\frac{1}{6}}\sigma$, at which $cond(x)$ gives asymptotic behaviour. This is the distance where the Pauli repulsion overcomes the Van der Walls forces, causing the particles to repel one another. It is therefore natural to expect the behaviour of the force to become ill-posed around this point, and a good scheme would adapt its time step accordingly around this point to account for this ill-posedness.

Convergence Order

We can also (numerically) prove the convergence order of our scheme. From the plot above, we can see that our scheme converges linearly (note the logarithmic scale). This can be proven more concretely using the formula given in Lecture 7. Taking $F^{\infty} = 6.72689669354509\text{E}-09$, for $p = 1$ our values remain bounded, but for $p = 2$ our values increase rapidly. Therefore, we have a convergence order of 1, as expected:

Time Step Size (i)	Distance After 100s ($F^{(i)}$)	$\lvert F^{(i)} - F^{\infty} \rvert$	$\frac{\lvert F^{(i+1)} - F^{\infty} \rvert}{\lvert F^{(i)} - F^{\infty} \rvert^{1}}$	$\frac{\lvert F^{(i+1)} - F^{\infty} \rvert}{\lvert F^{(i)} - F^{\infty} \rvert^{2}}$
0.1	3.03323669363363E-08	2.36055E-08	-	-
0.01	1.13772330182955E-08	4.65034E-09	0.197002486	8.35E+06
0.001	7.33577688077338E-09	6.0888E-10	0.130932506	2.82e07
0.0001	6.79041745839881E-09	6.35208E-11	0.104323915	1.71E+08
0.00001	6.73320060432675E-09	6.30391E-12	0.099241733	1.56E+09
0.000001	6.72744735758861E-09	5.50664E-13	0.087352766	1.39E+10
0.0000001	6.72686719154053E-09	2.9502E-14	0.053575324	9.73E+10

In lectures we proved that Explicit Euler has a second order local discretization error and thus second order consistency. Here we show that this second order local consistency reduces to first order global convergence, due to the three pollutions of our algorithm: non-exact inputs, non-exact time steps, and non-exact derivatives.