

Improving Computation Time of
Stochastic Electrodynamics Simulation of
Hydrogen Ground State

CSI702
High Performance Computing
Spring 2010 Project

Student: Doug Reitz
Professor: Dr. John Wallin

Background

Stochastic electrodynamics (SED) is a classical theory that introduces a classical electromagnetic background radiation representation of zero-point (ZP) fields. With ties to concepts and ideas from Plank, Nernst, and Einstein, SED was developed initially in the 1960-70's by Marshall and Boyer. The primary source for this project was from Cole [1]. That paper contains a good set of references and a summary of the background and history of SED. Rather than recreate the wording an excerpt from the abstract [1] summarizes a brief summary background of SED in a compact manner.

'Stochastic electrodynamics (SED) is a classical theory of nature advanced significantly in the 1960s by Trevor Marshall and Timothy Boyer. Since then, SED has continued to be investigated by a very small group of physicists. Early investigations seemed promising, as SED was shown to agree with quantum mechanics (QM) and quantum electrodynamics (QED) for a few linear systems. In particular, agreement was found for the simple harmonic electric dipole oscillator, physical systems composed of such oscillators and interacting electromagnetically, and free electromagnetic fields with boundary conditions imposed such as would enter into Casimir-type force calculations. These results were found to hold for both zero-point and non-zero temperature conditions. However, by the late 1970s and then into the early 1980s, researchers found that when investigating nonlinear systems, SED did not appear to provide agreement with the predictions of QM and QED. A proposed reason for this disagreement was advocated by Boyer and Cole that such nonlinear systems are not sufficiently realistic for describing atomic and molecular physical systems, which should be fundamentally based on the Coulombic binding potential. Analytic attempts on these systems have proven to be most difficult. Consequently, in recent years more attention has been placed on numerically simulating the interaction of a classical electron in a Coulombic binding potential, with classical electromagnetic radiation acting on the classical electron. Good agreement was found for this numerical simulation work as compared with predictions from QM.' [1]

The goal of this project is to help improve the computational performance of the numerical simulation of the interaction of a classical electron in a Coulombic binding potential with a Hydrogen nucleus. Specifically the non-windowing simulation used in [1]. Since computer performance has improved significantly since 2005, it was thought that applying concepts from the course would benefit this simulation by reducing run times making future use of the simulation more practical. The simulation is for the orbit of electron in a binding potential with the Hydrogen nucleus and moving through the background radiation field. As noted [1] the non-windowing computations took days to execute at the time.

None of the background or math material presented here should be considered original. Words, text, and formulas are lifted from [1] throughout this report.

Why is this interesting? Well SED has some appeal in that it introduces a potential physical explanation for some observations. It has has some success predicting results confirmed by observation and QM and QED. However, it has not been thoroughly through the ringer very much to date. There are other ideas out there for SED explaining inertia, mass, gravity, and maybe someday the strong nuclear force. This speculative potential has great appeal, but has a ways to go before being quantified and validated. So testing and validating the theory with a basic Hydrogen orbital model offers a means to run it through the ringer a bit.

Problem Description

The simulation is for a hydrogen atom in ground state. In the absence of the background electromagnetic radiation, the circular orbit of an electron has been shown to spiral into the nucleus reaching $r=0$ within $1.3e-11$ seconds. In SED, the fluctuating motion is accounted for by the ZP electromagnetic radiation described as follows:

$$\rho(\omega) = \frac{\hbar\omega^3}{2\pi^2c^3} = \frac{\omega^2}{\pi^2c^3} \frac{\hbar\omega}{2}$$

Eq.1 - Background Radiation Field Spectrum from [1] p 3

In SED, typically the electron is treated as a point particle following the classical relativistic Lorentz-Dirac equation where z^μ is the four vector space-time position, m is the particle normalized mass, q is the charge, T is the particle proper time, and F^μ is the sum of all four-vector forces acting on the particle. In SED atomic problems, this is typically the binding potential, the Lorentz force due to radiation fields, and other applicable external forces.

$$m \frac{d^2 z^\mu}{d\tau^2} = \frac{2q^2}{3c^3} \left[\frac{d^3 z^\mu}{d\tau^3} - \frac{1}{c^2} \left(\frac{d^2 z^\lambda}{d\tau^3} \frac{d^2 z_\lambda}{d\tau^3} \right) \frac{dz^\mu}{d\tau} \right] + F^\mu$$

Eq.2 - Relativistic Lorentz-Dirac EOM from [1] p 5

For this simulation where an assumed thermal spectrum is present, the radiation field are represented as follows as a sum of plane waves:

$$\mathbf{E}(\mathbf{x}, t) = \sum_{n_x, n_y, n_z = -\infty}^{\infty} \sum_{\lambda=1,2} \frac{\hat{\mathbf{e}}_{\mathbf{k}_n, \lambda}}{(L_x L_y L_z)^{1/2}} [A_{\mathbf{k}_n, \lambda} \cos(\mathbf{k}_n \cdot \mathbf{x} - \omega_n t) + B_{\mathbf{k}_n, \lambda} \sin(\mathbf{k}_n \cdot \mathbf{x} - \omega_n t)]$$

Eq.3 - Radiation Field Sum of Plane Waves from [1] p 6

With periodic boundary conditions:

$$\mathbf{k}_n = \frac{2\pi n_x}{L_x} \hat{\mathbf{x}} + \frac{2\pi n_y}{L_y} \hat{\mathbf{y}} + \frac{2\pi n_z}{L_z} \hat{\mathbf{z}}$$

where n_x , n_y , and n_z are integers, $\omega_n = c|\mathbf{k}_n|$, $\mathbf{k}_n \cdot \hat{\mathbf{e}}_{\mathbf{k}_n, \lambda} = 0$, and $\hat{\mathbf{e}}_{\mathbf{k}_n, \lambda} \cdot \hat{\mathbf{e}}_{\mathbf{k}_n, \lambda'} = 0$ for $\lambda \neq \lambda'$.

Eq.4 - Radiation Field Boundary Conditions from [1] p 6

A and B in Eq.3 are a Gaussian distribution. As show in [1] for a thermodynamic equilibrium condition these correlate by a function of frequency and temperature as follows:

$$\rho(\omega, T) = \frac{\omega^2}{\pi^2 c^3} \frac{f(\omega, T)}{4\pi}$$

with $f(\omega, T) \rightarrow 2\pi\hbar\omega$ as $T \rightarrow 0$.

Eq.5 – Spectral Energy Density Correlation from [1] p 6

And last but not least the Coulombic binding potential between the electron and the nucleus:

$$m \frac{d^2 \mathbf{z}}{dt^2} = -\frac{e^2 \mathbf{z}}{|\mathbf{z}|^3} + \frac{2 e^2}{3 c^3} \frac{d^3 \mathbf{z}}{dt^3} + (-e) \left\{ \mathbf{E}[\mathbf{z}(t), t] + \frac{\dot{\mathbf{z}}}{c} \times \mathbf{B}[\mathbf{z}(t), t] \right\}$$

Eq.6 – Coulombic Binding Potential from [1] p 6

The following illustrates the difference between classical electron radius with ZP radiation versus no radiation considered:

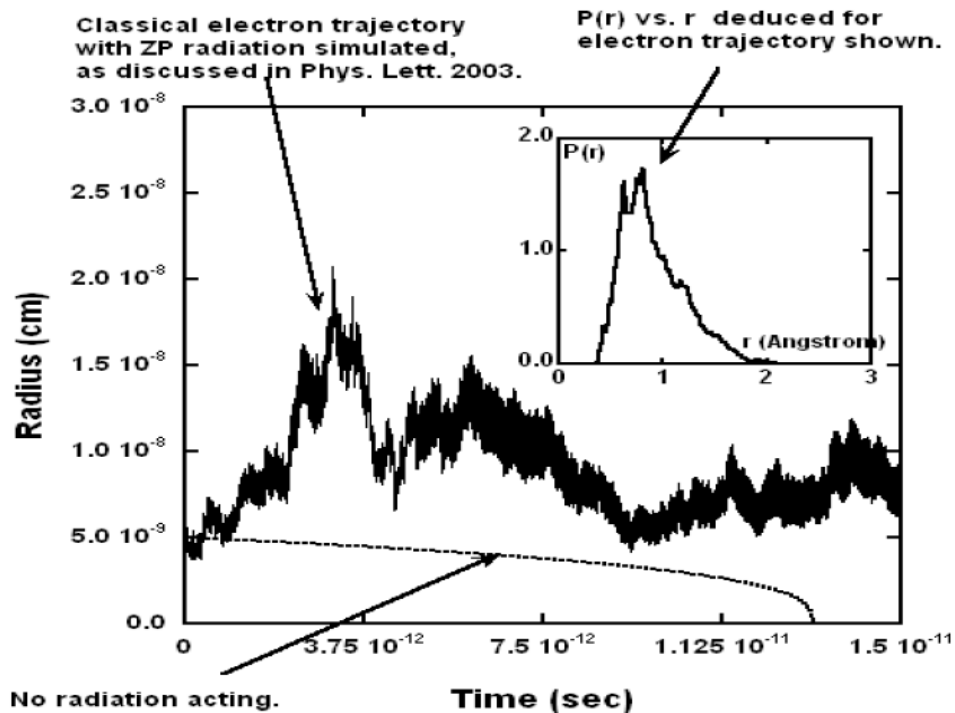


Figure 1 – From [1] p 8

The Simulation

The author of [1] was contacted to request the simulation code. He was able to provide a code that was thought to be the non-windowing one described in the paper. The paper also goes into simulation results with a windowing approximation, but for this project only the non-windowing approach is considered.

The simulation consists of creating the random background radiation represented by plane waves using of random number generator. The equation is then solved using a variable step size 5th order Runge-Kutta method with the summation of electric and magnetic fields computed at each time step. The simulation runs from $t=0$ to $t=1.6e-11$ seconds. This is to demonstrate that the electron does not collapse into the nucleus prior to $t=1.3e-11$ sec as would be the classical case in the absence of a radiation field. The electron r_0 is 0.53A at t_0 . The electron is then allowed to move subject to the binding potential and the background radiation. The computed electron radius probability distribution is output to file at regular intervals. The results if accurate should be representative of the ground Hydrogen distribution computed from the Schrödinger equation.

The time steps are small and the simulation takes a long time to get results. In [1] the simulation results were only presented up to $5.6e-13$ s, and the runs were done using nearly '900 CPU days of processing'. There were several different initial configurations presented in the paper so the amount spent on a single simulation is not recorded. For this project we are focused exclusively on the r_0 of 0.53A and a time range of 0 to $1.6e-11$ s.

The Serial Approach

The serial approach utilized a single thread. The initial code was able to compiled and run out of the box as received. Here are the sampled execution times running from t 0 to 2.0e-10 s

System 1	>16.6 hours
Gmice	>48 hours (did not complete)

The reason for the gmice larger execution time is due to the nature of the computation, when the electron is closer to the nucleus, the computation time increases dramatically. It should be noted that the code did not provide valid results initially on gmice with default optimization. This is attributed to the loop vectorization of the Intel compiler resulting in different computation sequences causing information loss. A bunch of time was spent trying to figure out what was going on with gmice and why the results were not the same. As noted in [1], the simulation is subject to large errors, “ionization” effects, due to numerical instability. In these cases the electron collapses into the nucleus or is ejected beyond 3A. These instabilities were attributed to insufficient numerical precision due to the different ordering of math operations with the vectorized loops. In the hopes of mitigating this precision issue, many man hours were spent trying to convert all floating points to long doubles and also adjusting the scientific constants to maximum known significant digits. The effort to migrate to long doubles failed to achieve the desired result. Numerical stability across both platforms remained elusive in the time allotted for the project. Since the instability took 16+hours of run-time to determine, this approach was eventually shelved prior to the presentation. The selected approach was to stay with doubles for floating points values, and the scientific constants were adjusted slightly within the differing published values or numbers of significant digits. Also gmice availability was limited in the closing days of the project, so other available spare computer assets were utilized that offered performance comparable to a single gmice node. These systems are System 2 in the Computer Systems Appendix.

Following the class presentation, the long double approach was completed with a run on gmice in the serial implementation. The run took over 50 hours to complete. These results are shown in the Parallel Implementation & Results section.

The serial implementation as recorded in the performance comparison, benefited from some non-parallelization optimizations that were done when developing the parallel implementation. Those optimizations are covered in the parallel approach section.

Overview of the Serial Code

- Allocate and initialize global storage and constants
- Initialize random plane waves
- Solve using 5th Order Runge-Kutta
 - until done
 - rkqs -> rkck -> computes loops -> calls derivs (5 times each pass)
 - derivs is the radiation field sum of plane waves

Optimization analysis:

The derivs() function is called many, many times - look at ways to optimize that function
Eliminate repeated allocation and deallocation of memory where possible

Reuse computed information where possible

The Parallel Approach & Results

The first step taken was to optimize the serial code. As noted in the Serial section, the following items were analyzed.

1. Derivs is called many, many times look at ways to optimize that functional
2. Eliminate repeated allocation and deallocation of memory where possible
3. Reuse computed information where possible

The derivs() function was optimized to eliminate repeated computations. Since we are in a loop inside a function that is itself called many times for each iteration, efficiency here is critical. Here the computation of the sin and cosine of theta and the square root of i are only computed once reducing these operations by half. (illustrated below)

```
for(i=Nmin; i<=Nmax; i++)
{
    // dreitz - optimize
    // Ex=sqrt(i)*(Amplitude1[i]*cos(theta)-Amplitude2[i]*sin(theta))+Ex;
    // Ey=sqrt(i)*(Amplitude3[i]*cos(theta)-Amplitude4[i]*sin(theta))+Ey;
    const long double& theta=omega*i*x;
    const long double& s=sin(theta);
    const long double& c=cos(theta);
    const long double& sqrt_i=sqrt(i);
    Ex += sqrt_i*(Amplitude1[i]*c - Amplitude2[i]*s);
    Ey += sqrt_i*(Amplitude3[i]*c - Amplitude4[i]*s);
}
```

There were also several storage arrays in rkqs and rkck that were being dynamically allocated. These were changed to a single allocation and the storage space was reused thereby saving the expense of allocating and deleting each function call. Rkck also had a repeated loop where the work was combined into a single loop.

The next challenge was how to parallelize a 5th order Runge Kutta method. There are 6 very small loops in rkck, however these are only loops of 4. As a result the expense of synchronizing threads or sending data to/from other nodes was estimated to be much larger than any potential benefit of parallelizing a loop of 4 with only one or 2 statements. Therefore no parallelization was done here.

The one area identified for parallelization was in derivs, where the summation over plane waves occurs. This loop is on the order of 10000 or more so parallelization here is worth the synchronization cost. There are two possible loops. Only one or the other loop is encountered per pass. When analyzing the means to parallelize, OpenMP was identified as a good candidate. It provides an easy way to parallelize the loops and provides easy syntax for reduction and synchronization. MPI was not pursued, but may be a viable future option, particularly if the plane waves density is configured to be greater than that it was for this project. Below is how the loop was parallelized.

```
#pragma omp parallel for schedule(static,10) reduction(+:Ex) reduction(+:Ey)
for(i=Nmin; i<=Nmax; i++)
{
    const double& theta=omega*i*x;
    const double& s=sin(theta);
    const double& c=cos(theta);
    const double& sqrt_i=sqrt(i);
    Ex += sqrt_i*(Amplitude1[i]*c - Amplitude2[i]*s);
    Ey += sqrt_i*(Amplitude3[i]*c - Amplitude4[i]*s);
}
```

As you can see the two summations leveraged the reduction feature of OpenMP. Static scheduling is shown and estimated that it should minimize synchronization overhead, and because the work per thread is fixed this can be predetermined before executing the loop. In some runs schedule(dynamic, 1536) was also used.

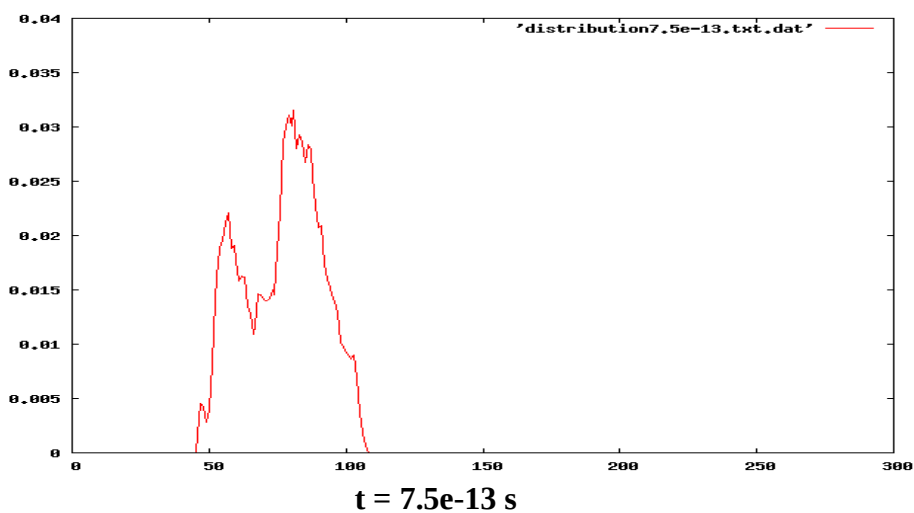
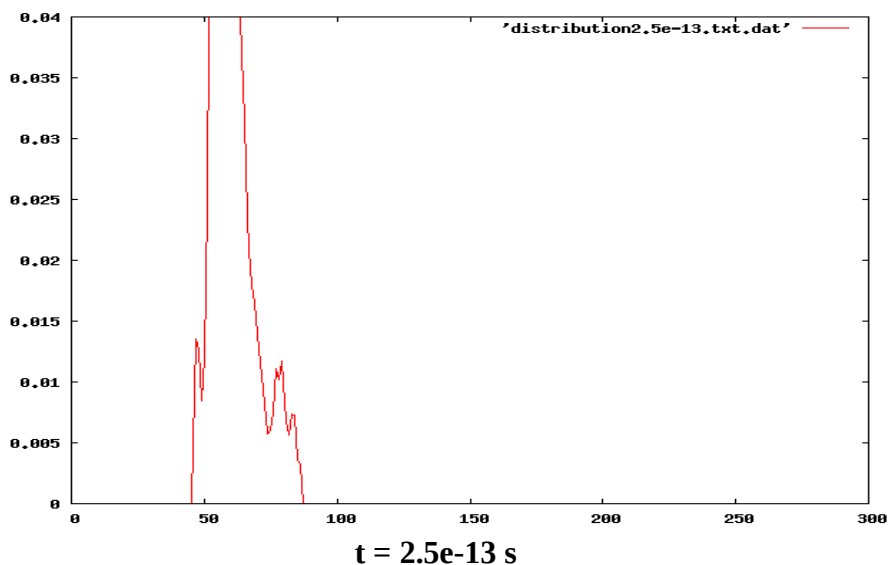
Parallelization of these two loops does make a difference in the simulation runtime as explained in the

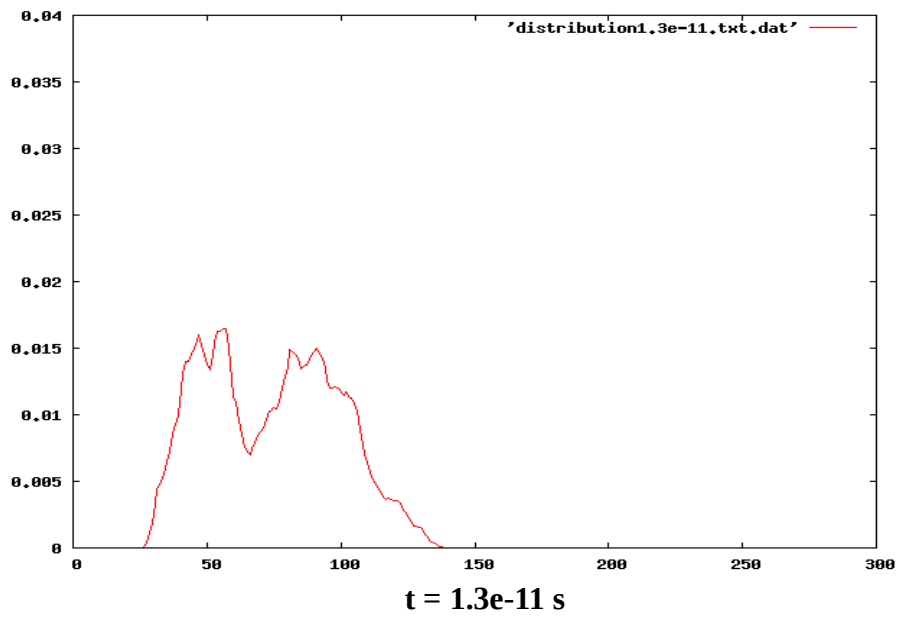
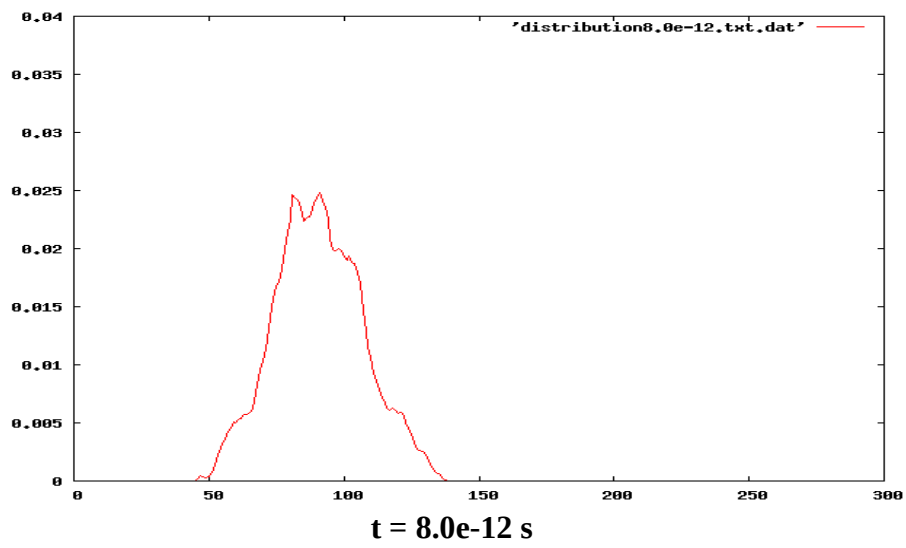
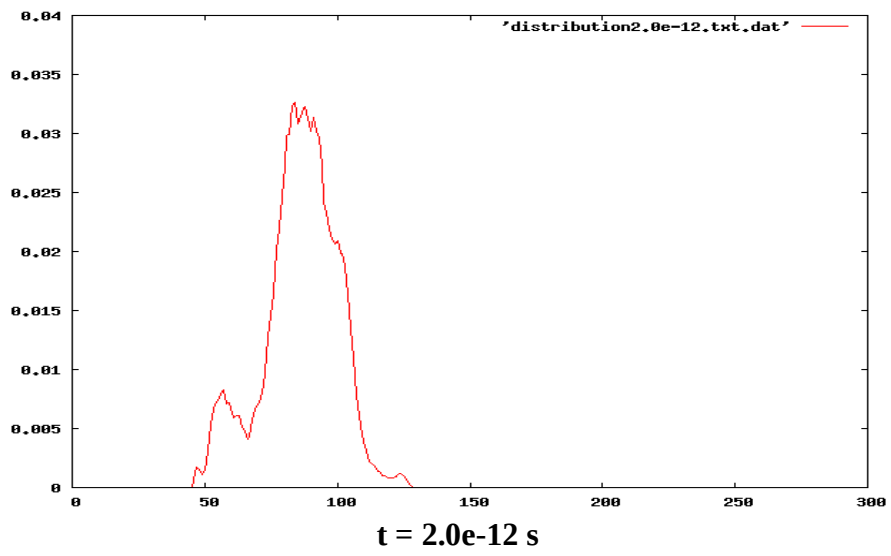
Performance Comparison section.

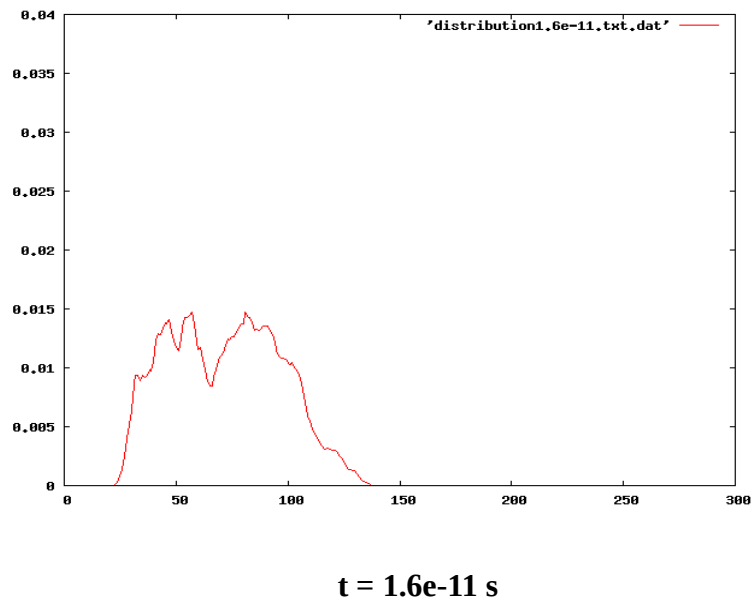
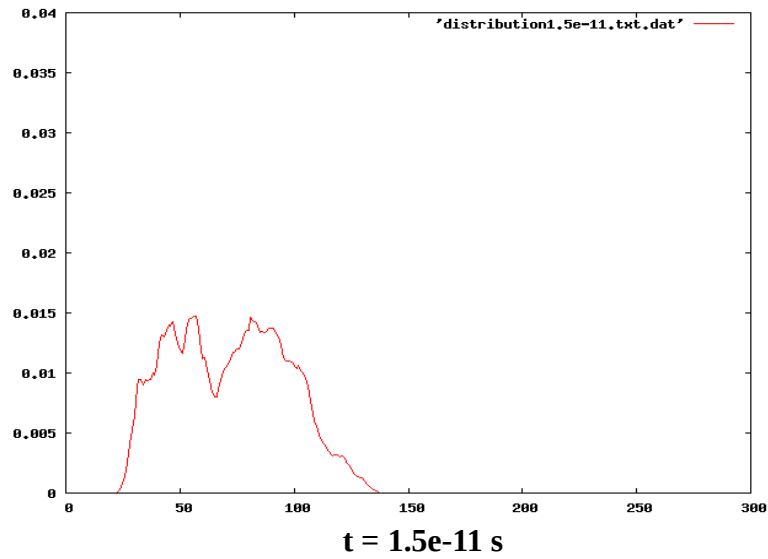
For the Parallel implementation running on System2 type systems. The complete runtime for simulation from $t = 0$ to 1.6×10^{-11} s was 39054 seconds or 10.8 hours. The number of threads used was 8 - matching the number of computing cores (or those counted as cores due to Hyper-Threading capability)

The results were processed using `plot_animate.sh` which was written to process the probability distributions at each major time-step. It uses gnuplot to generate gifs of the plots which are in turn placed into a single rudimentary animated gif.

And now the moment you've been waiting for... the simulation results of the electron radius probability distribution vs. time: The y axis is the probability and the x axis is the radius in hundredths of Angstroms.





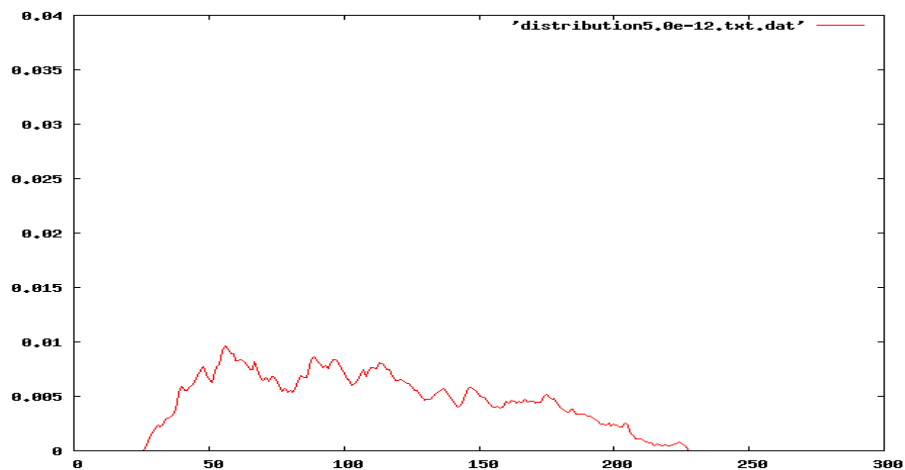


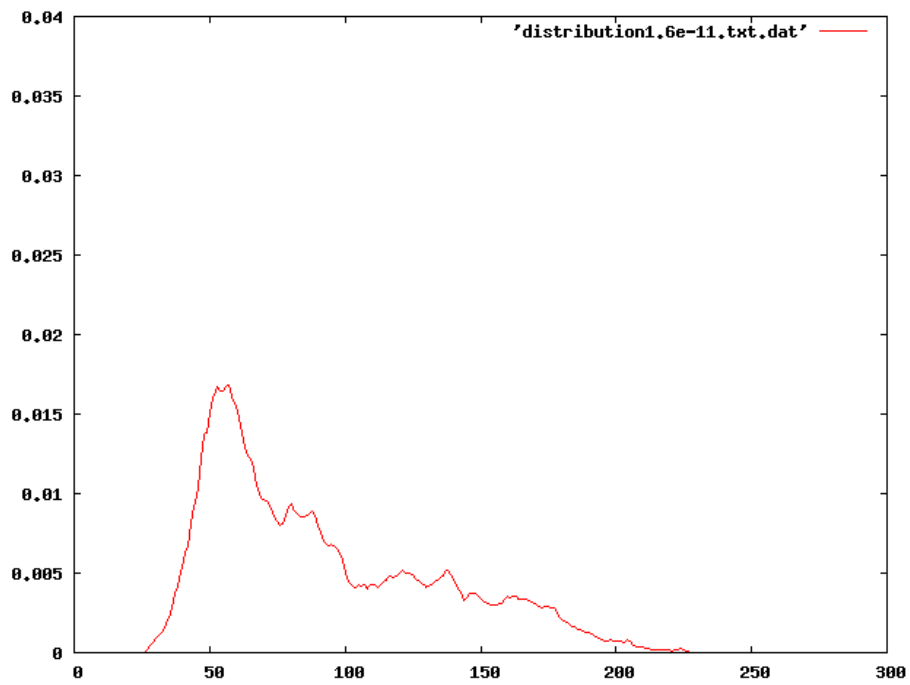
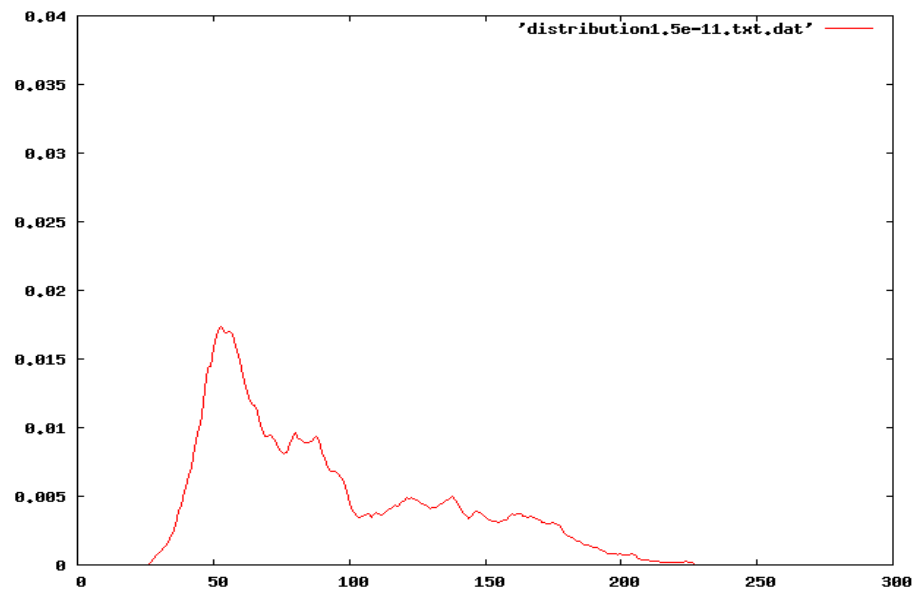
If you look at the animation, you will notice that the probability appears to be building up to peaking near the Bohr radius 0.53\AA before the sequence ends.

An animation of the the results is available on the course site:

<http://csi702.net/csi702/images/Hsed-animate-parallel.gif>

Results from a serial implementation using long doubles on gmice are below. This run took approximately 50.1 hours to reach $t=1.6e-11$ s. The y axis is the probability and the x axis is the radius in hundredths of Angstroms.





An animation of the the results is available on the course site:

<http://csi702.net/csi702/images/Hsed-animate.gif>

Speed Up Comparison

With the use of threading provided by OpenMP the parallel version can scale on SMP systems leveraging the number of cores available on the system. Comparing the distribution results between parallel and serial runs performed for the project is difficult. The parallel results will differ from the serial results due to precision loss due to a different mathematical operation sequence. The results are very sensitive to slight variations in precision. As a result the probability density in one case may be closer to nucleus, which increases the computation time dramatically. Therefore, a high precision definitive performance comparison is not possible.

The comparison we can make is for the a serial run of 16h which ran without collapse into or leaving the nucleus. The distribution in that run favored distances farther from the nucleus than the parallel version so the computation was less intensive so this is not a real good comparison.

Comparing these two runs we saw a speed up of approximately 1.5 from 16.6 hours to 10.8 hours. If we compare the run time of the gmice serial implementation using long doubles we saw a speed up of more than 4.5 times from 50.1 hours to 10.8 hours. For a comprehensive speed up analysis, more data needs to be collected. The simulation needs to be further analyzed and verified to provide consistent results in parallel and serial. Careful implementation of the parallel implementation should minimize the variance in results unless there is an underlying precision issue with the serial implementation.

Lessons Learned

The biggest lesson learned was that when working with codes that take a long time to run it is advisable to start very early and take the time to understand the code thoroughly before trying to adjust things. The brute-force pursuit of applying higher precision wherever possible did not lead to stable results quickly as hoped, but it did provide valuable learning and warrants further pursuit. The addition of the OpenMP parallelization needed to be thought through and took a few iterations to get it to the final solution. And last but not least, with simulations taking over 16 hours, patience and access to computing resources is essential. The more computing resources the better to evaluate permutations. It is also necessary to keep a good record of the permutations tried and the results.

Recommendations

There is still work to be done with this simulation. The variability and disagreement in the parallel results indicates that there are mathematical precision factors included in the results. It is expected that the precision error is also reflected in the serial results (although the serial results are consistent on the same computational platform). The tendency to be unstable with slight variations in precision needs to be further analyzed, understood, and eliminated or reduced if possible. Once it is more robust and the precision issues can be removed from the results, this simulation could be used for other further analysis and theory development.

References

[1] D. C. Cole, "[Simulation results related to stochastic electrodynamics](#)," published in AIP Conference Proceedings Vol. 810, No. 1, pp. 99-113. The international conference was entitled "Quantum Theory: Reconsideration of Foundations-3," and was held June 6-11, 2005, at Växjö University, Sweden. Proceedings edited by G. Adenier, A. Khrennikov, and T. Nieuwenhuizen.

Appendix

Computational Systems:

System 1: Dual Core CORE 2 Duo (2.33Ghz, 4MB L2 Cache)
Linux x86_64

System 2: Nehalem Xeon Quad Core Hyper-threaded (2.53GHz, 8MB L2+L3 Cache)
Linux x86_64