Análisis matemático para inteligencia artificial CEIA

TP 1
Aplicación filtro de Kalman

Alumno: Lic. Pedro Perez
Docente: Ing. Magdalena Bouza

Marco teórico Filtro de Kalman:

El Filtro de Kalman es conjunto de herramientas matemáticas que juega un importante papel para estimar el estado de un proceso, de un modo que minimiza la varianza estimada del error. Fue inventado por Rudolph Emil Kalman a finales de la década de 1950, con la finalidad de filtrar y predecir sistemas lineales. La idea fundamental del filtro de Kalman es la actualización, la llegada de una nueva observación supone un cambio en la mejor estimación. La figura 1 se esquematiza los procesos que forman parte del filtro de Kalman.

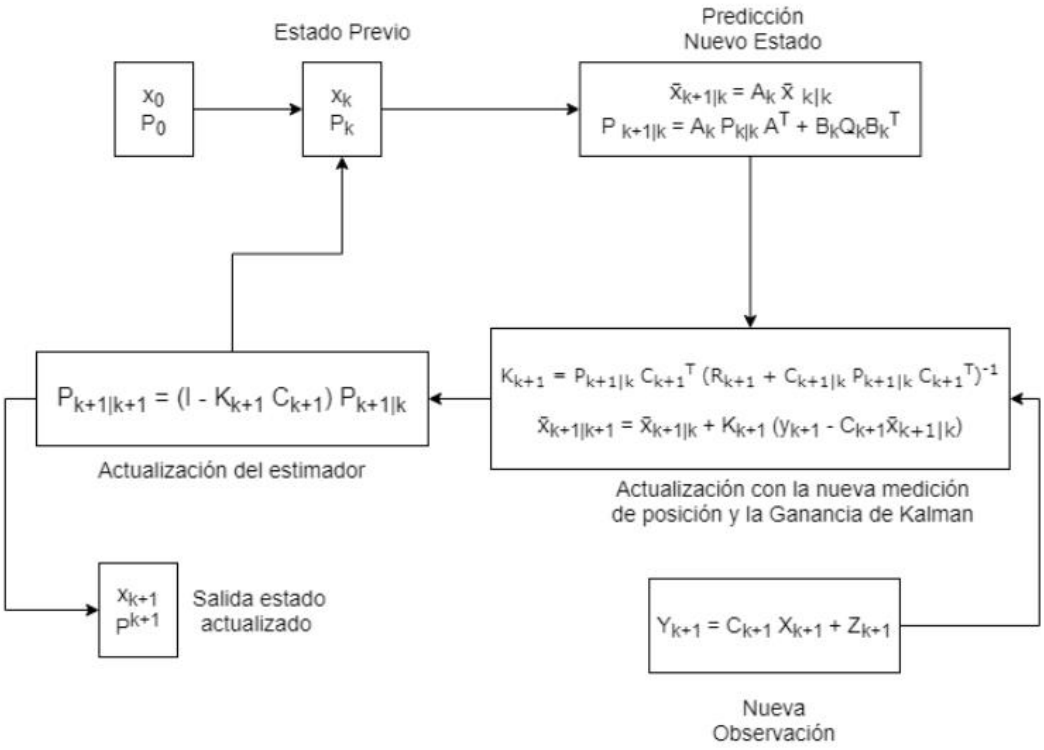

Figura 1 Modelo de Filtro de Kalman.

Modelo del sistema:

El modelo del sistema describe como es la evolución de la cantidad que se desea estimar en el tiempo, esa cantidad es expresada mediante un vector de estado $x_k \subseteq \mathbb{R}^n$. La transición entre el estado x_k y el estado x_{k+1}, se caracteriza por la matriz de transición A_k y la adición de ruido gaussiano w_k el cual tiene media cero y matriz de covarianza Q

$$x_{k+1} = A_k x_k$$

Modelo de la medición:

Este modelo relaciona el vector de medida Y_k con el estado del sistema X_k a través de la matriz de observación C_k y la adición de un ruido gaussiano z_k.

$$y_{k+1} = C_{k+1} X_{k+1} + z_{k+1}$$

Predicción:

El estimador de Kalman presenta dos pasos para la estimación, por un lado, la predicción y luego la corrección. En la etapa de predicción, se debe predecir la proyección del estado en el paso hacia adelante, así como la incertidumbre en la estimación.

$$x_{k+1} = A_k\, x_k$$

$$P_{k+1} = A_k\, P_k\, A^T + B_k Q_k B_k^T$$

P_{k+1} Es la matriz de covarianza del error del estimador en el instante antes de la observación actual y representa la incertidumbre en la observación.

Q_k Es la matriz que representa a la covarianza del ruido de las observaciones.

Actualización de la medición:

En el paso de corrección, se formula a través de una ecuación que asocia el estado estimado en la predicción del valor medido con la diferencia entre el valor real medido y el valor previsto.

$$K_{k+1} = P_{k+1}\, C_{k+1}{}^T\, (R_{k+1} + C_{k+1}\, P_{k+1}\, C_{k+1}T)^{-1}$$

$$x_{k+1} = x_k + K_{k+1}\, (y_{k+1} - C_{k+1}\, x_{k+1})$$

La matriz K, llamada matriz de ganancia del filtro Kalman, permite minimizar la covarianza del error de proceso. La ganancia de Kalman indica la confianza en las características observadas, empleando la incertidumbre de las observaciones junto con una medida de la calidad de las observaciones. La diferencia $(y_{k+1} - C_{k+1}\, x_{k+1})$ llamada innovaciones, indica la divergencia entre la medida a priori estimada y la medida actual y_{k+1}.

Si el error de la medición es grande en comparación con el error de la estimación, entonces la ganancia de Kalman va a ser pequeña. Por lo tanto, el estimador no sufre tanta actualización. Caso contrario, en la que el error de la estimación es grande en comparación con el error de medición, la ganancia de Kalman será grande y nuestra estimación sufrirá mayor actualización.

Actualización del estimador:

Es la etapa en la cual el error del estimador es actualizado. Si la ganancia de Kalman es grande quiere decir que la medición es buena en comparación con la estimación, por lo tanto, el error en la estimación se ve actualizado con mayor peso. Caso contrario, si la ganancia de Kalman es baja, la estimación es muy buena, por lo tanto, el error en la estimación no sufre actualización.

$$P_{k+1} = (I - K_{k+1}\, C_{k+1})\, P_{k+1}$$

Implementación en Python:

Se presenta un ejercicio en la cual hay que aplicar el filtro de Kalman para predecir la posición de un objeto en tres dimensiones, los datos suministrado son:

- Las observaciones de la posición, velocidad y aceleración en los ejes x,y,x.
- La posición del objeto en el momento cero $x_{0|0}$.
- El error de la estimación en el momento cero $P_{0|0}$

El primer paso fue leer los datos de las observaciones a un dataframe para su manipulación.

```
#leo los datos

posicion = pd.read_csv('posicion.dat',  sep='\s+', names=["x", "y", "z"]).to_numpy()
aceleracion = pd.read_csv('aceleracion.dat',  sep='\s+', names=["x", "y", "z"]).to_numpy()
velocidad = pd.read_csv('velocidad.dat',  sep='\s+', names=["x", "y", "z"]).to_numpy()


# Cargo los datos a un unico vector
X = []
for i in range(len(posicion)):
    X.append(np.array([posicion[i][0],   posicion[i][1],   posicion[i][2],
                       velocidad[i][0],  velocidad[i][1],  velocidad[i][2],
                      aceleracion[i][0],aceleracion[i][1],aceleracion[i][2]]))
```

Cada una de las funciones creadas corresponden un paso del filtro de Kalman:

<u>Función prediccion_estado:</u>

Parámetros:

t: instante de tiempo.

x_h: Estimación previa.

P_K: Estimador previo.

```
def prediccion_estado (t, x_k, P_k):
    a = (t**2)/2
    B = np.identity(9)
    Q = np.dot (0.3, B)
    A =np.array([[1, 0, 0, t, 0, 0, a, 0, 0],
                 [0, 1, 0, 0, t, 0, 0, a, 0],
                 [0, 0, 1, 0, 0, t, 0, 0, a],
                 [0, 0, 0, 1, 0, 0, t, 0, 0],
                 [0, 0, 0, 0, 1, 0, 0, t, 0],
                 [0, 0, 0, 0, 0, 1, 0, 0, t],
                 [0, 0, 0, 0, 0, 0, 1, 0, 0],
                 [0, 0, 0, 0, 0, 0, 0, 1, 0],
                 [0, 0, 0, 0, 0, 0, 0, 0, 1]])
    x_k_1 = np.dot(A, x_k)
    P_k_1 = A.dot(np.dot(P_k, A.T)) + B.dot(Q.dot(B.T))

    return x_k_1, P_k_1
```

la función calcula la proyección del estado en el paso hacia adelante x_{k+1}, así como la incertidumbre P_{k+1}, de la siguiente forma:

$$x_{k+1} = A_k \, x_k$$

$$P_{k+1} = A_k \, P_k \, A^T + B_k Q_k B_k^T$$

Donde Q = I * 0.3 por definición.

Función actualización_medicion:

Parámetros:

P_k_1: Estimador paso hacia adelante.

R: Error de medición.

x_k: Estimación en el momento inicial.

y_k_1: Nueva observación

```
#Actualizacion de la medición
def actualizacion_medicion (P_k_1, R, x_k, y_k_1):

    C =np.array([[1., 0., 0., 0., 0., 0., 0., 0.],
                 [0., 1., 0., 0., 0., 0., 0., 0.],
                 [0., 0., 1., 0., 0., 0., 0., 0.]])

    K = np.dot(P_k_1, C.T).dot(np.linalg.inv (R + C.dot(P_k_1.dot(C.T))))


    x_k_1 = x_k + K.dot(y_k_1 - C.dot(x_k))
    return x_k_1, K
```

Utiliza el valor calculado de la ganancia de Kalman K para actualizar el valor de la predicción del instante k+1.

$$K_{k+1} = P_{k+1}\, C_{k+1}^{T}\, (R_{k+1} + C_{k+1}\, P_{k+1}\, C_{k+1}T)^{-1}$$

$$x_{k+1} = x_k + K_{k+1}\, (y_{k+1} - C_{k+1}\, x_{k+1})$$

Función nueva_observacion:

Parámetros:

X_k_1: Nueva observación.

zk: Error en la medición.

```
def nueva_observacion (X_k_1, zk):
    C =np.array([[1, 0, 0, 0, 0, 0, 0, 0],
                 [0, 1, 0, 0, 0, 0, 0, 0],
                 [0, 0, 1, 0, 0, 0, 0, 0]])

    y_k_1 = np.dot(C,X_k_1) + zk
    return y_k_1
```

La función les da formato a las nuevas observaciones y las actualiza con el error de la medición zk:

$$y_{k+1} = C_{k+1}\, X_{k+1} + z_{k+1}$$

Función actualizar_estimador:

Parámetros:

K: Ganancia de Kalman.

P_k: Estimador en el momento previo.

```
def actualizar_estimador (K, P_k):
    I =  np.identity(9)
    C =np.array([[1., 0., 0., 0., 0., 0., 0., 0., 0.],
                 [0., 1., 0., 0., 0., 0., 0., 0., 0.],
                 [0., 0., 1., 0., 0., 0., 0., 0., 0.]])

    Pk_1 = (I - K.dot(C)).dot(P_k)
    return Pk_1
```

La función actualiza el estimador P utilizando la ganancia de Kalman utilizando la siguiente ecuación:

$$P_{k+1} = (I - K_{k+1} C_{k+1}) P_{k+1}$$

1) **Se mide la posición afectada por ruido blanco (gaussiano) de 10m de desvío estándar.**

En la figura 2 se muestra una grafico de posición del objeto, cada línea representa el valor estimado y el valor observado utilizando ruido gaussiano. Se observa que la predicción se acerca al valor observado cuando el ruido es gaussiano. Sin embargo, cuando nos acercamos vemos que existe una diferencia entre los valores.

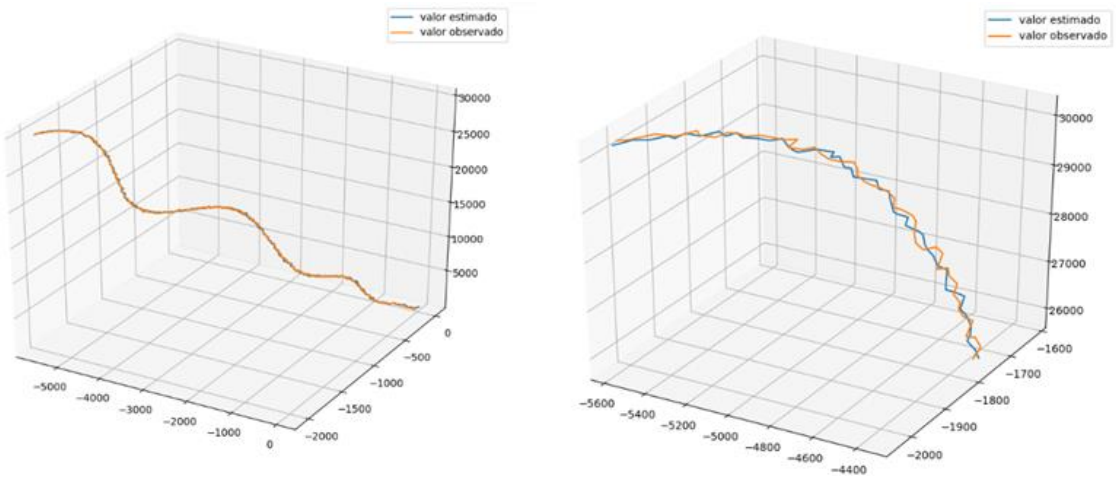

Figura 2 Posición estimada y observada ruido gaussiano.

2) **Se mide la posición afectada por ruido blanco (uniforme) de 10m de desvío estándar. Similar al ítem anterior. La idea es comparar con el ítem anterior. Analizar si hubo algún cambio.**

En la figura 3 se muestra una grafico de posición del objeto, cada línea representa el valor estimado y el valor observado utilizando ruido uniforme. Se observa que la predicción se acerca mucho mejor al valor observado en comparación con el caso anterior.

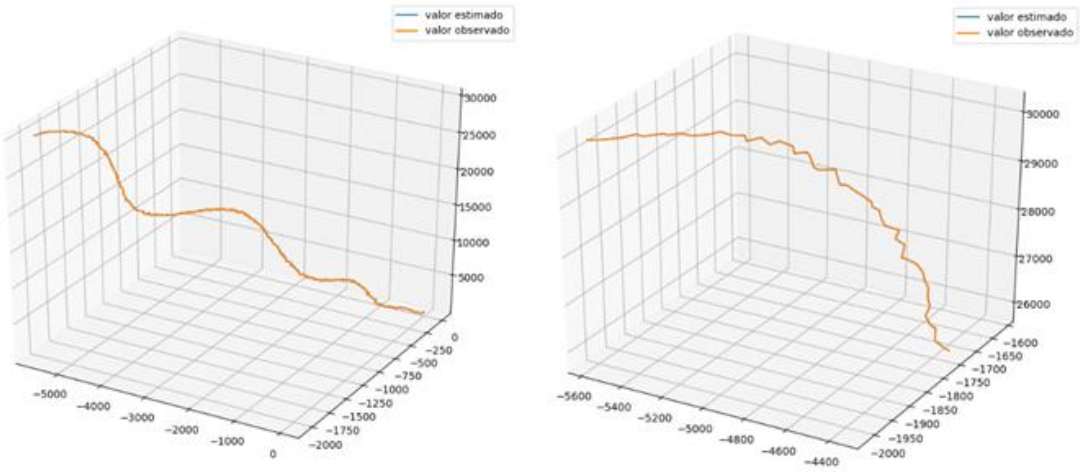

Figura 3 Posición estimada y observada ruido uniforme.

En la figura 4 se observa que la diferencia entre el valor estimado y el valor observado de la medición tienen a disminuir cuando el ruido es uniforme, y se mantiene constante cuando el ruido es gaussiano, por lo que podemos decir que el filtro de Kalman tiene mejor desempeño cuando el ruido de la medición es uniforme.

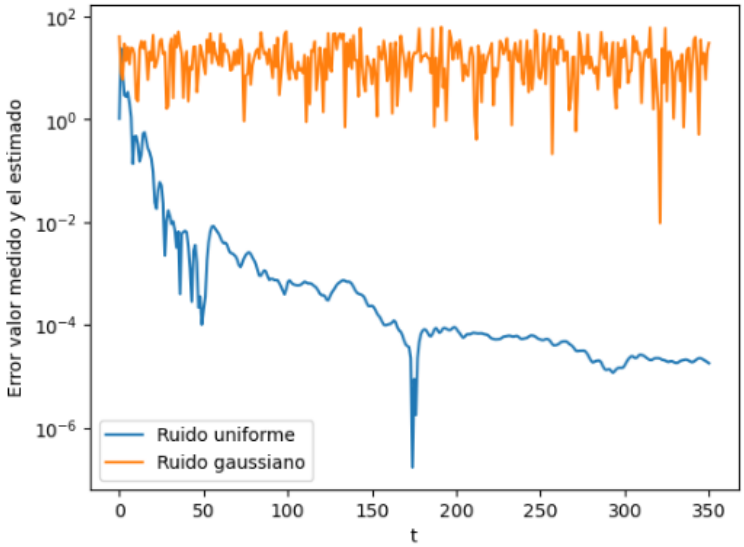

Figura 4 Diferencia entre posición calculada y la posición observada.

3) Se mide la posición y la velocidad afectadas por ruido blanco (gaussiano) de 10m y 0.2m/s de desvíos respectivamente. Analizar si mejora la estimación.

En este punto del ejercicio se pide considerar la posición y la velocidad de la medición utilizando un ruido gaussiano para ambas variables y comparar la estimación. En la figura 5 se observa que la predicción cuando se medie las variables posición y velocidad fue mucho mejor a la estimación obtenida cuando solo se mide solo la posición. En la figura 6 se observa que el error en la estimación decrece constantemente cuando se toma en consideración la posición y velocidad. Lo cual tiene sentido ya que se están considerando más variables para actualizar el estimador.

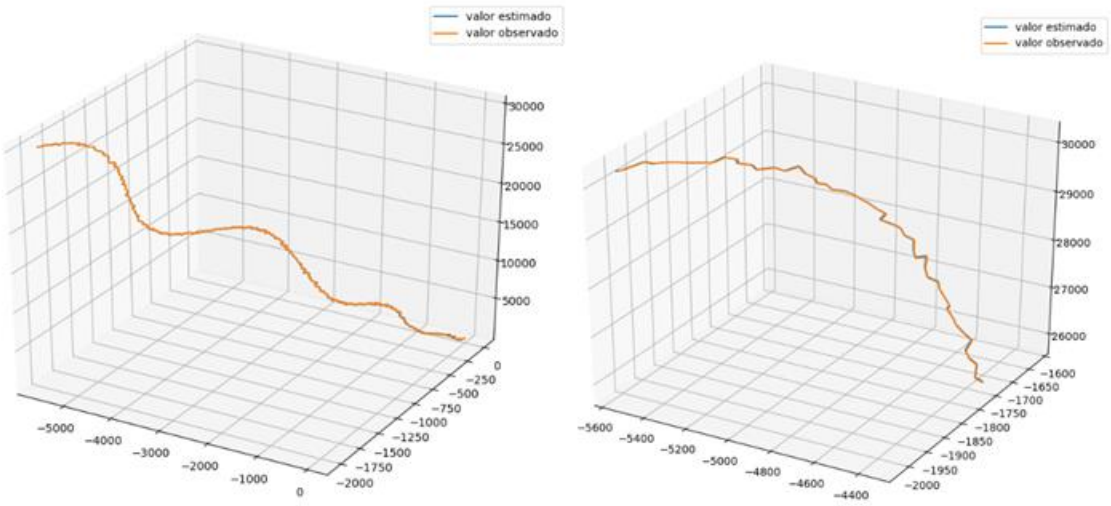

Figura 5 Posición estimada y observada medición con dos variables.

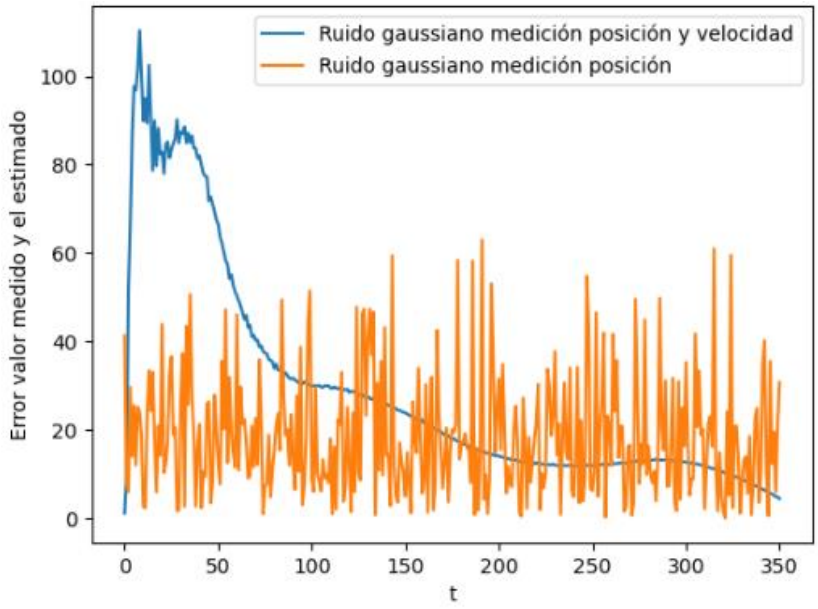

Figura 6 Diferencia entre posición calculada y la posición observada.