Radial Basis Functions generated Finite Differences (RBF-FD) for Solving High-Dimensional PDEs in Finance

S. Milovanović, L. von Sydow

Uppsala University
Department of Information Technology
Division of Scientific Computing

SIAM
Student Computational Finance Day
TU Delft, Netherlands, 2016-May-23
Overview

Introduction
  Option Pricing

Methods
  Radial Basis Functions generated Finite Difference methods

Results
  Numerical Experiments

Conclusion
  Further Research
Option Pricing

- Black-Scholes-Merton model in higher dimensions

\[
\begin{aligned}
\frac{dB(t)}{t} &= rB(t)dt, \\
\frac{dS_1(t)}{t} &= \mu_1 S_1(t)dt + \sigma_1 S_1(t)dW_1(t), \\
\frac{dS_2(t)}{t} &= \mu_2 S_2(t)dt + \sigma_2 S_2(t)dW_2(t), \\
& \vdots \\
\frac{dS_D(t)}{t} &= \mu_D S_D(t)dt + \sigma_D S_D(t)dW_D(t).
\end{aligned}
\] (1)

- Option price

\[u(S_1(t), \ldots, S_D(t), t) = e^{-r(T-t)}E_t^Q[g(S_1(T), \ldots, S_D(T))].\]

- The Black-Scholes-Merton equation

\[
\begin{aligned}
\frac{\partial u}{\partial t} + r \sum_{i}^{D} s_i \frac{\partial u}{\partial s_i} + \frac{1}{2} \sum_{i,j}^{D} \rho_{ij} \sigma_i \sigma_j s_i s_j \frac{\partial^2 u}{\partial u_{s_i s_j}} - ru &= 0, \\
u(s_1, s_2, \ldots, s_D, T) &= g(s_1, s_2, \ldots, s_D).
\end{aligned}
\] (2)
BENCHOP – The BENCHmarking project in Option Pricing*

- Monte Carlo methods converge slowly but are increasingly competitive in higher dimensions.
- Fourier-methods are fast, especially the COS-method.
- Finite difference methods are reasonably fast in lower dimensions but suffer from the curse of dimensionality.
- Methods based on approximations with Radial Basis Functions have the potential to be competitive in higher dimensions.

Radial Basis Functions Method

- Discretize space using $N$ nodes.
- Approximate solution

$$ u(s, t) \approx \sum_{k=1}^{N} \lambda_k(t) \phi(\varepsilon \|s - s_k\|), \quad k = 1, 2, \ldots, N, \quad (3) $$

where $\phi$ is a radial basis function and $\varepsilon$ is a shape parameter.
- The linear combination coefficients $\lambda_k$ are found by enforcing interpolation conditions.

Figure 1: Gaussian RBFs.
Local Radial Basis Functions Methods

- The global approximation leads to a dense linear system of equations which tends to be ill-conditioned when $\varepsilon$ is small
  ⇒ Local RBF approximations might be a better approach!
Local Radial Basis Functions Methods

- The global approximation leads to a dense linear system of equations which tends to be ill-conditioned when $\varepsilon$ is small
  $\Rightarrow$ Local RBF approximations might be a better approach!

- Ideas
  - Come up with a localization which leads to a sparse linear system of equations.
    - Radial Basis Functions Partition of Unity (RBF-PU)
    - Radial Basis Functions generated Finite Differences (RBF-FD)
  - Come up with basis transformation which will heal the ill-conditioning.
    - RBF-QR
    - RBF-GA
Try to exploit the best properties from both FD and RBF with minimal loss.

For each point $s_i$ in space, define its neighborhood of $M - 1$ points and observe it as a stencil.

Approximate the differential operator at every point

\[
[Lu(s)]_i \approx \sum_{k=1}^{M} w_k^{(i)} u_k^{(i)}.
\]
Radial Basis Functions generated Finite Differences

▶ Try to exploit the best properties from both FD and RBF with minimal loss.

▶ For each point \( s_i \) in space, define its neighborhood of \( M - 1 \) points and observe it as a stencil.

▶ Approximate the differential operator at every point

\[
[Lu(s)]_i \approx \sum_{k=1}^{M} w_k^{(i)} u_k^{(i)}. \quad (4)
\]

▶ Compute the RBF-FD weights and place them in matrix \( W \)

\[
\begin{bmatrix}
\phi(||s_1^{(i)} - s_1^{(i)}||) & \cdots & \phi(||s_1^{(i)} - s_M^{(i)}||) \\
\vdots & \ddots & \vdots \\
\phi(||s_M^{(i)} - s_1^{(i)}||) & \cdots & \phi(||s_M^{(i)} - s_M^{(i)}||)
\end{bmatrix}
\begin{bmatrix}
w_1 \\
\vdots \\
w_M
\end{bmatrix} =
\begin{bmatrix}
[L\phi(||s - s_1^{(i)}||)]_{s=s_i} \\
\vdots \\
[L\phi(||s - s_M^{(i)}||)]_{s=s_i}
\end{bmatrix}.
\]
Adding polynomial terms

\[
\begin{bmatrix}
\phi(\|s_1^{(i)} - s_1^{(i)}\|) & \ldots & \phi(\|s_1^{(i)} - s_1^{(i)}\|) \\
\vdots & \ddots & \vdots \\
\phi(\|s_M^{(i)} - s_1^{(i)}\|) & \ldots & \phi(\|s_M^{(i)} - s_M^{(i)}\|)
\end{bmatrix}
\begin{bmatrix}
w_1 \\
\vdots \\
w_M \\
w_{M+1}
\end{bmatrix} = 
\begin{bmatrix}
[L\phi(\|s - s_1^{(i)}\|)]_{s=s_i} \\
\vdots \\
[L\phi(\|s - s_M^{(i)}\|)]_{s=s_i}
\end{bmatrix}.
\]

![Graph showing Absolute Error for European Call Option](image)

**Figure 2:** Error in the solution.
Discretization

- Discretize the Black-Scholes-Merton equation operator in space using RBF-FD

\[
\begin{align*}
  u_t &= - \left[ r \sum_{i} D s_i u_{s_i} + \frac{1}{2} \sum_{i,j} \rho_{ij} \sigma_i \sigma_j s_i s_j u_{s_i s_j} - ru \right] \\
  &\approx W u.
\end{align*}
\]

- Integrate in time using the standard implicit schemes
  - BDF-1,
  - BDF-2.

\[
\Rightarrow
\]

\[
A u^{n+1} = b^n
\]
Solution of linear systems

Figure 3: Structure of $A$. 
Figure 4: The nearest-neighbor based stencils used for approximating the differential operator.
Figure 5: The absolute error distribution computed using 41 point in each dimension and a 5-point stencil on a full regular grid.
Figure 6: Square domain.
Figure 7: Triangular grid.
Grids

Figure 8: Adapted grid.
Figure 9: Adapted grid with some possible stencils.
European-style basket option with
\[ T = 1, \]
\[ K = 1, \]
\[ r = 0.03, \]
\[ \sigma_1 = \sigma_2 = 0.15, \]
\[ \rho = 0.5. \]

Payoff function
\[
\begin{align*}
\begin{array}{c}
u(s_1, s_2, T) = \left[ \frac{s_1(T) + s_2(T)}{2} - K \right]^+.
\end{array}
\end{align*}
\]
\[ (5) \]
Details

- **Boundary conditions**

  \[
  u(0, 0, t) = 0, \quad (6) \\
  u(s_1^*, s_2^*, t) = \frac{1}{2} [s_1^*(t) + s_2^*(t)] - e^{-r(T-t)} K. \quad (7)
  \]

- **RBF choice: Gaussian**

  \[
  \phi(r) = e^{-(\epsilon \cdot r)^2}. \quad (8)
  \]

- **Error measured on a subdomain:** \((s_1, s_2) \in \left[\frac{1}{3} K, \frac{5}{3} K \right].\)
Figure 10: The computed solution on a regular triangular grid.
Choice of $\varepsilon$, $n = 3$.

$\Delta u^{\text{max}}$ for $n=3$ with different values of $\varepsilon$ and $h$.

$\varepsilon(h) = 0.0020228(1/h) + 0.3451057$

Figure 11: Error as a function of $\varepsilon$. 
Choice of $\varepsilon$, $n = 7$.

Figure 12: Error as a function of $\varepsilon$. 

$\varepsilon(h)=0.05256(1/h)+7.60214$
Results

Figure 13: Spatial Convergence.

$\Delta u_{\text{max}}$
Results

Figure 14: Computational performance.
Conclusions

Proposed method:

- Works equally well with American options using operator splitting.
- Allows for local grid refinement which may deliver high accuracy in desired regions of the computational domain.
- Performs well due to sparsity and reduction in the computational domain allowed by the mesh-free discretization.
- Promises extendability to higher dimensional problems as well as for other models in the field.
Future Research

- Finalize study of shape parameter behavior.
- Developing adaptivity both in node placement and stencil size ($hp$-adaptivity).
- Optimizing stencils close to the boundaries.
- Smoothing of the terminal condition.
- Least squares instead of interpolation.
- Solving advanced financial models.
Future Research

- Higher dimensional problems.
Thank you for your attention.