ECE432: Homework 2
Kalman Filtering

Sven Olsen

Compiled at 2:39 PM on June 21, 2005

1 Equations

To restate the contents of the notes in a way that more closely reflects my own implementation:

Consider the case of modeling a single dimensional first order PDE:

$$x(t + \Delta t) = x(t) + \dot{x}\Delta t$$
$$\dot{x}(t + \Delta t) = \dot{x}(t)$$

We assume that there will be some noise in the actual process, as well as some noise in our ability to measure the state process. More exactly, we introduce $\mathbf{w_i}$ and v_i for each time step, which represent the process noise and measurement noise respectively, and which are both assumed to be drawn from a normal distribution. Let $\mathbf{Q}$ be the covariance matrix for $\mathbf{w}$, and $\mathbf{R}$ the covariance matrix for v. Let z_i be the measured value of x_i. Now we have an updated set of equations:

$$x_{i+1} = x_i + \dot{x}_i\Delta t + [\mathbf{w_i}]_1$$
$$\dot{x}_{i+1} = \dot{x}(t) + [\mathbf{w_i}]_2$$
$$z_i = x_i + v_i$$

We express these relations in matrix form like so:

$$\mathbf{S} = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}, \qquad \mathbf{H} = \begin{bmatrix} 1 & \Delta t \\ 0 & 1 \end{bmatrix}, \qquad \mathbf{F} = \begin{bmatrix} 1 & 0 \end{bmatrix}$$
$$\mathbf{S}_{i+1} = \mathbf{H}_i\mathbf{S}_i + \mathbf{w_i}$$
$$z_i = \mathbf{F}_i\mathbf{S}_i + v_i$$

More complex systems can be represented by the same matrix equations given different matrix definitions. For example, for 2 dimensional linear motion:

$$\mathbf{S}_i = \begin{bmatrix} x_i \\ y_i \\ \dot{x}_i \\ \dot{y}_i \end{bmatrix}, \qquad \mathbf{H} = \begin{bmatrix} 1 & 0 & \Delta t & 0 \\ 0 & 1 & 0 & \Delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \qquad \mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix}$$

The algorithm proceeds as follows:

We start with an initial guess of the position and velocity, as well as the expected error of our current guess, represented as a covariance matrix $\mathbf{P}$.

First we calculate a predicted state for the next iteration using the predicted state at the last iteration:

$$\hat{\mathbf{S}}_{i|i-1} = \mathbf{H}\hat{\mathbf{S}}_{i-1}$$

Then we predict the state covariance of the current iteration

$$\hat{\mathbf{P}}_{i|i-1} = \mathbf{H}\mathbf{P}_{i-1}\mathbf{H}^T + \mathbf{Q}$$

Now we take steps to correct our predictions by comparing them to the known measurements.

We start by taking note of the measurement residual and the covariance residual:

$$\tilde{\mathbf{y}}_i = \mathbf{z_i} - \mathbf{F}\hat{\mathbf{S}}_{i|i-1}$$
$$\tilde{\mathbf{U}}_i = \mathbf{F}_i\mathbf{P}_{i|i-1}\mathbf{F}_i^T + \mathbf{R}$$

We then define this curious creature called the "Kalman Gain"

$$\mathbf{K}_i = \mathbf{P}_{i|i-1}\mathbf{F}_i^T\tilde{\mathbf{U}}_i^{-1}$$

Using the Kalman Gain, we correct our predictions for the state and state covariance:

$$\hat{\mathbf{S}}_i = \hat{\mathbf{S}}_{i|i-1} + \mathbf{K}_i\tilde{\mathbf{y}}_i$$
$$\hat{\mathbf{P}}_i = (\mathbf{I} - \mathbf{K}_i\mathbf{F}_i)\mathbf{P}_{i|i-1}$$

2 Data

I've tried a couple different filter configurations. All of our tests start with the correct ground truth state. Test case 1 uses a very low initial expected error, a high measurement error, and no process error (Fig. 1). Test case 2 uses moderate initial expected error and measurement error, and no process error (Fig. 2). Test case 3 uses moderate initial expected error, measurement error, and process error (Fig. 3).

Test case 1 is the most "correct" configuration (as we are starting with ground truth, updating while ignoring the observation data completely would produce perfect results). However, it also produces poor results. I imagine that this is because $\mathbf{P}$ decreases throughout the life of the filter, and thus after the large initial errors have pushed the state vector a little off course, $\mathbf{P}$ has become so small that $\mathbf{K}$ can exert very little correcting influence on the state vector.

Test case 2, with the flexibility implied by its larger $\mathbf{P}$, performs much better.

Test case 3 introduces the expectation of processes error into the model ($\mathbf{Q}$ is nonzero). This has the result of making the filtered results much more likely to follow the measured behavior, rather than converging to a line. And this makes sense, because with this configuration, the algorithm believes that the true state behavior is likely to deviate from the linear motion model. Thus, insomuch as the observed data is not linear, the algorithm imagines that part of that nonlinearity is due to the fact that the actual object motion is nonlinear.

2.1 Disclaimer

As the ground truth data given was for 2D linear motion, I have only used a first order motion model in these experiments (for a study of the second order results to be interesting – I would need to generate my own noisy second order motion data, which I haven't gotten around to doing).

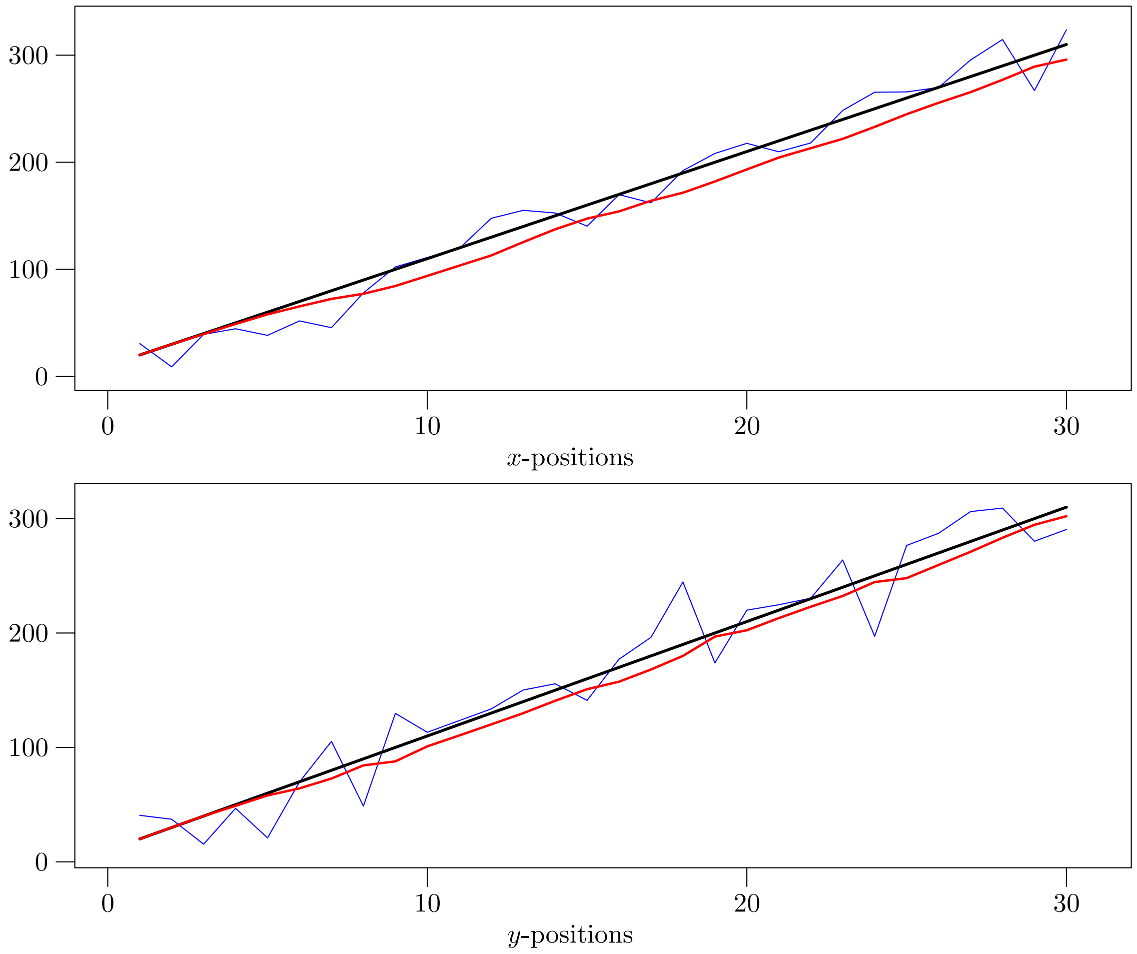

Figure 1: Test case 1: $\mathbf{P}_0 = \mathbf{diag}(0.1)$, $\mathbf{R} = \mathbf{diag}(30)$, $\mathbf{Q} = \mathbf{0}$. Observations are in blue, ground truth in black, Kalman filter results in red.

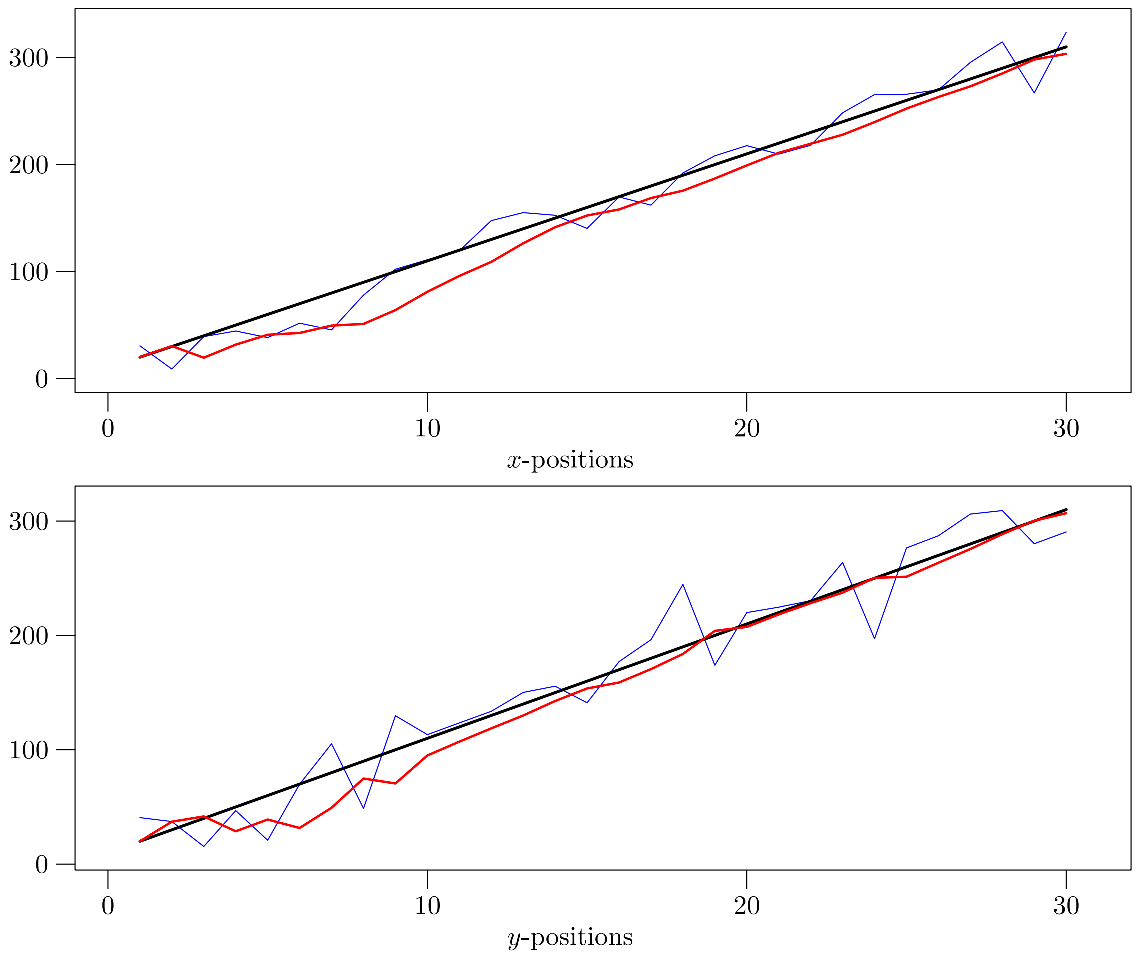

Figure 2: Test case 2: $\mathbf{P}_0 = \mathbf{diag}(10)$, $\mathbf{R} = \mathbf{diag}(10)$, $\mathbf{Q} = \mathbf{0}$. Observations are in blue, ground truth in black, Kalman filter results in red.

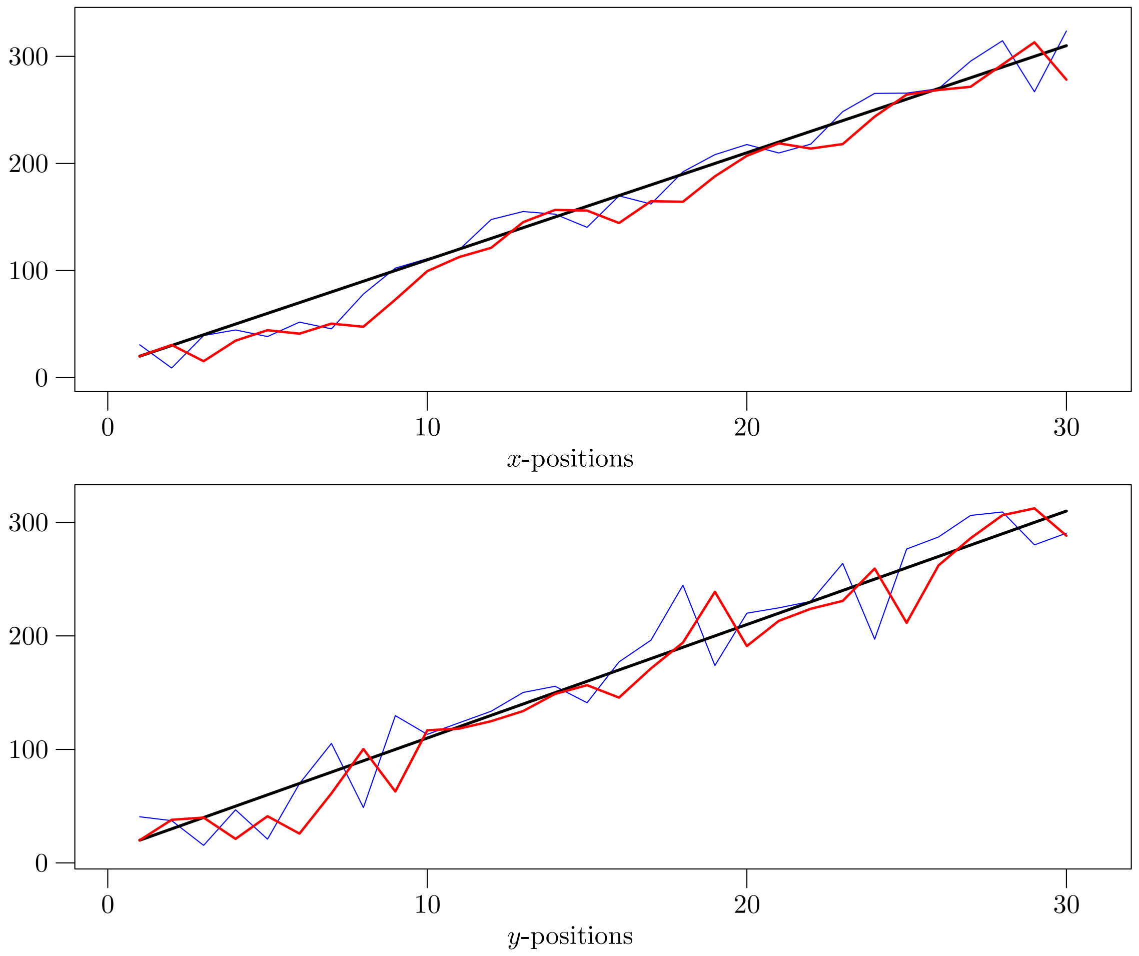

Figure 3: Test case 3: $\mathbf{P}_0 = \mathbf{diag}(10)$, $\mathbf{R} = \mathbf{diag}(10)$, $\mathbf{Q} = \mathbf{diag}(10)$. Observations are in blue, ground truth in black, Kalman filter results in red.