Dynamic Programming with Hermite Information

Kenneth L. Judd
(Hoover Institution)

Yongyang Cai
(Hoover Institution)

September 1, 2011
Derivative of Value Functions

- Maximization step in the conventional DP algorithm:

\[ V_t(x_i) = v_i = \max_{a_i \in D(x_i, t)} u_t(x_i, a_i) + \beta E\{V_{t+1}(x_{i+}) \mid x_i, a_i\}, \]

- Conventional fitting step: use the Lagrange data \{(x_i, v_i) : i = 1, \ldots, m\} to construct the approximated value function \( \hat{V}_t(x) \).

- Envelope theorem: Let

\[ V(x) = \max_y f(x, y) \]

s.t. \( g(x, y) = 0. \)

Let \( y^*(x) \) be the optimizer and \( \lambda^*(x) \) be the shadow price.

\[ \frac{\partial V}{\partial x} = \frac{\partial f}{\partial x}(x, y^*(x)) + \lambda^*(x)^\top \frac{\partial g}{\partial x}(x, y^*(x)). \]
Optimal Growth Models

- Optimal Growth Problem:

\[
V_0(k_0) = \max_{c,l} \sum_{t=0}^{T-1} \beta^t u(c_t, l_t) + \beta^T V_T(k_T),
\]

s.t. \( k_{t+1} = F(k_t, l_t) - c_t, \quad 0 \leq t < T, \)

- DP model of optimal growth problem:

\[
V_t(k) = \max_{c,l} \ u(c, l) + \beta V_{t+1}(F(k, l) - c),
\]
Multi-Stage Portfolio Optimization

- \( W_t \): wealth at stage \( t \); stocks’ random return: \( R = (R_1, \ldots, R_n) \); bond’s riskfree return: \( R_f \);
- \( S_t = (S_{t1}, \ldots, S_{tn})^\top \): money in the stocks; \( B_t = W_t - e^\top S_t \): money in the bond,
- \( W_{t+1} = R_f(W_t - e^\top S_t) + R^\top S_t \)
- Multi-Stage Portfolio Optimization Problem:

\[
V_0(W_0) = \max_{X_t, 0 \leq t < T} E\{u(W_T)\},
\]

- Bellman Equation:

\[
V_t(W) = \max_S E\{V_{t+1}(R_f(W - e^\top S) + R^\top S)\},
\]

\( W \): state variable; \( S \): control variables.
Derivative of Value Functions: Examples

- For the optimal growth DP model,
  \[ V'_t(k) = \beta V'_{t+1}(F(k, l^*) - c^*)F_k(k, l^*), \]
  where \( c^* \) and \( l^* \) are the optimal controls for the given \( k \).

- For the multi-stage portfolio optimization problem,
  \[ V'_t(W) = R_f E\{V'_{t+1}(R_f(W - e^T S^*) + R^T S^*)\}, \]
  where \( S^* \) are the optimal portfolio invested in stocks.

- It seems that we need to compute \( V'_{t+1}(F(k, l^*) - c^*) \) or \( E\{V'_{t+1}(R_f(W - e^T X^*) + R^T X^*)\} \), which are expensive. However, problem (1) has other equivalent forms. We choose a form such that \( dV(x)/dx \) can be easily computed.
Derivative of Value Functions in Optimal Growth Models

- For the optimal growth problem,

\[
V_t(k) = \max_{k^+, c, l} u(c, l) + \beta V_{t+1}(k^+),
\]

s.t. \( F(k, l) - c - k^+ = 0 \),

with \( k^+, c \) and \( l \) as control variables.

- New formula for computing \( V'_t(k) \):

\[
V'_t(k) = \lambda F_k(k, l^*),
\]

where \( \lambda \) is the shadow price for the constraint \( F(k, l) - c - k^+ = 0 \), and given directly by optimization packages.
Derivative of Value Functions in Portfolio Optimization

- For the multi-stage portfolio optimization problem,
  \[ V_t(W) = \max_{B, S} E\{V_{t+1}(R_f B + R^T S)\}, \]
  s.t. \( W - B - e^T S = 0, \)
  with the bond allocation \( B \) and the stock allocation \( S \).

- New formula for computing \( V'_t(W) \):
  \[ V'_t(W) = \lambda, \]
  where \( \lambda \) is the shadow price for the constraint \( W - B - e^T S = 0. \)
For an optimization problem,

\[ V(x) = \max_y f(x, y) \]

s.t. \( g(x, y) = 0, h(x, y) \geq 0, \)

we can modify it as

\[ V(x) = \max_{y,z} f(z, y) \]

s.t. \( g(z, y) = 0, h(z, y) \geq 0, x - z = 0, \)

by adding a trivial control variable \( z \) and a trivial constraint \( x - z = 0. \)

Then by the envelope theorem, we get

\[ V'(x) = \lambda, \]

where \( \lambda \) is the shadow price for the trivial constraint \( x - z = 0. \)
**Numerical DP Algorithm with Hermite Interpolation**

Initialization. Choose the approximation nodes, $X_t = \{x_{it} : 1 \leq i \leq m_t\}$ for every $t < T$, and choose a functional form for $\hat{V}(x; b)$. Let $\hat{V}(x; b^T) \equiv V_T(x)$.

Step 1. Maximization step. For each $x_i \in X_t$, $1 \leq i \leq m_t$, compute

$$v_i = \max_{a_i \in \mathcal{D}(y_i, t), y_i} u_t(y_i, a_i) + \beta E\{\hat{V}(x_i^+; b_t^{t+1}) | y_i, a_i\},$$

s.t. $x_i - y_i = 0$,

and $s_i = \lambda_i$, where $\lambda_i$ is the shadow price of the constraint $x_i - y_i = 0$.

Step 2. Hermite fitting step. Compute the $b^t$ such that $\hat{V}(x; b^t)$ approximates $(x_i, v_i, s_i)$ data.
If we have Hermite data \( \{(x_i, v_i, s_i) : i = 1, \ldots, m\} \) on \([a, b]\), then the following system of \(2m\) linear equations produces coefficients for degree \(2m - 1\) Chebyshev polynomial interpolation on the Hermite data:

\[
\sum_{j=0}^{2m-1} c_j T_j(z_i) = v_i, \quad i = 1, \ldots, m,
\]

\[
\frac{2}{b-a} \sum_{j=0}^{2m-1} c_j T'_j(z_i) = s_i, \quad i = 1, \ldots, m,
\]

where \(z_i = \frac{2x_i - a - b}{b - a} (i = 1, \ldots, m)\) are the Chebyshev nodes in \([-1, 1]\), and \(T_j(z)\) are Chebyshev basis polynomials.
Numerical Examples for Optimal Growth Problems

- $\beta = 0.95$; $f(k, l) = Ak^{\alpha/l^{1-\alpha}}$ with $\alpha = 0.25$, $A = (1 - \beta)/(\alpha \beta)$;

\[ u(c, l) = \frac{c^{1-\gamma}}{1-\gamma} - B \frac{l^{1+\eta}}{1+\eta} \]

with $B = (1 - \alpha)A^{1-\gamma}$. $k \in [0.2, 3]$.

- Errors for optimal consumptions at stage 0:

\[ \max_{k \in [0.2, 3]} \frac{|c_{0,DP}^*(k) - c_0^*(k)|}{1 + |c_0^*(k)|}, \]

where $c_{0,DP}^*$ is the optimal consumption at stage 0 computed by numerical DP algorithms, and $c_0^*$ is the optimal consumption directly computed by SNOPT in GAMS code on the model (1).
Errors of optimal solutions of numerical DP algorithms with Chebyshev interpolation on $m$ Chebyshev nodes using with Lagrange vs. Hermite data

<table>
<thead>
<tr>
<th>$\gamma$</th>
<th>$\eta$</th>
<th>$m$</th>
<th>error of $c_0^*$</th>
<th>error of $l_0^*$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td>Lagrange</td>
<td>Hermite</td>
</tr>
<tr>
<td>0.5</td>
<td>0.1</td>
<td>5</td>
<td>1.1(−1)</td>
<td>1.2(−2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>6.8(−3)</td>
<td>3.1(−5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>2.3(−5)</td>
<td>1.5(−6)</td>
</tr>
<tr>
<td>0.5</td>
<td>1</td>
<td>5</td>
<td>1.4(−1)</td>
<td>1.4(−2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>7.7(−3)</td>
<td>3.7(−5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>2.6(−5)</td>
<td>6.5(−6)</td>
</tr>
<tr>
<td>2</td>
<td>0.1</td>
<td>5</td>
<td>5.5(−2)</td>
<td>6.1(−3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>3.5(−3)</td>
<td>2.1(−5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>1.6(−5)</td>
<td>1.4(−6)</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>5</td>
<td>9.4(−2)</td>
<td>1.1(−2)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>5.7(−3)</td>
<td>3.9(−5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>2.8(−5)</td>
<td>4.7(−6)</td>
</tr>
<tr>
<td>8</td>
<td>0.1</td>
<td>5</td>
<td>2.0(−2)</td>
<td>2.2(−3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>1.2(−3)</td>
<td>8.5(−6)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>6.1(−6)</td>
<td>1.0(−6)</td>
</tr>
<tr>
<td>8</td>
<td>1</td>
<td>5</td>
<td>6.6(−2)</td>
<td>7.2(−3)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>10</td>
<td>3.0(−3)</td>
<td>2.6(−5)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>20</td>
<td>2.0(−5)</td>
<td>0.0(−7)</td>
</tr>
</tbody>
</table>

Note: $a(k)$ means $a \times 10^k$.  

Kenneth L. Judd (Hoover Institution) Yongyang Cai (Hoover Institution)  
Dynamic Programming with Hermite Information
Exact optimal bond allocation

Kenneth L. Judd (Hoover Institution) Yongyang Cai (Hoover Institution)
Dynamic Programming with Hermite Information
Relative Errors of Optimal Stock Allocations from Numerical DP

Wealth, $t=1$

$log_{10}(errors of S_1)$

Wealth, $t=2$

$log_{10}(errors of S_2)$

Wealth, $t=3$

$log_{10}(errors of S_3)$

Wealth, $t=4$

$log_{10}(errors of S_4)$

Wealth, $t=5$

$log_{10}(errors of S_5)$

Kenneth L. Judd (Hoover Institution) Yongyang Cai (Hoover Institution)

Dynamic Programming with Hermite Information