A Dynamic Programming Approach to Learning-By-Doing

Charles Ng
Stanford University

Thesis advisor: Alan Manne

June 24, 2003
International Energy Workshop, IIASA

Learning-by-Doing (LBD)

- Definition: The learning process that comes from implementing a technology.

- Fundamental assumption: Efficiency comes with experience. The more we do, the better we get at it.

- Efficiency shows up as savings in cost.
Modeling LBD variables and cost function

- $X_{j,t}$ – Level of production of technology $j$ in time period $t$
- $Y_{j,t}$ – Cumulative experience for technology $j$ by time period $t$

\[ Y_{j,t} = a_{cc} + \sum_{\tau=1}^{k} X_{j,\tau} \]

- $a_{cc}$ – initial experience

- Cost function

\[ C_{j,t} = \left( s_{c} + i_{ln} \right) \left[ \frac{Y_{j,t}}{a_{cc}^r} \right] X_{j,t} \]

- $s_{c}$ – static portion of cost
- $i_{ln}$ – initial learning cost (for non-LBD technologies, $i_{ln} = 0$)
- $r_{ln}$ – learning exponent < 0

Constraints

- Supply-demand
  - Total production equals total demand

- Expansion and Decline
  - Upper and lower bounds on production in period $t$ depend on the demand and production in period $t-1$

- Carbon limits
  - Cumulative carbon emissions at the time horizon cannot be higher than a pre-determined level
An example
(based on Manne & Barreto, 2002)

- 1 Region – the world
- 2 technologies (defender and challenger)
- Challenger technology is LBD, defender is not
- Challenger is carbon-free, defender is not
- Costs are not time dependent
- All demand at 2000 is satisfied by defender
- Time horizon at 2050, time periods by decades
- We only consider electric energy
- Exogenous demands for energy
- Cumulative carbon constraint at the time horizon
- Future costs discounted by factor $\beta = (1/1.05)^{10}$

Problem

- The cost function is non-linear, non-convex

- Global optimality is not guaranteed by conventional nonlinear programming solvers (CONOPT, MINOS)

- Therefore, solving models with LBD is not a trivial task
Heuristics and other approaches

- Use different starting points
- Force the entry of LBD technologies by setting lower bounds on cumulative experience at $T$
- Approximate the non-convex cost curve as piecewise linear and solve as a mixed integer program
- Use global optimization algorithms like BARON
- Use dynamic programming

Dynamic Programming Approach – The cost-to-go function

- Let $COST_t(prevprod, experience)$ be the minimum present value of all costs from time $t$ to the time horizon, given the values of challenger production in time period $t-1$ and accumulated experience.

- This is known as the cost-to-go function, and it is a central concept in dynamic programming.
Dynamic Programming Approach –
First calculate the last time period cost

- At the last time period $T$, if we know the production levels in $T-1$, and cumulative experience in $T-1$, we can calculate the optimal cost and optimal production easily.

- We do this calculation for a pre-determined number of \{prevprod, experience\} combinations

- Using MATLAB and linear interpolation, we get a cost function $\text{COST}_T(\text{prevprod, experience})$ for the cost in the last time period

\[ \text{COST}_T(\text{prevprod, experience}) \]

Optimal cost in the last period ($T=5$) given $\text{prevprod}$ and $\text{experience}$
Dynamic Programming Approach – Calculate $COST(\text{prevprod, experience})$ recursively

- To calculate $COST$ for other time periods, we use a recursive formula.

- Having calculated the function $COST_{t+1}(\text{prevprod, experience})$, we take a step back and calculate $COST_t(\text{prevprod, experience})$.

- We want to find the production level that minimizes cost of production today + discounted cost of production in the future. Future cost of production can be determined by the cost-to-go function $COST_{t+1}(\text{prevprod, experience})$.

Dynamic Programming Approach – Recursive formula

\[
COST_t(\text{pprod,exp}) = \min_{x \in A(\text{pprod,} t)} \left\{ \text{sc}_{\text{def}} \cdot (d_{\text{em}, t} - x) + (\text{sc}_{\text{ch}} + \text{inlc}_{\text{ch}} \cdot \left( \frac{\exp}{\text{acc}_{\text{ch}}} \right)^{\ln} )x + \beta \cdot COST_{t+1}(x, \text{exp} + x) \right\}
\]
Example parameters

<table>
<thead>
<tr>
<th>Time period</th>
<th>2010</th>
<th>2020</th>
<th>2030</th>
<th>2040</th>
<th>2050</th>
</tr>
</thead>
<tbody>
<tr>
<td>Demand (tkwh)</td>
<td>19</td>
<td>24</td>
<td>31</td>
<td>42</td>
<td>53</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>scost_{ld}</td>
<td>scost_{ld}</td>
</tr>
<tr>
<td>Case I (low learning costs)</td>
<td>40</td>
</tr>
<tr>
<td>Case II (high learning costs)</td>
<td>40</td>
</tr>
</tbody>
</table>

Example results (without carbon limit)

Optimal challenger production using CONOPT

<table>
<thead>
<tr>
<th>Time period</th>
<th>2010</th>
<th>2020</th>
<th>2030</th>
<th>2040</th>
<th>2050</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case I (low learning costs)</td>
<td>1.90</td>
<td>10.00</td>
<td>20.58</td>
<td>34.25</td>
<td>47.23</td>
</tr>
<tr>
<td>Case II (high learning costs)</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

Optimal challenger production using DP with linear interpolation (grid size 20 x 30)

<table>
<thead>
<tr>
<th>Time period</th>
<th>2010</th>
<th>2020</th>
<th>2030</th>
<th>2040</th>
<th>2050</th>
</tr>
</thead>
<tbody>
<tr>
<td>Case I (low learning costs)</td>
<td>1.90</td>
<td>9.84</td>
<td>20.46</td>
<td>34.02</td>
<td>46.64</td>
</tr>
<tr>
<td>Case II (high learning costs)</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Example results (carbon limit = 120 billion tons)

Comparison between DP and CONOPT results for Case II

<table>
<thead>
<tr>
<th></th>
<th>2010</th>
<th>2020</th>
<th>2030</th>
<th>2040</th>
<th>2050</th>
</tr>
</thead>
<tbody>
<tr>
<td>DP</td>
<td>1.90</td>
<td>6.72</td>
<td>17.05</td>
<td>31.50</td>
<td>41.34</td>
</tr>
<tr>
<td>CONOPT</td>
<td>0.59</td>
<td>4.76</td>
<td>16.68</td>
<td>31.35</td>
<td>45.07</td>
</tr>
</tbody>
</table>

- We get larger discrepancies due to interpolating between real numbers and “infinity”
- Increasing the grid size will result in a closer solution
- We can also refine the solution by using the DP solution as a starting point for CONOPT

Conclusions and further work

- The dynamic programming approach looks promising
  - Theoretical running time is non-polynomial (like BARON and MIPs)
  - Allows us to substitute between resolution and running time.
  - Allows us to include nonlinearities other than the LBD cost curve
- A more sophisticated model
  - More than two technologies
  - Other constraints?
- Robustness –
  - How to deal with interpolation with “infinity”?