Scott Hounsell and Sam Pfeiffer

Progress Report 10/26/15

We will begin with a brief summary of the paper by section in plain terms. We also provide a summary of our progress in replicating the results for a one-dimensional process. This can be found in Section 5. I have skipped some sections containing lengthy and involved proofs.

**1 Introduction**

We define some *d*-dimensional stochastic process by

, and .

The goal is to estimate

where

Our simulation which we will use to estimate will be denoted like so

, and

where and will be updated at exponential time points which I will henceforth refer to as “arrivals”.

Automatic differentiation technique from Elworthy’s formula from Malliavin calculus.

This technique allows us to deal with gradients and Hessian’s in the coefficient corrections (the Malliavin weights), avoiding the need to limit the PDE’s we wish to simulate.

Also, note that for our purposes, a single dimension suffices. Thus, throughout the paper we can consider .

**2 Regime switching diﬀusion representation**

This section begins by discussing some brief Lipschitz and other “niceness” conditions for the coefficients and functions.

A number of notations are defined here which I will summarize in simpler terms. is the time of the arrival (in fact the minimum of this with the final time , to stay inside the time range). is 0. is the number of arrivals strictly before and is a Poisson process with intensity . If we are at , then are the time and location of the most recent arrival. is the change in the driving Brownian motion between and (unless , in which case it is the change from to ). The authors also define shifted variables , the arrival since t, and , the number of arrivals between and (where ). Note that this is not used at all throughout the rest of the paper. Some more notation is covered but serves mainly to confuse the reader.

Next, we define the Malliavin weights and as

where is the gradient, is the Hessian, and (essentially some proxy for ).

where

and

Essentially, it seems that this notation is just saying that the parameters in the right tuple are unpacked into each function in the left tuple (i.e. and ). appears to be understood as error terms for our estimators of and , which are being “corrected” via and .

Next we define , which amounts to an estimator for in that it should have the same expectation at .

(Note that we start from , and ). In this case the colon operator is defined as . Thus the term inside the product is something like . This looks like the error of each coefficient dotted with a correction weight.

It is stated and subsequently proven that has the same expectation as . In addition there is a brief section discussing error analysis.

**3 The constant diffusion coefficient case**

This section offers an “algorithm” for SDE’s with drift functions and a constant matrix diffusion coefficient:

Note that . We assume that and are Lipschitz and “nice”.

**3.1 The algorithm**

It is proposed that we choose and . The discretized process is thus

,

Next, the formulas for the weights are simply given, and the reader is referred to the likelihood ratio method of Broadie and Glasserman. Studying that paper, I have yet to find how the application yields the following weights, but nonetheless, they are

Note that the is given but is not needed for this constant diffusion case (no correction term necessary). Therefore, starting at 0, our estimator for is

where apparently represents the combined error/correction product and is defined as

A proof of the square integrability of follows, but is skipped for the purposes of this summary. Similarly the author justifies a choice for the intensity , which we will skip for now, as well.

**4 One-dimensional driftless SDE**

The process we wish to simulate in this section is .

**4.1 The algorithm**

It is proposed that we choose and

Note that and represent the time and location at the most recent arrival. Technically all of our and functions have these parameters in front, since our process is regime-switching, however, they have been irrelevant and unnecessary notation until now. Essentially, looks like a Taylor expansion across the time between now and the last regime change (sort of our last “known” time point).

Therefore our discretized process is

Depending on whether or not the partial derivative is zero, there are two potential explicit solutions given. For brevity the following notations are used

If ,

Otherwise,

Our estimator for is

where again apparently represents the combined error/correction product and is defined as

where

Since is of infinite variance, the author provides an antithetic variable , which we will not cover here.

The choice of Malliavan weights are proven to be satisfactory following this section. However, their derivation is not shown, and we will skip this proof.

**5 Numerical examples**

This section provides a numerical example of the Exact method described in the previous sections. Equation 5.1 is a driftless process with a non-constant diffusion coefficient. The implementation for this type of process is described in section 4. Equation 5.2 is a process with a non-constant drift coefficient and a constant diffusion coefficient of 1. The implementation for this type of process is described in section 3. Note that equation 5.2 is the same as in 5.1 with a Lamperti transform applied. Figure 1 on page 18 shows the implied volatility of the process in 5.1 and 5.2 using both the Euler scheme and the Exact method described in the paper.

The code we have provided implements equation 5.1 using both the Euler scheme and the Exact method. Using Euler scheme, we match the implied volatility shown in figure 1. However, our implementation of the Exact method does not converge and sometimes gives negative values. We are not sure why.

Furthermore, we did not implement equation 5.2 since it depends on equation 5.1 which is not working properly. We did however implement the Exact simulation methods for processes of the same form as equation 5.2 (using section 3). We applied it to the process

This process did not converge either using the Exact method. Again, we are not sure why. Note that the simulation of this process using Euler scheme *does* converge.

One idea that we have not yet implemented, is to code the Exact simulation method described in Exact Simulation of Diffusions by Beskos and Roberts (attached). The paper we have been referencing improves upon the Beskos and Roberts paper since it works for multi-dimensional processes. Since this does not concern us, we will try to implement the method described in Beskos and Roberts.

**6 Further discussions**

The following subsections discuss application to general drift and diffusion along with path-dependent cases.

**6.1 The general drift and diffusion case**

Here, the paper offers a formula and gives general weights. However, the author remarks that our estimator, , has infinite variance and is therefore unsuitable in application in general. The remaining sections concerning path dependence are irrelevant to us.