A time-stepping deep gradient flow method for option pricing in (rough) diffusion models

Workshop on Computational and Mathematical Methods in Data Science

Jasper Rou
joint work with Antonis Papapantoleon

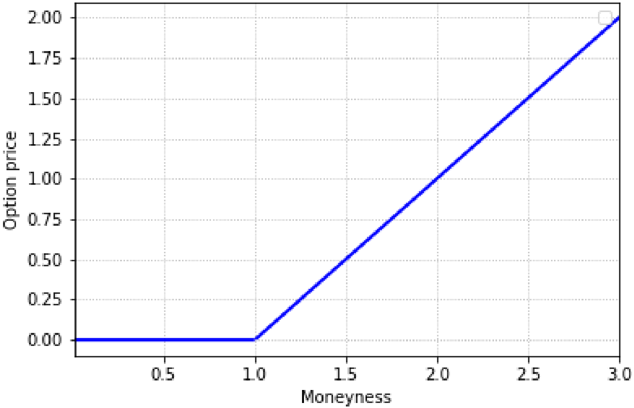
April 26, 2024

Options

A contract which gives the owner the right, but not the obligation, to buy a stock at a price K at a future time T

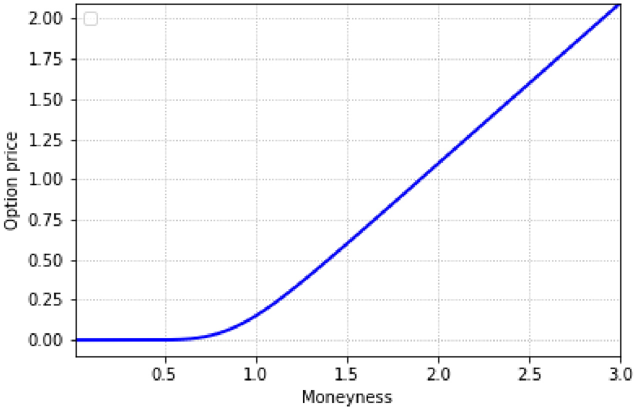
Pay-off

$$\Phi(S_T) = (S_T - K)^+$$



Pay-off

$$\Phi(S_T) = (S_T - K)^+$$

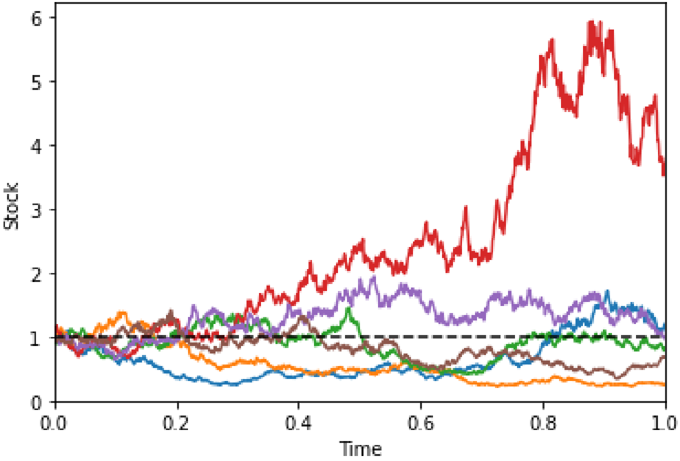


Stock price

$$dS_t = rS_t dt + \sigma S_t dW_t, \quad S_0 > 0,$$

Stock price

$$dS_t = rS_t dt + \sigma S_t dW_t, \quad S_0 > 0,$$



Pricing

Price of a derivative with pay-off $\Phi(S_T)$

$$u(t) = \mathbb{E}\left[e^{-r(T-t)}\Phi(S_T)|S_t\right]$$

Pricing

Price of a derivative with pay-off $\Phi(S_T)$

$$u(t) = \mathbb{E}\left[e^{-r(T-t)}\Phi(S_T)|S_t\right]$$

Feynman-Kac formula:

$$\frac{\partial u}{\partial t} + \sum_{i,j=0}^{n} a^{ij}\frac{\partial^2 u}{\partial x_i \partial x_j} - \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} - ru = 0,$$

$$u(T) = \Phi(S_T)$$

Pricing

Price of a derivative with pay-off $\Phi(S_T)$

$$u(t) = \mathbb{E}\left[e^{-r(T-t)}\Phi(S_T)|S_t\right]$$

Feynman-Kac formula:

$$\frac{\partial u}{\partial t} + \sum_{i,j=0}^{n} a^{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} - \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} - ru = 0,$$

$$u(T) = \Phi(S_T)$$

Can we solve this PDE using a neural network?

Deep Galerkin Method

$$\frac{\partial u}{\partial t} + \sum_{i,j=0}^{n} a^{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} - \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} - ru = 0,$$

$$u(T) = \Phi(S_T)$$

Deep Galerkin Method

$$\frac{\partial u}{\partial t} + \sum_{i,j=0}^{n} a^{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} - \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} - ru = 0,$$

$$u(T) = \Phi(S_T)$$

Minimize

$$\left\| \frac{\partial u}{\partial t} + \sum_{i,j=0}^{n} a^{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} - \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} - ru \right\|_{[0,T]\times\Omega}^2 + \| u(T) - \Phi(S_T) \|_{\Omega}^2 .$$

Deep Galerkin Method

$$\frac{\partial u}{\partial t} + \sum_{i,j=0}^{n} a^{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} - \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} - ru = 0,$$

$$u(T) = \Phi(S_T)$$

Minimize

$$\left\| \frac{\partial u}{\partial t} + \sum_{i,j=0}^{n} a^{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} - \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} - ru \right\|_{[0,T]\times\Omega}^2 + \| u(T) - \Phi(S_T) \|_\Omega^2 .$$

Issue: Taking second derivative makes training in high dimensions slow

Idea

Rewrite PDE as energy minimization problem

Idea

Rewrite PDE as energy minimization problem

- Only first order derivative
- No norm

Idea

Rewrite PDE as energy minimization problem

- Only first order derivative
- No norm

Split in symmetric and non-symmetric part

Splitting method

$$\frac{\partial u}{\partial t} = -\sum_{i,j=0}^{n} a^{ij} \frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} + ru$$

Splitting method

$$\frac{\partial u}{\partial t} = -\sum_{i,j=0}^{n} a^{ij}\frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} + ru$$

$$= -\sum_{i,j=0}^{n} \frac{\partial}{\partial x_j}\left(a^{ij}\frac{\partial u}{\partial x_i}\right) + \sum_{i,j=0}^{n}\frac{\partial a^{ij}}{\partial x_j}\frac{\partial u}{\partial x_i} + \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} + ru$$

Splitting method

$$\begin{aligned}\frac{\partial u}{\partial t} &= -\sum_{i,j=0}^{n} a^{ij}\frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} + ru \\ &= -\sum_{i,j=0}^{n} \frac{\partial}{\partial x_j}\left(a^{ij}\frac{\partial u}{\partial x_i}\right) + \sum_{i,j=0}^{n} \frac{\partial a^{ij}}{\partial x_j}\frac{\partial u}{\partial x_i} + \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} + ru \\ &= -\sum_{i,j=0}^{n} \frac{\partial}{\partial x_j}\left(a^{ij}\frac{\partial u}{\partial x_i}\right) + \sum_{i=0}^{n}\left(b^i + \sum_{j=0}^{n}\frac{\partial a^{ij}}{\partial x_j}\right)\frac{\partial u}{\partial x_i} + ru.\end{aligned}$$

Splitting method

$$\begin{aligned}
\frac{\partial u}{\partial t} &= -\sum_{i,j=0}^{n} a^{ij}\frac{\partial^2 u}{\partial x_i \partial x_j} + \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} + ru \\
&= -\sum_{i,j=0}^{n} \frac{\partial}{\partial x_j}\left(a^{ij}\frac{\partial u}{\partial x_i}\right) + \sum_{i,j=0}^{n} \frac{\partial a^{ij}}{\partial x_j}\frac{\partial u}{\partial x_i} + \sum_{i=0}^{n} b^i \frac{\partial u}{\partial x_i} + ru \\
&= -\sum_{i,j=0}^{n} \frac{\partial}{\partial x_j}\left(a^{ij}\frac{\partial u}{\partial x_i}\right) + \sum_{i=0}^{n}\left(b^i + \sum_{j=0}^{n}\frac{\partial a^{ij}}{\partial x_j}\right)\frac{\partial u}{\partial x_i} + ru. \\
&= -\nabla\cdot(A\nabla u) + ru + F(u), \\
F(u) &= \mathbf{b}\cdot\nabla u.
\end{aligned}$$

Example: Black-Scholes

$$dS_t = rS_t dt + \sigma S_t dW_t, \quad S_0 > 0,$$

Example: Black-Scholes

$$dS_t = rS_t dt + \sigma S_t dW_t, \quad S_0 > 0,$$

$$\frac{\partial u}{\partial \tau} - \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} - rS\frac{\partial u}{\partial S} + ru = 0$$

Example: Black-Scholes

$$dS_t = rS_t dt + \sigma S_t dW_t, \quad S_0 > 0,$$

$$\frac{\partial u}{\partial \tau} - \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} - rS\frac{\partial u}{\partial S} + ru = 0$$

Exact solution:

$$\begin{aligned} u(\tau, S) =& S\mathcal{N}\left(\frac{\log\left(\frac{S}{K}\right) + \left(r + \frac{\sigma^2}{2}\right)\tau}{\sigma\sqrt{\tau}}\right) \\ & - Ke^{-r\tau}\mathcal{N}\left(\frac{\log\left(\frac{S}{K}\right) + \left(r - \frac{\sigma^2}{2}\right)\tau}{\sigma\sqrt{\tau}}\right) \end{aligned}$$

Example: Black-Scholes

$$-\frac{\partial u}{\partial \tau} = -\frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} - rS\frac{\partial u}{\partial S} + ru$$

Example: Black-Scholes

$$-\frac{\partial u}{\partial \tau} = -\frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} - rS\frac{\partial u}{\partial S} + ru$$

Example: Black-Scholes

$$
\begin{aligned}
-\frac{\partial u}{\partial \tau} &= -\frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} - rS\frac{\partial u}{\partial S} + ru \\
&= -\frac{\partial}{\partial S}\left(\frac{1}{2}\sigma^2 S^2 \frac{\partial u}{\partial S}\right) + \sigma^2 S \frac{\partial u}{\partial S} - rS\frac{\partial u}{\partial S} + ru
\end{aligned}
$$

Example: Black-Scholes

$$
\begin{aligned}
-\frac{\partial u}{\partial \tau} &= -\frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} - rS\frac{\partial u}{\partial S} + ru \\
&= -\frac{\partial}{\partial S}\left(\frac{1}{2}\sigma^2 S^2 \frac{\partial u}{\partial S}\right) + \sigma^2 S \frac{\partial u}{\partial S} - rS\frac{\partial u}{\partial S} + ru \\
&= -\frac{\partial}{\partial S}\left(\frac{1}{2}\sigma^2 S^2 \frac{\partial u}{\partial S}\right) + \left(\sigma^2 S - rS\right)\frac{\partial u}{\partial S} + ru
\end{aligned}
$$

Example: Black-Scholes

$$
\begin{aligned}
-\frac{\partial u}{\partial \tau} &= -\frac{1}{2}\sigma^2 S^2 \frac{\partial^2 u}{\partial S^2} - rS\frac{\partial u}{\partial S} + ru \\
&= -\frac{\partial}{\partial S}\left(\frac{1}{2}\sigma^2 S^2 \frac{\partial u}{\partial S}\right) + \sigma^2 S \frac{\partial u}{\partial S} - rS\frac{\partial u}{\partial S} + ru \\
&= -\frac{\partial}{\partial S}\left(\frac{1}{2}\sigma^2 S^2 \frac{\partial u}{\partial S}\right) + \left(\sigma^2 S - rS\right)\frac{\partial u}{\partial S} + ru
\end{aligned}
$$

$$
A = \left[\frac{1}{2}\sigma^2 S^2\right], \quad \mathbf{b} = \left[\sigma^2 S - rS\right]
$$

Time Deep Gradient Flow Method

$$\begin{cases} u_\tau - \nabla \cdot (A\nabla u) + ru + F(u) = 0, & (\tau, \mathbf{x}) \in [0, T] \times \Omega, \\ u(0, \mathbf{x}) = \Phi(\mathbf{x}), & \mathbf{x} \in \Omega. \end{cases}$$

Time Deep Gradient Flow Method

$$\begin{cases} u_\tau - \nabla \cdot (A\nabla u) + ru + F(u) = 0, & (\tau, \mathbf{x}) \in [0, T] \times \Omega, \\ u(0, \mathbf{x}) = \Phi(\mathbf{x}), & \mathbf{x} \in \Omega. \end{cases}$$

- Divide $[0, T]$ in intervals $(\tau_{k-1}, \tau_k]$ with $h = \tau_k - \tau_{k-1}$

$$\frac{U^k - U^{k-1}}{h} - \nabla \cdot \left(A\nabla U^k\right) + rU^k + F\left(U^{k-1}\right) = 0$$

Time Deep **Gradient Flow** Method

$$\frac{U^k - U^{k-1}}{h} - \nabla \cdot \left(A \nabla U^k\right) + rU^k + F(U^{k-1}) = 0$$

Time Deep **Gradient Flow** Method

$$\frac{U^k - U^{k-1}}{h} - \nabla \cdot \left(A \nabla U^k\right) + rU^k + F(U^{k-1}) = 0$$

$$0 = \int_\Omega \left(\left(U^k - U^{k-1}\right) + h\left(-\nabla \cdot \left(A \nabla U^k\right) + rU^k + F\left(U^{k-1}\right)\right)\right) v d\mathbf{x}$$

Time Deep **Gradient Flow** Method

$$\frac{U^k - U^{k-1}}{h} - \nabla \cdot \left(A \nabla U^k\right) + rU^k + F(U^{k-1}) = 0$$

$$\begin{aligned} 0 &= \int_\Omega \left(\left(U^k - U^{k-1}\right) + h\left(-\nabla \cdot \left(A\nabla U^k\right) + rU^k + F\left(U^{k-1}\right)\right)\right) v d\mathbf{x} \\ &= i'(0) \end{aligned}$$

$$i(\tau) = I^k(U^k + \tau v)$$

Time Deep **Gradient Flow** Method

$$\frac{U^k - U^{k-1}}{h} - \nabla \cdot \left(A \nabla U^k\right) + rU^k + F(U^{k-1}) = 0$$

$$\begin{aligned} 0 &= \int_\Omega \left(\left(U^k - U^{k-1}\right) + h\left(-\nabla \cdot \left(A\nabla U^k\right) + rU^k + F\left(U^{k-1}\right)\right)\right) v d\mathbf{x} \\ &= i'(0) \end{aligned}$$

$$i(\tau) = I^k(U^k + \tau v)$$

$$\begin{aligned} I^k(u) &= \frac{1}{2}\left\|u - U^{k-1}\right\|^2 + h\int_\Omega \frac{1}{2}\left((\nabla u)^T A \nabla u + ru^2\right) + F\left(U^{k-1}\right) u dx \\ U^k &= \underset{u \in H^1(\Omega)}{\arg\min}\, I^k(u) \end{aligned}$$

Time **Deep** Gradient Flow Method

$$I^k(u) = \frac{1}{2}\left\|u - U^{k-1}\right\|^2 + h\int_\Omega \frac{1}{2}\left((\nabla u)^T A\nabla u + ru^2\right) + F\left(U^{k-1}\right) u dx$$

$$U^k = \operatorname*{arg\,min}_{u\in H^1(\Omega)} I^k(u)$$

Time **Deep** Gradient Flow Method

$$I^k(u) = \frac{1}{2}\left\|u - U^{k-1}\right\|^2 + h\int_\Omega \frac{1}{2}\left((\nabla u)^T A\nabla u + ru^2\right) + F\left(U^{k-1}\right) u dx$$

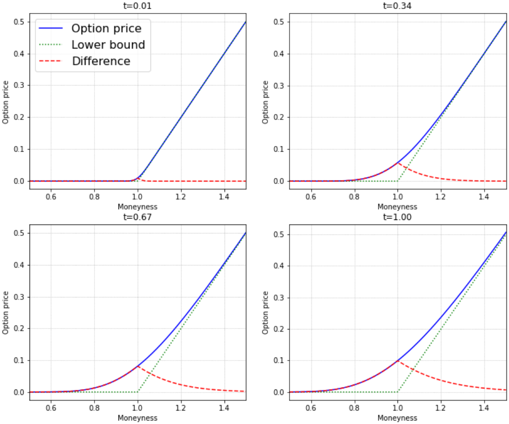
$$U^k = \arg\min_{u\in H^1(\Omega)} I^k(u)$$

$$f^k(\theta) = \arg\min_{u\in \mathcal{C}(\theta)} I^k(u)$$

$$\mathcal{C}(\theta) = \text{space of neural networks with parameters } \theta$$

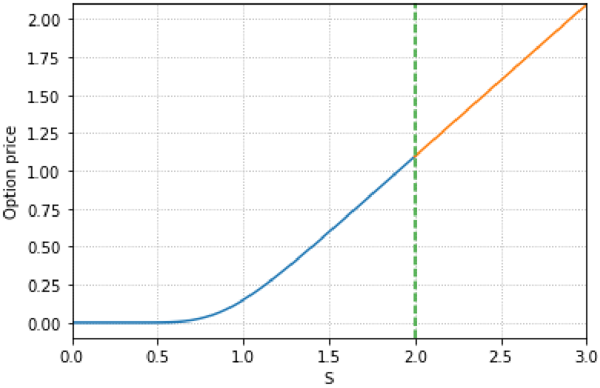
Base

No-arbitrage bound: $u(t, S) \geq S - Ke^{-rt}$



Linearization

$$u(x_p + y; \theta) = u(x_p; \theta) + y, \quad y > 0.$$



Architecture

$$
\begin{aligned}
S^1 &= \sigma_1 \left(W^1 \mathbf{x} + b^1 \right), \\
Z^l &= \sigma_1 \left(U^{z,l} \mathbf{x} + W^{z,l} S^l + b^{z,l} \right), && l = 1, ..., L, \\
G^l &= \sigma_1 \left(U^{g,l} \mathbf{x} + W^{g,l} S^1 + b^{g,l} \right), && l = 1, ..., L, \\
R^l &= \sigma_1 \left(U^{r,l} \mathbf{x} + W^{r,l} S^l + b^{r,l} \right), && l = 1, ..., L, \\
H^l &= \sigma_1 \left(U^{h,l} \mathbf{x} + W^{h,l} \left(S^l \odot R^l \right) + b^{h,l} \right), && l = 1, ..., L, \\
S^{l+1} &= \left(1 - G^l \right) \odot H^l + Z^l \odot S^l, && l = 1, ..., L, \\
f(\theta) &= \text{base} + \sigma_2 \left(W S^{L+1} + b \right), && \sigma_2 > 0.
\end{aligned}
$$

Algorithm

1: Initialize $f(\theta^0; \mathbf{x}) = \Phi(S)$.

Algorithm

Initialize $f(\theta^0; \mathbf{x}) = \Phi(S)$.
for each time step $k = 1, ..., N_t$ **do**
    Initialize $\theta_0^k = \theta^{k-1}$.

Algorithm

Initialize $f(\theta^0; \mathbf{x}) = \Phi(S)$.
for each time step $k = 1, ..., N_t$ **do**
    Initialize $\theta_0^k = \theta^{k-1}$.
    for each sampling stage n **do**
        Generate random points $\mathbf{x}^i$ for training.

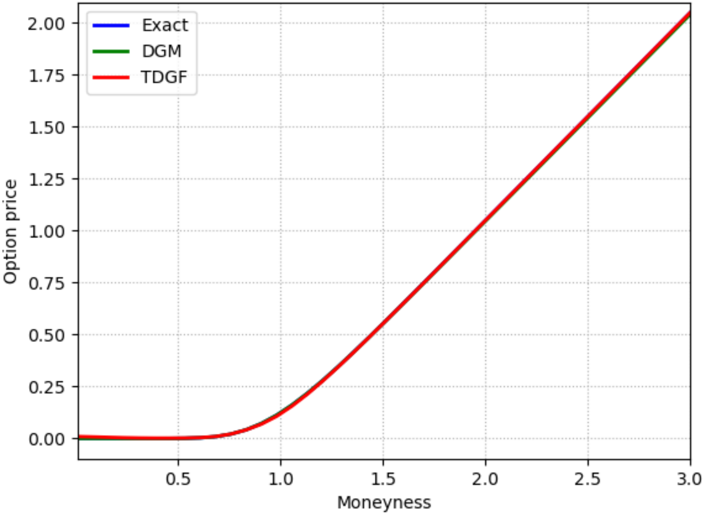
Algorithm

Initialize $f(\theta^0; \mathbf{x}) = \Phi(S)$.
for each time step $k = 1, ..., N_t$ **do**
    Initialize $\theta_0^k = \theta^{k-1}$.
    for each sampling stage n **do**
        Generate random points $\mathbf{x}^i$ for training.
        Calculate the cost functional $I^k(f(\theta_n^k; \mathbf{x}^i))$.

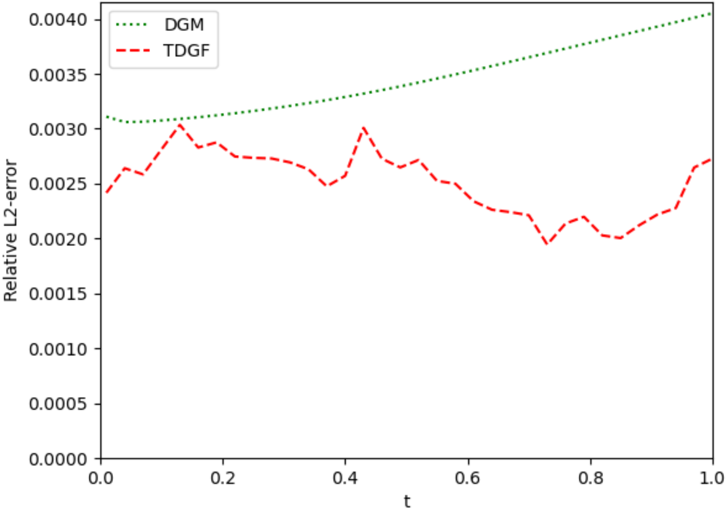
Algorithm

Initialize $f(\theta^0; \mathbf{x}) = \Phi(S)$.
for each time step $k = 1, ..., N_t$ **do**
    Initialize $\theta_0^k = \theta^{k-1}$.
    for each sampling stage n **do**
        Generate random points $\mathbf{x}^i$ for training.
        Calculate the cost functional $I^k(f(\theta_n^k; \mathbf{x}^i))$.
        Take a descent step $\theta_{n+1}^k = \theta_n^k - \alpha_n \nabla_\theta I^k(f(\theta_n^k; \mathbf{x}^i))$.
    end for
end for

Black-Scholes



Black-Scholes



Lifted Heston

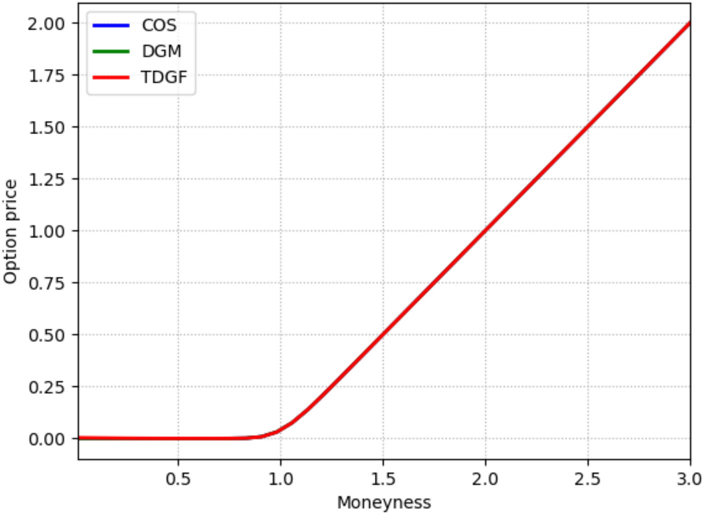
$$
\begin{aligned}
dS_t &= rS_t dt + \sqrt{V_t^n} S_t dW_t, & S_0 > 0, \\
V_t^n &= g^n(t) + \sum_{i=1}^{n} c_i^n V_t^{n,i}, & \\
dV_t^{n,i} &= -\left(\gamma_i^n V_t^{n,i} + \lambda V_t^n\right) dt + \eta \sqrt{V_t^n} dB_t, & V_0^{n,i} = 0, \\
g^n(t) &= V_0 + \lambda\theta \sum_{i=1}^{n} c_i^n \int_0^t e^{-\gamma_i^n (t-s)} ds. &
\end{aligned}
$$

Lifted Heston

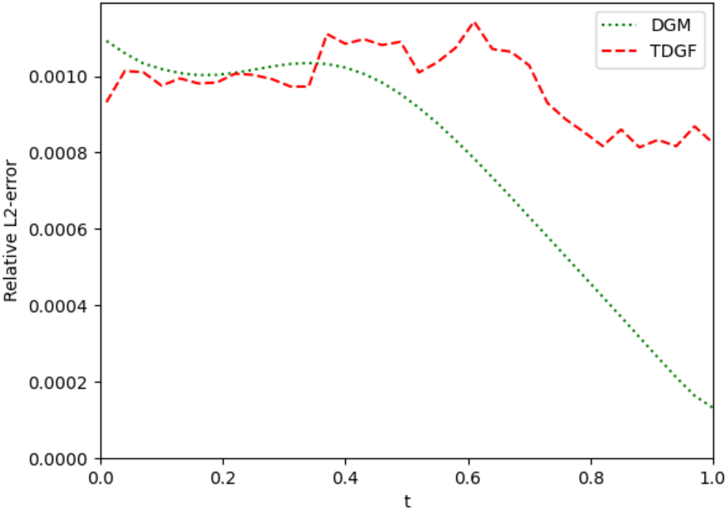
$$
\begin{aligned}
dS_t &= rS_t dt + \sqrt{V_t^n} S_t dW_t, && S_0 > 0, \\
V_t^n &= g^n(t) + \sum_{i=1}^{n} c_i^n V_t^{n,i}, \\
dV_t^{n,i} &= -\left(\gamma_i^n V_t^{n,i} + \lambda V_t^n\right) dt + \eta \sqrt{V_t^n} dB_t, && V_0^{n,i} = 0, \\
g^n(t) &= V_0 + \lambda\theta \sum_{i=1}^{n} c_i^n \int_0^t e^{-\gamma_i^n (t-s)} ds.
\end{aligned}
$$

No exact solution

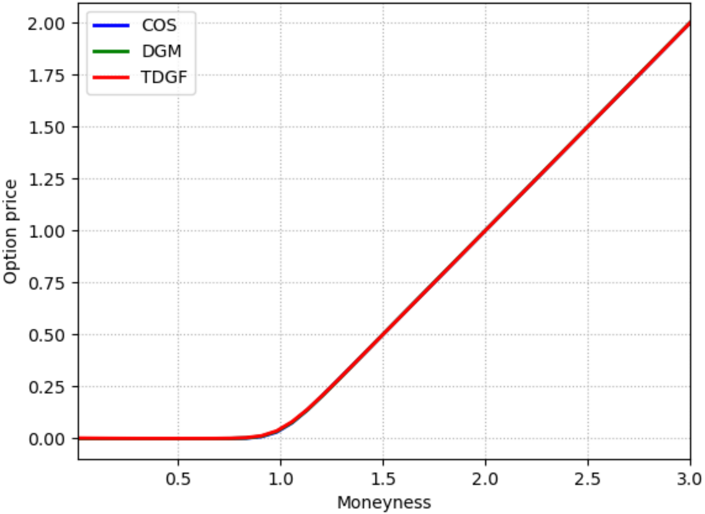
Lifted Heston, $n = 1$



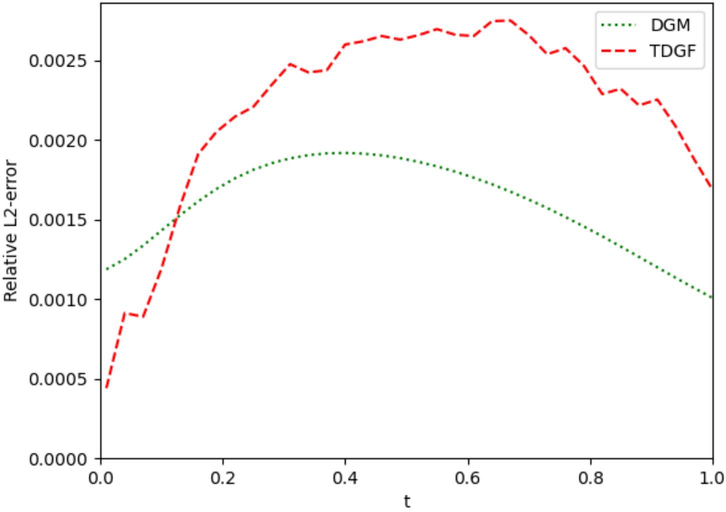
Lifted Heston, $n = 1$



Lifted Heston, $n = 20$



Lifted Heston, $n = 20$



Running times

Model	Black-Scholes	Heston	LH, n=1	LH, n=20
DGM	7.5×10^3	12.5×10^3	13.3×10^3	56.1×10^3
TDGF	4.1×10^3	6.0×10^3	6.4×10^3	7.6×10^3

Table: Training time

Running times

Model	Black-Scholes	Heston	LH, n=1	LH, n=20
DGM	7.5×10^3	12.5×10^3	13.3×10^3	56.1×10^3
TDGF	4.1×10^3	6.0×10^3	6.4×10^3	7.6×10^3

Table: Training time

Model	Black-Scholes	LH, n=1	LH, n=20
Exact/COS	0.00025	8.9	10.4
DGM	0.0043	0.0034	0.0053
TDGF	0.024	0.020	0.025

Table: Computing time

Conclusion

	Accurate	Fast
Simple model	×	✓
Complicated model	✓	×

Conclusion

	Accurate	Fast
Simple model	×	✓
Complicated model	✓	×
Complicated model with neural networks	✓	✓

A time-stepping deep gradient flow method for option pricing in (rough) diffusion models

Workshop on Computational and Mathematical Methods in Data Science

Jasper Rou

April 26, 2024

j.g.rou@tudelft.nl www.jasperrou.nl