Agenda

- Motivation/Problem
- State of the Art
- Setup
- Optimal Exercise Equations
- Fixed-Point Formulations
- Boundary Properties
- Algorithm
- Test Results
- Extension & Future Research
- Conclusion
Motivation/Problem

- “Let’s do something that is not CVA/Capital/Margin/.., please...
- Instead, let us try to build the world’s fastest high-precision American option pricer!”
- Also a motivating factor: we are developing real-time risk systems for large portfolios as American futures options (part of our “strat” effort).
- Extendability to time-dependent parameters and skews.
- Systems in this space are typically based on fast approximations to option price or to the optimal exercise. E.g., Ju and Zhong (1999), Barone-Adesi and Whaley (1987), and MANY others.
- Some of these approximations are pretty good, but fail to a) provide a reasonable mechanism to intelligently choose between performance and precision; and b) cannot be extended.
Existing Methods - (1)

- Numerous convergent numerical methods for American options have been proposed over the last 30+ years. For instance:
  - Binomial trees (ordinary, modified, accelerated,...)
  - SALI trees
  - Convolution methods, Fourier time-stepping
  - Trinomial Trees
  - Finite difference/element methods
  - Method of lines
  - Least squares Monte Carlo
  - Duality Monte Carlo
  - AND...

- *Integral Equations*
Existing Methods - (2)

- There is, of course, also quite a bit of literature on *comparisons* of various methods, but it is fair to say that there is a true winner.

- The integral equation approach is rarely listed as much of a contender, and in some academic papers it is (very) slow.

- Nevertheless, there have (long) been reports of good results in quite rudimentary implementations for applications in fixed income.

- So maybe worth taking another look?
For benchmarking and documentation purposes, we here consider the classical constant-parameter dynamics for a security price \( S(t) \):

\[
dS(t)/S(t) = (r - q) \, dt + \sigma \, dW(t),
\]

Introduce a \( K \)-strike, \( T \)-maturity American put, paying \((K - S(\nu))^+\) if exercised at time \( \nu \in [0, T] \). (American call: use put-call symmetry).

Price of American put is (\( \nu \) is stopping time)

\[
p(t) = \sup_{\nu \in [t, T]} \mathbb{E}_t \left( e^{-r(\nu-t)} (K - S(\nu))^+ \right)
\]

It is known that the \((t = 0)\) optimal exercise strategy is

\[
\nu^* = \inf \{ t \in [0, T] : S(t) \leq S_T^* (t) \},
\]

for some deterministic, and \( T \)-indexed, exercise boundary \( S_T^* \).
The exercise boundary satisfies (and is therefore sometimes discontinuous)

\[ S_T^*(T) = K, \quad S_T^*(T-) = K \min(1, r/q) \]

It is convenient to reverse arguments to write \( S_T^*(t) = B(T - t) \); this is possible due to time-homogeneity.

If also \( V(T - t, S) \) is the time \( t \) price of the American put for \( S(t) = S \) (i.e., \( p(t) = V(T - t, S(t)) \)), then with \( \tau = T - t \) and for \( S > B \),

\[
V_\tau - (r - q)V_S - \frac{1}{2}S^2\sigma^2V_{SS} + rV = 0, \quad V(0, S) = (K - S)^+,
\]

subject to the value match condition

\[
V(\tau, B(\tau)) = K - B(\tau), \quad (1)
\]
... and to the smooth pasting condition

\[ V_S(\tau, B(\tau)) = -1. \] (2)

Differentiating with respect to \( \tau \) and using the smooth pasting condition:

\[ V_\tau(\tau, B(\tau)) = 0. \] (3)

And using the PDE shows that

\[ V_{SS}(\tau, B(\tau)) = \frac{2(rK - qB(\tau))}{B(\tau)^2 \sigma^2}. \] (4)

From the basic PDE and (1)-(2), it is not difficult to show that the American put premium over the European price \( v(\tau, S) \) can be computed as a “carry” integral over the boundary.
Specifically, for $S \leq B$,

$$V(\tau, S) = v(\tau, S) + \int_0^\tau rK e^{-r(\tau-u)} \Phi(-d_- (\tau-u, S/B(u))) \, du$$

$$- \int_0^\tau qS e^{-q(\tau-u)} \Phi(-d_+ (\tau-u, S/B(u))) \, du,$$

$$d_{\pm} (s, x) = \frac{\ln x + s \left( r - q \pm \frac{1}{2} \sigma^2 \right)}{\sigma \sqrt{s}}$$

To use the integral pricing expression for $V(\tau, S)$, we need to locate the optimal exercise boundary $B$.

The various conditions (1)-(4) may be applied to (5) to write many alternative equations for the boundary.
Value Match Boundary Equation - (1)

Set \( S = B(\tau) \) in (5) and use (1):

\[
K - B(\tau) = v(\tau, B(\tau)) + \int_0^\tau rK e^{-r(\tau-u)} \Phi (-d_-(\tau-u, B(\tau)/B(u))) \, du \\
- \int_0^\tau qB(\tau) e^{-q(\tau-u)} \Phi (-d_+(\tau-u, B(\tau)/B(u))) \, du. \tag{6}
\]

This equation is Volterra-like, but more complicated due to the presence of \( B(\tau) \) on the r.h.s.

The numerical solution of this equation is typically done using a direct quadrature method.

Loosely here we work on a (typically equidistant) grid \( \{\tau_i\}_{i=1}^n \), and uncover \( B(\tau_i), i = 1, \ldots, n \), starting from known condition at \( B(0+) = K \min(1, r/q) \).
Value Match Boundary Equation - (2)

The integrals in (6) are resolved using a quadrature rule of the type

\[ \int_{0}^{\tau_i} g(u, \tau_i, B(\tau_i), B(u)) \, du = \sum_{j=0}^{i} g(\tau_j, \tau_i, B(\tau_i), B(\tau_j))w_j \]

where \( w_j \) are integration weights (e.g., trapezoid).

So, if \( B(\tau_1), B(\tau_2), \ldots, B(\tau_{k-1}) \) are known, the l.h.s. and r.h.s. of (6) are functions only of \( B(\tau_k) \), which can be found by a 1-dimensional root-search iteration.

After the boundary has been approximation, the option value can be computed from (5). If interpolation on the boundary is needed, we can draw a spline (say) between the \( B(\tau_i) \).

Computational effort is \( O(mn^2) \) where \( n \) is the number of discretization points and \( m \) is the number of root search iterations.
Other Boundary Equations

If we differentiate the expression for $V(\tau, S)$ w.r.t. $S$ and apply the smooth pasting condition, we get ($\Phi$ and $\phi$: Gaussian CDF and PDF)

$$B(\tau)e^{-q\tau}\Phi(d_+(\tau, B(\tau)/K)) = rK\int_0^\tau \frac{e^{ru}}{\sigma\sqrt{\tau - u}}\phi(d_-(\tau - u, B(\tau)/B(u))) \, du$$

$$+ B(\tau)e^{-q\tau} q\int_0^\tau e^{qu} \left( \Phi(d_+(\tau - u, B(\tau)/B(u))) + \frac{\phi(d_+(\tau - u, B(\tau)/B(u)))}{\sigma\sqrt{\tau - u}} \right) \, du.$$

And so on, using the remaining boundary equations (3)-(4). These equations may also be solved by direct quadrature methods.

As direct quadrature is not a particularly performant way of solving integral equations, one wonders at this point whether numerics improve if we rewrite the boundary equation(s) to the form

$$B(\tau) = f(\tau, B).$$
Fixed Point Algorithm - (1)

- If the functional $f$ is a contraction mapping, then we can locate $B$ by a fixed point iteration.

- First, usable forms of $f$ (which most are not) typically gives rise to equations like

$$B(\tau) = Ke^{-(r-q)\tau} \frac{N(\tau, B)}{D(\tau, B)}$$

where $N$ and $D$ are functionals of $B$.

- This suggests an algorithm where we iterate, starting from a guess,

$$B^{(j)}(\tau) = Ke^{-(r-q)\tau} \frac{N(\tau, B^{(j-1)})}{D(\tau, B^{(j-1)})}, \quad j = 1, 2, \ldots, m.$$

- Note: if this is done on a discrete set of $\tau$, we obviously have a parallelizable algorithm (something that we shall NOT exploit – this is cheating!)
Fixed Point Algorithm - (2)

The smooth pasting integral equation found earlier (after some minor manipulations to “symmetrize” integral and non-integral terms) gives rise to fixed point system A:

\[
N(\tau, B) = \frac{\phi((d_-(\tau, B(\tau)/K))}{\sigma \sqrt{\tau}} + r \int_0^\tau \frac{e^{ru}}{\sigma \sqrt{\tau - u}} \phi((d_- (\tau - u, B(\tau)/B(u)))) \, du,
\]

\[
D(\tau, B) = \frac{\phi((d_+(\tau, B(\tau)/K))}{\sigma \sqrt{\tau}} + \Phi((d_+ (\tau, B(\tau)/K))
\]

\[
+ q \left( \int_0^\tau e^{qu} \Phi((d_+ (\tau - u, B(\tau)/B(u)))) \, du + \int_0^\tau \frac{e^{qu}}{\sigma \sqrt{\tau - u}} \phi((d_+ (\tau - u, B(\tau)/B(u)))) \right).
\]

The value match integral equation leads to fixed point system B:

\[
N(\tau, B) = \Phi((d_-(\tau, B(\tau)/K)) + r \int_0^\tau e^{ru} (\Phi((d_- (\tau - u, B(\tau)/B(u)))))) \, du,
\]

\[
D(\tau, B) = \Phi((d_+(\tau, B(\tau)/K)) + q \int_0^\tau e^{qu} (\Phi((d_+ (\tau - u, B(\tau)/B(u)))))) \, du.
\]
For a fixed point iteration to work well, we need the r.h.s. of (7) to be insensitive to $B$. This will guarantee stability, and will also ensure high-order convergence speed in a region around the optimum.

Sensitivity to boundary can be measured through a functional derivative, e.g. the Gateaux derivative.

We don’t list this derivative here in all generality. For derivatives in the specific direction $\mu$ of flat proportional shifts (i.e., parallel shifts in log-$B$ space), for both fixed point systems $A$ and $B$:

$$\frac{\partial}{\partial \mu} \left( \frac{N(\tau, Q)}{D(\tau, Q)} \right) \bigg|_{Q=B} = 0.$$

This is obviously a good sign. Further analysis of the Gateaux derivative and of higher-order derivatives w.r.t. $\mu$, reveals that system $A$ typically can be expected to be faster-converging than $B$. 
However, for $r - q$ big and positive, and $\tau$ large, system A may become unstable if executed naively (e.g., without relaxation or similar modifications).

The fundamental fixed point formulation (7) is not unique, obviously. For a semi-classical relaxation modification, consider writing

$$B(\tau) = Ke^{-(r-q)\tau} \frac{N(\tau, B)}{D(\tau, B)} (1 - h(\tau, B)) + h(\tau, B)B(\tau)$$

$$\triangleq f(\tau, B) (1 - h(\tau, B)) + h(\tau, B)B(\tau)$$

for some exogenously specified $h$. If $h$ is set carefully, a fixed point iteration on the modified r.h.s. may become faster and/or more stable.
While $f$ depends on the entire function $B(u)$, $u \leq \tau$, it is typically “diagonally dominant”, in the sense that $B(\tau)$ figures more prominently than $B(u)$, $u < \tau$.

For such a system, a Jacobi-Newton modification of the iteration is reasonable. That is, we set $h$ such that the derivative of the r.h.s. is forced to zero.

Cranking through the math and also introducing an additional dampening factor $\eta \leq 1$, we get

$$B^{(j)}(\tau) = B^{(j-1)}(\tau) + \eta \frac{B^{(j-1)}(\tau) - f(\tau, B^{(j-1)})}{f'(\tau, B^{(j-1)}) - 1}$$

where $f' = \partial f / \partial B(\tau)$ is easily calculated (see paper for gory details).
Interlude: Boundary Properties - (1)

First, short-term behavior (e.g., Li-Zhang 2010):

\[
\frac{B(\tau)}{K \min(1, r/q)} \sim \begin{cases} 
\exp \left( -\sqrt{-k_1 \tau \ln(k_2 \tau)} \right), & r = q, \\
\exp \left( -\sqrt{-\frac{1}{2}k_1 \tau \ln(k_3 \tau)} \right), & r > q, \\
\exp(-k_4 \sqrt{\tau}), & r < q,
\end{cases}
\]

\[k_1 = 2\sigma^2, \quad k_2 = 4\sqrt{\pi}r, \quad k_3 = 8\pi \left( \frac{r - q}{\sigma} \right)^2, \quad k_4 \approx \sigma \sqrt{2} \cdot 0.451723.\]

Second, long-term asymptote level (Merton 1973)

\[B_{\inf} = K \frac{\theta}{\theta - 1}, \quad \theta = \alpha - \sqrt{\beta}, \quad \alpha = \frac{1}{2} - \frac{r - q}{\sigma^2}, \quad \beta = \alpha^2 + 2\sigma^{-2}r.\]
Transition to long-term level (Chen et al 2011, const is unknown):

\[
\ln B(\tau) \sim \ln B_{\text{inf}} + \text{const} \cdot \left(\frac{\sigma^2 \tau}{2}\right)^{-3/2} e^{-\beta \tau}
\]

Generally, for the shape of the boundary (Chen et al (2009)):

**Theorem 1.** The American put exercise boundary \( B(\tau) \) is infinitely differentiable \((C^\infty)\) on \( \tau \in (0, \infty) \). When \( r \geq q \) or \( q \gg r \), the boundary \( B(\tau) \) is convex for all \( \tau > 0 \). However, when \( q > r \), and \( q - r \ll 1 \) then \( B(\tau) \) is not uniformly convex in \( \tau \). In particular, if \( \varepsilon = \ln(q/r) \) is positive and sufficiently small, then there exists a \( \hat{\tau} \) for which \( d^2 B(\hat{\tau})/d\tau^2 < 0 \), where

\[
0 < \hat{\tau} \leq \frac{\varepsilon}{3\sigma^2 |\ln(\varepsilon)|}.
\]

**In other words,** \( B \) is smooth, but not always convex if \( q > r \) and close to \( r \).
Boundary Asymptotes - (1)

Boundary Asymptotes $r = q = 5\%, \sigma = 0.25\%, K = 130.$

Neither short nor long-dated asymptotes have wide range. Short asymptote ceases to exist after 2.8 years.
Boundary Asymptotes - (2)

- Boundary Asymptotes \( r = 4.5\%, q = 5\%, \sigma = 0.25\%, K = 130. \)

- Lack of convexity ruins the short-time asymptote, except for very close to boundary.
Approximations

The short-term expressions often contain logarithms and cease to exist everywhere.

If we want a single analytical formula for the barrier – for both approximation and for better understanding – a power function can be both justified and verified to work well:

\[ B(\tau) = B_{\text{inf}} + (K \min(1, r/q) - B_{\text{inf}})e^{-\sqrt{a\tau^{b+1}}} \]

The constants \(a\) and \(b\) can be found analytically (except for “bad case” \(r\) slightly less than \(q\)) by matching levels and slope to short-term approximation at a well-chosen level \(\epsilon\).

Details in paper + results on how to handle non-convex case.

Virtually always, the constant \(b\) is \textbf{small and less than zero} (e.g., around -0.1)
New Algorithm: Integrals - (1)

- Due to the more complicated integrals (singular kernels) in fixed point system A, we just focus on this. (B is easier).

- Jacobi-Newton iteration has already been defined, but needs supported from a numerical integration scheme to evaluate the integrals in $N$ and $D$ on the r.h.s.

- The integrals contain a singular element, $(\tau - u)^{-1/2}$, which must be handled. We use variable transformation:

$$z = \sqrt{\tau - u} \implies dz = -\frac{1}{2}(\tau - u)^{-1/2} \, du.$$  

- Additionally, we would like to normalize the integrals to the interval $[-1, 1]$, which suggests the final transformation

$$y = -1 + 2\frac{z}{\sqrt{\tau}} = -1 + 2\frac{\sqrt{\tau - u}}{\sqrt{\tau}}.$$
New Algorithm: Integrals - (2)

With this, we get (for $g$ representing the various integration kernels in fixed point system A)

$$
\int_{0}^{\tau} \frac{g(u, \tau)}{\sqrt{\tau - u}} \, du = 2 \int_{0}^{\sqrt{\tau}} g(\tau - z^2, \tau) \, dz = \sqrt{\tau} \int_{-1}^{1} g \left( \tau - \tau \left(1 + \frac{y^2}{4}\right), \tau \right) \, dy.
$$

To handle these types of integrals, we use $l$-point Gauss-Legendre quadrature, i.e. we write

$$
\int_{-1}^{1} g \left( \tau - \tau \left(1 + \frac{y^2}{4}\right), \tau \right) \, dy \approx \sum_{k=1}^{l} g \left( \tau - \tau \left(1 + y_k^2 / 4\right), \tau \right) w_k
$$

where the weights $w_k$ and the nodes $y_k$ are those of Gauss-Legendre quadrature.

Remember that this method integrates exactly polynomials up to degree $2l - 1$. 

Really Fast American Option Pricing – p. 24/35
The fixed point systems cannot practically be solved for all $\tau$ simultaneously, so we need a way to discretize the system.

We use a collocation approach, and discretize $\tau$ to a grid, $\{\tau_i\}_{i=1}^n$. The fixed point condition is enforced at these points only. Other points on the $B(\tau)$ curve are found by interpolation.

We wish to employ polynomial interpolation, which can be very effective if done right.

**WRONG**: a) interpolate on $B$ directly; b) use an equidistant grid.

The equidistant grid will cause oscillations in the polynomial (the Runge phenomenon) and will generally not provide an optimal fit.

And $B$ is, while smooth, not well-approximated by a low-dimensional polynomial – the gradients around the origin are too large.
**Collocation/Interpolation - (2)**

- **RIGHT**: a) interpolate on a transformed function \( H(\sqrt{\tau}) = \ln B(\tau)/X \), \( X = K \min(1, r/q) \); and b) Use Chebyshev spacing in \( \sqrt{\tau} \) domain.

- Function \( H, r = q = 5\%, \sigma = 0.25\%, K = 130 \).

- Much smoother – as expected from our earlier approximations.
Chebyshev spacing is known to eliminate the Runge phenomenon, and also to often be remarkable close in reproducing the minimax polynomial. And it is easy to compute the interpolating polynomial – see paper or any numerical recipes book. Cost is $O(n)$.

The location of the nodes is particularly simple. If $x = \sqrt{\tau}$, then we have

$$x_i = \frac{\sqrt{\tau_{\text{max}}}}{2} (1 + z_i), \quad z_i = \cos \left( \frac{i\pi}{n} \right), \quad i = 1, \ldots, n.$$  

This also defines the collocation grid in $\tau$-space: $\tau_i = x_i^2$, with $\tau_0 = 0^+$. Here, $\tau_{\text{max}}$ is the largest maturity we are interested in.
Summary of Algorithm - (1)

1. Preprocessing. Compute the Chebyshev nodes, establish the collocation grid \( \{ \tau_i \}_{i=1}^n \) by Chebyshev interpolation on \( \sqrt{\tau} \). Set the \( l \) Gauss-Legendre nodes and weights. Etc.

2. Use the earlier approximation methods (or some other method from the literature) to establish an initial guess for \( B \) on the \( n \) points \( \{ \tau_i \}_{i=1}^n \). Let the guess be denoted \( B^{(0)}(\tau_i), i = 0, \ldots, n \).

3. For \( j = 1 \) to \( j = m \), LOOP on Steps 4-7.

4. For all \( \tau_i, i = 1, \ldots, n \), use Chebyshev interp. on function \( H \) to find

\[
B^{(j-1)}(\tau_i - \tau_i (1 + y_k)^2 / 4), \quad k = 1, \ldots, l.
\]

5. Use Gaussian quadrature to establish \( N(\tau_i, B^{(j-1)}) \) and \( D(\tau_i, B^{(j-1)}) \). Compute \( f(\tau_i, B^{(j-1)}), i = 1, \ldots, n \).
6. Compute $f' (\tau_i, B^{(j-1)}), i = 1, \ldots, n$, or, if ordinary fixed point iteration is to be used, set $f' = 0$.

7. Compute $B^{(j)}(\tau_i), i = 1, \ldots, n$, from the Jacobi-Newton update equation.

8. We are done. The optimal exercise boundary at $\{\tau_i\}_{i=0}^n$ is approximated by $B(\tau_i) \approx B^{(m)}(\tau_i)$. For $\tau$ in between nodes in $\{\tau_i\}_{i=0}^n$, the Chebyshev interpolant of $B^{(m)}$ is used to approximate $B(\tau)$.

- There are a total on $ln$ interpolations per iteration (each of cost $O(n)$); and there are $n$ integrations per iteration, each of cost $O(l)$.

- Computation cost is therefore seen to be of order $(c_{\text{interp}} \ll c_{\text{integ}})$

$$c_{\text{interp}} \cdot lmn^2 + c_{\text{integ}} \cdot lmn.$$
There are two attempts at building fixed point iterations for American options in the literature: Kim et al (2013) and Cortazar et al (2013).

The former always has $q = 0$ and effectively uses fixed point system A, although it has been derived in a roundabout way. The implementation is simplistic.

The latter (which came out after we had finished our own implementations) uses fixed points system B and simple (no Jacobi-Newton) iteration. Also, their recommended approach involves trapezoid integration rule + splines.

In their Tables 2 and 3, Cortazar et al test a large variety of methods (including those of Kim et al (2013)) – numbers that we can conveniently re-use here!
Warm-Up: Test on 3-Yr Option

\( r = q = 4\%, \sigma = 20\%, K = 100 \)

<table>
<thead>
<tr>
<th>Spot S</th>
<th>True S Price</th>
<th>Bin 1,000</th>
<th>Bin 10,000</th>
<th>Bin-BS 15,000</th>
<th>FIK-F 60</th>
<th>FIK-F 400</th>
<th>KJK 32</th>
<th>FP-A (7,2,6)</th>
<th>FP-B (7,2,6)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>3.1E-04</td>
<td>4.8E-05</td>
<td>3.1E-05</td>
<td>8.7E-04</td>
<td>6.1E-05</td>
<td>2.1E-04</td>
<td>7.0E-06</td>
<td>2.0E-04</td>
</tr>
<tr>
<td></td>
<td></td>
<td>2.4E-03</td>
<td>2.4E-04</td>
<td>7.8E-05</td>
<td>7.1E-04</td>
<td>4.8E-05</td>
<td>2.7E-04</td>
<td>1.1E-05</td>
<td>2.0E-05</td>
</tr>
<tr>
<td></td>
<td></td>
<td>1.8E-03</td>
<td>1.8E-04</td>
<td>4.5E-05</td>
<td>4.7E-04</td>
<td>3.5E-05</td>
<td>2.6E-04</td>
<td>2.8E-06</td>
<td>6.6E-06</td>
</tr>
</tbody>
</table>

First row: FIK-F: Method in Cortazar et al; KJK: Method in Kim et al; Bin: Binomial Tree; Bin-BS: Modified Binomial Tree; FP-A and FP-B: new fixed point algorithm (systems A and B, respectively)

Second row: number of steps. For FP-A and FP-B: (l,m,n).

KJK and FIK: # of iterations is in the range 5-6 on average.
## More Tests

### All options in Cortazar et al, Tables 2 and 3

<table>
<thead>
<tr>
<th>Bin</th>
<th>Bin</th>
<th>Bin-BS</th>
<th>FIK-F</th>
<th>FIK-F</th>
<th>KJK</th>
<th>FP-A</th>
<th>FP-B</th>
</tr>
</thead>
<tbody>
<tr>
<td>1,000</td>
<td>10,000</td>
<td>15,000</td>
<td>60</td>
<td>400</td>
<td>32</td>
<td>(7,2,6)</td>
<td>(7,2,6)</td>
</tr>
<tr>
<td>RMS</td>
<td>2.7E-03</td>
<td>3.0E-04</td>
<td>7.9E-05</td>
<td>9.3E-04</td>
<td>6.5E-05</td>
<td>2.8E-03</td>
<td>1.3E-05</td>
</tr>
<tr>
<td>RRMS</td>
<td>8.1E-04</td>
<td>8.8E-05</td>
<td>2.4E-05</td>
<td>2.7E-04</td>
<td>1.9E-05</td>
<td>8.3E-04</td>
<td>3.8E-06</td>
</tr>
</tbody>
</table>
More Tests + Speed

- 1,675 different options, $T \in [0, 3]$. Benchmarks (3.33GHz PC):

<table>
<thead>
<tr>
<th></th>
<th>Bin 100</th>
<th>Bin 1,000</th>
<th>Bin 10,000</th>
</tr>
</thead>
<tbody>
<tr>
<td>RMSE</td>
<td>2.1E-02</td>
<td>2.0E-03</td>
<td>2.1E-04</td>
</tr>
<tr>
<td>RRMSE</td>
<td>3.2E-03</td>
<td>2.7E-04</td>
<td>3.0E-05</td>
</tr>
<tr>
<td>Options/sec</td>
<td>12,900</td>
<td>800</td>
<td>?</td>
</tr>
</tbody>
</table>

- Algo FP-A, various combinations of $l, m, n$. No caching, single CPU.

<table>
<thead>
<tr>
<th></th>
<th>(m,n):</th>
<th>(1,4)</th>
<th>(2,4)</th>
<th>(1,6)</th>
<th>(2,6)</th>
<th>(3,6)</th>
<th>(4,6)</th>
<th>(2,10)</th>
<th>(3,10)</th>
<th>(4,10)</th>
</tr>
</thead>
<tbody>
<tr>
<td>l=5</td>
<td>RMSE</td>
<td>3.1E-04</td>
<td>3.1E-04</td>
<td>8.3E-05</td>
<td>4.4E-05</td>
<td>5.6E-05</td>
<td>5.7E-05</td>
<td>5.2E-05</td>
<td>6.4E-05</td>
<td>6.6E-05</td>
</tr>
<tr>
<td></td>
<td>RRMSE</td>
<td>2.0E-05</td>
<td>1.8E-05</td>
<td>3.0E-06</td>
<td>1.5E-06</td>
<td>1.7E-06</td>
<td>1.7E-06</td>
<td>1.6E-06</td>
<td>1.8E-06</td>
<td>1.9E-06</td>
</tr>
<tr>
<td></td>
<td>Options/sec</td>
<td>79,700</td>
<td>61,200</td>
<td>61,200</td>
<td>45,500</td>
<td>36,000</td>
<td>29,900</td>
<td>29,300</td>
<td>22,600</td>
<td>18,500</td>
</tr>
<tr>
<td>l=7</td>
<td>RMSE</td>
<td>3.4E-04</td>
<td>3.3E-04</td>
<td>8.4E-05</td>
<td>1.5E-05</td>
<td>1.3E-05</td>
<td>1.4E-05</td>
<td>7.3E-06</td>
<td>1.6E-05</td>
<td>1.8E-05</td>
</tr>
<tr>
<td></td>
<td>RRMSE</td>
<td>2.0E-05</td>
<td>1.9E-05</td>
<td>2.6E-06</td>
<td>6.6E-07</td>
<td>6.8E-07</td>
<td>6.4E-07</td>
<td>3.6E-07</td>
<td>5.3E-07</td>
<td>5.6E-07</td>
</tr>
<tr>
<td></td>
<td>Options/sec</td>
<td>74,500</td>
<td>55,300</td>
<td>57,200</td>
<td>39,500</td>
<td>30,900</td>
<td>25,500</td>
<td>25,500</td>
<td>19,200</td>
<td>15,400</td>
</tr>
<tr>
<td>l=15</td>
<td>RMSE</td>
<td>3.5E-04</td>
<td>3.4E-04</td>
<td>8.9E-05</td>
<td>2.3E-05</td>
<td>1.3E-05</td>
<td>1.2E-05</td>
<td>1.6E-05</td>
<td>4.2E-06</td>
<td>3.0E-06</td>
</tr>
<tr>
<td></td>
<td>RRMSE</td>
<td>2.0E-05</td>
<td>1.9E-05</td>
<td>2.6E-06</td>
<td>7.6E-07</td>
<td>6.5E-07</td>
<td>6.5E-07</td>
<td>4.8E-07</td>
<td>3.7E-07</td>
<td>3.5E-07</td>
</tr>
<tr>
<td></td>
<td>Options/sec</td>
<td>56,300</td>
<td>37,500</td>
<td>41,250</td>
<td>26,300</td>
<td>19,300</td>
<td>15,200</td>
<td>15,500</td>
<td>11,100</td>
<td>8,700</td>
</tr>
</tbody>
</table>
Extensions & Future Research

- The method can easily be extended to time-dependence. Basically, make substitutions along the lines of

\[ x \mapsto x(t), \quad x = r, q, \sigma^2, \]

\[ x \cdot \tau \mapsto \int_0^\tau x(u) \, du, \quad x = r, q, \sigma^2 \]

- If there is strong term structures, expect that higher values of \( l, m, n \) will be needed. As is the case for any numerical method.

- Also, method can be extended to other dynamics, as long as there is a closed-form (or very rapid approximate) European option pricing formula. For example: CEV, displaced LN.
Extensions & Future Research

- Trivial: Exchange options
- To stochastic volatility and jumps: more complex, but not inconceivable.
- To interest rate models: definitely for Gaussian models (Hull-White), possibly for Cheyette.
- Usage in CVA?