EE365: Dynamic Programming

Optimal value function and dynamic programming

Proof of optimality

Examples

Dynamic programming for modified information pattern

Dynamic programming for modified information pattern II

Outline

Markov decision problem

- dynamics: $x_{t+1} = f_t(x_t, u_t, w_t)$
- $x_0, w_0, \ldots, w_{T-1}$ independent, with known distributions
- state feedback policy: $u_t = \mu_t(x_t)$
- we consider deterministic cost for simplicity:

$$J = \mathbf{E}\left(\sum_{t=0}^{T-1} g_t(x_t, u_t) + g_T(x_T)\right)$$

- find policy $\mu = (\mu_0, \ldots, \mu_{T-1})$ that minimizes J
- data:
  - dynamics functions $f_0, \ldots, f_{T-1}$
  - stage cost functions $g_0, \ldots, g_{T-1}$ and terminal cost g_T
  - distributions of $x_0, w_0, \ldots, w_{T-1}$

Optimal value function

- define

$$V_t^\star(x) = \min_{\mu_t, \mu_{t+1}, \ldots, \mu_{T-1}} \mathbf{E} \left(\sum_{\tau=t}^{T-1} g_\tau(x_\tau, u_\tau) + g_T(x_T) \middle| x_t = x \right)$$

- minimization is over policies $\mu_t, \ldots, \mu_{T-1}$; $x_{t+1} = f_t(x_t, u_t, w_t)$
- since $x_t = x$ is known, we can just as well minimize over action u_t and policies $\mu_{t+1}, \ldots, \mu_{T-1}$.
- $V_t^\star(x)$ is expected cost-to-go, using an optimal policy, if you are in state x at time t
- $J^\star = \sum_x \pi_0(x) V_0^\star(x) = \pi_0 V_0^\star$
- $V_t^\star$ also called Bellman value function, optimal cost-to-go function

Optimal policy

- the policy

$$\mu_t^\star(x) \in \underset{u}{\operatorname{argmin}} \left(g_t(x, u) + \mathbf{E}\, V_{t+1}^\star(f_t(x, u, w_t)) \right)$$

  is optimal (we'll show this later)

- expectation is over w_t
- can choose any minimizer when minimizer is not unique
- there can be optimal policies not of the form above
- looks circular and useless: need to know optimal policy to find $V_t^\star$ (we'll see later this is not correct)

Interpretation

$$\mu_t^\star(x) \in \underset{u}{\operatorname{argmin}} \left(g_t(x, u) + \mathbf{E}\, V_{t+1}^\star(f_t(x, u, w_t)) \right)$$

assuming you are in state x at time t, and take action u

- $g_t(x, u)$ (a number) is the current stage cost you pay
- $V_{t+1}^\star(f_t(x, u, w_t))$ (a random variable) is the cost-to-go from where you land, if you follow an optimal policy for $t + 1, \ldots, T - 1$
- $\mathbf{E}\, V_{t+1}^\star(f_t(x, u, w_t))$ (a number) is the expected cost-to-go from where you land

optimal action is to minimize sum of current stage cost and expected cost-to-go from where you land

Greedy policy

- greedy policy is $\mu_t^{\text{gr}}(x) \in \text{argmin}_u\, g_t(x,u)$
- at any state, minimizes current stage cost without regard for effect of current action on future states
- in optimal policy

$$\mu_t^\star(x) \in \underset{u}{\text{argmin}}\left(g_t(x,u) + \mathbf{E}\, V_{t+1}^\star(f_t(x,u,w_t))\right)$$

second term summarizes effect of current action on future states

Dynamic programming

- define $V_T^\star(x) := g_T(x)$
- for $t = T-1, \ldots, 0$,
  - find optimal policy for time t in terms of $V_{t+1}^\star$:

    $$\mu_t^\star(x) \in \operatorname*{argmin}_u \left(g_t(x,u) + \mathbf{E}\, V_{t+1}^\star(f_t(x,u,w_t)) \right)$$

  - find $V_t^\star$ using $\mu_t^\star$:

    $$V_t^\star(x) := g_t(x, \mu_t^\star(x)) + \mathbf{E}\, V_{t+1}^\star(f_t(x, \mu_t^\star(x), w_t))$$

- a recursion that runs backward in time
- complexity is $T|\mathcal{X}||\mathcal{U}||\mathcal{W}|$ operations (fewer when P is sparse)

Variations

- random costs:

$$\mu_t^\star(x) \in \operatorname{argmin}_u \mathbf{E}\left(g_t(x, u, w_t) + V_{t+1}^\star(f_t(x, u, w_t))\right)$$
$$V_t^\star(x) := \mathbf{E}\, g_t(x, \mu_t^\star(x), w_t) + \mathbf{E}\, V_{t+1}^\star(f_t(x, \mu_t^\star(x), w_t))$$

- state-action separable cost $g_t(x, u) = q_t(x) + r_t(u)$:

$$\mu_t^\star(x) \in \operatorname{argmin}_u \left(r_t(u) + \mathbf{E}\, V_{t+1}^\star(f_t(x, u, w_t))\right)$$
$$V_t^\star(x) := q_t(x) + r_t(\mu_t^\star(x)) + \mathbf{E}\, V_{t+1}^\star(f_t(x, \mu_t^\star(x), w_t))$$

- deterministic system:

$$\mu_t^\star(x) \in \operatorname{argmin}_u \left(g_t(x, u) + V_{t+1}^\star(f_t(x, u))\right)$$
$$V_t^\star(x) := g_t(x, \mu_t^\star(x)) + V_{t+1}^\star(f_t(x, \mu_t^\star(x)))$$

Outline

Bellman operator

- deterministic cost case for simplicity
- define Bellman or dynamic programming operator $\mathcal{T}_t$ as

$$\mathcal{T}_t(h)(x) = \min_u \left(g_t(x,u) + \mathbf{E}\, h(f_t(x,u,w_t))\right)$$

  for any $h : \mathcal{X} \to \mathbf{R}$ (expectation is over w_t)

- then we have $V_T^\star = g_T$, and

$$V_t^\star = \mathcal{T}_t(V_{t+1}^\star), \quad t = T-1,\ldots,0$$

- for policy $\mu_t^\star$ we have

$$V_t^\star(x) = g_t(x,\mu_t^\star(x)) + \mathbf{E}\, V_{t+1}^\star(f_t(x,\mu_t^\star(x),w_t)), \quad t = T-1,\ldots,0$$

- this is value iteration for evaluating $J^\star$, so $J^\star = \pi_0 V_0^\star$

Monotonicity of Bellman operator

- Bellman operator is monotone:

$$h \le \tilde{h} \implies \mathcal{T}_t(h) \le \mathcal{T}_t(\tilde{h})$$

(inequalities mean for all x)

- to see this, assume $h \le \tilde{h}$; note that for any x and u,

$$g_t(x,u) + \mathbf{E}\, h(f_t(x,u,w_t)) \le g_t(x,u) + \mathbf{E}\, \tilde{h}(f_t(x,u,w_t))$$

(by monotonicity of expectation)

- minimizing each side over u (and using monotonicity of minimization)

$$\mathcal{T}_t(h)(x) \le \mathcal{T}_t(\tilde{h})(x)$$

Proof of optimality

- let μ be any policy, with cost J^μ, and value functions V_t^μ
- we will show that $J^\mu \geq J^\star$, which shows $\mu^\star$ is optimal

- for any $h : \mathcal{X} \to \mathbf{R}$, we have

$$g_t(x, \mu_t(x)) + \mathbf{E}\, h(f_t(x, \mu_t(x), w_t)) \geq \mathcal{T}_t(h)(x)$$

  since RHS minimizes LHS over all choices of $u = \mu_t(x)$

- value functions with policy μ satisfy $V_T^\mu = g_T$ and

$$\begin{aligned} V_t^\mu(x) &= g_t(x, \mu_t(x)) + \mathbf{E}\, V_{t+1}^\mu(f_t(x, \mu_t(x), w_t)) \\ &\geq \mathcal{T}_t(V_{t+1}^\mu)(x) \end{aligned}$$

Proof of optimality

- using $V_t^\star = \mathcal{T}_t(V_{t+1}^\star)$, $V_t^\mu \geq \mathcal{T}_t(V_{t+1}^\mu)$, and $V_T^\star = V_T^\mu = g_T$,

$$\begin{aligned} V_t^\mu &\geq \mathcal{T}_t(V_{t+1}^\mu) \\ &\geq \mathcal{T}_t\mathcal{T}_{t+1}(V_{t+2}^\mu) \\ &\vdots \\ &\geq \mathcal{T}_t\mathcal{T}_{t+1}\cdots\mathcal{T}_{T-1}(V_T^\mu) \\ &= \mathcal{T}_t\mathcal{T}_{t+1}\cdots\mathcal{T}_{T-1}(g_T) \\ &= V_t^\star \end{aligned}$$

- and so $J^\mu = \pi_0 V_0^\mu \geq \pi_0 V_0^\star = J^\star$

Summary

- any policy defined by dynamic programming is optimal
- (can replace 'any' with 'the' when the argmins are unique)
- $V_t^\star$ is minimal for any t, over all policies (*i.e.*, $V_t^\star \leq V_t^\mu$)
- there can be other optimal (but pathological) policies; for example we can set $\mu_0(x)$ to be anything you like, provided $\pi_0(x) = 0$

Outline

Example: Inventory model

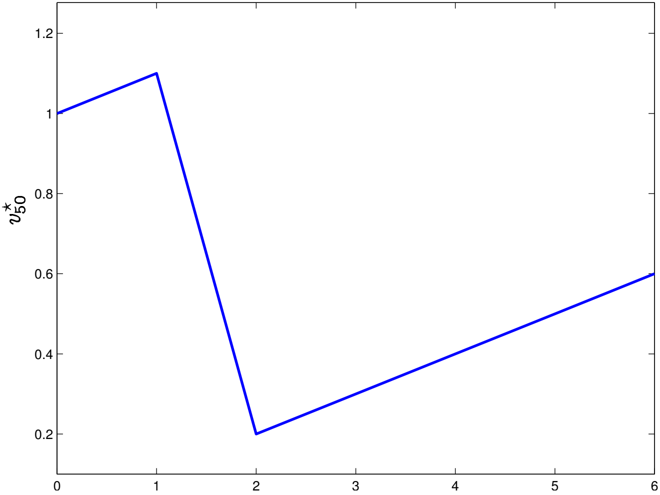
(our old friend) the inventory model

- $x_t \in \{0, 1, \ldots, 6\}$; $x_0 = 6$
- $x_{t+1} = x_t - d_t + u_t$
- $\mathbf{Prob}(d_t = 0, 1, 2) = (0.7, 0.2, 0.1)$
- $g_t(x, u) = sx + o1_{u>0}$, $s = 0.1$, $o = 1$
- add constraints $2 - x_t \leq u_t \leq 6 - x_t$ (so $x_{t+1} \in \{0, 1, \ldots, 6\}$ for any d_t)

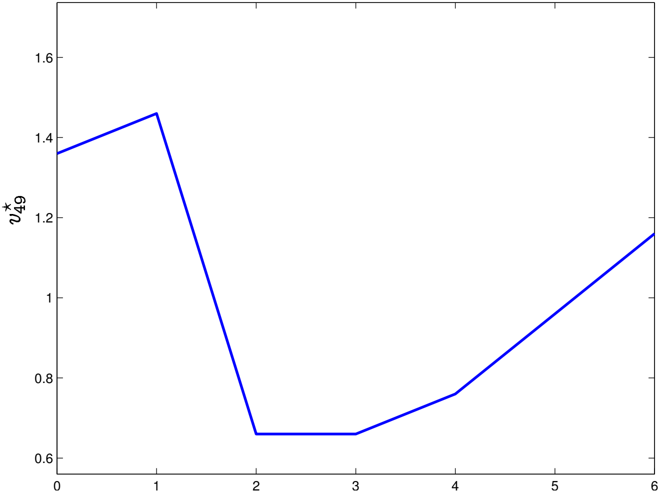
- recall heuristic policy: refill if $x_t \leq 1$

$$\mu(x) = \begin{cases} 6 - x & x = 0 \text{ or } 1 \\ 0 & \text{otherwise} \end{cases}$$

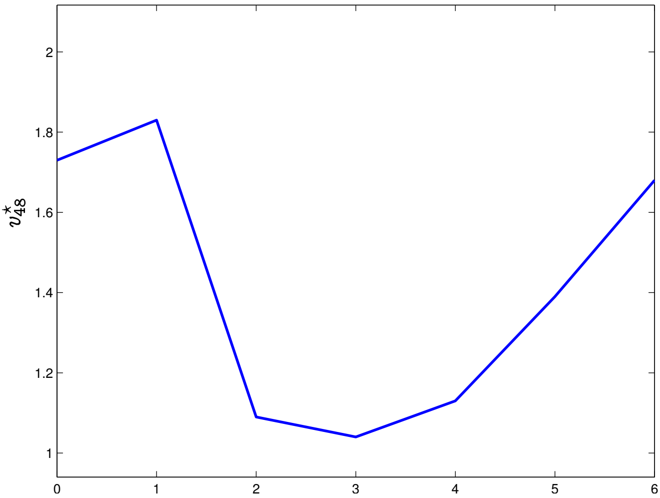
Example: Inventory model



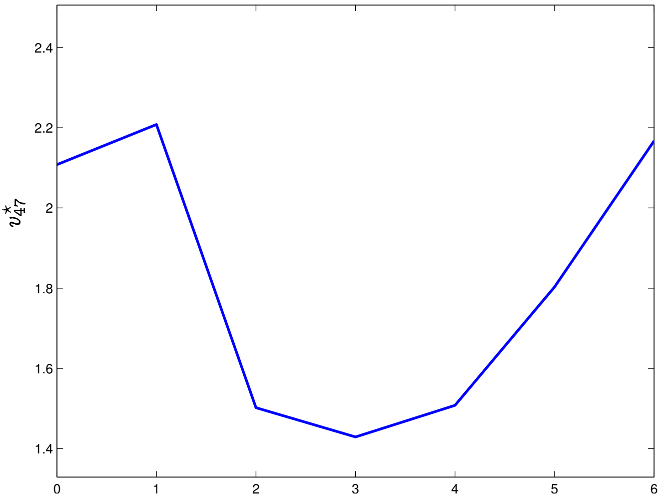
Example: Inventory model



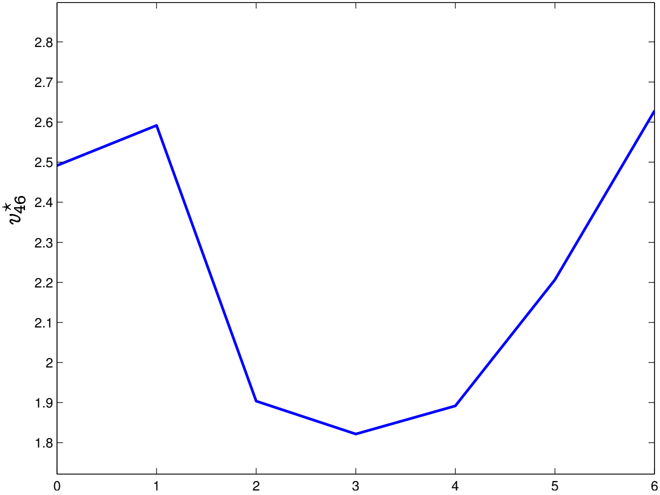
Example: Inventory model



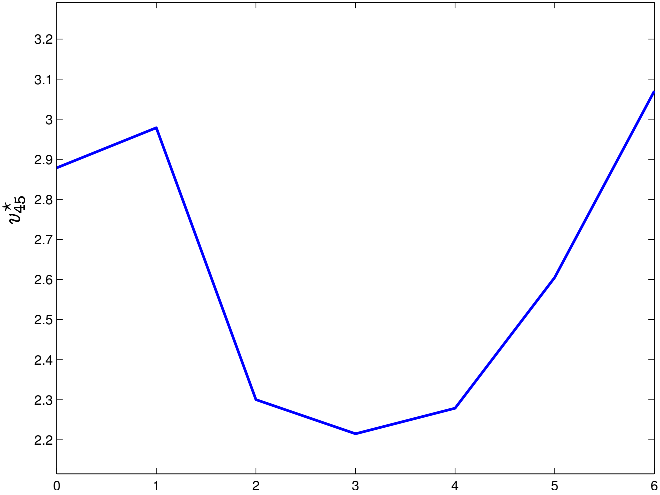
Example: Inventory model



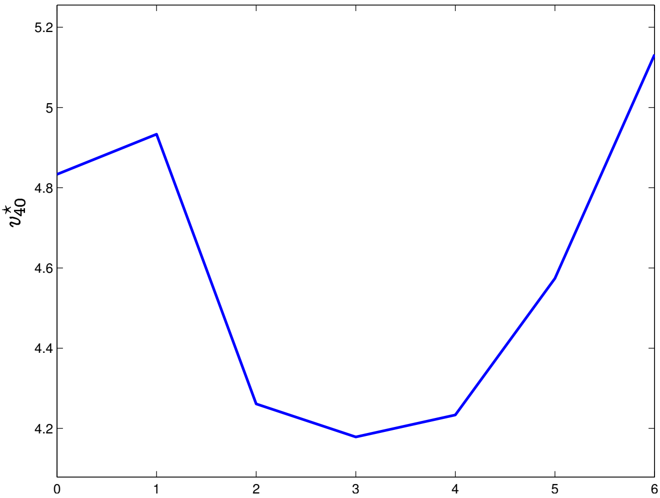
Example: Inventory model



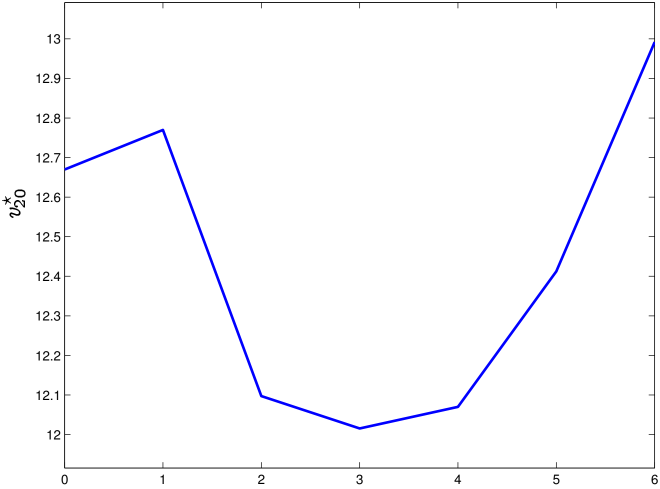
Example: Inventory model



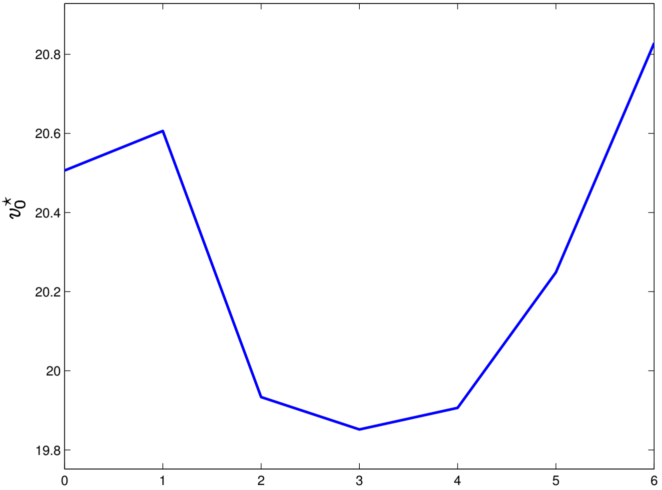
Example: Inventory model



Example: Inventory model



Example: Inventory model



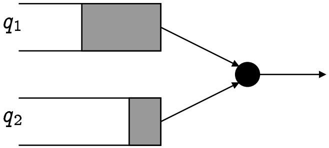
Example: Inventory model

- optimal policy vs. heuristic policy

$$\mu^* = \begin{bmatrix} 4 & \cdots & 4 & 4 \\ 3 & \cdots & 3 & 3 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \end{bmatrix}, \qquad \mu^{\mathtt{heur}} = \begin{bmatrix} 6 & \cdots & 6 & 6 \\ 5 & \cdots & 5 & 5 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \\ 0 & \cdots & 0 & 0 \end{bmatrix}$$

- expected total costs: $J^\star = 20.83$, $J^{\mathtt{heur}} = 23.13$
- heuristic policy over-orders!

Example: Queue serving



- two queues, each with maximum queue length Q
- queue lengths at time t is $q_t \in \{0, \dots, Q\}^2$
- customer arrivals at time t is $d_t \in \{0, 1\}^2$; $d_0, \dots, d_T$ are IID
  (zero or one arrival in each queue in each time period)
- server can process one customer from either queue in each time period

Example: Queue serving

- action: serve a customer from first or second queue, or neither

$$u_t \in \{(0,0),(0,1),(1,0)\}$$

- dynamics is $q_{t+1} = \min((q_t + d_t - u_t), Q)$
  - min is component-wise
  - we'll add constraint that $(u_t)_i = 0$ when $(q_t)_i = 0$, so $q_{t+1} \geq 0$
- rejected customers: $r_t = (q_t + d_t - u_t - Q)_+$
  - $(r_t)_i = 1$ when $(q_t)_i = Q$, $(d_t)_i = 1$, and $(u_t)_i = 0$
  - $(r_t)_i = 0$ otherwise

Example: Queue serving

- cost function is

$$g_t(q_t, u_t, d_t) = a^T q_t^2 + b^T q_t + c^T r_t + I_{u_t \leq q_t}(q_t, u_t)$$

- first two terms are queue length costs; third is rejection cost
- constraint $u_t \leq q_t$ is enforced by stage cost term

$$I_{u_t \leq q_t}(q_t, u_t) = \begin{cases} 0 & u_t \leq q_t \\ \infty & \text{otherwise} \end{cases}$$

- $a, b, c \in \mathbf{R}_+^2$ are cost coefficients

Example: Queue serving

problem instance:

- $Q = 5$, $T = 100$, $a = (5, 1)$, $b = (1, 10)$, $c = (10, 10)$

queue length	0	1	2	3	4	5
cost (q_1)	0	6	22	48	84	130
cost (q_2)	0	11	24	39	56	75

- distribution of d_t is

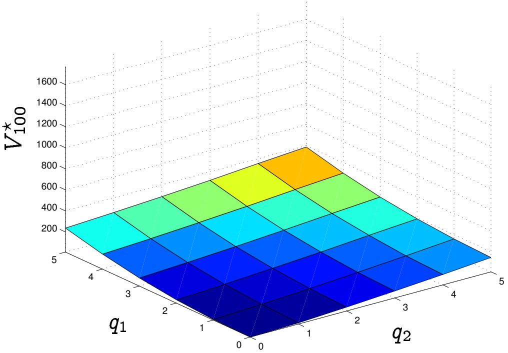
$$\begin{aligned}
\mathbf{Prob}(d_t = (0,0)) &= 0.2 \\
\mathbf{Prob}(d_t = (0,1)) &= 0.15 \\
\mathbf{Prob}(d_t = (1,0)) &= 0.45 \\
\mathbf{Prob}(d_t = (1,1)) &= 0.2
\end{aligned}$$

(arrivals at queue 1 and queue 2 are not independent)

- $\mathbf{E}\, d_t = (0.65, 0.35)$

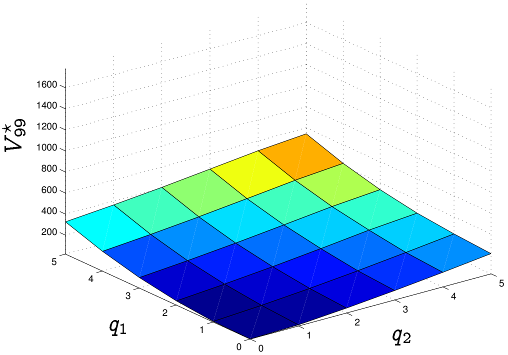
Example: Queue serving

$t = 100$



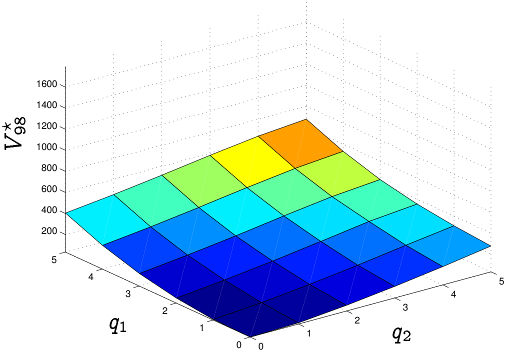
Example: Queue serving

$t = 99$



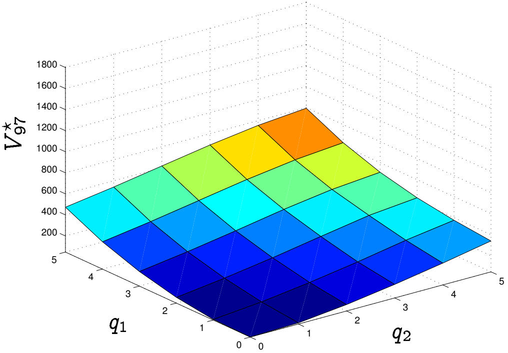
Example: Queue serving

$t = 98$



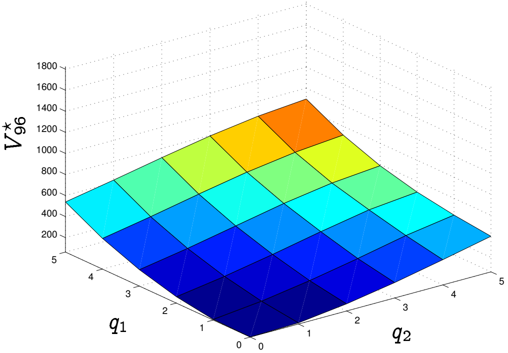
Example: Queue serving

$t = 97$



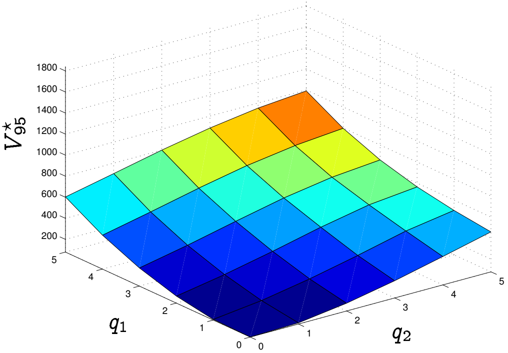
Example: Queue serving

$t = 96$



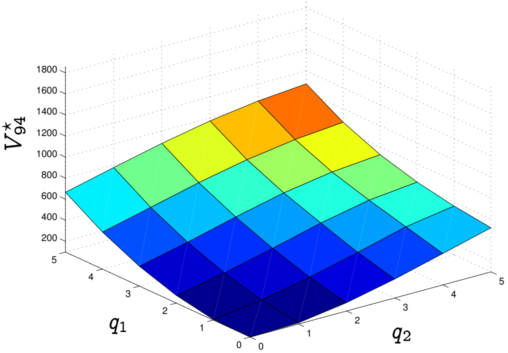
Example: Queue serving

$t = 95$



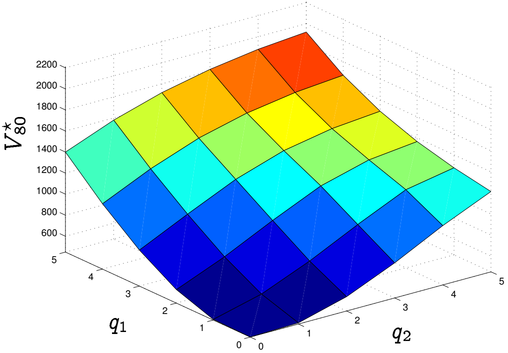
Example: Queue serving

$t = 94$



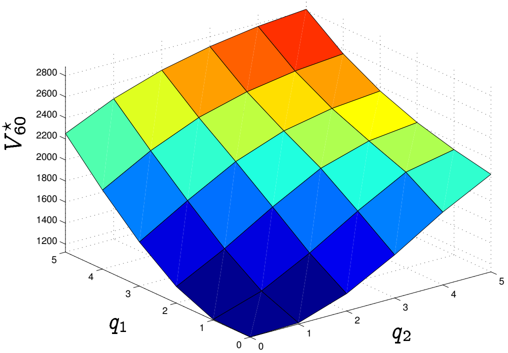
Example: Queue serving

$t = 80$



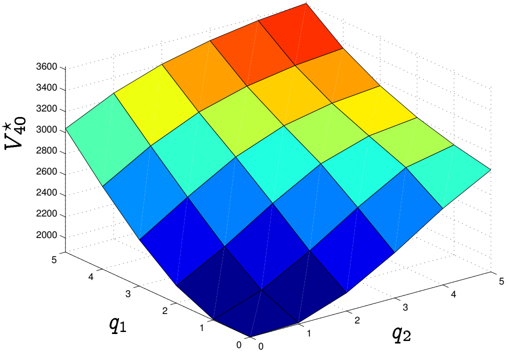
Example: Queue serving

$t = 60$



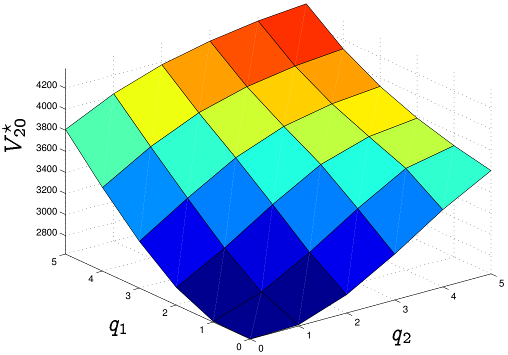
Example: Queue serving

$t = 40$



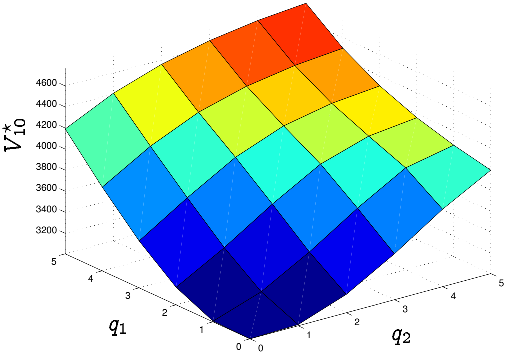
Example: Queue serving

$t = 20$



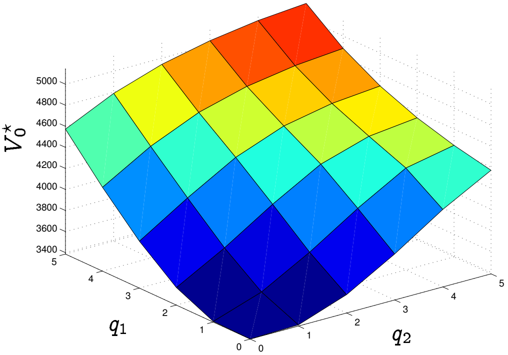
Example: Queue serving

$t = 10$



Example: Queue serving

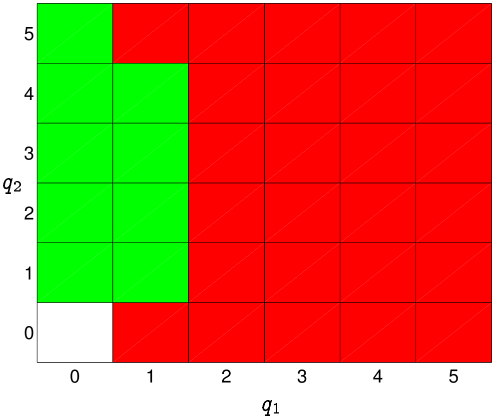
$t = 0$



Example: Queue serving

red: $\mu_t^\star(x) = (1,0)$; green: $\mu_t^\star(x) = (0,1)$

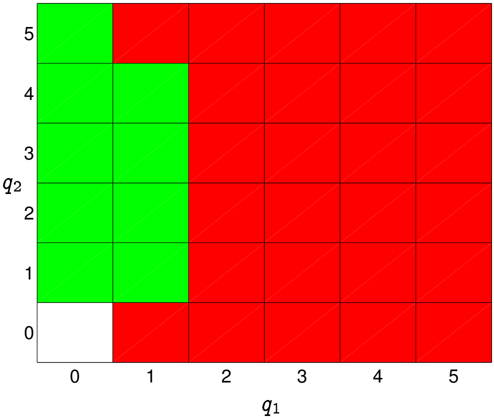
$t = 99$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

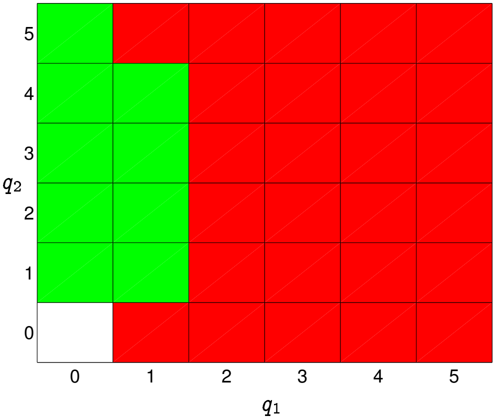
$t = 98$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

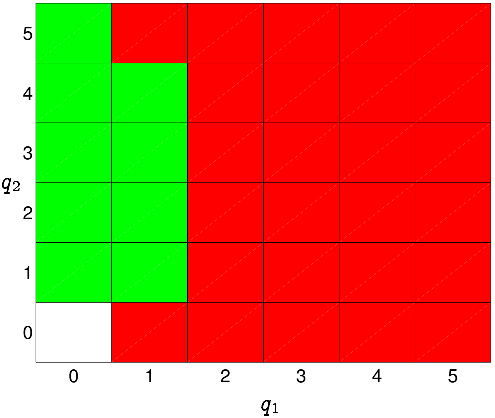
$t = 97$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

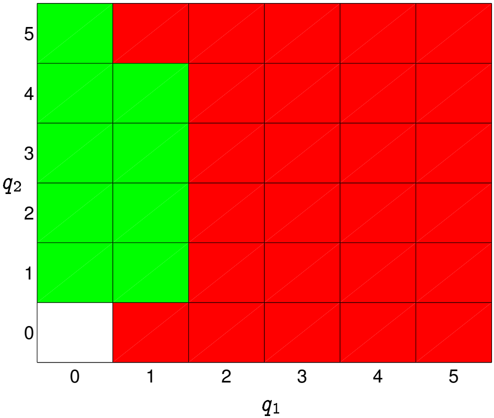
$t = 96$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

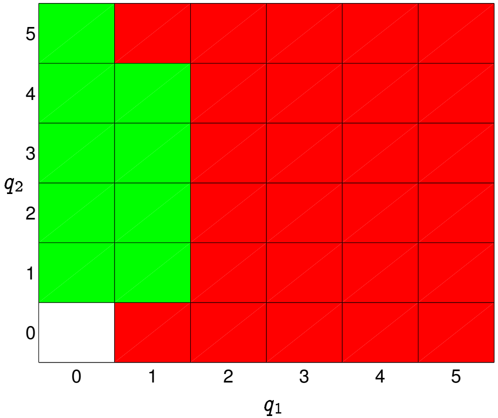
$t = 95$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

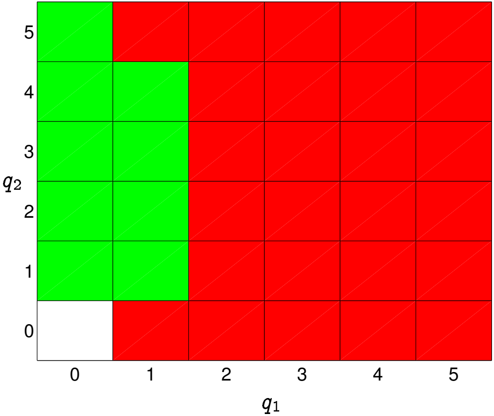
$t = 80$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

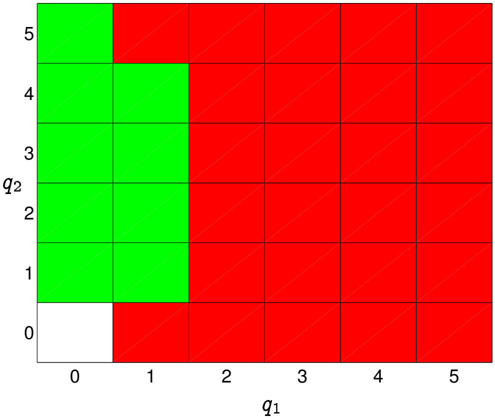
$t = 60$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

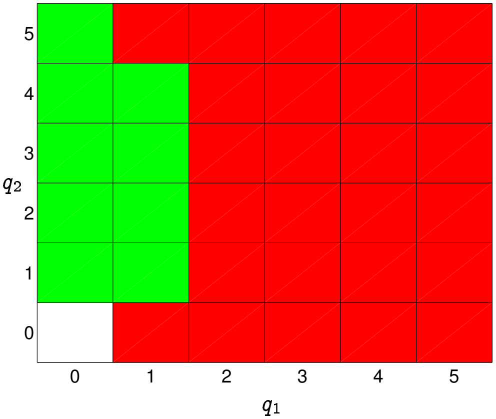
$t = 40$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$;   green: $\mu_t^\star(x) = (0, 1)$

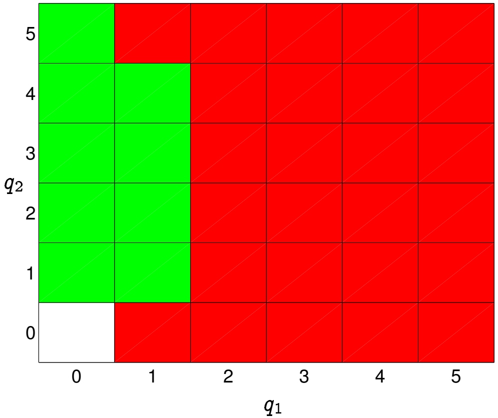
$t = 30$



Example: Queue serving

red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

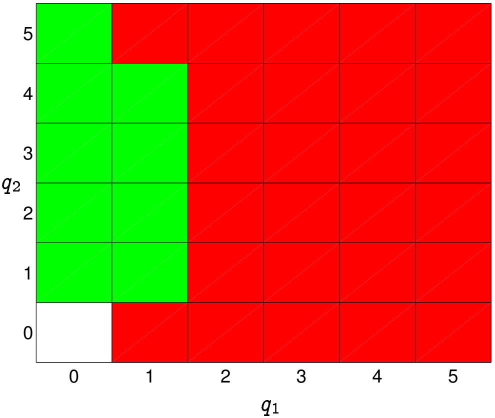
$t = 20$



Example: Queue serving

red: $\mu_t^\star(x) = (1,0)$;   green: $\mu_t^\star(x) = (0,1)$

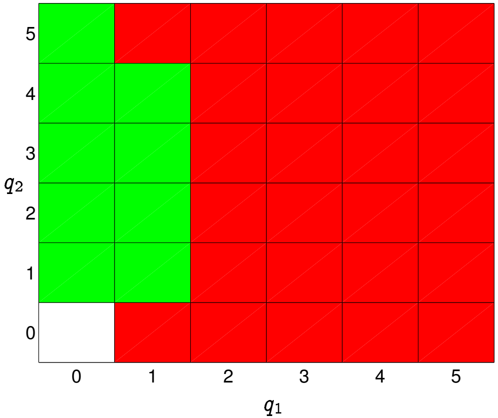
$t = 10$



Example: Queue serving

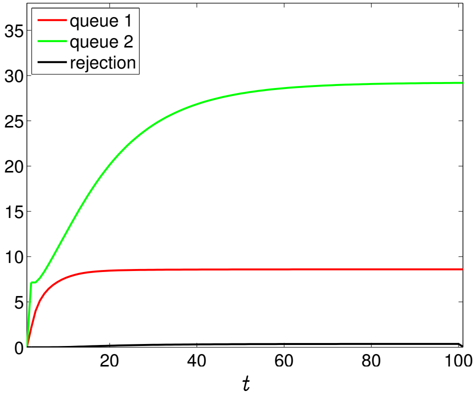
red: $\mu_t^\star(x) = (1, 0)$; green: $\mu_t^\star(x) = (0, 1)$

$t = 0$



Example: Queue serving

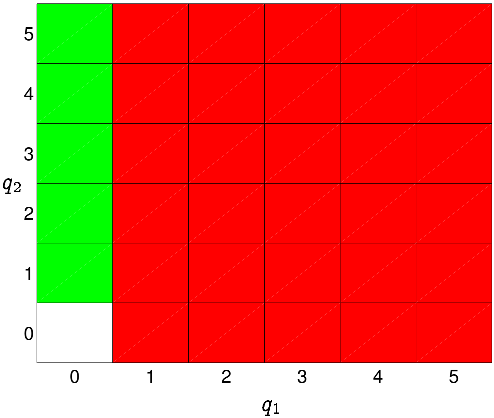
- starting with both queues empty
- expected cost over time, under the optimal policy



- total expected cost is $J^\star = 3387$

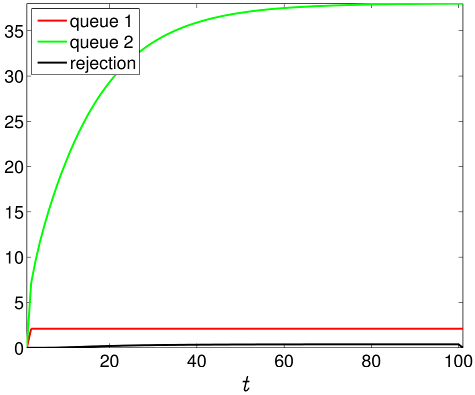
Example: Queue serving

consider q_1 priority policy, μ^1



Example: Queue serving

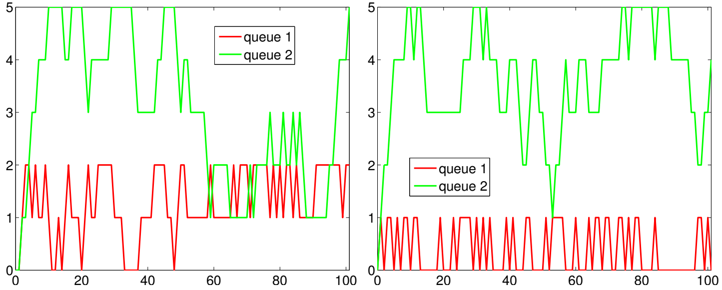
- expected cost over time, under policy μ^1



- total expected cost is $J^1 = 3632$

Example: Queue serving

time traces: optimal policy $\mu^\star$ (left), q_1 priority policy μ^1 (right)



Observations

- for time-invariant dynamics and stage costs, as t goes down
  - the policy appears to converge: $\mu_{t-1} = \mu_t$
  - V_t seems to converge to a fixed shape, plus an offset:

$$V_{t-1} \approx V_t + \alpha$$

    (α is average stage cost)
- more on these phenomena later

Outline

DP for modified information pattern

- suppose w_t is known (as well as x_t) before u_t is chosen
- typical applications: action is chosen *after* current (random) price, cost, demand, congestion is revealed
- policy has form $u_t = \mu_t(x_t, w_t)$, $\mu_t : \mathcal{X}_t \times \mathcal{W}_t \to \mathcal{U}_t$
- can map this into our standard form, but it's more natural to modify DP to handle it directly

Optimal value function when w_t is known

- define

$$V_t^\star(x) = \min_{\mu_t,\mu_{t+1},\ldots,\mu_{T-1}} \mathbf{E}\left(\sum_{\tau=t}^{T-1} g_\tau(x_\tau, u_\tau, w_\tau) + g_T(x_T) \,\middle|\, x_t = x \right)$$

  - minimization is over policies $\mu_t, \ldots, \mu_{T-1}$, functions of x *and* w
  - subject to dynamics $x_{t+1} = f_t(x_t, u_t, w_t)$

- $V_t^\star(x)$ is expected cost-to-go, using an optimal policy, if you are in state x at time t, *before* w_t *is revealed*

Dynamic programming for w_t known

- define $V_T^\star(x) := g_T(x)$
- for $t = T-1, \ldots, 0$,
  - find optimal policy for time t in terms of $V_{t+1}^\star$:
    $$\mu_t^\star(x,w) \in \operatorname*{argmin}_u \left(g_t(x,u,w) + V_{t+1}^\star(f_t(x,u,w)) \right)$$
  - find $V_t^\star$ using $\mu_t^\star$:
    $$V_t^\star(x) := \mathbf{E}\left(g_t(x, \mu_t^\star(x,w_t), w_t) + V_{t+1}^\star(f_t(x, \mu_t^\star(x,w_t), w_t)) \right)$$
    (expectation is over w_t)
- only need to store a value function on $\mathcal{X}_t$, even though policy is a function on $\mathcal{X}_t \times \mathcal{W}_t$

Outline

Optimal value function and dynamic programming

Proof of optimality

Examples

Dynamic programming for modified information pattern

Dynamic programming for modified information pattern II

DP for modified information pattern II

- suppose $w_t = (w_t^{(1)}, w_t^{(2)})$ splits into independent components
- $w_t^{(1)}$ is known (as well as x_t) before u_t is chosen
- $w_t^{(2)}$ is not known before u_t is chosen
- policy has form $u_t = \mu_t(x_t, w_t^{(1)})$, $\mu_t : \mathcal{X}_t \times \mathcal{W}_t^{(1)} \to \mathcal{U}_t$
- can map this into our standard form, but it's more natural to modify DP to handle it directly

Optimal value function when $w_t^{(1)}$ is known

- define

$$V_t^\star(x) = \min_{\mu_t,\mu_{t+1},\ldots,\mu_{T-1}} \mathbf{E}\left(\sum_{\tau=t}^{T-1} g_\tau(x_\tau, u_\tau, w_\tau) + g_T(x_T) \middle| x_t = x \right)$$

  - minimization is over policies $\mu_t, \ldots, \mu_{T-1}$, functions of x *and* $w^{(1)}$
  - subject to dynamics $x_{t+1} = f_t(x_t, u_t, w_t)$
- $V_t^\star(x)$ is expected cost-to-go, using an optimal policy, if you are in state x at time t, *before* $w_t^{(1)}$ *is revealed*

Dynamic programming for $w_t^{(1)}$ known

- define $V_T^\star(x) := g_T(x)$
- for $t = T-1, \ldots, 0$,
  - find optimal policy for time t in terms of $V_{t+1}^\star$:

    $$\mu_t^\star(x, w^{(1)}) \in \underset{u}{\mathrm{argmin}}\, \mathbf{E}\left(g_t(x, u, (w^{(1)}, w_t^{(2)})) + V_{t+1}^\star(f_t(x, u, (w^{(1)}, w_t^{(2)})))\right)$$

    (expectation is over $w_t^{(2)}$)
  - find $V_t^\star$ using $\mu_t^\star$:

    $$V_t^\star(x) := \mathbf{E}\left(g_t(x, \mu_t^\star(x, w_t^{(1)}), w_t) + V_{t+1}^\star(f_t(x, \mu_t^\star(x, w_t^{(1)}), w_t))\right)$$

    (expectation is over w_t)
- only need to store a value function on $\mathcal{X}_t$, even though policy is a function on $\mathcal{X}_t \times \mathcal{W}_t^{(1)}$