Implementation with sensitivity analysis on Hybrid scheme for Brownian semistationary processes

Weiyi Chen[*], Rongxin Yu[†], Wenbo Zhang[‡]

November 2, 2015

Abstract

In this report, we implement the hybrid scheme for Brownian semistationary processes [4] (2015), which is an approximation of the process via discretizing the stochastic integral representation of the process in the time domain, and it is a combination of Wiener integrals of the power function and a Riemann sum. We exemplify the use of the hybrid scheme by two numerical experiments, where we replicate the study of Monte Carlo option pricing in the rough Bergomi model of Bayer et al. [3], and analyze stability and sensitivity of parameters of hybrid scheme on price, respectively.

Keywords: Stochastic simulation; discretization; Brownian semistationary process; stochastic volatility; regular variation; estimation; option pricing; rough volatility; volatility smile.

[*]Department of Mathematics, Baruch College, CUNY. `weiyi.chen@baruchmail.cuny.edu`
[†]Department of Mathematics, Baruch College, CUNY. `rongxin.yu@baruchmail.cuny.edu`
[‡]Department of Mathematics, Baruch College, CUNY. `wenbo.zhang@baruchmail.cuny.edu`

1 Introduction

1.1 Brownian semistationary processes

A $\mathcal{BSS}$ process X is defined via the integral representation

$$X(t) = \int_{-\infty}^{t} g(t-s)\sigma(s)dW(s) \tag{1.1}$$

where W is a two-sided Brownian motion providing the fundamental noise innovations, the amplitude of which is modulated by a stochastic volatility (intermittency) process σ that may depend on W. This driving noise is then convolved with a deterministic kernel function g that specifies the dependence structure of X.

In the applications mentioned above, the case where X is not a semimartingale is particularly relevant. This situation arises when the kernel function g behaves like a power-law near zero; more specifically, when for some $\alpha \in (-\frac{1}{2}, \frac{1}{2})$ and $\alpha \neq 0$,

$$g(x) \propto x^{\alpha} \text{ for small } x > 0 \tag{1.2}$$

1.2 Hybrid scheme

In Bennedsen et al.'s paper [4], we study a new discretization scheme for $\mathcal{BSS}$ processes based on approximating the kernel function g in the time domain. The starting point is the Riemann-sum discretization of (1.1). The Riemann-sum scheme builds on an approximation of g using step functions, which has the pitfall of failing to capture appropriately the steepness of g near zero. In particular, this becomes a serious defect under (1.2) when $\alpha \in (-\frac{1}{2}, 0)$.

In hybrid scheme, the problem is mitigated by approximating g using an appropriate power function near zero and a step function elsewhere. The resulting discretization scheme can be realized as a linear combination of Wiener integrals with respect to the driving Brownian motion W and a Riemann sum.

1.3 Truncated Brownian semistationary process

It is useful to extend the hybrid scheme to a class of non-stationary processes that are closely related to $\mathcal{BSS}$ processes. This extension is important in connection with an application to the rough Bergomi model below. More precisely, we consider processes of the form

$$Y(t) = \int_{0}^{t} g(t-s)\sigma(s)dW(s), t \leq 0, \tag{1.3}$$

where the kernel function g, volatility process σ and driving Brownian motion W are as before. We call Y a truncated Brownian semistationary ($\mathcal{TBSS}$) process, as Y is obtained from the $\mathcal{BSS}$ process X by truncating the stochastic integral in (1.1) at 0.

1.4 Implementation

To replicate the hybrid scheme implementation recipe from Bennedsen et al.'s paper [4], we perform the numerical experiment of Monte Carlo option pricing in the rough Bergomi stochastic volatility model of Bayer et al. [3], in R programming language. We use the hybrid scheme to simulate the volatility process in this model and we verify the resulting implied volatility smiles are indistinguishable from those simulated using a method that involves exact simulation of the volatility process. After verification we optimize the codes via vectorizing and utilize multiprocessing technique to accelerate Monte Carlo methods in Python.

1.5 Sensitivity analysis

After replication, we perform further numerical experiment of sensitivity analysis on option pricing in the rough Bergomi stochastic volatility model of Bayer et al. [3]. Parameter values used in the rBergomi model include $S_0, \epsilon, \eta, \alpha, \rho$ listed on Bennedsen et al.'s paper [4]. Parameter values used in Hybrid Scheme include number of periods n, the period index to separate the approximate power function near zero and the step function κ, and the time period unit T. And there are other parameter values coming from Monte Carlo method. We keep track of the option price by changing one of the parameter while keep other constant, to figure out the sensitivity of each parameter on option price.

1.6 Report Organization

The rest of this report is organized as follows. In Section 2 we recall the implementation of Hybrid Scheme for $\mathcal{BSS}$ process and introduce our initial assumptions for parameters, including the extension of the scheme to a class of truncated BSS processes, then proceed to option pricing under rough volatility. Section 3 briefly discusses the code structure with results, and presents the numerical experiments mentioned above. Section 4 contains discussing on computational complexity and implementation improvements from both theory perspective, i.e. vectorization programming, and technique perspective, i.e. multiprocessing. Finally, Section 5 presents the sensitivities of the experiment results.

2 Implementation

2.1 Brownian semistationary process

2.1.1 Numerical recall

Simulating the $\mathcal{BSS}$ process X on the equidistant grid $\{0, \frac{1}{n}, \frac{2}{n}, \dots, \frac{\lfloor nT \rfloor}{n}\}$ for some $T > 0$ using the hybrid scheme entails generating

$$X_n(\frac{i}{n}), i = 0, 1, \dots, \lfloor nT \rfloor. \tag{2.1}$$

Provided that we can simulate the random variables

$$\begin{aligned} W^n_{i,j} &:= \int_{\frac{i}{n}}^{\frac{i+1}{n}} (\frac{i+j}{n} - s)^\alpha dW(s), i = -N_n, -N_n + 1, \dots, \lfloor nT \rfloor, j = 1, \dots, \kappa \\ W^n_i &:= \int_{\frac{i}{n}}^{\frac{i+1}{n}} dW(s), i = -N_n, -N_n + 1, \dots, \lfloor nT \rfloor \\ \sigma^n_i &:= \sigma\left(\frac{i}{n}\right), i = -N_n, -N_n + 1, \dots, \lfloor nT \rfloor \end{aligned} \tag{2.2}$$

We can compute (2.1) via the formula

$$\begin{aligned} X_n(\frac{i}{n}) &= \check{X}_n(\frac{i}{n}) + \hat{X}_n(\frac{i}{n}) \\ &= \sum_{k=1}^{\kappa} L_g\left(\frac{k}{n}\right) \sigma^n_{i-k} W^n_{i-k,k} + \sum_{k=\kappa+1}^{N_n} g\left(\frac{b^*_k}{n}\right) \sigma^n_{i-k} W^n_{i-k} \end{aligned} \tag{2.3}$$

In order to simulate (2.2), it is instrumental to note that the $\kappa + 1$-dimensional random vectors

$$\boldsymbol{W}^n_i = (W^n_i, W^n_{i,1}, \dots, W^n_{i,\kappa}, i = -N_n, N_n + 1, \dots, \lfloor nT \rfloor - 1 \tag{2.4}$$

are i.i.d. according to a multivariate Gaussian with mean zero and covariance matrix Σ given by

$$\begin{aligned} \Sigma_{1,1} &:= \frac{1}{n}, \\ \Sigma_{1,j} &= \sum_{j,1} := \frac{(j-1)^{\alpha+1} - (j-2)^{\alpha+1}}{(\alpha+1)n^{\alpha+1}}, \\ \Sigma_{j,j} &:= \frac{(j-1)^{2\alpha+1} - (j-2)^{2\alpha+1}}{(2\alpha+1)n^{2\alpha+1}}, \\ \Sigma_{j,k} &:= \frac{1}{(j-k)(\alpha+1)n^{2\alpha+1}}(((j-1)(k-1))^{\alpha+1} F_1(1, 2(\alpha+1), \alpha+2, \frac{k-1}{k-j}) \\ &\quad - ((j-2)(k-2))^{\alpha+1} F_1(1, 2(\alpha+1)), \alpha+2, \frac{k-2}{k-j}) \end{aligned} \tag{2.5}$$

for $j, k = 2, \dots, \kappa + 1$ such that $j \neq k$, where F_1 stands for the *Gauss hypergeometric function*.

Thus, $\{\boldsymbol{W}_i^n\}_{i=-N_n}^{\lfloor nT \rfloor - 1}$ can be generated by taking independent draws from the can be generated by taking independent draws from the multivariate Gaussian distribution $N_{\kappa+1}(\mathbf{0}, \Sigma)$. If the volatility process σ is driven by a standard Brownian motion Z, correlated with W, say, one could reply on a factor decomposition

$$Z(t) := \rho W(t) + \sqrt{1 - \rho^2} W_{\perp}(t), t \in \mathcal{R} \tag{2.6}$$

where $\rho \in [-1, 1]$ is the correlation parameter and $\{W_{\perp}(t)\}t \in [0, T]$ is a standard Brownian motion independent of W. Then one would first generate $\{W_n\}\lfloor nT \rfloor - 1$, use (2.6) to generate $\{Z(\frac{i+1}{n}) - Z(\frac{i}{n})\}_{i=-N_n}^{\lfloor nT \rfloor - 1}$ and employ some appropriate approximate method to produce $\{\sigma_i^n\}_{i=-N_n}^{\lfloor nT \rfloor - 1}$ thereafter.

2.1.2 Code

The multivariate Gaussian with mean zero and covariance matrix Σ implemented as

```r
covMatrix <- function(n, kappa){
  ### generate the covariance matrix
  ### @param n: the distance between two points is 1/n
  ### @param kappa: how many power low terms to include around zero

  sigma <- matrix(0, nrow=kappa+1, ncol=kappa+1)
  sigma[1,1] <- 1/n
  for(j in 2:(kappa+1)){
    sigma[1, j] <- ((j-1)^(alpha+1)-(j-2)^(alpha+1)) / ((alpha+1)*n^(
        alpha+1))
    sigma[j, 1] <- ((j-1)^(alpha+1)-(j-2)^(alpha+1)) / ((alpha+1)*n^(
        alpha+1))
  }
  for(j in 2:(kappa+1)){
    for(k in 2:(kappa+1)){
      if(j==k){
        sigma[j,k] <- ((j-1)^(2*alpha+1)-(j-2)^(2*alpha+1)) / ((2*alpha
            +1)*n^(2*alpha+1))
      }
      else{
      }
    }
  }
  return(sigma)
}

convolve <- function(x, y){
    return(cumsum(x * rev(y)))
}

bstar <- function(k, alpha){
  result <- ((k^(alpha+1) - (k-1)^(alpha+1)) / (alpha+1)) ^ (1/alpha)
  return(result)
}
```

2.2 Truncated Brownian semistationary process

2.2.1 Numerical recall

In the case of the $\mathcal{TBSS}$ process Y, the observations $Y_n(\frac{i}{n}), i = 0, 1, \dots, \lfloor nT \rfloor$, given by the hybrid scheme can be computed via

$$Y_n(\frac{i}{n}) = \sum_{k=1}^{\min(i,\kappa)} L_g(\frac{k}{n})\sigma^n_{i-k}W^n_{i-k,k} + \sum_{k=\kappa+1}^{i} g(\frac{b^*_k}{n}\sigma^n_{i-k}W^n_{i-k}), \tag{2.7}$$

using the random vectors $\{\boldsymbol{W}^n_i\}_{i=0}^{\lfloor nT \rfloor -1}$ and random variables $\{\sigma^n_i\}_{i=0}^{\lfloor nT \rfloor -1}$.

In the hybrid scheme, it typically suffices to take κ to be at most 3. Thus, in (2.3), the first sum $\check{X}(\frac{i}{n})$ requires only a negligible computational effort. By contrast, the number of terms in the second sum $\hat{X}(\frac{i}{n})$ increases as $n \to \infty$. It is then useful to note that

$$\hat{X}(\frac{i}{n}) = \sum_{k=1}^{N_n} \Gamma_k \Xi_{i-k} = (\Gamma \star \Xi)_i \tag{2.8}$$

where

$$\Gamma_k := \begin{cases} 0, k = 1, 2, \dots, \kappa \\ g(\frac{b^*_k}{n}, k = \kappa+1, \kappa+2, \dots, N_n \end{cases} \qquad \Xi_k := \sigma^n_k W^n_k, k = -N_n, -N_n+1, \dots, \lfloor nT \rfloor - 1 \tag{2.9}$$

and $\Gamma \star \Xi$ stands for the discrete convolution of the sequences Γ and Ξ.

2.2.2 Code

```
    Simulation <- function(n, kappa, T){
      W <- mvrnorm(floor(n*T), mu=rep(0, kappa+1), Sigma=covMatrix(n,
          kappa))
      Wperp <- rnorm(floor(n*T), sd=sqrt(1/n))
      Z <- rho * W[,1] + sqrt(1-rho*rho)*Wperp

      Gamma <- sapply(seq(1:floor(n*T)), function(x){(bstar(x, alpha)/n
          )^alpha})
      Gamma[1:kappa] <- 0
      Y2 <- convolve(Gamma, W[,1])
      Y1 <- rep(0, floor(n*T))
      for(i in 1:floor(n*T)){
        Y1[i] <-0
        for (k in 1:min(i,kappa)){
        Y1[i] = Y1[i] + W[i+1-k,k+1]
        }
      }
      Y <- Y1+Y2 ## The simulated series of main interst
      v <- xi*exp(eta*sqrt(2*alpha+1)*Y - eta*eta/2*sapply(seq(1:floor(
          n*T)),function(x){(x/n)^(2*alpha+1)}))
      S <- S0 * exp(sum(v^0.5*Z) - 1/2*sum(v)/n)
      return(S)
    }
```

2.3 Option pricing under rough volatility

2.3.1 Numerical recall

We study Monte Carlo option pricing in the rough Bergomi (rBergomi) model of Bayer et al. [3]. In the rBergomi model, the logarithmic spot variance of the price of the underlying is modeled by a rough Gaussian process.

More precisely, the price of the underlying in the rBergomi model with time horizon $T > 0$ is defined, under an equivalent martingale measure identified with $\mathbb{P}$, as

$$S(t) := S(0) \exp\left(\int_0^t \sqrt{v(s)} dZ(s) - \frac{1}{2}\int_0^t v(s) ds\right), t \in [0,T], \tag{2.10}$$

using the spot variance process

$$v(t) := \xi_0 \exp\left(\eta\sqrt{2\alpha+1}\int_0^t (t-s)^\alpha dW(s) - \frac{\eta^2}{2} t^{2\alpha+1}\right), t \in [0,T]. \tag{2.11}$$

Above, $S(0) > 0$, $\eta > 0$ and $\alpha \in (-\frac{1}{2}, 0)$ are deterministic parameters, and Z is a standard Brownian motion given by

$$Z(t) := \rho W(t) + \sqrt{1-rho^2} W_\perp(t), t \in [0,T], \tag{2.12}$$

where $\rho \in (-1,1)$ is the correlation parameter and $\{W_\perp(t)\}_{t\in[0,T]}$ is a standard Brownian motion independent of W. The process $\{\xi_0(t)\}_{t\in[0,T]}$ is the forward variance curve, which we assume here to be flat, $\xi_0(t) = \xi > 0$ for all $t \in [0,T]$.

We aim to compute using Monte Carlo simulation the price of a European call option struck at $K > 0$ with maturity T, which is given by

$$C(S(0), K, T) := \mathbb{E}[(S_T - K)^+] \tag{2.13}$$

We use the hybrid scheme to simulate Y. As the hybrid scheme involves simulating increments of the Brownian motion W driving Y, we can conveniently simulate the increments of Z, needed for the Euler discretization of S, using the representation (2.12).

2.3.2 Code

```
BSImpliedVolCall <- function(S0, K, T, r, C){
  nK <- length(K);
  sigmaL <- rep(1e-10,nK);
  CL <- BSFormula(S0, K, T, r, sigmaL);
  sigmaH <- rep(10,nK);
  CH <- BSFormula(S0, K, T, r, sigmaH);
  while (mean(sigmaH - sigmaL) > 1e-10)
  {
    sigma <- (sigmaL + sigmaH)/2;
    CM <- BSFormula(S0, K, T, r, sigma);
    CL <- CL + (CM < C)*(CM-CL);
    sigmaL <- sigmaL + (CM < C)*(sigma-sigmaL);
    CH <- CH + (CM >= C)*(CM-CH);
    sigmaH <- sigmaH + (CM >= C)*(sigma-sigmaH);
  }
  return(sigma);
}

BSFormula <- function(S0, K, T, r, sigma){
  x <- log(S0/K)+r*T;
  sig <- sigma*sqrt(T);
  d1 <- x/sig+sig/2;
  d2 <- d1 - sig;
  pv <- exp(-r*T);
  return( S0*pnorm(d1) - pv*K*pnorm(d2));
}

impvol <- function(k, st, T){
  payoff <- (st > exp(k)) * (st - exp(k))
  return(BSImpliedVolCall(1, exp(k), T, 0, mean(payoff)))
}
```

3 Numerical experiment and result

4 Complexity improvements

5 Sensitivities analysis

6 Conclusion

Acknowledgments

We are very grateful to Jim Gatheral's guide.

References

[1] Barndorff-Nielsen, Ole E and Schmiegel, Jürgen Ambit processes; with applications to turbulence and tumour growth *Springer* (2007)

[2] Barndorff-Nielsen, Ole E and Schmiegel, Jürgen Advanced financial modelling Radon Ser. Comput. Appl. Math (2009)

[3] Bayer, Christian, Friz, Peter K and Gatheral, Jim Available at SSRN 2015

[4] Bennedsen, Mikkel, Lunde, Asger and Pakkanen, Mikko S Hybrid scheme for Brownian semistationary processes, *arXiv preprint arXiv:1507.03004* (2015)

[5] Gatheral, J., The Volatility Surface: A Practitioner's Guide, Wiley Finance (2006).

[6] Gatheral, J., Hsu, E.P., Laurence, P., Ouyang, C., and Wang, T.-H., Asymptotics of implied volatility in local volatility models, *Mathematical Finance* (2011) forthcoming.

@articlegatheral2012asymptotics, title=Asymptotics of implied volatility in local volatility models, author=Gatheral, Jim and Hsu, Elton P and Laurence, Peter and Ouyang, Cheng and Wang, Tai-Ho, journal=Mathematical Finance, volume=22, number=4, pages=591–620, year=2012, publisher=Wiley Online Library