ACM 116: The Kalman filter

- Example
- General Setup
- Derivation
- Numerical examples
  - Estimating the voltage
  - 1D tracking
  - 2D tracking

Example: Navigation Problem

- Truck on “frictionless” straight rails
- Initial position $X_0 = 0$
- Movement is buffeted by random accelerations
- We measure the position every Δt seconds
- State variables (X_k, V_k) position and velocity at time $k\Delta t$

$$
\begin{aligned}
X_k &= X_{k-1} + V_{k-1}\Delta t + a_{k-1}\Delta t^2/2 \\
V_k &= V_{k-1} + a_{k-1}\Delta t
\end{aligned}
$$

  where a_{k-1} is a random acceleration

- Observations

$$Y_k = X_k + Z_k$$

  where Z_k is a noise term.

The goal is to estimate the position and velocity at all times.

General Setup

Estimation of a stochastic dynamic system

- Dynamics

$$\boldsymbol{X_k = F_{k-1} X_{k-1} + B_{k-1} u_{k-1} + W_{k-1}}$$

  - $\boldsymbol{X_k}$: state of the system at time $\boldsymbol{k}$
  - $\boldsymbol{u_{k-1}}$: control-input
  - $\boldsymbol{W_{k-1}}$: noise

- Observations

$$\boldsymbol{Y_k = H_k X_k + Z_k}$$

  - $\boldsymbol{Y_k}$ is observed
  - $\boldsymbol{Z_k}$ is noise

- The noise realizations are all independent
- Goal: predict state $\boldsymbol{X_k}$ from past data $\boldsymbol{Y_0, Y_1, \ldots, Y_{k-1}}$.

Derivation

Derivation in the simpler model where the dynamics is of the form

$$X_k = a_{k-1}X_{k-1} + W_{k-1}$$

and the observations

$$Y_k = X_k + Z_k$$

The objective is to find, for each time k, the minimum MSE filter based on $Y_0, Y_1, \ldots, Y_{k-1}$

$$\hat{X}_k = \sum_{j=1}^{k} h_j^{(k-1)} Y_{k-j}$$

To find the filter, we apply the orthogonality principle

$$E((X_k - \sum_{j=1}^{k} h_j^{(k-1)} Y_{k-j})Y_\ell) = 0, \quad \ell = 0, 1, \ldots, k-1.$$

Recursion

The beautiful thing about the Kalman filter is that one can *almost* deduce the optimal filter to predict X_{k+1} from that predicting X_k.

$$h_{j+1}^{(k)} = (a_k - h_1^{(k)})h_j^{(k-1)}, \quad j = 1, \ldots, k.$$

Given the filter $h^{(k-1)}$, we only need to find $h_1^{(k)}$ to get the filter at the next time step.

How to find $h_1^{(k)}$?

Observe that the next prediction is equal to

$$\begin{aligned}\hat{X}_{k+1} &= h_1^{(k)} Y_k + \sum_{j=1}^{k} (a_k - h_1^{(k)}) h_j^{(k-1)} Y_{k-j} \\ &= a_k \hat{X}_k + h_1^{(k)} (Y_k - \hat{X}_k)\end{aligned}$$

Interpretation

$$\hat{X}_{k+1} = a_k \hat{X}_k + h_1^{(k)} I_k$$

- $a_k \hat{X}_k$ is the prediction based on the estimate at time k
- $h_1^{(k)} I_k$ is a corrective term which is available since we now see Y_k
  - $h_1^{(k)}$ is called the gain
  - $I_k = Y_k - \hat{X}_k$ is called the innovation

Error of Prediction

To find $h_1^{(k)}$, we look at the error of prediction

$$\epsilon_k = X_k - \hat{X}_k$$

We have the recursion

$$\epsilon_{k+1} = (a_k - h_1^{(k)})\epsilon_k + W_k - h_1^{(k)} Z_k$$

- $\epsilon_0 = Z_0$
- $E(\epsilon_k) = 0$
- $E(\epsilon_{k+1}^2) = [a_k - h_1^{(k)}]^2 E(\epsilon_k^2) + E(W_k^2) + [h_1^{(k)}]^2 E(Z_k^2)$

To minimize the MSE ϵ_{k+1}, we adjust $h_1^{(k)}$ so that

$$\partial_{h_1^{(k)}} E(\epsilon_{k+1}^2) = 0 = -2(a_k - h_1^{(k)})E(\epsilon_k^2) + 2h_1^{(k)} E(Z_k^2)$$

which is given by

$$h_1^{(k)} = \frac{a_k E(\epsilon_k^2)}{E(\epsilon_k^2) + E(Z_k^2)}$$

Note that this gives the recurrence relation

$$E(\epsilon_{k+1}^2) = a_k(a_k - h_1^{(k)})E(\epsilon_k^2) + E(W_k^2)$$

The Kalman Filter Algorithm

- Initialization $\hat{X}_0 = 0$, $E(\epsilon_0^2) = E(Z_0^2)$
- Loop: for $k = 0, 1, \ldots$

$$h_1^{(k)} = \frac{a_k E(\epsilon_k^2)}{E(\epsilon_k^2) + E(Z_k^2)}$$

$$\hat{X}_{k+1} = a_k \hat{X}_k + h_1^{(k)} (Y_k - \hat{X}_k)$$

$$E(\epsilon_{k+1}^2) = a_k (a_k - h_1^{(k)}) E(\epsilon_k^2) + E(W_k^2)$$

Benefits

- Requires no knowledge about the structure of $\boldsymbol{W_k}$ and $\boldsymbol{Z_k}$ (only variances)
- Easy implementation
- Many applications
  - Inertial guidance system
  - Autopilot
  - Satellite navigation system
  - Many others

General Formulation

$$\boldsymbol{X_k = F_{k-1}X_{k-1} + W_{k-1}}$$
$$\boldsymbol{Y_k = H_k X_k + Z_k}$$

The covariance of $\boldsymbol{W_k}$ is $\boldsymbol{Q_k}$ and that of $\boldsymbol{Z_k}$ is $\boldsymbol{R_k}$.

Two variables:

- $\boldsymbol{\hat{X}_{k|k}}$ estimate of the state at time $\boldsymbol{k}$ based upon $\boldsymbol{Y_0, \ldots, Y_{k-1}}$
- $\boldsymbol{E_{k|k}}$ error covariance matrix, $\boldsymbol{E_{k|k}} = \text{Cov}(\boldsymbol{X_k - \hat{X}_{k|k}})$

Prediction

$$\hat{\boldsymbol{X}}_{k+1|k} = \boldsymbol{F}_k \hat{\boldsymbol{X}}_{k|k-1}$$

$$\boldsymbol{E}_{k+1|k} = \boldsymbol{F}_k \boldsymbol{E}_{k|k-1} \boldsymbol{F}_k^T + \boldsymbol{Q}_k$$

Update

$$\boldsymbol{I}_k = \boldsymbol{Y}_k - \boldsymbol{H}_k \hat{\boldsymbol{X}}_{k+1|k} \qquad \text{Innovation}$$

$$\boldsymbol{S}_k = \boldsymbol{H}_k \boldsymbol{E}_{k+1|k} \boldsymbol{H}_k^T + \boldsymbol{R}_k \qquad \text{Innovation covariance}$$

$$\boldsymbol{K}_k = \boldsymbol{E}_{k+1|k} \boldsymbol{H}_k^T \boldsymbol{S}_k^{-1} \qquad \text{Kalman Gain}$$

$$\hat{\boldsymbol{X}}_{k+1|k+1} = \hat{\boldsymbol{X}}_{k+1|k} + \boldsymbol{K}_k \boldsymbol{I}_k \qquad \text{Updated state estimate}$$

$$\boldsymbol{E}_{k+1|k+1} = (\boldsymbol{Id} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{E}_{k+1|k} \qquad \text{Updated error covariance}$$

Estimating Constant Voltage

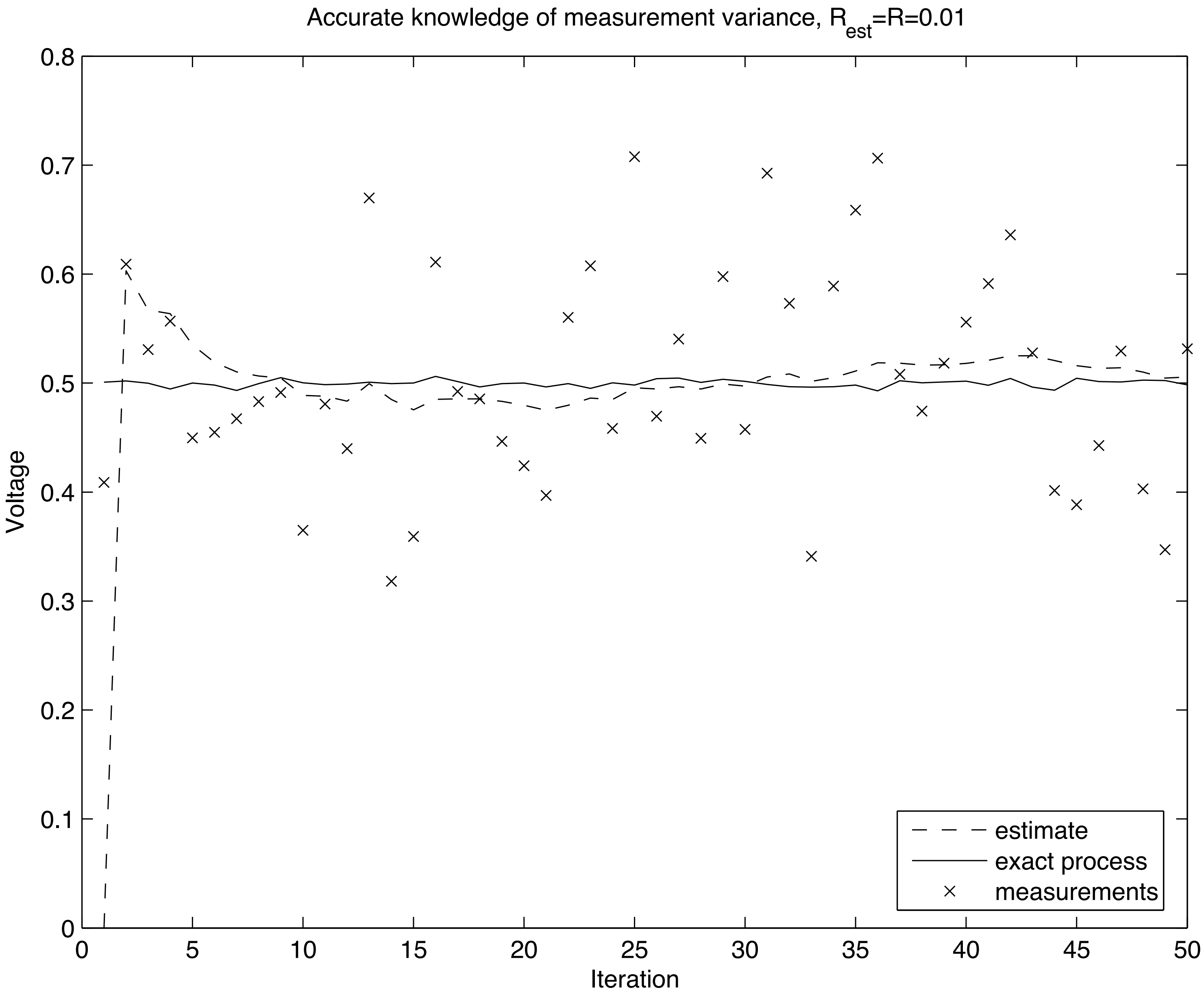
We wish to estimate some voltage which is almost constant except for some small random fluctuations. Our measuring device is imperfect (e.g. because of a poor A/D conversion). The process is governed by:

$$X_k = X_0 + W_k, \quad k = 1, 2, \ldots$$

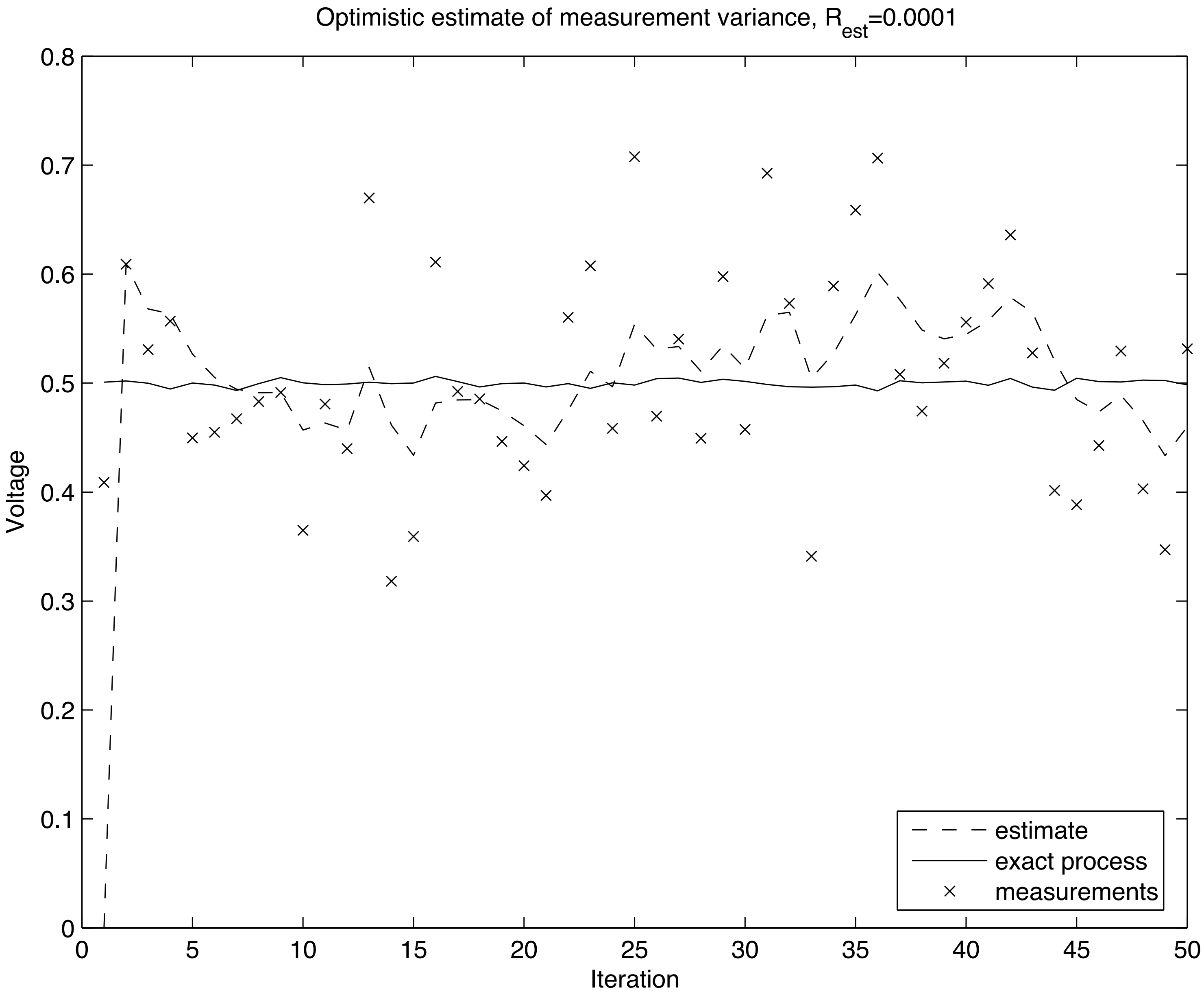
with $X_0 = 0.5V$, and the measurements are

$$Z_k = X_k + V_k, \quad k = 1, 2, \ldots$$

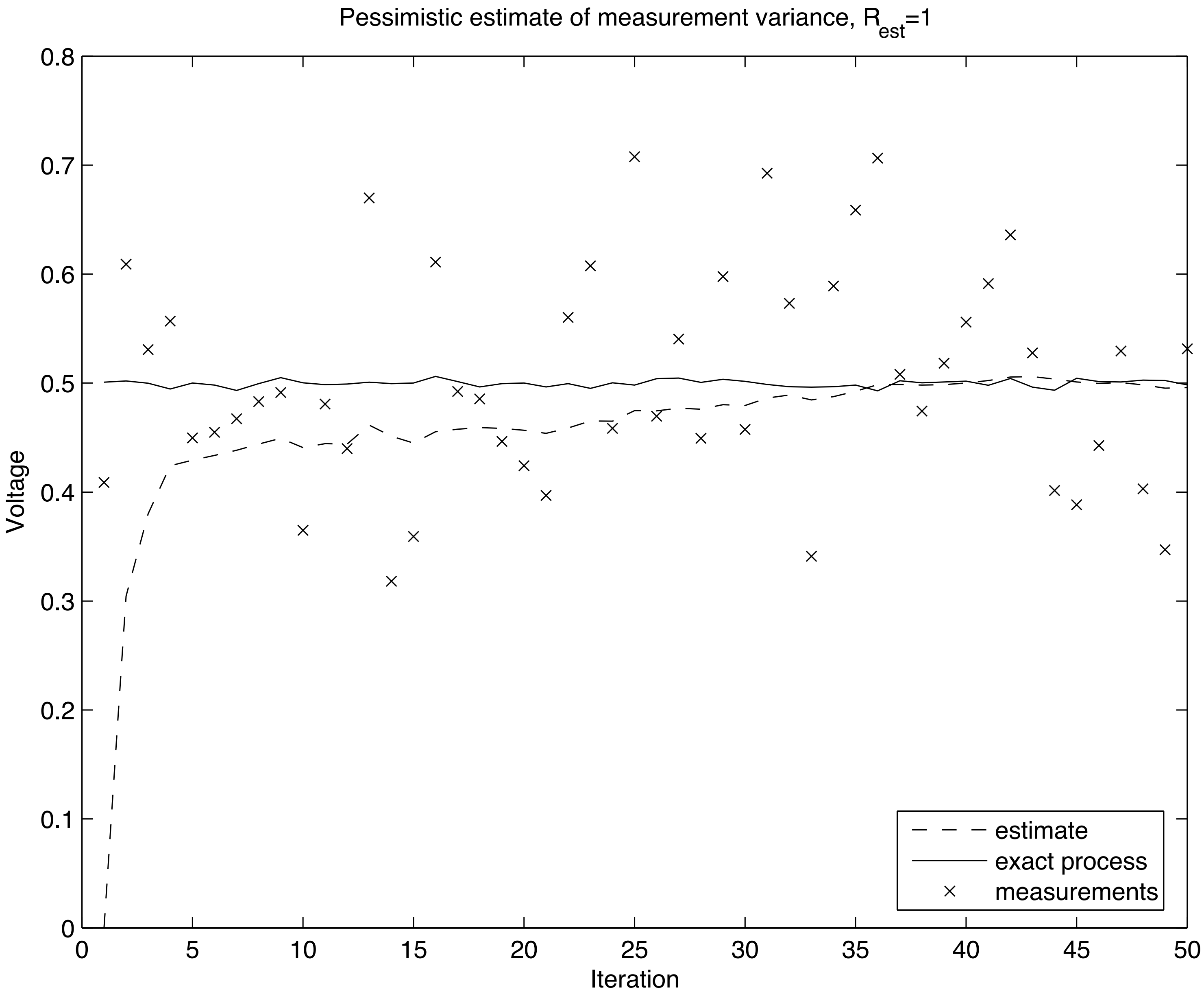
where W_k, V_k are uncorrelated Gaussian white noise processes, with $R :=$ Var$(V_k) = 0.01$, Var$(W_k) = 10^{-5}$.



Accurate knowledge of measurement variance, R_{est}=R=0.01
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Voltage
0
5
10
15
20
25
30
35
40
45
50
Iteration
estimate
exact process
measurements



Optimistic estimate of measurement variance, R_{est}=0.0001
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Voltage
0
5
10
15
20
25
30
35
40
45
50
Iteration
estimate
exact process
measurements



Pessimistic estimate of measurement variance, R_{est}=1
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
Voltage
0
5
10
15
20
25
30
35
40
45
50
Iteration
estimate
exact process
measurements

1D Tracking

Estimation of the position of a vehicle.

Let $\boldsymbol{X}$ be a state variable (position and speed), and $\boldsymbol{A}$ is a transition matrix

$$\boldsymbol{A} = \begin{pmatrix} \mathbf{1} & \boldsymbol{\Delta t} \\ \mathbf{0} & \mathbf{1} \end{pmatrix}.$$

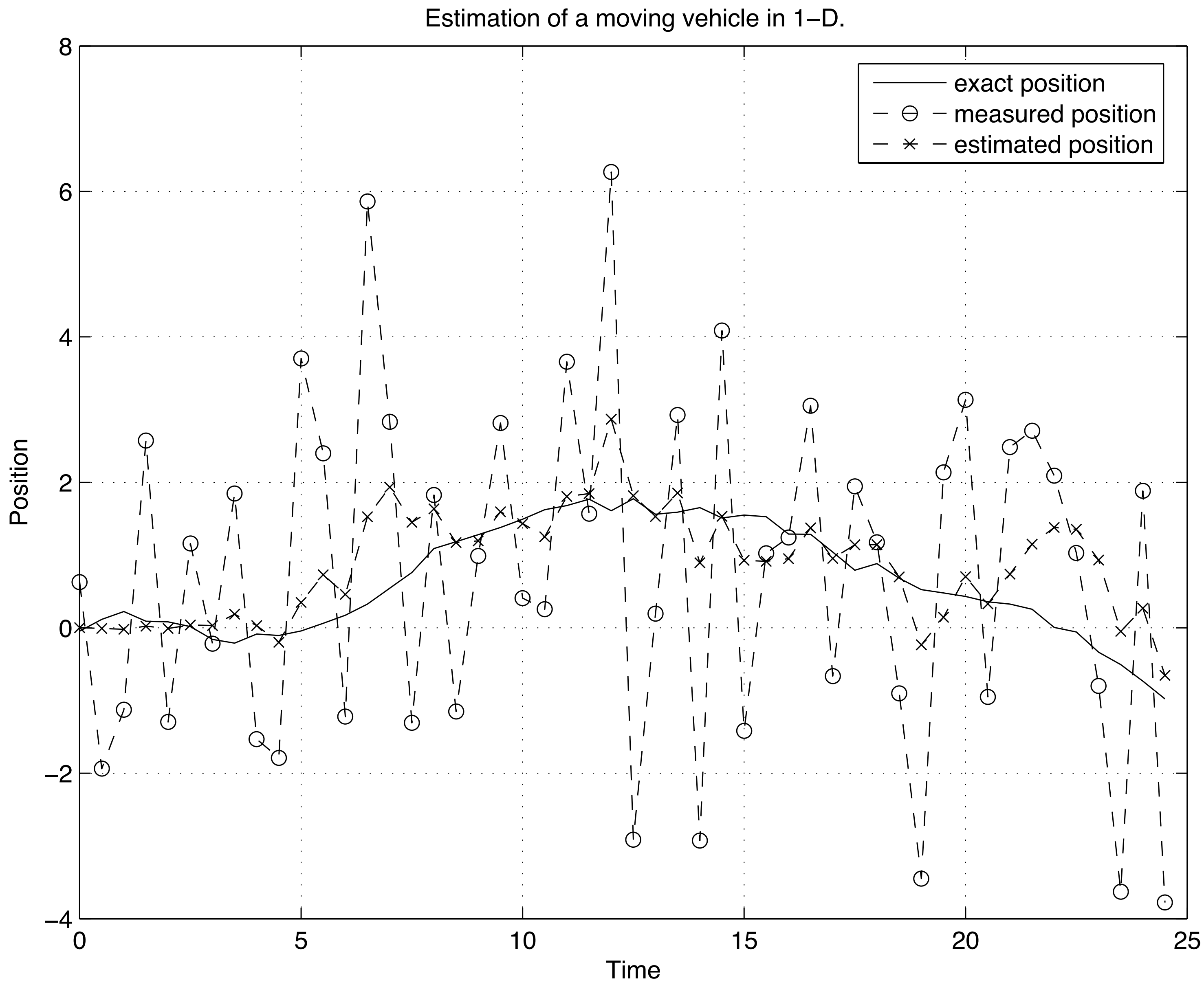
The process is governed by:

$$\boldsymbol{X}_{n+1} = \boldsymbol{A}\boldsymbol{X}_n + \boldsymbol{W}_n$$

where $\boldsymbol{W}_n$ is a zero-mean Gaussian white noise process. The observation is

$$\boldsymbol{Y}_n = \boldsymbol{C}\boldsymbol{X}_n + \boldsymbol{Z}_n$$

where the matrix $\boldsymbol{C}$ only picks up the position and $\boldsymbol{Z}_n$ is another zero-mean Gaussian white noise process independent of $\boldsymbol{W}_n$.


Estimation of a moving vehicle in 1−D.
exact position
measured position
estimated position
Position
Time
8
6
4
2
0
−2
−4
0
5
10
15
20
25

2D Example

General setup

$$\boldsymbol{X}(\boldsymbol{t}+\mathbf{1}) = \boldsymbol{F}\boldsymbol{X}(\boldsymbol{t}) + \boldsymbol{W}(\boldsymbol{t}), \quad \boldsymbol{W} \sim \boldsymbol{N}(\mathbf{0}, \boldsymbol{Q}),$$
$$\boldsymbol{Y}(\boldsymbol{t}) = \boldsymbol{H}\boldsymbol{X}(\boldsymbol{t}) + \boldsymbol{V}(\boldsymbol{t}), \quad \boldsymbol{V} \sim \boldsymbol{N}(\mathbf{0}, \boldsymbol{R})$$

Moving particle at constant velocity subject to random perturbations in its trajectory. The new position $(\boldsymbol{x1}, \boldsymbol{x2})$ is the old position plus the velocity $(\boldsymbol{dx1}, \boldsymbol{dx2})$ plus noise $\boldsymbol{w}$.

$$\begin{pmatrix} x_1(t) \\ x_2(t) \\ dx_1(t) \\ dx_2(t) \end{pmatrix} = \begin{pmatrix} 1 & 0 & 1 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} x_1(t-1) \\ x_2(t-1) \\ dx_1(t-1) \\ dx_2(t-1) \end{pmatrix} + \begin{pmatrix} w1(t-1) \\ w_2(t-1) \\ dw_1(t-1) \\ dw_2(t-1) \end{pmatrix}$$

Observations

We only observe the position of the particle.

$$\begin{pmatrix} \boldsymbol{y_1(t)} \\ \boldsymbol{y_2(t)} \end{pmatrix} = \begin{pmatrix} \mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{1} & \mathbf{0} & \mathbf{0} \end{pmatrix} \begin{pmatrix} \boldsymbol{x_1(t)} \\ \boldsymbol{x_2(t)} \\ \boldsymbol{dx_1(t)} \\ \boldsymbol{dx_2(t)} \end{pmatrix} + \begin{pmatrix} \boldsymbol{v_1(t)} \\ \boldsymbol{v_2(t)} \end{pmatrix}$$

Source: http://www.cs.ubc.ca/∼murphyk/Software/Kalman/kalman.html

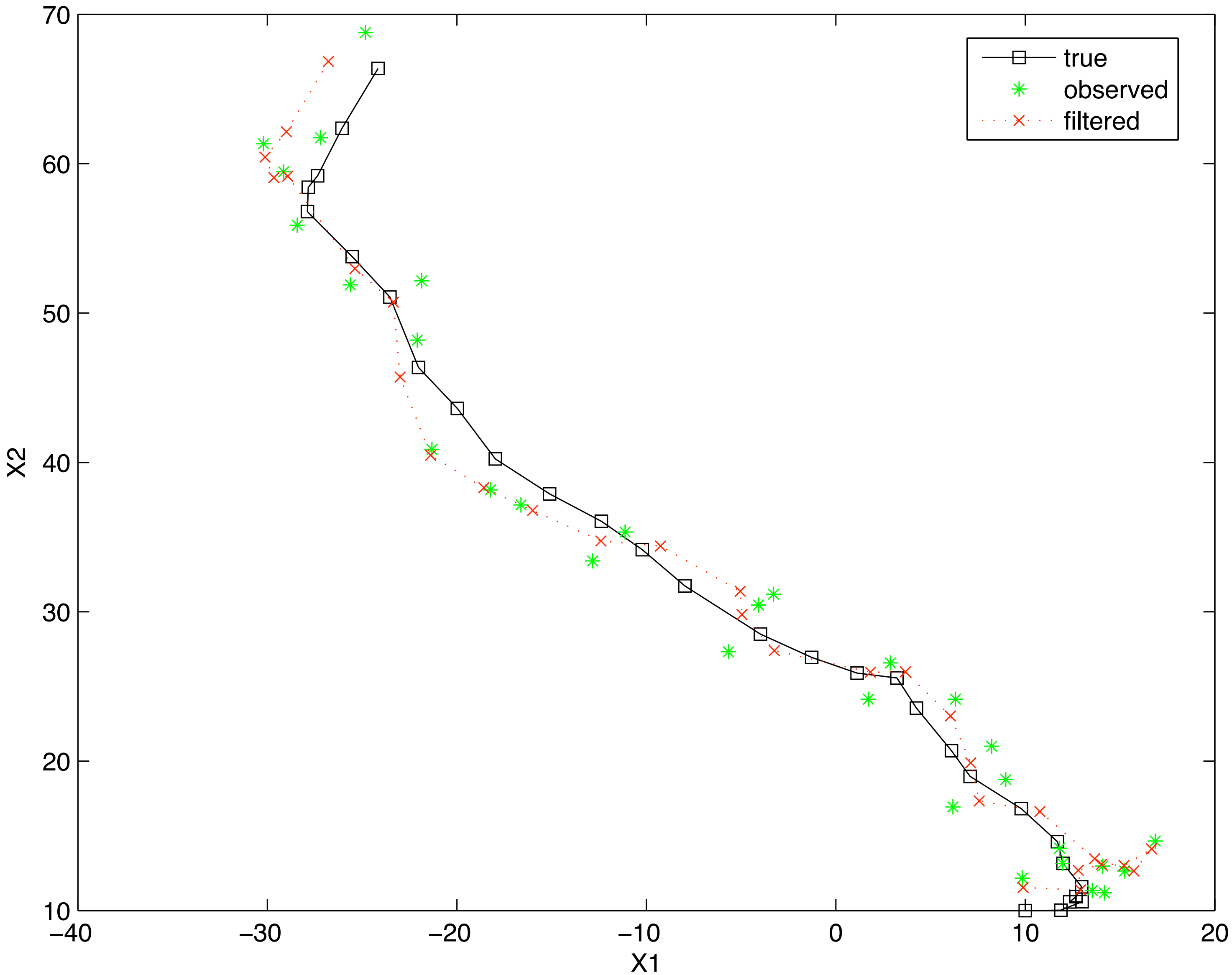
Implementation

```
% Make a point move in the 2D plane
% State = (x y xdot ydot). We only observe (x y).

% This code was used to generate Figure 17.9 of
% "Artificial Intelligence: a Modern Approach",
% Russell and Norvig, 2nd edition, Prentice Hall, in preparation

% X(t+1) = F X(t) + noise(Q)
% Y(t) = H X(t) + noise(R)

ss = 4; % state size
os = 2; % observation size
F = [1 0 1 0; 0 1 0 1; 0 0 1 0; 0 0 0 1];
H = [1 0 0 0; 0 1 0 0];
Q = 1*eye(ss);
R = 10*eye(os);
initx = [10 10 1 0]';
initV = 10*eye(ss);
```

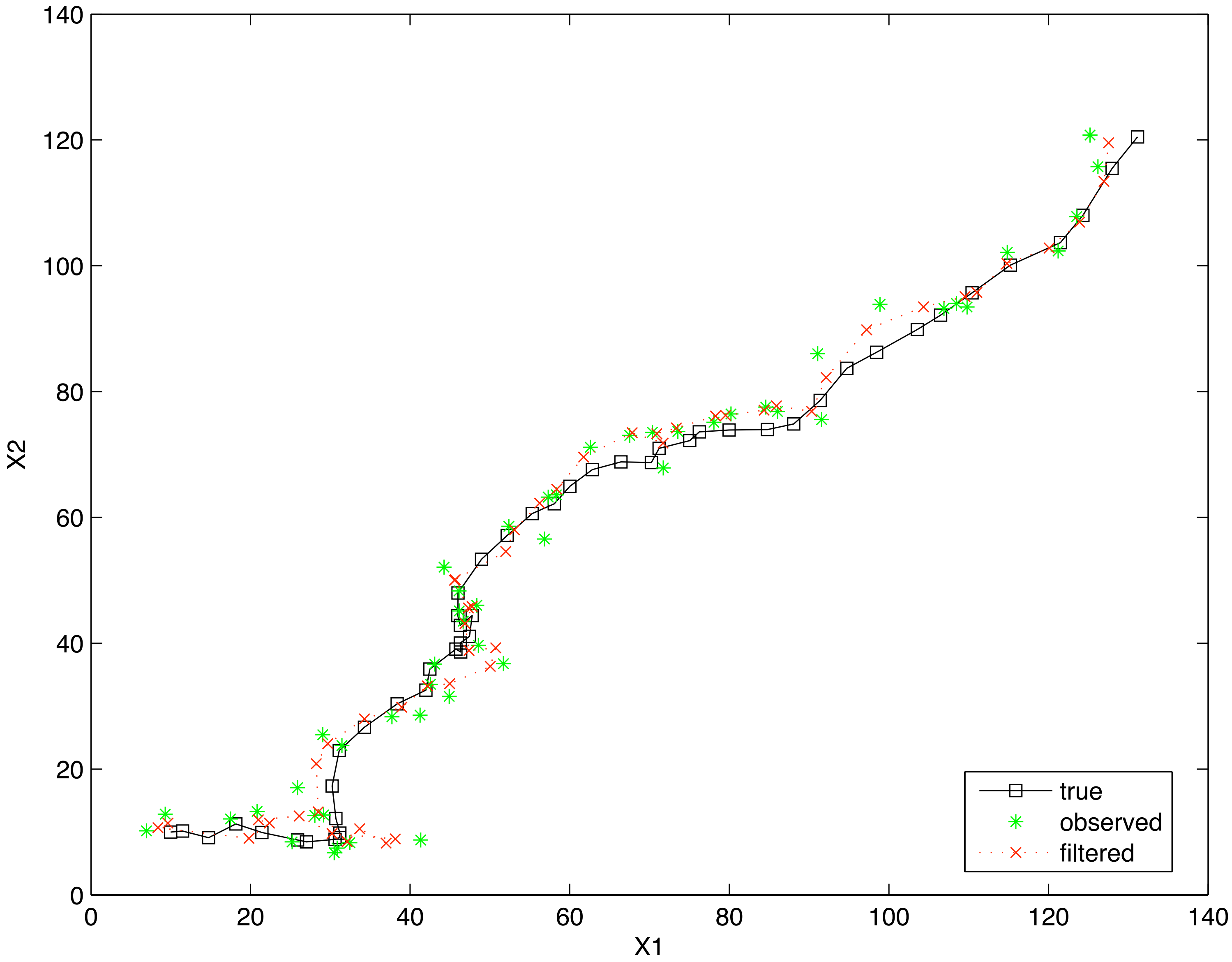
```
seed = 8; rand('state', seed);
randn('state', seed);
T = 50;
[x,y] = sample_lds(F,H,Q,R,initx,T);
```

Apply Kalman Filter

```
[xfilt,Vfilt] = kalman_filter(y,F,H,Q,R,initx,initV);
dfilt = x([1 2],:) - xfilt([1 2],:);
mse_filt = sqrt(sum(sum(dfilt.^2)))

figure;
plot(x(1,:), x(2,:), 'ks-');
hold on
plot(y(1,:), y(2,:), 'g*');
plot(xfilt(1,:), xfilt(2,:), 'rx:');
hold off
legend('true', 'observed', 'filtered', 0)
xlabel('X1'), ylabel('X2')
```



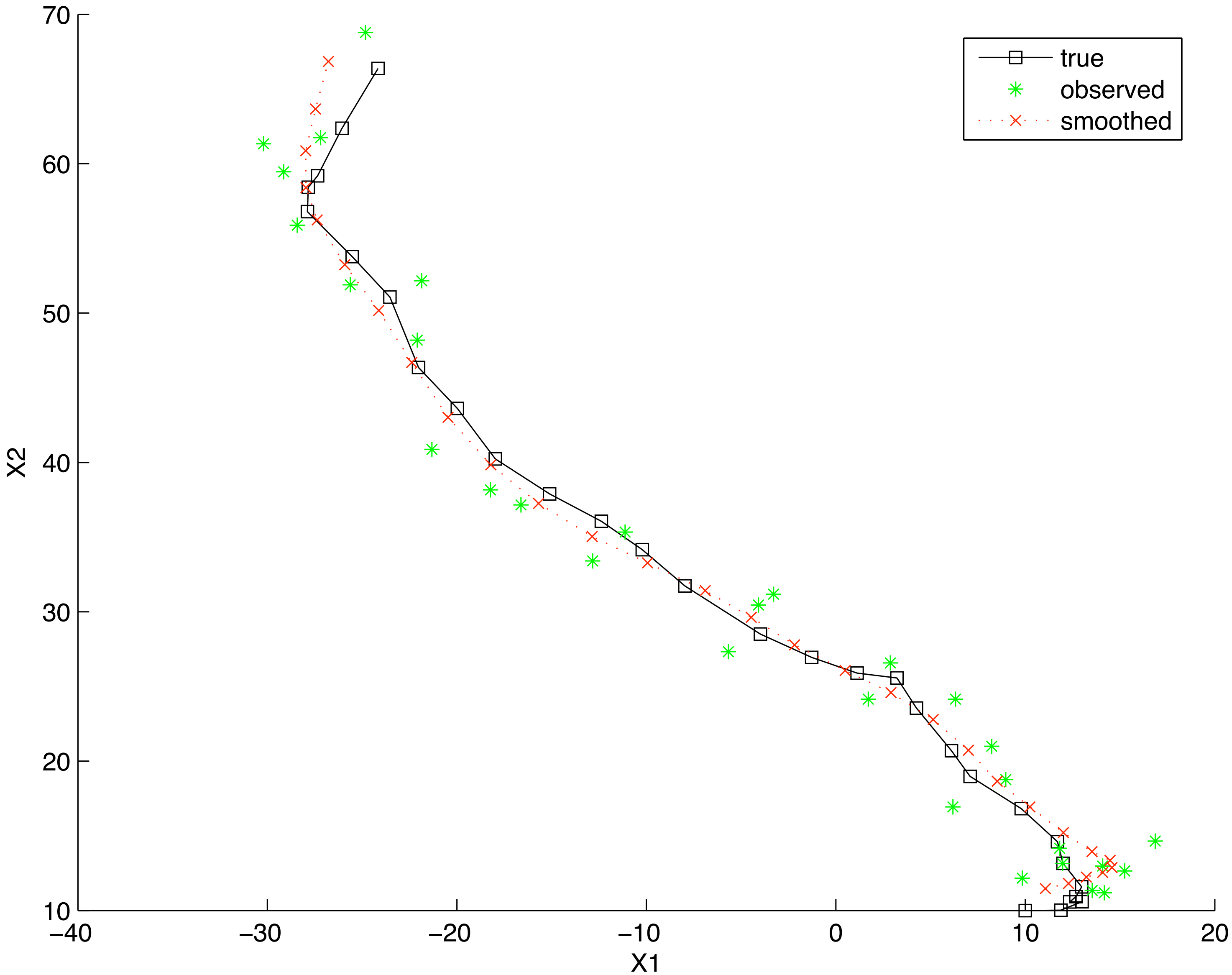
true
observed
filtered
X1
X2
−40
−30
−20
−10
0
10
20
70
60
50
40
30
20
10



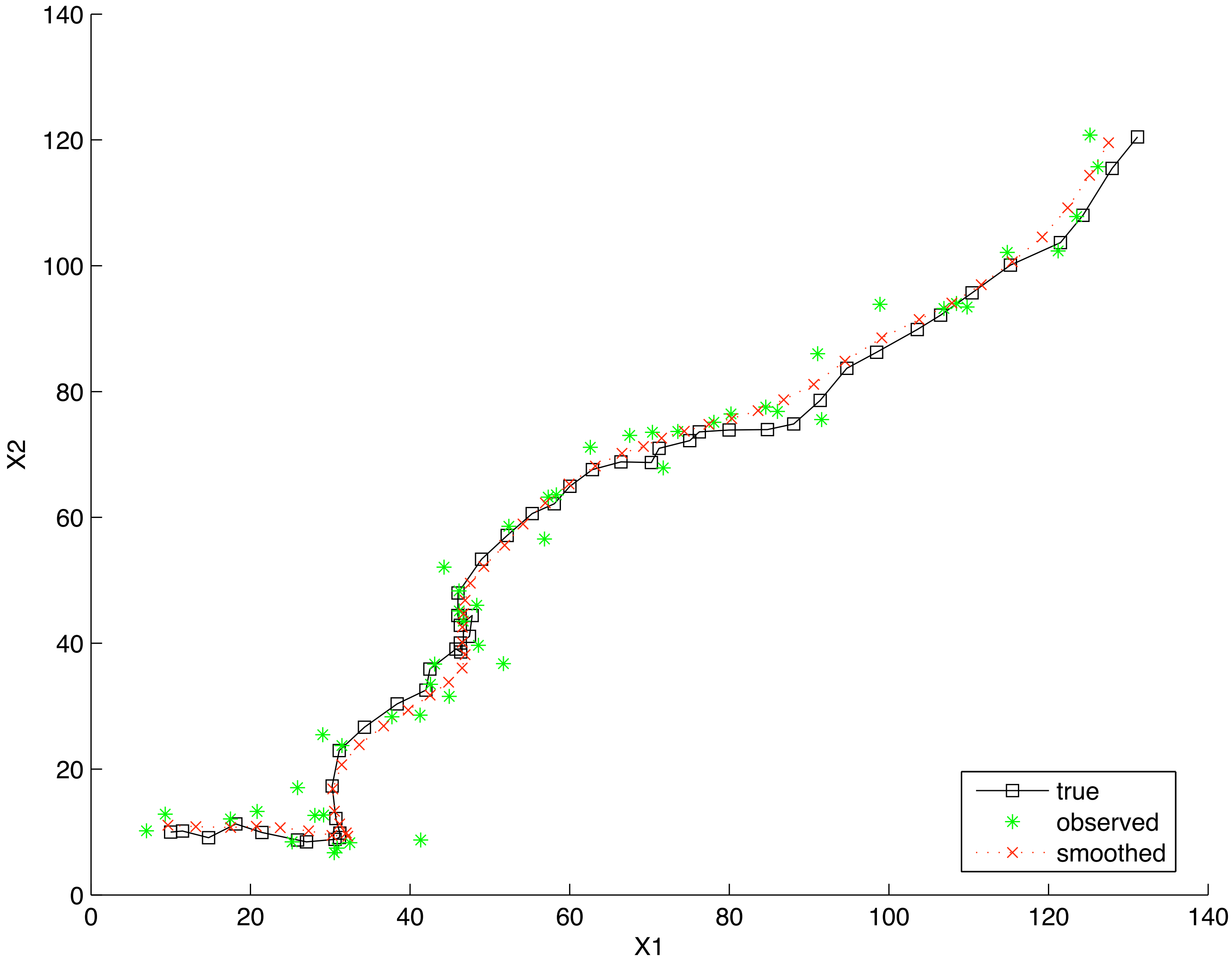
140
120
100
80
60
40
20
0
0
20
40
60
80
100
120
140
X1
X2
true
observed
filtered

Apply Kalman Smoother

```
[xsmooth, Vsmooth] = kalman_smoother(y,F,H,Q,R,initx,initV);
dsmooth = x([1 2],:) - xsmooth([1 2],:);
mse_smooth = sqrt(sum(sum(dsmooth.^2)))

figure;
hold on
plot(x(1,:), x(2,:), 'ks-');
plot(y(1,:), y(2,:), 'g*');
plot(xsmooth(1,:), xsmooth(2,:), 'rx:');
hold off
legend('true', 'observed', 'smoothed', 0)
xlabel('X1'), ylabel('X2')
```



true
observed
smoothed
X2
X1
70
60
50
40
30
20
10
−40
−30
−20
−10
0
10
20



140
120
100
80
60
40
20
0
X2
0
20
40
60
80
100
120
140
X1
true
observed
smoothed