Risk analysis and high-dimensional integrals

Alexander Podkopaev, Nikita Puchkin, Igor Silin

Skolkovo Institute of science and technology

December 10, 2016

1 Description of algorithm and PDE solver

We implement a solver for numerical solution of PDE

$$\frac{\partial}{\partial t}u(x,t) = \sigma\frac{\partial^2}{\partial x^2}u(x,t) - V(x,t)u(x,t), \quad t \in [0,T],\, x \in \mathbb{R}$$
$$u(x,0) = f(x)$$

The exact solution is given by the Feynman-Kac formula

$$u(x,T) = \int\limits_{C\{x,0;T\}} f(\xi(T)) \exp\left\{-\int\limits_0^T V(\xi(\tau), T-\tau)d\tau\right\}\mathcal{D}_\xi,$$

where the integration is done over the set $C\{x,0;T\}$ of all continuous paths $\xi(T) : [0,T] \to \mathbb{R}$ from the Banach space $\Xi([0,T],\mathbb{R})$ starting at $\xi(0) = x$ and stopping at arbitrary endpoints at time T. $\mathcal{D}_\xi$ is the Wiener measure and $\xi(t)$ is the Wiener process.

For numerical computation one can break the time range $[0,T]$ into n intervals by points

$$\tau_k = k\delta t, \qquad 0 \le k < n, \qquad n : \tau_n = T$$

The average path of a Brownian particle $\xi(\tau_k)$ after k steps is defined as

$$\xi^{(k)} = \xi(\tau_k) = x + \xi_1 + \cdots + \xi_k,$$

where every random step ξ_i, $1 \le i \le k$, is independently taken from a normal distribution $\mathcal{N}(0, 2\sigma\delta t)$. By definition $\xi^{(0)} = x$.

On the introduced uniform grid one approximates

$$\Lambda(T) = \int\limits_0^T V(\xi(\tau), T-\tau)d\tau \simeq \sum_{i=0}^n w_i V_i^{(n)}\delta t, \qquad V_i^{(n)} \equiv V(\xi(\tau_i), \tau_{n-i}),$$

where the set of weights $\{w_i\}_{i=0}^n$ is taken according to trapezoid rule or Simpson rule. Then

$$\exp\{-\Lambda(T)\} \simeq \prod_{i=0}^n \exp\{-w_i V_i^(n)\delta t$$

The Wiener measure transforms to n-dimensional measure

$$\mathcal{D}_\xi^{(n)} = \left(\frac{\lambda}{\pi}\right)^{\frac{n}{2}} \prod_{k=1}^{n} \exp\{-\lambda\xi_k^2\} d\xi_k, \qquad \lambda = \frac{1}{4\sigma\delta t},$$

and a numerical approximation of the exact solution can be written in the form

$$u^{(n)}(x,T) = \int\limits_{-\infty}^{\infty} \mathcal{D}_{xi} f(\xi^{(n)}) \prod_{i=0}^{n} e^{-w_i v_i^{(n)}\delta t}$$

The multidimensional integral can be represented in terms of n one-dimensional convolutions. Define

$$F_k^{(n)}(x) = \sqrt{\frac{\lambda}{\pi}} \int\limits_{-\infty}^{\infty} \Phi_{k+1}^{(n)}(x+\xi) e^{-\lambda\xi^2} d\xi, \qquad x \in \mathbb{R}, \quad k = n, n-1, \ldots, 1,$$

where

$$\Phi_{k+1}^{(n)}(x) = F_{k+1}^{(n)}(x) \exp\{-w_k V(x, \tau_{n-k})\delta t,$$

and

$$F^{(n)}(x)_{n+1} = f(x)$$

Then the numerical solution is given by formula

$$u^{(n)}(x,T) = F_1^{(n)}(x) e^{-w_0 V(x,T)\delta t}$$

Since $F_k^{(n)}(x)$ is represented as integral over all real values, it is replaced by an integral over a segment

$$F_k^{(n)}(x) \simeq \tilde{F}_k^{(n)}(x) = \sqrt{\frac{\lambda}{\pi}} \int\limits_{-a_x}^{a_x - h_x} \Phi_{k+1}^{(n)}(x+\xi) e^{-\lambda\xi^2} d\xi$$

The function $F_k^{(n)}(x)$ is computed on the uniform mesh

$$x_i^{(k)} = -ka_x + ih_x, \qquad 0 \le i \le kM, \qquad h_x = \frac{a_x}{N_x}, \qquad M = 2N_x$$

and the integration mesh is taken with the same step h_x

$$\xi_j = -a_x + jh_x, \qquad 0 \le j < M$$

Then

$$\tilde{F}_k^{(n)}(x_i^{(k)}) \simeq \sum_{j=0}^{M-1} \mu_j \Phi_{k+1}^{(n)}(x_{i+j}^{(k+1)} p(\lambda, \xi_j), \qquad p(\lambda, \xi) = \sqrt{\frac{\lambda}{\pi} e^{-\lambda\xi^2}}$$

$$\Phi_{k+1}^{(n)}(x_i^{(k+1)} = \tilde{F}_{k+1}^{(n)}(x_i^{(k+1)} \exp\{-w_k V(x_i^{(k+1)}, \tau_{n-k})\delta t$$

or in the matrix form

$$\tilde{F}_k^{(n)} = \Phi_{k+1}^{(n)} \circ \tilde{\mu}, \qquad \tilde{\mu}_j = \mu_j p(\lambda, \xi_j)$$

$$\Phi_{k+1}^{(n)} = \tilde{F}_{k+1}^{(n)} \exp\{-w_k V(x^{(k+1)}, \tau_{n-k})\delta t,$$

where $a \circ b$ denotes a convolution of vectors $a \in \mathbb{R}^m$ and $b \in \mathbb{R}^k$, i. e. a vector $c \in \mathbb{R}^{m+k-1}$, such that

$$c_i = \sum_{j=0}^{k-1} a_{i+j} b_j, \quad a_i = 0, \forall i : (i < 0) \bigvee (i \ge m)$$

The algorithm for computation of $u^{(n)}(x,T)$ is following.

1. Given T, choose a time step δt and the number of steps $n\frac{T}{\delta t}$.
2. Create 1D array τ of size n, $\tau_i = i\delta t$.
3. Create 1D array w of size $(n+1)$, corresponding to the set of weights in trapezoid or Simpson rule.
4. Choose a_x, size of coordinate grid $M = 2N_x$ and a coordinate step $h_x = \frac{a_x}{N_x}$.
5. Initialize 1D array $\tilde{F}_{n+1}^{(n)}$ of size $(n+1)M$, where $F_{n+1}^{(n)} = f(x)$ and $x_i = -(n+1)a_x + ih_x$, $0 \leq i < (n+1)M$.
6. For $k = n, n-1, \ldots, 1$ do
   (a) Create 1D array $x^{(k+1)}$ of size $(k+1)M$, $x_i^{(k+1)} = -(k+1)a_x + ih_x$, $0 \leq i < (k+1)M$
   (b) Create 1D array of size $(k+1)M$ $e^{-w_k V(x^{(k)}, \tau_{n-k})\delta t}$.
   (c) Create 1D array $\Phi_{k+1}^{(n)} = F_{k+1}^{(n)} \odot e^{-w_k V(x^{(k+1)}, \tau_{n-k})\delta t}$.
   (d) Create 1D array μ of size $M+1$, corresponding to the set of weights.
   (e) Create 1D array ξ of size $M+1$, where $\xi_j = -a_x + jh_x$, $0 \leq j < M+1$.
   (f) Create 1D array $\tilde{\mu} = \mu \odot p(\lambda, \xi)$.
   (g) Compute convolution $\tilde{F}_k^{(n)} = \Phi_{k+1}^{(n)} \circ \tilde{\mu}$. $\tilde{F}_k^{(n)}$ is 1D array of size kM.
7. Create 1D array $x^{(1)}$ of size M, $x_i^{(1)} = -a_x + ih_x$, $0 \leq i < M$
8. Create 1D array of size M $e^{-w_0 V(x^{(1)}, T)\delta t}$.
9. Compute the numerical solution $u^{(n)} = F_1^{(n)} \odot e^{-w_0 V(x^{(1)}, T)\delta t}$.

The proposed algorithm is implemented in class 'PDE_Solver'. An object of the class has following attributes:

- sigma
- V
- f
- a_x
- M
- h_x
- T
- n
- delta_t
- u – numerical solution

Names of all attributes correspond to notations used in this paper. Methods of an object of the class 'PDE_solver' are following:

- Set_Limits(float a) – sets a_x = a.
- Convolve(1D array a, 1D array b) – returns a 1D array $c = a \circ b$.
- Convolve_Low-rank(1D array a, 1D array b) – returns a 1D array $c = a \circ b$ using low-rank cross approximation.
- Solve() – computes a numerical solution u according to the proposed algorithm.