Introduction to Markov Decision processes (dynamic programming)
Anders Ringgaard Kristensen

Outline
- The history
- A simple example of dynamic programming in a Markov decision process
- Areas of potential applications
- Criteria of optimality
- Traditional optimization methods
  - Value iteration ("dynamic programming")
  - Policy iteration
- Future profitability

Historical development
- A.A. Markov: Linked chains of events
- Bellman (1957): Dynamic programming for sequential decision problems
- Howard (1960): Markov decision processes
- Jenkins and Halter (1963): First application to animal production
- Johnston (1965): A new religion
- Numerous applications in herd management!

Example of a Markov decision process (a cow)

Stage 1 Stage 2 Stage 3
Stage length e.g. 1 lactation cycle
At the beginning of each stage, the state, $i$, of the cow is observed:
- $i=1$: Low milk yield
- $i=2$: Average milk yield
- $i=3$: High milk yield
The state is in this example defined by the value of only one state variable (trait)

Having observed the state $i$, an action, $d$, is taken:
- $d=1$: Keep the cow
- $d=2$: Replace the cow at the end of the stage

Rewards
Depending on state $i$ and action $d$, a reward $r_{i}^{d}$ is gained

<table>
<thead>
<tr>
<th>$i$</th>
<th>$d=1$ (Keep)</th>
<th>$d=2$ (Replace)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i=1$  (Low yield)</td>
<td>10,000</td>
<td>9,000</td>
</tr>
<tr>
<td>$i=2$  (Average yield)</td>
<td>12,000</td>
<td>11,000</td>
</tr>
<tr>
<td>$i=3$  (High yield)</td>
<td>14,000</td>
<td>13,000</td>
</tr>
</tbody>
</table>

Output
Depending on state $i$ and action $d$ a physical output $m^{d}$ (in this case milk yield) is involved.

<table>
<thead>
<tr>
<th>$i$</th>
<th>$d=1$ (Keep)</th>
<th>$d=2$ (Replace)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$i=1$  (Low yield)</td>
<td>5,000</td>
<td>5,000</td>
</tr>
<tr>
<td>$i=2$  (Average yield)</td>
<td>6,000</td>
<td>6,000</td>
</tr>
<tr>
<td>$i=3$  (High yield)</td>
<td>7,000</td>
<td>7,000</td>
</tr>
</tbody>
</table>
Transition probabilities

Transitions between states are governed by transition probabilities $p^d_j$.

<table>
<thead>
<tr>
<th>$p^d_j$</th>
<th>$d=1$ (Keep)</th>
<th>$d=2$ (Replace)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$j=1$</td>
<td>0.6</td>
<td>0.3</td>
</tr>
<tr>
<td>$j=2$</td>
<td>0.2</td>
<td>0.6</td>
</tr>
<tr>
<td>$j=3$</td>
<td>0.1</td>
<td>0.3</td>
</tr>
</tbody>
</table>

A policy $s$ is a map (rule) assigning to each state an action. An example of a policy for this model is to replace if $j=1$ and keep if $j>1$. Thus, in functional notation: $s(1)=2$ ("Replace"), and $s(2)=s(3)=1$ ("Keep").

**Problem**: To determine an optimal policy.

Discount factor

To account for time preferences, a discount factor $\beta$, $0 < \beta < 1$, is used. If stage length is $l$ and the interest rate is $r$, the discount factor is calculated as $\beta = \exp(-rl)$.

Herd management decisions

Dynamic (no well defined time horizon)

Constraints

Random (biological) variation

Production cycles

Sequential and recurrent

Uncertainty about the "true" state of the animals

Partial solution

Markov decision processes

- Handle
  - Dynamics
  - Random variation
  - Cyclic production
  - Recurrence
  - (Constraints)
- Do not handle
  - Uncertainty about true state
  - (Constraints)

Examples of decisions

Replacement/culling:
- Breeding animals: dairy cows, sows, ewes
- Slaughter pigs, bull calves

Mating/insuination

Medical treatment

Feeding level

Essential for the benefit

Few well-defined decisions

Recurrent decisions

Dynamics


### Criteria of optimality: Finite time horizon

Finite time horizon
- Sum of rewards over stages
- Sum of discounted rewards over stages

Let the random variable $R(n, s^n)$ be the (unknown) reward at stage $n$ under the policy $s^n$ for the stage.
- If the state observed at stage $n$ turns out to be $i$, and $s^n = s$, we have $R(n, s^n) = r^n_i$

$$E[\sum_i R(n, s^n)] = E[\sum_i \beta^{n-1} R(n, s^n)]$$

### State transitions over stages

Let $\pi_n$ be the state distribution at stage $n$. The expected state distribution at stage $n+1$ is then (under the policy $\pi$):
- $\pi_{n+1} = \pi_n P$

As an example, assume that
- We now have a low yielding cow, i.e., $\pi = (1, 0, 0)$
- The policy $\pi$ is to keep in all stages

Then $\pi = \{0.6, 0.3, 0.1\}$

And

$$\pi_{n+1} = \pi P = \begin{pmatrix} 0.6 & 0.3 & 0.1 \\ 0.1 & 0.3 & 0.6 \\ 0.3 & 0.6 & 0.1 \end{pmatrix} \begin{pmatrix} 0.6 \\ 0.3 \\ 0.1 \end{pmatrix} = \begin{pmatrix} 0.56 \\ 0.42 \\ 0.02 \end{pmatrix}$$

It is rather obvious that
- $\pi_{n+1} = \pi_n P$
- $\pi_{n+2} = \pi_n P^2$
- $\pi_{n+k} = \pi_n P^k$

Thus we can calculate the expected distribution as many stages forward as we like.

### Criteria of optimality: Infinite time horizon

Under infinite time horizon we can’t just sum the rewards ($\sum \to \infty$).

Possible alternative criteria
- Sum of discounted rewards
- Average rewards $s = \sum_i r_i s_i$
- Average rewards over physical output $g^s = \frac{\sum_i r_i s_i}{\sum_i s_i}$
- The symbol $s_i$ is the limiting (or steady) state probability.

### Optimization methods

Policy iteration
- Exact under infinite time horizon
- Solution of a set of linear equations (size of state space)
- Efficient (only very few iterations)

Value iteration “Dynamic programming”
- Backwards recursion using the functional equations
- Slow (many iterations needed), approximate

$$f(n) = \max_{\pi} \{ r^n + B\pi f(n+1) \}$$

(Linear programming)

### Value iteration, example

<table>
<thead>
<tr>
<th>Value</th>
<th>Low</th>
<th>Average</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 stages left:</td>
<td>$f(0) = \max r^n_i$</td>
<td>$f(0) = \max { r^n_i + \beta \Sigma p_i f(0) }$</td>
<td>$f(0) = \max { r^n_i + \beta \Sigma p_i f(n-1) }$</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Value</th>
<th>Low</th>
<th>Average</th>
<th>High</th>
</tr>
</thead>
<tbody>
<tr>
<td>2 stages left</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1 stage left</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0 stages left</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

$$f(n) = \max_{\pi} \{ r^n + B\pi f(n+1) \}$$
Policy iteration

1. Choose an arbitrary policy \( s \).
2. Solve the set of simultaneous equations in the table
3. For each state \( i \), find the action \( d' \) that maximizes the expression in the table and put \( s'(i) = d' \). If \( s' = s \) then stop, since \( s' \) is then optimal. Otherwise put \( s = s' \) and go back to 2.

Table 13.6: Equations and expressions to be used in the policy iteration cycle with different objective functions.

<table>
<thead>
<tr>
<th>Obj.</th>
<th>Linear equations of Step 2</th>
<th>Expression</th>
<th>Add. eq.</th>
<th>Step 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>13.3</td>
<td>[ f_i = r_i + \beta \sum_j p_{ij} f_j ]</td>
<td>[ f_i = r_i + \beta \sum_j p_{ij} f_j ]</td>
<td>[ f_i = 0 ]</td>
<td>13.3</td>
</tr>
<tr>
<td>13.4</td>
<td>[ g_i = r_i + \beta \sum_j p_{ij} g_j ]</td>
<td>[ g_i = r_i + \beta \sum_j p_{ij} g_j ]</td>
<td>[ g_i = 0 ]</td>
<td>13.4</td>
</tr>
<tr>
<td>13.5</td>
<td>[ \beta g_i^2 + f_i = r_i + \beta \sum_j p_{ij} f_j^2 ]</td>
<td>[ \beta g_i^2 + f_i = r_i + \beta \sum_j p_{ij} f_j^2 ]</td>
<td>[ f_i = 0 ]</td>
<td>13.5</td>
</tr>
</tbody>
</table>

Future profitability

Problem:
- The optimal policy tells me to keep this cow in state \( i \), but I don't like it, because it's ugly.
- What will it cost me to get rid of it?

Future profitability \( = i + \beta \sum_j p_{ij} f_j - i + \beta \sum_j p_{ij} f_j^2 \)

Policy iteration, example

Choose arbitrary policy \( s \) (e.g. "Keep" for small and average milk yield and "Replace" for big) so that \( s = (1, 1, 2) \).

Solve the equations

\[
\begin{bmatrix}
  f_1 \\
  f_2 \\
  f_3 \\
\end{bmatrix}
= \begin{bmatrix}
  0.6 & 0.3 & 0.1 \\
  0.6 & 0.3 & 0.1 \\
  0.6 & 0.3 & 0.1 \\
\end{bmatrix}
\begin{bmatrix}
  f_1 \\
  f_2 \\
  f_3 \\
\end{bmatrix}
\]

Determine a new policy state by state like an iteration step in value iteration.

Exercise I

Continue the calculations on slide 17 in order to determine:
- \( f_j(1) \) and \( f_j(2) \)
- Optimal actions for States 2 and 3 when there is one stage left.
- \( f_j(2) \) and \( f_j(2) \)
- Optimal actions for all 3 states when there are two stages left.

Find a matrix expression for the vector \( f \) on slide 20.

Exercise II

Under the optimal policy \( s = ("Replace", "Keep", "Keep") \), the vector of present values is

\[
\begin{bmatrix}
  f_1 \\
  f_2 \\
  f_3 \\
\end{bmatrix}
= \begin{bmatrix}
  2.40018 \\
  2.40018 \\
  2.40018 \\
\end{bmatrix}
\]

What is the interpretation of the elements?
Calculate the future profitability for each of the 3 states as explained on slide 21.