Administrivia

- HW1 scores out
  - Some students (who got really poor marks) given chance to upgrade
    - Make it all the way to the 50 percentile for each problem

- HW2 scores to be out by next week

- Please send us project updates
Recap: An automotive example

• Determine automatically, by only *listening* to a running automobile, if it is:
  – Idling; or
  – Travelling at constant velocity; or
  – Accelerating; or
  – Decelerating

• Assume (for illustration) that we only record energy level (SPL) in the sound
  – The SPL is measured once per second
The state-space model
- Assuming all transitions from a state are equally probable
Overall procedure

\[
P(S_T | x_{0:T-1}) = \sum_{S_{T-1}} P(S_{T-1} | x_{0:T-1}) P(S_T | S_{T-1})
\]

• At \( T=0 \) the predicted state distribution is the initial state probability
• At each time \( T \), the current estimate of the distribution over states considers \textit{all} observations \( x_0 \ldots x_T \)
  – A natural outcome of the Markov nature of the model
• The prediction+update is identical to the forward computation for HMMs to within a normalizing constant
Estimating the state

- The state is estimated from the updated distribution
  - The updated distribution is propagated into time, not the state

Estimate($S_T$) = argmax$_{S_T} P(S_T | x_{0:T})$

P($S_T | x_{0:T-1}$) = $\sum_{S_{T-1}} P(S_{T-1} | x_{0:T-1}) P(S_T | S_{T-1})$

P($S_T | x_{0:T}$) = C. P($S_T | x_{0:T-1}$) P($x_T | S_T$)

T=T+1

Predict the distribution of the state at T

Update the distribution of the state at T after observing $x_T$

Estimate($S_T$)
Predicting the next observation

- The probability distribution for the observations at the next time is a mixture:
  \[
  P(x_T|\mathbf{x}_{0:T-1}) = \sum_{S_T} P(x_T|S_T) P(S_T|\mathbf{x}_{0:T-1})
  \]
- The actual observation can be predicted from \( P(x_T|\mathbf{x}_{0:T-1}) \)
Continuous state system

• The state is a continuous valued parameter that is not directly seen
  – The state is the position of navlab or the star

• The observations are dependent on the state and are the only way of knowing about the state
  – Sensor readings (for navlab) or recorded image (for the telescope)
Discrete vs. Continuous State Systems

Prediction at time $t$:

$$P(s_t \mid O_{0:t-1}) = \sum_{s_{t-1}} P(s_{t-1} \mid O_{0:t-1})P(s_t \mid s_{t-1})$$

Update after $O_t$:

$$P(s_t \mid O_{0:t}) = CP(s_t \mid O_{0:t-1})P(O_t \mid s_t)$$

$$P(s_t \mid O_{0:t}) = \int_{-\infty}^{\infty} P(s_{t-1} \mid O_{0:t-1})P(s_t \mid s_{t-1})ds_{t-1}$$

$$P(s_t \mid O_{0:t}) = CP(s_t \mid O_{0:t-1})P(O_t \mid s_t)$$

$s_t = f(s_{t-1}, \varepsilon_t)$

$o_t = g(s_t, \gamma_t)$
Special case: Linear Gaussian model

\[ s_t = A_t s_{t-1} + \varepsilon_t \]

\[ o_t = B_t s_t + \gamma_t \]

- A **linear** state dynamics equation
  - Probability of state driving term \( \varepsilon \) is Gaussian
  - Sometimes viewed as a driving term \( \mu_\varepsilon \) and additive zero-mean noise

- A **linear** observation equation
  - Probability of observation noise \( \gamma \) is Gaussian

- \( A_t, B_t \) and Gaussian parameters assumed known
  - May vary with time

\[
P(\varepsilon) = \frac{1}{\sqrt{(2\pi)^d |\Theta_\varepsilon|}} \exp\left(-0.5(\varepsilon - \mu_\varepsilon)^T \Theta_\varepsilon^{-1}(\varepsilon - \mu_\varepsilon)\right)
\]

\[
P(\gamma) = \frac{1}{\sqrt{(2\pi)^d |\Theta_\gamma|}} \exp\left(-0.5(\gamma - \mu_\gamma)^T \Theta_\gamma^{-1}(\gamma - \mu_\gamma)\right)
\]
The Linear Gaussian model (KF)

\[ P_0(s) = \text{Gaussian}(s; \bar{s}, R) \]
\[ P(s_t \mid s_{t-1}) = \text{Gaussian}(s_t; \mu_\varepsilon + A_t s_{t-1}, \Theta_\varepsilon) \]
\[ P(o_t \mid s_t) = \text{Gaussian}(o_t; B_t s_t, \Theta_\gamma) \]

\[ P(s_t \mid o_{0:t-1}) = \text{Gaussian}(s; \bar{s}_t, R_t) \]
\[ P(s_t \mid o_{0:t}) = \text{Gaussian}(s; \hat{s}_t, \hat{R}_t) \]

\[ \hat{s}_t = \bar{s}_t + R_t B_t^T (B_t R_t B_t^T + \Theta_\gamma)^{-1} (o - B_t \bar{s}_t) \]
\[ \hat{R}_t = (I - R_t B_t^T (B_t R_t B_t^T + \Theta_\gamma)^{-1} B_t) R_t \]

- Iterative prediction and update
The Kalman filter

• Prediction

\[ s_t = A_t s_{t-1} + \varepsilon_t \]

\[ \bar{s}_t = A_t \hat{s}_{t-1} + \mu \varepsilon \]

\[ R_t = \Theta \varepsilon + A_t \hat{R}_{t-1} A_t^T \]

• Update

\[ K_t = R_t B_t^T \left( B_t R_t B_t^T + \Theta \gamma \right)^{-1} \]

\[ \hat{s}_t = \bar{s}_t + K_t (o_t - B_t \bar{s}_t) \]

\[ \hat{R}_t = (I - K_t B_t) R_t \]
The Kalman filter

• Prediction

\[ s_t = A_t s_{t-1} + \varepsilon_t \]

\[ o_t = B_t s_t + \gamma_t \]

The *predicted* state at time \( t \) is obtained simply by propagating the estimated state at \( t-1 \) through the state dynamics equation.

\[ \hat{s}_t = \bar{s}_t + K_t (o_t - B_t \bar{s}_t) \]

\[ \hat{R}_t = (I - K_t B_t) R_t \]
The Kalman filter

- Prediction

\[ s_t = A_t s_{t-1} + \varepsilon_t \]

\[ o_t = B_t s_t + \gamma_t \]

\[ R_t = \Theta \varepsilon + A_t \hat{R}_{t-1} A_t^T \]

The prediction is imperfect. The variance of the predictor = variance of \( \varepsilon_t \) + variance of \( A s_{t-1} \)

The two simply add because \( \varepsilon_t \) is not correlated with \( s_t \)
The Kalman filter

• Prediction

\[ \bar{s}_t = A_t \hat{s}_{t-1} + \mu_{\varepsilon} \]

\[ R_t = \Theta_{\varepsilon} + A_t \hat{R}_{t-1} A_t^T \]

\[ o_t = B_t s_t + \gamma_t \]

\[ \hat{o}_t = B_t \bar{s}_t \]

We can also predict the observation from the predicted state using the observation equation

\[ s_t = s_t + K_t (o_t - B_t \bar{s}_t) \]

\[ \hat{R}_t = (I - K_t B_t) R_t \]
MAP Recap (for Gaussians)

- If $P(x,y)$ is Gaussian:

$$P(x,y) = N\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} C_{xx} & C_{xy} \\ C_{yx} & C_{yy} \end{bmatrix}\right)$$

$$P(y|x) = N(\mu_y + C_{yx} C^{-1}_{xx}(x - \mu_x), C_{yy} - C_{yx} C^{-1}_{xx} C_{xy})$$

$$\hat{y} = \mu_y + C_{yx} C^{-1}_{xx}(x - \mu_x)$$
MAP Recap: For Gaussians

- If $P(x,y)$ is Gaussian:

$$P(y, x) = N\left(\begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}, \begin{bmatrix} C_{xx} & C_{xy} \\ C_{yx} & C_{yy} \end{bmatrix}\right)$$

$$P(y \mid x) = N(\mu_y + C_{yx} C^{-1}_{xx} (x - \mu_x), C_{yy} - C_{yx}^{T} C_{xx}^{-1} C_{xy})$$

$$\hat{y} = \mu_y + C_{yx} C^{-1}_{xx} (x - \mu_x)$$

“Slope” of the line
The Kalman filter

• Prediction

\[ s_t = A_t s_{t-1} + \varepsilon_t \]

\[ \bar{s}_t = A_t \hat{s}_{t-1} + \mu_\varepsilon \]

\[ R_t = \Theta_\varepsilon + A_t \hat{R}_{t-1} A_t^T \]

• Update

\[ o_t = B_t s_t + \gamma_t \]

\[ K_t = R_t B_t^T \left( B_t R_t B_t^T + \Theta_\gamma \right)^{-1} \]

This is the slope of the MAP estimator that predicts \( s \) from \( o \)

\[ R B^T = C_{so}, \quad (B R B^T + \Theta) = C_{oo} \]

This is also called the Kalman Gain
The Kalman filter

- **Prediction**

\[ s_t = A_t s_{t-1} + \varepsilon_t \]

\[ R_t = \Theta_{\varepsilon} + A_t \hat{R}_{t-1} A_t^T \]

\[ \tilde{s}_t = A_t \hat{s}_{t-1} + \mu_{\varepsilon} \]

**We must correct the predicted value of the state after making an observation**

\[ \hat{s}_t = \tilde{s}_t + K_t (o_t - B_t \tilde{s}_t) \]

\[ \hat{o}_t = B_t \tilde{s}_t \]

The correction is the difference between the **actual** observation and the **predicted** observation, scaled by the Kalman Gain.
The Kalman filter

• Prediction

\[ \bar{s}_t = A_t \hat{s}_{t-1} + \mu_\varepsilon \]

\[ R_t = \Theta_\varepsilon + A_t \hat{R}_{t-1} A_t^T \]

\[ s_t = A_t s_{t-1} + \varepsilon_t \]

• Update:

The uncertainty in state decreases if we observe the data and make a correction

The reduction is a multiplicative “shrinkage” based on Kalman gain and B

\[ \hat{R}_t = (I - K_t B_t) R_t \]

\[ \hat{o}_t = B_t \bar{s}_t \]
The Kalman filter

- Prediction
  \[ \bar{s}_t = A_t \hat{s}_{t-1} + \mu_\varepsilon \]

- Update:
  \[ R_t = \Theta \varepsilon + A_t \hat{\mathbf{R}}_{t-1} A^T_t \]

- Update
  \[ K_t = R_t B_t^T \left( B_t R_t B_t^T + \Theta \gamma \right)^{-1} \]

  \[ \hat{s}_t = \bar{s}_t + K_t (o_t - B_t \bar{s}_t) \]

  \[ \hat{\mathbf{R}}_t = \left( I - K_t B_t \right) R_t \]

\[ s_t = A_t s_{t-1} + \varepsilon_t \]

\[ o_t = B_t s_t + \gamma_t \]
Linear Gaussian Model

\[ s_t = A_t s_{t-1} + \epsilon_t \]
\[ o_t = B_t s_t + \gamma_t \]

All distributions remain Gaussian
Problems

\[ s_t = f(s_{t-1}, \varepsilon_t) \]
\[ o_t = g(s_t, \gamma_t) \]

• \( f() \) and/or \( g() \) may not be nice linear functions
  – Conventional Kalman update rules are no longer valid

• \( \varepsilon \) and/or \( \gamma \) may not be Gaussian
  – Gaussian based update rules no longer valid
Problems

\[ s_t = f(s_{t-1}, \varepsilon_t) \]
\[ o_t = g(s_t, \gamma_t) \]

- \( f() \) and/or \( g() \) may not be nice linear functions
  - Conventional Kalman update rules are no longer valid

- \( \varepsilon \) and/or \( \gamma \) may not be Gaussian
  - Gaussian based update rules no longer valid
The problem with non-linear functions

\[ s_t = f(s_{t-1}, \mathcal{E}_t) \]

\[ o_t = g(s_t, \gamma_t) \]

\[ P(s_t \mid o_{0:t-1}) = \int_{-\infty}^{\infty} P(s_{t-1} \mid o_{0:t-1}) P(s_t \mid s_{t-1}) ds_{t-1} \]

\[ P(s_t \mid o_{0:t}) = Cp(s_t \mid o_{0:t-1}) P(o_t \mid s_t) \]

- Estimation requires knowledge of \( P(o \mid s) \)
  - Difficult to estimate for nonlinear \( g() \)
  - Even if it can be estimated, may not be tractable with update loop

- Estimation also requires knowledge of \( P(s_t \mid s_{t-1}) \)
  - Difficult for nonlinear \( f() \)
  - May not be amenable to closed form integration
The problem with nonlinearity

\[ o_t = g(s_t, \gamma_t) \]

- The PDF may not have a closed form

\[ P(o_t \mid s_t) = \sum_{\gamma : g(s_t, \gamma) = o_t} \frac{P(\gamma)}{|J_{g(s_t, \gamma)}(o_t)|} \]

\[ |J_{g(s_t, \gamma)}(o_t)| = \begin{vmatrix} \frac{\partial o_t(1)}{\partial \gamma(1)} & \cdots & \frac{\partial o_t(1)}{\partial \gamma(n)} \\ \vdots & \ddots & \vdots \\ \frac{\partial o_t(n)}{\partial \gamma(1)} & \cdots & \frac{\partial o_t(n)}{\partial \gamma(n)} \end{vmatrix} \]

- Even if a closed form exists initially, it will typically become intractable very quickly
Example: a simple nonlinearity

\[ o = \gamma + \log(1 + \exp(s)) \]

- \( P(o \mid s) = ? \)
  - Assume \( \gamma \) is Gaussian
  - \( P(\gamma) = Gaussian(\gamma; \mu_\gamma, \Theta_\gamma) \)
Example: a simple nonlinearity

\[ o = \gamma + \log(1 + \exp(s)) \]

- \( P(o \mid s) = ? \)

\[
P(\gamma) = \text{Gaussian}(\gamma; \mu_\gamma, \Theta_\gamma)
\]

\[
P(o \mid s) = \text{Gaussian}(o; \mu_\gamma + \log(1 + \exp(s)), \Theta_\gamma)
\]
Example: At $T=0$.

\[ o = \gamma + \log(1 + \exp(s)) \]

- Assume initial probability $P(s)$ is Gaussian

\[ P(s_0) = P_0(s) = \text{Gaussian}(s; \bar{s}, R) \]

- Update

\[ P(s_0 \mid o_0) = CP(o_0 \mid s_0)P(s_0) \]

\[ P(s_0 \mid o_0) = C\text{Gaussian}(o; \mu_\gamma + \log(1 + \exp(s_0)), \Theta_\gamma) \text{Gaussian}(s_0; \bar{s}, R) \]
**UPDATE: At T=0.**

\[ o = \gamma + \log(1 + \exp(s)) \]

\[ P(s_0 \mid o_0) = CGaussian(o; \mu_\gamma + \log(1 + \exp(s_0)), \Theta_\gamma)Gaussian(s_0; \bar{s}, R) \]

\[
P(s_0 \mid o_0) = C \exp \left( -0.5(\mu_\gamma + \log(1 + \exp(s_0)) - o)^T \Theta_\gamma^{-1}(\mu_\gamma + \log(1 + \exp(s_0)) - o) - 0.5(s_0 - \bar{s})^T R^{-1}(s_0 - \bar{s}) \right)\]

- = Not Gaussian
Prediction for $T = 1$

\[ S_t = S_{t-1} + \varepsilon \]

\[ P(\varepsilon) = \text{Gaussian}(\varepsilon; 0, \Theta_{\varepsilon}) \]

- Trivial, linear state transition equation

\[ P(s_t \mid s_{t-1}) = \text{Gaussian}(s_t; s_{t-1}, \Theta_{\varepsilon}) \]

- Prediction

\[ P(s_1 \mid o_0) = \int_{-\infty}^{\infty} P(s_0 \mid o_0) P(s_1 \mid s_0) ds_0 \]

\[ P(s_1 \mid o_0) = \int_{-\infty}^{\infty} C \exp \left( -0.5(\mu_\gamma + \log(1+\exp(s_0)) - o)^T \Theta_\gamma^{-1} (\mu_\gamma + \log(1+\exp(s_0)) - o) \right. \]
\[ \left. -0.5(s_0 - s)^T R^{-1} (s_0 - s) \right) \exp \left( (s_1 - s_0)^T \Theta_{\varepsilon}^{-1} (s_1 - s_0) \right) ds_0 \]

- = intractable
Update at T=1 and later

• Update at T=1

\[ P(s_t \mid o_{0:t}) = CP(s_t \mid o_{0:t-1})P(o_t \mid s_t) \]

– Intractable

• Prediction for T=2

\[ P(s_t \mid o_{0:t-1}) = \int_{-\infty}^{\infty} P(s_{t-1} \mid o_{0:t-1})P(s_t \mid s_{t-1})ds_{t-1} \]

– Intractable
The State prediction Equation

\[ s_t = f(s_{t-1}, \varepsilon_t) \]

- Similar problems arise for the state prediction equation

- \( P(s_t|s_{t-1}) \) may not have a closed form

- Even if it does, it may become intractable within the prediction and update equations
  - Particularly the prediction equation, which includes an integration operation
Simplifying the problem: Linearize

- The tangent at any point is a good local approximation if the function is sufficiently smooth.

$$o = \gamma + \log(1 + \exp(s))$$
Simplifying the problem: Linearize

- The tangent at any point is a good local approximation if the function is sufficiently smooth.

\[ o = \gamma + \log(1 + \exp(s)) \]
Simplifying the problem: Linearize

- The **tangent** at any point is a good **local** approximation if the function is sufficiently smooth

\[ o = \gamma + \log(1 + \exp(s)) \]
Simplifying the problem: Linearize

- The tangent at any point is a good local approximation if the function is sufficiently smooth.
Linearizing the observation function

\[ P(s) = \text{Gaussian}(\bar{s}, R) \]

\[ o = \gamma + g(s) \quad \rightarrow \quad o \approx \gamma + g(\bar{s}) + J_g(\bar{s})(s - \bar{s}) \]

- Simple first-order Taylor series expansion
  - \( J() \) is the Jacobian matrix
    - Simply a determinant for scalar state

- Expansion around a priori (or predicted) mean of the state
Most probability is in the low-error region

\[ P(s) = \text{Gaussian}(\bar{s}, R) \]

- \( P(s) \) is small approximation error is large
  - Most of the probability mass of \( s \) is in low-error regions
Linearizing the observation function

\[ P(s) = \text{Gaussian}(\bar{s}, R) \]

\[ o = \gamma + g(s) \quad \rightarrow \quad o \approx \gamma + g(\bar{s}) + J_g(\bar{s})(s - \bar{s}) \]

• Observation PDF is Gaussian

\[ P(\gamma) = \text{Gaussian}(\gamma; 0, \Theta_\gamma) \]

\[ P(o \mid s) = \text{Gaussian}(o; g(\bar{s}) + J_g(\bar{s})(s - \bar{s}), \Theta_\gamma) \]
UPDATE.

\[ o \approx \gamma + g(\bar{s}) + J_g(\bar{s})(s - \bar{s}) \]

\[
P(o \mid s) = \text{Gaussian}(o; g(\bar{s}) + J_g(\bar{s})(s - \bar{s}), \Theta_\gamma)
\]

\[
P(s) = \text{Gaussian}(s; \bar{s}, R) \quad P(s \mid o) = CP(o \mid s)P(s)
\]

\[
P(s \mid o) = \text{CGaussian}(o; g(\bar{s}) + J_g(\bar{s})(s - \bar{s}), \Theta_\gamma)\text{Gaussian}(s; \bar{s}, R)
\]

\[
P(s \mid o) = \text{Gaussian}\left(s; \bar{s} + RJ_g(\bar{s})^T(J_g(\bar{s})RJ_g(\bar{s})^T + \Theta_\gamma)^{-1}(o - g(\bar{s})), \left(I - RJ_g(\bar{s})^T(J_g(\bar{s})RJ_g(\bar{s})^T + \Theta_\gamma)^{-1}J_g(\bar{s})\right)R \right)
\]

- **Gaussian!!**
  - Note: This is actually only an approximation
Prediction?

\[ s_t = f(s_{t-1}) + \epsilon \]

\[ P(\epsilon) = Gaussian(\epsilon; 0, \Theta_\epsilon) \]

- Again, direct use of \( f() \) can be disastrous

- Solution: Linearize

\[ P(s_{t-1} | o_{0:t-1}) = Gaussian(s_{t-1}; \hat{s}_{t-1}, \hat{R}_{t-1}) \]

\[ s_t = f(s_{t-1}) + \epsilon \]

\[ s_t \approx \epsilon + f(\hat{s}_{t-1}) + J_f(\hat{s}_{t-1})(s_{t-1} - \hat{s}_{t-1}) \]

- Linearize around the mean of the updated distribution of \( s \) at \( t-1 \)

- Which should be Gaussian
The state transition probability is now:

\[ s_t = f(s_{t-1}) + \varepsilon \]

\[ s_t \approx \varepsilon + f(\hat{s}_{t-1}) + J_f(\hat{s}_{t-1})(s_{t-1} - \hat{s}_{t-1}) \]

\[ P(s_{t-1} \mid o_{0:t-1}) = \text{Gaussian}(s_{t-1} ; \hat{s}_{t-1}, \hat{R}_{t-1}) \quad P(\varepsilon) = \text{Gaussian}(\varepsilon;0, \Theta_\varepsilon) \]

- The state transition probability is now:

\[ P(s_t \mid s_{t-1}) = \text{Gaussian}(s_t ; f(\hat{s}_{t-1}) + J_f(\hat{s}_{t-1})(s_{t-1} - \hat{s}_{t-1}), \Theta_\varepsilon) \]

- The predicted state probability is:

\[ P(s_t \mid o_{0:t-1}) = \int_{-\infty}^{\infty} P(s_{t-1} \mid o_{0:t-1})P(s_t \mid s_{t-1}) ds_{t-1} \]
The predicted state probability is:

\[
P(s_t | o_{0:t-1}) = Gaussian\left(s_t; \hat{f}(s_{t-1}), J_f(\hat{s}_{t-1})\hat{R}_{t-1}J_f(\hat{s}_{t-1})^T + \Theta_\epsilon\right)
\]

- Gaussian!!
  - This is actually only an approximation
The linearized prediction/update

- Given: two non-linear functions for state update and observation generation

- Note: the equations are deterministic non-linear functions of the state variable
  - They are linear functions of the noise!
  - Non-linear functions of stochastic noise are slightly more complicated to handle

\[
\begin{align*}
  o_t &= g(s_t) + \gamma \\
  s_t &= f(s_{t-1}) + \varepsilon
\end{align*}
\]
Linearized Prediction and Update

- Prediction for time $t$

$$\tilde{s}_t = f(\hat{s}_{t-1})$$

$$R_t = J_f(\hat{s}_{t-1}) \hat{R}_{t-1} J_f(\hat{s}_{t-1})^T + \Theta$$

- Update at time $t$

$$\hat{s}_t = \tilde{s}_t + R_t J_g(\tilde{s}_t)^T (J_g(\tilde{s}_t)R_t J_g(\tilde{s}_t)^T + \Theta)^{-1}(o_t - g(\tilde{s}_t))$$

$$\hat{R}_t = \left(I - R_t J_g(\tilde{s}_t)^T (J_g(\tilde{s}_t)R_t J_g(\tilde{s}_t)^T + \Theta)^{-1} J_g(\tilde{s}_t)\right)R_t$$
Linearized Prediction and Update

• Prediction for time $t$

$$P(s_t \mid o_{0:t-1}) = \text{Gaussian}(s_t; \bar{s}_t, R_t)$$

$$\bar{s}_t = f(\hat{s}_{t-1}) \quad R_t = A_t \hat{R}_{t-1} A_t^T + \Theta \varepsilon$$

■ Update at time $t$

$$P(s_t \mid o_{0:t}) = \text{Gaussian}(s_t; \hat{s}_t, \hat{R}_t)$$

$$\hat{s}_t = \bar{s}_t + R_t B_t^T (B_t R_t B_t^T + \Theta \gamma)^{-1} (o_t - g(\bar{s}_t))$$

$$\hat{R}_t = \left( I - R_t B_t^T (B_t R_t B_t^T + \Theta \gamma)^{-1} B_t \right) R_t$$

$$A_t = J_f(\hat{s}_{t-1})$$

$$B_t = J_g(\bar{s}_t)$$
The Extended Kalman filter

• Prediction

\[ \bar{s}_t = f(\hat{s}_{t-1}) \]

\[ R_t = \Theta_\varepsilon + A_t \hat{R}_{t-1} A_t^T \]

• Update

\[ K_t = R_t B_t^T (B_t R_t B_t^T + \Theta_\gamma)^{-1} \]

\[ \hat{s}_t = \bar{s}_t + K_t (o_t - g(\bar{s}_t)) \]

\[ \hat{R}_t = (I - K_t B_t) R_t \]
The Kalman filter

- **Prediction**

\[ \hat{s}_t = A_t \hat{s}_{t-1} + \mu_e \]

\[ R_t = \Theta \gamma + A_t \hat{R}_{t-1} A_t^T \]

- **Update**

\[ K_t = R_t B_t^T \left( B_t R_t B_t^T + \Theta \gamma \right)^{-1} \]

\[ \hat{s}_t = \bar{s}_t + K_t \left( o_t - B_t \bar{s}_t \right) \]

\[ \hat{R}_t = \left( I - K_t B_t \right) R_t \]
The Extended Kalman filter

• Prediction

\[ \bar{s}_t = f(\hat{s}_{t-1}) \]

\[ R_t = \Theta \varepsilon + A_t \hat{R}_{t-1} A_t^T \]

• Update

\[ K_t = R_t B_t^T \left( B_t R_t B_t^T + \Theta \gamma \right)^{-1} \]

\[ \hat{s}_t = \bar{s}_t + K_t \left( o_t - g(\bar{s}_t) \right) \]

\[ \hat{R}_t = \left( I - K_t B_t \right) R_t \]

\[ s_t = f(s_{t-1}) + \varepsilon \]

\[ o_t = g(s_t) + \varepsilon \]

\[ A_t = J_f(\hat{s}_{t-1}) \]

\[ B_t = J_g(\bar{s}_t) \]
The Extended Kalman filter

• Prediction

The predicted state at time $t$ is obtained simply by propagating the estimated state at $t-1$ through the state dynamics equation

$$s_t = f(s_{t-1}) + \epsilon$$

$$o_t = g(s_t) + \epsilon$$

$$\hat{s}_t = \bar{s}_t + K_t (o_t - g(\bar{s}_t))$$

$$\hat{R}_t = (I - K_t B_t) R_t$$
The Extended Kalman filter

• Prediction

\[ \hat{s}_t = f(\hat{s}_{t-1}) \]

\[ R_t = \Theta \varepsilon + A_t \hat{R}_{t-1} A_t^T \]

The prediction is imperfect. The variance of the predictor = variance of \( \varepsilon_t \) + variance of \( A s_{t-1} \)

\[ A_t = J_f(\hat{s}_{t-1}) \]

\[ s_t = f(s_{t-1}) + \varepsilon \]

\[ o_t = g(s_t) + \varepsilon \]

\[ R_t = I_t(\hat{s}_t) \]

\[ \begin{align*} R_t &= (I_t - K_t D_t) R_{t-1} \end{align*} \]
The Extended Kalman filter

- **Prediction**

\[
\hat{s}_t = f(\hat{s}_{t-1})
\]

\[
R_t = \Theta_\epsilon + A_t \hat{R}_{t-1} A_t^T
\]

- **Update**

\[
s_t = f(s_{t-1}) + \epsilon
\]

\[
o_t = g(s_t) + \epsilon
\]

\[
B_t = J_g(\bar{s}_t)
\]

\[
K_t = R_t B_t^T \left( B_t R_t B_t^T + \Theta_\gamma \right)^{-1}
\]

The Kalman gain is the slope of the MAP estimator that predicts \(s\) from \(o\)

\[
R B T = C_{so}, \quad (BRB^T + \Theta) = C_{oo}
\]

\(B\) is obtained by linearizing \(g()\)
The Extended Kalman filter

- Prediction

\[ \overline{s}_t = f(\hat{s}_{t-1}) \]

\[ s_t = f(s_{t-1}) + \epsilon \]

\[ o_t = g(s_t) + \epsilon \]

\[ R_t = \Theta \epsilon + A_t \hat{R}_{t-1} A_t^T \]

We can also predict the observation from the predicted state using the observation equation

\[ \hat{s}_t = \overline{s}_t + K_t (o_t - g(\overline{s}_t)) \]

\[ \hat{R}_t = (I - K_t B_t) R_t \]

\[ \overline{o}_t = g(\overline{s}_t) \]
The Extended Kalman filter

- **Prediction**

\[
\hat{s}_t = f(\hat{s}_{t-1})
\]

\[
R_t = \Theta \varepsilon + A_t \hat{R}_{t-1} A_t^T
\]

We must correct the predicted value of the state after making an observation

\[
\hat{s}_t = \bar{s}_t + K_t (o_t - g(\bar{s}_t))
\]

\[
\bar{o}_t = g(\bar{s}_t)
\]

The correction is the difference between the actual observation and the predicted observation, scaled by the Kalman Gain

\[
s_t = f(s_{t-1}) + \varepsilon
\]

\[
o_t = g(s_t) + \varepsilon
\]
The Extended Kalman filter

- **Prediction**

\[ \tilde{s}_t = f(\hat{s}_{t-1}) \]

\[ R_t = \Theta \epsilon + A_t \hat{R}_{t-1} A_t^T \]

\[ B_t = J_g(\tilde{s}_t) \]

The uncertainty in state decreases if we observe the data and make a correction

The reduction is a multiplicative “shrinkage” based on Kalman gain and B

\[ \hat{R}_t = (I - K_t B_t) R_t \]

\[ s_t = f(s_{t-1}) + \epsilon \]

\[ o_t = g(s_t) + \epsilon \]
The Extended Kalman filter

• Prediction

\[ \ddot{s}_t = f(\dot{s}_{t-1}) \]

\[ R_t = \Theta \varepsilon + A_t \ddot{R}_{t-1} A_t^T \]

• Update

\[ K_t = R_t B_t^T \left( B_t R_t B_t^T + \Theta_\gamma \right)^{-1} \]

\[ \dot{s}_t = \ddot{s}_t + K_t \left( o_t - g(\ddot{s}_t) \right) \]

\[ \hat{R}_t = \left( I - K_t B_t \right) R_t \]
EKF

• EKFs are probably the most commonly used algorithm for tracking and prediction
  – Most systems are non-linear
  – Specifically, the relationship between state and observation is usually nonlinear
  – The approach can be extended to include non-linear functions of noise as well

• The term “Kalman filter” often simply refers to an extended Kalman filter in most contexts.

• But..
EKF have limitations

- If the non-linearity changes too quickly with $s$, the linear approximation is invalid
  - Unstable
- The estimate is often biased
  - The true function lies entirely on one side of the approximation

Various extensions have been proposed:
- Invariant extended Kalman filters (IEKF)
- Unscented Kalman filters (UKF)
A different problem: Non-Gaussian PDFs

\[ o_t = g(s_t) + \gamma \]
\[ s_t = f(s_{t-1}) + \varepsilon \]

• We have assumed so far that:
  – \( P_0(s) \) is Gaussian or can be approximated as Gaussian
  – \( P(\varepsilon) \) is Gaussian
  – \( P(\gamma) \) is Gaussian

• This has a happy consequence: All distributions remain Gaussian
Linear Gaussian Model

\[ s_t = A_t s_{t-1} + \mathcal{E}_t \]

\[ o_t = B_t s_t + \gamma_t \]

\[ P(s) = \]

\[ P(s_0|s_{-1}) = \]

\[ P(O_t|s_t) = \]

\[ P(s_0) = P(s) \]

\[ P(s_0|O_0) = C \, P(s_0) \, P(O_0|s_0) \]

\[ P(s_1|O_0) = \int_{-\infty}^{\infty} P(s_0|O_0) P(s_1|s_0) ds_0 \]

\[ P(s_1|O_{0:1}) = C \, P(s_1|O_0) \, P(O_1|s_0) \]

\[ P(s_2|O_{0:1}) = \int_{-\infty}^{\infty} P(s_1|O_{0:1}) P(s_2|s_1) ds_1 \]

\[ P(s_2|O_{0:2}) = C \, P(s_2|O_{0:1}) \, P(O_2|s_2) \]

All distributions remain Gaussian
A different problem: Non-Gaussian PDFs

\[ o_t = g(s_t) + \gamma \]
\[ s_t = f(s_{t-1}) + \varepsilon \]

- We have assumed so far that:
  - \( P_0(s) \) is Gaussian or can be approximated as Gaussian
  - \( P(\varepsilon) \) is Gaussian
  - \( P(\gamma) \) is Gaussian

- This has a happy consequence: All distributions remain Gaussian

- But when any of these are not Gaussian, the results are not so happy
A simple case

\[ o_t = B s_t + \gamma \]

\[ P(\gamma) = \sum_{i=0}^{1} w_i \text{Gaussian}(\gamma; \mu_i, \Theta_i) \]

- \( P(\gamma) \) is a mixture of only two Gaussians
- \( o \) is a linear function of \( s \)
  - Non-linear functions would be linearized anyway
- \( P(o \mid s) \) is also a Gaussian mixture!

\[ P(o_t \mid s_t) = P(\gamma = o_t - B s_t) = \sum_{i=0}^{1} w_i \text{Gaussian}(o; \mu_i + B s_t, \Theta_i) \]
When distributions are not Gaussian

\[ P(s) = \]

\[ P(s_t | s_{t-1}) = \]

\[ P(O_t | s_t) = \]

\[ a \text{ priori} \]

Transition prob.

State output prob

\[ P(s_0) = P(s) \]
When distributions are not Gaussian

\[ P(s) = \text{a priori} \]

\[ P(s_t|s_{t-1}) = \text{Transition prob.} \]

\[ P(O_t|s_t) = \text{State output prob} \]

\[ P(s_0) = P(s) \]

\[ P(s_0|O_0) = C \cdot P(s_0) \cdot P(O_0|s_0) \]
When distributions are not Gaussian

\[ P(s) = \begin{cases} \text{a priori} \\
\end{cases} \]

\[ P(s_t | s_{t-1}) = \begin{cases} \text{Transition prob.} \\
\end{cases} \]

\[ P(O_t | s_t) = \begin{cases} \text{State output prob} \\
\end{cases} \]

\[ P(s_0) = P(s) \]

\[ P(s_0 | O_0) = C \int P(s_0 | O_0)P(s_1 | s_0)ds_0 \]
When distributions are not Gaussian

<table>
<thead>
<tr>
<th>Distribution</th>
<th>Description</th>
<th>Formula</th>
</tr>
</thead>
<tbody>
<tr>
<td>$P(s)$</td>
<td><em>a priori</em></td>
<td>$P(s_0) = P(s)$</td>
</tr>
<tr>
<td>$P(s_t</td>
<td>s_{t-1})$</td>
<td>Transition prob.</td>
</tr>
<tr>
<td>$P(O_t</td>
<td>s_t)$</td>
<td>State output prob</td>
</tr>
</tbody>
</table>

$$P(s_0) = P(s)$$

$$P(s_0| O_0) = C \int P(s_0| O_0)P(s_1|s_0)ds_0$$

$$P(s_1| O_0) = \int P(s_0| O_0)P(s_1|s_0)ds_0$$

$$P(s_1| O_{0:1}) = C \int P(s_1| O_0)P(O_1|s_0)ds_0$$
When distributions are not Gaussian

\[
P(s) = \text{a priori} \quad P(s_t | s_{t-1}) = \text{Transition prob.} \quad P(O_t | s_t) = \text{State output prob}
\]

\[
P(s_0) = P(s)
\]

\[
P(s_0 | O_0) = C \cdot P(s_0) \cdot P(O_0 | s_0)
\]

\[
P(s_1 | O_0) = \int_{-\infty}^{\infty} P(s_0 | O_0)P(s_1 | s_0)ds_0
\]

\[
P(s_1 | O_{0:1}) = C \cdot P(s_1 | O_0) \cdot P(O_1 | s_0)
\]

\[
P(s_2 | O_{0:1}) = \int_{-\infty}^{\infty} P(s_1 | O_{0:1})P(s_2 | s_1)ds_1
\]
When distributions are not Gaussian

When \( P(O_t|s_t) \) has more than one Gaussian, after only a few time steps...
When distributions are not Gaussian

$$P(s_t \mid O_{0:t}) =$$

We have too many Gaussians for comfort..
Related Topic: How to sample from a Distribution?

• “Sampling from a Distribution $P(x; \Gamma)$ with parameters $\Gamma$”
• Generate random numbers such that
  – The distribution of a large number of generated numbers is $P(x; \Gamma)$
  – The parameters of the distribution are $\Gamma$

• Many algorithms to generate RVs from a variety of distributions
  – Generation from a uniform distribution is well studied
  – Uniform RVs used to sample from multinomial distributions
  – Other distributions: Most commonly, transform a uniform RV to the desired distribution
Sampling from a multinomial

• Given a multinomial over N symbols, with probability of $i^{\text{th}}$ symbol = $P(i)$

• Randomly generate symbols from this distribution

• Can be done by sampling from a uniform distribution
Sampling a multinomial

- Segment a range (0,1) according to the probabilities $P(i)$
  - The $P(i)$ terms will sum to 1.0

![Diagram of sampling a multinomial](image)
Sampling a multinomial

- Segment a range (0,1) according to the probabilities $P(i)$
  - The $P(i)$ terms will sum to 1.0

- Randomly generate a number from a uniform distribution
  - Matlab: “rand”.
    - Generates a number between 0 and 1 with uniform probability

- If the number falls in the $i^{th}$ segment, select the $i^{th}$ symbol
Related Topic: Sampling from a Gaussian

• Many algorithms
  – Simplest: add many samples from a uniform RV
  – The sum of 12 uniform RVs (uniform in (0,1)) is approximately Gaussian with mean 6 and variance 1
  – For scalar Gaussian, mean $\mu$, std dev $\sigma$:

\[
x = \sum_{i=1}^{12} r_i - 6
\]

• Matlab: $x = \mu + \text{randn} \ast \sigma$
  – “randn” draws from a Gaussian of mean=0, variance=1
Related Topic: Sampling from a Gaussian

- Multivariate (d-dimensional) Gaussian with mean $\mu$ and covariance $\Theta$
  - Compute eigenvalue matrix $\Lambda$ and eigenvector matrix $E$ for $\Theta$
  - $\Theta = E \Lambda E^T$
  - Generate $d$ 0-mean unit-variance numbers $x_1..x_d$
  - Arrange them in a vector:
    $$X = [x_1 .. x_d]^T$$
  - Multiply $X$ by the square root of $\Lambda$ and $E$, add $\mu$
    $$Y = \mu + E \sqrt{\Lambda} X$$
Sampling from a Gaussian Mixture

\[ \sum_{i} w_i \text{Gaussian}(X; \mu_i, \Theta_i) \]

- Select a Gaussian by sampling the multinomial distribution of weights:
  \[ j \sim \text{multinomial}(w_1, w_2, \ldots) \]

- Sample from the selected Gaussian
  \[ \text{Gaussian}(X; \mu_j, \Theta_j) \]
When distributions are not Gaussian

\[ P(s) = \begin{cases} \text{a priori} \end{cases} \]

\[ P(s_t | s_{t-1}) = \text{ Transition prob.} \]

\[ P(O_t | s_t) = \text{ State output prob} \]

When \( P(O_t | s_t) \) has more than one Gaussian, after only a few time steps...
The problem of the exploding distribution

- The complexity of the distribution increases exponentially with time
- This is a consequence of having a continuous state space
  - Only Gaussian PDFs propagate without increase of complexity

- Discrete-state systems do not have this problem
  - The number of states in an HMM stays fixed
  - However, discrete state spaces are too coarse

- Solution: Combine the two concepts
  - Discretize the state space dynamically
Discrete approximation to a distribution

- A large-enough collection of randomly-drawn samples from a distribution will approximately quantize the space of the random variable into equi-probable regions
  - We have more random samples from high-probability regions and fewer samples from low-probability regions
Discrete approximation: Random sampling

- A PDF can be approximated as a uniform probability distribution over randomly drawn samples
  - Since each sample represents approximately the same probability mass \((1/M)\) if there are \(M\) samples

\[
P(x) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(x - x_i)
\]
Note: Properties of a discrete distribution

\[ P(x) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(x - x_i) \]

\[ P(x)P(y \mid x) \propto \sum_{i=0}^{M-1} P(y \mid x_i) \delta(x - x_i) \]

- The product of a discrete distribution with another distribution is simply a weighted discrete probability

\[ P(x) \approx \sum_{i=0}^{M-1} w_i \delta(x - x_i) \]

\[ \int_{-\infty}^{\infty} P(x)P(y \mid x)dx = \sum_{i=0}^{M-1} w_i P(y \mid x_i) \]

- The integral of the product is a mixture distribution
Discretizing the state space

- At each time, discretize the predicted state space

\[ P(s_t \mid o_{0:t}) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(s_t - s_i) \]

- \( s_i \) are randomly drawn samples from \( P(s_t \mid o_{0:t}) \)
- Propagate the discretized distribution
Particle Filtering

Assuming that we only generate \textbf{FOUR} samples from the predicted distributions

\[ P(s) = \text{a priori} \quad P(s_t|s_{t-1}) = \text{Transition prob.} \quad P(O_t|s_t) = \text{State output prob} \]

predict \hspace{1cm} P(s_0) = P(s)
Assuming that we only generate **FOUR** samples from the predicted distributions
Assuming that we only generate \textbf{FOUR} samples from the predicted distributions
Assuming that we only generate \textbf{FOUR} samples from the predicted distributions
Particle Filtering

Assuming that we only generate **FOUR** samples from the predicted distributions
Assuming that we only generate \textbf{FOUR} samples from the predicted distributions
Particle Filtering

- $P(s) = \text{a priori}$
- $P(s_t|s_{t-1}) = \text{Transition prob.}$
- $P(O_t|s_t) = \text{State output prob prob}$

$P(s_0) = P(s)$

$P(s_0|O_0) = C \cdot P(s_0) \cdot P(O_0|s_0)$

$P(s_1|O_0) = \int_{-\infty}^{\infty} P(s_0|O_0)P(s_1|s_0)ds_0$

$P(s_1|O_{0:1}) = C \cdot P(s_1|O_0) \cdot P(O_1|s_0)$

$P(s_2|O_{0:1}) = \int_{-\infty}^{\infty} P(s_1|O_{0:1})P(s_2|s_1)ds_1$

Assuming that we only generate **FOUR** samples from the predicted distributions
Assuming that we only generate **FOUR** samples from the predicted distributions
Particle Filtering

Assuming that we only generate **FOUR** samples from the predicted distributions
Particle Filtering

- Discretize state space at the prediction step
  - By sampling the continuous predicted distribution
    - If appropriately sampled, all generated samples may be considered to be equally probable
    - Sampling results in a discrete uniform distribution
  - Update step updates the distribution of the quantized state space
    - Results in a discrete non-uniform distribution

- Predicted state distribution for the next time instant will again be continuous
  - Must be discretized again by sampling

- At any step, the current state distribution will not have more components than the number of samples generated at the previous sampling step
  - The complexity of distributions remains constant
Particle Filtering

Number of mixture components in predicted distribution governed by number of samples in discrete distribution

By deriving a small (100-1000) number of samples at each time instant, all distributions are kept manageable.
Particle Filtering

\[ o_t = g(s_t) + \gamma \]

\[ s_t = f(s_{t-1}) + \varepsilon \]

\[ P_\gamma(\gamma) \]

\[ P_\varepsilon(\varepsilon) \]

• At \( t = 0 \), sample the initial state distribution

\[ P(s_0 \mid o_{-1}) = P(s_0) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(s_0 - \bar{s}_i^0) \text{ where } \bar{s}_i^0 \leftarrow P_0(s) \]

• Update the state distribution with the observation

\[ P(s_t \mid o_{0:t}) = C \sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t))\delta(s_t - \bar{s}_i^t) \]

\[ C = \frac{1}{\sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t))} \]
Particle Filtering

\[ o_t = g(s_t) + \gamma \]

\[ s_t = f(s_{t-1}) + \epsilon \]

\[ P_\gamma(\gamma) \]

\[ P_\epsilon(\epsilon) \]

- Predict the state distribution at the next time

\[
P(s_t \mid o_{0:t-1}) = C \sum_{i=0}^{M-1} P_\gamma(o_{t-1} - g(\bar{s}_i^{t-1})) P_\epsilon(s_t - f(\bar{s}_i^{t-1}))
\]

- Sample the predicted state distribution

\[
P(s_t \mid o_{0:t-1}) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(s_t - \bar{s}_i^t) \text{ where } \bar{s}_i^t \leftarrow P(s_t \mid o_{0:t-1})
\]
Particle Filtering

\[ o_t = g(s_t) + \gamma \quad P_\gamma(\gamma) \quad s_t = f(s_{t-1}) + \varepsilon \quad P_\varepsilon(\varepsilon) \]

• Predict the state distribution at t

\[ P(s_t \mid o_{0:t-1}) = C \sum_{i=0}^{M-1} P_\gamma(o_{t-1} - g(\bar{s}_{i}^{t-1}))P_\varepsilon(s_t - f(\bar{s}_{i}^{t-1})) \]

• Sample the predicted state distribution at t

\[ P(s_t \mid o_{0:t-1}) \approx \frac{1}{M} \sum_{i=0}^{M-1} \delta(s_t - \bar{s}_i^t) \text{ where } \bar{s}_i^t \leftarrow P(s_t \mid o_{0:t-1}) \]

• Update the state distribution at t

\[ P(s_t \mid o_{0:t}) = C \sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t))\delta(s_t - \bar{s}_i^t) \]

\[ C = \frac{1}{\sum_{i=0}^{M-1} P_\gamma(o_t - g(\bar{s}_i^t))} \]
Estimating a state

• The algorithm gives us a discrete updated distribution over states:

\[ P(s_t \mid o_{0:t}) = C \sum_{i=0}^{M-1} P_\gamma (o_t - g(s_i^t)) \delta(s_t - s_i^t) \]

• The actual state can be estimated as the mean of this distribution

\[ \hat{s}_t = C \sum_{i=0}^{M-1} s_i^t P_\gamma (o_t - g(s_i^t)) \]

• Alternately, it can be the most likely sample

\[ \hat{s}_t = s_j^t : j = \arg \max_i P_\gamma (o_t - g(s_i^t)) \]
Simulations with a Linear Model

\[ S_t = S_{t-1} + \mathcal{E}_t \]
\[ O_t = S_t + x_t \]

- \( \mathcal{E}_t \) has a Gaussian distribution with 0 mean, known variance
- \( x_t \) has a mixture Gaussian distribution with known parameters
- Simulation:
  - Generate state sequence \( S_t \) from model
  - Generate sequence of \( X_t \) from model with one \( X_t \) term for every \( S_t \) term
  - Generate observation sequence \( O_t \) from \( S_t \) and \( X_t \)
  - Attempt to estimate \( S_t \) from \( O_t \)
Generate state sequence according to:

\[ \varepsilon_t \] is Gaussian with mean 0 and variance 10

\[ s_t = s_{t-1} + \varepsilon_t \]
Simulation: Synthesizing data

Generate state sequence according to:
\[ \varepsilon_t = \text{Gaussian with mean 0 and variance 10} \]

\[ s_t = s_{t-1} + \varepsilon_t \]

Generate observation sequence from state sequence according to:
\[ x_t = \text{mixture Gaussian with parameters:} \]
- Means = \([-4, 0, 4, 8, 12, 16, 18, 20]\]
- Variances = \([10, 10, 10, 10, 10, 10, 10, 10]\]
- Mixture weights = \([0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125, 0.125]\)

\[ o_t = s_t + x_t \]
Simulation: Synthesizing data

Combined figure for more compact representation
SIMULATION: TIME = 1

PREDICTED STATE DISTRIBUTION AT TIME = 1
SIMULATION: TIME = 1

SAMPLED VERSION OF PREDICTED STATE DISTRIBUTION AT TIME = 1
SIMULATION: TIME = 1

SAMPLED VERSION OF PREDICTED STATE DISTRIBUTION AT TIME = 1
SIMULATION: TIME = 1

UPDATED VERSION OF SAMPLED VERSION OF PREDICTED STATE DISTRIBUTION AT TIME = 1 AFTER SEEING FIRST OBSERVATION
SIMULATION: TIME = 1

update

update, t <= 1
SIMULATION: TIME = 2

predict

update, t <= 1
SIMULATION: TIME = 2

predict

update, t <= 1
SIMULATION: TIME = 2

update, t <= 1
SIMULATION: TIME = 2

update

update, t <= 1
SIMULATION: TIME = 2

update

update, t <= 2
SIMULATION: TIME = 3

update

predict

update, t <= 2
SIMULATION: TIME = 3

predict

update, t <= 2
SIMULATION: TIME = 3

predict

update, t <= 2
SIMULATION: TIME = 3

update, t <= 2
SIMULATION: TIME = 3

update, t <= 2

update
The figure below shows the contour of the updated state probabilities for all time instants until the current instant.
Simulation: Updated Probs Until $T=3$

update, $t \leq 3$
Simulation: Updated Probs Until $t \leq 100$
Simulation: Updated Probs Until $t \leq 200$
Simulation: Updated Probs Until T=300

update, t <= 300
Simulation: Updated Probs Until T=500

update, t <= 500
Simulation: Updated Probs Until T=1000

update, t <= 1000
Updated Probs Until $T = 1000$

update, $t \leq 1000$
Updated Probs Until T = 1000
Updated Probs: Top View

update, $t \leq 1000$
ESTIMATED STATE
Observation, True States, Estimate

![Graph showing observation, true states, and estimate over sample index.](image-url)
Particle Filtering

• Generally quite effective in scenarios where EKF/UKF may not be applicable
  – Potential applications include tracking and edge detection in images!
  – Not very commonly used however

• Highly dependent on sampling
  – A large number of samples required for accurate representation
  – Samples may not represent mode of distribution
  – Some distributions are not amenable to sampling
    • Use importance sampling instead: Sample a Gaussian and assign non-uniform weights to samples
Prediction filters

• HMMs

• Continuous state systems
  – Linear Gaussian: Kalman
  – Nonlinear Gaussian: Extended Kalman
  – Non-Gaussian: Particle filtering

• EKFs are the most commonly used kalman filters..