Stochastic Receding Horizon Control for Dynamic Option Hedging

Alberto Bemporad

http://www.dii.unisi.it/~bemporad

University of Siena

joint work with L. Bellucci and T. Gabbriellini, MPS Capital Services
**Talk outline**

- **Dynamic hedging** problem for financial options

- Dynamic hedging as a **linear stochastic control** problem

- **Stochastic receding horizon control** (SRHC)

- **Pricing engine + SRHC** = dynamic option hedging tool

- **Simulation results**
Dynamic hedging problem for financial options

- The financial institution sells a synthetic option to a customer and gets $w(0)$ (€)

- Such money is used to create a portfolio $w(t)$ of underlying assets (e.g. stocks) whose prices at time $t$ are $x_1(t), x_2(t), ..., x_n(t)$

- At the expiration date $T$, the option is worth the payoff $p(T) = \text{wealth (€)}$ to be returned to the customer

How to adjust dynamically the portfolio so that wealth $w(T) = \text{payoff } p(T)$?...

.. for any price realization $x_i(t)$?
Dynamics of traded assets

• Trading instants: \(\{0, \Delta t, 2\Delta t, \ldots, t\Delta t, \ldots, (T - 1)\Delta t\}\)

• Generic asset price dynamics:

\[
\begin{align*}
  x_i(t + 1) &= f_i(x_i(t), y_i(t), z^x_i(t)) \\
  y_i(t + 1) &= g_i(y_i(t), z^y_i(t))
\end{align*}
\]

• Example: \(x_i(t)\) = stock price, log-normal model (BS, Black-Scholes)

\[
dx_i = (\mu dt + \sigma dz^x_i) x_i
\]

geometric Brownian motion

Ito’s lemma

\[
x_i(t + 1) = e^{\left(\mu - \frac{1}{2} \sigma^2\right) \Delta t + \sigma \sqrt{\Delta t} z^x_i(t)} x_i(t)
\]

\(y_i(t) = 0, \quad y_i(t) \equiv 0\)

• Example: \(x_i(t)\) = stock price, GARCH(1,1) model (Heston, Nandi, 2000)

• Example: \(x_i(t)\) = option price of European call, based on BS model

Note: numerical integration can be also used to express \(x_i(t+1), y_i(t+1)\) as a function of \(x_i(t), y_i(t)\).
Portfolio dynamics

• Portfolio wealth at time $t$:

$$w(t) = u_0(t) + \sum_{i=1}^{n} x_i(t)u_i(t)$$

- money in bank account (risk-free asset)
- number of assets #i
- price of asset #i

• Assets traded at discrete-time intervals under the self-balancing constraint:

$$w(t + 1) = (1 + r)u_0(t) + \sum_{i=1}^{n} x_i(t + 1)u_i(t)$$

$$w(t + 1) = (1 + r)w(t) + \sum_{i=0}^{n} b_i(t)u_i(t)$$

$$b_i(t) \triangleq x_i(t + 1) - (1 + r)x_i(t)$$

• Assumption: transaction costs can be neglected (this can be relaxed...).

• Assumption: option pricing engine is available (more later in this talk ...), only focus on dynamic hedging. In particular, $w(0)$ is assigned.
Payoff function

- Let $x(t) = \begin{bmatrix} x_1(t) \\ \vdots \\ x_n(t) \end{bmatrix}$ be the vector of assets, and $y(t) = \begin{bmatrix} y_1(t) \\ \vdots \\ y_n(t) \end{bmatrix}$

- Option price $p(t)$: $p(t) = f(x(t), y(t))$
  
  $p(t) = f(x(0), x(1), \ldots, x(t), y(t))$  \hspace{1cm} \text{path-dependent options}$

- Payoff $p(T)$: $p(T) = f(x(0), x(1), \ldots, x(T))$

- Examples (n=1):
  
  $p(T) = \max\{x(T) - K, 0\}$  \hspace{1cm} \text{European call}$

  $p(T) = \max\left\{ 0, C + \min_{i \in \{1, \ldots, N_{\text{fix}}\}} \frac{x(t_i) - x(t_{i-1})}{x(t_{i-1})} \right\}$  \hspace{1cm} \text{Napoleon cliquet}$

\hspace{1cm} (t_i = \text{fixing dates})
Talk outline

✓ Dynamic hedging problem for financial options

• Dynamic hedging as a linear stochastic control problem

• Stochastic receding horizon control (SRHC)

• Pricing engine + SRHC = dynamic option hedging tool

• Simulation results
Option hedging = linear stochastic control

- Portfolio dynamics is linear, with multiplicative stochastic noise

\[ w(t + 1) = (1 + r)w(t) + \sum_{i=0}^{n} b_i(t) u_i(t) \]

- Block diagram of stochastic control problem:

- Control objective: \( w(T) \) should be as close as possible to \( p(T) \), for any possible realization of the asset prices \( x(t) \) ("tracking w/ disturbance rejection")
Control objective

- Define **hedging error** \( e(t) \triangleq w(t) - p(t) \) — we want \( e(T) \) small!

- Minimize **expected** hedging error:
  \[
  \min E [e(T)]^2
  \]

- Minimize **expected squared** error:
  \[
  \min E [e(T)^2]
  \]

In fact:
\[
E [e(T)^2] = E [(e(T) - E[e(T)])^2] + \alpha E [e(T)]^2 \quad \text{for } \alpha = 1
\]

- Minimize **variance** of hedging error:
  \[
  \min E [(e(T) - E[e(T)])^2]
  \]

Very risky, variance of \( e(T) \) may be large!

If minimum is 0 then both mean and variance of hedging error are 0

How about \( E[e(T)] \)?

A. Bemporad

“Stochastic RHC for Dynamic Option Hedging” - Siena, September 7, 2009
Minimum variance control

• Let us minimize the variance of hedging error:

\[ \min E \left[ (e(T) - E[e(T)])^2 \right] \]

**Theorem**  Let \( w(0) = p(0) \). Under **non-arbitrage conditions**, if a strategy exists such that \( \text{Var}[w(T) - p(T)] = 0 \) (\( = \)perfect hedging) then

\[ w(t) - p(t) = 0, \quad \forall t = 0, 1, \ldots, T \]

and in particular \( E[w(T) - p(T)] = 0 \), that is the final hedging error is deterministically 0.

(cf. Black-Scholes theory)

**Proof**: By induction.

**Note**: \( \text{Var}[e(T)] = 0 \) means \( e(T) = w(T) - p(T) \) is deterministic

\( e(T) > 0 \) would imply \( w(T) > p(T) \) → always **gain** wealth, which is impossible if no arbitrage exists, as

\[ w(0) = (1 + r)^{-T} E[p(T)] \]
Existing approaches to dynamic option hedging

• **Analytical approaches**: choose $u(t)$ to “reject” exactly the effect of stochastic noise $z(t)$
  
  **PROS**: lots of insight!
  **CONS**: limited to simple stock models & payoff functions

  (Black and Scholes, 1973)  
  (Merton, 1973)

• **Multi-stage stochastic programming**: choose $u(t)$ by solving a (large-scale) optimization problem.
  
  **PROS**: pricing & hedging in one shot, check arbitrage conditions
  **CONS**: rough discretization of probability space & trading dates

  (Edirisinghe et al., 1993)  
  (Gondzio et al., 2003)  
  (Klaassen, 1998)  
  (Zhao and Ziemba, 1998)

• **Stochastic dynamic programming**:
  
  **PROS**: rather versatile
  **CONS**: rough discretization of probability space & trading dates

  (Bertsimas et al., 2001)

• **Stochastic model predictive control**: solve recursively and on-line an optimization problem over a short prediction horizon
  
  **PROS**: very versatile
  **CONS**: requires on-line optimization

  (Primbs, 2009)  
  (this talk)
Talk outline

✓ Dynamic hedging problem for financial options

✓ Dynamic hedging as a linear stochastic control problem

• Stochastic receding horizon control (SRHC)

• Pricing engine + SRHC = dynamic option hedging tool

• Simulation results
A model of the process is used to predict the future evolution of the process in order to optimize the control signal.
Receding horizon philosophy

- **At time** $t$: solve an **optimal control** problem over a finite future horizon of $N$ steps:

$$\min_z \sum_{k=0}^{N-1} \|w_{t+k} - p(t)\|^2 + \rho\|u_{t+k}\|^2$$

**s.t.**

- $w_{t+k+1} = f(w_{t+k}, u_{t+k})$
- $u_{\text{min}} \leq u_{t+k} \leq u_{\text{max}}$
- $w_{\text{min}} \leq w_{t+k} \leq w_{\text{max}}$
- $w_t = w(t), \ k = 0, \ldots, N - 1$

- **Only apply the first optimal move** $u^*(t)$

- **At time** $t+1$: **Get new measurement** $w(t+1)$, repeat the optimization. And so on …

- **Stochastic MPC**: minimize **expected value** of cost function
Stochastic linear MPC

• Assume stochastic model of the process

\[ w(t + 1) = A(z_1(t))w(t) + B(z_1(t))u(t) + E z_2(t) \quad w \in \mathbb{R}^n \]

\[ z_1(t), z_2(t) = \text{stochastic noise} \]

• Optimize stochastic performance under (chance) constraints

\[
\min E \left[ w_N^t P w_N + \sum_{k=0}^{N-1} w_k^t Q w_k + u_k^t R u_k \right]
\]

• Ensure mean-square convergence \( E[w'(t)w(t)] = 0 \)

• A few SMPC approaches exist in the control literature

(Schwarme & Nikolaou, 1999) (Munoz de la Pena, Bemporad, Alamo, 2005) (Ono, Williams, 2008)
(Wendt & Wozny, 2000) (Couchman, Cannon, Kouvaritakis, 2006)
(Batina, Stoorvogel, Weiland, 2002) (Primbs, 2007)
(van Hessem & Bosgra 2002) (Oldewurtel, Jones, Morari, 2008) (Bernardini & Bemporad, 2009)
## Talk outline

- Dynamic hedging problem for financial options
- Dynamic hedging as a linear stochastic control problem
- Stochastic receding horizon control (SRHC)

- **Pricing engine + SRHC** = dynamic option hedging tool

- **Simulation results**
SMPC for dynamic option hedging

• Stochastic finite-horizon optimal control problem:

\[
\begin{align*}
\min_{\{u(k,z)\}} \quad & \text{Var}_z [w(T, z) - p(T, z)] \\
\text{s.t.} \quad & w(k + 1, z) = (1 + r)w(k, z) + \sum_{i=0}^{n} b_i(k, z)u_i(k, z), \ k = t, \ldots, T - 1 \\
& w(t, z) = w(t)
\end{align*}
\]

\[
z = \{z(t + 1), \ldots, z(T)\} \in \mathcal{Z}_t
\]
SMPC for dynamic option hedging

- Stochastic finite-horizon optimal control problem (fixed horizon $N$):

$$\min_{\{u(k,z)\}} \text{Var}_z [w(t + N, z) - p(t + N, z)]$$

subject to

$$w(k + 1, z) = (1 + r)w(k, z) + \sum_{i=0}^{n} b_i(k, z)u_i(k, z), \quad k = t, \ldots, t + N$$

$$w(t, z) = w(t)$$
SMPC for dynamic option hedging

• Drawback: the longer the horizon $N$, the largest the number of scenarios!

• Special case: use $N=1$!

$$\min_{u(t)} \text{Var}_z [w(t+1, z) - p(t+1, z)]$$

s.t. 

$$w(t+1, z) = (1 + r)w(t) + \sum_{i=0}^{n} b_i(t, z)u_i(t)$$

✓ Only one vector $u(t)$ to optimize
(no dependence of $u$ on $z$!)

✓ No further branching, so we can generate a lot of scenarios for $z$

๏ Need to compute target wealth $p(t+1, z)$ for all $z$

A. Bemporad

“Stochastic RHC for Dynamic Option Hedging” - Siena, September 7, 2009
SMPC hedging algorithm

- Let \( t \) = current hedging date, \( w(t) \) = wealth of portfolio, \( x(t) \in \mathbb{R}^n \) = asset prices

- Use **Monte Carlo simulation** to generate \( M \) scenarios of future asset prices
  \[
  x^1(t + 1), x^2(t + 1), \ldots, x^M(t + 1)
  \]
  \[
  y^1(t + 1), y^2(t + 1), \ldots, y^M(t + 1)
  \]

- Use a **pricing engine** to generate the corresponding future option prices
  \[
  p^1(t + 1), p^2(t + 1), \ldots, p^M(t + 1)
  \]

- **Optimize** sample variance, get new asset quantities \( u(t) \in \mathbb{R}^n \), rebalance portfolio:

  \[
  \min_{u(t)} \sum_{j=1}^{M} \left( w^j(t + 1) - p^j(t + 1) - \left( \frac{1}{M} \sum_{i=1}^{M} w^i(t + 1) - p^i(t + 1) \right) \right)^2
  \]

  This is a very simple **least squares** problem with \( n \) variables!

  \( (n = \text{number of traded assets}) \)

With **transaction costs**, the problem can be rewritten as **mixed-integer quadratic program** using a **mixed-logical dynamical (MLD)** reformulation of the portfolio dynamics

(Bemporad, Morari, 1999)
Talk outline

✓ Dynamic hedging problem for financial options
✓ Dynamic hedging as a linear stochastic control problem
✓ Stochastic receding horizon control (SRHC)
✓ Pricing engine + SRHC = dynamic option hedging tool

• Simulation results

Is SMPC good for option hedging?
Example: BS model, European call

- Black-Scholes model (=log-normal)
- volatility=0.2, risk-free=0.04
- $T=24$ weeks ($\Delta t=1$ week)
- 50 simulations
- $M=100$ scenarios
- Pricing method: Monte Carlo sim.
- SMPC

- CPU time = 7.52 ms per SMPC step (Matlab R2009 on this mac)
Example: BS model, European call

- Black-Scholes model (=log-normal)
- volatility=0.2, risk-free=0.04
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- **Delta-hedging**

**Portfolio wealth vs. payoff at expiration**

- CPU time $= 0.2$ ms per SMPC step (Matlab R2009 on this mac)

**Hedging error** $e(T)=w(T)-p(T)$

SMPC and delta-hedging are almost indistinguishable
Example: BS model, European call

<table>
<thead>
<tr>
<th>TIME</th>
<th>x(t)</th>
<th>w(t)</th>
<th>p(t)</th>
<th>u0(t)</th>
<th>x(t)*u1(t)</th>
<th>x(t)*dp/dx(t)</th>
</tr>
</thead>
<tbody>
<tr>
<td>t=0.0000: S=100.000, P= 6.196, O= 6.196, P(B)=-52.151, P(S)= 58.348 (BS delta= 57.926)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.0185: S=101.367, P= 6.955, O= 6.865, P(B)=-56.091, P(S)= 63.046 (BS delta= 62.628)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.0370: S= 96.897, P= 4.134, O= 4.261, P(B)=-42.629, P(S)= 46.762 (BS delta= 46.307)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.0556: S= 94.582, P= 2.985, O= 3.108, P(B)=-35.080, P(S)= 38.065 (BS delta= 37.607)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.0741: S= 93.057, P= 2.345, O= 2.415, P(B)=-29.877, P(S)= 32.222 (BS delta= 31.771)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.0926: S= 93.371, P= 2.431, O= 2.395, P(B)=-30.200, P(S)= 32.632 (BS delta= 32.165)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.1111: S= 94.295, P= 2.732, O= 2.591, P(B)=-32.518, P(S)= 35.250 (BS delta= 34.760)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.1296: S= 88.192, P= 0.426, O= 0.859, P(B)=-14.985, P(S)= 15.411 (BS delta= 15.053)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.1481: S= 90.411, P= 0.803, O= 1.199, P(B)=-19.776, P(S)= 20.579 (BS delta= 20.147)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.1667: S= 88.586, P= 0.373, O= 0.754, P(B)=-14.236, P(S)= 14.609 (BS delta= 14.234)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.1852: S= 87.683, P= 0.214, O= 0.544, P(B)=-11.312, P(S)= 11.526 (BS delta= 11.186)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.2037: S= 90.998, P= 0.641, O= 1.000, P(B)=-18.744, P(S)= 19.385 (BS delta= 18.910)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.2222: S= 94.742, P= 1.425, O= 1.867, P(B)=-30.734, P(S)= 32.158 (BS delta= 31.555)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.2407: S= 99.890, P= 3.149, O= 3.945, P(B)=-52.320, P(S)= 55.469 (BS delta= 54.841)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.2593: S=102.720, P= 4.682, O= 5.466, P(B)=-64.736, P(S)= 69.418 (BS delta= 68.857)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.2778: S= 99.723, P= 2.609, O= 3.439, P(B)=-51.468, P(S)= 54.077 (BS delta= 53.379)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.2963: S= 99.591, P= 2.499, O= 3.147, P(B)=-50.513, P(S)= 53.012 (BS delta= 52.268)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.3148: S= 98.178, P= 1.709, O= 2.233, P(B)=-42.460, P(S)= 44.169 (BS delta= 43.336)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.3333: S=100.471, P= 2.709, O= 3.142, P(B)=-55.135, P(S)= 57.845 (BS delta= 57.034)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.3519: S=102.804, P= 4.012, O= 4.363, P(B)=-69.359, P(S)= 73.371 (BS delta= 72.719)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.3704: S= 97.457, P= 0.144, O= 1.202, P(B)=-34.892, P(S)= 35.037 (BS delta= 33.884)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.3889: S= 97.789, P= 0.238, O= 1.030, P(B)=-34.692, P(S)= 34.930 (BS delta= 33.564)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.4074: S= 98.881, P= 0.602, O= 1.089, P(B)=-41.289, P(S)= 41.891 (BS delta= 40.275)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.4259: S= 97.699, P= 0.071, O= 0.308, P(B)=-22.850, P(S)= 22.921 (BS delta= 20.300)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>t=0.4444: S= 96.002, P= -0.344, O= 0.000, P(B)= -0.344, P(S)= 0.000 (BS delta= 0.000)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Example: Heston model, European call

- Heston’s model
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free=0.04
- Pricing method: Monte Carlo sim.
- SMPC

Heston's model

$$dx_i(\tau) = (\mu_x^i \, d\tau + \sqrt{y_i(\tau)} \, dz_i^x)x_i(\tau)$$

$$dy_i(\tau) = \theta_i(k_i - y_i(\tau)) \, d\tau + \omega_i \sqrt{y_i(\tau)} \, dz_i^y$$

CPU time = 85.5 ms per SMPC step (Matlab R2009 on this mac)
Example: Heston model, European call

- Heston’s model
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free=0.04
- **Delta hedging**

- CPU time = 1.85 ms per SMPC step (Matlab R2009 on this mac)
Example: BS model, Napoleon cliquet

\[
p(T) = \max \left\{ 0, C + \min_{i \in \{1, \ldots, N_{\text{fix}}\}} \frac{x(t_i) - x(t_{i-1})}{x(t_{i-1})} \right\}
\]

- Black-Scholes model (=log-normal)
- volatility=0.2
- \(T=24\) weeks (hedging every week)
- 50 simulations
- \(M=100\) scenarios
- risk-free=0.04
- Pricing method: Monte Carlo sim.
- SMPC: only trade underlying stock

- CPU time = 1400 ms per SMPC step
  (Matlab R2009 on this mac)

\(t_i = 0, 8, 16, 24\) weeks
Example: BS model, Napoleon cliquet

- Black-Scholes model (=log-normal)
- volatility=0.2
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free=0.04
- Delta hedging, only trade underlying stock

$p(T) = \max \left\{ 0, C + \min_{i \in \{1, \ldots, N_{\text{fix}}\}} \frac{x(t_i) - x(t_{i-1})}{x(t_{i-1})} \right\}$

- CPU time = 2.41 ms per SMPC step
  (Matlab R2009 on this mac)

$t_i = 0, 8, 16, 24$ weeks
Example: BS model, Napoleon cliquet

- Black-Scholes model (=log-normal)
- volatility=0.2
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- risk-free=0.04
- Pricing method: Monte Carlo sim.
- SMPC: Trade underlying stock & European call with maturity $t+T$

$$ p(T) = \max \left\{ 0, C + \min_{i \in \{1, \ldots, N_{\text{fix}}\}} \frac{x(t_i) - x(t_{i-1})}{x(t_{i-1})} \right\} $$

- CPU time = 1625 ms per SMPC step
  (Matlab R2009 on this mac)

$t_i=0, 8, 16, 24$ weeks
Approximate option pricing

- **Bottleneck** of the approach for exotic options: price $M$ future option values $p_1(t+1), p_2(t+1), \ldots, p_M(t+1)$

- **Monte Carlo pricing** can be time consuming: say $L$ scenarios to evaluate a single option value ⇒ need to simulate $ML$ paths to build optimization problem (e.g.: $M=100, L=10000, ML=10^6$)

- Use off-line **function approximation** techniques to estimate $p(t)$ as a function of current asset parameters and other option-related parameters

  Example: Napoleon cliquet, Heston model

  $$p(t) = f(x(t), \sigma(t), x(t_1), \ldots, x(t_{N_{fix}}))$$

- Most suitable method for estimating pricing function $f$: **least-squares Monte Carlo** approach based on polynomial approximations
  
  (Longstaff, Schwartz, 2001)
Example: BS model, Napoleon cliquet

- Black-Scholes model (=log-normal)
- volatility=0.2, risk-free=0.04
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- Pricing method: LS approximation
- SMPC: only trade underlying stock

CPU time = 1400 ms per SMPC step (Matlab R2009 on this mac)

CPU time = 50.5 ms per SMPC step (Matlab R2009 on this mac)

CPU time = 76.7 s to compute LS approximation (off-line)

Hedging quality very similar!
Example: BS model, Napoleon cliquet

- Black-Scholes model (=log-normal)
- volatility=0.2, risk-free=0.04
- $T=24$ weeks (hedging every week)
- 50 simulations
- $M=100$ scenarios
- Pricing method: LS approximation
- SMPC: Trade underlying stock & European call with maturity $t+T$

CPU time = 1625 ms per SMPC step (Matlab R2009 on this mac)

CPU time = 59.2 ms per SMPC step (Matlab R2009 on this mac)

CPU time = 76.7 s to compute LS approximation (off-line)

Hedging quality very similar!
Example: Heston model, Napoleon cliquet

SMPC: only trade underlying stock

CPU time = 220 ms per SMPC step

SMPC: Trade underlying stock & European call with maturity $t+T$

CPU time = 277 ms per SMPC step

Delta hedging
only trade underlying stock
CPU time = 156 ms per SMPC step

CPU time = 156 s to compute LS approximation (off-line)
Conclusions

• Dynamic option hedging = stochastic control problem

• Propose Stochastic MPC as a very versatile tool for dynamic option hedging:
  • it’s based on Monte Carlo simulation ⇒ can use arbitrary stock models
  • can use any payoff function for which a numerically efficient pricing engine is available

• Computational demand mostly due to pricing future option values

• On-line use of SMPC: suggest trading moves to traders

• Off-line use of SMPC: run extensive simulations to quantify the average hedging error for a given market model and option type

• On-going work: test SMPC’s robustness w.r.t. market model mismatch