EE365: Introduction

About the course

Optimization

Dynamic system

Stochastic control

Outline

About the course

- EE365 is the same as MS&E251
- created by Ben Van Roy, Sanjay Lall, and Stephen Boyd last year
- taught by Sanjay Lall and Stephen Boyd this year

Requirements & prerequisites

- class attendance
- homework assigned asynchronously as we make it up
- 24 hour take-home exam (as in EE263, EE364a)
- willingness to program in matlab or python
- flexibility/tolerance, since it's a new(ish) course
- prerequisites:
  - linear algebra (EE263 or MS&E211; more than Math 51)
  - probability (EE178/278A or MS&E220)

It's a new(ish) course

- we'll make mistakes (in lectures, homework, ...)
- notation will be inconsistent
- notes/slides will change often
- if disorganization bothers you, or you're squeamish about seeing professors make mistakes, wait until next year

Stochastic control

- *multi-step decision making, in an uncertain dynamic environment*
- act; learn/observe; act; learn/observe, . . .
  - your current action affects the future
  - there is uncertainty in what the effect of your action will be
- goal is to find *policy*
  - what you do in any situation
  - map from what you know to what you do
- key concept is recourse (a.k.a. feedback):
  taking corrective action based on new information
- richer concept than optimization

Applications

- multi-period investment
- automatic control
- supply chain optimization
- internet ad display
- revenue management
- operation of a smart grid
- data center operation

...and many, many others

Outline

Optimization problem

$$\begin{array}{ll} \text{minimize} & f(x) \\ \text{subject to} & x \in \mathcal{X} \end{array}$$

- x is decision variable (discrete, continuous)
- $\mathcal{X}$ is constraint set
- $f : \mathcal{X} \to \mathbf{R}$ is objective (cost function)
- x is feasible if $x \in \mathcal{X}$
- x is optimal (or a solution) if $f(x) = \inf_{z \in \mathcal{X}} f(z)$
- f and $\mathcal{X}$ can depend on parameters (data)
- can maximize by minimizing $-f$ (reward, utility, profit, ...)
- standard trick: allow $f(x) = \infty$ (to embed further constraints in objective)

Solving optimization problems

- a solution method or algorithm computes a solution, given parameters
- difficulty of solving optimization problem depends on
  - mathematical properties of f, $\mathcal{X}$
  - problem size (*e.g.*, dimension of x when $x \in \mathbf{R}^n$)
- a few problems can be solved 'analytically'
- but this is not particularly relevant, since we adopt algorithmic approach

Examples

- find shortest path on weighted graph from node S to node T
  - x is path
  - $f(x)$ is weighted path length (sum of weights on edges)
  - $\mathcal{X}$ is set of paths from S to T
- allocate a total resource B among n entities to maximize total profit
  - $x \in \mathbf{R}^n$ gives allocation
  - (maximize) objective $f(x) = \sum_{i=1}^n P_i(x_i)$
  - $P_i(x_i)$ is profit of entity i given resource amount x_i
  - $\mathcal{X} = \{x \mid x \geq 0,\ \mathbf{1}^T x = B\}$

Outline

(Deterministic) dynamic system

$$x_{t+1} = f_t(x_t, u_t), \quad t = 0, 1, \ldots$$

- t is time (epoch, stage, period)
- $x_t \in \mathcal{X}_t$ is state
- initial state x_0 is known or given
- $u_t \in \mathcal{U}_t$ is input (action, decision, choice, control)
- $f_t : \mathcal{X}_t \times \mathcal{U}_t \to \mathcal{X}_{t+1}$ is state transition function
- called time-invariant if f_t, $\mathcal{X}_t$, $\mathcal{U}_t$ don't depend on t
- variation: $\mathcal{U}_t$ can depend on x_t

Idea of state

- current action affects future states, but not current or past states
- current state depends on past actions
- state is link between past and future
  - if you know state x_t and actions $u_t, \ldots, u_{s-1}$, you know x_s
  - $u_0, \ldots, u_{t-1}$ not relevant
- state is sufficient statistic (summary) for past

Examples (with finite state and inputs spaces)

discrete dynamical system:

- $\mathcal{X} = \{1, \dots, n\}$, $\mathcal{U} = \{1, \dots, m\}$
- $f_t : \mathcal{X} \times \mathcal{U} \to \mathcal{X}$ called transition map, given by table (say)

moving on directed graph $(\mathcal{V}, \mathcal{E})$:

- $\mathcal{X} = \mathcal{V}$, $\mathcal{U}(x_t)$ is set of out-going edges from x_t
- $f_t(x_t, u_t) = v$, where $u_t = (x_t, v)$

Examples (with infinite state and input spaces)

linear dynamical system:

- $\mathcal{X} = \mathbf{R}^n$, $\mathcal{U} = \mathbf{R}^m$
- $x_{t+1} = f_t(x_t, u_t) = A_t x_t + B_t u_t + c_t$

very special form for dynamics, but arises in many applications

Dynamic optimization (deterministic optimal control)

$$\begin{array}{ll} \text{minimize} & J = \sum_{t=0}^{T-1} g_t(x_t, u_t) + g_T(x_T) \\ \text{subject to} & x_{t+1} = f_t(x_t, u_t), \quad t = 0, \ldots, T-1 \end{array}$$

- initial state x_0 is given
- $g_t : \mathcal{X}_t \times \mathcal{U}_t \to \mathbf{R} \cup \{\infty\}$ is stage cost function
- $g_T : \mathcal{X}_T \to \mathbf{R} \cup \{\infty\}$ is terminal cost function
- variables are $x_1, \ldots, x_T$, $u_0, \ldots, u_{T-1}$
  (or just $u_0, \ldots, u_{T-1}$, since these determine $x_1, \ldots, x_T$)
- just an optimization problem (possibly big)
- also called classical or open-loop control

Deterministic optimal control

- addresses dynamic effect of actions across time
- no uncertainty or randomness in model
- is widely used (often, by simply ignoring uncertainty in the application)

Outline

Stochastic dynamic system

$$x_{t+1} = f_t(x_t, u_t, w_t), \quad t = 0, 1, \ldots$$

- w_t are random variables (usually assumed independent for $t \neq s$)
- state transitions are nondeterministic, uncertain
- choice of input u_t determines *distribution* of x_{t+1}
- initial state x_0 is random variable (usually assumed independent of $w_0, w_1, \ldots$)

Objective

- objective (to be minimized) is

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

- $g_t : \mathcal{X}_t \times \mathcal{U}_t \times \mathcal{W}_t \to \mathbf{R} \cup \{\infty\}$ is stage cost function
- $g_T : \mathcal{X}_T \times \mathcal{W}_T \to \mathbf{R} \cup \{\infty\}$ is terminal cost function
- often g_t, g_T don't depend on w_t, *i.e.*, stage and terminal costs are deterministic
- infinite values of g_t encode constraints
- objective is mean total stage cost plus terminal cost

Information pattern constraints

- information pattern constraint: u_t *depends on what you know at time* t

$$u_t = \phi_t(Z_t)$$

- Z_t is what you know at time t
- $(\phi_0, \ldots, \phi_{T-1})$ is called policy
- goal is to find policy that minimizes J, subject to dynamics

Information patterns

- full knowledge (prescient): $Z_t = (w_0, \ldots, w_{T-1})$
  - for each realization, reduces to deterministic optimal control problem
- no knowledge: $Z_t = \emptyset$
  - reduces to an optimization problem; called open-loop
- in between: $Z_t = x_t$ (called state feedback)
- a little more: $Z_t = (x_t, w_t)$

these are very different problems!

Example: Stochastic shortest path

- move from node S to node T in directed weighted graph
- minimize expected total weight along path
- edge weights are random variables, independent in each time period

information patterns:

- no knowledge: commit to path beforehand
  (knowing distributions of weights, but not actual values)
- full knowledge: weights on all edges at all times are revealed before path is chosen
- local knowledge: at each node, at each time, weights of out-going edges are revealed before next edge on path is chosen

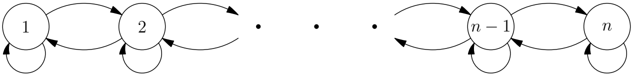
Example: Optimal disposition of stock

- sell a total amount S of a stock in T periods
- price (and transaction cost) varies randomly
- maximize expected revenue

information patterns:

- no knowledge: commit to sales amounts beforehand
- in each time period, the price and transaction cost is known before amount sold is chosen

Stochastic shortest path example



- chain of $n = 100$ nodes
- move from node 1 to node n in $T = 300$ steps
- random edge weights (say, delays) in each period (including self-loops)
- can only move forward, stay put, or move backward
- minimize total expected delay

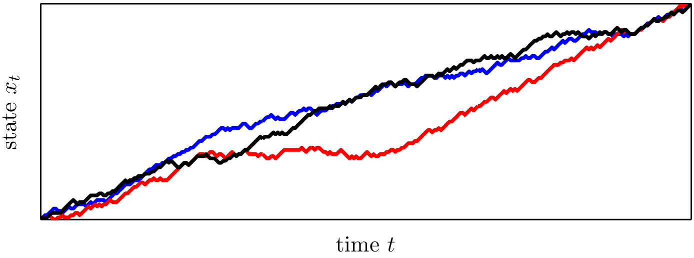
Information patterns

three different information patterns:

1. open loop: only know delay statistics
2. prescient: know everything (delays on all edges, all times)
3. local: at each time, know outgoing delays at current node (including self-loop)

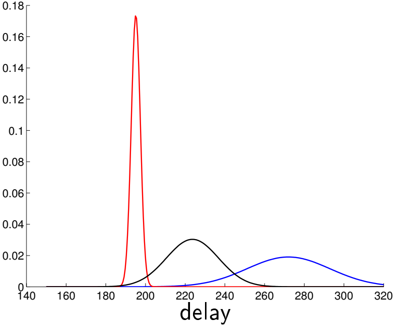
Trajectories

sample trajectory under optimal policy for each information pattern (open loop, local, prescient)



Delay distributions

delay distributions for each information pattern (open loop, local, prescient)



clearly shows value of information, recourse

Example: Vehicle intercept

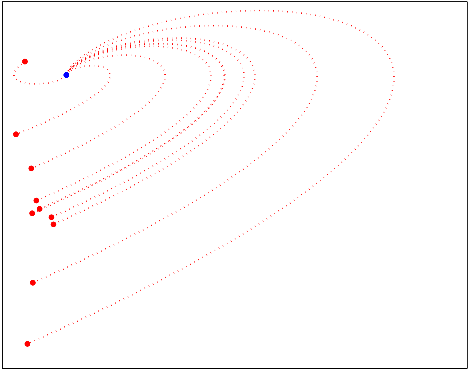
- vehicle moving in $\mathbf{R}^2$, with linear dynamics

$$x_{t+1} = Ax_t + Bu_t, \quad p_t = Cx_t, \quad t = 0, 1, \ldots$$

- $p_t \in \mathbf{R}^2$ is the vehicle position at time t
- vehicle must reach one of K (equally likely) destinations at time $t = T$ (terminal constraint is random)
- destination is revealed at time $t = M$:
  - $u_0, \ldots, u_{M-1}$ are chosen without knowledge of final destination
  - $u_M, \ldots, u_{T-1}$ can depend on the final destination
- minimize $\mathbf{E} \sum_{t=0}^{T-1} \|u_t\|_2^2$

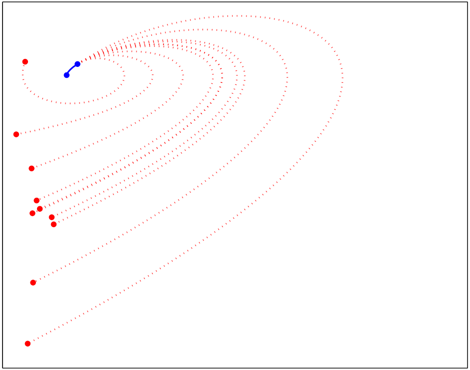
Optimal policy

$M = 0$; $T = 120$; optimal cost 0.000006



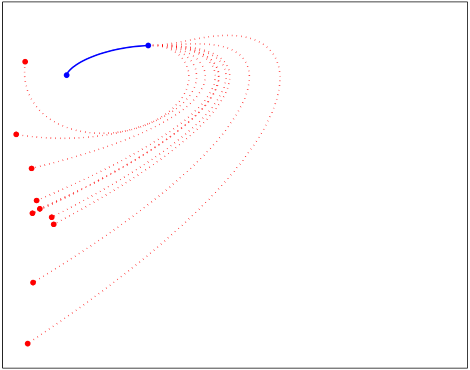
Optimal policy

$M = 30$; $T = 120$; optimal cost 0.000011



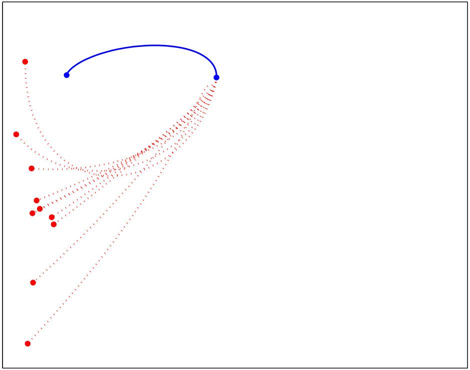
Optimal policy

$M = 60$; $T = 120$; optimal cost 0.000049



Optimal policy

$M = 90$; $T = 120$; optimal cost 0.001108



Optimal policy

$M = 110$; $T = 120$; optimal cost 0.063338