COSC428 Computer Vision

Tracking – Kalman Filter

Readings: F&P Ch 17

Tracking Applications

- Motion capture
- Recognition from motion
- Surveillance
- Targeting

Things to consider in tracking

What are the

- Real world dynamics
- Approximate / assumed model
- Observation / measurement process

Density propogation

- Tracking == Inference over time
- Much simplification is possible with linear dynamics and Gaussian probability models

Tracking and Recursive estimation

- Real-time / interactive imperative.
- Task: At each time point, re-compute estimate of position or pose.
  - At time n, fit model to data using time 0…n
  - At time n+1, fit model to data using time 0…n+1
- Repeat batch fit every time?

Recursive estimation

- Decompose estimation problem
  - part that depends on new observation
  - part that can be computed from previous history

- E.g., running average:

$$a_t = \alpha\, a_{t-1} + (1-\alpha)\, y_t$$

- Linear Gaussian models: Kalman Filter
- First, general framework…

Tracking

- Very general model:
  - We assume there are moving objects, which have an underlying state X
  - There are measurements 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, measurements are stereo pairs
  - object is person, state is body configuration, measurements are frames, clock is in camera (30 fps)

Three main issues in tracking

- **Prediction:** we have seen $\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}$ — what state does this set of measurements predict for the i'th frame? to solve this problem, we need to obtain a representation of $P(\boldsymbol{X}_i | \boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_{i-1} = \boldsymbol{y}_{i-1})$.

- **Data association:** Some of the measurements obtained from the i-th frame may tell us about the object's state. Typically, we use $P(\boldsymbol{X}_i | \boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_{i-1} = \boldsymbol{y}_{i-1})$ to identify these measurements.

- **Correction:** now that we have $\boldsymbol{y}_i$ — the relevant measurements — we need to compute a representation of $P(\boldsymbol{X}_i | \boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_i = \boldsymbol{y}_i)$.

Simplifying Assumptions

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

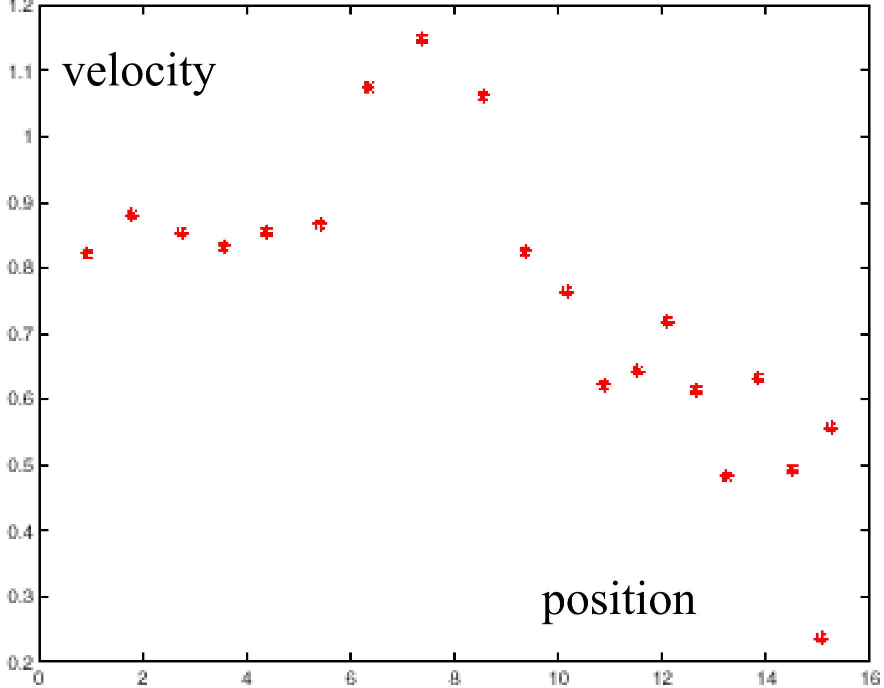
$$P(\boldsymbol{X}_i|\boldsymbol{X}_1,\ldots,\boldsymbol{X}_{i-1}) = P(\boldsymbol{X}_i|\boldsymbol{X}_{i-1})$$

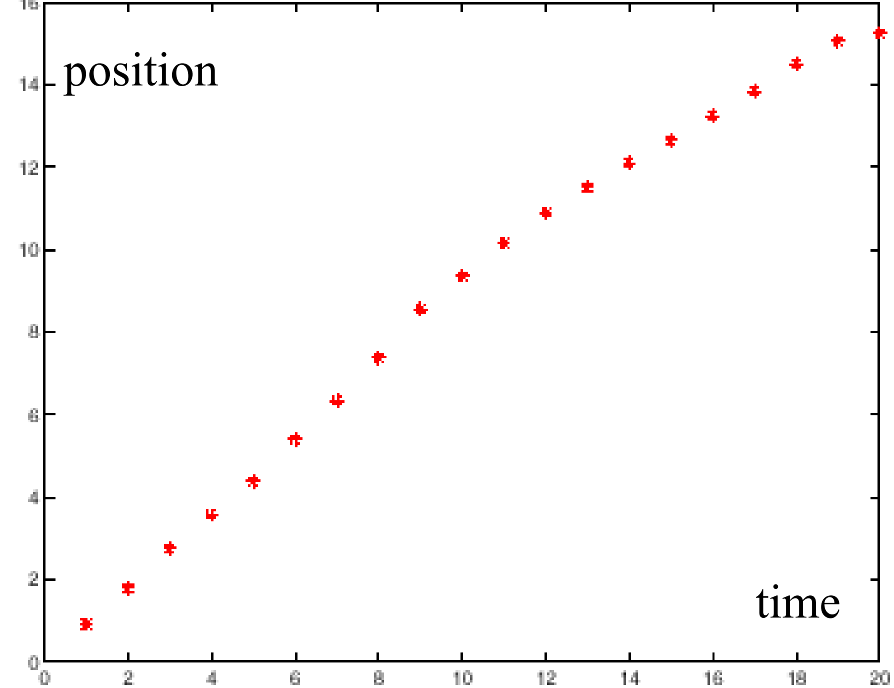
  This assumption hugely simplifies the design of algorithms, as we shall see; furthermore, it isn't terribly restrictive if we're clever about interpreting $\boldsymbol{X}_i$ as we shall show in the next section.

- **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

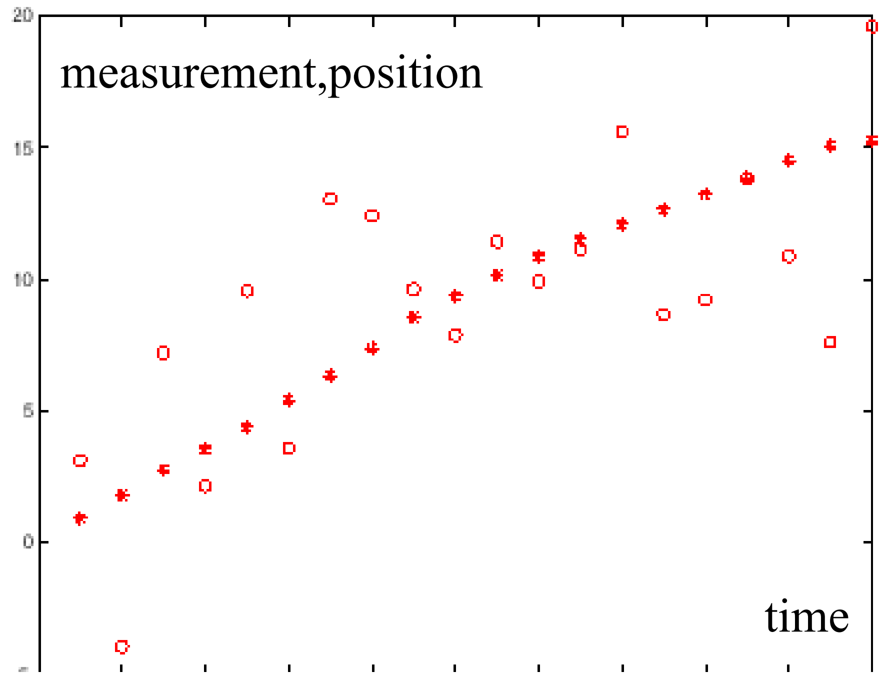
$$P(\boldsymbol{Y}_i, \boldsymbol{Y}_j, \ldots \boldsymbol{Y}_k|\boldsymbol{X}_i) = P(\boldsymbol{Y}_i|\boldsymbol{X}_i)P(\boldsymbol{Y}_j, \ldots, \boldsymbol{Y}_k|\boldsymbol{X}_i)$$

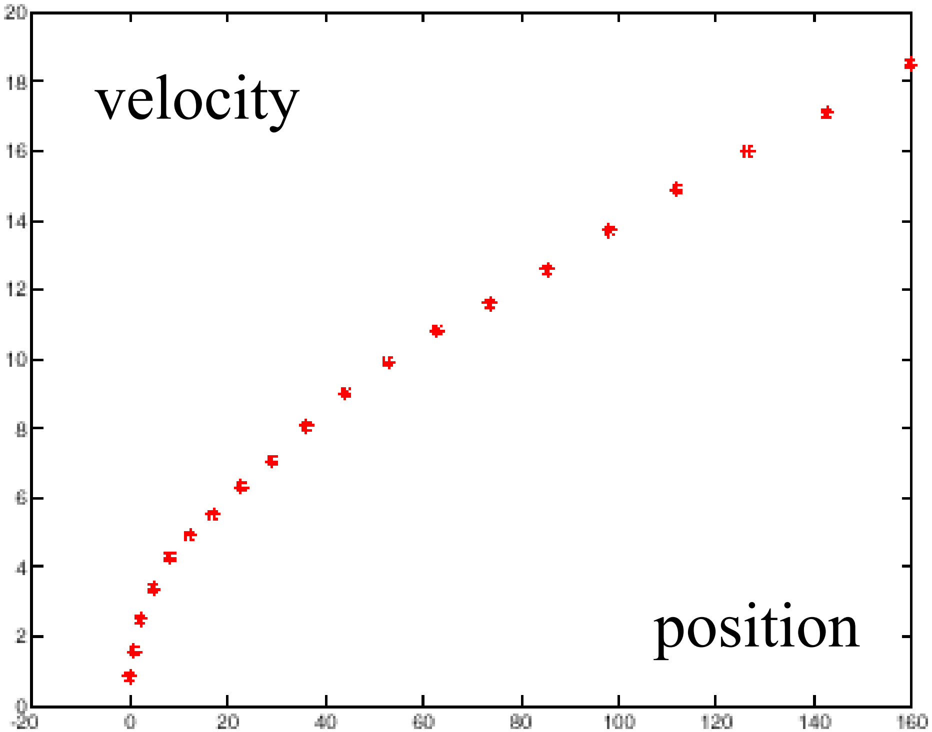
  Again, this isn't a particularly restrictive or controversial assumption, but it yields important simplifications.

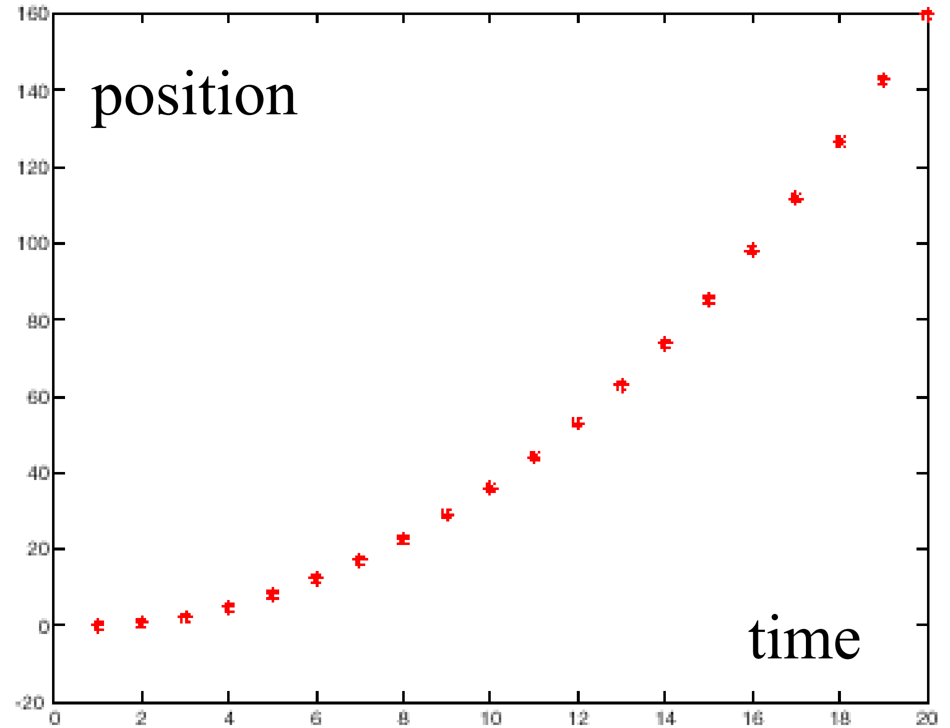




Constant Velocity Model







Constant Acceleration Model

The Kalman Filter

- Key ideas:
  – Linear models interact uniquely well with Gaussian noise - make the prior Gaussian, everything else Gaussian and the calculations are easy
  – Gaussians are really easy to represent --- once you know the mean and covariance, you’re done

Recall the three main issues in tracking

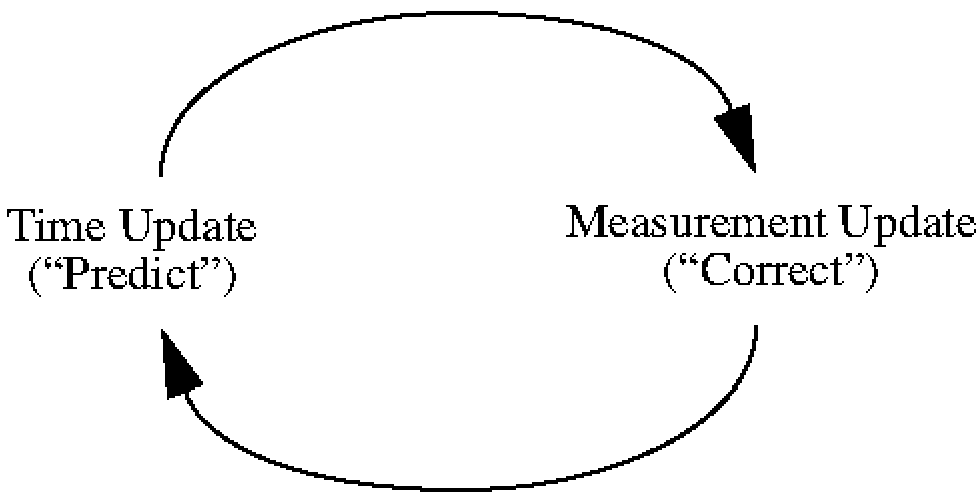
- **Prediction:** we have seen $\boldsymbol{y}_0, \ldots, \boldsymbol{y}_{i-1}$ — what state does this set of measurements predict for the i'th frame? to solve this problem, we need to obtain a representation of $P(\boldsymbol{X}_i | \boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_{i-1} = \boldsymbol{y}_{i-1})$.

- **Data association:** Some of the measurements obtained from the i-th frame may tell us about the object's state. Typically, we use $P(\boldsymbol{X}_i | \boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_{i-1} = \boldsymbol{y}_{i-1})$ to identify these measurements.

- **Correction:** now that we have $\boldsymbol{y}_i$ — the relevant measurements — we need to compute a representation of $P(\boldsymbol{X}_i | \boldsymbol{Y}_0 = \boldsymbol{y}_0, \ldots, \boldsymbol{Y}_i = \boldsymbol{y}_i)$.

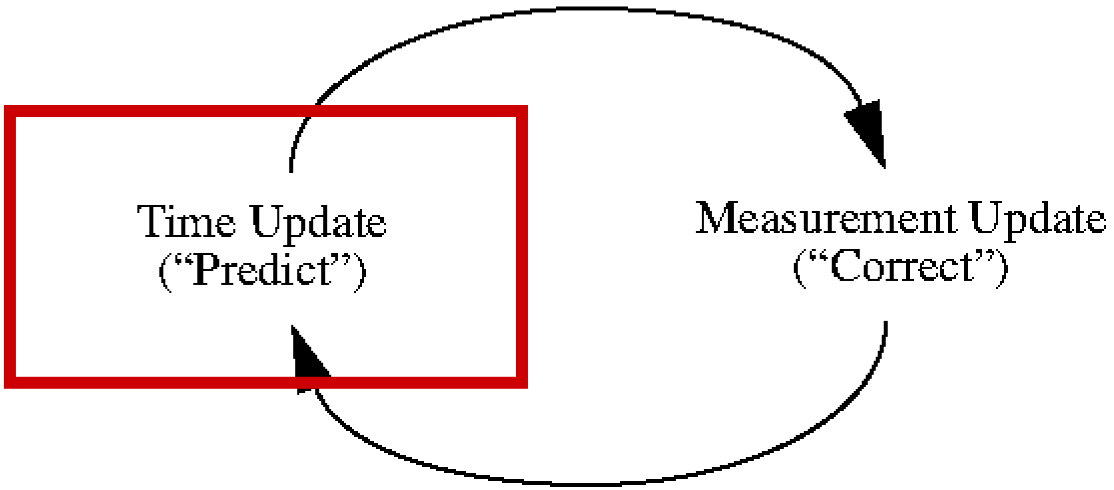
(Ignore data association for now)

The Kalman Filter



[figure from http://www.cs.unc.edu/~welch/kalman/kalmanIntro.html]

The Kalman Filter



Prediction for 1D Kalman filter

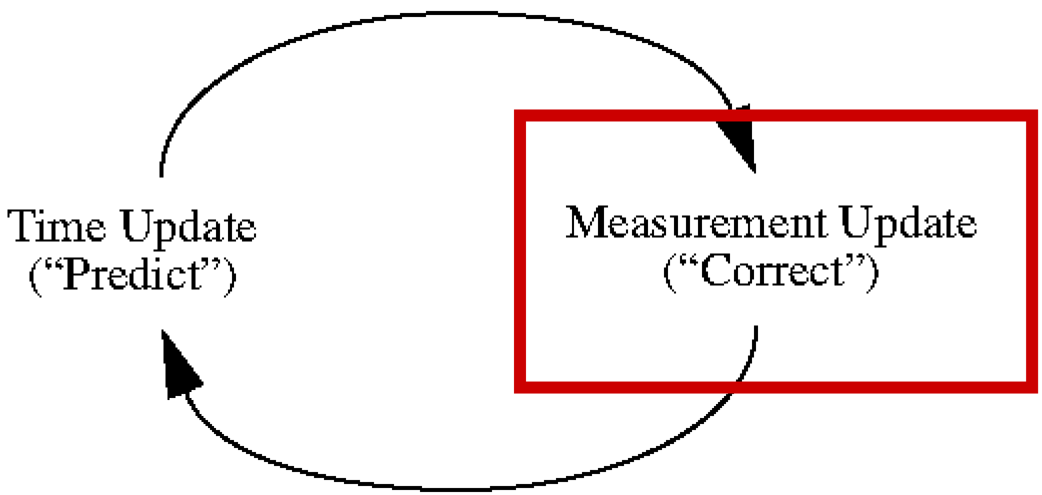
$$\boxed{x_i \sim N(d_i x_{i-1}, \sigma_{d_i}^2)}$$

- The new state is obtained by
  - multiplying old state by known constant
  - adding zero-mean noise

- Therefore, predicted mean for new state is
  - constant times mean for old state
- Old variance is normal random variable
  - variance is multiplied by square of constant
  - and variance of noise is added.

$$\overline{X}_i^- = d_i \overline{X}_{i-1}^+ \qquad (\sigma_i^-)^2 = \sigma_{d_i}^2 + (d_i \sigma_{i-1}^+)^2$$

The Kalman Filter



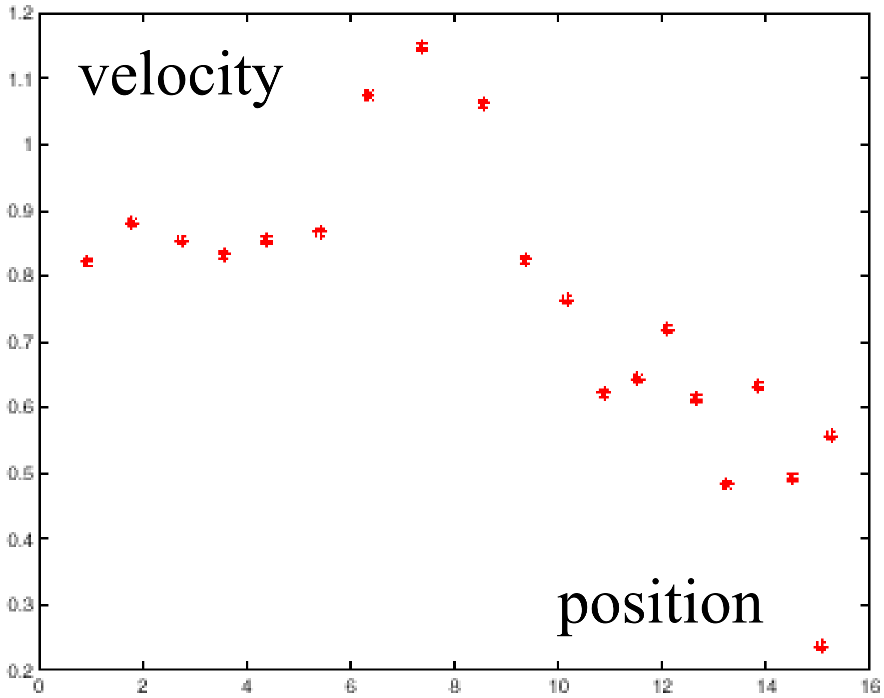
Correction for 1D Kalman filter

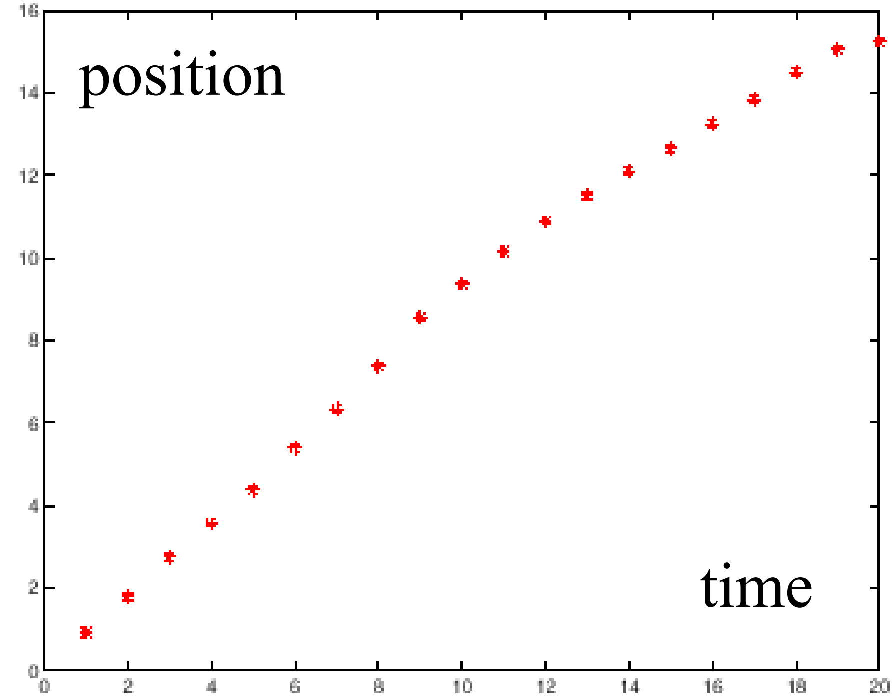
$$x_i^+ = \left(\frac{\overline{x}_i^- \sigma_{m_i}^2 + m_i y_i (\sigma_i^-)^2}{\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2} \right)$$

$$\sigma_i^+ = \sqrt{\left(\frac{\sigma_{m_i}^2 (\sigma_i^-)^2}{(\sigma_{m_i}^2 + m_i^2 (\sigma_i^-)^2)} \right)}$$

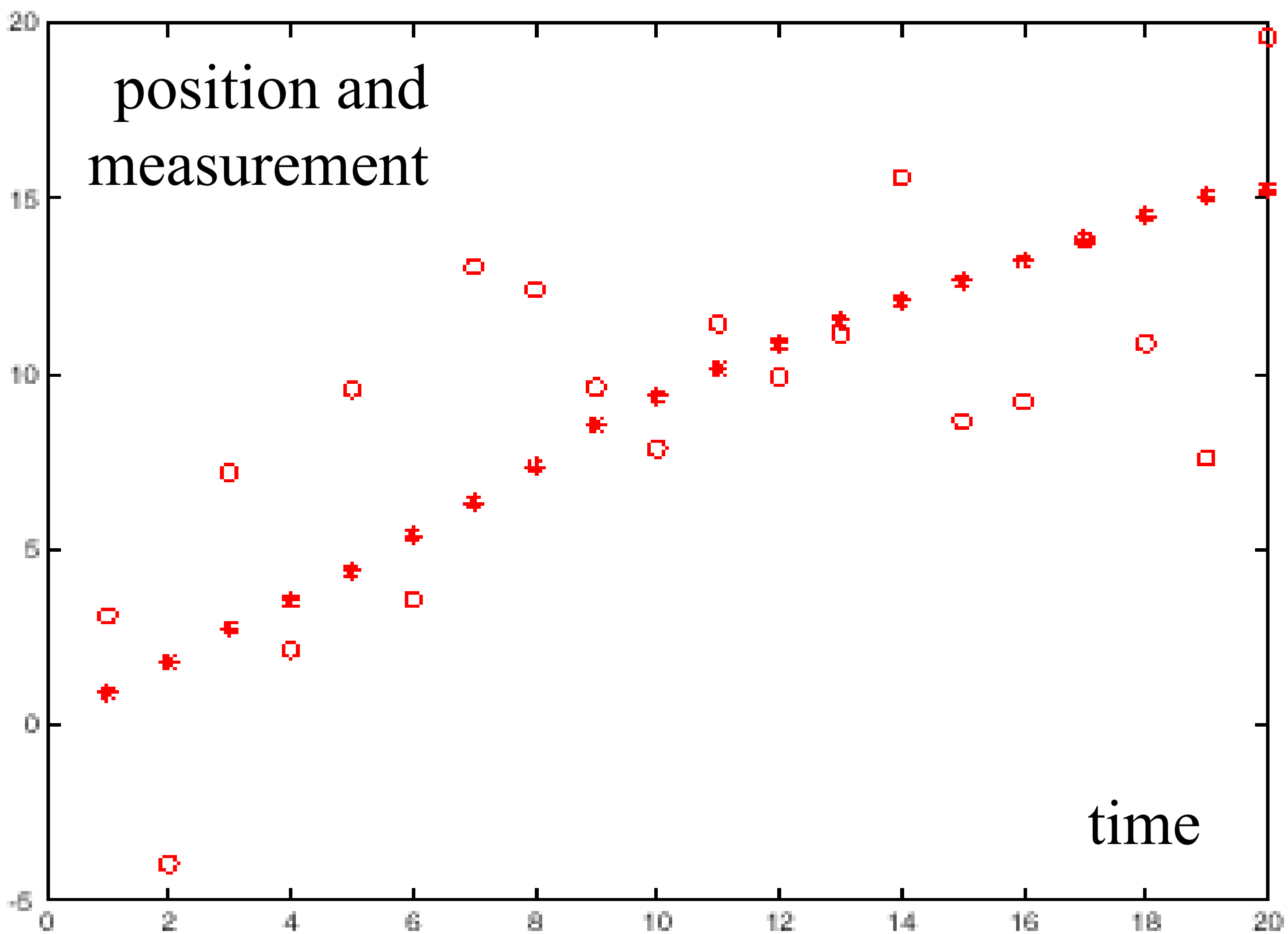
Notice:

– if measurement noise is small,
we rely mainly on the measurement,
– if it’s large, mainly on the prediction
– σ does not depend on y

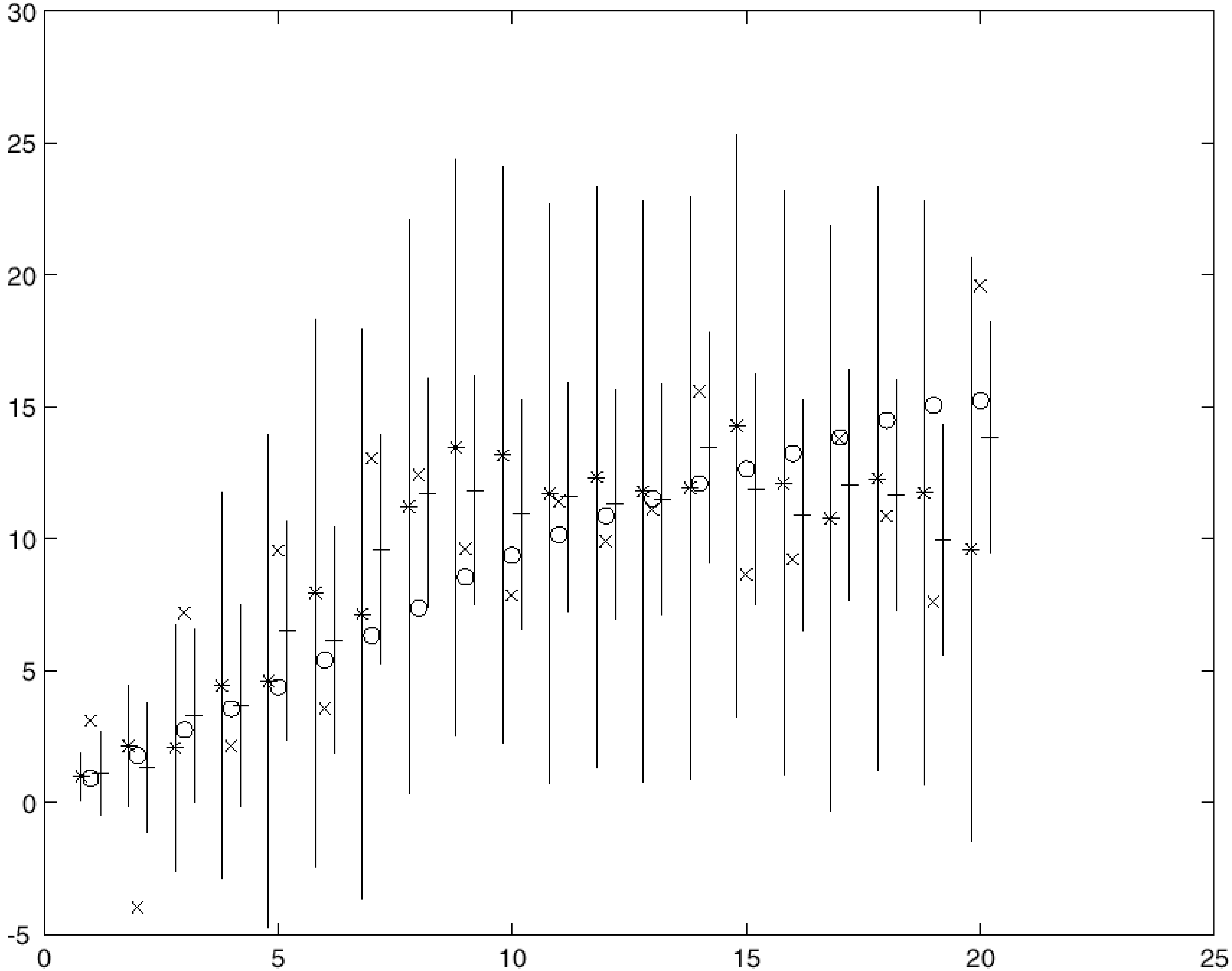




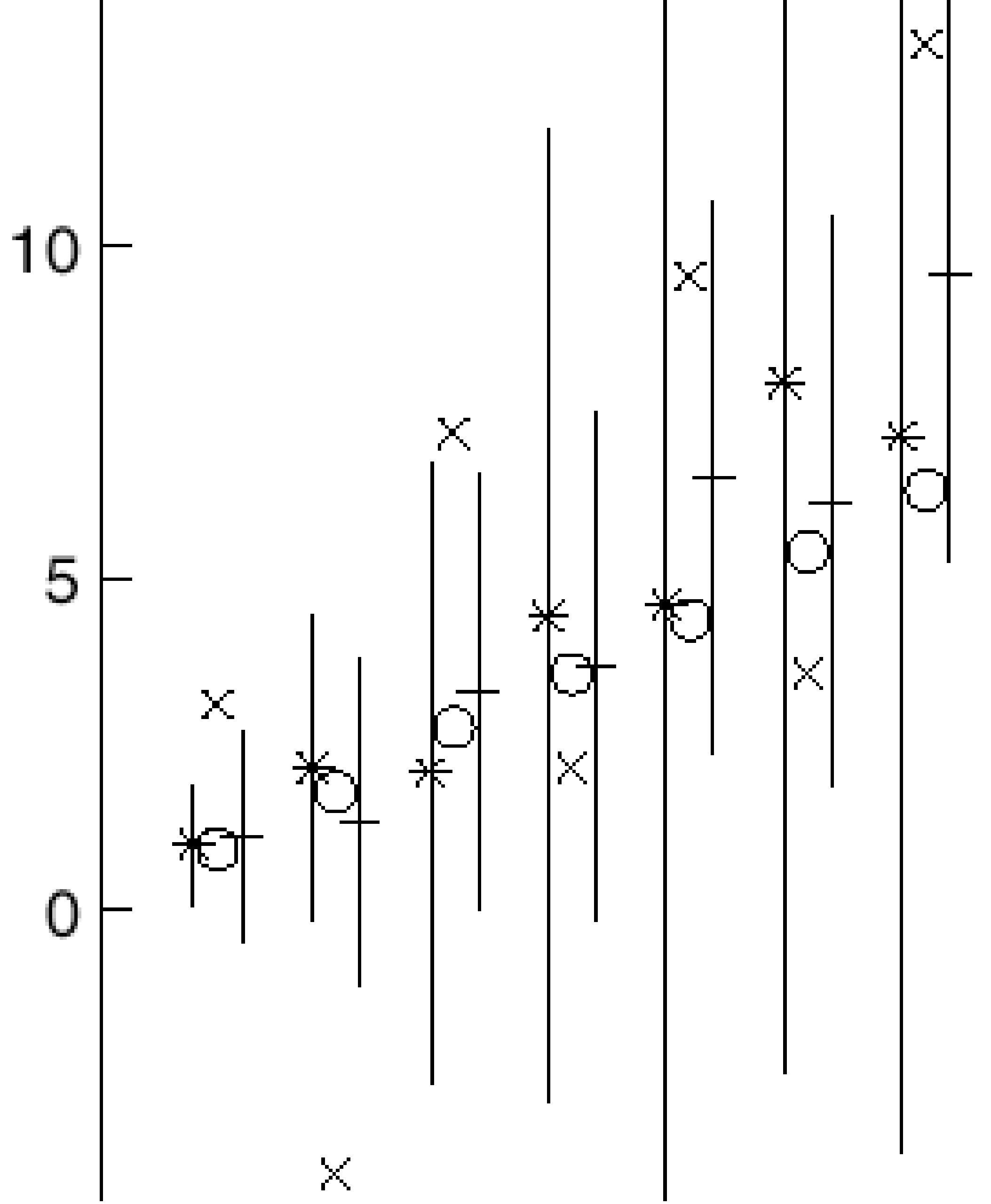
Constant
Velocity
Model



position and measurement
time
20
15
10
5
0
-5
0
2
4
6
8
10
12
14
16
18
20

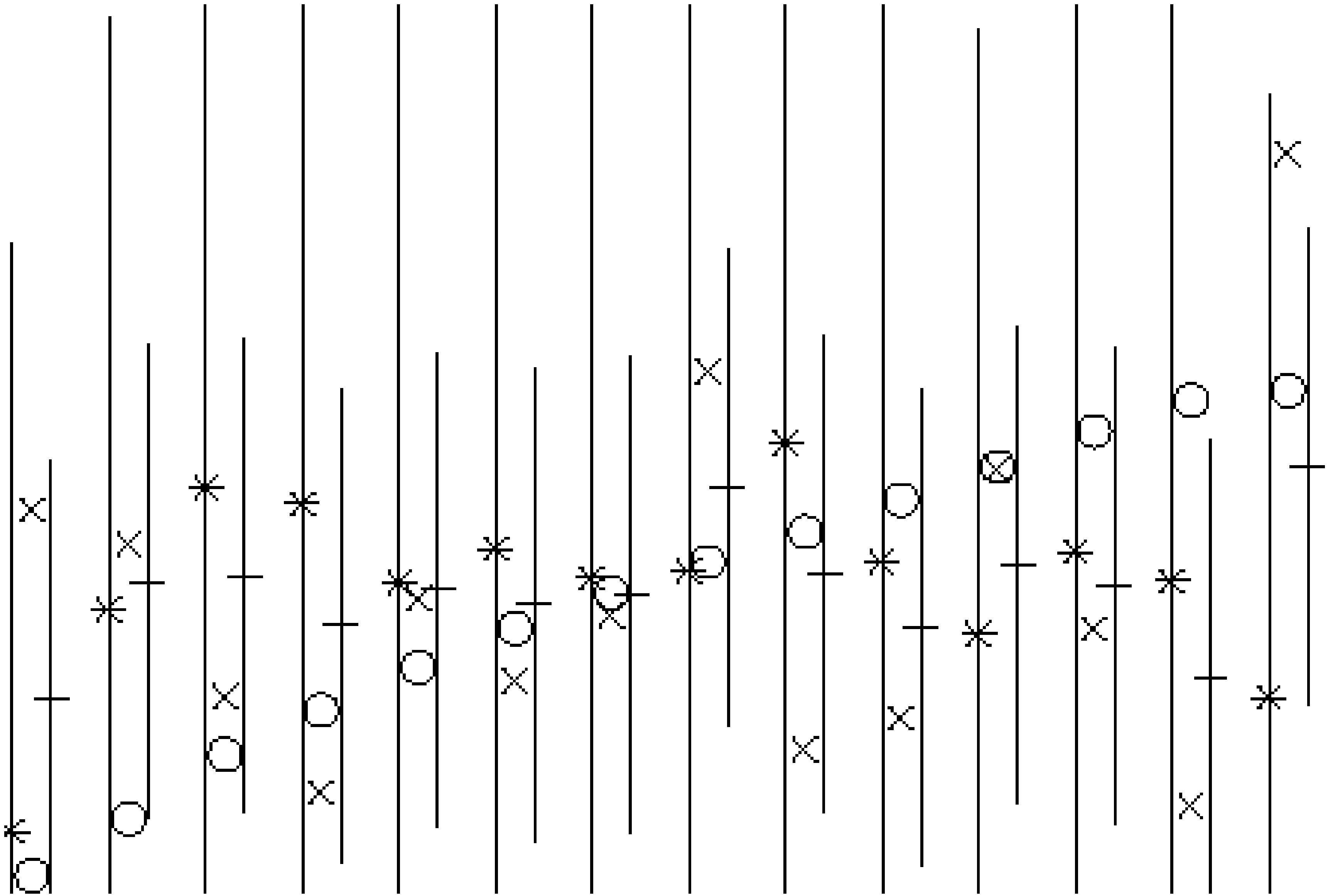


The *-s give $\overline{\boldsymbol{x}}_i^-$, +-s give $\overline{\boldsymbol{x}}_i^+$, vertical bars are 3 standard deviation bars



The o-s give state, x-s measurement.

The *-s give $\overline{\boldsymbol{x}}_i^-$, +-s give $\overline{\boldsymbol{x}}_i^+$, vertical bars are 3 standard deviation bars

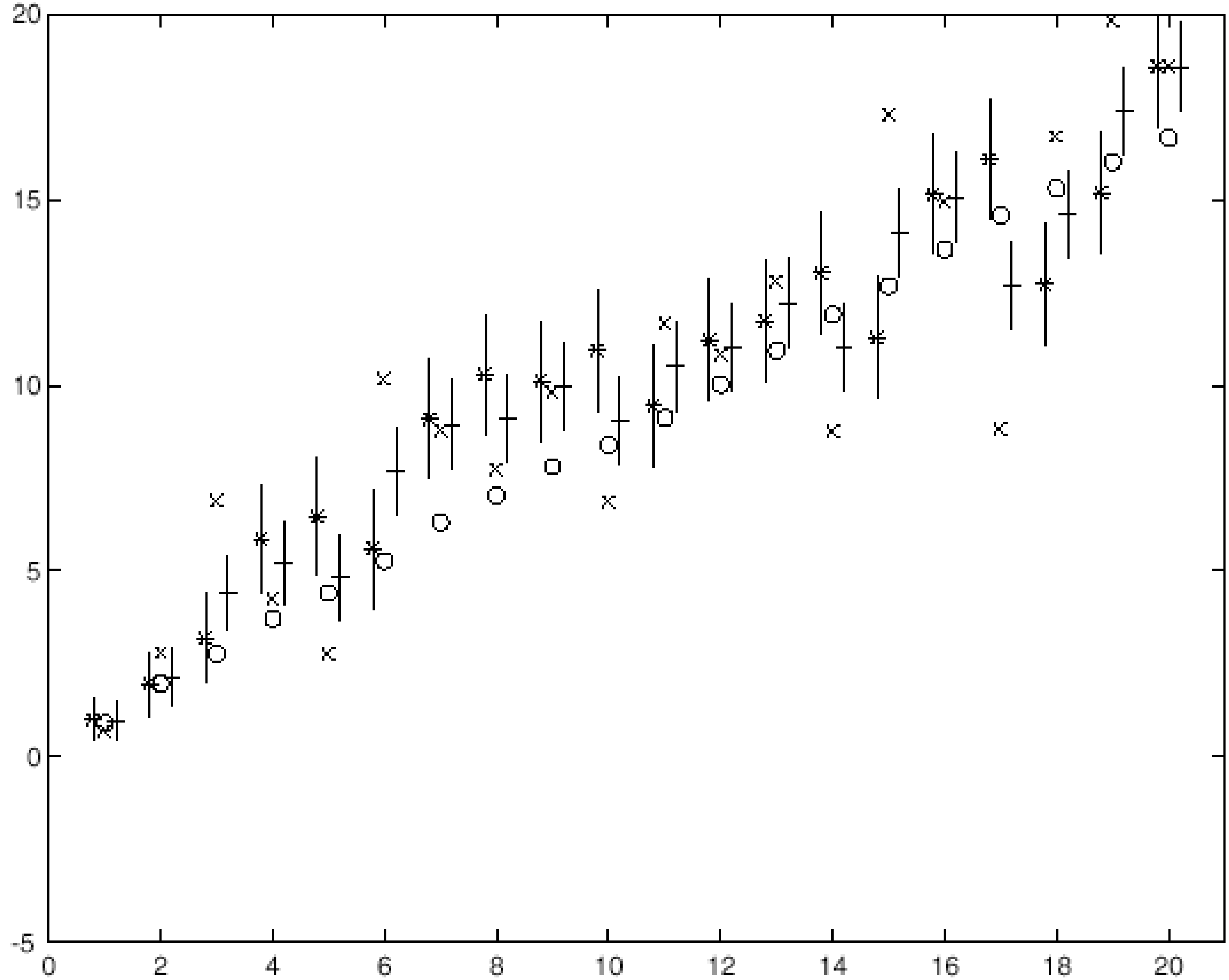


The o-s give state, x-s measurement.

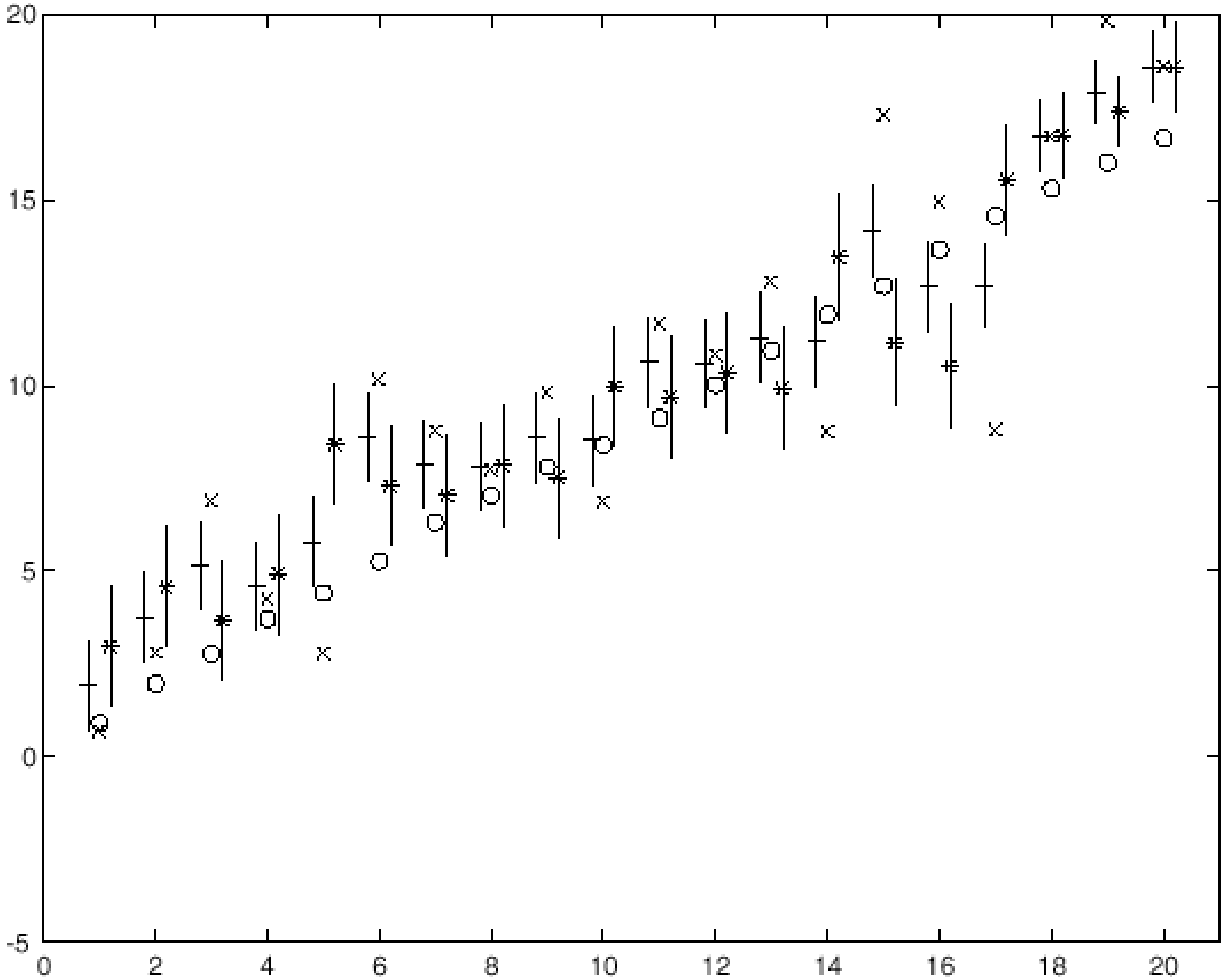
The *-s give $\overline{\boldsymbol{x}}_i^-$, +-s give $\overline{\boldsymbol{x}}_i^+$, vertical bars are 3 standard deviation bars

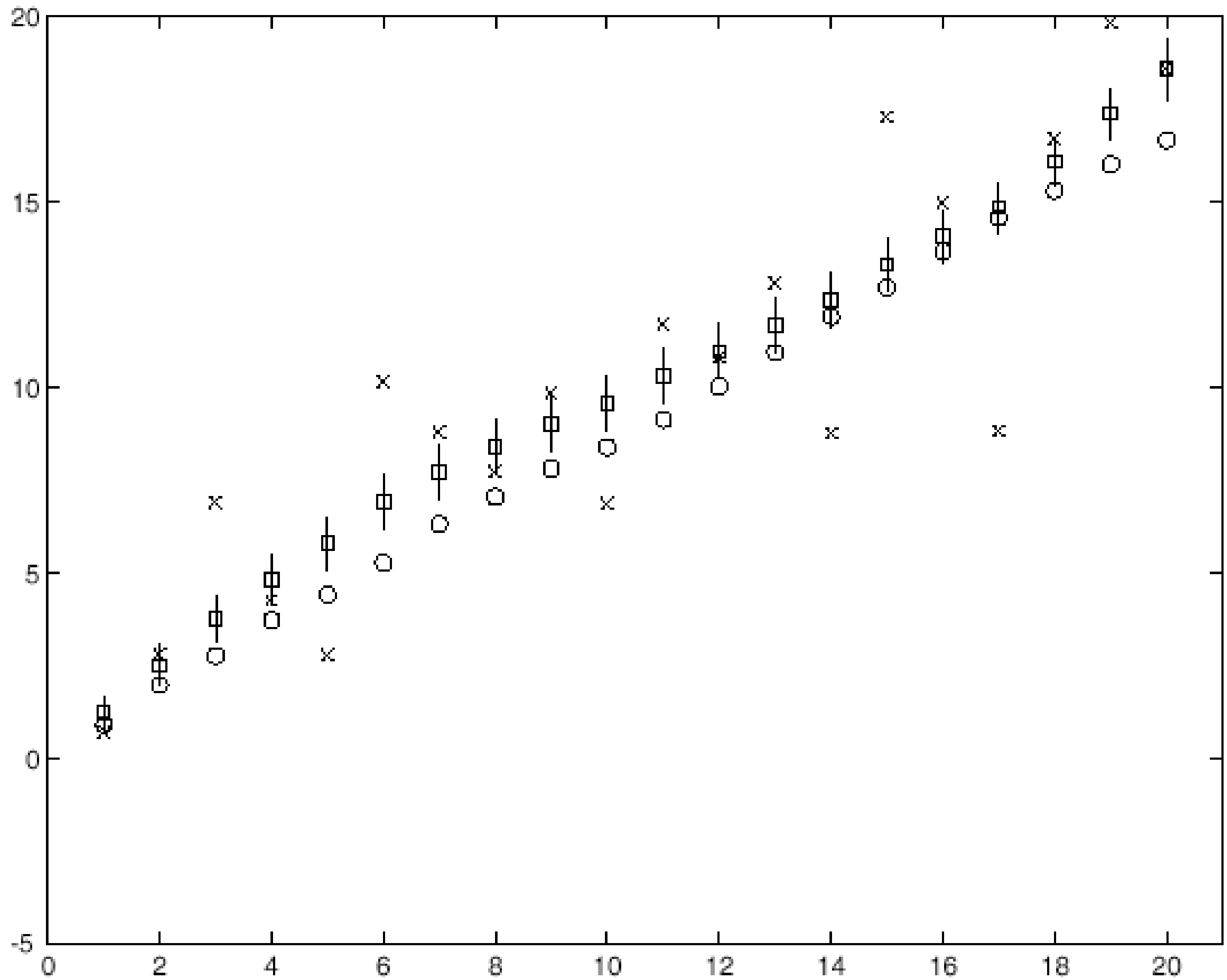
Smoothing

- Idea
  - We don’t have the best estimate of state - what about the future?
  - Run two filters, one moving forward, the other backward in time.
  - Now combine state estimates
    - The crucial point here is that we can obtain a smoothed estimate by viewing the backward filter’s prediction as yet another measurement for the forward filter



20
15
10
5
0
-5
0
2
4
6
8
10
12
14
16
18
20


20
15
10
5
0
-5
0
2
4
6
8
10
12
14
16
18
20

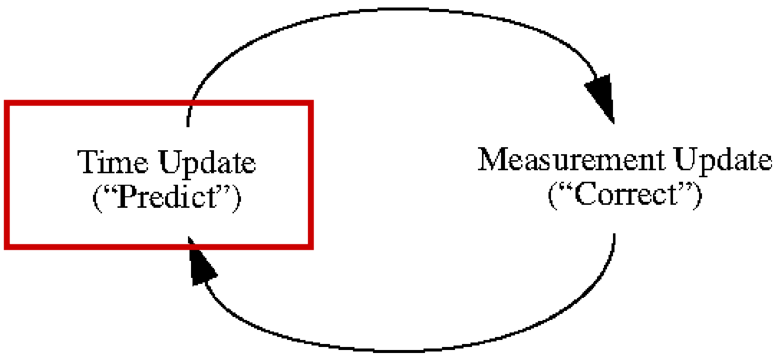

20
15
10
5
0
-5
0
2
4
6
8
10
12
14
16
18
20

n-D

Generalization to n-D is straightforward but more complex.

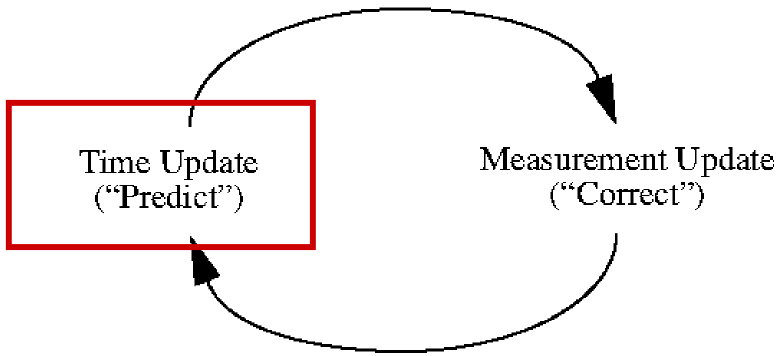
n-D

Generalization to n-D is straightforward but more complex.



n-D Prediction

Generalization to n-D is straightforward but more complex.



Prediction:

- Multiply estimate at prior time with forward model:

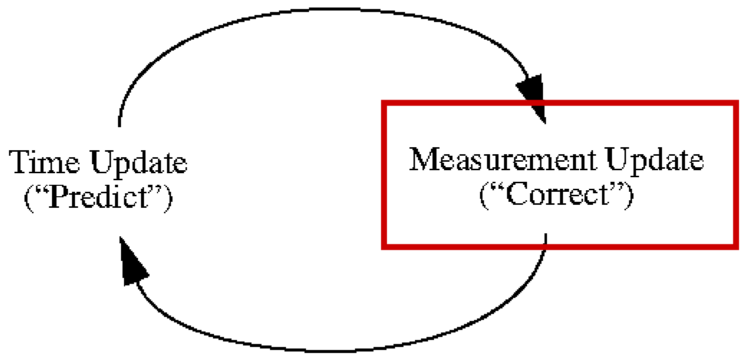
$$\overline{\boldsymbol{x}}_i^- = \mathcal{D}_i \overline{\boldsymbol{x}}_{i-1}^+$$

- Propagate covariance through model and add new noise:

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

n-D Correction

Generalization to n-D is straightforward but more complex.



Correction:

- Update *a priori* estimate with measurement to form *a posteriori*

Resources

- Kalman filter homepage

http://www.cs.unc.edu/~welch/kalman/

- Kevin Murphy's Matlab toolbox:

http://www.ai.mit.edu/~murphyk/Software/Kalman/kalman.html