A Dynamic Programming Approach to Learning-By-Doing

Charles Ng
Stanford University

Thesis advisor: Alan Manne

June 24, 2003
International Energy Workshop, IIASA

1

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.

2

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

$$Y_{j,t} = acc_j + \sum_{\tau=1}^{t-1} X_{j,\tau}$$

  - acc – initial experience
- Cost function

$$C_{j,t} = \left(sc_j + inlc_j \left[\frac{Y_{j,t}}{acc_j} \right]^{lrn_j} \right) X_{j,t}$$

  - sc – static portion of cost
  - inlc – initial learning cost (for non-LBD technologies, inlc = 0)
  - lrn – learning exponent < 0

3

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

4

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 $\beta = (1/1.05)^{10}$

5

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

6

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.

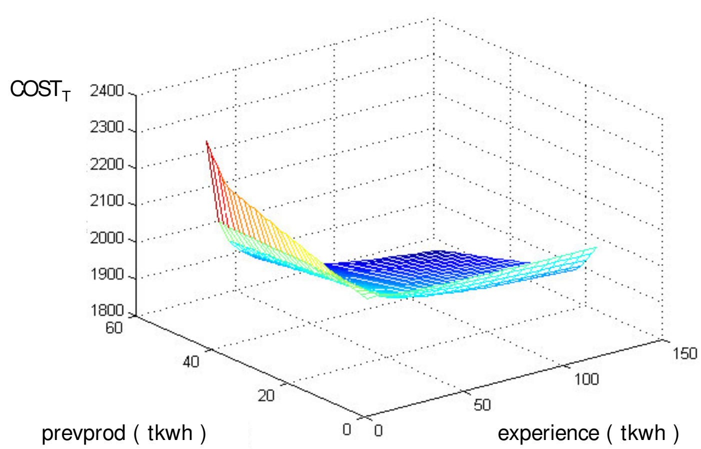
- This is known as the cost-to-go function, and it is a central concept in dynamic programming.

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 $COST_T$(*prevprod, experience*) for the cost in the last time period

9

$COST_T$(*prevprod, experience*)
Optimal cost in the last period (T=5) given *prevprod* and *experience*



10

Dynamic Programming Approach –

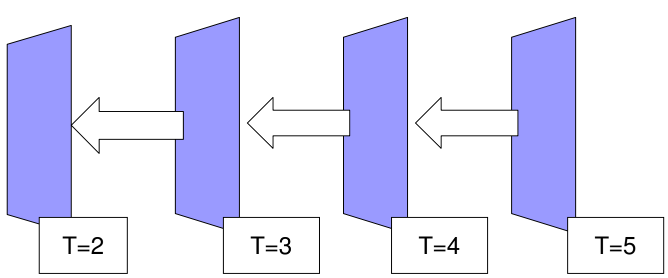
Calculate COST(*prevprod, experience*) recursively

- To calculate COST for other time periods, we use a recursive formula.
- Having calculated the function $COST_{t+1}$(*prevprod, experience*), we take a step back and calculate $COST_t$(*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 $COST_{t+1}$(*prevprod, experience*).

Dynamic Programming Approach –

Recursive formula

$$COST_t(pprod, exp) = \min_{x \in A(pprod,t)} \{ sc_{def} \cdot (dem_t - x) + \left(sc_{chl} + inlc_{chl} \cdot \left(\frac{\exp}{acc_{chl}}\right)^{-lrn}\right)x + \beta \cdot COST_{t+1}(x, \exp + x) \}$$



Example

parameters

Time period	2010	2020	2030	2040	2050
Demand (tkwh)	19	24	31	42	53

	$scost_{def}$	$scost_{lbd}$	$lcost_{lbd}$	lrn
Case I(low learning costs)	40	30	15	-0.2
Case II (high learning costs)	40	30	50	-0.2

Parameter	Value
Decline factor	$(1/1.03)^{10}$
Expansion factor	4
Demand factor (expansion)	0.1

13

Example

results (without carbon limit)

Optimal challenger production using CONOPT

	2010	2020	2030	2040	2050
Case I (low learning costs)	1.90	10.00	20.58	34.25	47.23
Case II (high learning costs)	0	0	0	0	0

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

	2010	2020	2030	2040	2050
Case I(low learning costs)	1.90	9.84	20.46	34.02	46.64
Case II(high learning costs)	0	0	0	0	0

14

Example

results (carbon limit = 120 billion tons)

Comparison between DP and CONOPT results for Case II

	2010	2020	2030	2040	2050
DP	1.90	6.72	17.05	31.50	41.34
CONOPT	0.59	4.76	16.68	31.35	45.07

-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

15

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"?

16