Optimization of Paths of Least Action in Classical Mechanics

Scott Hanna – shanna7@jh.edu

Final Paper – 625.743 Stochastic Optimization and Control – Johns Hopkins University

# Abstract

The principle of least action is a powerful method for solving the classical equations of motion and states that a system of particles will follow a path over space and time that minimizes the total action (the difference between kinetic and potential energies) summed over the path. Thus, the method is sometimes called the path integral approach. A useful feature of this method is the ease with which we can discretize the path over time and therefore compute path integrals that may be too complex using analytical methods. This paper analyses efficient methods for implementing stochastic search optimization methods to the path integral approach of solving classical least action problems. Three algorithms: localized random search (RAND), simulated annealing with metropolis Monte Carlo sampling (SAN), and simultaneous perturbation stochastic optimization (SPSA) are applied and compared in two classical, single particle problems in two spatial dimensions. The SPSA algorithm achieves superior normalized loss and error values for the case with a unique local minimum.

# Introduction

In classical dynamics, physicists wish to make predictions about the future state of a system given the current state of the system and given constraints. In such classical systems the state of a single particle at a given time is defined entirely by its position and velocity. In this paper, our objective is to analyze the performance of optimization algorithms with the goal of finding the most physically accurate paths for single particle dynamics by applying the principle of least action.

Many classical mechanics problems involving a single particle, interacting with its environment, have been solved exactly through deterministic means, for example by classical differential equations such as Newton’s laws, the classical wave equation, or Poisson’s equation for electrostatics (Sears, 2004). However, for systems with complicated potentials, direct analytical solutions are not always feasible. Therefore, numerical techniques are required to accurately make predictions. Another consideration is the presence of measurement error in any real system. This error could include imprecise measurements in the state of the particles or in unknown external environmental factors.

The principle of least action of a physical system is a method which can derive the predictive equation of motion for that system. The principle is powerful in its universality and applicability to both classical and quantum systems (Dirac, 1933). One of the reasons for my interest in understanding this technique is that while working at the NIST center for neutron research (NCNR) we compared our empirical findings with path integral simulations of the sample medium (Prisk, 2020). These simulations used Monte-Carlo path sampling to determine the dynamics of quantum systems. The last action principle and the techniques used this paper therefore have far ranging applications outside of the classical framework.

One reason that efficient optimization methods are desired for path integral techniques is that, in quantum systems, the number of possible state configurations grows exponentially with the number of particles (Fernando, 2019). If we can develop a path space that is less dependent on the number of total state configurations, we could speed up the modeling of quantum systems with many entangled particles. Current techniques can only model in the 10’s to 100’s of particles at a time due to this issue (Fernando, 2019). Although this paper focuses on classical systems, it is possible that the path integral methods can be extended to quantum systems as well as in (Dornheim, 2019).

The tool used to solve classical least action problems is Lagrangian mechanics, where the action is defined as the sum over the difference of the kinetic energy and potential energy of a system over its path. This summation represents the path integral. The system will evolve in a way that minimizes this action over the entirety of the path. One important characteristic of the Lagrangian formalism of the principle of least action is the ease with which we can discretize the path in the time domain. This allows for the utilization of numerical methods to approximate the action of a system for problems that are too complicated to solve analytically.

In prior work, optimization techniques applied to the problem of minimizing a path integral have focused on Monte-Carlo path sampling techniques (González, 2020). In this paper, we use the work in (González, 2020) as an initial guide for further study. (González, 2020) utilized simulated annealing (SAN) with Monte-Carlo sampling under the metropolis condition to analyze the motion of a single particle in one spatial dimension. We extend this by comparing the SAN method to localized random search (RAND) and simultaneous perturbation stochastic approximation (SPSA) while also extending the original paper to 2 (or more) spatial dimensions. Two problems are analyzed: A potential field for which the action has a single known local minimum, and a more complicated field with multiple local minima representing a global optimization problem.

# Path Integral Formalism

(González, 2020) provides much of the setup for a one-dimensional version of the least action path integral technique applied to solving a system of one particle. To extend the path to a 2 (or more) dimensional space, we define the path of a particle by parameterizing the position of the particle over time: where representing the *d*-dimensional position of the particle at a particular time, . For the entirety of this paper, we constrain the initial and final positions of the particle: where are the initial time and position and are the final time and position of the particle.

The action of the path of the particle is defined as: which is a functional taking the path as the input and returning a real value representing the total action for that path. In classical mechanics the action is defined as the sum of the Lagrangian over the path where the Lagrangian is the difference in the kinetic energy and potential energy at every point on the path. Therefore:

Where is the mass of the particle and represents the instantaneous velocity of the particle at time, *t*. The potential energy is dependent on the problem and acts as our input to define the environment through which the particle travels. For a classical particle we can find the stationary points of the action by taking the first derivative of the action over the path and setting it equal to zero: . The solution to this results in the Euler-Lagrange equation which, for classical particles, reduces to Newton’s second law of dynamics (Forsyth, 1960). However, in this paper we focus on solving the minimum of the action by discretizing and algorithmically searching for the minimum value. We can discretize the action by splitting time into *N* discrete chunks and finding the position for each discrete time as shown.

We split the path into equally sized time spans, therefore is equal to for all . Our discrete path space is then defined by the *N* dimensional position vector for . We could let our optimization parameters be (we do not include the constrained endpoints). However, by defining the parameters over the positions of the path in an independent way, we have over-sampled the problem. Path positions have influence over other nearby positions due to the term from the kinetic energy. This implies that changing the distance between two positions will, in turn, affect the distances (and therefore energies) between adjacent positions. Without taking this into account path positions can vary wildly from iteration to iteration during optimization. Note that in this paper the indexing for elements of a vector will be labeled by subscript *i*, while the iterations will be labeled by subscript *k.*

We can capture the interdependence of path positions and reduce the parameter dimensionality by reparametrizing the path space over an orthonormal basis:

where the become the parameters and the orthnormal bases can be determined in a problem specific way. As described in (González, 2020), and used in this paper, a basis set determined by Bezier curves was chosen. Bezier curves are a rational choice in that they smooth out difference between path positions. In classical physics, small perturbations in the path of an object (due to thermal or quantum effects for example) are averaged out over time and, for many problems, the objects tend to move in smooth curves. An additional benefit is that Bezier curves allow us to easily set control points at the beginning and end of our path. The Bezier curve bases coefficients are computed as follows:

where *n* is the number of chosen control points, . Also, and are our two constrained control points. We can therefore we let (again we do not include the fixed end points). The number of free parameters will therefore be and, generally, we want n < N. Note that the indexing of must start at 0 in order that the Bezier computation gives correct endpoints.

Directly converting the *n* control point parameters into the *N* path positions and then finding the action in equation (1) at every iteration is costly for large *n* or *N*. However, we can transform the action into a function of the control points by recognizing that the orthonormal basis is only dependent on the time discretization and number of control points, *n.* We can therefore precompute as an matrix and it will remain static for every iteration of the algorithm. This allows us to directly convert into at every th iteration by the linear transformation .

Furthermore, we can split the action into the difference of two terms: the sum over the kinetic energy and the sum over the potential . Then, we can redefine the sum over the KE as follows:

where and *D* is an matrix with down the main diagonal and down the upper diagonal and represents a difference-between-components transformation. The matrix can be precomputed at the start of the problem. We can also define a sum over the potential, , that depends on but that will be problem specific. Therefore, the action at some iteration, *k*, becomes a function of the control parameters:

Minimizing this action along the control parameters will allow us to construct a terminal path for the particle. In the next section we discuss how the three optimization methods can be applied to this task.

# Optimization Methods

Because we are minimizing the action, we define the loss function as where with the initial and final control points fixed in the problem as described in the previous section. We also consider noisy measurements of the action therefore we define where will be our pre-defined measurement error for the problem analysis.

The minimization parameters are the free control points whose elements are *d*-dimensional: corresponding to the number of spatial dimensions, *d*, that define the positions on the path. Fortunately, the three algorithms utilized in this paper can be slightly modified such that the perturbation vector components can be defined as *d*-dimensional vectors. It was therefore unnecessary to form a set of real values formed from unpacking the components of each . The implementation for each algorithm is described below.

The optimization technique utilized in (González, 2020) is simulated annealing (SAN) with Monte Carlo sampling using the Metropolis criterion. We compare this technique to localized random search (RAND B - random search algorithm B in (Spall (2003))) as a baseline, and to simultaneous perturbation stochastic approximation (SPSA).

The implementation of SAN in (González, 2020) corresponds closely to the method outlined in (Spall, 2003). Like in RAND B, at every iteration a randomized jump is performed. For both SAN and RAND B, is chosen such that every element **.** The action of the new path is then compared to the old path. If the new action is smaller, then the new jump location is selected. In SAN, if the new action is greater than the old action, then the new path may be selected only if it meets the Metropolis criterion:

The coefficient is defined as the inverse of the temperature which determines the rate of convergence. As and it becomes less and less likely that SAN will jump away from local minimums.

A disadvantage of SAN is that, as in random search algorithms, the perturbation vector is unweighted by local gradient information. The question arises: can we get increased efficiency and improved normalized error from an algorithm that does consider gradient information in the path space? For some potentials it is possible that such a gradient over the action can be found analytically. However, we are going to assume such an analysis is not possible either because of the complexity of the potential or due to random error in the measurement of the action. We therefore focus on gradient-free stochastic approximation (SA) methods.

Finite Difference SA (FDSA) would be inefficient for path integral evaluations as there would need to be *n* function evaluations at each iteration where *n* is the number of control points chosen for the problem. There would therefore need to be *n\*N* evaluations of where N is the number of path space positions. SPSA is an SA method that requires only two function evaluations regardless of the number of control points chosen. Therefore, only 2N potential evaluations are required for each iteration. SPSA is defined using the standard SA gradient descent here the gradient approximation is defined as:

where is a chosen perturbation distribution. Note that this is a modification to SPSA in (Spall (2003)) where the gradient difference is multiplied by the element-wise inverse of . We apply this modification because by multiplying by we avoid defining (and computing) the inverse of each vector element . For the implementation of SPSA in this paper perturbations are chosen such that . For this Bernoulli random perturbation multiplying by is equivalent to multiplying by its element-wise inverse (Chin, 1997).

The gain coefficients and are defined as , where parameters are free parameters which affect convergence rate and gradient weightings. Due to the numerous inputs into the action evaluation (initial path, initial and final positions and times, mass of the particle, number of path locations N, and the form of the potential) manually turning the gain coefficients was difficult. However, because we have restricted the initial and final states of the path, we can get an initial guess for the gain coefficient semi-automatically by taking into consideration the initial path length. First, as outlined in (Spall, 2003), a good initial value for c is equal to the standard deviation of the measurement error. In our case then we therefore set . Also, a good initial guess for A is 10% of the number of iterations for our SA algorithm. We also start with the slowest convergence exponents: .

These restrictions then let us focus on finding the initial guess for . We want to choose a value such that is not too far from the desired change in the elements. Since can vary due to the perturbation and measurement error at every iteration, I take the average over the elements . I then take the maximum of the magnitude over several iterations: . We do not necessarily know what the desired change to the elements should be; however, we can take an initial guess that it should not be more than some fraction of the total path length of the path formed by which we will denote as . I therefore divide the path length by the number of elements, *p*, in , which corresponds to the number of free control points:

For some potentials, there may be more than one local minima. Therefore, global optimization techniques are required to increase the likelihood of jumping out of local minima to find global minima. SAN accomplishes this with the Metropolis criterion which gives a chance for it to possibly jump to locations of higher loss value. Localized random search is less robust than SAN for global optimization as it can only jump to locations with smaller loss values. This does not prevent it from finding new local minima, however, if it can search over hills into new regions of lower loss.

SPSA as a global optimizer is studied in (Maryak, 2001) both with and without injected noise. Even without injected noise, the algorithm is randomly sampling directions such that it can jump to values of higher loss. In this paper I implement SPSA without injected noise, but a study of SPSA with injected noise to path integral minimization problems is an avenue of further research.

# Analysis

**Problem 1: Parabolic Path due to Gravity**

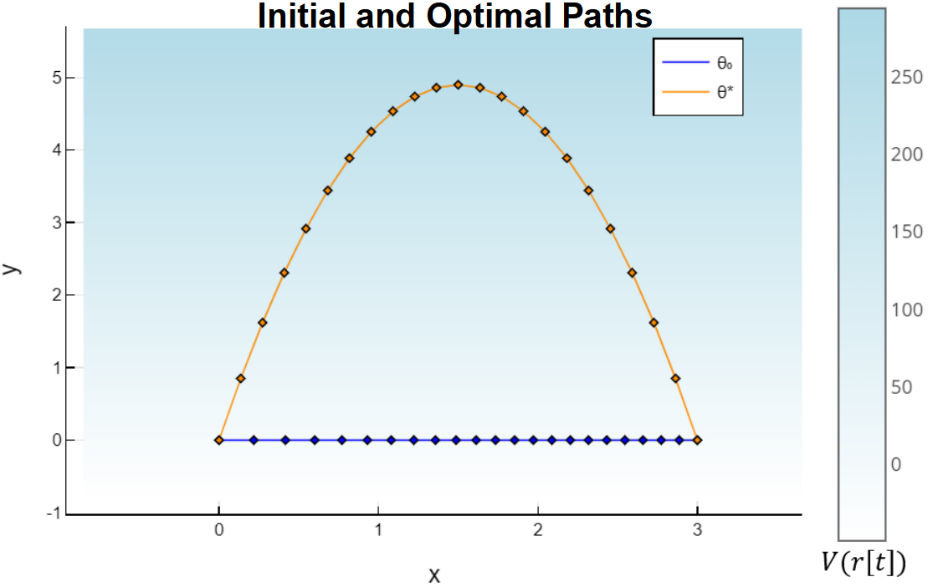
We start with a simple problem, with a unique global minimum where and are known and there are no other local minimums. For this problem, we define the potential of the particle at a certain time as . Where is the chosen mass of the particle and is the magnitude of the gravitational acceleration near the surface of Earth. The value is the second component of the position, at time . This is the potential for a particle undergoing a downward gravitational force close to the surface of Earth. Initial and final positions (units in meters) are chosen to be and . The initial control points, , were chosen such that it makes a straight line from to . The initial and optimal loss are . The optimal path can be found using Newton’s second law and gives the optimal path: which results in a minimum loss of . There were path locations sampled and control points. Initial path and optimal path are shown in figure 1.

Figure : Initial (Blue) and Optimal (Orange) paths for problem 1. The optimal path is found using Newton’s Laws.

The three algorithms were run for 500 loss (action) measurements. Thus, the random algorithm B and SAN both ran for 500 iterations and SPSA, which requires 2 loss measurements per iteration, was run for 250 iterations. The tuning parameters for both the RAND B and SAN were both optimized first. They both used perturbations for both spatial dimensions for each . SAN was run with an initial temperature of , with temperature drops every 20 iterations, dropping by with .

|  |  |  |
| --- | --- | --- |
|  | Terminal Normalized Loss  [95% CI] | Terminal Normalized Error  [95% CI] |
| **RAND B** | 0.0187  [0.0146,0.0229] | 0.081  [0.068,0.094] |
| **SAN** | 0.0155  [0.0122,0.0189] | 0.076  [0.063,0.088] |
| **SPSA** | 0.0016  [0.0010,0.0023] | 0.012  [0.010,0.014] |

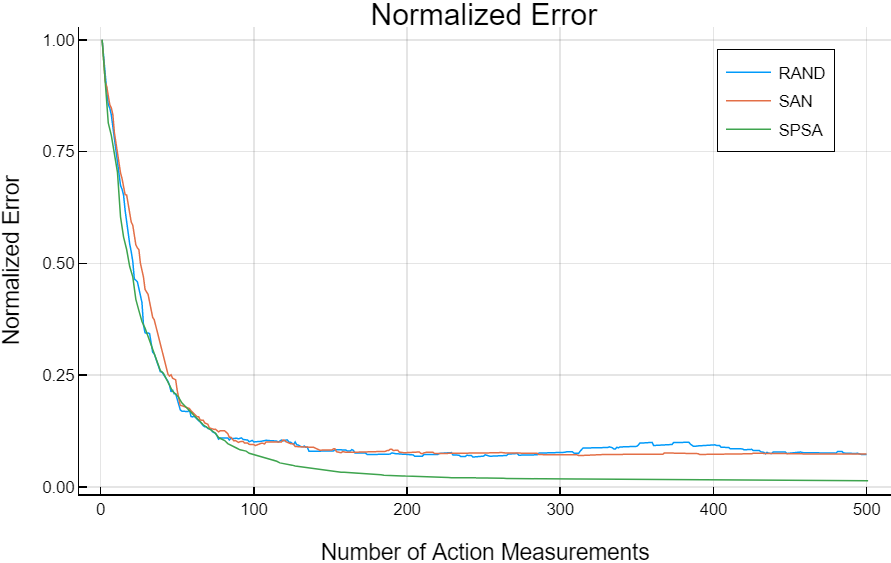
SPSA settings were chosen as described in the previous section. For this problem that resulted in the following values: . Using the equation (4) but I increased to after manual turning.

Figure : Normalized Error as a function of function measurements. SPSA (in green) converges quickly with the other algorithms but has better asymptotic accuracy.

Table 1: Normalized Loss and Normalized Error results for the three algorithms. SPSA shows better performance on both measures.

Mean results over 20 replications for the terminal normalized loss: (and normalized error in values are shown in table 1 with 95% confidence intervals. A plot of the normalized error over the function measurements is provided in figure 2.

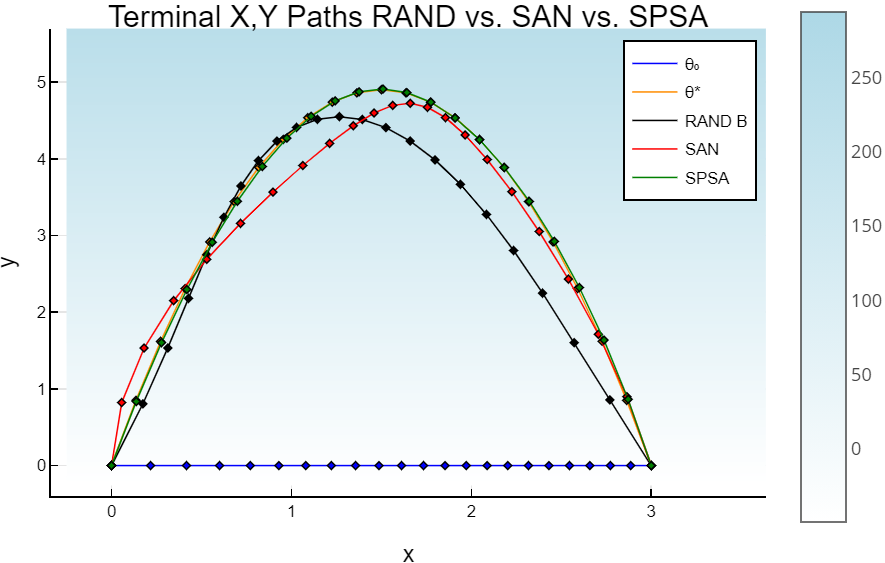
Table 1 provides evidence that supports the superior performance of SPSA for this problem. The best visual demonstration of the strength of the SPSA algorithm for this problem is the plot of the terminal paths for each algorithm, shown to the right. SPSA can get closer to the physically accurate path than either of the other two algorithms after 500 loss measurements.

Figure : Terminal Paths for the three algorithms. SPSA (in green) is nearly identical to the optimal solution.

**Problem 2: Path of Least Mud**

The second problem is premised on a scenario in which someone wishes to cross a muddy field with a minimum of mud on his or her shoes. This problem has a more complicated potential function than the previous problem in which there are many local minima and unknown global minimum. The potential for an object at time is defined as follows:

Where are uniformly locations chosen over a pre-defined rectangle over which the path will traverse. Each represents a location where a patch of mud is centered. Also, if then it is set to -1000. This formed identifiable plateaus of low potential that the algorithms would try to avoid. The value M adjusts the strength of the “muddiness” and was set to . I chose S = 60 to get a good spread of mud around a rectangle about 4 meters wide and 6 meters high. The path was chosen to start at the left side of the muddy area and end on the right side in a time of and with a particle mass of 50 kg. There were sampled path locations and control points. The initial was chosen randomly and the elements were made to be an order of magnitude greater than the smallest distance between the fixed endpoints, resulting in a relatively high . Gain values and initial values were the same as in problem 1 for all three algorithms except for SPSA where (which was re-evaluated semi-automatically based on the new energy of the particle). Table 2 displays the results of for the three algorithms after 500 loss measurements and 20 replications.

|  |  |  |  |
| --- | --- | --- | --- |
|  | **RAND – Algorithm B** | **SAN** | **SPSA** |
| 95% CI | 0.210  [0.207,0.214] | 0.209  [0.205,0.213] | 0.207  [0.203,0.211] |

Table 2: values for the three algorithms with 95% confidence intervals. The three algorithms matched performance for this problem.

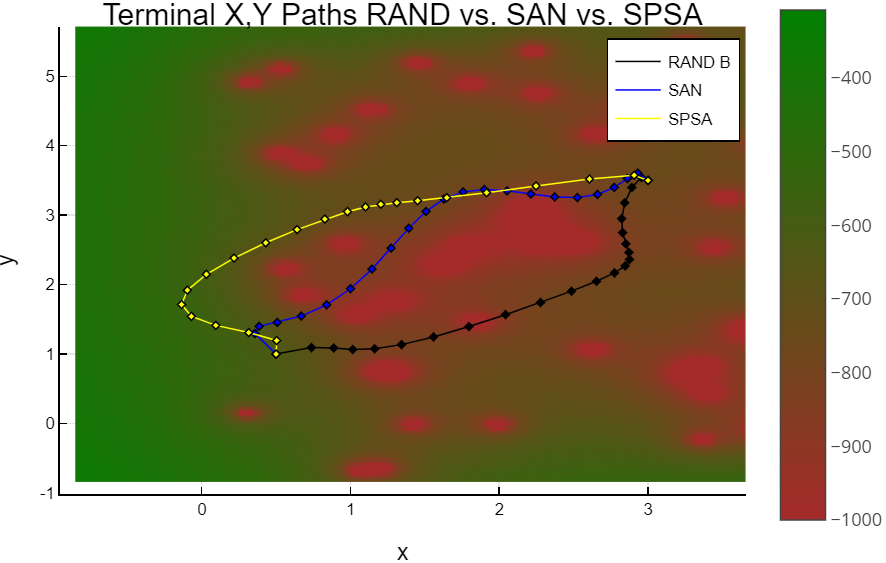
 The performance of SPSA with respect to the normalized loss was on par with the random algorithm and SAN for this problem. These small differences in loss values result in very different paths as can be seen in the terminal paths in figure 4.

Figure : Terminal paths for the three algorithms. Each algorithm took a unique path to the end point.

There are a few considerations to note in this problem. First, the amount of mud traversed is not the only constraint. There is an end time set to 1.5s over this fixed distance which constrains the KE of the path. If there were no such constraint (or time is set to a high value) then the particle will simply find a way around the mud without going through the muddy area at all.

Another consideration is the discrete sampling of the path. You can observe places where the path cross over a muddy patch. The algorithm effectively has no information about the potential between sampled points. If this were a continuous problem this could be a problem which might require more path samples. However, the mud problem is modelling a person walking or running from start to finish. We can consider each sampled point to be a discrete step the person is taking. Where points are farther spread apart the person is moving faster, or even jumping. This is one reason why all the algorithms increase the speed of the person through the muddy area but slow them down outside of it.

# Discussion

**Results**

This paper demonstrated the implementation techniques and validity of the SPSA algorithm for use in path integral minimization. SPSA outperformed both SAN and localized random search on a problem with a unique global minimum and no other local minimums and with known solutions. It achieved more realistic physical paths and lower normalized error over 500 function evaluations. SPSA also performed as well as the other algorithms on a problem with multiple local minima. This suggests further study could be made into improving the efficiency of SPSA for global path problems such as new perturbation distributions or the addition of injected noise.

I was impressed by the robustness of both SAN and random search. Unlike SPSA, they required very few parameter modifications to work for both problems and under different initial conditions. For the first problem they both converged rapidly even though they did not converge to a physically realistic path after 500 iterations. (González, 2020) ran SAN over 10000 iterations to get accurate results for the 1-dimensional problems analyzed in that paper.

**Further Research**

The transformation under Bezier curve bases worked well for these simple classical problems, but it would be interesting to know what other bases might work. For example, for confined quantum systems, stationary energy eigenstates are usually chosen:

This would require the complex valued eigenvalues to be minimized. Some work has been done to apply SPSA to the problem of minimization over a complex space (Utreras-Alarcón (2019)).

Additional techniques could be used to reduce the number of potential function evaluations per iteration. Reducing the number of evaluations is important in quantum mechanical problems where the number of states increases greatly with each entangled particle. Despite the transformation into the basis vectors at each iteration still required the multiplication of matrices on the order of with the state vector. One possible avenue to reduce this is to reduce the number of path locations sampled per iteration, but randomly distribute them in the time domain at each iteration. Therefore, much like the way SPSA works, we are doing time averaging of the path samples to determine the total potential along the path.

Another avenue of further research is in the optimization of global optimization problems. SPSA with Bernoulli (+1, -1) perturbations without injected noise was used in this paper, but a comparison with other distributions with and without injected noise is needed. Also, multiplying the perturbation vector in instead of its element-wise inverse places new regularity conditions on the perturbations and has been explored in other contexts (Chin, 1997) but not for global optimization or for path integral problems. It would also be interesting to compare results with an implementation of single-measurement SPSA since that would align the number of measurement evaluations directly with the SAN and random search algorithms. Finally, other global optimization techniques, such as genetic algorithms could be compared as well.

# References

Chin, D.C. (1997),"Comparative Study of Stochastic Algorithms for System Optimization Based on Gradient Approximations," IEEE Transactions on Systems, Man, and Cybernetics — B, vol. 27, pp. 244-249 (theoretical and numerical efficiency analysis).

Dirac, Paul A. M. (1933), “The Lagrangian in Quantum Mechanics,” Physikalische Zeitschrift der Sowjetunion. 3 (1): 64–72.

Dornheim, T., Groth, S., Filinov, V., and Bonitz, M. (2019), “Path integral Monte Carlo simulation of degenerate electrons: Permutation-cycle properties,” The Journal of Chemical Physics 151, 014108.

Fernando G. S. L., Chemissany, W., Hunter-Jones N., Kueng, R., and Preskill, J. (2019), “Models of Quantum Complexity Growth,” Models of quantum complexity growth, arXiv:1912.04297 [hep-th].

Forsyth, A. R. (1960), “Calculus of Variations”. New York: Dover, pp. 17-20 and 29.

González Diaz, D., Davis, S., & Curilef, S. (2020), “Solving Equations of Motion by Using Monte Carlo Metropolis: Novel Method Via Random Paths Sampling and the Maximum Caliber Principle,” Entropy (Basel, Switzerland), 22(9), 916.

Marx, D., and Parrinello, M. (1996), “Ab initio path integral molecular dynamics: Basic ideas,” The Journal of Chemical Physics 104, 4077-4082.

Maryak, J.L., and Chin, D.C. (2001), “Global Random Optimization by Simultaneous Perturbation Stochastic Approximation," Proceedings of the American Control Conference, 25-27 June 2001, Arlington, VA, pp. 756-762.

Prisk, T.R., Hanna, S. & Azuah, R.T. (2020), “Self-diffusion of Liquid Hydrogen: A Quasi-elastic Neutron Scattering Study,” J Low Temp Phys 201, 451–462.

Sears (2004). “Sears and Zemansky's University Physics: With Modern Physics,” San Francisco: Pearson Addison Wesley, Print.

Spall, J. C. (2003), “Introduction to Stochastic Search and Optimization,” Hoboken, NJ, USA: Wiley-Interscience.

Utreras-Alarcón, A., Rivera-Tapia, M., Niklitschek, S. et al. (2019), “Stochastic optimization on complex variables and pure-state quantum tomography,” Sci Rep 9, 16143.

Code for project available at: <https://github.com/sphanna/PathIntegralOptimization>