The Kalman Filter and the Extended Kalman Filter

D.A. Forsyth

Tracking - more formal view

- Very general model:
  - We assume there are moving objects, which have an underlying state X
  - There are observations Y, some of which are functions of this state
  - There is a clock
    - at each tick, the state changes
    - at each tick, we get a new observation
- Examples
  - object is ball, state is 3D position+velocity, observations are stereo pairs
  - object is person, state is body configuration, observations are frames, clock is in camera (30 fps)

Tracking - Probabilistic formulation

- Given
  - P(X_i-1|Y_0, ..., Y_i-1)
    - "Prior"
- We should like to know
  - P(X_i|Y_0, ..., Y_i-1)
    - "Predictive distribution"
  - P(X_i|Y_0, ..., Y_i)
    - "Posterior"

Key assumptions:

- **Only the immediate past matters:** formally, we require

$$P(\boldsymbol{X}_i|\boldsymbol{X}_1, \ldots, \boldsymbol{X}_{i-1}) = P(\boldsymbol{X}_i|\boldsymbol{X}_{i-1})$$

  This assumption hugely simplifies the design of algorithms, as we shall see; furthermore, it isn't terribly restrictive if we're clever about interpreting $\boldsymbol{X}_i$ as we shall show in the next section.

- **Measurements depend only on the current state:** we assume that $\boldsymbol{Y}_i$ is conditionally independent of all other measurements given $\boldsymbol{X}_i$. This means that

$$P(\boldsymbol{Y}_i, \boldsymbol{Y}_j, \ldots \boldsymbol{Y}_k|\boldsymbol{X}_i) = P(\boldsymbol{Y}_i|\boldsymbol{X}_i)P(\boldsymbol{Y}_j, \ldots, \boldsymbol{Y}_k|\boldsymbol{X}_i)$$

  Again, this isn't a particularly restrictive or controversial assumption, but it yields important simplifications.

Tracking as Induction - base case

Firstly, we assume that we have $P(\boldsymbol{X}_0)$

Then we have

$$\begin{aligned} P(\boldsymbol{X}_0|\boldsymbol{Y}_0 = \boldsymbol{y}_0) &= \frac{P(\boldsymbol{y}_0|\boldsymbol{X}_0)P(\boldsymbol{X}_0)}{P(\boldsymbol{y}_0)} \\ &= \frac{P(\boldsymbol{y}_0|\boldsymbol{X}_0)P(\boldsymbol{X}_0)}{\int P(\boldsymbol{y}_0|\boldsymbol{X}_0)P(\boldsymbol{X}_0)d\boldsymbol{X}_0} \\ &\propto P(\boldsymbol{y}_0|\boldsymbol{X}_0)P(\boldsymbol{X}_0) \end{aligned}$$

Tracking as induction - induction step

Given $P(\boldsymbol{X}_{i-1}|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})$.

Prediction

Prediction involves representing

$$P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})$$

Notice this is i-1
current state based
on previous
measurements

Our independence assumptions make it possible to write

$$\begin{aligned}
P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) &= \int P(\boldsymbol{X}_i, \boldsymbol{X}_{i-1}|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) d\boldsymbol{X}_{i-1} \\
&= \int P(\boldsymbol{X}_i|\boldsymbol{X}_{i-1}, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) P(\boldsymbol{X}_{i-1}|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) d\boldsymbol{X}_{i-1} \\
&= \int P(\boldsymbol{X}_i|\boldsymbol{X}_{i-1}) P(\boldsymbol{X}_{i-1}|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}) d\boldsymbol{X}_{i-1}
\end{aligned}$$

Tracking as induction - induction step

Correction

Correction involves obtaining a representation of

$$P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)$$

Notice this is i
Prediction based on
current measurement
as well.

Our independence assumptions make it possible to write

$$\begin{aligned} P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i) &= \frac{P(\boldsymbol{X}_i, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)}{P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)} \\ &= \frac{P(\boldsymbol{y}_i|\boldsymbol{X}_i, \boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})}{P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)} \\ &= P(\boldsymbol{y}_i|\boldsymbol{X}_i)P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})\frac{P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})}{P(\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)} \\ &= \frac{P(\boldsymbol{y}_i|\boldsymbol{X}_i)P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})}{\int P(\boldsymbol{y}_i|\boldsymbol{X}_i)P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1})d\boldsymbol{X}_i} \end{aligned}$$

The Kalman Filter

- Assume that:
  - All state follows a linear dynamical model
  - Measurements are a linear function of state, plus noise
- Then (if first prior is Gaussian)
  - All PDF's are Gaussian
    - and so easy to represent
      - just need to keep track of mean and covariance
- The Kalman Filter correctly updates mean and covariance

In 1 D

- We have

When both $P(\mathcal{D}|\theta)$ and $P(\theta)$ are normal with known standard deviation, the posterior is normal, too.

$$-\log\left(P(x|\theta)\right) = \frac{1}{2\sigma^2}\left(x-\theta\right)^2 + K(\sigma)$$

$$-\log\left(P(\theta)\right) = \frac{1}{2s^2}\left(\theta-\mu\right)^2 + K(s)$$

$$-\log\left(P(\theta|x)\right) = \frac{1}{(\text{something})}\left(\theta^2 - 2(\text{something else})\theta\right) + K'$$

Problem

- x - measurement
- theta - length of cable
- p(theta) normal
- p(x|theta) normal; mean will theta*const standard dev known

- p(theta|x)

- log p(theta|x)=-1/(2sigma^2) theta^2 + mu/sigma^2 theta +other

In 1 D

$$\log P(\theta) = -\frac{(\theta - \mu_\pi)^2}{2\sigma_\pi^2} + \text{constant not dependent on } \theta.$$

Start by assuming that $\mathcal{D}$ is a single measurement x_1. The measurement x_1 could be in different units from θ, and we will assume that the relevant scaling constant c_1 is known. We assume that $P(x_1|\theta)$ is normal with known standard deviation $\sigma_{m,1}$, and with mean $c_1\theta$. Equivalently, x_1 is obtained by adding noise to $c_1\theta$. The noise will have zero mean and standard deviation $\sigma_{m,1}$. This means that

$$\log P(\mathcal{D}|\theta) = \log P(x_1|\theta) = -\frac{(x_1 - c_1\theta)^2}{2\sigma_{m,1}^2} + \text{constant not dependent on } x_1 \text{ or } \theta.$$

We would like to know $P(\theta|x)$. We have that

$$\begin{aligned}
\log P(\theta|x_1) &= \log p(x_1|\theta) + \log p(\theta) + \text{terms not depending on } \theta \\
&= -\frac{(x_1 - c_1\theta)^2}{2\sigma_{m,1}^2} - \frac{(\theta - \mu_\pi)^2}{2\sigma_\pi^2} \\
&+ \text{terms not depending on } \theta. \\
&= -\left[\theta^2\left(\frac{c_1^2}{2\sigma_{m,1}^2} + \frac{1}{2\sigma_\pi^2}\right) - \theta\left(\frac{c_1 x_1}{2\sigma_{m,1}^2} + \frac{\mu_\pi}{2\sigma_\pi^2}\right)\right] \\
&+ \text{terms not depending on } \theta.
\end{aligned}$$

In 1D

Now some trickery will get us an expression for $P(\theta|x_1)$. Notice first that $\log P(\theta|x_1)$ is of degree 2 in θ (i.e. it has terms θ^2, θ and things that don't depend on θ). This means that $P(\theta|x_1)$ must be a normal distribution, because we can rearrange its log into the form of the log of a normal distribution.

Now we can show that $P(\theta|\mathcal{D})$ is normal when there are more measurements. Assume we have N measurements, $x_1, \ldots, x_N$. The measurements are IID samples from a normal distribution conditioned on θ. We will assume that each measurement is in its own set of units (captured by a constant c_i), and each measurement incorporates noise of different standard deviation (with standard deviation $\sigma_{m,i}$). So

$$\log P(x_i|\theta) = -\frac{(x_i - c_i\theta)^2}{2\sigma_{m,i}^2} + \text{constant not dependent on } x_1 \text{ or } \theta.$$

Now

$$\log P(\mathcal{D}|\theta) = \sum_i \log P(x_i|\theta)$$

in 1D

so we can write

$$\begin{aligned}\log P(\theta|\mathcal{D}) &= \log p(x_N|\theta) + \ldots + \log p(x_2|\theta) + \log p(x_1|\theta) + \log p(\theta) + \text{terms not depending on } \theta \\ &= \log p(x_N|\theta) + \ldots + \log p(x_2|\theta) + \log p(\theta|x_1) + \text{terms not depending on } \theta \\ &= \log p(x_N|\theta) + \ldots + \log p(\theta|x_1, x_2) + \text{terms not depending on } \theta.\end{aligned}$$

This lays out the induction. We have that $P(\theta|x_1)$ is normal, with known standard deviation. Now regard this as the prior, and $P(x_2|\theta)$ as the likelihood; we have that $P(\theta|x_1, x_2)$ is normal, and so on. So under our assumptions, $P(\theta|\mathcal{D})$ is normal. We now have a really useful fact.

Remember this: *A normal prior and a normal likelihood yield a normal posterior when both standard deviations are known*

In 1D

Useful Fact: 9.2 *The parameters of a normal posterior with a single measurement*

Assume we wish to estimate a parameter θ. The prior distribution for θ is normal, with known mean μ_π and known standard deviation σ_π. We receive a single data item x_1 and a scale c_1. The likelihood of x_1 is normal with mean $c_1\theta$ and standard deviation $\sigma_{m,1}$, where $\sigma_{m,1}$ is known. Then the posterior, $p(\theta|x_1, c_1, \sigma_{m,1}, \mu_\pi, \sigma_\pi)$, is normal, with mean

$$\mu_1 = \frac{c_1 x_1 \sigma_\pi^2 + \mu_\pi \sigma_{m,1}^2}{\sigma_{m,1}^2 + c_1^2 \sigma_\pi^2}$$

and standard deviation

$$\sigma_1 = \sqrt{\frac{\sigma_{m,1}^2 \sigma_\pi^2}{\sigma_{m,1}^2 + c^2 \sigma_\pi^2}}.$$

Recursion

Useful Fact: 9.3 *Normal posteriors can be updated online*

Assume we wish to estimate a parameter θ. The prior distribution for θ is normal, with known mean μ_π and known standard deviation σ_π. We write x_i for the i'th data item. The likelihood for each separate data item is normal, with mean $c_i\theta$ and standard deviation $\sigma_{m,i}$. We have already received k data items. The posterior $p(\theta|x_1, \ldots, x_k, c_1, \ldots, c_k, \sigma_{m,1}, \ldots, \sigma_{m,k}, \mu_\pi, \sigma_\pi)$ is normal, with mean μ_k and standard deviation σ_k. We receive a new data item x_{k+1}. The likelihood of this data item is normal with mean $c_{k+1}\theta$ and standard deviation $\sigma_{m,(k+1)}$, where c_{k+1} and $\sigma_{m,(k+1)}$ are known. Then the posterior, $p(\theta|x_1, \ldots, x_{k+1}, c_1, \ldots, c_k, c_{k+1}, \sigma_{m,1}, \ldots, \sigma_{m,(k+1)}, \mu_\pi, \sigma_\pi)$, is normal, with mean

$$\mu_{k+1} = \frac{c_{k+1} x_{k+1} \sigma_k^2 + \mu_k \sigma_{m,(k+1)}^2}{\sigma_{m,(k+1)}^2 + c_{k+1}^2 \sigma_k^2}$$

and

$$\sigma_{k+1}^2 = \frac{\sigma_{m,(k+1)}^2 \sigma_k^2}{\sigma_{m,(k+1)}^2 + c_{k+1}^2 \sigma_k^2}.$$

Linear models

Read this as: x_i is normally distributed. The mean is a linear function of x_i-1 and whose variance is known (and can depend on i).

$$x_i \sim N(\mathcal{D}_i \boldsymbol{x}_{i-1}; \Sigma_{d_i})$$

$$y_i \sim N(\mathcal{M}_i \boldsymbol{x}_i; \Sigma_{m_i})$$

Read this as: y_i is normally distributed. The mean is a linear function of x_i and whose variance is known (and can depend on i)

Examples

- Dynamical models
  - Drifting points
    - new state = old state + gaussian noise
  - Points moving with constant velocity
    - new position=old position + (dt) old velocity + gaussian noise
    - new velocity= old velocity+gaussian noise
  - Points moving with constant acceleration
    - etc
- Measurement models
  - state=position; measurement=position+gaussian noise
  - state=position and velocity; measurement=position+gaussian noise
    - but we could infer velocity
  - state=position and velocity and acceleration; measurement=position+gaussian noise

The Kalman Filter

- Dynamic Model

$$\mathbf{x}_i \sim N(\mathcal{D}_i \mathbf{x}_{i-1}, \Sigma_{d_i})$$

$$\mathbf{y}_i \sim N(\mathcal{M}_i \mathbf{x}_i, \Sigma_{m_i})$$

- Notation

$$\text{mean of } P(X_i | y_0, \ldots, y_{i-1}) \text{ as } \bar{X}_i^-$$

$$\text{mean of } P(X_i | y_0, \ldots, y_i) \text{ as } \bar{X}_i^+$$

$$\text{covar of } P(X_i | y_0, \ldots, y_{i-1}) \text{ as } \Sigma_i^-$$

$$\text{covar of } P(X_i | y_0, \ldots, y_i) \text{ as } \Sigma_i^+$$

Dynamic Model:

$$\begin{aligned} \boldsymbol{x}_i &\sim N(\mathcal{D}_i \boldsymbol{x}_{i-1}, \Sigma_{d_i}) \\ \boldsymbol{y}_i &\sim N(\mathcal{M}_i \boldsymbol{x}_i, \Sigma_{m_i}) \end{aligned}$$

Start Assumptions: $\overline{\boldsymbol{x}}_0^-$ and Σ_0^- are known

Update Equations: Prediction

$$\overline{\boldsymbol{x}}_i^- = \mathcal{D}_i \overline{\boldsymbol{x}}_{i-1}^+$$

$$\Sigma_i^- = \Sigma_{d_i} + \mathcal{D}_i \Sigma_{i-1}^+ \mathcal{D}_i^T$$

Update Equations: Correction

$$\begin{aligned} \mathcal{K}_i &= \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i} \right]^{-1} \\ \overline{\boldsymbol{x}}_i^+ &= \overline{\boldsymbol{x}}_i^- + \mathcal{K}_i \left[\boldsymbol{y}_i - \mathcal{M}_i \overline{\boldsymbol{x}}_i^- \right] \\ \Sigma_i^+ &= \left[Id - \mathcal{K}_i \mathcal{M}_i \right] \Sigma_i^- \end{aligned}$$

Algorithm 11.3: The Kalman Filter.

What's going on here?

- Dynamics (not looking at i'th measurement yet):

$$\mathbf{x}_{i-1} \sim N(\bar{X}^{+}_{i-1}, \Sigma^{+}_{i-1})$$

$$\mathbf{x}_i \sim N(\mathcal{D}_i \mathbf{x}_{i-1}, \Sigma_{d_i})$$

means that $\Sigma^{-}_{i} = \Sigma_{d_i} + \mathcal{D}_i \Sigma^{+}_{i-1} \mathcal{D}_i^T$ where $\zeta \sim N(0, \Sigma^{-}_{i})$

So

$$\text{mean}(\mathbf{x}_i) = \mathcal{D}_i \text{mean}(\mathbf{x}_{i-1})$$

$$\bar{X}^{-}_{i} = \mathcal{D}_i \bar{X}^{+}_{i-1}$$

$$\mathbf{x}_i = \mathcal{D}_i \mathbf{x}_{i-1} + \zeta$$

$$\text{cov}(\mathbf{x}_i) = \mathcal{D}_i \text{cov}(\mathbf{x}_{i-1}) \mathcal{D}_i^T + \text{cov}(\zeta)$$

Useful Fact: 9.2 *The parameters of a normal posterior with a single measurement*

Assume we wish to estimate a parameter θ. The prior distribution for θ is normal, with known mean μ_π and known standard deviation σ_π. We receive a single data item x_1 and a scale c_1. The likelihood of x_1 is normal with mean $c_1\theta$ and standard deviation $\sigma_{m,1}$, where $\sigma_{m,1}$ is known. Then the posterior, $p(\theta|x_1, c_1, \sigma_{m,1}, \mu_\pi, \sigma_\pi)$, is normal, with mean

$$\mu_1 = \frac{c_1 x_1 \sigma_\pi^2 + \mu_\pi \sigma_{m,1}^2}{\sigma_{m,1}^2 + c_1^2 \sigma_\pi^2}$$

and standard deviation

$$\sigma_1 = \sqrt{\frac{\sigma_{m,1}^2 \sigma_\pi^2}{\sigma_{m,1}^2 + c^2 \sigma_\pi^2}}.$$

posterior mean is weighted combo of prior mean and measurement

posterior covar is weighted combo of prior covar, measurement matrix and measurement covar

$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}\right]$$

$$\bar{X}_i^+ = \bar{X}_i^- + \mathcal{K}_i \left[\mathbf{y}_i - \mathcal{M}_i \bar{X}_i^-\right]$$

posterior mean is weighted combo of prior mean and measurement

$$\Sigma_i^+ = \left[\mathcal{I} - \mathcal{K}_i \mathcal{M}_i\right] \Sigma_i^-$$

posterior covar is weighted combo of prior covar, measurement matrix and measurement covar

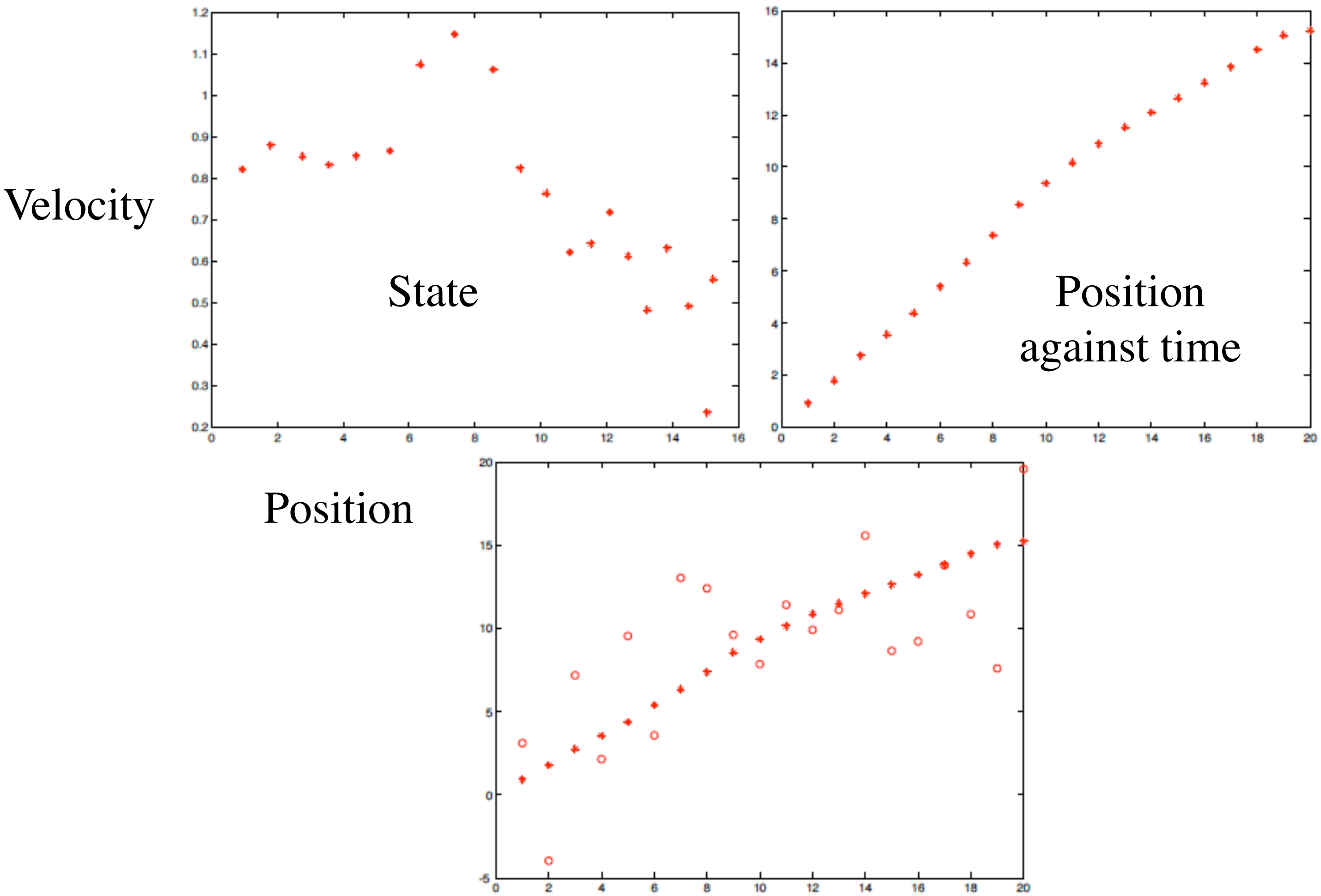
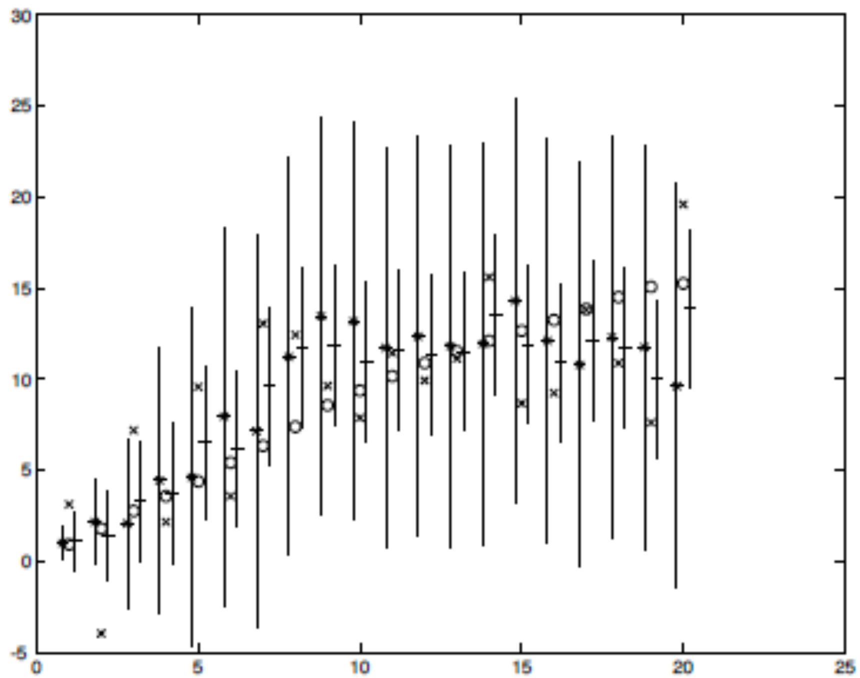


FIGURE 11.7: A constant velocity dynamic model for a point on the line. In this case, the state space is two dimensional, with one coordinate for position, one for velocity. The figure on the **top left** shows a plot of the state; each asterisk is a different state. Notice that the vertical axis (velocity) shows some small change compared with the horizontal axis. This small change is generated only by the random component of the model, so the velocity is constant up to a random change. The figure on the **top right** shows the first component of state (which is position) plotted against the time axis. Notice we have something that is moving with roughly constant velocity. The figure on the **bottom** overlays the measurements (the circles) on this plot. We are assuming that the measurements are of position only, and are quite poor; as we see, this doesn't significantly affect our ability to track.



Notice how uncertainty in state grows with movement and is reduced with measurement.

FIGURE 11.9: The Kalman filter for a point moving on the line under our model of constant velocity (compare with Figure 11.7). The state is plotted with open circles as a function of the step i. The *s give $\overline{x}_i^-$, which is plotted slightly to the left of the state to indicate that the estimate is made before the measurement. The xs give the measurements, and the +s give $\overline{x}_i^+$, which is plotted slightly to the right of the state. The vertical bars around the *s and the +s are three standard deviation bars, using the estimate of variance obtained before and after the measurement, respectively. When the measurement is noisy, the bars don't contract all that much when a measurement is obtained (compare with Figure 11.10).

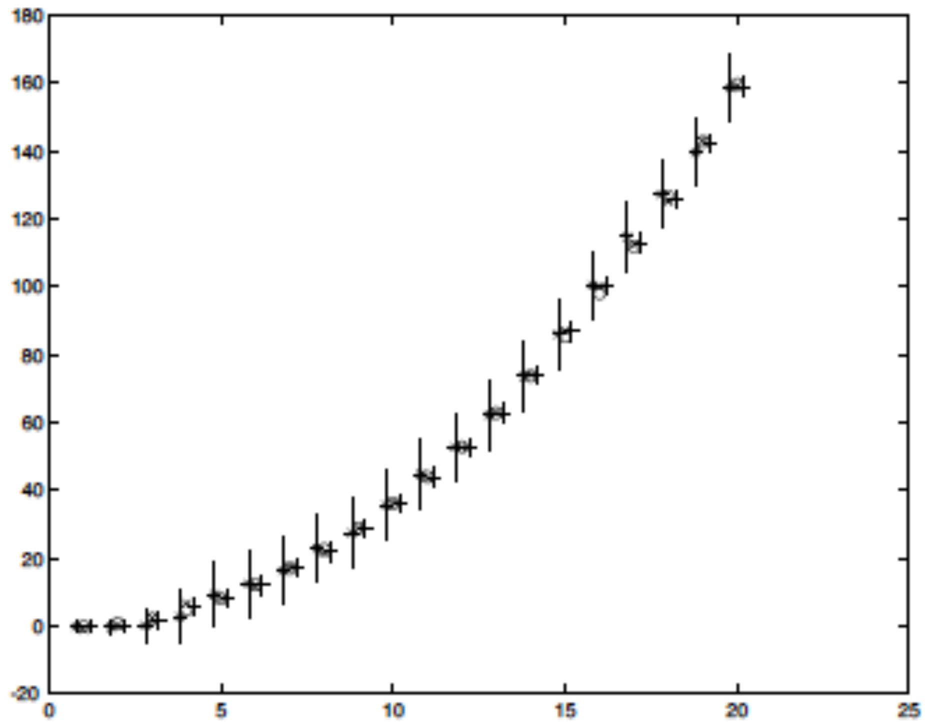


FIGURE 11.10: The Kalman filter for a point moving on the line under our model of constant acceleration (compare with Figure 11.8). The state is plotted with open circles as a function of the step i. The *s give $\overline{x}_i^-$, which is plotted slightly to the left of the state to indicate that the estimate is made before the measurement. The xs give the measurements, and the +s give $\overline{x}_i^+$, which is plotted slightly to the right of the state. The vertical bars around the *s and the +s are three standard deviation bars, using the estimate of variance obtained before and after the measurement, respectively. When the measurement is noisy, the bars don't contract all that much when a measurement is obtained.

Tricks

- Smoothing
  - You can build a representation of $P(X_i|Y_0, \ldots Y_N)$
    - (i.e. incorporating future measurements)
    - run one filter forward, one backward
      - posterior of forward filter is normal
      - predictive for backward is normal
      - etc.
- Polishing
  - This means that, if I can endure latency, I can have two estimates
    - one at the time of the i'th measurement
    - one a few measurements later, that is more accurate

Forward filter: Obtain the mean and variance of $P(\boldsymbol{X}_i|\boldsymbol{y}_0, \ldots, \boldsymbol{y}_i)$ using the Kalman filter. These are $\overline{\boldsymbol{X}}_i^{f,+}$ and $\Sigma_i^{f,+}$.

Backward filter: Obtain the mean and variance of $P(\boldsymbol{X}_i|\boldsymbol{y}_{i+1}, \ldots, \boldsymbol{y}_N)$ using the Kalman filter running backward in time. These are $\overline{\boldsymbol{X}}_i^{b,-}$ and $\Sigma_i^{b,-}$.

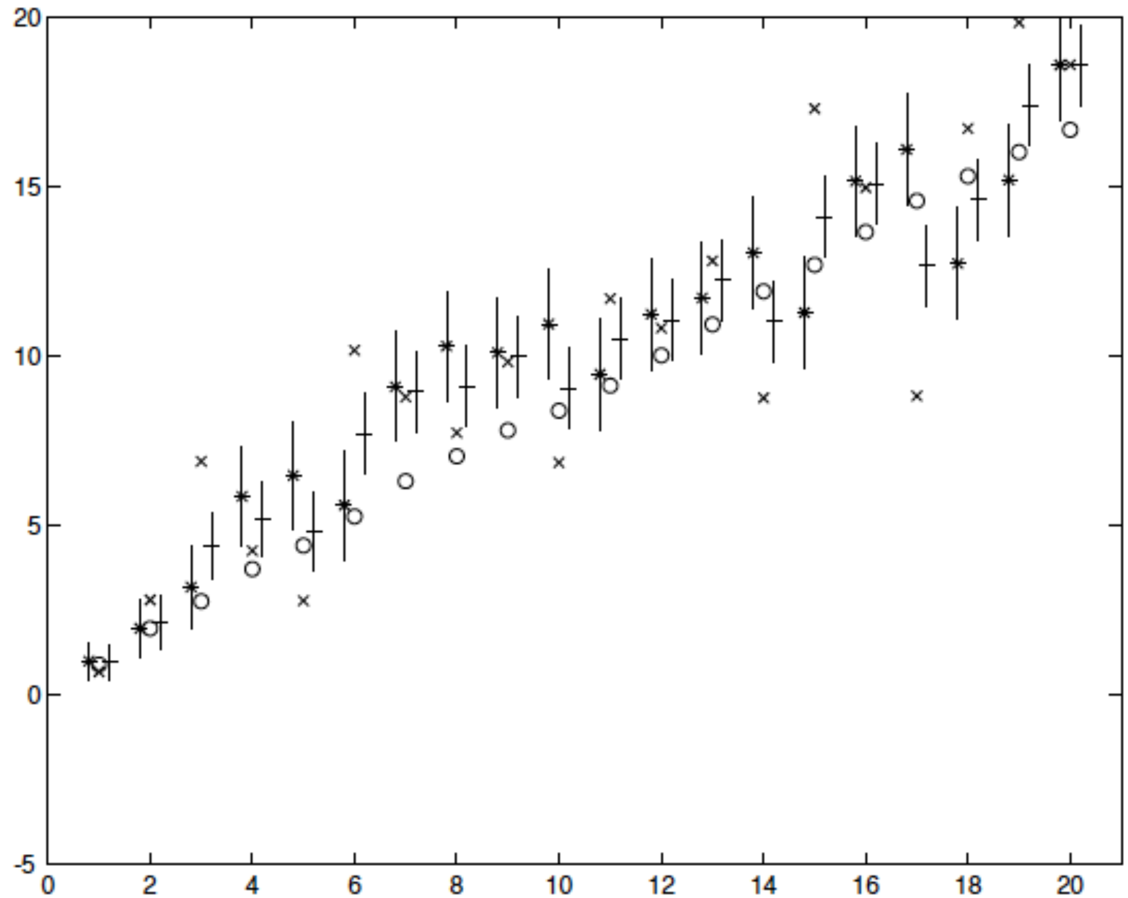
Combining forward and backward estimates: Regard the backward estimate as a new measurement for $\boldsymbol{X}_i$, and insert into the Kalman filter equations to obtain

$$\Sigma_i^* = \left[(\Sigma_i^{f,+})^{-1} + (\Sigma_i^{b,-})^{-1}\right]^{-1};$$

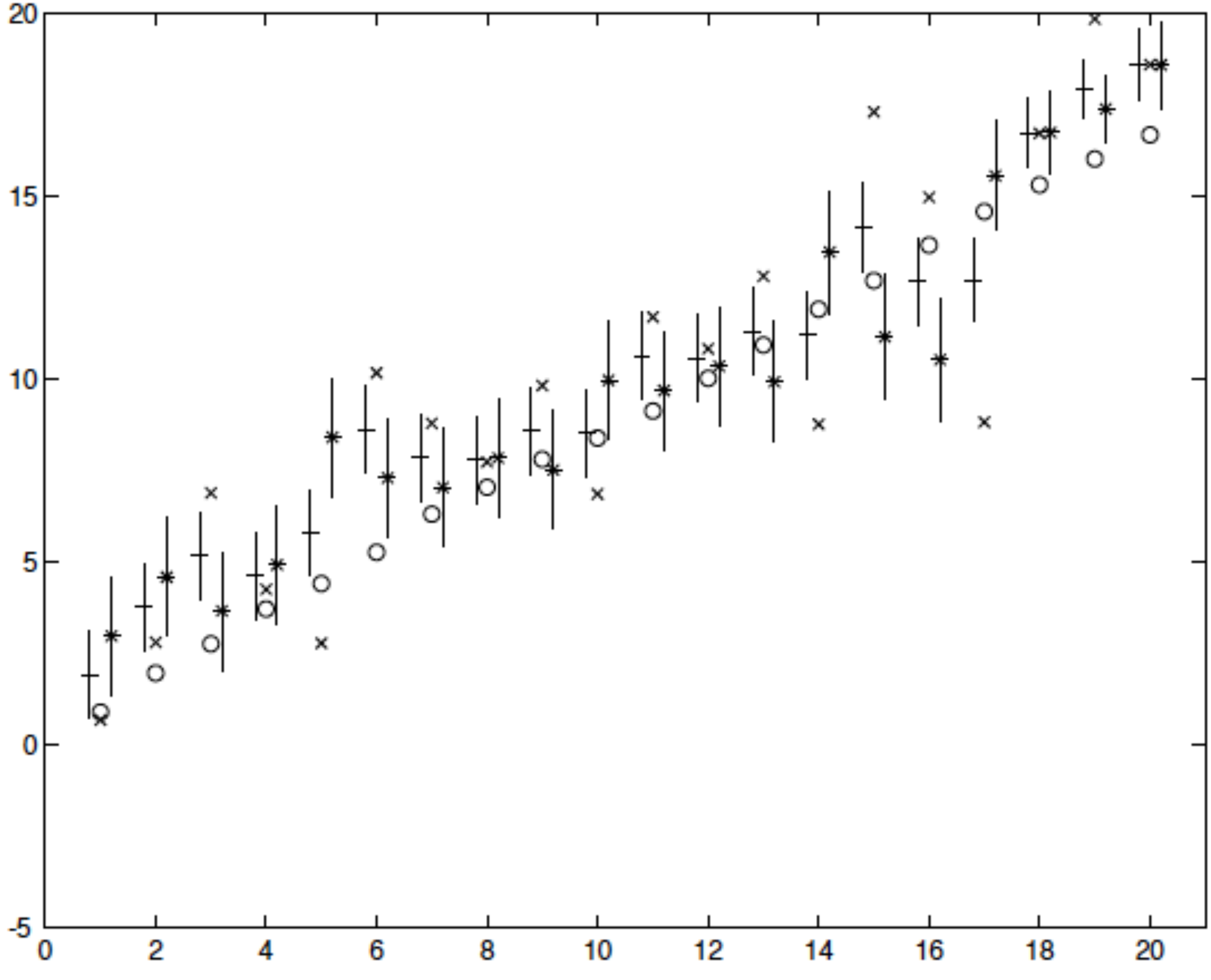
$$\overline{\boldsymbol{X}}_i^* = \Sigma_i^* \left[(\Sigma_i^{f,+})^{-1}\overline{\boldsymbol{X}}_i^{f,+} + (\Sigma_i^{b,-})^{-1}\overline{\boldsymbol{X}}_i^{b,-}\right].$$

Algorithm 11.4: Forward-Backward Smoothing.

Forward



Backward



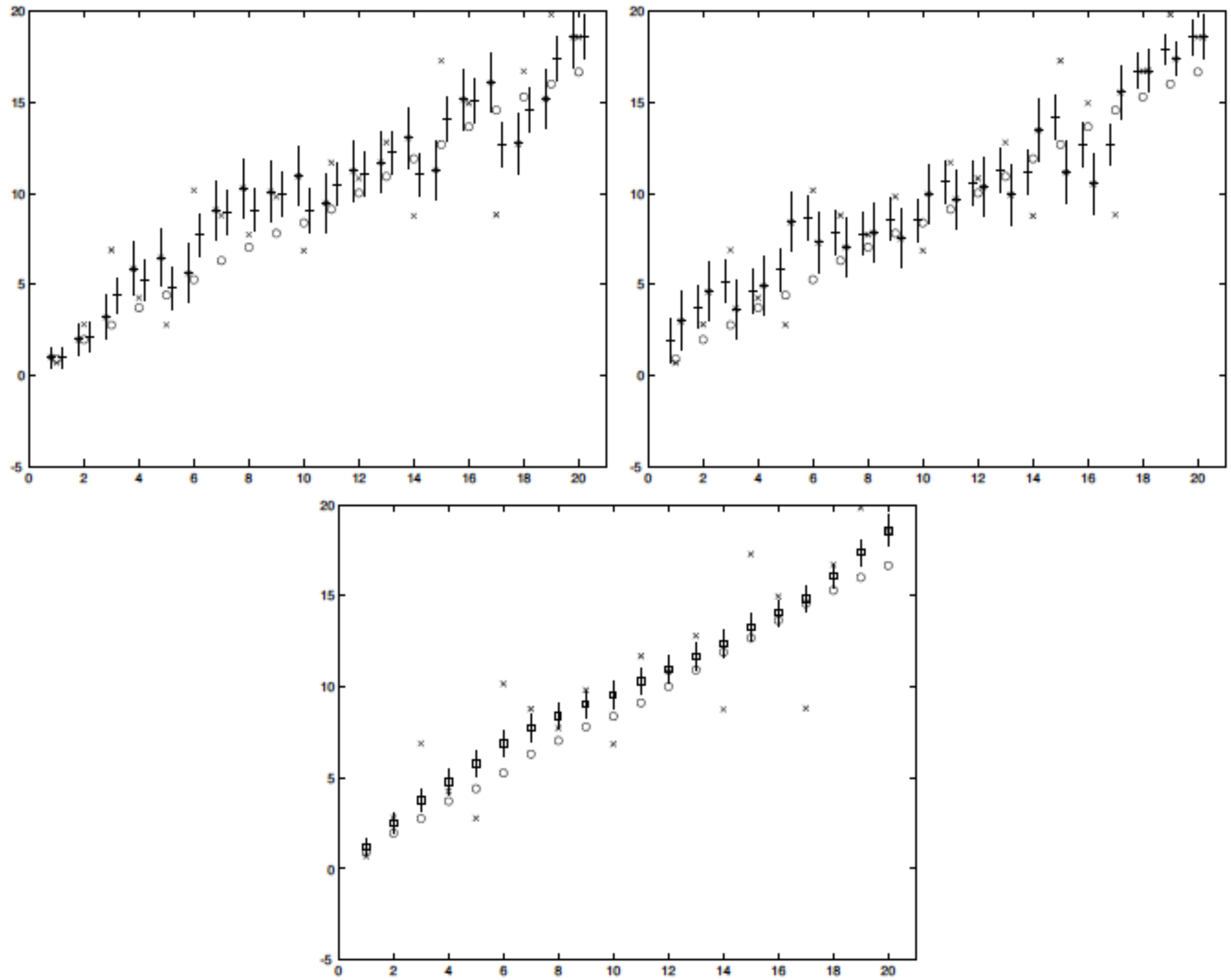


FIGURE 11.11: Forward-backward estimation for a dynamic model of a point moving on the line with constant velocity. We are plotting the position component of state against time. On the **top left**, we show the forward estimates, again using the convention that the state is shown with circles, the data is shown with an x, the prediction is shown with a *, and the corrected estimate is shown with a +; the bars give one standard deviation in the estimate. The predicted estimate is shown slightly behind the state, and the corrected estimate is shown slightly ahead of the state. You should notice that the measurements are noisy. On the **top right** we show the backward estimates. Now time is running backward (although we have plotted both curves on the same axis), so that the prediction is slightly ahead of the measurement and the corrected estimate is slightly behind. We have used the final corrected estimate of the forward filter as a prior. Again, the bars give one standard deviation in each variable. On the **bottom**, we show the combined forward-backward estimate. The squares give the estimates of state. Notice the significant improvement in the estimate.

Data Association

- Nearest neighbours
  - choose the measurement with highest probability given predicted state
  - popular, but can lead to catastrophe
- Probabilistic Data Association
  - combine measurements, weighting by probability given predicted state
  - gate using predicted state

Example: Localization

- Assume
  - car state is (position; velocity; acceleration)
    - it doesn't rotate!
    - this yields D_i, and noise
  - we know M_i and noise
- Then it's all easy (plug in equations and go)
  - but what if we have a LIDAR map and localize with IRLS?

Example: Harder localization

- Write state of vehicle: $$\mathbf{x}_i = \begin{pmatrix} \text{position} \\ \text{velocity} \end{pmatrix}$$
- Can extract position as: $$\mathbf{p}_i = \Pi_p \mathbf{x}_i$$
- State update is: $$\mathbf{x}_i = \mathcal{D}_i \mathbf{x}_{i-1} + \xi$$
- Measurement is:

$$\mathbf{y}_i = \text{argmin}_{\mathbf{u}} C(\mathbf{u}, \mathbf{x}_i)$$

HUH?

Harder localization, II

- Model the cost function as:

$$C(\mathbf{u}, \mathbf{x}_i) \approx c_0 + \mathbf{v}^T(\mathbf{u} - \mathbf{p}_i) + (\mathbf{u} - \mathbf{p}_i)^T \frac{\mathcal{H}}{2}(\mathbf{u} - \mathbf{p}_i)$$

  - at the minimum - so actually, v=0
  - now the cost function might be slightly wrong, which will cause errors in u
  - if we use the model:

$$\mathbf{y}_i = \mathbf{u} = \mathbf{p}_i + \mathcal{H}^{-1/2}\zeta$$

  - then we have:

$$C(\mathbf{y}_i, \mathbf{x}_i) \sim N(c_0, \frac{d\sigma^2}{2})$$

$$\uparrow \quad N(0, \sigma^2 I)$$

And a kind of “evenness” property

Harder localization, III

- This property is reasonable:
    - we can't tell noise directions apart by their effect on the cost function
- Now we're in business:

$$\mathbf{x}_i \sim N(\mathcal{D}_i \mathbf{x}_{i-1}, \Sigma_i)$$

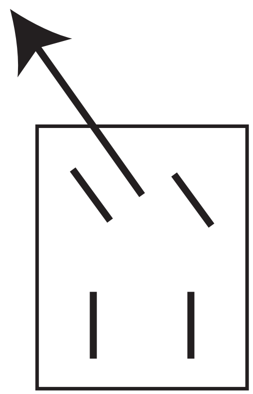
$$\mathbf{y}_i \sim N(\Pi_p \mathbf{x}_i, \sigma^2 \mathcal{H}_i^{-1})$$

But what is this?

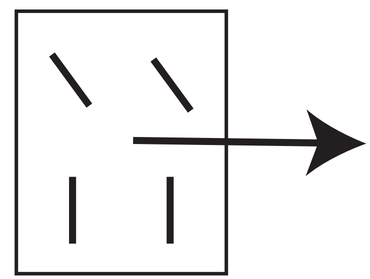
Choose this

Hessian of cost function at best location

Example: Even harder localization!



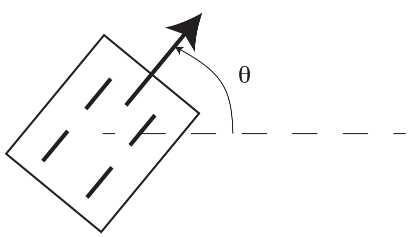
OK



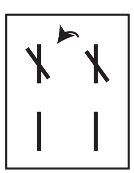
Not OK

Formally: car is non-holonomic

Building a movement model

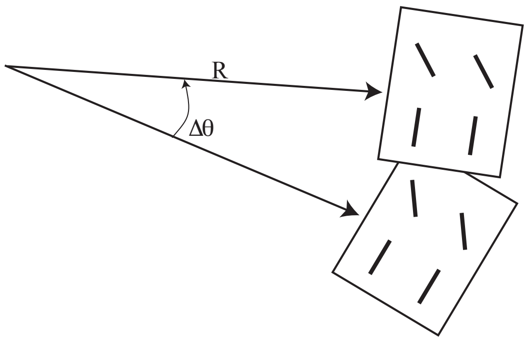


$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \rightarrow \begin{bmatrix} x + v\Delta t cos\theta \\ y + v\Delta t \sin\theta \\ \theta \end{bmatrix}$$



$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \rightarrow \begin{bmatrix} x \\ y \\ \theta + \Delta\theta \end{bmatrix}$$

A general movement model



$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \rightarrow \begin{bmatrix} x + R(\sin(\theta + \Delta\theta) - \sin\theta) \\ y - R(\cos(\theta + \Delta\theta) - \cos\theta) \\ \theta + \Delta\theta \end{bmatrix}$$

THIS ISN'T LINEAR!

For sufficiently small timestep, bounded rate of change in angle, we get

$$\begin{bmatrix} x \\ y \\ \theta \end{bmatrix} \rightarrow \begin{bmatrix} x + v\cos\theta \\ y + v\sin\theta \\ \theta + u \end{bmatrix}$$

v, u parameters of motion

THIS ISN'T LINEAR!

The extended Kalman filter

- What happens if state update, measurement aren't linear?
  - particle filter
  - linearize and approximate (EKF)

$$\mathbf{x}_i = f(\mathbf{x}_{i-1}, \mathbf{n})$$

Noise - normal, mean 0, Cov known

$$\mathbf{y}_i = g(\mathbf{x}_i, \mathbf{n})$$

Recall: The Kalman filter

Dynamic Model:

$$x_i \sim N(\mathcal{D}_i x_{i-1}, \Sigma_{d_i})$$
$$y_i \sim N(\mathcal{M}_i x_i, \Sigma_{m_i})$$

Assumption: state update and measurement are linear with normal noise

Start Assumptions: $\overline{x}_0^-$ and Σ_0^- are known
Update Equations: Prediction

$$\overline{x}_i^- = \mathcal{D}_i \overline{x}_{i-1}^+$$
$$\Sigma_i^- = \Sigma_{d_i} + \mathcal{D}_i \Sigma_{i-1}^+ \mathcal{D}_i^T$$

Update Equations: Correction

$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}\right]^{-1}$$
$$\overline{x}_i^+ = \overline{x}_i^- + \mathcal{K}_i \left[y_i - \mathcal{M}_i \overline{x}_i^-\right]$$
$$\Sigma_i^+ = \left[Id - \mathcal{K}_i \mathcal{M}_i\right] \Sigma_i^-$$

Difference between predicted and observed measurement

Algorithm 11.3: The Kalman Filter.

Linearization and noise

- Two ways in which noise could affect x_i
  - x_{i-1} is noisy
  - AND there is n to account for
- Now consider some nonlinear function with noisy input
  - first case

$$\downarrow$$

$$\mathbf{x}_i = f(\mathbf{x}_{i-1}, \mathbf{n})$$

$$h(\mathbf{x}) \text{ where } \mathbf{x} \sim N(\bar{\mathbf{x}}, \Sigma_x) \quad h(\bar{\mathbf{x}} + \zeta) \text{ where } \zeta \sim N(0, \Sigma_x)$$

Approximate

$$J_{h,x} = \begin{bmatrix} \frac{\partial h_1}{\partial x_1} & \cdots & \cdots \\ \cdots & \frac{\partial h_i}{\partial x_j} & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix}$$

$$h(\bar{\mathbf{x}} + \zeta) \approx h(\bar{\mathbf{x}}) + J_{h,x}\zeta$$

Yields

Jacobian === derivative

$$h(\mathbf{x}) \sim N(h(\bar{\mathbf{x}}), J_{h,x}\Sigma_x J_{h,x}^T)$$

Linearization and noise

- Two ways in which noise could affect x_i
  - x_{i-1} is noisy
  - AND there is n to account for
- Now consider some nonlinear function with fixed input, noise
  - second case

$$\mathbf{x}_i = f(\mathbf{x}_{i-1}, \mathbf{n})$$

$$h(\mathbf{x}, \mathbf{n}) \text{ where } \mathbf{n} \sim N(0, \sigma_n)$$

Approximate

$$h(\mathbf{x}, \mathbf{n}) \approx h(\mathbf{x}, \mathbf{0}) + J_{h,n}\mathbf{n}$$

$$J_{h,n} = \begin{bmatrix} \frac{\partial h_1}{\partial n_1} & \cdots & \cdots \\ \cdots & \frac{\partial h_i}{\partial n_j} & \cdots \\ \cdots & \cdots & \cdots \end{bmatrix}$$

Yields

Jacobian === derivative

$$h(\mathbf{x}, \mathbf{n}) \sim N(h(\mathbf{x}, \mathbf{0}), J_{h,n}\Sigma_n J_{h,n}^T)$$

Recall: The Kalman filter

Dynamic Model:

$$\begin{aligned} \boldsymbol{x}_i &\sim N(\mathcal{D}_i \boldsymbol{x}_{i-1}, \Sigma_{d_i}) \\ \boldsymbol{y}_i &\sim N(\mathcal{M}_i \boldsymbol{x}_i, \Sigma_{m_i}) \end{aligned}$$

Assumption: state update and measurement are linear with normal noise

Start Assumptions: $\overline{\boldsymbol{x}}_0^-$ and Σ_0^- are known

Update Equations: Prediction

$$\overline{\boldsymbol{x}}_i^- = \mathcal{D}_i \overline{\boldsymbol{x}}_{i-1}^+$$

$$\Sigma_i^- = \Sigma_{d_i} + \mathcal{D}_i \Sigma_{i-1}^+ \mathcal{D}_i^T$$

Update Equations: Correction

$$\begin{aligned} \mathcal{K}_i &= \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i} \right]^{-1} \\ \overline{\boldsymbol{x}}_i^+ &= \overline{\boldsymbol{x}}_i^- + \mathcal{K}_i \left[\boldsymbol{y}_i - \mathcal{M}_i \overline{\boldsymbol{x}}_i^- \right] \\ \Sigma_i^+ &= \left[Id - \mathcal{K}_i \mathcal{M}_i \right] \Sigma_i^- \end{aligned}$$

Difference between predicted and observed measurement

Algorithm 11.3: The Kalman Filter.

The extended Kalman filter

- Linearize:

$$\mathbf{x}_i = f(\mathbf{x}_{i-1}, \mathbf{n})$$

$$\mathcal{F}_x = \begin{bmatrix} \frac{\partial f_1}{\partial x_1} & \cdots & \cdots \\ \cdots & \frac{\partial f_i}{\partial x_j} & \cdots \end{bmatrix}$$

$$\mathcal{F}_n = \begin{bmatrix} \frac{\partial f_1}{\partial n_1} & \cdots & \cdots \\ \cdots & \frac{\partial f_i}{\partial n_j} & \cdots \end{bmatrix}$$

Posterior covariance of x_{i-1}

$$\mathbf{x}_i \sim N(f(\overline{\mathbf{x}}_{i-1}^{+}, \mathbf{0}), \mathcal{F}_x \Sigma_{i-1}^{+} \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T)$$

Noise covariance

Dynamic Model:

$$\boldsymbol{y}_i \sim N(\mathcal{M}_i \boldsymbol{x}_i, \Sigma_{m_i})$$

Start Assumptions: $\overline{\boldsymbol{x}}_0^-$ and Σ_0^- are known

Update Equations: Prediction

$$\overline{x}_i^- \qquad \Sigma_i^-$$

$$\mathbf{x}_i \sim \mathcal{N}(f(\overline{\mathbf{x}}_{i-1}, \mathbf{0}), \mathcal{F}_x \Sigma_{i-1}^+ \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T$$

Update Equations: Correction

$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}\right]^{-1}$$

$$\overline{\boldsymbol{x}}_i^+ = \overline{\boldsymbol{x}}_i^- + \mathcal{K}_i \left[\boldsymbol{y}_i - \mathcal{M}_i \overline{\boldsymbol{x}}_i^-\right]$$

$$\Sigma_i^+ = \left[Id - \mathcal{K}_i \mathcal{M}_i\right] \Sigma_i^-$$

Algorithm 11.3: The Kalman Filter.

The extended Kalman filter

- Linearize:

$$\mathbf{y}_i = g(\mathbf{x}_i, \mathbf{n})$$

$$\mathcal{G}_x = \begin{bmatrix} \frac{\partial g}{\partial x_1} & \cdots & \cdots \\ \cdots & \frac{\partial g}{\partial x_1} & \cdots \end{bmatrix}$$

$$\mathcal{G}_n = \begin{bmatrix} \frac{\partial g}{\partial n_1} & \cdots & \cdots \\ \cdots & \frac{\partial g}{\partial n_1} & \cdots \end{bmatrix}$$

$$\mathbf{y}_i \approx \mathcal{N}(g(\mathbf{x}_i, \mathbf{0}), \mathcal{G}_x \Sigma_i^- \mathcal{G}_x^T + \mathcal{G}_n \Sigma_{m,i} \mathcal{G}_n^T)$$

Dynamic Model:

$$\boldsymbol{y}_i \sim N(\mathcal{M}_i \boldsymbol{x}_i, \Sigma_{m_i})$$

Start Assumptions: $\overline{\boldsymbol{x}}_0^-$ and Σ_0^- are known

Update Equations: Prediction

$\overline{x}_i^-$ $\qquad$ Σ_i^-

$$\mathbf{x}_i \sim \mathcal{N}(f(\mathbf{x}_{i-1}, \mathbf{0}), \mathcal{F}_x \Sigma_{i-1}^{+} \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T)$$

Update Equations: Correction

$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}\right]^{-1}$$

$$\overline{\boldsymbol{x}}_i^+ = \overline{\boldsymbol{x}}_i^- + \mathcal{K}_i \left[\boldsymbol{y}_i - \mathcal{M}_i \overline{\boldsymbol{x}}_i^-\right]$$

$$\Sigma_i^+ = \left[Id - \mathcal{K}_i \mathcal{M}_i\right] \Sigma_i^-$$

This is the inverse of the covariance of y_i

Algorithm 11.3: The Kalman Filter.

Dynamic Model:

$$y_i \sim N(\mathcal{M}_i x_i, \Sigma_{m_i})$$

Start Assumptions: $\overline{x}_0^-$ and Σ_0^- are known

Update Equations: Prediction

$\overline{x}_i^-$ $\qquad$ Σ_i^-

$$\mathbf{x}_i \sim \mathcal{N}(f(\mathbf{x}_{i-1}, \mathbf{0}), \mathcal{F}_x \Sigma_{i-1}^+ \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T$$

Update Equations: Correction

$$\begin{aligned}
\mathcal{K}_i &= \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}\right]^{-1} \\
\overline{x}_i^+ &= \overline{x}_i^- + \mathcal{K}_i \left[y_i - \mathcal{M}_i \overline{x}_i^-\right] \\
\Sigma_i^+ &= \left[Id - \mathcal{K}_i \mathcal{M}_i\right] \Sigma_i^-
\end{aligned}$$

Difference between predicted and true measurement

Algorithm 11.3: The Kalman Filter.

Dynamic Model:

$$y_i \sim N(\mathcal{M}_i x_i, \Sigma_{m_i})$$

Start Assumptions: $\overline{x}_0^-$ and Σ_0^- are known

Update Equations: Prediction

$\overline{x}_i^-$ Σ_i^-

$$\mathbf{x}_i \sim \mathcal{N}(f(\mathbf{x}_{i-1}, \mathbf{0}), \mathcal{F}_x \Sigma_{i-1}^+ \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T$$

Update Equations: Correction

$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{M}_i \Sigma_i^- \mathcal{M}_i^T + \Sigma_{m_i}\right]^{-1}$$

$$\overline{x}_i^+ = \overline{x}_i^- + \mathcal{K}_i \left[y_i - \mathcal{M}_i \overline{x}_i^-\right]$$

$$\Sigma_i^+ = [Id - \mathcal{K}_i \mathcal{M}_i]\, \Sigma_i^-$$

Linear measurement model

Algorithm 11.3: The Kalman Filter.

Dynamic Model:

$$\mathbf{x}_i = f(\mathbf{x}_{i-1}, \mathbf{n})$$

$$\mathbf{y}_i = g(\mathbf{x}_i, \mathbf{n})$$

Start Assumptions: $\overline{\boldsymbol{x}}_0^-$ and Σ_0^- are known

Update Equations: Prediction $\overline{x}_i^-$ Σ_i^-

$$\mathbf{x}_i \sim \mathcal{N}\left(f(\mathbf{x}_{i-1}, \mathbf{0}), \mathcal{F}_x \Sigma_{i-1}^+ \mathcal{F}_x^T + \mathcal{F}_n \Sigma_{n,i} \mathcal{F}_n^T\right)$$

Update Equations: Correction

$$\mathcal{K}_i = \Sigma_i^- \mathcal{M}_i^T \left[\mathcal{G}_x \Sigma_i^- \mathcal{G}_x^T + \mathcal{G}_n \Sigma_{m,i} \mathcal{G}_n^T\right]^{-1}$$

$$\overline{\boldsymbol{x}}_i^+ = \overline{\boldsymbol{x}}_i^- + \mathcal{K}_i \left[\mathbf{y}_i - g(\mathbf{x}_i^-, \mathbf{0})\right]$$

$$\Sigma_i^+ = [Id - \mathcal{K}_i \mathcal{G}_x] \Sigma_i^-$$

The extended kalman filter

Outcome and issues

- In principle, can now filter position/orientation wrt map
  - linearize dynamics following recipe above
  - linearize measurements ditto
- There could be problems
  - EKF's are fine if the linearization is reliable
    - can be awful if not (next slides…)
  - in fact, the map points are uncertain
    - why not try to make/update map while moving? SLAM, to follow