Numerical integration of stochastic differential equations

Gerasimos Angelatos, Zachary Hervieux-Moore

CONTENTS

I. INTRODUCTION AND BACKGROUND

A. Motivation

In this course, we studied algorithms such as Runge-Kutta and Adams-Bashforth for solving systems of ordinary differential equations (ODEs). However, in our research we occasionally encounter stochastic differential equations (SDEs): these models describe a wide variety of phenomena, from financial systems to quantum measurement. Generally, any system whose evolution is continuous and has a non-deterministic evolution can be described by a set of SDEs. This is equivalent to saying the probability density function of such a system obeys a Fokker-Plank equation (FPE), and indeed, the ability to map a FPE to an equivalent set of SDEs is a powerful solution method, regularly exploited in quantum optics [1], to treat FPEs which are otherwise difficult to solve.

However, the solution of SDEs, both numerically and even analytically, is considerably different than their deterministic counterparts. All but the most simple algorithms for ODEs cannot be directly adapted to their stochastic counterparts because of the non-differentiability of the noise portion of their evolution. Furthermore, these algorithms when adapted to SDEs result in a lower order of convergence. There are only a few straightforward and broadly applicable algorithms for the solution of general systems of SDEs, which we present in this report. In addition, there are no widely-used python packages or libraries of SDE algorithms, and a naive attempt at solving a set of SDEs, for the reasons presented above, often ends in failure. In this report, we consider the convergence and stability of three popular algorithms for solving SDEs: Explicit Euler, Milstein, and Semi-Implicit Euler. After presenting a background on SDEs and describing each algorithm, we discuss, evaluate, and compare their convergence and stability properties. We also wrote our own python SDE integrator class as a part of this project, the documentation for which is found in the appendix.

B. The Weiner Process

The most intuitive introduction of an SDE is through a Langavin equation, which describes the evolution of some variable subject to a fluctuating force, which in 1-dimension can be written:

$$\frac{dx}{dt} = A(x,t) + B(x,t)\zeta(t) \tag{1}$$

where ζ is a rapidly fluctuating random term, such that $\zeta(t)$ and $\zeta(t')$ are statistically independent [2]. $\zeta(t)$ is white noise, meaning it obeys a Gaussian probability distribution with zero mean $\langle\zeta(t)\rangle = 0$ and correlation $\langle\zeta(t)\zeta(t')\rangle = \delta(t-t')$, meaning the force at one time is completely uncorrelated with any other time. To quantify this white noise, we integrate Eq. (1), and since we expect it to be integrable

$$W(t) = \int_0^t ds\zeta(s) \tag{2}$$

exists and is a continuous quantity. Furthermore, by splitting the above integral at $t' < t$ we see that $\zeta(s)$ in each integral are statistically independent: $W(t')$ and $W(t) - W(t')$ are independent as well then and is a Markov process. Indeed by calculating moments it can readily be seen that $W(t)$ is the Weiner process (often referred to as Brownian motion), with the steady state distribution (for initial condition $p(W, t_0|0, t_0) = \delta(W)$):

$$p(W, t|0, t_0) = [(2\pi)(t-t_0)]^{-\frac{1}{2}} e^{\frac{-W^2}{2(t-t_0)}} \tag{3}$$

with zero mean $\langle W(t)\rangle = 0$ and linearly growing variance [2] $\langle W(t)^2\rangle = t - t_0$. The Weiner process can be thought of as the limit of a continuous-time random walk.

If one breaks the Weiner process into increments (which we will do shortly when solving SDEs) $\Delta W_i = W(t_i) - W(t_{i-1})$, from Eq. (3) we have the joint probability density [2]:

$$p(\Delta W_n; \Delta W_{n-1}; \ldots \Delta W_0) = \prod_{i=1}^{n} (2\pi\Delta t_i)^{-\frac{1}{2}} e^{\frac{-\Delta W_i^2}{2\Delta t_i}} p(W_0, t_0). \tag{4}$$

Since the joint probability can been factorized into a product of individual probabilities for each Wiener increment ΔW_i, the $\{\Delta W_i\}$ are statistically independent from each other. Using this independence of increments, it can be seen that

$$\langle W(t)W(s)\rangle = \min(t,s). \tag{5}$$

Lastly, the Wiener process is non-differentiable, since $\lim_{h\to 0}\text{Prob}(|W(t+h)-W(t)|/h > k) = \lim_{h\to 0} 2\int_{kh}^{\infty}(2\pi h)^{-\frac{1}{2}}e^{\frac{-W^2}{2h}}dW = 1$. Thus the derivative is infinite, since $\lim_{h\to 0}|W(t+h)-W(t)|/h$ is larger than any value of k.

Because of this non-differentiablity, we should instead consider the integral form of Eq. (1):

$$x(t) = x(t_0) + \int_{t_0}^{t} A(x(s),s)ds + \int_{t_0}^{t} B(x(s),s)dW(s). \tag{6}$$

where

$$dW(t) = W(t+dt) - W(t) = \zeta(t)dt \tag{7}$$

is the Wiener increment.

C. Ito calculus

Now once the Wiener increment has been defined, the question immediately arises of how to evaluate a stochastic integral, such as the final term of Eq. (6). Defining the stochastic integral $S = \int f(t')dW(t')$ as the $n\to\infty$ limit of partial sums:

$$S_n = \sum_{i}^{n} f(\tau_i)\left(W(t_i) - W(t_{i-1})\right), \tag{8}$$

it is apparent that the value of this quantity depends on where τ_i falls in the interval $t_{i-1}\to t_i$. The Ito calculus is defined by the choice $\tau_i = t_{i-1}$ so that the Ito stochastic integral is

$$\int_{t_0}^{t} f(t')dW(t') = \lim_{n\to\infty}\sum_{i}^{n} f(t_{i-1})\left(W(t_i) - W(t_{i-1})\right) \tag{9}$$

where the limit above is technically the mean square limit, indicating that the mean of the sequence converges to this value. For a non-anticipating (ie.e. physical) function, $f(t)$ in the Ito integral will be statistically independent of the Weiner increment and since $\langle dW(t)\rangle = 0$

$$\langle\int_{t_0}^{t} f(t')dW(t')\rangle = 0 \tag{10}$$

Furthermore, a straightforward application of Eq. (5) to Eq. (7) gives $\langle dW(t)^2\rangle = dt$. It is also straightforward to derive $\int f(t')dW(t')^2 = \lim_{n\to\infty}\sum_i^n f(t_{i-1})\left(W(t_i)-W(t_{i-1})\right)^2 = \lim_{n\to\infty}\sum_i^n f(t_{i-1})\Delta t_i = \int f(t')dt'$. Since $dW(t')$ only appears under an integral, we can therefore make the identification for Ito calculus that $dW(t)^2 = dt$. This can be quickly extended to multi-dimensional systems $d\mathbf{W}$ and using the statistical independence of components $dW_i(t)$ and $dW_j(t)$ we have

$$dW_i(t)dW_j(t) = \delta_{ij}dt \qquad dW_i(t)^{2+n} = 0 \tag{11}$$

These results can be readily understood by treating $dW(t)$ as an infinitesimal of order $\frac{1}{2}$, thus $dW(t)^2 = dt$ and $dW(t)^{2+n} = 0$ since it is an infinitesimal of order greater than 1. Now that an understanding of the Wiener process has been developed, we can define the Ito SDE for a multidimensional stochastic quantity $\mathbf{x}(t)$

$$d\mathbf{x}(t) = \mathbf{A}(\mathbf{x}, t)dt + \mathbf{B}(\mathbf{x}, t)d\mathbf{W} \tag{12}$$

solved by

$$\mathbf{x}(t) = \mathbf{x}(t_0) + \int_{t_0}^{t} \mathbf{A}(\mathbf{x}, t')dt' + \int_{t_0}^{t} \mathbf{B}(\mathbf{x}, t)d\mathbf{W}(t'). \tag{13}$$

From here we can calculate various moments of $\mathbf{x}(t)$ using Ito calculus or numerically integrate the above to solve for sample trajectories using the algorithms discussed in the next section. Since dW is an infinitesimal of order $\frac{1}{2}$, one cannot blindly apply standard calculus to stochastic quantities $\mathbf{x}(t)$. Consider an arbitrary function $f(\mathbf{x}(t))$:

$$\begin{aligned} df(\mathbf{x}) =& f(\mathbf{x} + d\mathbf{x}) - f(\mathbf{x}) = \partial_{x_i} f(\mathbf{x}) dx_i + \frac{1}{2}\partial_{x_i}\partial_{x_j} f(\mathbf{x}) dx_i dx_j + \ldots \\ =& \partial_{x_i} f(\mathbf{x}) \left(A_i + B_{ij} dW_j\right) + \frac{1}{2}\partial_{x_i}\partial_{x_j} f(\mathbf{x}) \left[\mathbf{B}\mathbf{B}^T\right]_{ij} dt \ldots \end{aligned} \tag{14}$$

where in the both lines we drop all terms of order greater than dt (note that we needed to continue the expansion to second order to produce all terms of order dt, unlike in standard calculus), and used Eq. (11) to yield the second line. Grouping terms we arrive at Ito's formula:

$$df(\mathbf{x}) = \left(A_i(\mathbf{x})\partial_{x_i} f(\mathbf{x}) + \frac{1}{2}\left[\mathbf{B}(\mathbf{x})\mathbf{B}^T(\mathbf{x})\right]_{ij} \partial_{x_i}\partial_{x_j} f(\mathbf{x})\right) dt + B_{ij}(\mathbf{x})\partial_{x_i} f(\mathbf{x}) dW_j \tag{15}$$

which is used to derive SDEs for arbitrary functions of a fundamental stochastic variable $\mathbf{x}(t)$

D. Stratonovich calculus

Equation (9) is not the only choice of τ_i that can be used to evaluate the stochastic integral; the Stratonovich formulation evaluates the integrand at the center of the interval $\tau_i = \frac{t_i + t_{i-1}}{2}$:

$$\int_{S,t_0}^{t} f(x(t'))dW(t') = \lim_{n\to\infty} \sum_{i}^{n} f(x(\frac{t_i + t_{i-1}}{2}))\left(W(t_i) - W(t_{i-1})\right) \tag{16}$$

where the additional subscript S differentiates this from an Ito stochastic integral. Now since $f(x(t'))$ is evaluated in the middle of the interval, it is no longer statistically independent of $dW(t')$ and $\langle \int_{t_0}^{t} f(x(t'))dW(t')\rangle \neq 0$. The connection between Stratanovich and Ito calculus can be seen by application of a discretized Eq. (15) to Eq. (16):

$$\begin{aligned} \lim_{n\to\infty} \sum_{i}^{n} f(x(\frac{t_i + t_{i-1}}{2}))\left(W(t_i) - W(t_{i-1})\right) =& \lim_{n\to\infty} \sum_{i}^{n} f(x(t_{i-1}))\left(W(t_i) - W(t_{i-1})\right) \\ &+ \frac{1}{2}\sum_{i}^{n} B(x(t_{i-1}))\partial_x B(x(t_{i-1}))(t_i - t_{i-1}) \end{aligned}$$

The first term on the right side is the Ito integral, and the second is a standard integral over time and can thus be grouped in with the integration over the drift vector. In general, an Ito SDE $d\mathbf{x} = \mathbf{A}^{\text{Ito}}dt + \mathbf{B}d\mathbf{W}$ is equivalent to the Stratonovich SDE $d\mathbf{x} = \mathbf{A}^{\text{strat}}dt + \mathbf{B}d\mathbf{W}$ with $\mathbf{A}^{\text{strat}} = \mathbf{A}^{\text{Ito}} + \mathbf{A}^{\text{corr}}$, with

$$A_i^{\text{corr}} = -\frac{1}{2}\sum_{jk} B_{kj}\partial_{x_k}B_{ij} \tag{17}$$

The Stratonovich formulation may seem like a lot of unnecessary work, but it has one significant advantage: numerical convergence and stability, as will be noted later. In addition, from Eqs. (17) and (15), one can show that the Stratonovich formulation of stochastic calculus obeys the same rules for a change of variables as ordinary calculus:

$$df(\mathbf{x}) = \left(\mathbf{A}^{\text{strat}}dt + \mathbf{B}d\mathbf{W}\right) f'(\mathbf{x}) \tag{18}$$

II. ALGORITHMS TO INTEGRATE STOCHASTIC DIFFERENTIAL EQUATIONS

We now consider how to numerically integrate SDEs of the form presented in Eq. (12). We will assume we are discussing Ito SDEs in much of what follows, noting that one can convert between formulations via Eq. (17). Furthermore, most of the discussion assumes a 1-D system of SDEs for simplicity, but we also give the n-D forms of the Explicit and Semi-Implicit Euler algorithms, which are implemented in our SDE integrator code.

Consider a 1-D SDE of the form presented in Eq. (6). Integrating Ito's formula Eq. (15) we have:

$$f(x(s)) = f(x(t_0)) + \int_{t_0}^{s}\left(A(x(s'))f'(x(s')) + \frac{1}{2}B^2(x(s'))f''(x(s'))\right)ds' + \int_{t_0}^{s} B(x(s'))f'(x(s'))dW(s') \tag{19}$$

This can then be used to evaluate $A(x(t))$ and $B(x(t))$ in Eq. (6). Substituting this in and dropping explicit time-dependence of these coefficients for simplicity we have: [2]

$$x(t) = x(t_0) + A(x(t_0))\int_{t_0}^{t} ds + B(x(t_0))\int_{t_0}^{t} dW(s) + B(x(t_0))B'(x(t_0))\int_{t_0}^{t}\int_{t_0}^{s} dW(s)dW(s') + \mathcal{R} \tag{20}$$

where $\mathcal{R}$ includes stochastic integrals of order $dW(s)ds'$ and higher in $t - t_0$. In this form, it is clear that the numerical integration of an SDE is arrived at via the mapping $(t_0, t) \rightarrow (t_k, t_{k+1})$ in Eq. (20). Defining the timestep $\Delta t = t_{k+1} - t_k$, we have

$$\int_{t_k}^{t_{k+1}} dW(s) = W(t_{k+1}) - W(t_k) = \Delta W_k = \sqrt{\Delta t}\mathcal{N}(0,1) \tag{21}$$

where we discretized Eq. (9) in the first equality and used the properties of the Weiner process, particularly Eq. (4) to arrive at the second. $\mathcal{N}(0,1)$ is a sampling from the Gaussian distribution with zero mean and unit variance, the $\sqrt{\Delta t}$ factor satisfies $\langle \Delta W_k \Delta W_{k'}\rangle = \Delta t\delta_{k,k'}$ as required from the discussion of the Weiner process.

A. Explicit Euler algorithm

From here, we can immediately arrive at the Explicit Euler algorithm by retaining and discretizing the two single integrals of Eq. (20), resulting in

$$\boxed{\text{Explicit Euler}: \quad x_{k+1} = x_k + A(x_k)\Delta t + B(x_k)\Delta W_k} \tag{22}$$

This is clearly an explicit algorithm, since x_{k+1} depends only on x_k. Since the As we will further ito stochastic integral defines the integrand evaluation at the beginning of each interval, an explicit algorithm is the only self-consistent simulation of an Ito SDE. As explore in the next section, this algorithm is only convergent to order Δt at best due to the discarded terms in $\mathcal{R}$. Actually, the situation is even worse, since the double stochastic integral which was also discarded has zero mean but is of order Δt itself, resulting in sample trajectories only converging to the true solution on order $\sqrt{\Delta t}$. Despite this flaw, the Explicit Euler algorithm is popular due to its speed, which allows for many trajectories to be simulated simultaneously. It can also readily be extended to higher dimensions. For a general, n-D system with m independent Weiner increments and potential explicit-time-dependence, the Explicit Euler algorithm is

$$\boxed{\text{General Explicit Euler}: \quad \mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{A}(\mathbf{x}_k, t_k)\Delta t + \mathbf{B}(\mathbf{x}_k)\Delta \mathbf{W}_k} \tag{23}$$

where each element of the m-dimensional Weiner increment vector is a different random Gaussian sample:

$$(\Delta W_k)_i = \sqrt{\Delta t}(\mathcal{N}(0,1))_i \tag{24}$$

B. Milstein algorithm

The Milstein algorithm is the obvious (at least from the way this discussion was constructed) extension to the Explicit Euler algorithm: we also retain and discretize the third stochastic integral of Eq. (20). In one dimension, this is straightforward to evaluate [2]

$$\int_{t_k}^{t_{k+1}} \int_{t_k}^{s} dW(s)dW(s') = \int_{t_k}^{t_{k+1}} \big(W(s) - W(t_k)\big) dW(s) = \frac{1}{2}\big(\Delta W_k^2 - \Delta t\big) \tag{25}$$

Adding this resultant term to Eq. (22) gives

$$\boxed{\text{Milstein}: \quad x_{k+1} = x_k + A(x_k, t_k)\Delta t + B(x_k, t_k)\Delta W_k + C(x_k, t_k)\big(\Delta W_k^2 - \Delta t\big)} \tag{26}$$

where

$$C(x,t) = \frac{1}{2} B(x,t)\partial_x B(x,t) \tag{27}$$

Can either be a function passed directly to the algorithm or be approximated by numerically differentiating B. By definition $\langle \Delta W_k^2 - \Delta t\rangle = 0$, so this has the same convergence properties in the distribution to the explicit Euler algorithm, however for individual trajectories it is now convergent to order Δt. This is a striking example of the difference between ODEs and SDEs: because dW is of order $\sqrt{\Delta t}$, we need to evaluate stochastic integrals to twice the order of ordinary integrals to achieve the same convergence. Further complicating this is the fact that higher-order simulations require the evaluation of stochastic integrals such as $\int_{t_k}^{t_{k+1}}\int_{t_k}^{s} s dW(s')$ or $\int_{t_k}^{t_{k+1}}\int_{t_k}^{s}\int_{t_k}^{s'} dW(s)dW(s')dW(s'')$, none of which can be expressed directly in terms of ΔW_k and must be computed explicitly [2]. As a result, one is typically forced to accept that a much smaller time-step is needed to achieve the same accuracy.

The extension of the Milstein algorithm to higher-dimensions is not just difficult analytically; it is also so computationally expensive that is not worthwhile [3]. The difficulty arises in evaluating the higher-dimensional form of Eq. (25): $\int_{t_k}^{t_{k+1}}\int_{t_k}^{s} dW_i(s)dW_j(s')$, which cannot be expressed in terms of a Weiner increment unless $i = j$

C. Semi-Implicit Euler

The final algorithm we consider in this work is the Semi-Implicit Euler algorithm. This is best thought of the direct-application of the Euler algorithm to a Stratonovich SDE of the form of Eq. (6). Recalling that we used Ito's formula to derive Eq. (20), in Statonovich calculus the appropriately discretized equation for a scalar Stratonovich SDE is

$$\boxed{\mathrm{Semi-Implicit\,Euler:}\quad x_{k+1} = x_k + A^{\mathrm{Strat}}(\frac{x_{k+1}+x_k}{2}, \frac{t_{k+1}+t_k}{2})\Delta t + B(\frac{x_{k+1}+x_k}{2}, \frac{t_{k+1}+t_k}{2})\Delta W_k} \tag{28}$$

Where as before, $A^{\mathrm{strat}} = A^{\mathrm{Ito}} + A^{\mathrm{corr}}$ via Eq. (17). This algorithm discards the double stochastic integral term, as the Explicit Euler algorithm did, and is only strongly convergent of order $\sqrt{\Delta t}$ as a consequence. However, it is again weakly convergent of order Δt and for many practical problems results are found to converge far quicker than for explicit methods [2, 4]. More importantly, it has far better stability than either explicit method considered, as will be discussed in the next section. Thus, it is often the case that it is worthwhile to transform a set of Ito SDEs to an equivalent set of Stratonovich SDEs and then simulate them via the Semi-Implicit Euler algorithm (Note that A^{corr} can be readily approximated from an Ito SDE via numerical differentiation).

As with the explicit Euler method, the Semi-Implicit Euler algorithm is also easy to extend to an n-dimensional system of SDEs:

$$\boxed{\mathrm{Semi-Implicit\,Euler:}\quad \mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{A}^{\mathrm{Strat}}(\frac{\mathbf{x}_{k+1}+\mathbf{x}_k}{2}, \frac{t_{k+1}+t_k}{2})\Delta t + \mathbf{B}(\frac{\mathbf{x}_{k+1}+\mathbf{x}_k}{2}, \frac{t_{k+1}+t_k}{2})\Delta \mathbf{W}_k} \tag{29}$$

In order to numerically solve Eq. (29), the algorithm must employ some kind of root-finding mechanism. Often simply iterating a few times is sufficient. Below is a code snippet from our Semi-Implicit Euler integrator which implements this:

```
x[:, 0]=x0
Weiners = local_state.normal(size=(m, N-1))
for k in range(N-1):
  xtemp = x[:, k]
  for l in range(Niters):
    xtemp =  x[:, k] + Dt*Af(xtemp, tspan[k]+Dt/2)/2  \
    + np. sqrt(Dt)*Bf(xtemp, tspan[k]+Dt/2) @ Weiners[:, k]/2
  x[:, k+1] = 2*xtemp-x[:, k]
```

III. CONVERGENCE AND STABILITY

A. Stability

In this section, we very briefly discuss the stability of the numerical schemes presented above when applied to a simple linear SDE:

$$dx = \lambda x dt + dW, \tag{30}$$

following Ref. 2. For both the Explicit Euler and Milstein algorithms we have

$$x_{k+1} = (1 + \lambda \Delta t) x_k + \Delta W_k \tag{31}$$

If we consider the evolution of an error which occurred at time k for trajectory x' relative to the "true trajectory" x we have

$$|x'_{k+1} - x_{k+1}| = |1 + \lambda \Delta t| |x'_k - x_k| \tag{32}$$

(since the Weiner increments are the same). After n timesteps this becomes

$$|x'_{k+n} - x_{k+n}| = |1 + \lambda \Delta t|^n |x'_k - x_k|. \tag{33}$$

It is thus apparent that for these explicit schemes, errors will grow exponentially and the algorithm will thus be unstable unless

$$\mathrm{Explicit\,Schemes} : |1 + \lambda \Delta t| < 1 \tag{34}$$

The region of stability is thus a disk of radius $1/\Delta t$ centered on $\lambda = -1/\Delta t$.

For the Semi-Implicit Euler algorithm we instead have

$$x_{k+1} = (1 + \lambda \Delta t) \frac{x_{k+1} + x_k}{2} + \Delta W_k \tag{35}$$

Carrying out the same process as above, it is quickly seen that

$$|x'_{k+n} - x_{k+n}| = \left| \frac{1 + \lambda \Delta t/2}{1 + \lambda \Delta t/2} \right|^n |x'_k - x_k|. \tag{36}$$

and the semi Implicit Euler algorithm is thus stable for $|1 + \lambda \Delta t/2| > |1 + \lambda \Delta t/2|$, equivalently:

$$\mathrm{Semi-Implicit\,Euler} : \mathrm{Re}\{\lambda\} < 0 \tag{37}$$

Indeed, the Semi-Implicit Euler algorithm is A-stable: whenever the exact solution is stable (sample paths not diverging) the algorithm is as well (error does not grow). This is in contrast to what was seen for the explicit algorithms, which have a much smaller region of stability and can be unstable even if the SDE itself is not. This has the effect of significantly restricting the maximum Δt beyond what was needed for convergence.

B. Strong and weak convergence

Just as probability theory has multiple notions of convergence (in distribution, in probability, etc.), there are two notions of convergence in the numerical analysis of stochastic differential equations. The first is strong convergence which is completely analogous to the convergence in L^1 norm of probability. Something is strongly convergent if:

$$\lim_{\delta t \to 0} \mathbb{E}\left[|X_T - X_T^{\delta t}|\right] = 0$$

In words, it is strongly convergent if the expected absolute difference between the actual process, X, and the process generated by a numerical algorithm with time step δt, $X^{\delta t}$, agree with each other at the end time point T. Note that this is a very strong statement about the generation of the path trajectories. It essentially says that the algorithm generates paths identical to the actual process if the time step becomes small enough. Thus, strong convergence is a necessity if you need a numerical scheme that keeps the paths along similar trajectories as the underlying process. To measure the rate of convergence, we say that it is strongly convergent with order γ if

$$\mathbb{E}\left[|X_T - X_T^{\delta t}|\right] \leq O(\delta t)^{\gamma}$$

The second notion of convergence is weak convergence which is similar to the convergence in distribution of probability. Something is weakly convergent if:

$$\lim_{\delta t \to 0} \left|\mathbb{E}\left[g(X_T)\right] - \mathbb{E}\left[g(X_T^{\delta t})\right]\right| = 0 \text{ for all polynomial } g$$

Loosely speaking, it is weakly convergent if the expected absolute difference between the moments of the actual process and the moments of process generated by a numerical algorithm with time step δt, agree with each other at the end time point T. This is much weaker than the previous type of convergence. We allow for the numerical algorithm to deviate from the actual process as long as the mean remains unchanged. Intuitively, if we had a Gaussian process, we might expect an algorithm to introduce more variance. If it did so, it would no longer converge strongly but still converges weakly. To measure the rate of convergence, we say that it is weakly convergent with order γ if

$$\left|\mathbb{E}\left[X_T\right] - \mathbb{E}\left[X_T^{\delta t}\right]\right| \leq O(\delta t)^{\gamma}$$

In practice, requiring strong convergence is too much. Chiefly, it is often the case that the actual distribution of the process is unknown. However, we may know the moments of the process in which case weak convergence is readily available to test against. Also, in applications that care only about the end point such as European options pricing, we do not need strong convergence. This is often the case where we do not care how a particle gets to a position as long as the final position is accurate.

For completeness, the rates of convergences for the algorithms discussed are shown below:

Algorithm	**Strong Rate**	**Weak Rate**
Euler	1/2	1
Semi Implicit	1/2	1
Milstein	1	1

C. Computational resources

A nice feature of solving a set of SDEs is that the calculation of large sets of individual trajectories is parallelizable. Even in this case however, it is worth questioning where one should dedicate computational resources. If they are used to reduce the SDE step-size Δt, the stability and convergence of individual sample paths will increase and they will approach those of the true underlying distribution. Increasing the number of trajectories calculated (and used to evaluate various functions and moments) reduces the sampling error. It is shown in Ref. [3] that the mean squared error can be minimized for

$$N_{\text{steps}} = \frac{T}{\Delta t} \propto C^{1/(2\beta+1)} \quad N_{\text{samps}} \propto C^{2\beta/(2\beta+1)} \tag{38}$$

such that the total computational resources $C = N_{\text{steps}} N_{\text{samps}}$ is fixed.

IV. RESULTS

A. Geometric Brownian Motion

The first case study of the different algorithm we considered was Geometric Brownian Motion (GBM). GBM is the system that satisfies the following SDE:

$$dS_t = \mu S_t dt + \sigma S_t dW_t$$

This process has multiplication noise and so it tends to be unstable. Furthermore, we know the solution of this SDE is

$$S_t = S_0 e^{(\mu - \frac{\sigma^2}{2})t + \sigma W_t}$$

Recall that the noise term is normal and hence S_t is a lognormal distribution with parameters:

$$\mu_{dist} = (\mu - \frac{\sigma^2}{2})t + \log(S_0)$$
$$\sigma_{dist} = \sigma\sqrt{T}$$

Where μ_{dist} and σ_{dist} are the parameters of the lognormal distribution and μ amd σ are parameters of the SDE. Below is a plot of sample paths for $\mu = -5$, $\sigma = 1$, and $S_0 = 10$.

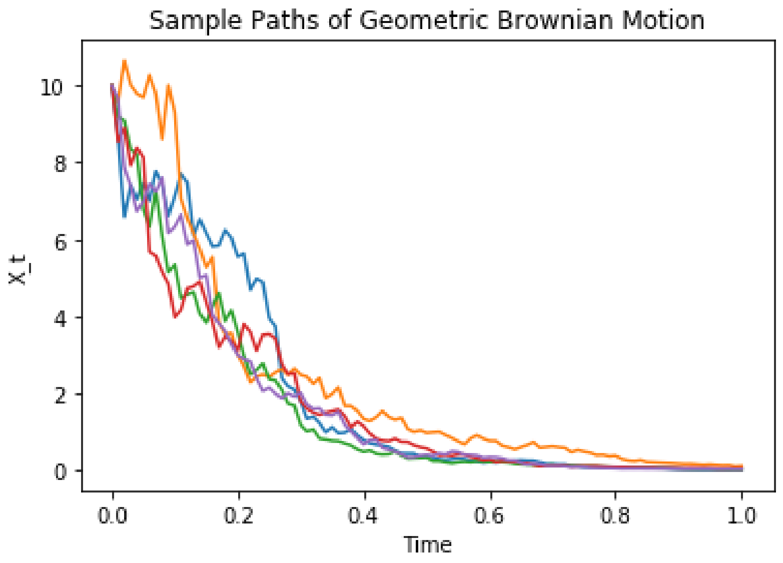

FIG. 1. Five sample paths of GBM with parameters $\mu = -5$, $\sigma = 1$, and $S_0 = 10$

See how this function starts pretty volatile but decreases in variance as it approaches 0 as the multiplicative noise term becomes smaller and smaller. It also tends towards zero because our μ is negative. Otherwise, it would tend to infinity and be unstable. To confirm that our sample paths follow GBM, we generated 1000 paths and compared the empirical distribution to the expected analytical distirubtion. This is show in the figure below.

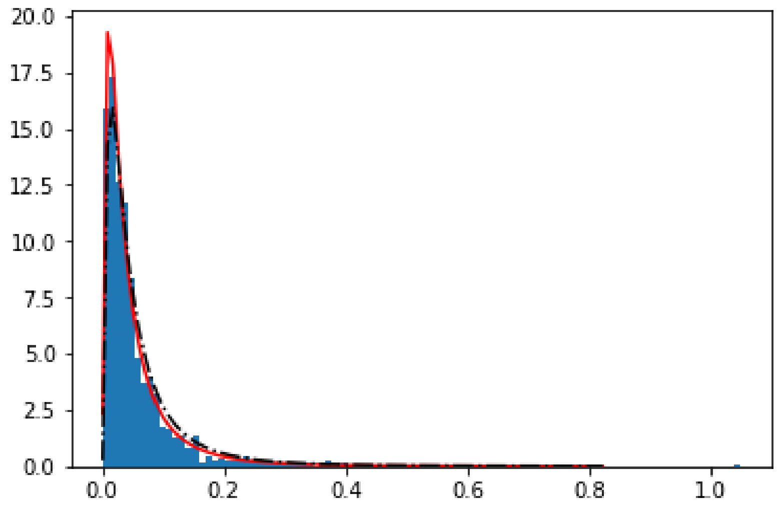

FIG. 2. The histogram of 1000 sample path endpoints. The red line is the estimated log normal. The dotted black line is the theoretical distribution.

We now compare the convergence rate of GBM across the various algorithms. To highlight differences, we used a GBM model with parameters $\mu = -10$, $\sigma = 1$, and $S_0 = 100$. In the three following graphs, the x-axis is a log scale.

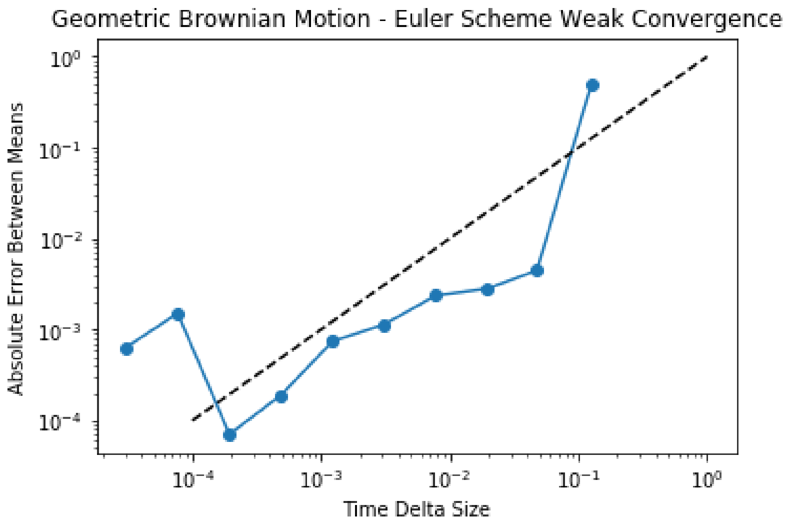

FIG. 3. GBM Weak Convergence for the Euler Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

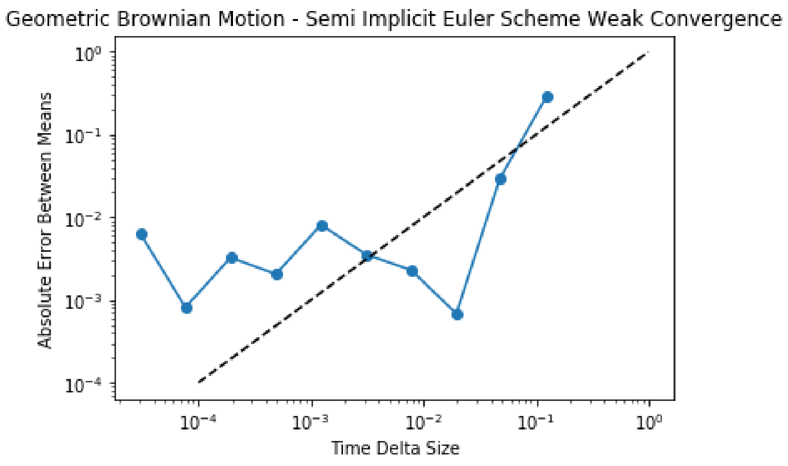

FIG. 4. GBM Weak Convergence for the Semi Implicit Euler Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

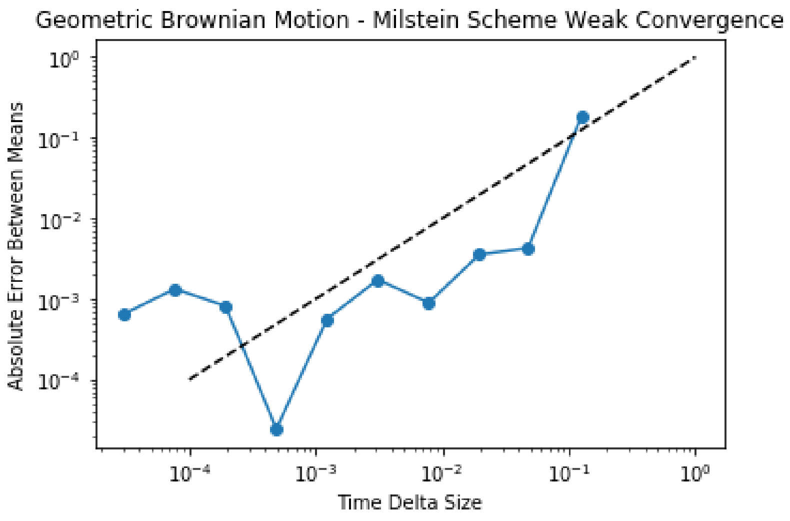

FIG. 5. GBM Weak Convergence for the Milstein Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

We note here that the actual convergence of the algorithms matches pretty similarly to the . All three seem to track the line with slope 1 on the log-log plot which implies that the convergence rate is 1. The only method in question is the semi implicit Euler scheme. We hypothesize the flattening of the curve is due to it reaching some lower bound earlier than the others.

B. Ornstein-Uhlenbeck Process

For the second case study, we considered the Ornstein-Uhlenbeck (OU) Process. The OU process has the following dynamics

$$dX_t = \mu(\theta - X_t)dt + \sigma dW_t$$

This process differs from GBM in that the noise is only additive. Intuitively, this process tends towards θ as time goes on. The rate at which it tends there is dictated by μ. The solution of the OU process is given by

$$X_t = X_0 e^{-\mu t} + \theta(1 - e^{-\mu T}) + \sqrt{\frac{\sigma}{\mu}(1 - e^{-2\mu t})}W_t$$

Where this is clearly normal with parameters

$$\mu_{dist} = X_0 e^{-\mu t} + \theta(1 - e^{-\mu T})$$
$$\sigma_{dist} = \sqrt{\frac{\sigma}{\mu}(1 - e^{-2\mu t})}$$

Where μ_{dist} and σ_{dist} are the parameters of the lognormal distribution and μ amd σ are parameters of the SDE. Below is a plot of sample paths for $\mu = 0.01$, $\theta = 1$, $\sigma = 0.01$, and $X_0 = 0$.

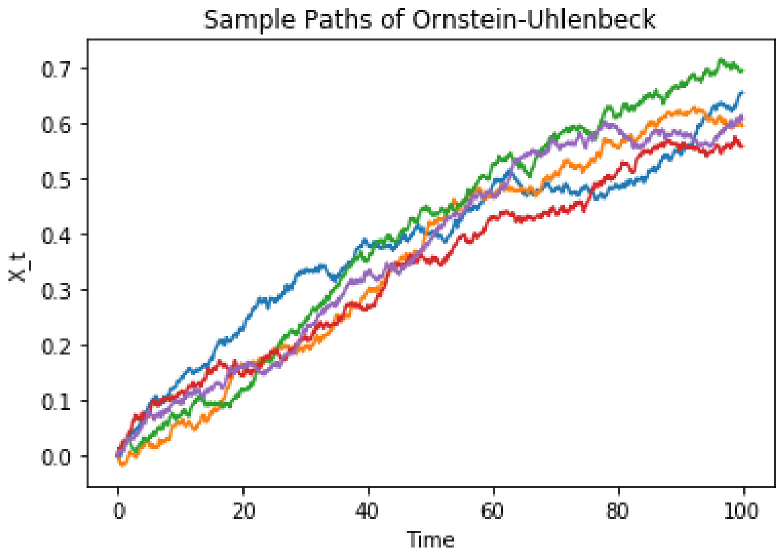

FIG. 6. Five sample paths of GBM with parameters $\mu = 0.01$, $\theta = 1$, $\sigma = 0.01$, and $X_0 = 0$

Generating 1000 sample paths and fitting them to a normal yields the following comparison to the expected normal distribution

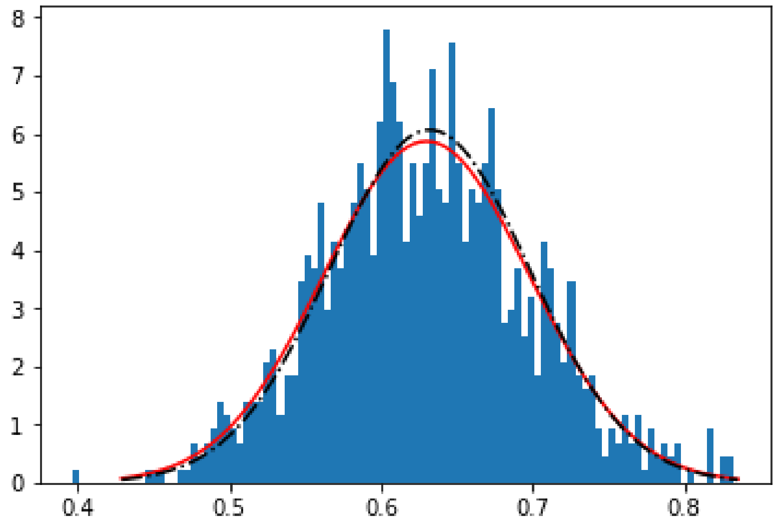

FIG. 7. The histogram of 1000 sample path endpoints. The red line is the estimated normal. The dotted black line is the theoretical distribution.

Finally, we compare the different schemes and their convergence rates. For the OU process, we are able to analytically determin $|X_T - X_T^{\delta t}|$ because everything is a Gaussian and thus the absolute value of the difference is a well known folded normal distribution. For the Euler scheme, we get that the convergence rates for both strong and weak seems to equal exactly 1. This is different than the weak convergence rate of 1 and strong convergence rate of 1/2 expected from theory, but the OU process is so well behaved that you do not see the reducting in strong convergence rates.

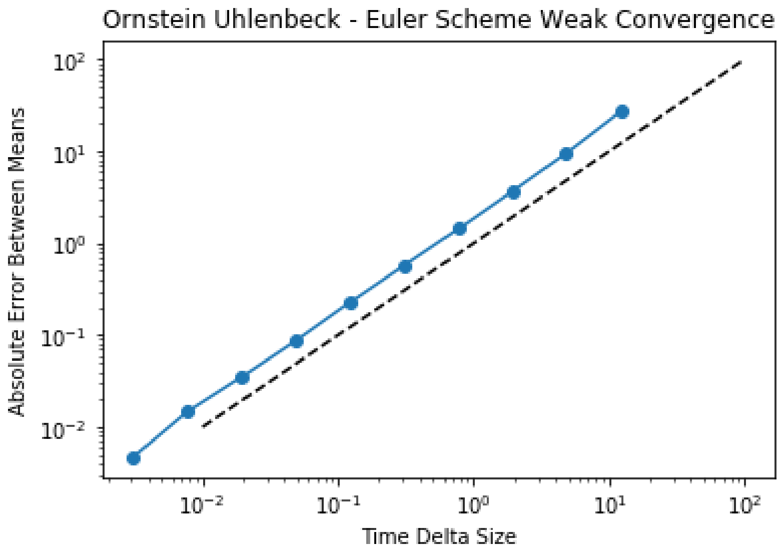

FIG. 8. OU Weak Convergence for the Euler Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

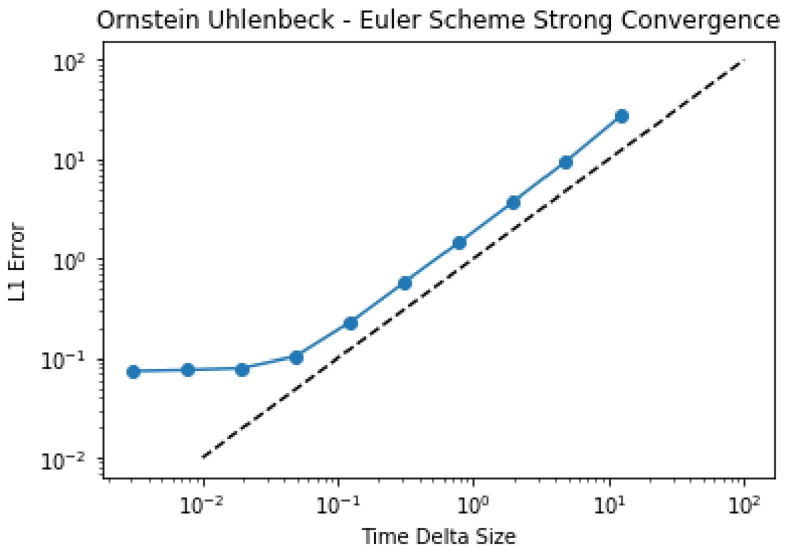

FIG. 9. OU Strong Convergence for the Euler Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

For the semi implicit Euler method, we get something of a different result. We get that the weak convergence rate is greater than 1, something close to 2. Whereas the strong convergence is 1.

While this does not match the theory of 1 and 1/2, they do differ by a factor of 2. This might be explained by the fact that the OU process is in itself very convergent as it has many nice properties due to its Gaussian nature.

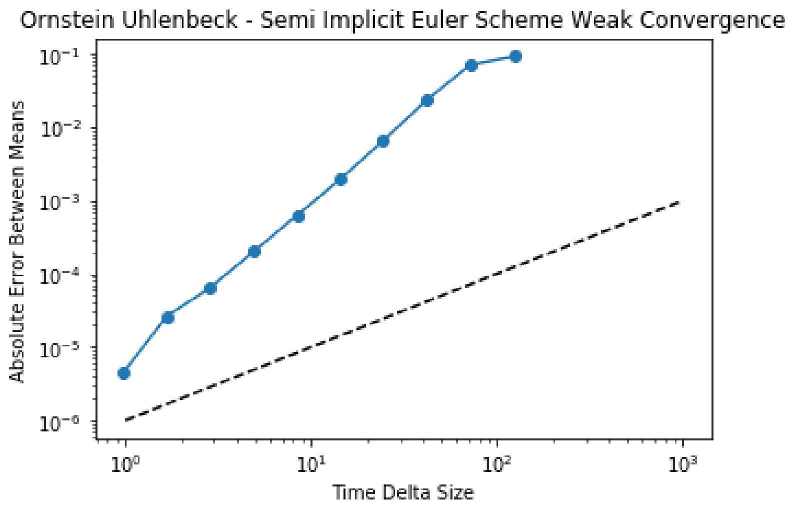

FIG. 10. OU Weak Convergence for the Semi Implicit Euler Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

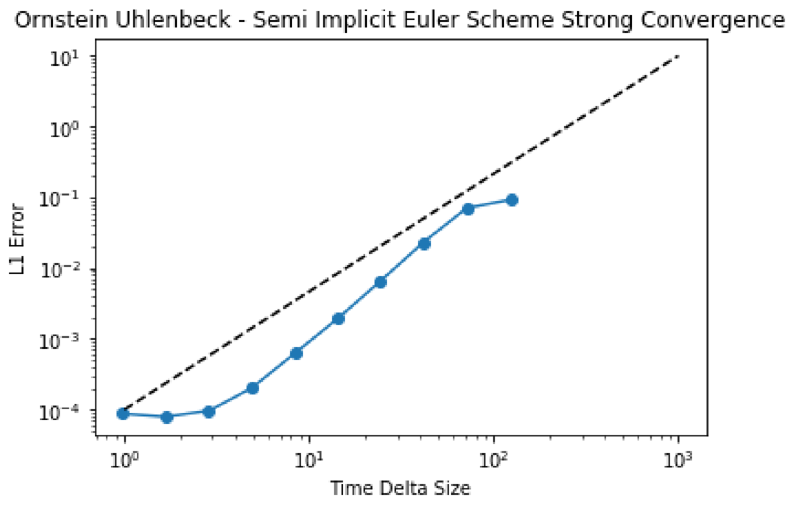

FIG. 11. OU Strong Convergence for the Semi Implicit Euler Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

Finally, for the Milstein algorithm, we get again that both convergence rates equal 1.

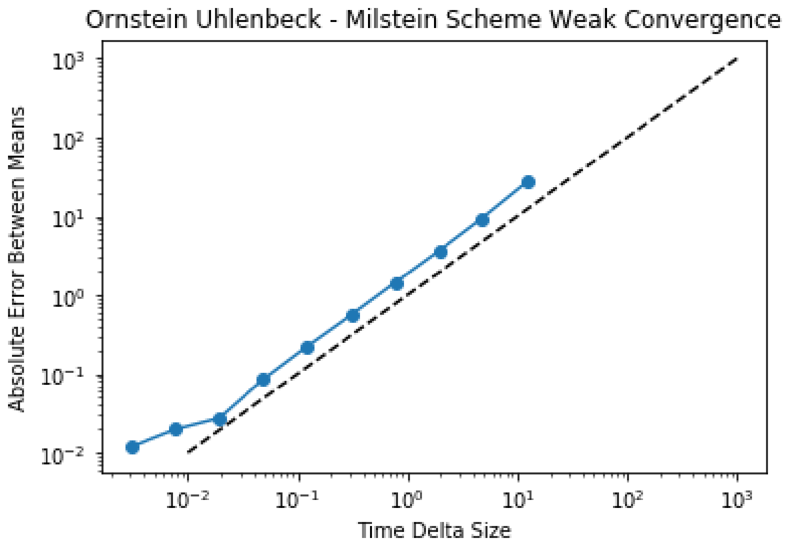

FIG. 12. OU Weak Convergence for the Milstein Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

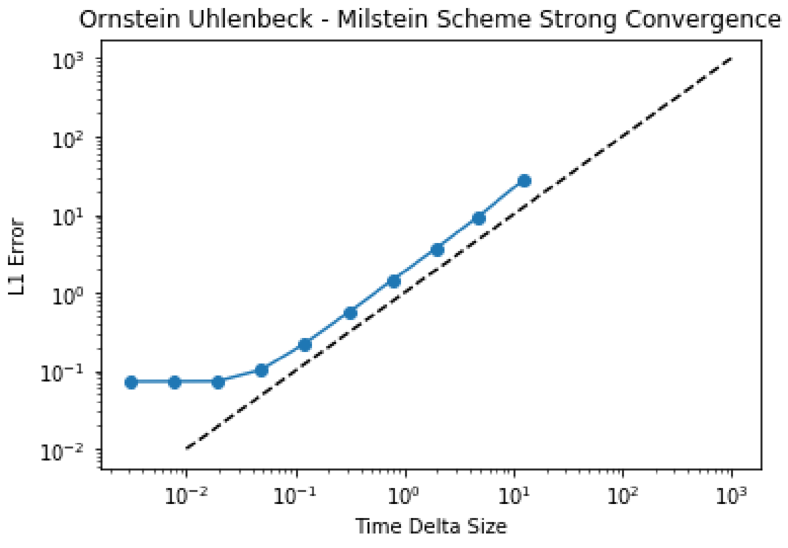

FIG. 13. OU Strong Convergence for the Milstein Method. Absolute Error of Means vs. Time Step Size. Dotted line has slope 1.

V. CONCLUSIONS

To conclude, this project has been a cursory overview of the numerical methods literature on stochastic differential equations. If one wanted to explore the field in depth, there are more advanced algorithms taking into account second order effects. Another immediate extention of the work on this project is to apply the algorithms to multidimensional systems and see which ones provide the best results. With regards to implementation detail, the work on this project yielded a few tangential avenues to explore.

One of the possible avenues is curve fitting. Curve fitting was done to compare the convergence rates of the various algorithms. In general, the problem of fitting a distribution to a dataset is very hard. Even if one knows the form of the distribution, the problem of finding the best parameters could be nonconvex. This is illustrated by a simple mixture of Gaussians. However, even if one has a convex problem, the classic problems of numerical analysis apply. What are the best schemes to minimize the objective, are these gradient steps stable (i.e. do we invert a matrix), how does one handle distributions that yield extremely large numbers? All of these concerns have to be addressed before attempting to fit a curve. Fortunately, in our cases, we studied equations with well known and well behaved solutions. In general, this area is ripe for exploration.

[1] H. J. Carmichael, *Statistical Methods in Quantum Optics 1: Master Equations and Fokker-Planck Equations*. Berlin: Springer, 1999.

[2] C. W. Gardiner, *Handbook of Stochastic Methods - For Physics, Chemisty, and the Natural Sciences*. Springer, 3 ed., 2004.

[3] M. Haugh, *Simulating Stochastic Differential Equations*. 2017.

[4] P. D. Drummond and I. K. Mortimer, “Computer simulations of multiplicative stochastic differential equations,” *Journal of Computational Physics*, vol. 93, no. 1, pp. 144–170, 1991.