Wavelet based PDE Valuation of Complex Derivatives

M A H Dempster

Centre for Financial Research
Judge Institute of Management, University of Cambridge

&

Cambridge Systems Associates Limited

Co-worker: Benjamin Carton de Wiart

Research partially sponsored by Citigroup (Schroder Salomon Smith Barney) and ESRC UK
Outline

1. Introduction
2. Wavelet Transforms
3. Wavelets and PDE’s
4. European Option Valuation
5. Cross-currency Swap Valuation
6. Thresholding
7. Conclusions and Further Work
1. Introduction

• What are wavelets?

• Wavelets are nonlinear functions which can be scaled and translated to form a basis for the Hilbert space $L^2(\mathbb{R})$ of square integrable functions.

• Wavelets generalize the trigonometric functions $e^{ist}$ ($s \in \mathbb{R}$) which generate the classical Fourier basis for $L^2$.

• Hence there are wavelet -- and fast wavelet transforms -- which generalize the time to frequency map of the Fourier transform to pick up both the space and time behavior of a function.
Applications of wavelets

• Digital image compression

• Signal processing

• De-noising of signals – filtering

• Numerical analysis
Wavelets and PDEs

A wavelet based approach to the solution of PDEs has been studied by

• Beylkin (1993)
• Monasse and Perrier (1998)
• Prosser and Cant (1999, 2000)
• Dahmen et al (1999)
• Cohen et al (2000)
• Dempster & Eswaran (2001)
Numerical methods to solve PDEs

- Finite-difference (finite volume)
- Weighted residuals
  - Finite-elements
  - Spectral methods
 Finite difference

• Define discrete finite grid

• Replace differential operator by difference operators

\[
\frac{\partial C}{\partial S}(S, t) = \frac{C(S + \delta S, t) - C(S - \delta S, t)}{2\delta S}
\]
Weighted residual

- Solution approximated by a linear combination of a set of linearly independent functions
  - **Finite elements**: trial functions are piecewise continuous and non-vanishing on certain elements of the domain
  - **Spectral methods**: basis functions infinitely differentiable and non-vanishing on the whole domain
Advantages of using wavelets to solve derivative valuation PDEs

- Wavelet PDE methods combine the advantages of both spectral (Fourier) and finite-difference methods and allow both space and time dependent coefficients.
- Large classes of operators and functions are sparse or sparse to high accuracy when transformed into the wavelet domain.
- Wavelets are suitable for problems with the multiple spatial scales common in financial problems.
- Wavelets can be used for nonlinear terms.
- Wavelets accurately represent the PDE solution in regions of sharp transitions.
2. Wavelet Transforms

Scaling Functions and Wavelets

- The scaling function $\phi$ is the solution of a dilation equation

$$\phi(x) = \sqrt{2} \sum_{k=0}^{\infty} h_k \phi(2x - k)$$

where $\phi$ is normalised i.e. $\int_{-\infty}^{\infty} \phi(x) dx = 1$

- The wavelet $\psi$ is defined in terms of the scaling function as

$$\psi(x) = \sqrt{2} \sum_{k=0}^{\infty} g_k \phi(2x - k)$$
• The basis functions $\phi_{j,k}$ and $\psi_{j,k}$ are generated from $\phi$ and $\psi$ through scaling and translation as

$$\phi_{j,k}(x) := 2^{-j/2} \phi(2^{-j} x - k) = 2^{-j/2} \phi\left(\frac{x - 2^j k}{2^j}\right)$$

$$\psi_{j,k}(x) := 2^{-j/2} \psi(2^{-j} x - k) = 2^{-j/2} \psi\left(\frac{x - 2^j k}{2^j}\right)$$
• The coefficients $h$ and $g$ are chosen so that dilations and translations of the wavelets $\psi_{j,k}$ form an orthonormal basis of $L^2(\mathbb{R})$ i.e.

$$\langle \psi_{j,k}(x), \psi_{j',k'}(x) \rangle = \delta_{j,j'} \delta_{k,k'}$$

where $\delta$ denotes the Kronecker delta
Multiresolution analysis

- $L^2(\mathbb{R})$ can be approximated as a nested hierarchy of finite dimensional subspaces

$$\{0\} \subset \ldots \subset V_{j-1} \subset V_j \subset V_{j+1} \subset \ldots \subset L^2(\mathbb{R})$$

- The information “added” from $V_{j-1}$ to $V_j$ is taken from a detail space $W_j$

$$V_j = V_{j-1} \oplus W_{j-1}$$
• Consider spans of $\phi_{j,k}$ and $\psi_{j,k}$ over the location parameter $k$ with the scale parameter $j$ fixed

\[ V_j = \text{span}_{k \in \mathbb{Z}^+} \phi_{j,k}(x) \]

\[ W_j = \text{span}_{k \in \mathbb{Z}^+} \psi_{j,k}(x) \]

• Define two nested hierarchies of finite dimensional subspaces

\[ W_{J-1} \subset W_J \subset W_{J+1} \quad \text{and} \quad V_{J-1} \subset V_J \subset V_{J+1} \]
In practice

- Since computers are finite object manipulators we must define a finite direct sum of $J_2 - J_1$ finite dimensional vector spaces

\[ V_{J_2} = V_{J_1} \oplus W_{J_1+1} \oplus W_{J_1+2} \oplus W_{J_1+3} \oplus \ldots \oplus W_{J_2-1} \]
• $J_1$ and $J_2$ are chosen according to the required grid size in numerical computation

• The orthogonal wavelet approximation to a continuous function $f$ is given by

$$f(x) \approx \sum_{k} s_{J_1,k} \phi_{J_1,k}(x) + \sum_{j=J_1}^{J_2-1} \sum_{k} d_{j,k} \psi_{j,k}(x)$$

where $k$ ranges from 1 to the number of coefficients in the specified component
Biorthogonal wavelets
Cohen, Daubechies & Feauveau (1992)

• Four basic function types -- two primals and two duals

• The biorthogonal wavelet approximation is expressed in terms of the dual wavelet functions

\[ f(x) \approx \sum_{k} s_{j_1,k} \phi_{j_1,k}(x) + \sum_{j=J_1}^{J_2-1} \sum_{k} d_{j,k} \psi_{j,k}(x) \]

• With \( s_{j,k} = \langle f(x), \tilde{\phi}_{j,k}(x) \rangle \) and \( d_{j,k} = \langle f(x), \tilde{\psi}_{j,k}(x) \rangle \)
Biorthogonal B-spline wavelets
• Biorthogonal wavelets are not orthogonal but satisfy the biorthogonal relationships

\[ \langle \phi_{j,k}(x), \tilde{\phi}_{j',k'}(x) \rangle = \delta_{j,j'}\delta_{k,k'} \]

\[ \langle \phi_{j,k}(x), \tilde{\psi}_{j',k}(x) \rangle = 0 \]

\[ \langle \psi_{j,k}(x), \tilde{\psi}_{j',k'}(x) \rangle = \delta_{j,j'}\delta_{k,k'} \]

\[ \langle \psi_{j,k}(x), \tilde{\phi}_{j,k'}(x) \rangle = 0 \]
Biorthogonal approach

• Biorthogonal wavelets are symmetrical and generalize the orthogonal wavelet approximation

• Basis functions for biorthogonal wavelet spaces are generated from the primal scaling function $\phi$ and the dual scaling function $\tilde{\phi}$

• Biorthogonal systems are thus derived from a paired hierarchy of approximation subspaces

$$V_{J-1} \subset V_J \subset V_{J+1}$$
$$\tilde{V}_{J-1} \subset \tilde{V}_J \subset \tilde{V}_{J+1}$$
• Wavelet innovation spaces \( W_J \) and \( \tilde{W}_J \) are defined by

\[
V_{J+1} = V_J \oplus W_J \\
\tilde{V}_{J+1} = \tilde{V}_J \oplus \tilde{W}_J
\]

with \( \tilde{V}_J \perp W_J \) \( V_J \perp \tilde{W}_J \)

• Basis functions for the wavelet spaces are generated by the primal and dual wavelets \( \psi \) and \( \tilde{\psi} \)

• Projections of a function \( f \) onto finite dimensional scaling function space \( V_J \)
or wavelet space \( W_J \) are given by

\[
P_{V_J} f(x) = \sum_{k \in \mathbb{Z}} \left< f(u), \tilde{\phi}_{J,k}(x) \right> \phi_{J,k}(x)
\]

\[
P_{W_J} f(x) = \sum_{k \in \mathbb{Z}} \left< f(u), \tilde{\psi}_{J,k}(x) \right> \psi_{J,k}(x)
\]
Biorthogonal interpolating wavelet transforms
Deslauriers & Dubuc (1989) and Donoho (1992)

- The biorthogonal interpolating wavelet transform has basis functions of the form

$$\phi_{j,k}(x) = \phi (2^j x - k)$$

$$\psi_{j,k}(x) = \phi (2^{j+1} x - 2k - 1)$$

$$\tilde{\phi}_{j,k}(x) = \delta (x - x_{j,k})$$

$$\tilde{\psi}_{j,k}(x) = ?$$ (but can be derived)

where $\delta$ is the Dirac delta function and $x_{j,k} = k / 2^j$ is a grid point in the spatial dimension
Properties of the interpolating scaling function of order $N$

- Compact support: $\phi = 0$ outside $[-N+1, N-1]$

- Interpolation: $\phi(0) = 1$ and $\phi(k) = 0$ for $k \neq 0$

- Smooth: $\phi_{j,k} \in C^\alpha$ with $\alpha = \alpha(N)$ (with $\alpha(6) < 2.830$)

- Refinability: $\phi(x) = \sqrt{2} \sum_{k=0}^{2N+1} h_k \phi(2x - k)$

- Polynomial reproduction: polynomial of degree $N-1$ can be expressed as a linear combination of scaling functions
Mother interpolating scaling function ($N=4$)
What do we need?

- Value of mother scaling function at half integer nodes for the wavelet transform

- Value of derivative of mother scaling function at integer nodes
Fast interpolating wavelet transform algorithm

- The projection of a function $f$ onto a finite dimensional scaling function space $V_J$ is given by

$$P_{V_J} f(x) = \sum_{k \in \mathbb{Z}} \langle f(u), \tilde{\phi}_{J,k}(x) \rangle \phi_{J,k}(x)$$

$$= \sum_{k \in \mathbb{Z}} f\left( \frac{k}{2^J} \right) \phi_{J,k}(x)$$

$$:= \sum_{k \in \mathbb{Z}} s_{J,k} \phi_{J,k}(x)$$

Remember

$$\tilde{\phi}_{j,k}(x) = \delta \left( x - k / 2^J \right)$$
Fast interpolating wavelet transform algorithm structure

\[
\begin{align*}
\left\{ s_{J,0}, s_{J,1}, s_{J,2} \cdots s_{J,2^{J-1}-1} \right\}^T \\
\downarrow \\
\left\{ d_{J-1,0}, \cdots, d_{J-1,2^{J-1}-1} \mid s_{J-1,0}, s_{J-1,1}, s_{J-1,2} \cdots s_{J-1,2^{J-1}-1} \right\}^T \\
\downarrow \\
\left\{ d_{J-1,0}, \cdots, d_{J-1,2^{J-1}-1} \mid d_{J-2,0}, \cdots, d_{J-2,2^{J-2}-1} \mid s_{J-2,0}, \cdots s_{J-2,2^{J-2}-1} \right\}^T \\
\downarrow \\
\left\{ d_{J-1,0}, \cdots, d_{J-1,2^{J-1}-1} \mid \cdots d_{J-P,0}, \cdots, d_{J-P,2^{j-P-1}-1} \mid s_{J-P,0} \cdots s_{J-P,2^{j-P-1}-1} \right\}^T \\
\downarrow \\
W_{J-1} \oplus W_{J-2} \oplus W_{J-3} \oplus \cdots V_{J-P}
\end{align*}
\]

Example: J := 6  P := 3
Computation of wavelet coefficients

\[ S_{j,m} = S_{j+1,2m} \quad \text{(remember} \quad s_{j,m} := f(m/2^J)) \]

\[ d_{j,m} = s_{j+1,2m+1} - \sum_{n \in \mathbb{Z}} s_{j,n} \phi(m-n+1/2) \]

- All we need to know is \( \phi \)'s value at half integer nodes
Interpolating biorthogonal wavelet transform complexity

• The number of operations required for the transform algorithm for $P$ resolution levels is $2M \sum_{i=J-P}^{J} 2^i = 2^J P M (2^{P+1} - 1)$ as $2M$ filter coefficients define the primal scaling function which spans the space of polynomial of degree less than $M-1$

• Calculation of the wavelet coefficients $d_{j,k}^{f}$ for a given resolution level $j$ can be accomplished in $2(M-1)+1$ floating point operations and the sub-sampling process for the scaling function coefficients requires a further $2^j$ operations for a total of $2^{j+1}M$ operations required per resolution level $j$

• For fixed $J$ and $P$ the fast interpolating wavelet transform algorithm is $O(M)$ for exact $(M-1)^{st}$ order polynomial approximation

• Since the finest resolution in a spatial grid of $N$ points is $J=\log_2 N$ for fixed $M$ and $P$ the complexity of the transform is $O(N)$

• Dempster & Eswaran (2001)
3. Wavelets and PDEs

Decomposition of differential operators

- We derive the projection of our function

\[
\frac{d}{dx} P_{V_j} f(x) = \sum_{k \in \mathbb{Z}} \left\langle f(u), \tilde{\phi}_{J,k}(x) \right\rangle \frac{d}{dx} \phi_{J,k}(x)
\]

- But \( \frac{d}{dx} \phi_{J,k}(x) \) might not be (and often is not) in \( V_j \) so we project again and define the wavelet derivative \( \partial_J \) as

\[
\partial_J f(x) := P_{V_j} \frac{d}{dx} P_{V_j} f(x)
\]
Repeating gives the decomposition of the wavelet derivative as

\[ \partial J_2 f(x) := \left( P_{V_{J_1}} + \sum_{i=J_1}^{J_2-1} P_{W_i} \right) \frac{d}{dx} \left( P_{V_{J_1}} + \sum_{i=J_1}^{J_2-1} P_{W_i} \right) f(x) \]
• If we know $d\phi/dx$ and $d\psi/dx$ we can build a differential operator in the wavelet space
• All we need is the value of the derivatives of $\phi$ at integer nodes
• Coefficient values can be computed numerically or analytically  Beylkin (1992)
• We make higher order operators by repetitive application of the first order operator  Prosser & Cant (1999)
Non-constant coefficients

- Wavelet-based PDE methods have mainly dealt with constant coefficients but financial PDEs frequently have non-constant coefficients.
- Canonical transformation of the variables is usually undesirable or impractical.
- Our solution methodology should be able to handle a wide range of PDEs with non-constant coefficients.
Combined operator approach
Dempster & Eswaran (2001)

• We construct a new differential operator which combines non-constant coefficients and derivative terms

• Start with the usual projection $P_J$ of $f$ onto a wavelet (or scaling) space $W_J$

\[ P_J f(x) = \sum_k s_{j,k}^f \phi_{j,k}(x) \]

• Multiplying by a nonlinear function gives

\[ g(x)[P_J f(x)] = \sum_k s_{j,k}^f [g(x)\phi_{j,k}(x)] \]
• Since $g(x)\phi_{J,k}(x)$ is not a basis for $V_J$, project again to obtain

$$P_J(g(x)[P_J f(x)]) = \sum_{\alpha} \sum_{k} S^f_{J,k} < g(x)\phi_{J,k}(x), \tilde{\phi}_{J,\alpha} > \phi_{J,\alpha}(x)$$

• To determine the inner product in the case of a differential operator recall the interpolating nature of the dual scaling function to yield

$$< g(x)\phi_{J,k}(x), \tilde{\phi}_{J,\alpha} > = 2^J g(2^{-J} \alpha)\phi'(\alpha - \kappa)$$

• We apply the standard decomposition to the above expression to get the combined differential operator
Wavelet method of lines

• A traditional finite difference scheme replaces partial derivatives with algebraic approximations at grid points and solves the system of algebraic equations to obtain a numerical solution of the PDE

• The method of lines transforms the PDE into a vector ODE by replacing the spatial partial derivatives with their wavelet approximations but retaining the time derivatives

• The vector ODE is solved in time using a stiff ODE solver

• A method based on the backward differentiation formula (LSODE) from Lawrence Livermore Laboratories and an Euler method have been implemented in C/Fortran 90 on an IBM RS6000/590 and an Athlon 650

• The complexity of the method is $O(N\tau)$ for time discretization $\tau$
PDEs in $d$ space dimensions

- The entire multiresolution wavelet machinery presented so far can be extended to several space dimensions $d$ by taking straightforward Cartesian products of appropriate approximation and scaling subspaces -- i.e. tensor products of appropriate wavelet bases -- to result in a fast wavelet transform of $O(N)$ for $N := n^d$ for spatial discretization $n$

- The imposition of boundary conditions for nonlinearly bounded domains is nontrivial but these are fortunately rare in PDE derivative valuation problems which are usually Cauchy problems on a strip
4. European Option Valuation

- The Black Scholes PDE is

\[ \frac{\partial C}{\partial t} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 C}{\partial S^2} + rS \frac{\partial C}{\partial S} - rC = 0 \]

where \( S \) is stock price, \( \sigma \) is volatility and \( r \) is the risk free rate of interest.

- Transform to the heat diffusion equation

\[ \frac{\partial u}{\partial \tau} = \frac{\partial^2 u}{\partial x^2} \quad \text{for} \quad -\infty < x < \infty, \tau > 0 \]

using \( S = X e^x \) and \( t = T - \frac{2\tau}{\sigma^2} \).

- Then \( C(S, T) := \max(S-X, 0) \) becomes for \( k := \frac{2r}{\sigma^2} \)

\[ C(x, \tau) = e^{-\frac{1}{2}(k-1)x - \frac{1}{2}(k+1)^2 \tau} Xu(x, \tau) \]
European vanilla call

- For a vanilla European call option the boundary conditions are
  \[ C(0, t) = 0, \ C(S, t) \sim S \text{ as } S \to \infty \]

- The boundary conditions for the transformed PDE are
  \[ u(x, \tau) = 0 \text{ as } x \to -\infty \]
  \[ u(x, \tau) \sim e^{\frac{1}{2}(k+1)x + \frac{1}{4}(k+1)^2 \tau} \text{ as } x \to \infty \]
  \[ u(x, 0) = \max \left\{ e^{\frac{1}{2}(k+1)x} - e^{\frac{1}{2}(k-1)x}, 0 \right\} \]

- The heat equation solution can be transformed back to original variables as
  \[ C(S, t) = X^2 \left( S^{-2} e^{\frac{1}{2}(k+1)x + \frac{1}{4}(k+1)^2 \sigma^2 (T-t)} e^{\frac{1}{2}(1-k)x} \right) u\left( \log(S/X), \frac{1}{2} \sigma^2 (T-t) \right) \]
  where \( k = 2r/\sigma^2 \)
## Vanilla European Call

- S =10, strike =10, r = 5%, volatility = 20%, maturity=1 Year
- Exact value: 1.04505

### Wavelet method of lines

<table>
<thead>
<tr>
<th>Space Steps</th>
<th>Time Steps</th>
<th>Value</th>
<th>Solution time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>60</td>
<td>1.03515</td>
<td>.05</td>
</tr>
<tr>
<td>128</td>
<td>100</td>
<td>1.04220</td>
<td>.10</td>
</tr>
<tr>
<td>256</td>
<td>200</td>
<td>1.04502</td>
<td>.13</td>
</tr>
<tr>
<td><strong>512</strong></td>
<td><strong>200</strong></td>
<td><strong>1.04505</strong></td>
<td><strong>.30</strong></td>
</tr>
<tr>
<td>1024</td>
<td>200</td>
<td>1.04505</td>
<td>.90</td>
</tr>
</tbody>
</table>

### Crank-Nicolson Finite Difference Method

<table>
<thead>
<tr>
<th>Space Steps</th>
<th>Time Steps</th>
<th>Value</th>
<th>Solution time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>60</td>
<td>1.03184</td>
<td>.02</td>
</tr>
<tr>
<td>128</td>
<td>100</td>
<td>1.04184</td>
<td>.04</td>
</tr>
<tr>
<td>256</td>
<td>200</td>
<td>1.04426</td>
<td>.09</td>
</tr>
<tr>
<td>512</td>
<td>200</td>
<td>1.04486</td>
<td>.16</td>
</tr>
<tr>
<td>1024</td>
<td>200</td>
<td>1.04501</td>
<td>.30</td>
</tr>
<tr>
<td><strong>2000</strong></td>
<td><strong>200</strong></td>
<td><strong>1.04505</strong></td>
<td><strong>.57</strong></td>
</tr>
</tbody>
</table>

- Speedup : 1.9
Vanilla European Call

- $S = 10$, strike = 10, $r = 5\%$, volatility = 20%, maturity = 1 Year
- Exact value: 1.04505

Wavelet method of lines (explicit Euler solver for the ODE)

<table>
<thead>
<tr>
<th>Space steps</th>
<th>Times Steps</th>
<th>Value</th>
<th>Solution time (seconds)</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>100</td>
<td>1.04465</td>
<td>0.005</td>
<td>0.00038</td>
</tr>
<tr>
<td>128</td>
<td>400</td>
<td>1.04496</td>
<td>0.02</td>
<td>0.00009</td>
</tr>
<tr>
<td>256</td>
<td>1400</td>
<td>1.04504</td>
<td>0.11</td>
<td>0.00001</td>
</tr>
<tr>
<td>512</td>
<td>6000</td>
<td>1.04505</td>
<td>0.94</td>
<td>0.00000</td>
</tr>
<tr>
<td>1024</td>
<td>60000</td>
<td>1.04505</td>
<td>8.95</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

Explicit Finite Difference Method

<table>
<thead>
<tr>
<th>Space steps</th>
<th>Times Steps</th>
<th>Value</th>
<th>Solution time (seconds)</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>64</td>
<td>200</td>
<td>1.04333</td>
<td>0.005</td>
<td>0.00165</td>
</tr>
<tr>
<td>128</td>
<td>800</td>
<td>1.04463</td>
<td>0.01</td>
<td>0.00041</td>
</tr>
<tr>
<td>256</td>
<td>3200</td>
<td>1.04495</td>
<td>0.05</td>
<td>0.00010</td>
</tr>
<tr>
<td>512</td>
<td>12800</td>
<td>1.04503</td>
<td>0.48</td>
<td>0.00002</td>
</tr>
<tr>
<td>1024</td>
<td>51200</td>
<td>1.04505</td>
<td>3.94</td>
<td>0.00000</td>
</tr>
</tbody>
</table>

- Speedup: 4.9
Cash or nothing binary call

- The cash-or-nothing call option has a payoff
  \[ \Pi(S) = B H(S - X) \]
  where \( H \) is the Heaviside function, i.e. if at expiry the stock price \( S > X \) the payoff is \( B \)
- The boundary conditions for this option in the transformed domain are
  \[
  u(x, \tau) = 0 \text{ as } x \to -\infty
  \]
  \[
  u(x, \tau) \sim \frac{B}{X} e^{2(k-1)x + \frac{1}{4}(k+1)^2 \tau} \text{ as } x \to \infty
  \]
  \[
  u(x, 0) = e^{2(k-1)x} \frac{B}{X} H\left( e^{2(k+1)x} - e^{2(k-1)x} , 0 \right)
  \]
Cash-or-nothing Call

- Same parameters as before, cash given B=3
- Exact value 1.59297

**Wavelet method of lines**

<table>
<thead>
<tr>
<th>Space Steps</th>
<th>Time Steps</th>
<th>Value</th>
<th>Solution time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>100</td>
<td>1.49683</td>
<td>.10</td>
</tr>
<tr>
<td>256</td>
<td>200</td>
<td>1.54904</td>
<td>.13</td>
</tr>
<tr>
<td>512</td>
<td>200</td>
<td>1.59216</td>
<td>.30</td>
</tr>
<tr>
<td>1024</td>
<td>400</td>
<td>1.59288</td>
<td>1.02</td>
</tr>
</tbody>
</table>

**Crank-Nicolson Finite Difference Method**

<table>
<thead>
<tr>
<th>Space Steps</th>
<th>Time Steps</th>
<th>Value</th>
<th>Solution time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>200</td>
<td>1.46296</td>
<td>.04</td>
</tr>
<tr>
<td>256</td>
<td>400</td>
<td>1.53061</td>
<td>.10</td>
</tr>
<tr>
<td>512</td>
<td>400</td>
<td>1.56391</td>
<td>.18</td>
</tr>
<tr>
<td>1024</td>
<td>400</td>
<td>1.58046</td>
<td>.31</td>
</tr>
<tr>
<td>2048</td>
<td>800</td>
<td>1.58872</td>
<td>1.35</td>
</tr>
<tr>
<td>4096</td>
<td>800</td>
<td>1.59285</td>
<td>2.56</td>
</tr>
</tbody>
</table>

- Speedup : 2.5
Supershare binary call

- The supershare binary call option pays an amount $1/d$ if the stock price lies between $X$ and $X+d$ at expiry.
- The payoff of this option is

$$
\Pi(S) = \frac{1}{d} \left( H(S - X) - H(S - X - d) \right)
$$

which becomes the Dirac delta function in the limit as $d \downarrow 0$.
- The boundary conditions for this option in the transformed domain are

$$
\begin{align*}
&u(x, \tau) = 0 \text{ as } x \to -\infty \\
&u(x, \tau) \sim 0 \text{ as } x \to \infty \\
&u(x, 0) = e^{2(k-1)x} \left( H(\text{e}^x X - X) - H(\text{e}^x X - X - d) \right) / dX
\end{align*}
$$
Supershare Binary Call

- Same parameters as before, parameter $d=3$
- Exact value 0.13855

**Wavelet method of lines**

<table>
<thead>
<tr>
<th>Space Steps</th>
<th>Time Steps</th>
<th>Value</th>
<th>Solution time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>100</td>
<td>.12796</td>
<td>.10</td>
</tr>
<tr>
<td>256</td>
<td>200</td>
<td>.13310</td>
<td>.14</td>
</tr>
<tr>
<td>512</td>
<td>200</td>
<td>.13808</td>
<td>.30</td>
</tr>
<tr>
<td><strong>1024</strong></td>
<td><strong>400</strong></td>
<td><strong>.13848</strong></td>
<td><strong>1.04</strong></td>
</tr>
</tbody>
</table>

**Crank-Nicolson Finite Difference Method**

<table>
<thead>
<tr>
<th>Space Steps</th>
<th>Time Steps</th>
<th>Value</th>
<th>Solution time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>128</td>
<td>200</td>
<td>.12369</td>
<td>.04</td>
</tr>
<tr>
<td>256</td>
<td>400</td>
<td>.13290</td>
<td>.09</td>
</tr>
<tr>
<td>512</td>
<td>400</td>
<td>.13435</td>
<td>.16</td>
</tr>
<tr>
<td>1024</td>
<td>400</td>
<td>.13666</td>
<td>.34</td>
</tr>
<tr>
<td>2048</td>
<td>800</td>
<td>.13787</td>
<td>1.35</td>
</tr>
<tr>
<td>4096</td>
<td>800</td>
<td>.13800</td>
<td>2.56</td>
</tr>
<tr>
<td><strong>8000</strong></td>
<td><strong>800</strong></td>
<td><strong>.13835</strong></td>
<td><strong>5.11</strong></td>
</tr>
</tbody>
</table>

- Speedup : 4.9
5. Cross-currency Swap Valuation

- We consider a 10 year cross-currency cancellable fixed-fixed swap with quarterly reset dates, a 2 day decision period and exchange of principal on first and last days.


- 1-factor extended Vasicek yield curve models lead to a 3D parabolic PDE of the form:

\[
\frac{\partial V}{\partial t} - \frac{1}{2} \nabla \left[ \Lambda(t)(\nabla V)' \right] = 0
\]
• Solved for the normalized deal value 39 times in backwards time with a backwards dynamic programming reset of the initial condition at each quarter

• Boundary conditions are set at $3\sigma$ of the state distributions and the computed deal is symmetric --value 0-- with the same initial term structures, mean reversions and forward volatility specification in both economies

• A 3D wavelet method of lines code in C/Fortran 90 and an explicit finite difference method in C have been implemented on an IBM RS6000/590 and an Athlon 650
Cross Currency Swap

- Domestic Fixed Rate=10%, Foreign Fixed Rate=10%
- Exact value : 0

<table>
<thead>
<tr>
<th>Wavelet method of lines</th>
<th>Discretization</th>
<th>Value</th>
<th>Solution time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>20 x 8 x 8 x 8</td>
<td>-0.00082</td>
<td>1.2</td>
</tr>
<tr>
<td></td>
<td>20 x 16 x16 x16</td>
<td>-0.00052</td>
<td>6.54</td>
</tr>
<tr>
<td></td>
<td>20 x 32 x32 x 32</td>
<td>-0.00047</td>
<td>40.40</td>
</tr>
<tr>
<td></td>
<td>40 x 64 x 64 x 64</td>
<td>-0.00034</td>
<td>410.10</td>
</tr>
<tr>
<td></td>
<td>100 x 128 x 128 x 128</td>
<td>-0.00028</td>
<td>4240.30</td>
</tr>
<tr>
<td></td>
<td>160 x 256 x 256 x 256</td>
<td>-0.00025</td>
<td>53348.10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Explicit Finite Difference Method</th>
<th>Value</th>
<th>Solution time (seconds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>20 x 8 x 8 x 8</td>
<td>-0.00109</td>
<td>0.28</td>
</tr>
<tr>
<td>20 x 16 x16 x16</td>
<td>-0.00101</td>
<td>1.70</td>
</tr>
<tr>
<td>20 x 32 x32 x 32</td>
<td>-0.00074</td>
<td>16.82</td>
</tr>
<tr>
<td>40 x 64 x 64 x 64</td>
<td>-0.00058</td>
<td>188.10</td>
</tr>
<tr>
<td>100 x 128 x 128 x 128</td>
<td>-0.00046</td>
<td>2421.6</td>
</tr>
<tr>
<td>160 x 256 x 256 x 256</td>
<td>-0.00038</td>
<td>33341.8</td>
</tr>
</tbody>
</table>

- Speedup : 81+
6. Thresholding

• Wavelets were originally created to reduce information storage

• How can this feature be used for PDEs?

• Some coefficients in the operator are small
  – They can be excluded from the computation without any significant effect on the solution

• Some coefficients in the wavelet space are small
  – They can also be excluded from the computation without any significant effect on the solution
Solving PDEs using the wavelet Galerkin method

\[ f(0) \rightarrow \tilde{f}(0) \rightarrow \tilde{f}(t + 1) = \tilde{L}\tilde{f}(t) \rightarrow \tilde{f}(T) \rightarrow f(T) \]

1. Transform initial condition to wavelet space
2. Build operator in the wavelet space and apply wavelet operator at each time step using method of lines
3. Find solution in wavelet space
4. Transform back to the physical space
Operator thresholding

• Apply the differential operator only for significant changes over time steps
  → Makes the operator matrix even more sparse

• The threshold is the cut-off point below which we exclude points
Galerkin

Interpolation error is orthogonal to the basis

Wavelet-Galerkin

• Apply operator in wavelet space
• Threshold and use sparse vector techniques

Collocation

Interpolation error is zero at collocation points

Wavelet-Collocation

• Use wavelets for compression and interpolation
• Wavelet coefficients design irregular grid
• Apply differential operator in physical space
Complete thresholding with the wavelet collocation method

- Use wavelets for compression and interpolation (define “transformed sparse grid”)
  → Wavelet coefficients adaptively design an irregular grid (multi-grid)
- Apply differential operator in physical space (collocation method)
- Far fewer grid points are needed in both time and space than for finite difference, finite element of spectral methods
Irregular grid definition

- Transform functions into the wavelet world
- Get rid of small coefficients
  - Create basic grid
- Create a mask
  - Add collocation (grid) points around non zero value
  - They will be used when the function changes around those values
- Update mask and grid after every time step
- No need to construct a full matrix approximation of the differential operator in time and physical space
**Alternative derivative evaluations in physical space**

- Evaluate **derivatives of wavelet functions**
  - computation of all coefficients contributions

- Use **finite differences on irregular grids** in physical space
  - construction of local finite difference operators

- **Interpolate solution and use finite differences on a uniform grid**
  - interpolation to the finest level  
    Vasiliev and Bowman (2000)
7. Conclusions and Further Work

- $O(N)$ wavelet based PDE methods generalize $O(N \log_2 N)$ spectral methods without their drawbacks.

- Wavelets are ideally suited for complex derivative valuations which involve several space scales e.g. due to payoff curvature or discontinuities and result in greater accuracy at a given discretization level with substantial speedups -- using prototype code -- over optimized finite difference codes in dimensions up to 3.

- We are currently developing a thresholded 3D wavelet collocation code which should improve both speedup and memory use by orders of magnitude through sparse wavelet representation.

- However, fast Monte Carlo techniques are currently the method to beat!

- On to 2 and 3 factor cross currency swap valuation in 5 and 7D…