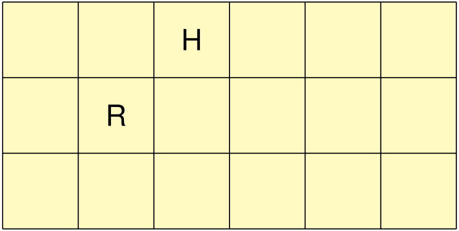
Linear Dynamical Systems

Jeremy Stober

May 7, 2009

Robot Perception and Action

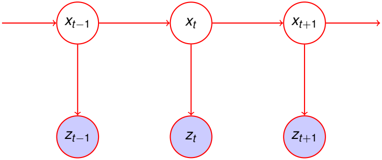
Recall the robot example from Lecture 17:



- The robot has limited noisy perception.
- The hidden system state: coordinates for human and robot.

Hidden Markov Models

- We used a Hidden Markov Model to describe the system



- And learned about efficient algorithms for
  - state estimation (Dynamic Programming)
  - finding most probable path (Viterbi Algorithm)
  - learning parameters (Expectation Maximization)

A Question

Is this a good model for a robot?

A Few Possible Issues

- Robots are *controlled* systems. Can we use a robot's observed controls to improve state estimation?
- Environments are *continuous*. How does this affect our ability to estimate robot state?
- Extra Credit: Perception is often high-dimensional, even if the hidden state is not. What impact does this have on computational feasibility?
- Extra Extra Credit (My Research): What if the properties of robot perception and action are not fixed?

A Few Possible Issues

- Robots are *controlled* systems. Can we use a robot's observed controls to improve state estimation?
- Environments are *continuous*. How does this affect our ability to estimate robot state?
- Extra Credit: Perception is often high-dimensional, even if the hidden state is not. What impact does this have on computational feasibility?
- Extra Extra Credit (My Research): What if the properties of robot perception and action are not fixed?

A Few Possible Issues

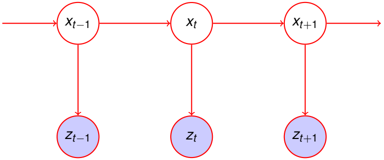
- Robots are *controlled* systems. Can we use a robot's observed controls to improve state estimation?
- Environments are *continuous*. How does this affect our ability to estimate robot state?
- Extra Credit: Perception is often high-dimensional, even if the hidden state is not. What impact does this have on computational feasibility?
- Extra Extra Credit (My Research): What if the properties of robot perception and action are not fixed?

A Few Possible Issues

- Robots are *controlled* systems. Can we use a robot's observed controls to improve state estimation?
- Environments are *continuous*. How does this affect our ability to estimate robot state?
- Extra Credit: Perception is often high-dimensional, even if the hidden state is not. What impact does this have on computational feasibility?
- Extra Extra Credit (My Research): What if the properties of robot perception and action are not fixed?

Input-Output Hidden Markov Model

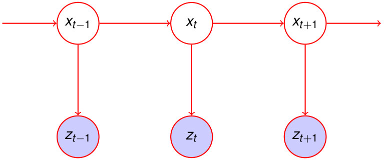
Original HMM



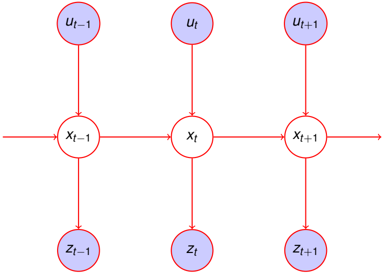
Revised HMM

Input-Output Hidden Markov Model

Original HMM

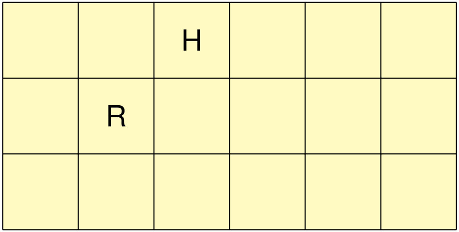


Revised HMM



Continuous State and Action Spaces

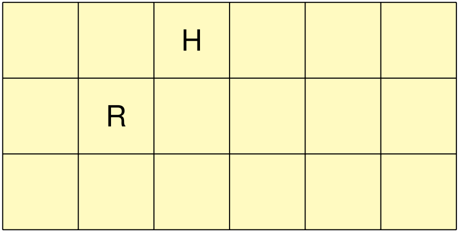
Original States and Actions



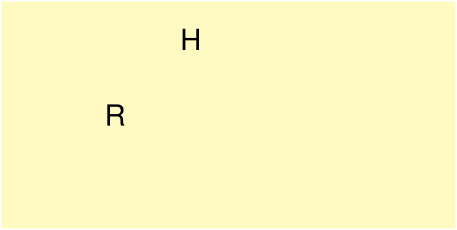
Continuous State and Action Spaces

Continuous State and Action Spaces

Original States and Actions



Continuous State and Action Spaces



Linear Dynamical Systems

We consider linear dynamical systems of the form

$$x_t = Ax_{t-1} + Bu_t + \mu_t \qquad [\mu_t \sim \mathcal{N}(0, \Gamma)]$$
$$z_t = Cx_t + \nu_t \qquad [\nu_t \sim \mathcal{N}(0, \Sigma)]$$

For linear dynamical systems we have efficient algorithms for

- performing state estimation (Kalman Filtering)
- path estimation (Kalman Smoothing)
- learning model parameters (EM)

Kalman Filtering

- Prediction Step
  - We have an estimate $\hat{x}_{t-1}$ and a covariance $\hat{P}_{t-1}$.
  - We perform action u_t.
  - What new state do we expect?
- Correction Step
  - We have now observed z_t.
  - How should we correct our prediction?
- The filter is recursive (depends on previous estimates).

$$p(\mathbf{x}_k|\mathbf{U}_k, \mathbf{Z}_k) = \int p(\mathbf{x}_k|\mathbf{z}_k, \mathbf{x}_k^-)p(\mathbf{x}_k^-|\mathbf{x}_{k-1}, u_k)p(\mathbf{x}_{k-1}|\mathbf{Z}_{k-1}, \mathbf{U}_{k-1})\, d\mathbf{x}_{k-1}$$

Prediction Step

- We use our current state estimate and observed action in the linear equation that describes the system.

$$x_t = Ax_{t-1} + Bu_t$$
$$\hat{x}_t^- = A\hat{x}_{t-1} + Bu_t \quad \text{[substitute estimate]}$$

- We also need to update our uncertainty. Sketch:

$$\hat{P}_{t-1} = E[\hat{x}_{t-1}\hat{x}_{t-1}^T]$$
$$\hat{P}_t^- = E[(A\hat{x}_{t-1})(A\hat{x}_{t-1})^T] + \textit{motor noise}$$
$$\hat{P}_t^- = A\hat{P}_{t-1}A^T + \Gamma$$

Correction Step

- Involves computing someing called the *Kalman gain*.
    - Determines how to weight the observational evidence.
    - We will skip the derivation.

$$K_t = \hat{P}_t^- K_t (C\hat{P}_t^- C^T + \Sigma)^{-1}$$

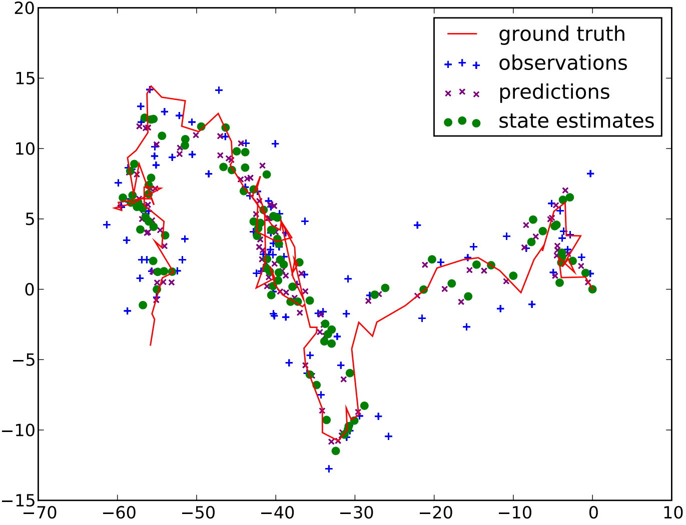
- We update our prediction using the gain:

$$\hat{x}_t = \hat{x}_t^- - K_t(z_t - C\hat{x}_t^-)$$

- We also update our covariance:

$$\hat{P}_t = (I - K_t C)\hat{P}_t^-$$

Tracking a Robot



Mean Squared Error

Predictions	6.43
Observations	9.58
Estimates	3.366

Table: Mean Squared Error Comparison

Comments

- For a LDS - Kalman Filtering is optimal (MSE).
- Linearity is a strong assumption for state estimation.
    - Extended Kalman Filter - Assumes state transitions are locally linear.
- State estimation requires knowing a number of parameters: $A, B, C, \Gamma, Sigma$.
- Can use Expectation Maximization to learn missing parameters.