Comp 141 Probabilistic Robotics Homework 1: Kalman Filter

Travis Clauson

14 February 2023

1 Kalman Filter: Prediction

A balloon drone has encountered a glitch in its program and needs to reboot its on-board computer. While rebooting, the drone is helpless and cannot issue motor commands. To help the drone, you'll need some understanding of the Kalman filter algorithm.

The drone operates in a 1-D world where x_t is the position at time t, while $\dot{x}_t$ and $\ddot{x}_t$ are the velocity and acceleration. For simplicity, assume that $\triangle t = 1$.

Due to random wind fluctuations, at each new time step, your acceleration is set randomly accordingly to the distribution $\mathcal{N}(\mu_{wind}, \sigma^2_{wind})$, where $\mu_{wind} = 0.0$ and $\sigma^2_{wind} = 1.0$.

Question 1.1: What is the minimal state vector for the Kalman filter so that the resulting system is Markovian?

Solution:

$$\begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix}$$

The state of the drone can be completely described through two variables, position and velocity. Therefore, the minimal Markovian state vector is:

Question 1.2: Design the state transition probability function $p(x_t|u_t, x_{t-1})$. The transition function should contain linear matrices A and B and a noise covariance R.

Solution:

$$p(x_t|u_t, x_{t-1}) = \bar{x_t} = Ax_{t-1} + Bu_t$$
$$\text{Covariance: } \bar{\Sigma_t} = A_t\Sigma_{t-1}A_t^T + R_t$$

where,

$$A_t = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} B_t = \begin{bmatrix} 1/2 \\ 1 \end{bmatrix} \qquad G = \begin{bmatrix} 0.5 \\ 1 \end{bmatrix}, R_t = \sigma^2 G G^T = \begin{bmatrix} .25 & .5 \\ .5 & 1 \end{bmatrix}$$

A comes from the Equations of Motion which describe the problem. It explains how the past positions and velocities impact the next position and velocity. Matrix B comes from problem 3.1 ahead where it states that the control (propeller on the drone) increases velocity proportionally.

$$x_t = x_{t-1} + \dot{x}_{t-1} * \Delta t + 0.5 * \ddot{x}_{t-1} * \Delta t^2$$
$$\dot{x}_t = \dot{x}_{t-1} + \ddot{x}_{t-1} * \Delta t$$

Question 1.3: Implement the state prediction step of the Kalman filter, assuming that at time $t = 0$, we start at rest, i.e., $x_t = \dot{x}_t = \ddot{x}_t = 0.0$. Use your code to calculate the state distribution for times $t = 1, 2, \ldots, 5$.

Solution:

$$\mu_1 = \mu_2 = \mu_3 = \mu_4 = \mu_5 = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$\Sigma_1 = \begin{bmatrix} 0.25 & 0.5 \\ 0.5 & 1 \end{bmatrix}$$

$$\Sigma_2 = \begin{bmatrix} 2.5 & 2 \\ 2 & 2 \end{bmatrix}$$

$$\Sigma_3 = \begin{bmatrix} 8.75 & 4.5 \\ 4.5 & 3 \end{bmatrix}$$

$$\Sigma_4 = \begin{bmatrix} 21 & 8 \\ 8 & 4 \end{bmatrix}$$

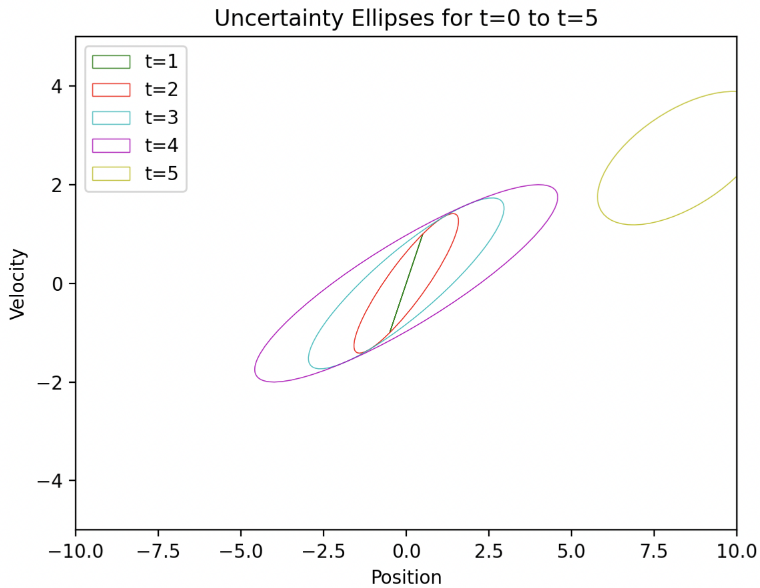
$$\Sigma_5 = \begin{bmatrix} 41.25 & 12.5 \\ 12.5 & 5 \end{bmatrix}$$

As time progresses without a sensor reading, the accuracy of the model decreases exponentially, while the mean remains 0 since the wind is normally distributed. Essentially, the bell curve that represents possible positions and velocities of the drone is stretching wider and wider.

Question 1.4: For each value of t in the previous question, plot the joint posterior over x and $\dot{x}$ in a diagram where x is the horizontal and $\dot{x}$ is the vertical axis. For each posterior, you are asked to plot the uncertainty ellipse which is the ellipse of points that are one standard deviation away from the mean.

Some additional information about uncertainty ellipses and how to calculate them using MATLAB or C++ can be found here: `http://www.visiondummy.com/2014/04/draw-error-ellipse-representing-covariance-matrix/`.

Solution:



The plot shows how uncertainty increases for position and velocity as time increases. This is because the model is not receiving any sensor readings, so the estimation model becomes less accurate (variance increases) as time goes on. The uncertainty ellipse at t=5 is not centered because of upcoming part **2.2** where a sensor reading of z=10 is read, which moves the estimate and reduces the variance.

2 Kalman Filter: Measurement

Prediction alone will result in greater and greater uncertainty as time goes on. Fortunately, your drone has a GPS sensor, which in expectation, measures the true position. However, the measurement is corrupted by Gaussian noise with covariance $\sigma^2_{gps} = 8.0$.

Question 2.1: Define the measurement model. You will need to define matrices C and Q.

Solution:

$$K_t = \bar{\Sigma}_t\, C_t^T (C_t\, \bar{\Sigma}_t\, C_t^T + Q_t)^{-1}$$
$$\mu_t = \bar{\mu}_t + K_t(z_t - C_t\, \bar{\mu}_t)$$
$$\Sigma_t = (I - K_t\, C_t)\, \bar{\Sigma}_t$$

$$z_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} x_t \\ \dot{x}_t \end{bmatrix} + \delta_t$$

where,

$$C_t = \begin{bmatrix} 1 & 0 \end{bmatrix} \qquad \delta_t \sim N(0, \sqrt{8}) \qquad Q = \begin{bmatrix} 8 \end{bmatrix}$$

The C matrix tells us that the sensor value explains position, and tells us nothing about velocity. The Q matrix is simply the covariance of the sensor.

Question 2.2: Implement the measurement update. Suppose at time $t = 5$, the drone's computer has rebooted and we query our sensor for the first time to obtain the measurement $z = 10$. State the parameters of the Gaussian estimate before and after incorporating the measurement. Afterwards, implement the sensor modal to randomly sample the true position, corrupted with noise σ_{gps}^2.

Solution:

The parameters of the Gaussian estimate before the measurement are:

$$\bar{\mu}_5 = \begin{bmatrix} 0 \\ 0 \end{bmatrix} \qquad \bar{\Sigma}_5 = \begin{bmatrix} 41.25 & 12.5 \\ 12.5 & 5 \end{bmatrix}$$

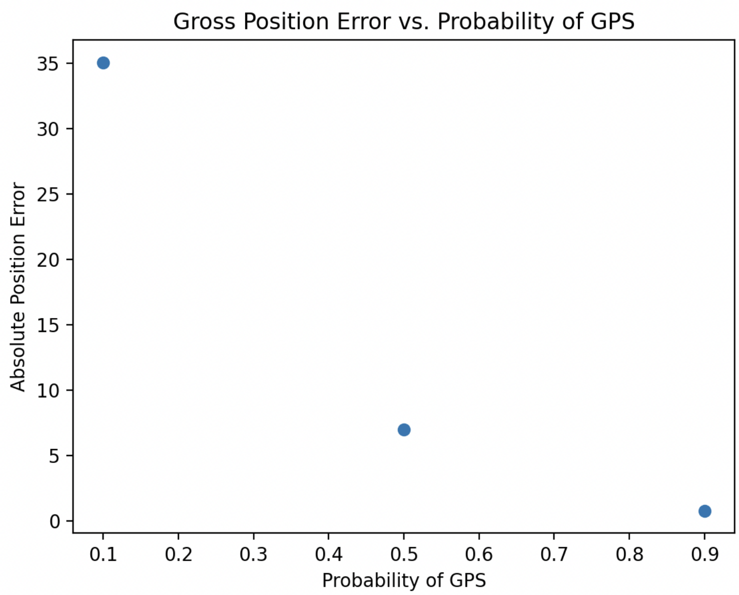
The parameters of the Gaussian estimate after the measurement are:

$$\mu_5 = \begin{bmatrix} 8.37563452 \\ 2.53807107 \end{bmatrix} \qquad \Sigma_5 = \begin{bmatrix} 6.70050761 & 2.03045685 \\ 2.03045685 & 1.82741117 \end{bmatrix}$$

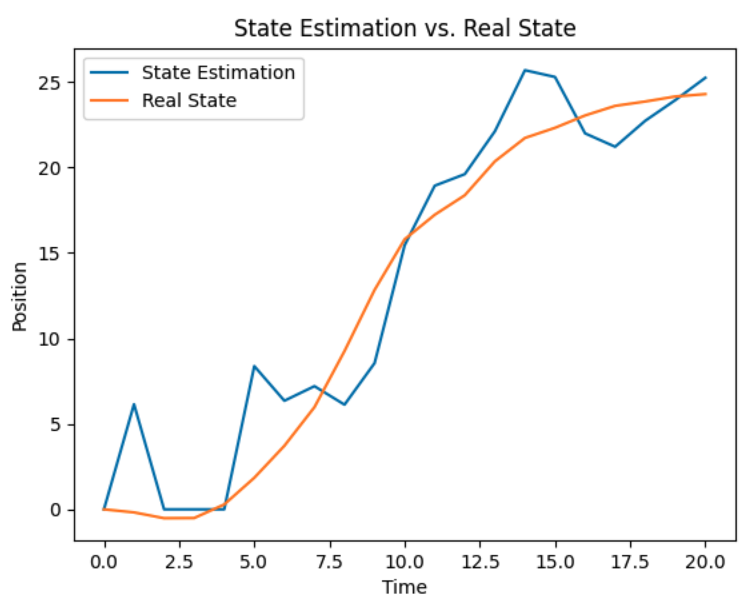
The mean adjusts towards the sensor reading; however it does not estimate the state to be x=10 since the sensor has a measured variance. The covariance matrix of the state significantly decreases because there is now more certainty to the current state.

Question 2.3: All of a sudden, the sky gets cloudy which may cause the sensor to fail and not produce a measurement with probability $p_{gps-fail}$. For three different values of this probability (e.g., 0.1, 0.5, and 0.9), compute and plot the expected error from the true position at time $t = 20$. You may do so by running up to N simulations and use the observed errors to obtain the expected error empirically.

Solution:



This figure shows the Error of the model for a single simulation of the model under three different GPS sensor probabilities. Note that the dependent variable here is $gps_{success}$ or $1 - gps_{failure}$, opposite from the way the problem was framed. If I had more time, I would have run each case at least 20 times. Averaging the errors from these simulations would offer a value much closer to the real expected error.



The second figure shows a single simulation of the Kalman filter estimate versus the actual state at a $P(gps_{success}) = .9$

3 Kalman Filter: Movement

Question 3.1 The drone is now fully operational and can not only take measurements, but also issue motor commands in the form of acceleration commands to its propeller. For example, a command of 1.0 will increase the drone's velocity by 1.0. Revisit Question 1.3 to provide the matrix B. If at time $t-1$, the drone's position and velocity are 5.0 and 1.0, compute the mean estimate for the state at time t given a motor command of 1.0. Your answer should be based on the constants provided but also include a random variable due to the wind effects. State the distribution of that random variable.

Solution:

$$x_{t-1} = \begin{bmatrix} 5 \\ 1 \end{bmatrix} \qquad \Sigma_{t-1} = \begin{bmatrix} 0 & 0 \\ 0 & 0 \end{bmatrix}$$

**No covariance since we were given the state.

$$p(x_t | u_t, x_{t-1}) = A x_{t-1} + B u_t$$
$$\Sigma_t = A_t \Sigma_{t-1} A_t^T + R_t$$

Using matrices A and B solved in Problem 2.1 we get

$$p(x_t | u_t, x_{t-1}) = \bar{x}_t = \begin{bmatrix} 6.5 \\ 2 \end{bmatrix} \qquad \bar{\Sigma_t} = R_t = \begin{bmatrix} .25 & .5 \\ .5 & 1 \end{bmatrix}$$

Since we are calculating the mean estimate, we can use the mean values of our Gaussian distributions (wind, sensor noise). Therefore, the sensor will give us an exact reading of the state and will not impact the estimation:

$Wind = \epsilon \sim N(0, 1)$ $\quad$ $\delta_t \sim N(0, \sqrt{8})$ **Note how each distribution has mean 0

$$x_t = \bar{x}_t = \begin{bmatrix} 6.5 \\ 2 \end{bmatrix}$$

$$\Sigma_t = (I - K_t C_t)\bar{\Sigma_t} = \Sigma_t = \begin{bmatrix} .24 & .48 \\ .48 & .97 \end{bmatrix}$$

In an example simulation, where there is sensor noise and wind, here are the results.

Simulation:

$$X_{real} = \begin{bmatrix} P_{real} \\ V_{real} \end{bmatrix} = \begin{bmatrix} 5.98 \\ .98 \end{bmatrix}$$

$$Wind = \epsilon_t \sim N(0,1) = -1.02 \qquad SensorReading = z_t = 7.41$$

Kalman Filter Estimation:

$$x_t = \begin{bmatrix} 6.53 \\ 2.06 \end{bmatrix} \qquad \Sigma_t = \begin{bmatrix} .24 & .48 \\ .48 & .97 \end{bmatrix}$$

$$\text{Error} = .55$$

4 Extra Credit

Now, formulate both the prediction and measurement steps in the 2-D case. Construct a plot showing the true position and the position tracked by the Kalman filter over the first 30 time steps.

What to turn in: A PDF document with the answers to the questions, along with the code implementation and a README file that describes what to run in order to get the results in your PDF. You can use a language of your choice.