Week 1 Summary

Subteam 2

May 2020

1 Introduction

The Kalman filter is rooted in the state-space formulation of linear dynamical systems. The state space model is described by two equations, with a general form of:

$$x'(t) = Ax(t) + +u(t) + w(t)$$

$$y(t) = Hx(t) + v(t)$$

where where x is the state vector, A is a square transformation matrix, B is the input matrix, u is a system input, w is the process noise vector, y is the transformed prediction, H is the observation matrix, v is the measurement noise vector, and t is time. It provides a recursive solution to the linear optimal filtering problem. The recursive solution estimates states using a combination of previous estimates and new input data.

The Kalman filter is described using a state space model. The state space model is described using two equations, the process equations which describes the latent states, denoted x_k and the measurement equation which describes the measurable variables, denoted y_k.

Process equation: describes the latent states. The latent state for the next iteration is computed by applying the transition matrix F to the state at the previous iteration

$$x_{k+1} = F_{k+1,k}x_k + w_k \ (1.1)$$

The matrix $F_{k+1,k}$ is the transition which takes the state x_k from time k to time $k+1$.
The process noise w is white, additive, and Gaussian with a mean of 0 and covariance matrix described in

$$E[w_n w_k^T] = \begin{cases} Q_k \text{ for n = k} \\ 0 \text{ for } \ n \neq k \end{cases} \quad (1.2)$$

Measurement equation:

$$y_k = H_k x_k + v_k \ (1.3)$$

y_k is the observable at time k and H_k is the measurement matrix. Measurement noise v is added in (1.3) and is white, additive, and Gaussian with a mean of 0 and covariance matrix described in

$$E[v_n v_k^T] = \begin{cases} R_k \text{ for n = k} \\ 0 \text{ for } n \neq k \end{cases} \quad (1.4)$$

$F_{k+1,k}$ and H_k are known quantities. It also is important to note that v and w uncorrelated.

2 Approaching the Problem

We will denote the posterior estimate of x_k by $\hat{x}_k$. The difference between x_k and $\hat{x}_k$ is called the estimation error, and it is defined as

$$\tilde{x}_k = x_k - \hat{x_k}$$

In order to derive this estimate, we need to find a cost function that is non-negative and non-decreasing function of $\tilde{x}_k$. Our solution is to use the MSE J, which is E$[\tilde{x}_k^2]$.

We will now state two theorems that will be helpful in our derivation of $\hat{x}_k$.

Theorem 1.1 (Conditional Mean Estimator): If the stochastic processes x_k and y_k are jointly Gaussian, then the optimum estimate $\hat{x}_k$ that minimizes the mean-square error J_k is the conditional mean estimator:

$$E[x|y_1, ..., y_k]$$

Theorem 1.2 (Principle of Orthogonality): Let the stochastic processes $\{x_k\}$ and $\{y_k\}$ be of zero mean, that is,

$$E[x_k] = E[y_k] = 0 \ for \ all \ k$$

Then:
(i) the stochastic processes x_k and y_k are jointly Gaussian; or
(ii) if the optimal estimate $\hat{x}_k$ is restricted to be a linear function of the observables and the cost function is the mean square error,
(iii) then the optimum estimate $\hat{x}_k$ given the observables $y_1, y_2, ..., y_k$ is the orthogonal projection of x_k on the space spanned by those observables.

Since we are using the mean-squared error to estimate $\hat{x}_k$, we know that (iii) holds. Thus, our x and y vectors are orthogonal to one another, meaning that their dot product must equal 0. As a result, we are able to make some assumptions about the orthogonality of certain variables. This is how we get most of the assumptions in our derivation, such as:

$$E[\hat{x}_k y_i^T] = 0$$

3 Understanding the Kalman Filter

We will now use the theorems above to derive the Kalman Filter.

3.1 Deriving the Kalman Gain

We will use $\hat{x}_k^-$ to refer to a prior estimate of x_k at time k. Using this prior estimate, a posterior estimate $\hat{x}_k$ can be described as a linear combination of the prior and the observable data y_k,

$$\hat{x}_k = G_k^{(1)}\hat{x_k}^- + G_k y_k \ (1.5)$$

These matrix factors will now be determined. The matrix factor G_k is referred to as the Kalman Gain. We will do this by invoking the principle of orthogonality defined in Theorem 1.2. The state error vector is defined as:

$$\tilde{x}_k = x_k - \hat{x}_k \ (1.6)$$

Applying Theorem 1.2 to this situation, we know that

$$E[\tilde{x}_k y_i^T] = 0 \ for \ all \ i = 1, ..., k-1 \ (1.7)$$

Using Equations. (1.3), (1.5), and (1.6) in (1.7), we get

$$E[(x_k - G_k^{(1)}\hat{x}_k^- - G_k H_k x_k - G_k w_k y_i^T] = 0 \ for \ all \ i = 1, ..., k-1 \ (1.8)$$

Due to w and v being uncorrelated, we know $E[w_k y_i^T] = 0$. Using this, we can expand (1.8) and arrive at

$$E[(I - G_k H_k - G_k^{(1)})x_k y_i^T + G_k^{(1)}(x_k - \hat{x}_k^-)y_i^T] = 0 \ (1.9)$$

We note from the principle of orthogonality that

$$E[(x_k - \hat{x}_k^-){y_i}^T] = 0$$

, Using this, we can simplify the previous equation to

$$(I - G_k H_k - G_k^{(1)})E[x_k y_i^T] = 0 \ (1.10)$$

In order to satisfy this equation for arbitrary values of x_k and y_i^T, the first term in (1.10) must equal 0, as below.

$$I - G_k H_k - G_k^{(1)} = 0$$

Solving this for $(G_k)^(1)$ yields

$$G_k^{(1)} = I - G_k H_k \ (1.11)$$

Substituting equation (1.11) into equation (1.5) yields

$$\hat{x}_k = \hat{x}_k^- + G_k(y_k - H_k x_k^-) \ \hat{(}1.12)$$

This describes how to get the posterior estimate $\hat{x_k}$ and is dependent on the Kalman Gain G_k. Next, we will derive an explicit formula for G_k First, we use the orthogonality principle to get

$$E[(x_k - \hat{x}_k)y_k^T] = 0 \ (1.13)$$

from which follows

$$E[(x_k - \hat{x}_k)\hat{y}_k^T] = 0 \ (1.14)$$

Here, $\hat{y}_k$ is an estimate of y_k given previous observable measurements $y_1, y_2, ..., y_k$ We will define the innovation process as:

$$\tilde{y_k} = y_k - \hat{y}_k \ (1.15)$$

The innovation process is a measure of the new information contained in y_k. By substituting in for $\hat{y}_k$ and doing some algebra, it may also be expressed as:

$$\begin{aligned} \tilde{y}_k &= y_k - H_k\hat{x}_k^- \\ &= H_k x_k + v_k - H_k\hat{x}_k^- \\ &= H_k\tilde{x}_k^- + v_k \ (1.16) \end{aligned}$$

By subtracting Equation (1.14) from (1.13) and then using the definition of Equation (1.15), we may write

$$E[(x_k - \hat{x}_k)\tilde{y}_k^T] = 0 \ (1.17)$$

Using (1.3) and (1.12), we can express the state-error vector as

$$\begin{aligned} & x_k - \hat{x}_k \\ &= \tilde{x}_k^- - G_k(H_k\tilde{x}_k^- + v_k) \\ &= (I - G_k Hk)\tilde{x}_k^- - G_k v_k \ (1.18) \end{aligned}$$

We can now plug in Equations (1.16) and (1.18) into Equation (1.17) to get the expression

$$E[(I - G_k H_k)\tilde{k}^- G_k v_k(H_k\tilde{x}_k + v_k)] = 0 \ (1.19)$$

Due to independence of v_k and x_k, can simplify to

$$(I - G_k H_k)E[\tilde{x}_k\tilde{x}_k^{T-}]H_k^T - G_k E[v_k v_k^T] \ (1.20)$$

Next is to develop the prior covariance matrix ${P_k}^-$, defined as

$$P_k^- = E[(x_k - \hat{x}_k^-)(x_k - \hat{x}_k^-)^T] = E[\tilde{x}_k^-\tilde{x}_k^{T-}] \ (1.21)$$

Using Equation (1.21) and the definition of covariance matrix of v_k given in beginning in (1.4), can rewrite (1.20) as

$$(I - G_k H_k)P_k^- H_k^T - G_k R_k$$

and solve for Kalman Gain, G_k:

$$G_k = P_k^- H_k^T[H_k P_k^- H_k^T + R_k]^{-1} \ (1.22)$$

This is crucial as it is the Kalman Gain function used earlier that we needed to solve for.

3.2 Covariance Matrices

Now that we understand the formulation for the Kalman Gain G_k and have a formula for how to calculate it, we must look at P_k^-, which is our one remaining unknown in (1.22). This serves as our motivation for understanding both P_k and P_k^- through the process of **error covariance propagation**. First, let us set up our definitions of both of these terms as:

$$P_k = E[\tilde{x}_k \tilde{x}_k^T]$$

and

$$P_k^- = E[\tilde{x}_k^- \tilde{x}_k^{T-}]$$

We can see that these are analogous terms where the **posterior prediction covariance** P_k depends on our posterior prediction error and the **prior prediction covariance** P_k^- depends on the prior prediction error.

We will proceed with error covariance propagation in two stages:

1. To predict the posterior covariance P_k given the prior P_k^-
2. To predict the prior covariance P_k^- given the posterior for time $k-1$, P_{k-1}

To begin with stage 1, first consider the definition of P_k, where the definition of $\tilde{x_k}$ is $x_k - \hat{x_k}$:

$$P_k = E[(x_k - \hat{x_k})(x_k - \hat{x_k})^T] \quad (1.23)$$

Now by using the definition of the difference $x_k - \hat{x_k}$ as is in (1.18), P_k can be rewritten as:

$$P_k = (I - G_k H_k) E[\tilde{x}_k \tilde{x}_k^T](I - G_k Hk)^T + G_k E[v_k vk^T] G_k^T$$

Now, we know that $E[\tilde{x}_k \tilde{x}_k^T]$ is precisely P_k^- and that $E[v_k vk^T]$ is exactly the covariance of the measurement noise, R_k, meaning that we obtain a new equation for P_k of:

$$P_k = (I - G_k H_k) P_k^- (I - G_k H_k)^T + G_k R_k G_k^T \quad (1.24)$$

Finally, by expanding and rearranging terms, we can arrive at:

$$\begin{aligned} P_k &= (I - G_k H_k) P_k^- (I - G_k H_k)^T + G_k R_k G_k^T \\ &= (I - G_k H_k) P_k^- - G_k R_k G_k^T + G_k R_k G_k^T \\ &= (I - G_k H_k) P_k^- \quad (1.25) \end{aligned}$$

Thus, since we are assuming that P_k^- is known and we have the goal of then calculating P_k, (1.25) thus establishes a clear relationship between the two that can be easily calculated since G_k and H_k are both known at this point. We can now proceed to stage 2.

To begin, let us rewrite our prior prediction in terms of the posterior at time $k-1$ using (1.26) as follows:

$$\hat{x}_k^- = F_{k,k-1}\hat{x}_{k-1} \quad (1.26)$$

Here, we are simply applying the transition matrix to move our posterior prediction at $k-1$ into time k, without any new data/observables being used yet. Now, we can use this definition for $\hat{x}_k^-$ to rewrite the prior estimation error as is shown in

$$\begin{aligned}\tilde{x}_k^- &= x_k - \hat{x}_k^- \\ &= (F_{k,k-1}x_{k-1} + w_{k-1}) - (F_{k,k-1}\hat{x}_{k-1}) \\ &= F_{k,k-1}(x_{k-1} - \hat{x}_{k-1}) + w_{k-1} \\ &= F_{k,k-1}\tilde{x}_{k-1} + w_{k-1} \quad (1.27)\end{aligned}$$

Finally, substitute (1.27) into our definition of the posterior covariance P_k^- described in (1.21) and use independence of w_k and $\tilde{x}_{k-1}$ to get expression for prior covariance matrix in

$$\begin{aligned}P_k^- &= F_{k,k-1}E[\hat{x}_{k-1}\hat{x}_{k-1}^T]F_{k,k-1} + E[w_{k-1}w_{k-1}^T] \\ &= F_{k,k-1}P_{k-1}F_{k,k-1}^T + Q_{k-1} \quad (1.28)\end{aligned}$$

Where Q_{k-1} resembles the covariance of the process noise w_{k-1} at time $k-1$. Thus, as was desired, (1.28) expresses a clear relationship between the posterior covariance at time $k-1$, P_{k-1}, and the prior covariance at time k, P_k^-. Now, using equations (1.25) and (1.28) together we are constantly able to understand our prior and posterior covariance matrices through simple calculations. This, in turn, means that the Kalman Gain G_k can now be calculated and used in the above equations.

3.3 Initialization

We have now developed an understanding of how the Kalman gain can be used to, given a state at time $k-1$, make predictions for the state of the system at time k. All that is left is to understand how to begin at time $k=0$. It is important to note that, in general, the initial guess one makes for the system at $k=0$ will not be very helpful or accurate, however it must be done in a consistent manner.

To begin, we must assume some sort of distribution for the x variables, and thus can initialize the posterior estimate as:

$$\hat{x}_0 = E[x_0] \quad (1.29)$$

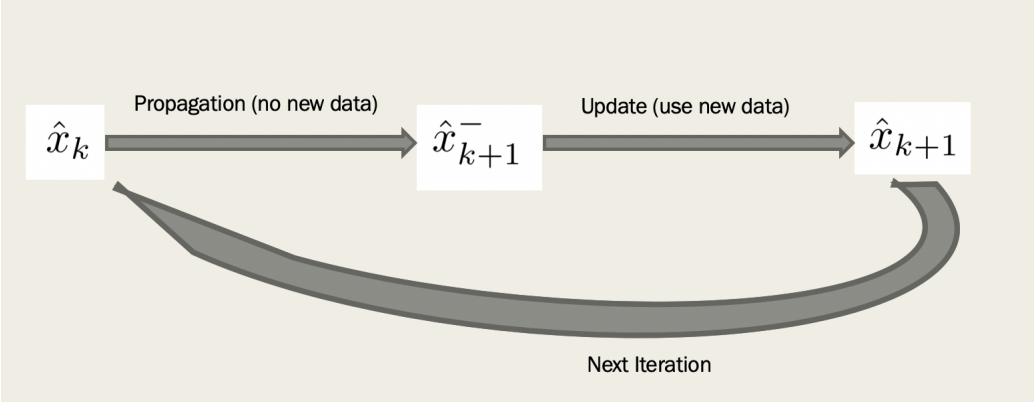
Under this assumption, the posterior variance is thus initialized to:

$$P_0 = E[(x_0 - E[x_0])(x_0 - E[x_0)^T] \quad (1.30)$$

Now, we have all the pieces in place to use the Kalman Filter. In the next section, let's put all of these equations next to each other to formalize the ordered process by which the Kalman Filter is used.

3.4 Summary

First, let's visualize the process by which the latent states are updated:



Here, we begin with a posterior estimate $\hat{x_k}$, to which the matrix F is applied, with **no new data considered** to reach $\hat{x}_{k+1}$. Then, **new data** is looked at, in the form of the observables y_k, in order to produce the posterior prediction $\hat{x}_{k+1}$. Next, $k := k+1$ and we proceed to the next iteration, using the posterior estimate we just calculated as our new starting point.

Now, let's look at a summary of the algorithm:

1. Model
   - Describe latent variable x through $x_{k+1} = F_{k+1,k}x_k + w_k$
   - Describe observable data though $y_k = H_k x_k + v_k$
2. Initialization
   - For k = 0, set $\hat{x}_0 = E[x_0]$
   - Also set $P_0 = E[(x_0 - E[x_0])(x_0 - E[x_0])^T]$
3. Computation
   - Propagation Step
     - State estimate Propagation: $\hat{x}_k^- = F_{k,k-1}\hat{x}_{k-1}$ (1.26)
     - Error Covariance Propagation: $P_k{}^- = F_{k,k-1}P_{k-1}F_{k,k-1}^T + Q_{k-1}$ (1.28)
   - Update Step
     - Kalman Gain Matrix: $G_k = P_k{}^- H_k{}^T [H_k P_k{}^- H_k{}^T + R_k]^{-1}$ (1.22)
     - State Estimate Update: $\hat{x}_k = \hat{x}_k^- + G_k(y_k - H_k\hat{x}_k^-)$ (1.12)
     - Error Covariance Update: $P_k = (I - G_k H_k)P_k{}^-$ (1.25)

In the **Model** section, we set up both our **Process** and **Measurement** equations. Next, as described in the **Initialization** section, we treat x_0 as a random variable and use $E[x_0]$ as our first posterior prediction, $\hat{x_0}$, from which the initial prior covariance matrix P_0 follows. Now, let's recap the purpose of each equation in the **Computation** section.

The purpose of (1.26) is to make an initial guess on our state at time k solely based on our "finalized" prediction for $k-1$ which was $\hat{x}_{k-1}$. Next, in (1.28), we create our prior covariance matrix, which needs to be used to calculate the Kalman Gain Matrix.

In (1.22), the Kalman Gain Matrix is calculated. Now having calculated the Kalman Gain, the prior estimate we made in (1.26) can be updated to our posterior estimate (i.e. our "finalized" estimate for the state), $\hat{x}_k$ in (1.12). Finally, our posterior covariance matrix P_k is calculated in (1.25) in order to be used in the next iteration.

This ordered process can be used to continuously move from time k to time $k+1$, and predictions will improve as more and more data is incorporated.

See the figure below for a visual depiction of what is input and output to both the Propagate and Update steps.

We have yet to discuss how one goes from a system of ODE's to the process equation that has been used throughout the derivation. The **linearization** that is required to do so will be covered in the next section, in which we step through an example.

FullDiagram.png

4 Example

Having now developed the theory to use a linear Kalman Filter, it is now useful to see an example of how the dimensions of all these matrix equations work out, as well as how a problem is initially set up. In order to do so, let us consider an example where we have a system of 2 ODE's representing our latent states:

$$\frac{dx^{(1)}}{dt} = ...$$

$$\frac{dx^{(2)}}{dt} = ...$$

And have three observable variables, $y^{(1)}$, $y^{(2)}$, and $y^{(3)}$ that can easily be calculated from the given data and are thus known.

Let us also assume that our transition matrix F and H are both known and contain only constant values, meaning that they are not time dependent. This assumption is made for the purpose of simplicity of the example.

The first step will be to linearize and discretize our differential equations. By this, we mean that we must turn the system into one that has data points only at discrete time points such as k and $k+1$ rather than a continous system, and in addition to this must use the discrete times to approximate derivatives. As a result of the linearization, our system will now look like:

$$x_{k+1}^{(1)} = F^{(1)} * x_k^{(1)} + w^{(1)}$$

$$x_{k+1}^{(2)} = F^{(2)} * x_k^{(2)} + w^{(2)}$$

Here, superscripts denote individual elements of what will become our vector of $x's$ and $w's$ and, in the case of F, denote a single row vector which, when stacked on top of one another, result in the transition matrix F.

After having linearized the system, we must now initialize our values $\hat{x}_0$ and P_0:

$$\hat{x}_0 = E[x_0] = \bar{X} = \begin{bmatrix} 2x1 \end{bmatrix}$$

where this is a 2 x 1 vector of the expectations of the two latent states.

As a result, we can also initialize:

$$P_0 = E[(x_0 - \bar{X})(x_0 - \bar{X})^T] = \begin{bmatrix} \ 2x2 \ \end{bmatrix}$$

We now have the framework needed to set up the model equations for movie from time 0 to time 1, and will include the matrices dimensions inside:
The process equation:

$$x_1 = Fx_0 + w_0$$

$$\begin{bmatrix} 2x1 \end{bmatrix} = \begin{bmatrix} \ 2x2 \ \end{bmatrix} \begin{bmatrix} 2x1 \end{bmatrix} + \begin{bmatrix} 2x1 \end{bmatrix}$$

And the measurement equation:

$$y_0 = Hx_0 + v_0$$

$$\begin{bmatrix} 3x1 \end{bmatrix} = \begin{bmatrix} \ 3x2 \ \end{bmatrix} \begin{bmatrix} 2x1 \end{bmatrix} + \begin{bmatrix} 3x1 \end{bmatrix}$$

As a reminder, the x vector represents latent states (unobservable aspects of the system) and the y vector measures observable variables, which will each depend in some way on the latent states. For example, if the measurement matrix, H was $\begin{pmatrix} 1 & 2 \\ 3 & 1 \\ 0 & 2 \end{pmatrix}$, then the measurement equation would be

$$\begin{pmatrix} y_1 \\ y_2 \\ y_3 \end{pmatrix} = \begin{pmatrix} 1 & 2 \\ 3 & 1 \\ 2 & 2 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} + \begin{pmatrix} w_1 \\ w_2 \\ w_3 \end{pmatrix}$$

which if we broke it up into individual equations for each y_i would give us:

$$y_1 = x_1 + 2x_2 + w_1$$
$$y_2 = 3x_1 + 2x_2 + w_2$$
$$y_3 = 2x_1 + 2x_2 + w_3$$

Basically, the model equations are representing a system of equations. Next, we perform the **State Estimate Propagation**, where we move our posterior prediction at time 0, $\bar{X}$ to time 1:

$$\hat{x}_1^- = F\bar{X}$$
$$\begin{bmatrix} 2x1 \end{bmatrix} = \begin{bmatrix} \ 2x2 \ \end{bmatrix} \begin{bmatrix} 2x1 \end{bmatrix}$$

The next step, following the order laid out in the **Summary**, is the **Error Covariance Propagation**, where P_1^- is calculated using P_0:

$$P_1^- = FP_0F^T + Q_0$$
$$\begin{bmatrix} \ 2x2 \ \end{bmatrix} = \begin{bmatrix} \ 2x2 \ \end{bmatrix} \begin{bmatrix} \ 2x2 \ \end{bmatrix} \begin{bmatrix} \ 2x2 \ \end{bmatrix} + \begin{bmatrix} \ 2x2 \ \end{bmatrix}$$

This concludes the **Propagation** portion of the algorithm, meaning that we now start the **Update** phase by first calculating the **Kalman Gain**:

$$G_1 = {P_1}^- * {H_1}^T * [H_1 * {P_1}^- * {H_1}^T + R_1]^{-1}$$
$$\begin{bmatrix} \ 2x3 \ \end{bmatrix} = \begin{bmatrix} \ 2x2 \ \end{bmatrix} \begin{bmatrix} \ 2x3 \ \end{bmatrix} \left(\begin{bmatrix} \ 3x2 \ \end{bmatrix} \begin{bmatrix} \ 2x2 \ \end{bmatrix} \begin{bmatrix} \ 2x3 \ \end{bmatrix} + \begin{bmatrix} \ 3x3 \ \end{bmatrix} \right)^{-1}$$

Having calculated the Kalman Gain, we can now perform the **State Estimate Update**:

$$\hat{x}_1 = \hat{x}_1^- + G_1(y_1 - H_1\hat{x}_1^-)$$
$$\begin{bmatrix} 2x1 \end{bmatrix} = \begin{bmatrix} 2x1 \end{bmatrix} + \begin{bmatrix} \ 2x3 \ \end{bmatrix} \left(\begin{bmatrix} 3x1 \end{bmatrix} - \begin{bmatrix} \ 3x2 \ \end{bmatrix} \begin{bmatrix} 2x1 \end{bmatrix} \right)$$

Our final calculation is now to do the **Error Covariance Update**:

$$P_1 = (I - G_1 H_1) {P_1}^-$$

$$\begin{bmatrix} \ 2x2 \ \end{bmatrix} = \left(\begin{bmatrix} \ 2x2 \ \end{bmatrix} - \begin{bmatrix} \ 2x3 \ \end{bmatrix} \begin{bmatrix} \ 3x2 \ \end{bmatrix} \right) \begin{bmatrix} \ 2x2 \ \end{bmatrix}$$

At this point we have done all the necessary calculations to predict the state at time 1 and could thus follow the same exact process, minus the initialization, to predict the state at time 2 and so on. Next, let's explore a possible issue that can arise while using the Linear Kalman Filter.

5 Divergence Issues

When using a linear Kalman Filter, severe problems can occur that will cause the filter to spiral out of control and no longer function properly. One possible source of issues concerns the definiteness of our covariance matrix P_k. In:

$$P_k = (I - G_k H_k) P_k^- \ (1.25)$$

There exists the possibility of the output of the computation, unless the numerical accuracy of the algorithm is guranteed to be high enough, to **not** be non-negative definite, meaning it will have negative eigenvalues. However, P_k is meant to represent a covariance matrix, which must at least be positive semi-definite and is, moreover, involved in equations where inverses are taken in (1.22). The solution lies in using the **Cholesky factorization**.

At every iteration of the Kalman Filter, one must perform the following step:

$$P_k = {P_k}^{1/2} {P_k}^{T/2}$$

Where ${P_k}^{1/2}$ is the **cholesky factor** and is in lower-triangular form, and ${P_k}^{T/2}$ is the cholesky factor's transpose. Here, P_k is now the product of a square matrix and its tranpose, which is **guranteed to be positive definite**. In order to find the cholesky factor, a function such as Matlab's *chol* (https://www.mathworks.com/help/matlab/ref/chol.html) can be used. By using this approach, one will have a better chance of maintaining the stability of the Kalman Filter across iterations.

6 Moving Forward

Throughout this week, we have gained an understanding of the manner in which Linear Kalman Filters can be used for state estimates. Our goal is now to expand our knowledge to Kalman Filters utilized in the context of nonlinear systems, specifically **Unscented Kalman Filters**. Then, combining UKF's with **Dual** and **Joint** approaches, we hope to be able to perform parameter estimation using UKF's first in the context of a Predator-Prey Model and then for the T1D Model we have read about.

In a **Dual** Kalman Filter approach, two separate chains of Kalman Filters are ran, one

for the latent states and another for the parameters one is trying to estimate. On the other hand, in a **Joint** approach, the parameters are treated as latent states, meaning they are directly incorporated into the x_k vector, so latent states and parameters are simply estimated at the same time within the same vector.