



<http://elec3004.com>

Adaptive Filters & Estimation

ELEC 3004: **Systems**: Signals & Controls
Dr. Surya Singh

Lecture 14

elec3004@itee.uq.edu.au

<http://robotics.itee.uq.edu.au/~elec3004/>

April 27, 2017

© 2017 School of Information Technology and Electrical Engineering at The University of Queensland

CC BY-NC-SA

Lecture Schedule:

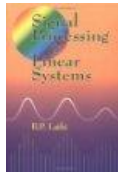
Week	Date	Lecture Title
1	28-Feb	Introduction
	2-Mar	Systems Overview
	7-Mar	Systems as Maps & Signals as Vectors
2	9-Mar	Systems: Linear Differential Systems
	14-Mar	Sampling Theory & Data Acquisition
3	16-Mar	Aliasing & Antialiasing
	21-Mar	Discrete Time Analysis & Z-Transform
4	23-Mar	Second Order LTID (& Convolution Review)
	28-Mar	Frequency Response
5	30-Mar	Filter Analysis
	4-Apr	Digital Filters (IIR) & Filter Analysis
6	6-Apr	Digital Filter (FIR)
	11-Apr	Digital Windows
7	13-Apr	FFT
	18-Apr	Holiday
	20-Apr	
	25-Apr	
8	27-Apr	Active Filters & Estimation
9	2-May	Introduction to Feedback Control
	4-May	Servoregulation/PID
10	9-May	Introduction to (Digital) Control
	11-May	Digital Control
11	16-May	Digital Control Design
	18-May	Stability
12	23-May	Digital Control Systems: Shaping the Dynamic Response
	25-May	Applications in Industry
13	30-May	System Identification & Information Theory
	1-Jun	Summary and Course Review



ELEC 3004: **Systems**

27 April 2017 2

Follow Along Reading:



B. P. Lathi
*Signal processing
and linear systems*
1998
[TK5102.9.L38 1998](#)

Today

- Chapter 12
Frequency Response & Digital Filters



**G. Franklin,
J. Powell,
M. Workman**
*Digital Control
of Dynamic Systems*
1990

[TJ216.F72 1990](#)
[\[Available as
UQ Ebook\]](#)

- Lathi: Chapter 13
 - State-Space Analysis
- FPW: Chapter 2
 - Chapter 2: Linear, Discrete,
Dynamic-Systems Analysis

Next Time



ELEC 3004: Systems

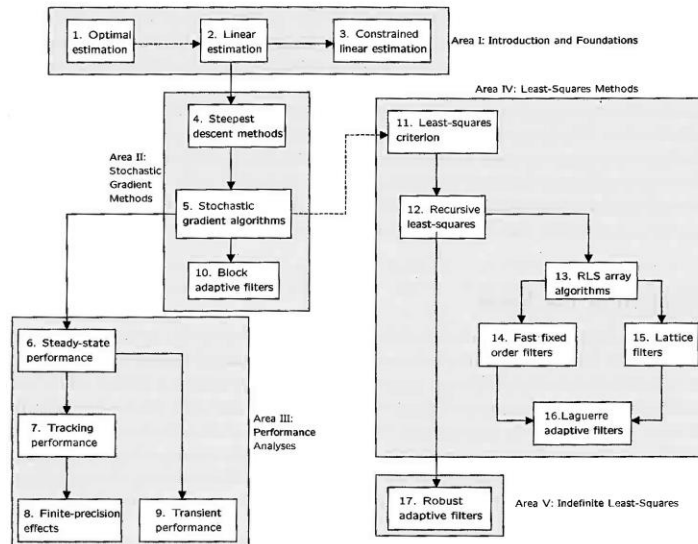
27 April 2017 3

Adaptive Filters & Random Variables

ELEC 3004: Systems

27 April 2017 4

Adaptive Filters



Random or Stochastic Variables

- A random variable is one described by its **Expectation (E)**

$$\bar{x} \triangleq E x, \quad \sigma_x^2 \triangleq E (x - \bar{x})^2 = E x^2 - \bar{x}^2$$

- The Variance is the Expectation of the difference between the variable and its mean
- When x has zero mean, its variance is simply given by

$$\sigma_x = E x^2$$



Kalman Filter: A Gaussian way to beat the noise

Along multiple dimensions



State Space

- We collect our set of uncertain variables into a vector ...
 $\mathbf{x} = [x_1, x_2, \dots, x_N]^T$
- The set of values that \mathbf{x} might take on is termed the *state space*
- There is a *single* true value for \mathbf{x} ,
but it is unknown



State Space Dynamics

$$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$

$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

$$H(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B}$$



Measured versus True

- Measurement errors are inevitable

- So, add Noise to State...
 - State Dynamics becomes:

$$\begin{aligned}\dot{\mathbf{x}} &= \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u} + \mathbf{w} \\ \mathbf{y} &= \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u} + \mathbf{v}\end{aligned}$$

- Can represent this as a “Normal” Distribution

$$\mathcal{N}(x; \mu, \sigma) = \frac{1}{(\sqrt{2\pi}) \sigma} \exp\left(-\frac{(x - \mu)^2}{2\sigma^2}\right)$$



Recovering The Truth

- Numerous methods
- Termed “Estimation” because we are trying to estimate the truth from the signal
- A strategy discovered by Gauss
- Least Squares in Matrix Representation

$$\begin{bmatrix} p_0 \\ p_1 \end{bmatrix} = \begin{bmatrix} n & \sum_1^n t_i \\ \sum_1^n t_i & \sum_1^n t_i^2 \end{bmatrix}^{-1} \begin{bmatrix} \sum_1^n z_i \\ \sum_1^n t_i z_i \end{bmatrix}$$



Recovering the Truth: Terminology

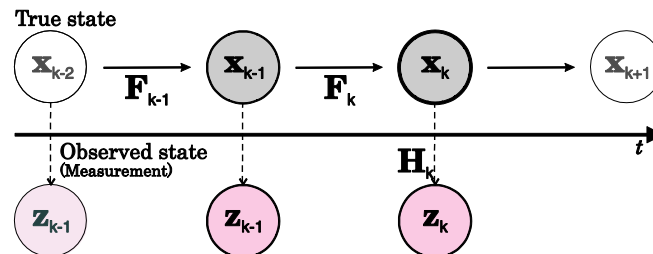
$$\dot{\mathbf{x}} = \mathbf{F}\mathbf{x} + \mathbf{G}\mathbf{u} + \mathbf{w}$$

$$\mathbf{z} = \mathbf{H}\mathbf{x} + \mathbf{v}$$

- \mathbf{x} : the state vector
- $\mathbf{x}_{A|B}$: the state of \mathbf{x} at time A based on data taken up to time B
- $\hat{\mathbf{x}}$: estimate of the true state vector
- \mathbf{F} : system dynamics matrix in continuous time (equivalent to \mathbf{A} in Eq. 1)
- \mathbf{G} : system control matrix relating deterministic input, \mathbf{u} , to the state (equivalent to \mathbf{B} in Eq. 1)
- \mathbf{H} : measurement matrix in continuous time (equivalent to \mathbf{C} in Eq. 2)
- \mathbf{F}_i : system model in **discrete** time at $t = t_i$
- \mathbf{H}_i : measurement model in **discrete** time at $t = t_i$
- \mathbf{P}_i : estimate covariance in **discrete** time at $t = t_i$
- \mathbf{w} : process uncertainty (noise) vector (of type $\mathcal{N}(0, s)$)
- \mathbf{Q} : process noise matrix, $\mathbf{Q} = E[\mathbf{w}\mathbf{w}^T]$
- \mathbf{Q}_i : \mathbf{Q} in discrete time at $t = t_i$
- \mathbf{v} : measurement noise vectors (of type $\mathcal{N}(0, \sigma)$)
- \mathbf{R}_i : the measurement variance matrix, $\mathbf{R} = E[\mathbf{v}\mathbf{v}^T]$, in discrete time at $t = t_i$



General Problem...

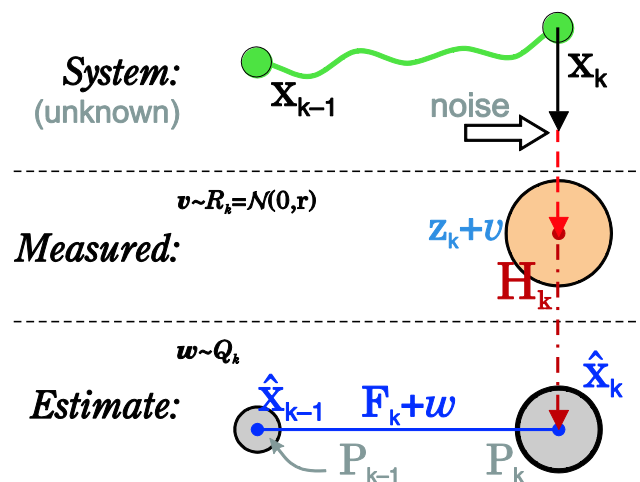


Duals and Dual Terminology

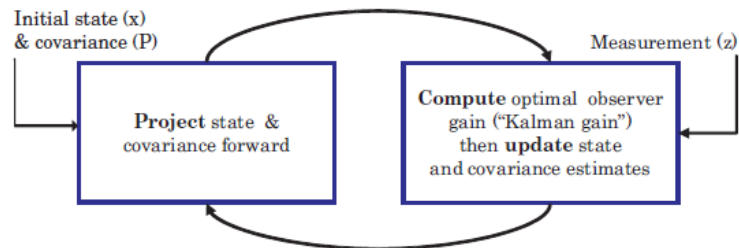
	Estimation		Control
Model:	$\dot{\mathbf{x}} = \mathbf{F}\mathbf{x}$ (discrete: $\mathbf{x} = \mathbf{F}_k\mathbf{x}$)	\leftrightarrow	$\dot{\mathbf{x}} = \mathbf{A}\mathbf{x}$, $\mathbf{A} = \mathbf{F}^1$
Regulates:	\mathbf{P} (covariance)	\leftrightarrow	\mathbf{M} (performance matrix)
Minimized function:	\mathbf{Q} (or $\mathbf{G}\mathbf{Q}\mathbf{G}^1$)	\leftrightarrow	\mathbf{V}
Optimal Gain:	\mathbf{K}	\leftrightarrow	\mathbf{G}
Completeness law:	Observability	\leftrightarrow	Controllability



Estimation Process in Pictures



Kalman Filter Process



KF Process in Equations

$$\begin{aligned}
 \text{Prediction: } \hat{\mathbf{x}}_{k|k-1} &= \mathbf{F}_{k-1} \hat{\mathbf{x}}_{k-1|k-1}, && \text{(state prediction)} \\
 \mathbf{P}_{k|k-1} &= \mathbf{Q}_{k-1} + \mathbf{F}_{k-1} \mathbf{P}_{k-1|k-1} \mathbf{F}_{k-1}^T, && \text{(covariance prediction)} \\
 \text{Kalman Gain: } \mathbf{K}_k &= \mathbf{P}_{k|k-1} \mathbf{H}^T [\mathbf{H} \mathbf{P}_{k|k-1} \mathbf{H}^T + \mathbf{R}_k]^{-1}, \\
 \text{Update: } \mathbf{P}_{k|k} &= [\mathbf{I} - \mathbf{K}_k \mathbf{H}] \mathbf{P}_{k|k-1}, && \text{(covariance update)} \\
 \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \mathbf{K}_k (\mathbf{z}_k - \mathbf{H} \hat{\mathbf{x}}_{k|k-1}) && \text{(state update)}
 \end{aligned}$$



KF Considerations

$$\begin{aligned}
 \underbrace{\hat{\mathbf{x}}_{k|k-1}}_{n \times 1} &= \underbrace{\mathbf{F}_{k-1}}_{n \times n} \hat{\mathbf{x}}_{k-1|k-1} + \underbrace{\mathbf{G}_{k-1}}_{n \times j} \underbrace{\mathbf{u}_{k-1}}_{j \times 1} \\
 \underbrace{\mathbf{P}_{k|k-1}}_{n \times n} &= \underbrace{\mathbf{Q}_{k-1}}_{n \times n} + \mathbf{F}_{k-1} \mathbf{P}_{k-1|k-1} \mathbf{F}_{k-1}^T \\
 \underbrace{\mathbf{K}_k}_{n \times m} &= \mathbf{P}_{k|k-1} \underbrace{\mathbf{H}^T}_{n \times m} \underbrace{[\mathbf{H} \mathbf{P}_{k|k-1} \mathbf{H}^T + \mathbf{R}_k]^{-1}}_{m \times m} \\
 \mathbf{P}_{k|k} &= [\mathbf{I} - \mathbf{K}_k \mathbf{H}] \mathbf{P}_{k|k-1} \\
 \hat{\mathbf{x}}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \underbrace{\mathbf{K}_k}_{m \times 1} \left(\underbrace{\mathbf{z}_k}_{m \times 1} - \underbrace{\mathbf{H}}_{m \times n} \hat{\mathbf{x}}_{k|k-1} - \mathbf{H} \mathbf{G}_k \mathbf{u}_{k-1} \right)
 \end{aligned}$$



Ex: Kinematic KF: Tracking

- Consider a System with Constant Acceleration

$$\begin{aligned}
 \ddot{y} &= -g \\
 \dot{y} &= gt + p_1 \\
 y &= p_0 + p_1 t + \frac{gt^2}{2}
 \end{aligned}$$

$$\begin{bmatrix} \dot{y} \\ \ddot{y} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} y \\ \dot{y} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ 0 & -1 \end{bmatrix} \begin{bmatrix} 0 \\ g \end{bmatrix}$$

$$\mathbf{F} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 0 & 0 & \ddots & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{bmatrix} \quad \mathbf{F}_k = \begin{bmatrix} 0 & t_s & \frac{t_s^2}{2} \\ 0 & 0 & t_s \\ 0 & 0 & 0 \end{bmatrix}$$

$$\hat{\mathbf{x}}_k = \mathbf{F}_{k-1} \hat{\mathbf{x}}_{k-1} + \mathbf{K}_k (\mathbf{z}_k - \mathbf{H} \mathbf{F}_{k-1} \hat{\mathbf{x}}_{k-1})$$



In Summary

- KF:
 - The true state (x) is separate from the measured (z)
 - Lets you **combine** prior controls knowledge with measurements to filter signals and find the truth
 - It **regulates** the covariance (P)
 - As P is the scatter between z and x
 - So, if $P \rightarrow 0$, then $z \rightarrow x$ (measurements \rightarrow truth)
- EKF:
 - Takes a Taylor series approximation to get a local “F” (and “G” and “H”)



Estimation: “Bayesian Perspective”

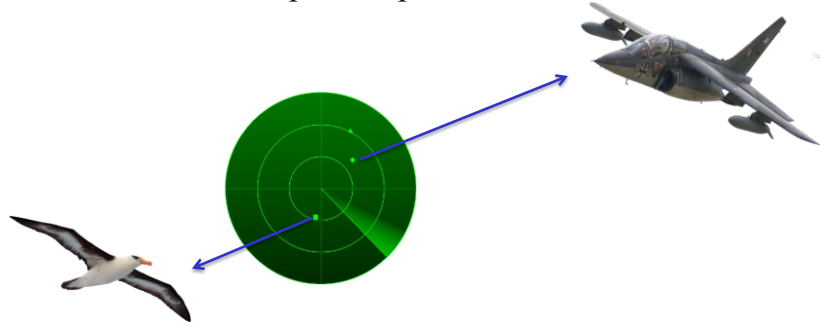


$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

Based on Material from Jur van den Berg, Introduction to Robotics

Kalman Filtering

- (Optimal) estimation of the (hidden) state of a linear dynamic process of which we obtain noisy (partial) measurements
- Example: radar tracking of an airplane.
What is the state of an airplane given noisy radar measurements of the airplane's position?



Model

- Discrete time steps, continuous state-space
- (Hidden) state: \mathbf{x}_t , measurement: \mathbf{y}_t
- Airplane example:
- Position, speed and acceleration

$$\mathbf{x}_t = \begin{pmatrix} x_t \\ \dot{x}_t \\ \ddot{x}_t \end{pmatrix}, \quad \mathbf{y}_t = (\tilde{x}_t)$$



Dynamics and Observation model

- Linear dynamics **model** describes relation between the state and the next state, and the observation:
- Airplane example (if process has time-step δ):

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + \mathbf{w}_t, \quad \mathbf{w}_t \sim W_t = N(\mathbf{0}, Q)$$

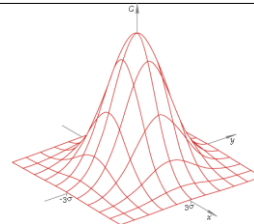
$$\mathbf{y}_t = C\mathbf{x}_t + \mathbf{v}_t, \quad \mathbf{v}_t \sim V_t = N(\mathbf{0}, R)$$

$$A = \begin{pmatrix} 1 & \delta & \frac{1}{2}\delta^2 \\ 0 & 1 & \delta \\ 0 & 0 & 1 \end{pmatrix}, \quad C = \begin{pmatrix} 1 & 0 & 0 \end{pmatrix}$$



Normal distributions

- Let X_0 be a normal distribution of the initial state \mathbf{x}_0
- Then, every X_t is a normal distribution of hidden state \mathbf{x}_t . Recursive definition:
- And every Y_t is a normal distribution of observation \mathbf{y}_t .
Definition: $X_{t+1} = AX_t + W_t$
- **Goal of filtering:** compute conditional distribution



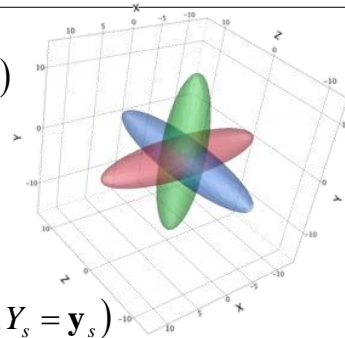
$$Y_t = CX_t + V_t$$

$$(X_t | Y_0 = \mathbf{y}_0, \dots, Y_t = \mathbf{y}_t)$$



Normal distribution

- Because X_t 's and Y_t 's are normal distributions, $(X_t | Y_0 = \mathbf{y}_0, \dots, Y_t = \mathbf{y}_t)$ is also a normal distribution
- Normal distribution is fully specified by mean and covariance
- We denote:



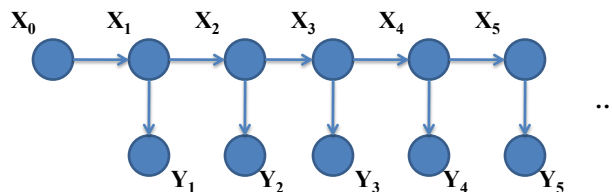
$$\begin{aligned} X_{t|s} &= (X_t | Y_0 = \mathbf{y}_0, \dots, Y_s = \mathbf{y}_s) \\ &= N(\mathbf{E}(X_t | Y_0 = \mathbf{y}_0, \dots, Y_s = \mathbf{y}_s), \text{Var}(X_t | Y_0 = \mathbf{y}_0, \dots, Y_s = \mathbf{y}_s)) \\ &= N(\hat{\mathbf{x}}_{t|s}, P_{t|s}) \end{aligned}$$

Problem reduces to computing $\hat{\mathbf{x}}_{t|t}$ and $P_{t|t}$



Recursive update of state

- Kalman filtering algorithm: repeat...
 - Time update:
from $X_{t|t}$, compute **a priori** distribution $X_{t+1|t}$
 - Measurement update:
from $X_{t+1|t}$ (and given \mathbf{y}_{t+1}), compute **a posteriori** distribution $X_{t+1|t+1}$



Time update

- From $X_{t|t}$, compute **a priori** distribution $X_{t+1|t}$:

$$\begin{aligned}
 X_{t+1|t} &= AX_{t|t} + W_t \\
 &= N(E(AX_{t|t} + W_t), \text{Var}(AX_{t|t} + W_t)) \\
 &= N(AE(X_{t|t}) + E(W_t), A \text{Var}(X_{t|t})A^T + \text{Var}(W_t)) \\
 &= N(A\hat{\mathbf{x}}_{t|t}, AP_{t|t}A^T + Q)
 \end{aligned}$$

- So:

$$\begin{aligned}
 \hat{\mathbf{x}}_{t+1|t} &= A\hat{\mathbf{x}}_{t|t} \\
 P_{t+1|t} &= AP_{t|t}A^T + Q
 \end{aligned}$$



Measurement update

From $X_{t+1|t}$ (and given \mathbf{y}_{t+1}), compute $X_{t+1|t+1}$.

1. Compute **a priori** distribution of the observation

$Y_{t+1|t}$ from $X_{t+1|t}$:

$$\begin{aligned}
 Y_{t+1|t} &= CX_{t+1|t} + V_{t+1} \\
 &= N(E(CX_{t+1|t} + V_{t+1}), \text{Var}(CX_{t+1|t} + V_{t+1})) \\
 &= N(CE(X_{t+1|t}) + E(V_{t+1}), C \text{Var}(X_{t+1|t})C^T + \text{Var}(V_{t+1})) \\
 &= N(C\hat{\mathbf{x}}_{t+1|t}, CP_{t+1|t}C^T + R)
 \end{aligned}$$



Measurement update (cont'd)

2. Look at joint distribution of $X_{t+1|t}$ and $Y_{t+1|t}$:

$$\begin{aligned} (X_{t+1|t}, Y_{t+1|t}) &= N\left(\begin{pmatrix} E(X_{t+1|t}) \\ E(Y_{t+1|t}) \end{pmatrix}, \begin{pmatrix} \text{Var}(X_{t+1|t}) & \text{Cov}(X_{t+1|t}, Y_{t+1|t}) \\ \text{Cov}(Y_{t+1|t}, X_{t+1|t}) & \text{Var}(Y_{t+1|t}) \end{pmatrix}\right) \\ &= N\left(\begin{pmatrix} \hat{\mathbf{x}}_{t+1|t} \\ C\hat{\mathbf{x}}_{t+1|t} \end{pmatrix}, \begin{pmatrix} P_{t+1|t} & P_{t+1|t}C^T \\ CP_{t+1|t} & CP_{t+1|t}C^T + R \end{pmatrix}\right) \end{aligned}$$

where

$$\begin{aligned} \text{Cov}(Y_{t+1}, X_{t+1|t}) &= \text{Cov}(CX_{t+1|t} + V_{t+1}, X_{t+1|t}) \\ &= C \text{Cov}(X_{t+1|t}, X_{t+1|t}) + \text{Cov}(V_{t+1}, X_{t+1|t}) \\ &= C \text{Var}(X_{t+1|t}) \\ &= CP_{t+1|t} \end{aligned}$$



Measurement update (cont'd)

- Recall that if

$$(Z_1, Z_2) = N\left(\begin{pmatrix} \mu_1 \\ \mu_2 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{12} \\ \Sigma_{21} & \Sigma_{22} \end{pmatrix}\right)$$

then

$$(Z_1 | Z_2 = \mathbf{z}_2) = N(\mu_1 + \Sigma_{12}\Sigma_{22}^{-1}(\mathbf{z}_2 - \mu_2), \Sigma_{11} - \Sigma_{12}\Sigma_{22}^{-1}\Sigma_{21})$$

3. Compute $X_{t+1|t+1} = (X_{t+1|t} | Y_{t+1|t} = \mathbf{y}_{t+1})$:

$$\begin{aligned} X_{t+1|t+1} &= (X_{t+1|t} | Y_{t+1|t} = \mathbf{y}_{t+1}) \\ &= N\left(\hat{\mathbf{x}}_{t+1|t} + P_{t+1|t}C^T(CP_{t+1|t}C^T + R)^{-1}(\mathbf{y}_{t+1} - C\hat{\mathbf{x}}_{t+1|t}), \right. \\ &\quad \left. P_{t+1|t} - P_{t+1|t}C^T(CP_{t+1|t}C^T + R)^{-1}CP_{t+1|t}\right) \end{aligned}$$



Measurement update (cont'd):

This can also (often) be written in terms of the **Kalman gain** matrix:

$$\begin{aligned}K_{t+1} &= P_{t+1|t} C^T (C P_{t+1|t} C^T + R)^{-1} \\ \hat{\mathbf{x}}_{t+1|t+1} &= \hat{\mathbf{x}}_{t+1|t} + K_{t+1} (\mathbf{y}_{t+1} - C \hat{\mathbf{x}}_{t+1|t}) \\ P_{t+1|t+1} &= P_{t+1|t} - K_{t+1} C P_{t+1|t}\end{aligned}$$



Initialization

- Choose distribution of initial state by picking $\underline{\mathbf{x}}_0$ and P_0
- Start with measurement update given measurement \mathbf{y}_0
- Choice for Q and R (identity)
 - small Q: dynamics “trusted” more
 - small R: measurements “trusted” more



(Bayesian) Kalman Filter Summary

I. Model:

$$\mathbf{x}_{t+1} = A\mathbf{x}_t + \mathbf{w}_t, \quad \mathbf{w}_t \sim W_t = N(\mathbf{0}, Q)$$

$$\mathbf{y}_t = C\mathbf{x}_t + \mathbf{v}_t, \quad \mathbf{v}_t \sim V_t = N(\mathbf{0}, R)$$

II. Algorithm: Repeat...

– Time update:

$$\hat{\mathbf{x}}_{t+1|t} = A\hat{\mathbf{x}}_{t|t}$$

$$P_{t+1|t} = AP_{t|t}A^T + Q$$

– Measurement update:

$$K_{t+1} = P_{t+1|t}C^T(CP_{t+1|t}C^T + R)^{-1}$$

$$\hat{\mathbf{x}}_{t+1|t+1} = \hat{\mathbf{x}}_{t+1|t} + K_{t+1}(\mathbf{y}_{t+1} - C\hat{\mathbf{x}}_{t+1|t})$$

$$P_{t+1|t+1} = P_{t+1|t} - K_{t+1}CP_{t+1|t}$$



(Bayesian) Kalman Filter Summary [II]

Take Aways:

- Kalman filter can be used in real time
- Use $\hat{\mathbf{x}}_{t|t}$'s as optimal estimate of state at time t, and use $P_{t|t}$ as a measure of uncertainty.

Extensions:

- Dynamic process with known **control input**
- **Non-linear** dynamic process
- **Kalman smoothing**: compute optimal estimate of state \mathbf{x}_t given all data $\mathbf{y}_1, \dots, \mathbf{y}_T$, with $T > t$ (not real-time).
- Automatic parameter (Q and R) fitting using EM-algorithm



Viterbi Algorithm

Based on Material from S Salzberg CMSC 828N

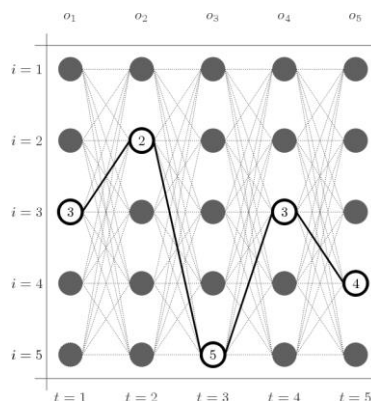
ELEC 3004: **Systems**

27 April 2017 **37**

The Viterbi Algorithm

Question: What does it do?

Answer: It finds the **most likely** sequence of hidden states:

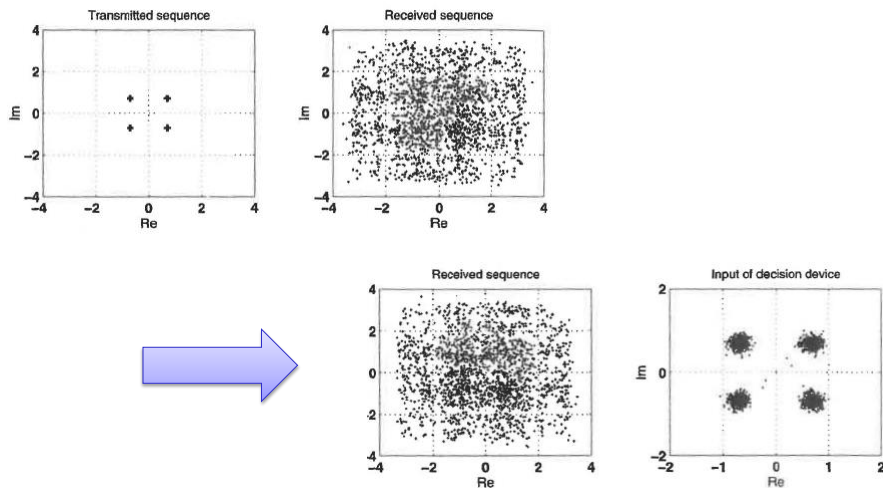


ELEC 3004: **Systems**

27 April 2017 **38**

What Sequence?

- Remember QPSK Constellations?



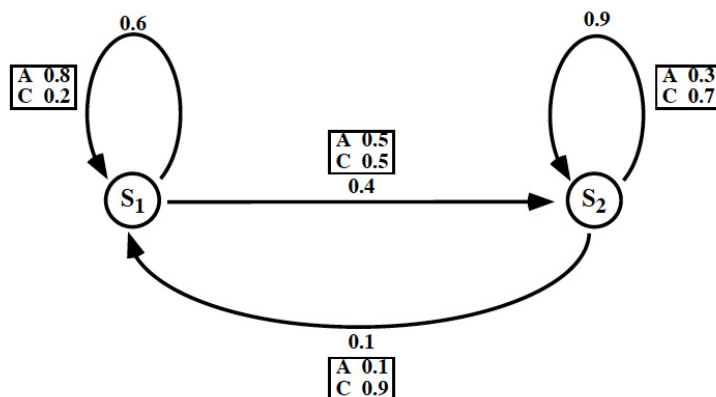
Viterbi algorithm

$$V_i(t) = \begin{cases} 0 & : t=0 \wedge i \neq S_I \\ 1 & : t=0 \wedge i = S_I \\ \max_j V_j(t-1) a_{ji} b_{ji}(y) & : t > 0 \end{cases}$$

Where $V_i(t)$ is the probability that the HMM is in state i after generating the sequence y_1, y_2, \dots, y_t , following the *most probable path* in the HMM



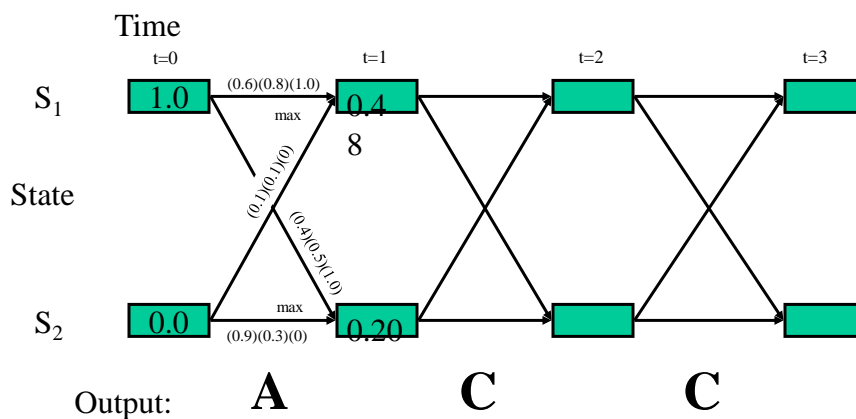
Our sample HMM



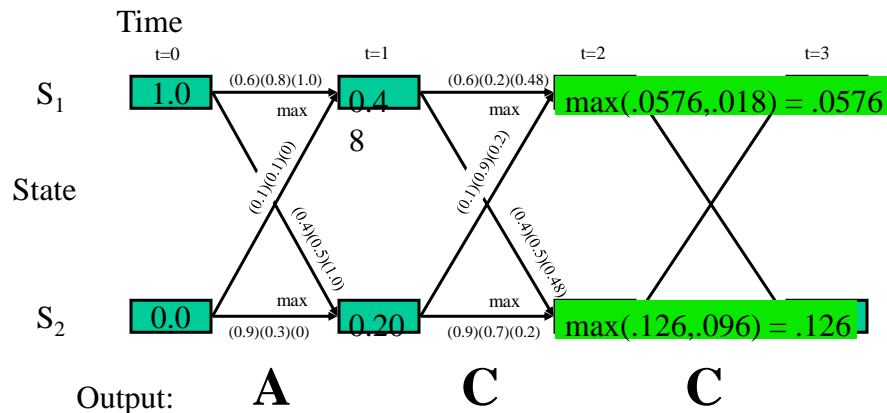
Let S_1 be initial state, S_2 be final state



A trellis for the Viterbi Algorithm



A trellis for the Viterbi Algorithm

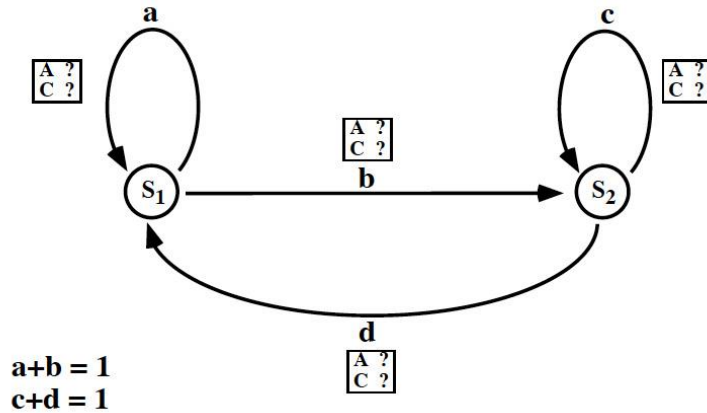


Learning in HMMs: the E-M algorithm

- In order to learn the parameters in an “empty” HMM, we need:
 - The topology of the HMM
 - Data - the more the better
- The learning algorithm is called “Estimate-Maximize” or E-M
 - Also called the Forward-Backward algorithm
 - Also called the Baum-Welch algorithm



An untrained HMM



Some HMM training data

- CACAACAAAACCCCCACAA
- ACAACACACACACACCAAAC
- CAACACACAAACCCC
- CAACCACCACACACACCCCA
- CCCAAAACCCCAAAAACCC
- ACACAAAAAACCCAACACACAACA
- ACACAACCCCAAAACCACCAAAAA



Step 1: Guess all the probabilities

- We can start with random probabilities, the learning algorithm will adjust them
- If we can make good guesses, the results will generally be better



Step 2: the Forward algorithm

- Reminder: each box in the trellis contains a value $\alpha_i(t)$
- $\alpha_i(t)$ is the probability that our HMM has generated the sequence y_1, y_2, \dots, y_t and has ended up in state i .



Reminder: notations

- sequence of length T: y_1^T
- all sequences of length T: Y_1^T
- Path of length T+1 generates Y: x_1^{T+1}
- All paths: X_1^{T+1}

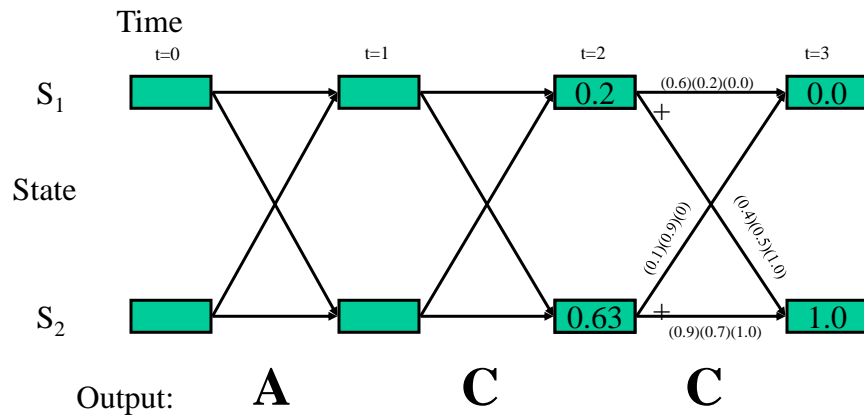


Step 3: the Backward algorithm

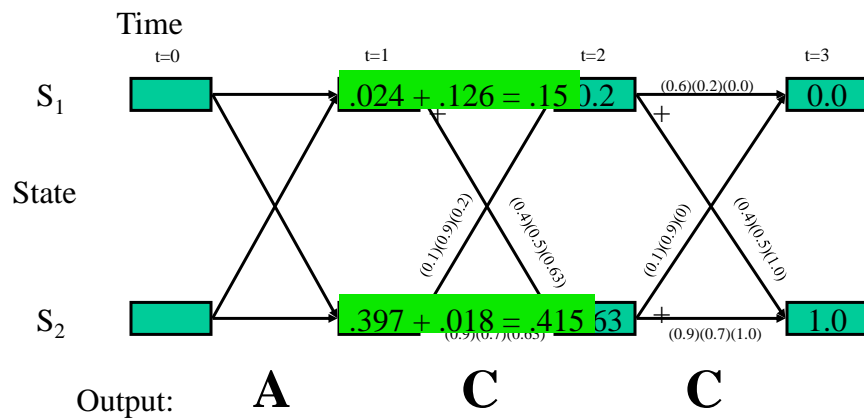
- Next we need to compute $\beta_i(t)$ using a Backward computation
- $\beta_i(t)$ is the probability that our HMM will generate the rest of the sequence $y_{t+1}, y_{t+2}, \dots, y_T$ beginning in state i



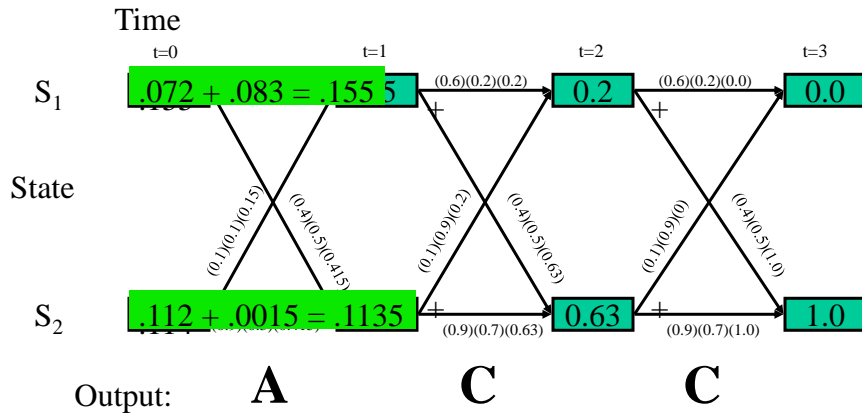
A trellis for the Backward Algorithm



A trellis for the Backward Algorithm (2)



A trellis for the Backward Algorithm (3)



Step 4: Re-estimate the probabilities

- After running the Forward and Backward algorithms once, we can re-estimate all the probabilities in the HMM
- α_{SF} is the prob. that the HMM generated the entire sequence
- Nice property of E-M: the value of α_{SF} never decreases; it converges to a local maximum
- We can read off α and β values from Forward and Backward trellises



Compute new transition probabilities

- γ is the probability of making transition i-j at time t, given the observed output
 - γ is dependent on data, plus it only applies for one time step; otherwise it is just like $a_{ij}(t)$

$$\gamma_{ij}(t) = P(X_t = i, X_{t+1} = j | y_1^T)$$

$$\gamma_{ij}(t) = \frac{\alpha_i(t-1) a_{ij} b_{ij}(y_t) \beta_j(t)}{\alpha_{S_F}}$$



What is gamma?

- Sum γ over all time steps, then we get the expected number of times that the transition i-j was made while generating the sequence Y:

$$C_1 = \sum_{t=1}^T \gamma_{ij}(t)$$



How many times did we leave i?

- Sum γ over all time steps and all states that can follow i, then we get the expected number of times that the transition i-x as made for any state x:

$$C_2 = \sum_{t=1}^T \sum_k \gamma_{ik}(t)$$



Recompute transition probability

$$a_{ij} = \frac{C_1}{C_2}$$

In other words, probability of going from state i to j is estimated by counting how often we took it for our data (C1), and dividing that by how often we went from i to other states (C2)



Recompute output probabilities

- Originally these were $b_{ij}(k)$ values
- We need:
 - expected number of times that we made the transition $i \rightarrow j$ and emitted the symbol k
 - The expected number of times that we made the transition $i \rightarrow j$



New estimate of $b_{ij}(k)$

$$b_{ij}(k) = \frac{\sum_{t: y_t = k} \gamma_{ij}(t)}{\sum_{t=1}^T \gamma_{ij}(t)}$$



Step 5: Go to step 2

- Step 2 is Forward Algorithm
- Repeat entire process until the probabilities converge
 - Usually this is rapid, 10-15 iterations
- “Estimate-Maximize” because the algorithm first estimates probabilities, then maximizes them based on the data
- “Forward-Backward” refers to the two computationally intensive steps in the algorithm

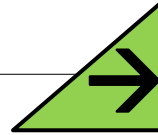


Computing requirements

- Trellis has N nodes per column, where N is the number of states
- Trellis has S columns, where S is the length of the sequence
- Between each pair of columns, we create E edges, one for each transition in the HMM
- Total trellis size is approximately $S(N+E)$



Next Time...



- **State-Space & Digital Control!**
 - State-space filters add feedback and become a “control filter”



Summary

- FT of sampled data is known as
 - discrete-time Fourier transform (DTFT)
 - discrete in time
 - continuous & periodic in frequency
- DFT is sampled version of DTFT
 - discrete in both time and frequency
 - periodic in both time and frequency
 - due to sampling in both time and frequency
- DFT is implemented using the FFT
- Leakage reduced (dynamic range increased)
 - with non-rectangular window functions

