Investigating Zero-Point Energy in a Water Trimer with Diffusion Monte Carlo

Will Solow, Skye Rhomberg, Lindsey Madison, and Eric Aaron*

Department of Computer Science, Colby College, Waterville, ME

E-mail: whsolo22@colby.edu

Abstract

We give an overview of the Diffusion Monte Carlo (DMC) algorithm and its applications into finding solutions to the Schrödinger Equation when no analytical solution is available. An implementation of the DMC algorithm is presented with a specific focus on understanding the behavior of a three-water molecule system. Given the same data structure, a four-dimensional NumPy array, we demonstrate how a fully vectorized implementation is at least 35 times faster than a traditional for loop implementation, and how to validate an inherently less obvious implementation. In such molecularly complicated systems, we show how the role of timestep factors into both the population of walkers and the calculated Zero-Point energy of the system. Following up, we give necessary criteria to find the Zero-Point energy to the desired precision, and show the wave function, a normal distribution of bond lengths or angles within the studied molecular system. Finally, we show the difficulty of equilibrating a complicated system like the water trimer, and we illustrate techniques for how build an equilibrated system when the DMC algorithm is not sufficient on its own when the system is initialized with random values.

Introduction

The Schrödinger equation has been at the heart of physical chemistry given how it describes the wave function of a quantum system. Since its introduction, it has advanced the knowledge of quantum mechanics by allowing chemists to understand subatomic forces between atoms and molecules. However, solving the Schrödinger equation to determine the energies of a molecular system is difficult, and in practice can only be solved in a few simple molecular systems. The Schrodinger equation can explain many of the physical properties of water are of great interest and have been studied extensively, yet many of the properties of water are still not fully understood.

The Diffusion Monte Carlo[?] (DMC) method allows for the Schrodinger wave function to be approximated using Monte Carlo methods given that the Schrödinger equation can only be solved analytically for a few specific molecular systems. For complex, many bodied systems, the DMC model is one of the few options as its computational complexity scales well large input sizes. Even so, it remains a computationally hefty model for complex systems, even as the best option.

As a complex molecular system, water is an ideal candidate to be studied using the DMC model. In fact, complex water structures, namely the hexamer, have been studied, albeit with simplifying assumptions. The DMC model is not without its limitations, however, and struggles with both population size and time step errors.

To alleviate some of the computational load in a traditional DMC implementation, we present a Python based implementation utilizing the NumPy library. NumPy gives tools to work efficiently through multidimensional matrices by splitting up the computational load amongst the cores of a computer's processor to allow for simulations to be run faster. This method is not only faster, but allows for the implementation of the DMC model to be generalized to a variety of homogeneous molecular systems.

As such, the water molecule and the water trimer molecular system were investigated in order to find the wave function given by the Schrodinger equation and the Zero-Point

Energy (ZPE) of the system. Using this novel DMC implementation, we provide necessary simulation constants in order to accurately compute the wave function of the Schrödinger equation and ZPE within a desired variability. We also show how the selection of time step within the DMC model plays an important role in the convergence of both the walker population and the ZPE within the water molecule and water trimer systems.

The specific aims of this project were to 1) employ a novel analytic implementation approach to study many-bodied molecular systems with the DMC model; 2) investigate the influence of different simulation constants on the ZPE for a single water molecule and water trimer; 3) determine the efficiency of the novel analytic implementation approach when compared with more traditional implementations.

Methods

The methods for this project will be presented in the following sections: a description of the existing analytical model; the implementation and validation of a new analytical model that includes three simulation constants; and lastly the data analysis employed to achieve the aims of the project.

The Model

Pseudocode for the DMC model was received from Professor Lindsey Madison (see Appendix 1). The important constants in the model are dependent on the system being simulated; for this project which involved modeling a water molecule, the constants are: 1) the time step; this constant controls the distance over which the walkers can propagate during an iteration of the simulation loop; 2) the duration of the simulation; this constant is typically dependent on the time step in order to run the simulation for the same amount of real world time; 3) the number of copies of the same molecular system, also called the number of walkers in the simulation; 4) the masses of the molecules in the system; and 5) the equilibrium positions for

the bond lengths and bond angles present in the system, typically calculated experimentally before use is the model.

The time step dictates the distance by which each coordinate in each atom in each molecule in each walker moves from its starting position at the beginning of the iteration. Additionally, the time step also influences the acceptable range of the reference energy at each iteration of the simulation loop as well as the threshold for which an individual walker is replicated or deleted within the model.

The duration of the simulation is how many iterations of the simulation loop are run before the wave function and zero-point energy are calculated. Typically, a longer simulation gives more time for the molecular system to equilibrate. With smaller time steps, a longer simulation is necessary as equilibration takes many more iterations given that the atoms are moving much slower towards equilibrium because they are only able to move over shorter distances.

The number of walkers determines the size of the walker list and can impact the range of an acceptable reference energy which, in turn, determines replication and deletion of walkers. Reference energy is determined as the average of the potential energy over all walkers plus a statistical constant to penalize populations of walkers larger than the initial value. The reference energy is used to calculate a 1000-step rolling average of the reference energyR which converges to the ZPE. After equilibration, the average of the rolling average, is used to calculate the ZPE of the simulation. We define this average as the ZPER as it is the average of the rolling average of the reference energy.

Typically, an equilibration phase is run with a duration based on the time step. Then, 10 to 20 production phases, which are repeated simulations of the DMC model, are run, each of which calculates a ZPE. The average of the ZPE over these production phases we define as the ZPEG and is considered the computed ZPE of that trial. The ZPEG, along with the wave function, which is calculated based on the distribution of the bond lengths or bond angles in the molecular system, are considered the outputs of the DMC model. These

outputs are then used as an approximation of the solution to Schrodinger's equation.

Implementation

The implementation of the new model described above is broken down into three sections. In the first section, the initial constants are described, with attention only paid to the initial walker array. In the second section, the simulation loop is described and how it relates to the DMC model. In the final section, the researchers give an overview of the potential energy function. The implementation was broken down into these three sections so that the code could be generalized to multiple systems. The way the potential energy function is utilized in the simulation loop ensures that only the potential energy function needs to be changed when moving between molecular systems of study.

To implement the new model described above, the researchers chose to store the walkers in a four-dimensional (4D) NumPy ndarray object. The researchers chose to use NumPy given that NumPy's functions can operate quickly on multi-dimensional arrays. The 4D array had axes of the number of walkers by the number of molecules by the number of atoms in each molecule by the Cartesian coordinates (x, y, and z in 3D space). As a result, the researchers hypothesized that the 4D array implementation could generalize to any system of homogeneous molecules. In this project, this data structure allowed the researchers to run trials on the single water molecule system and water trimer system without making any changes to the simulation loop. It should also be noted that this 4D array implementation allowed the code to be very readable. A 2D implementation was considered, with axes being the number of walkers by the coordinates of every atom in the system. However, the structure that the 4D array provides allows for quick slicing and broadcasting of the array which made the code more easily understood.

In the simulation loop implementation, the researchers directly followed the pseudocode presented by the model. NumPy is able to operate on a large array of data simultaneously such that it was not necessary to use for loops to access individual data points within the 4D

array. The atoms in each walker are propagated by a pseudorandom number in the normal distribution of $\sqrt{\text{dt/atomic mass}}$. When studying the water molecule system, researchers adopted the convention that the atom dimension of the 4D walker array would have the oxygen, hydrogen, hydrogen atoms in the same order each time. Given that each atom appeared in the same index in each walker in the 4D array, the NumPy tile function was used to broadcast the range of the normal distribution to each coordinate in the 4D array. This approach allowed for a one line propagation step within the simulation loop.

Another point of interest in the implementation of the simulation loop was the walker replication and deletion step. Based on the potential energy of each walker compared to the overall reference energy of the system, and a random threshold number, the walker was selected to be either replicated or deleted from the array. The index of each such walker was converted into a one-dimensional array of booleans with a True value indicating that the walker should remain in the final array. NumPy arrays support boolean indexing, so the code then simply returns only the walkers that are not deleted and a copy of the walkers that are replicated.

The potential energy functions for the single water molecule and water trimer were perhaps the most interesting part of the simulation and were where the researchers could demonstrate the powerful utility of NumPy. The potential energy functions were based on the code provided by Professor Lindsey Madison. The researchers spent a great amount of time understanding what the function did and brainstorming how to do those operations more efficiently in NumPy. The result was innovative yet unintuitive calculations that lent themselves to very efficient potential energy function calculations. For example, consider the line below within the water molecule intermolecular potential energy:

The key intuition is that the input to the potential energy function is a 4D array of walkers. To calculate the intermolecular potential energy, the function must compare the distances between the atoms of every distinct pair of water molecules in the system. The researchers first used the NumPy array indexing to obtain arrays that have the distinct

pairs (molecule_index_a and molecule_index_b). Then, using arrays that have the distinct pairs, notice that matrix multiplication behaves very similarly to distance calculation in that the corresponding elements are multiplied pointwise and then summed. However, pointwise multiplication is not exactly what is desired given that the distance between the two vectors is variable of interest. NumPy does matrix multiplication by broadcasting the array to a higher dimension, performing pointwise multiplication, and then summing along the extra axes resulting in a smaller array of the correct size. This is exactly what the code demonstrates, and the resulting array has the following dimensions: number of walkers by number of distinct pairs by number of atoms by number of atoms, where each datapoint is the distance between the atoms.

Code Validation

The next step was to validate the simulation loop separately from the potential energy function by using a CO harmonic oscillator as the ZPE of the CO molecule has been calculated experimentally. If the calculated ZPER of the system converged to the experimentally calculated ZPE, then the researchers could verify that the simulation loop was functioning correctly. The potential energy function of a harmonic oscillator system is simply $.5k(x - xo)^2$, or the displacement of the oscillator from its equilibrium position. Without having to consider the accuracy of the potential energy function, the researchers were able to focus on validating the simulation loop.

The simulation loop was validated in a variety of ways. The wave function, the main output of the DMC algorithm, is known and is given by the equation: $N \cdot exp[-(r^2)(\sqrt{k} \cdot mass)/2]$ When the implementation of the DMC algorithm is working correctly, the density histogram of the CO bond length will form a normal distribution that converges to the given wave function. Additionally, the ZPE of the CO molecule system is known to be .00494317, which has been calculated experimentally. The implementation of the DMC algorithm is considered correct if the 1000-step rolling average of the reference energy converges to this

ZPE within the range of $1 \cdot 10^{-4}$.

Given how NumPy was utilized in the implementation of the model, it was not self-evident that the replication and deletion of the walkers were mutually exclusive. The model would be ineffective if the same walker was both replicated and deleted in the same time step. To verify that this was not the case, auxiliary code was generated, and the replication and deletion was tested using unit values, or values where the expected outcome is known. These values are useful as it is already known ahead of time which values should be replicated and deleted based on the way the test is set up. Thus, the result of the replication and deletion steps were able to be compared to the expected outcome to verify that they were the same.

Finally, the propagation of the walkers was tested given that the NumPy tile function was utilized, which is not a traditional implementation. Again, the researchers turned to unit testing to verify that the code written was working correctly. Given that the propagation step chooses from a normal distribution based on the mass of the system, masses with drastically different magnitudes would yield very different results. As such, by printing an array of randomly generated propagations using masses with different magnitudes, the researchers were able to confirm that the correct index in the 4D array was being propagated by the correct magnitude.

With the simulation loop validated, the researchers turned to the potential energy function in the single water molecule which calculates the intramolecular potential energy of the system. Our collaborator Dr. Lindsey Madison provided Python code that had already been validated as correct in corroboration with experimental results on the water molecule. The researchers' goal was then to replicate the outputs of Dr. Madison's code on different inputs. Once this was done, the intramolecular potential energy function was assumed to be correct. Given that the OH bond in a water molecule behaves much like a harmonic oscillator, the researchers were able to confirm that the convergence of the water molecule gave an OH bond length in the normal distribution given by the wave function of the corresponding harmonic oscillator.

This process was repeated for the intermolecular potential energy in the water trimer. The researchers can confirm that the output of the code matches previous versions of the code up to the 18th decimal place. However, the ZPE convergence behaviour of the water trimer is an open research question. It is expected that the Oxygen atoms in the trimer form a equiangular triangle, so a normal distribution of Oxygen angles centered at 60 degrees would be expected. However, it remains to be seen what simulation constants are sufficient to observe this behaviour given the stochastic nature of the DMC algorithm.

Simulations

To investigate the ZPE of the water molecule and water trimer, the researchers employed the following strategy. Using the simulation loop, an equilibrated array of walkers was created and stored for later use. At the beginning of each trial, this equilibrated array of walkers was put through another equilibration phase with a duration based on the time step. A system in equilibrium will not deviate far from equilibrium; it will only move around slightly. This serves to randomize the start of our production phases to ensure that the data that we generate are from a random distribution of possible molecular system configurations.

The researchers ran trials with timesteps of [10,5,1,0.5,0.1] with walkers [1000,5000,10000] over 10 or 20 production phases. In each trial, a walker array was equilibrated and then put through a series of production phases to determine the ZPEG. Over the 10 trials for each of these variables, an average of the ZPEG was calculated which is the ZPET. Calculating the ZPET in this case is not equivalent to calculating a ZPEG with 100 production phases. This is due to the fact that each ZPEG is calculated off of one initial equilibration phase which the production phases are run from. Also, the researchers were interested in the standard deviation of the ZPEG calculated as well, so by computing the ZPET over the 10 trials, the researchers could understand how the simulation constants affected both the variability in the ZPEG as well as its computed value.

Using the existing implementation of the intramolecular potential energy function given by Professor Lindsey Madison, and the presented NumPy implementation, simulations were run and obtained similar results, further confirming the validation of the newer model. The run time for the simulations using both models were recorded and compared by an empirical analysis over varying populations of walkers and simulation iterations. The time step was kept constant at .1, given that such a time step gives a walker population convergent to 99.99% of initial value.

Results

By decreasing the time step or by increasing the number of walkers, the ZPET (the average of the ZPEG over 10 trials) increased between time steps of 10 to 1.0 and then plateaued at time steps lower than 1.0. However, note that at a time step of 10, the ZPET is significantly lower than the ZPET produced by smaller timesteps. Until recently with the exploration of the water molecule system, a time step of 10 had been considered sufficiently small to produce accurate results which is not supported by our results. In contrast, at the smaller time steps (e.g., 0.1 and 0.5), the ZPET stays constant, implying that it is in a desirable range. This result supports our findings that smaller time steps should be used in the modelling of more complicated systems.

In addition to the calculated ZPET of the system, the standard deviation of the ZPEG for each time step and walker population was investigated. While the overall average (the ZPET) is important, a small variance is also desirable as it demonstrates that the simulation produces the similar results consistently. In Figure 2B, we show how the standard deviation consistently decreases with both an increase in walker population and an increase in the number of production phases over which the ZPEG is calculated. However, of interest is that a larger time step drastically decreases the standard deviation of the ZPEG, which will be highlighted in the Discussion.

In Figure 3A, we show the equilibrium walker population at the end of the production phase compared to the ZPEG calculated. The percent of walkers retained at the end of the simulation remained virtually unchanged with more production phases or more walkers; thus, Figure 3A only shows the data for 10000 walkers over 20 production phases. We concluded that a decrease in time step is directly correlated to an increase in the remaining walker population. A retained walker population of 100% is ideal. Furthermore, the connection between walker population and ZPET is illuminated in that fewer remaining walkers results in a smaller ZPET, which is assumed to be more inaccurate.

In contrast, notice in Figure 3B that a smaller time step correlated with a larger standard deviation. That said, a smaller time step gave a presumably more accurate ZPET, and combined with a convergence of the ZPET calculated after a time step of 1, we saw that such a smaller time step was a necessary condition for calculating the correct ZPET value. Note that the standard deviation was still minimal with a small time step, and can be lowered by adding production phases or more walkers. Typically, a smaller time step would be associated with more accurate results and a smaller variance. Here we see more accurate results in a higher ZPET, but also see the variance increase. Reasons for this are highlighted in the Discussion section.

By incorporating the novel analytic model, the run times improved from 32-45 times faster than the existing model for simulating a single water model (see Table 1). Table 1 shows the rate at which the Solow-Rhomberg intramolecular potential energy function improved over the Madison intramolecular potential energy function. All simulations were run using the simulation loop generated by the researchers. 10 trials were run for each walker value and simulation duration to obtain an average. Notably, for 10000 walkers over 10000 time steps, the Solow-Rhomberg potential energy function yielded an average run time of 66 seconds while the Madison potential energy function yielded an average run time of 2644 seconds, giving an improvement rate of 39.75.

The efficiencies gained by using NumPy have implications for expanding the model to

a water timer and clathrate hydrates. To obtain a meaningful result from the water trimer takes around an hour. We would expect that more complicated molecular systems take longer given the extra computations required in the potential energy function. As such, this run time improvement only becomes more meaningful.

Discussion

Based on our results, we demonstrated that a time step of at most 1 is necessary to properly collect ZPEG data on the water molecule. We also demonstrated that both 10 and 20 production phases produced the same ZPEG; however, 20 production phases also lowered the standard deviation among the 10 ZPEG values that were calculated. More walkers lowered the standard deviation as well and impacted the ZPEG calculated when the initial walker population was less than 5000. Finally, we saw how the remaining walker population impacted the ZPEG and how it corresponded to the time step. All of this points to the fact that a smaller time step was better when modelling more complicated molecular systems.

The importance of a correct time step cannot be understated. With a smaller timestep the ZPEG calculated is more accurate. Moreover, the aim of the DMC algorithm is also to compute the wave function, or the normal distribution curve that the Schrodinger equation gives. With a larger timestep, the walker population is not convergent to its initial value, and so with fewer walkers, the density histogram created does not converge to any sort of normal curve. As a result, the wave function cannot be calculated with a non-convergent walker population which arises from an incorrectly chosen time step. It should also be noted that a larger walker population gives rise to a histogram that converges closer to a normal curve. This is due to the fact that a larger sample size is more accommodating for small irregularities in the population.

We mentioned in our results that a smaller time step leads to a larger standard deviation in the ZPEG calculated. At first, this finding appears counterintuitive. However, in the

simulation loop, the walkers are propagated in a normal distribution in the range sqrt(time step/mass). A smaller time step means that each atom in each walker can only move a smaller distance in each iteration of the simulation; this fact supports why the number of iterations needs to be normalized to the time step. In a complex molecular system like the water molecule, the length of both OH bonds and the HOH bond angle are enforced. With a larger time step, more motion occurs at each iteration. Thus, it is easier for the walker to move outside an acceptable range from the reference energy, resulting in its deletion from the array.

This allowance of extra motion is exactly what we see with a larger time step in that enough walkers always move out of an acceptable range at each iteration resulting in a walker population that does not converge to the initial population. With regard to the standard deviation, we saw that with a larger time step, the range over which walkers are likely to move is increased, so a larger portion of the walkers moved outside an acceptable range at each step. When the walkers are deleted, they no longer contribute to the reference energy, resulting in a lower standard deviation in that the walkers remaining are more identical in their distribution. In contrast, with a smaller time step less motion occurs at each iteration. As a result, the walkers are able to move to the edge of an acceptable value without being deleted from the walker array given that they can move in small increments towards a non-valid atom configuration. As such, a smaller time step actually creates more variance in the system, which is what we would expect as we get closer to simulating continuous time within a system.

As mentioned in the modelling section, our implementation of the DMC algorithm heavily utilized the NumPy library. NumPy's vectorization techniques with matrices allowed the computations to be split up and thus done efficiently among the cores of the computer. Different hardware architectures allow for more compatibility with vectorization techniques, but even on an older laptop, we still saw a noticeable decrease in run time on the order of 40 times faster than a more traditional DMC implementation. At the core of NumPy,

however, the implementation still has to loop over every value within the 4D array. This explains why the implementation is a constant time improvement, and not linear time or better. Ultimately, we did not do anything meaningfully different in the computation in the simulation loop, but we did the same process much faster. Even so, an improvement of 40 times should not be underestimated as it is the difference between hours and days when running these lengthy trials, particularly when more walkers or production phases are added.

Limitations

Today, some DMC implementations incorporate importance sampling which is a strategy used to estimate the properties of a distribution while only having a sample at hand. As previously stated, the goal of the DMC algorithm is to approximate the wave function of the Schrodinger equation. With only samples at hand, those samples being the number of walkers, importance sampling allows for the wave function to be approximated more accurately given that the computational complexity of the algorithm limits the number of walkers on which we can realistically run trials. Future work may include adding importance-sampling techniques into our implementation.

Time is always a limiting factor in computational sciences. Given that it is necessary to standardize the number of iterations in a simulation to the time step, running trials on smaller time steps takes drastically more time. Time permitting, we would have been interested in quantifying the behaviour of the ZPEG with time steps of .05 and .01 as well. Additionally, it would have been interesting to see the decrease in standard deviation continued if 30 or 50 production phases were used. Finally, in the future, we would like to explore the trend in the ZPEG to determine if it remained unchanged after 10000 walkers. Running trials with 15000 and 20000 walkers should have been sufficient to confirm these conjectures.

Future Work

Moving forward, we hope to continue our investigations of the water trimer molecular system. We have strong data to support the necessary conditions for convergence of the ZPEG calculated based on the time step and walker populations. That said, we have seen that the density histogram of the oxygen angles does not always converge to an expected normal distribution. Other convergence conditions also need to be explored which involve the positions of the Hydrogen atoms in one water molecule with respect to the Oxygen atoms in the other two water molecules. If we can confirm that conditions are met, then, along with the ZPEG calculated, we will be able to determine with confidence that we have determined the ZPEG of the water trimer.

Following the water trimer validation, we then hope to investigate clathrate hydrates, also known as water cages. Clathrate hydrates present the unique challenge in that they are a system of heterogeneous molecules, unlike the systems that have been studied to date. The large advantage of NumPy is that it is very efficient for rectangular matrices. This advantage of rectangular matrices would be lost if we tried to simulate clathrate hydrates in the same way as the water trimer system given that some entries of the 4D array would be empty because each molecule in the system does not have the same number of atoms. It is possible that we could avoid this problem by using NumPy's masking feature or creating our own data type, but this remains to be investigated.

Furthermore, the potential energy function for a clathrate hydrate molecule is likely much more complicated than a water trimer system. As such, we would expect the computational complexity of the system to be much greater. As a final obstacle, equilibration of the system becomes extraordinarily difficult. We have already seen that it is hard to equilibrate a water trimer from random values, given the magnitude of the randomness introduced into the system. Thus, by the same logic, it follows that a clathrate hydrate system would be equally challenging to obtain an equilibrium configuration on which to run simulations.

Conclusion

The DMC algorithm remains at the forefront of computational chemistry research given its ability to provide an approximation of a solution to the Shrodinger equation. Our work furthers the knowledge of the DMC algorithm by giving a unique and more efficient implementation leveraging NumPy. With this implementation, we investigated the water molecule and water trimer to show the influence of different simulation constants on the ZPE for a single water molecule. Our findings point to the necessary criteria needed to obtain accurate data from the simulation.