Computing Densities for Stochastic Differential Equations

Harish S. Bhat, Applied Mathematics, University of California, Merced (hbhat@ucmerced.edu)

Motivation

Consider stochastic differential equation (SDE)

$$dX_t = f(X_t;\theta)\,dt + g(X_t;\theta)\,dW_t.$$

Often we wish to estimate parameters θ for f (drift) and g (diffusion), using data $\mathbf{x}$ consisting of observations of X_t at times $\{j\Delta t\}_{j=0}^{N}$. By Markov property, log likelihood is

$$\log p(\mathbf{x}\,|\,\theta) = \sum_{j=0}^{N-1} \log p(x_{j+1}\,|\,x_j,\theta) + C.$$

Computing each Markovian piece $p(x_{j+1}|x_j,\theta)$ is critical for estimation/inference of θ.
DTQ (Density Tracking by Quadrature) = direct approach to computing these Markovian pieces.

Derivation

Discretize SDE in time with fixed time step $h > 0$. Obtain discrete-time Markov chain with continuous state space. Chapman-Kolmogorov equation for this Markov chain is

$$\widetilde{p}(x,t_{i+1}) = \int_{\mathbb{R}} G(x,y)\widetilde{p}(y,t_i)\,dy.$$

Transition kernel G is normal PDF over x with mean $y + f(y)h$ and variance $g^2(y)h$.

Discretization of the above integral—via quadrature—yields DTQ. E.g., trapezoidal rule on equispaced grid $\{mk\}_{m=-M}^{M}$:

$$\widehat{p}(x_i,t_{i+1}) = \sum_{m=-M}^{m=M} \underbrace{kG(x_i,y_m)}_{\mathcal{G}_{im}} \widehat{p}(y_m,t_i)$$

Method reduces to iterated matrix multiplication:

$$\widehat{\mathbf{p}}(t_{i+1}) = \mathcal{G}\widehat{\mathbf{p}}(t_i).$$

Set $t_0 = 0$ and $h = (\Delta t)/F$ for integer F. Start with $\widetilde{p}(x,t_0) = \delta(x - x_j)$ so that $\widetilde{p}(x,t_1) = G(x,x_j)$. Now step forward in time to compute

$$\widehat{p}(x_{j+1},t_F) \approx p(x_{j+1}\,|\,x_j,\theta).$$

Theorems

- We have proven that $\|\widehat{p} - \widetilde{p}\|_{L^1}$ goes to zero exponentially in h, as $h \to 0$, provided that $k \propto h^\rho$ for $\rho > 1/2$. Here h and k are the temporal and spatial grid spacings, respectively.
- Earlier result, due to Bally and Talay, showed that $\|\widetilde{p} - p\|_{L^1} = O(h)$. Combining this with our result shows convergence of $\widehat{p} \to p$ in L^1.
- We have also proved Chernoff bound that explains how domain truncation affects accuracy.
- With no hand-tuning or additional constraints, obtain both nonnegativity and approximate normalization of computed densities.

Implementation

Suppose we have the SDE

$$dX_t = -X_t dt + dW_t.$$

With initial condition $X_0 = 0$, what is $p(x,1)$? Here is a naïve R implementation:

```
integrandmat <- function(xvec,yvec,h,f,g)
{
  X=replicate(length(yvec),xvec)
  Y=t(replicate(length(xvec),yvec))
  out = exp(-(X-Y-f(Y)*h)^2/(2*g(Y)^2*h))
  out = out/(g(Y)*sqrt(2*pi*h))
  return(out)
}

# simulation parameters
T = 1
s = 0.75
h = 0.02
init = 0
numsteps = ceiling(T/h)
k = h^s
yM = k*(pi/(k^2))
xvec = seq(-yM,yM,by=k)

# drift and diffusion functions
f <- function(x) { return(-x) }
g <- function(x) { return(rep(1,length(x))) }

# pdf after one time step with Dirac \delta(x-init) IC
A = integrandmat(xvec,xvec,h,f,g)
phat = exp(-(xvec-init-f(init)*h)^2/(2*g(init)^2*h))
phat = phat/sqrt(2*pi*g(init)^2*h)
phat = as.matrix(phat)

# main iteration loop
for (i in c(2:numsteps)) phat = k*(A%*%phat)
```

Rdtq

Rdtq—available on CRAN—provides R package for DTQ:

```
mydrift = function(x) { -x }
mydiff = function(x) { rep(1,length(x)) }
test = rdtq(h=0.1,k=0.01,bigm=250,init=0,fT=1,
            drift=mydrift,diffusion=mydiff,method="sparse")
plot(test$xvec,test$pdf,type='l')
```

Rdtq accommodates inline C++ implementation of f and g:

```
require(Rcpp)
sourceCpp(code = '#include <Rcpp.h>
          using namespace Rcpp;
          double drift(double& x)
          {
          return(-x);
          }
          double diff(double& x)
          {
          return(1.0);
          }
          typedef double (*funcPtr)(double& x);
          // [[Rcpp::export]]
          XPtr<funcPtr> driftXPtr()
          {
          return(XPtr<funcPtr>(new funcPtr(&drift)));
          }
          // [[Rcpp::export]]
          XPtr<funcPtr> diffXPtr()
          {
          return(XPtr<funcPtr>(new funcPtr(&diff)));
          }')
k = 0.01
M = 250
test = rdtq(h=0.1,k,bigm=M,init=0,fT=1,
drift=driftXPtr(),diffusion=diffXPtr(),method="cpp")
```

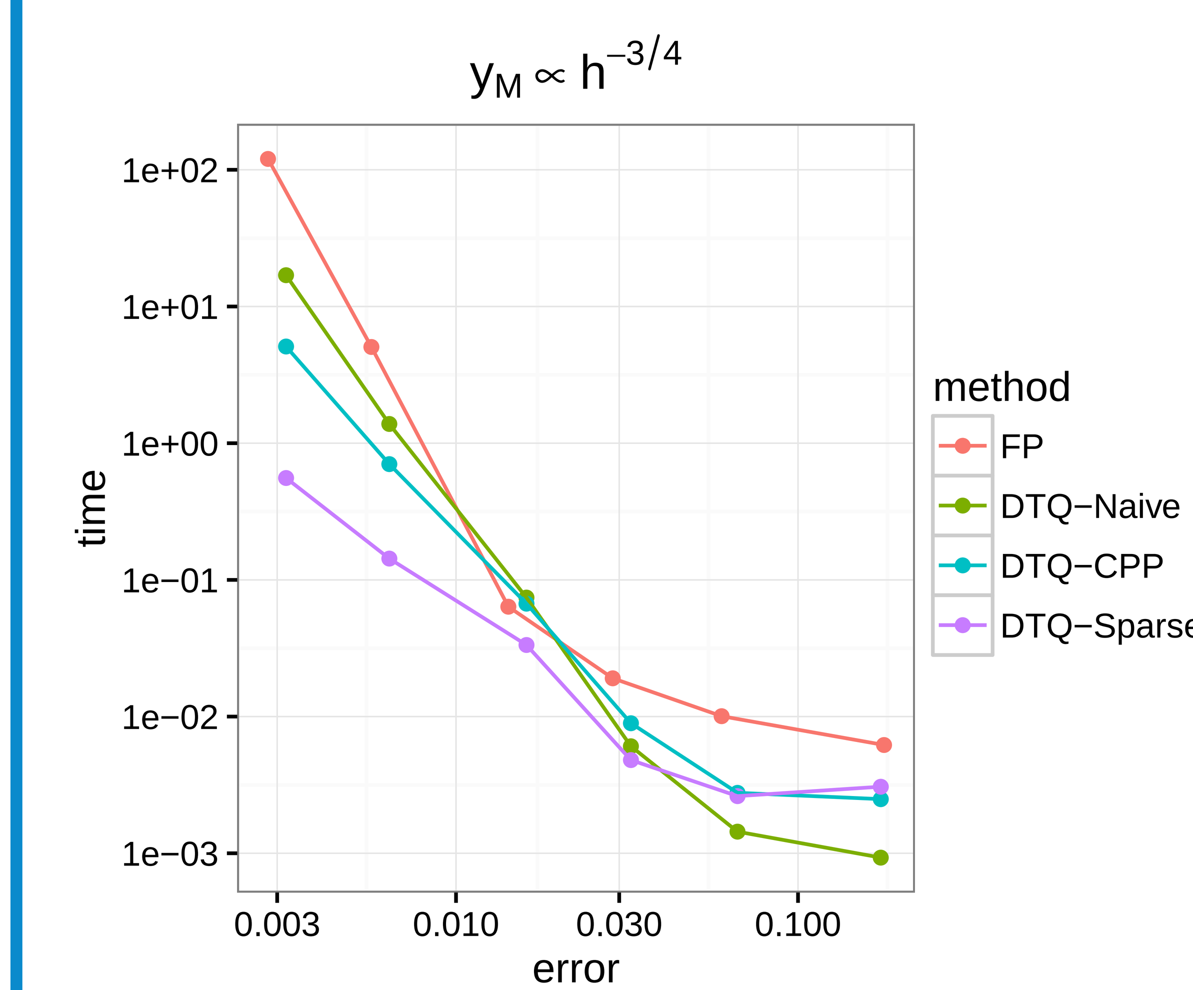
For More Information

- H. S. Bhat and R. W. M. A. Madushani (2018). Density tracking by quadrature for stochastic differential equations. arXiv:1610.09572 [stat.CO]
- `install.packages('Rdtq')`; `library(Rdtq)`; https://cran.r-project.org/package=Rdtq
- http://faculty.ucmerced.edu/hbhat/publications.html and https://github.com/hbhat4000/sdeinference

Extensions

- Improved $O(h^2)$ discretization in time of original SDE.
- Lévy SDE: simple modification to $G(x,y)$.
- Improved quadrature: sparse grids.
- Multidimensional versions.
- Adjoint DTQ method for fast, accurate computation of $\nabla_\theta \log p(\mathbf{x}\,|\,\theta)$.

Timing



DTQ methods are nearly 100x faster than FP (Fokker-Planck) solver for same level of error. Here we compare three implementation of DTQ:

- DTQ-Naïve: same method as in simple R code to the left.
- DTQ-CPP: C++ implementation using Rcpp and RcppArmadillo, "method=cpp" in Rdtq.
- DTQ-Sparse: sparse matrix implementation using Matrix package in R, "method=sparse" in Rdtq. Sparsity of $\mathcal{G}$ follows naturally from tail decay of Gaussians.

Acknowledgments

We are grateful for National Science Foundation support via award DMS-1723272.