# CS 242 HW 3

Samuel Ainsworth

Worked with Jake Soloff and Michael Scheer

November 16, 2014

# Problem 1

## Part (a)

See Figure 1. We run the Kalman filter on the given "track" data.

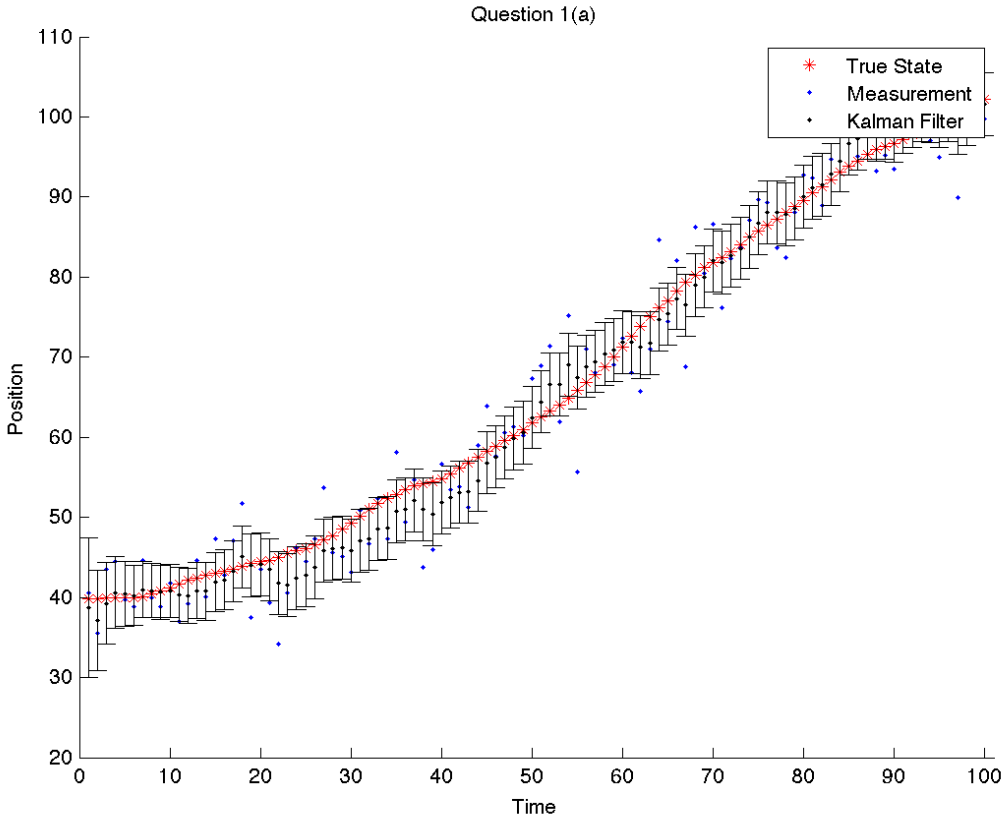


Figure 1: The true position in red, the noisy measurements in blue, and Kalman filter posterior distributions in black.

# Part (b)

See Figure 2. Instead of using the correct model parameters, we assume that the data was generated from a constant-position model. We estimate that $\sigma_y^2 = 20$ and consider both $\sigma_x^2 = 0.01/3$ and $\sigma_x^2 = 10$. For each of these two parameter choices, we run the Kalman filter on the "track" data.

As we can see in Figure 2, selecting $\sigma_x^2$ too small results in completely inaccurate estimates. As the plot on the left shows, the Kalman filtering estimates fail capture the variation in position since such variations are consider very unlikely when $\sigma_x^2 = 0.01/3$. When $\sigma_x^2 = 10$, we can make up for our incorrect constant-position assumption by accepting much more variation in the noise. Although the Kalman filtering posterior distributions roughly match the "track" data, they do so with much more variance than the correct filtering results from part (a).

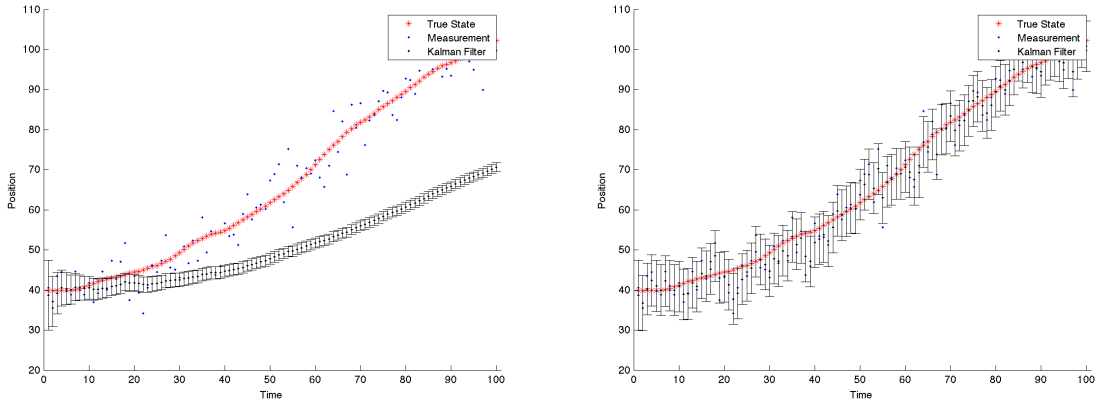


Figure 2: Kalman filtering with $\sigma_x^2 = 0.01/3$ on the left and with $\sigma_x^2 = 10$ on the right.

# Part (c)

See attached code.

# Part (d)

See Figure 3. We can see that the particle filtering estimates are just as accurate if not more accurate than the Kalman filtering estimates.

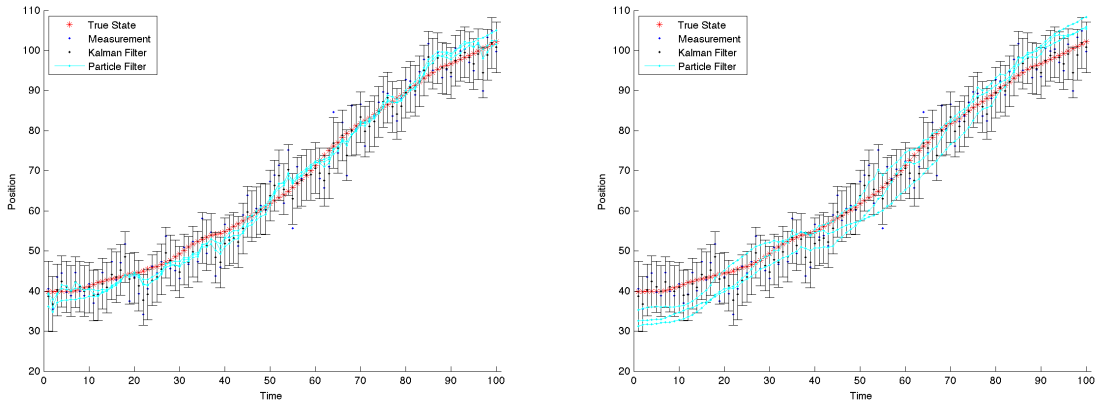


Figure 3: (left) Kalman filter estimates in black along with 3 particle filtering estimates in magenta. (right) Three particle filtering estimates with just 20 particles.

# Part (e)

See Figure 3. Although using only 20 particles results in more variance in the posterior estimates, the particle filtering estimates are surprisingly robust to the number of particles in this case. However, this is at least in part due to the fact that all of our posterior distributions are 1-dimensional and unimodal.

# Part (f)

See Figure 4. With the corrupted "track" data, the strengths of particle filtering become clear. In the presence of noise, the emission distribution is no longer Gaussian and as a result, the Kalman filter is easily thrown off track. However, the particle filter is able to model this non-Gaussian distribution with no trouble at all and therefore performs much, much better than the Kalman filter model.

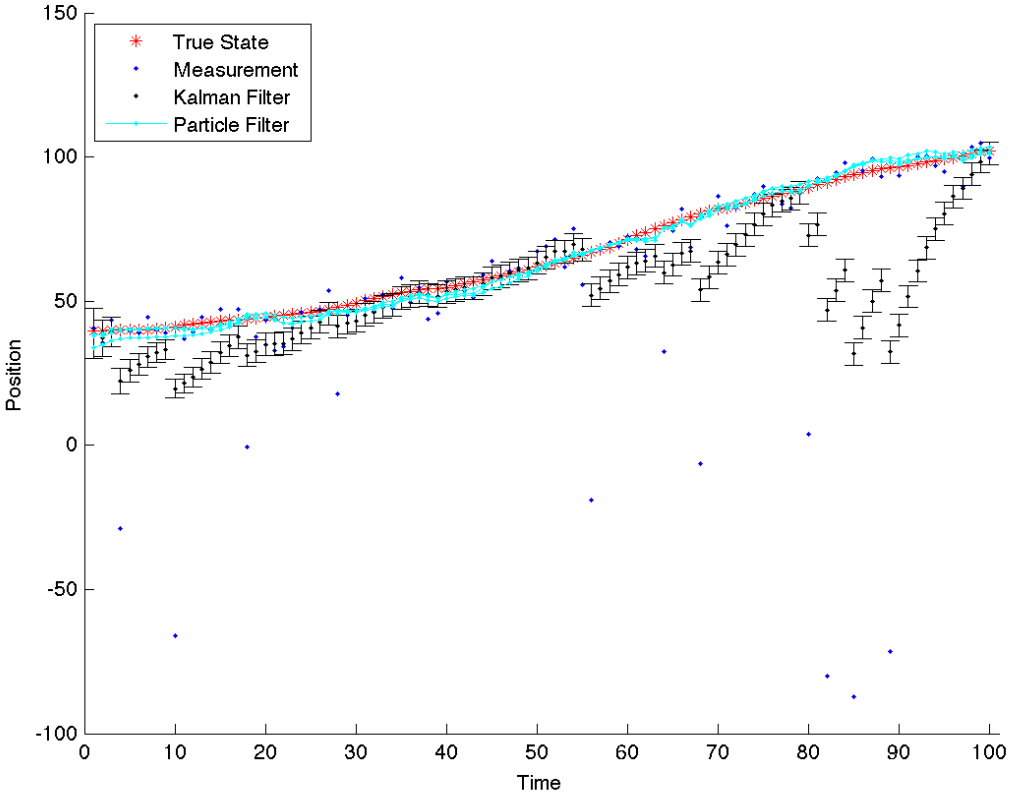


Figure 4: Kalman and particle filtering position estimates for the noisy data.

# Problem 2

## Part (a)

See Figure 5. The Kalman smoother estimates are clearly much more accurate and much smoother than the corresponding Kalman filtering estimates. This makes sense since the smoothing estimates are conditioned on more observations, which should clearly result in more accurate posterior distributions. Partly evident of this, we see that the smoothed posterior distributions have smaller variance than the same filtered posteriors and have no more bias than the filtered ones.

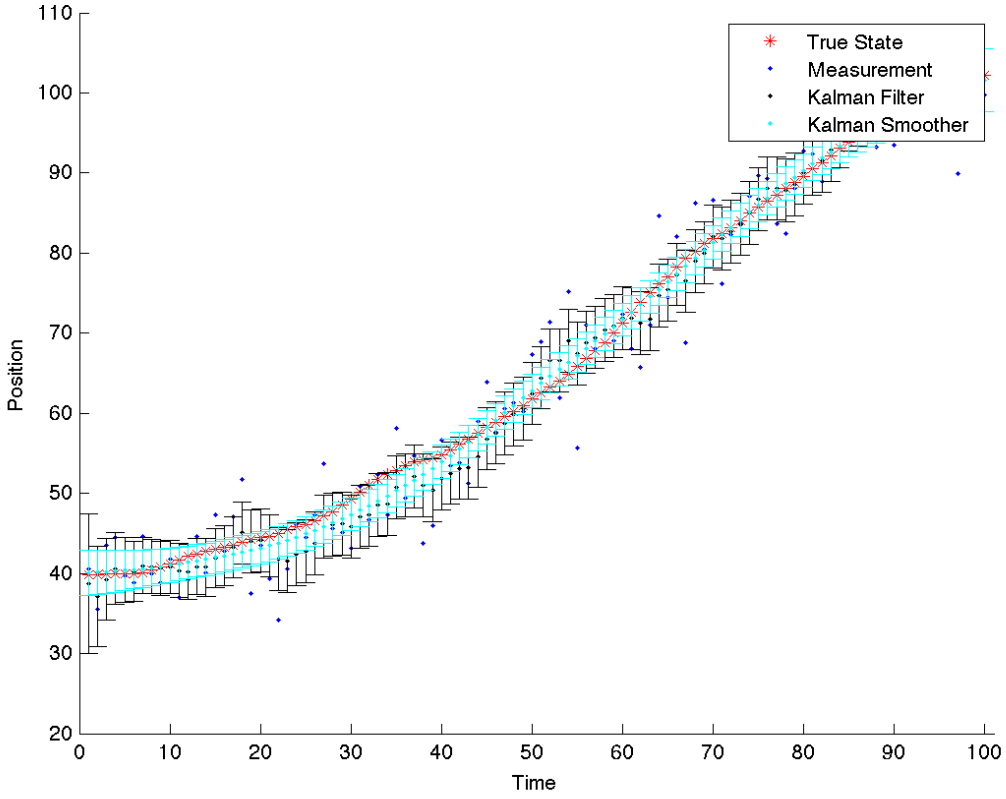


Figure 5: Kalman filtering estimates shown in black and Kalman smoothing estimates shown in magenta.

# Part (b)

See attached code for implementation.

$$
\begin{aligned}
\log p(y) &= \log p(y_1) + \sum_{t=1}^{T-1} \log p(y_{t+1}|y_1, \ldots, y_t) \\
&= \log \int_{\mathcal{X}_1} p(y_1|x_1)p(x_1)dx_1 + \sum_{t=1}^{T-1} \log \int_{\mathcal{X}_t} p(y_{t+1}|x_t)p(x_t|y_1, \ldots, y_t)dx_t \\
&= \log \int_{\mathcal{X}_1} \mathcal{N}\left(y_1|Cx_1, R\right)\mathcal{N}\left(x_1|0, I_d\right)dx_1 \\
&+ \sum_{t=1}^{T-1} \log \int_{\mathcal{X}_t} \left( \int_{\mathcal{X}_{t+1}} p(y_{t+1}|x_{t+1})p(x_{t+1}|x_t)dx_{t+1} \right) p(x_t|y_1, \ldots, y_t)dx_t \\
&= \log \mathcal{N}\left(y_1|Cx_1, CC^T + R\right) + \sum_{t=1}^{T-1} \log \int_{\mathcal{X}_t} \mathcal{N}\left(y_{t+1}|CAx_t, CQC^T + R\right)\mathcal{N}\left(x_t|\mu_{t|t}, \Sigma_{t|t}\right)dx_t \\
&= \log \mathcal{N}\left(y_1|Cx_1, CC^T + R\right) + \sum_{t=1}^{T-1} \log \mathcal{N}\left(y_{t+1}|CAx_t, CQC^T + R + (CA)\,\Sigma_{t|t}\,(CA)^T\right)
\end{aligned}
$$

# Part (c)

The M-step update for $R$ is

$$
R^{new} = \frac{1}{T}\sum_{t=1}^{T}\left(y_t y_t^T - y_t \mathbb{E}\left[x_t|y\right]^T \left(C^{new}\right)^T\right)
$$

This form can be derived by matching moments with the data.

# Part (d)

See Figure 6. The resulting model parameters are

$$
\begin{aligned}
A &= \begin{bmatrix} 1.0071 & -0.1593 \\ 0.1331 & 0.9544 \end{bmatrix} \\
Q &= \begin{bmatrix} 5.8014 & -0.7499 \\ -0.7499 & 6.1219 \end{bmatrix} \\
C &= \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix} \\
R &= \begin{bmatrix} 18.5733 & -1.5869 \\ -1.5869 & 20.9094 \end{bmatrix}
\end{aligned}
$$

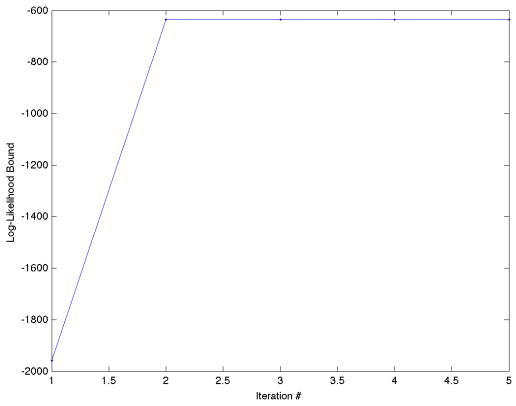


Figure 6: (left) The log-likelihood per EM iteration of the state space model on the training data. (right) The Kalman filtering and smoothing with the model parameters estimated via EM.

# Problem 3

## Part (a)

Our general model is a Bayes net, so marginalizing out irrelevant variables we have

$$p(z) = p(z_1) \prod_{t=2}^{T} p(z_t|z_{t-1})$$

Then we wish to find

$$\begin{aligned} \pi &= \arg\max_{\tilde{\pi}} \log p(z_1) + \sum_{t=2}^{T} \log p(z_t|z_{t-1}) \\ &\overset{(a)}{=} \arg\max_{\tilde{\pi}} \sum_{t=2}^{T} \log \tilde{\pi}_{t-1,t} \\ &\overset{(b)}{=} \begin{pmatrix} \arg\max_{\tilde{\pi}_1} \sum_{t\geq 2: z_{t-1}=1} \log \tilde{\pi}_{1,t} \\ \arg\max_{\tilde{\pi}_2} \sum_{t\geq 2: z_{t-1}=2} \log \tilde{\pi}_{2,t} \\ \arg\max_{\tilde{\pi}_3} \sum_{t\geq 2: z_{t-1}=3} \log \tilde{\pi}_{3,t} \end{pmatrix} \end{aligned}$$

where (a) follows from the fact that we assume $p(z_1)$ is uniform and (b) follows from the fact that each row of $\pi$ is exactly the pmf of a categorical distribution: $p(z_t|z_{t-1} = i) \sim \text{Cat}(\pi_i)$. Therefore, we can optimize of each $\pi_i$ individually using the standard ML estimation for categorical distributions:

$$\pi_{i,j} = \frac{\sum_{t=2}^{T} \mathbb{I}\{z_t = j, z_{t-1} = i\}}{\sum_{t=1}^{T} \mathbb{I}\{z_t = i\}}$$

This gives the following transition matrix for each training split:

$$\pi = \begin{bmatrix} 0.9735 & 0 & 0.0265 \\ 0 & 0.9794 & 0.0206 \\ 0.0207 & 0.0207 & 0.9586 \end{bmatrix}, \begin{bmatrix} 0.9793 & 0 & 0.0207 \\ 0 & 0.9798 & 0.0202 \\ 0.0143 & 0.0127 & 0.9731 \end{bmatrix}, \begin{bmatrix} 0.9749 & 0 & 0.0251 \\ 0 & 0.9777 & 0.0223 \\ 0.0187 & 0.0172 & 0.9641 \end{bmatrix}$$

## Part (b)

See attached code.

## Part (c)

See attached code.

## Part (d)

See Figure 7. The resulting model parameters for the first train-test split are

$$A_1 = \begin{bmatrix} 1.0093 & 0.0243 & 0.0571 & 0.0588 \\ -0.0325 & 0.9733 & 0.0656 & -0.0384 \\ -0.0674 & 0.0358 & 0.8880 & -0.1381 \\ -0.0067 & 0.0041 & 0.0791 & 0.9833 \end{bmatrix}$$

$$Q_1 = \begin{bmatrix} 0.0085 & 0.0007 & -0.0012 & -0.0002 \\ 0.0007 & 0.0101 & 0.0020 & 0.0008 \\ -0.0012 & 0.0020 & 0.0637 & 0.0038 \\ -0.0002 & 0.0008 & 0.0038 & 0.0047 \end{bmatrix}$$

$$C_1 = \begin{bmatrix} 1 & & & \\ & 1 & & \\ & & 1 & \\ & & & 1 \end{bmatrix}$$

$$R_1 = \begin{bmatrix} 0.0003 & -0.0003 & -0.0025 & 0.0003 \\ -0.0003 & 0.0011 & 0.0078 & -0.0013 \\ -0.0025 & 0.0078 & 0.0712 & -0.0131 \\ 0.0003 & -0.0013 & -0.0131 & 0.0063 \end{bmatrix}$$

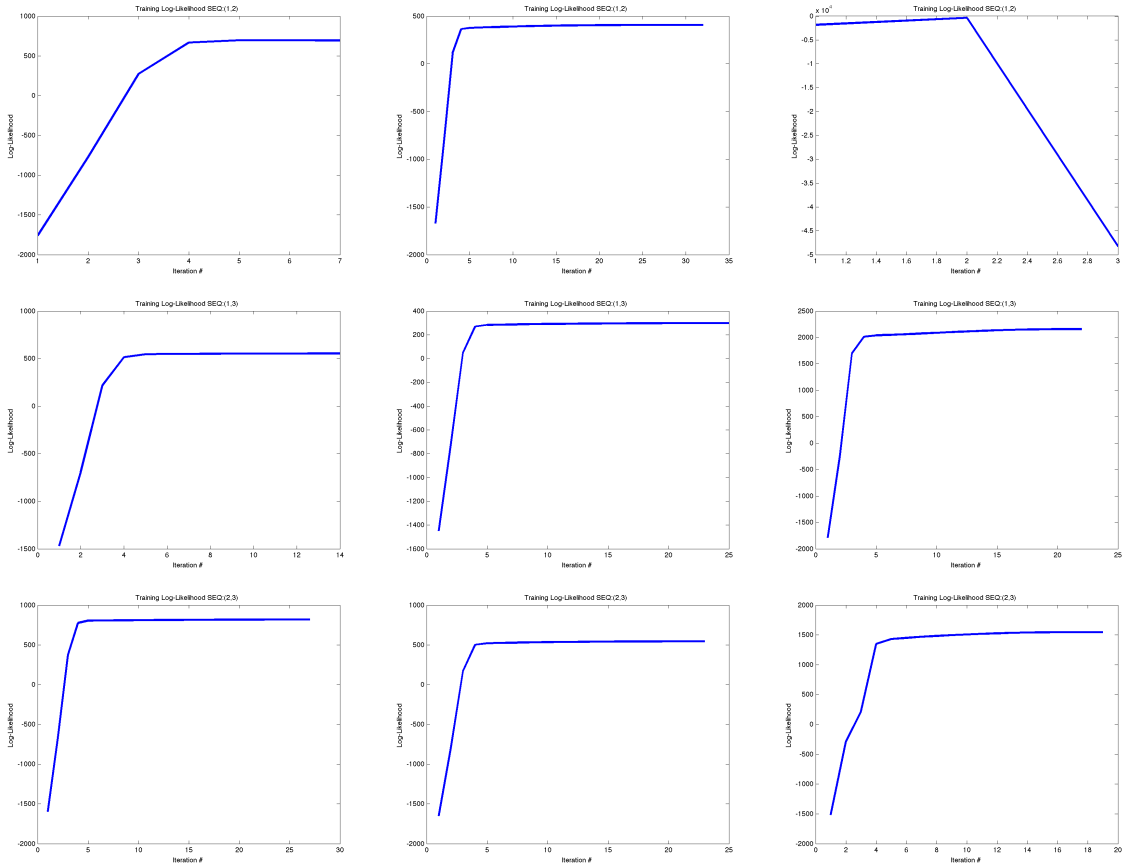


Figure 7: Log-likelihood per iteration for each train-test split and each $z = 1, 2, 3$. Every row corresponds to one train-test split and every column correspond to $z = 1, 2, 3$ in order.

# Part (e)

$$
\begin{aligned}
p(\tilde{y}_t|y_t, z_t = k) &= \int_{\mathcal{Y}} p(\tilde{y}_t|y_t, z_t) p(y_t|x_t, z_t) dy \\
&= \int_{\mathcal{Y}} \left(0.9\mathcal{N}\left(\tilde{y}_t|y_t, 0.1I_d\right) + 0.1\mathcal{N}\left(\tilde{y}_t|0, 5I_d\right)\right) \mathcal{N}\left(y_t|C_k x_t, R_k\right) dy_t \\
&= 0.9 \int_{\mathcal{Y}} \mathcal{N}\left(\tilde{y}_t|y_t, 0.1I_d\right) \mathcal{N}\left(y_t|C_k x_k, R\right) dy_t + 0.1 \int_{\mathcal{Y}} \mathcal{N}\left(\tilde{y}_t|0, 5I_d\right) \mathcal{N}\left(y_t|C_k x, R_k\right) dy_t \\
&= 0.9\mathcal{N}\left(\tilde{y}_t|C_k x_t, R_k + 0.1I_d\right) + 0.1\mathcal{N}\left(\tilde{y}_t|0, 5I_d\right) \int_{\mathcal{Y}} \mathcal{N}\left(y_t|C_k x_t, R_k\right) dy \\
&= 0.9\mathcal{N}\left(\tilde{y}|C_k x_t, R_k + 0.1I_d\right) + 0.1\mathcal{N}\left(\tilde{y}_t|0, 5I_d\right)
\end{aligned}
$$

# Part (f)

See attached code.

## Part (g)

See Figure 8. Our particle filtering approach performs very well even with only 1000 particles. Although the estimated state distributions have some small errors, the estimated angle and position look very accurate.

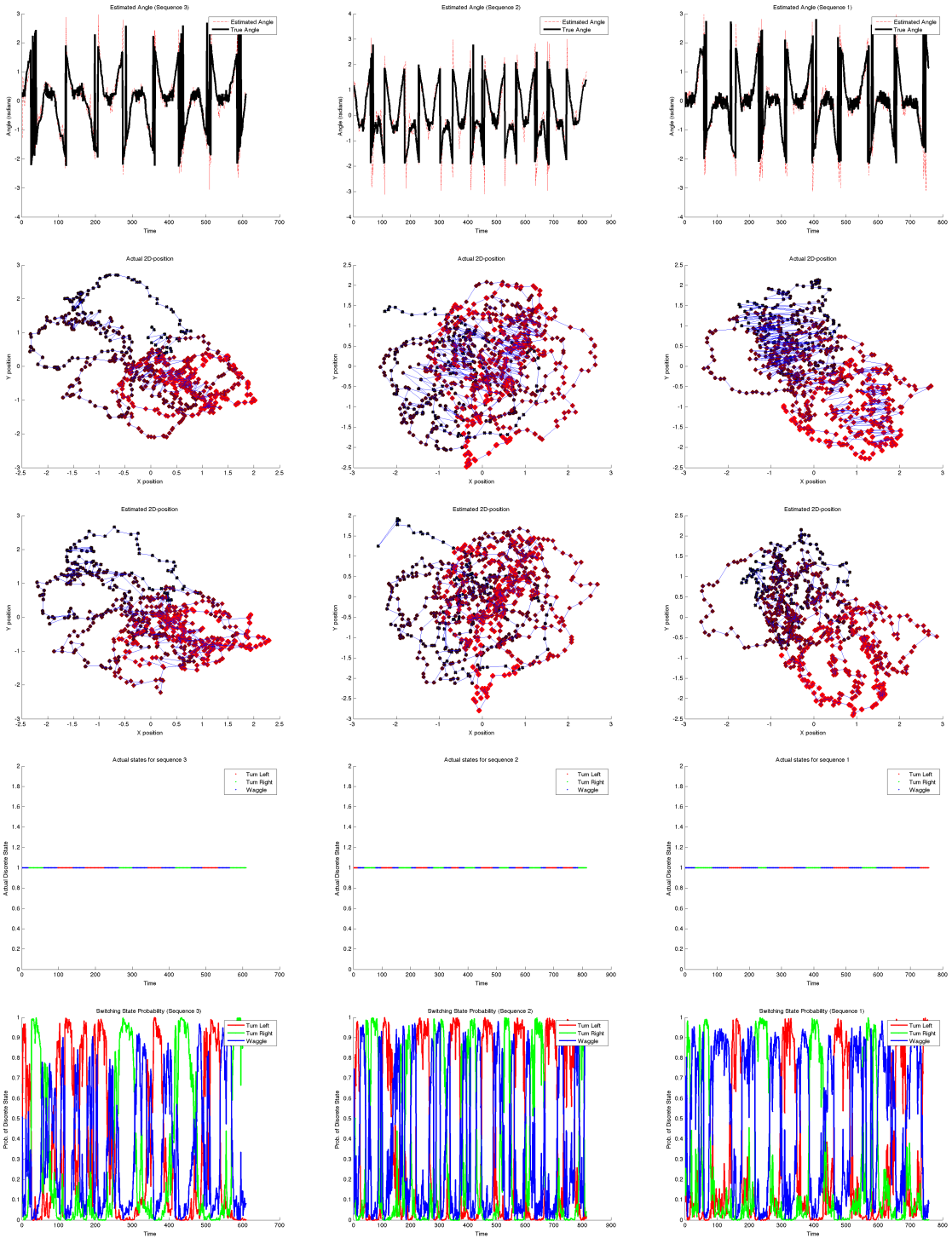


Figure 8: Visualizations of the predictions made in 3(g). Each column corresponds to a particular train/test split where the left column shows testing on the 3rd sequence, the middle column shows testing on the 2nd sequence and the right shows testing on the 1st sequence. Each row corresponds to different visualizations of our predictions on the held our sequence. In particular, the first row shows the true angle of the bee vs the estimated angle. The second row shows the actual 2D position of the bee. The third row shows our estimated 2D position of the bee. The fourth row shows the actual states of the bee in the testing sequence over time. Finally, the fifth row shows the estimated probabilities of each state over time in the testing sequence.