

---

---

# Filtro de Kalman y Filtro de Kalman Extendido

Eva Fernández de la Cruz  
Mario Delicado Díaz

---

# Enunciado (KF)

En este ejercicio, lo que se pedía era completar un código que implementaba un filtro de kalman simple, en el cual teníamos los siguientes datos:

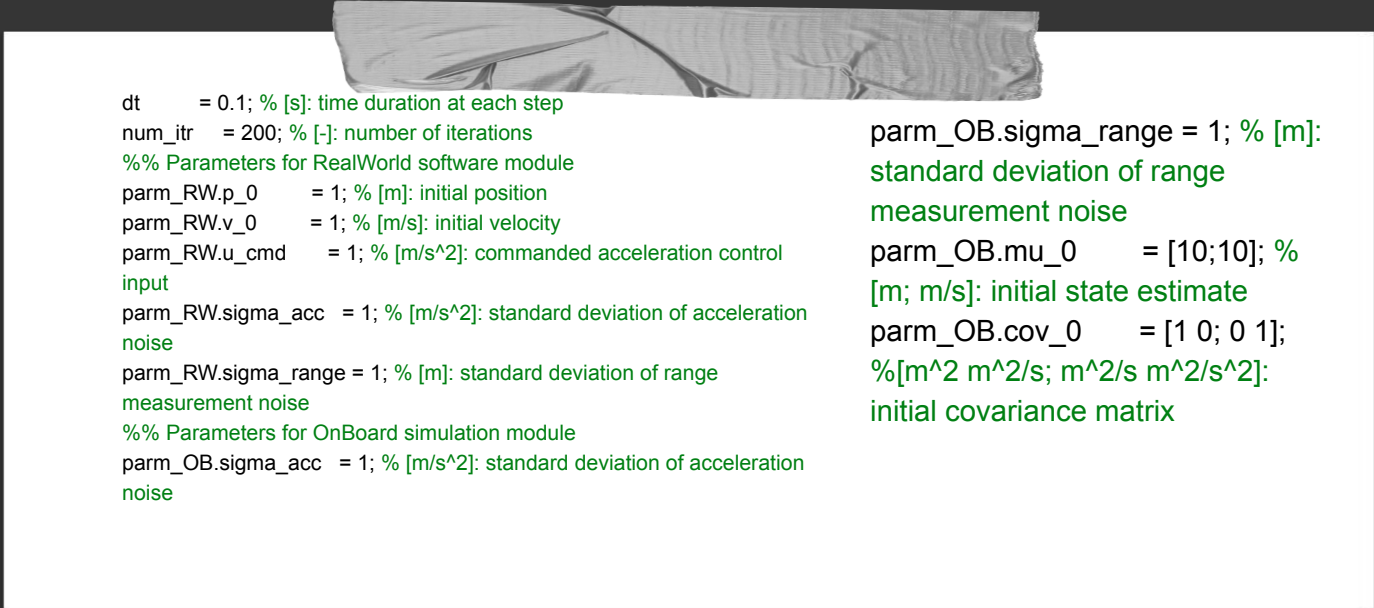
Un robot en el suelo se está moviendo en línea recta con una aceleración constante. Su aceleración y posición son medidas por un acelerómetro y un medidor de distancia, respectivamente. Se asume que el sistema de guía, el sistema de control y los actuadores son perfectos, de tal manera que el robot se está moviendo efectivamente con la aceleración constante ordenada. También se considera que no hay perturbaciones. Los únicos ruidos que deben tenerse en cuenta en las matrices de covarianza del ruido son los ruidos de los sensores.

Y, a continuación, el código a completar, que se muestra a continuación junto a la explicación del procedimiento.



# Solución


Lo primero que se hizo fue comprender las variables dadas en el enunciado



```
dt      = 0.1; % [s]: time duration at each step
num_itr = 200; % [-]: number of iterations
%% Parameters for RealWorld software module
parm_RW.p_0    = 1; % [m]: initial position
parm_RW.v_0    = 1; % [m/s]: initial velocity
parm_RW.u_cmd  = 1; % [m/s^2]: commanded acceleration control
input
parm_RW.sigma_acc = 1; % [m/s^2]: standard deviation of acceleration
noise
parm_RW.sigma_range = 1; % [m]: standard deviation of range
measurement noise
%% Parameters for OnBoard simulation module
parm_OB.sigma_acc = 1; % [m/s^2]: standard deviation of acceleration
noise
```

```
parm_OB.sigma_range = 1; % [m]:
standard deviation of range
measurement noise
parm_OB.mu_0        = [10;10]; %
[m; m/s]: initial state estimate
parm_OB.cov_0       = [1 0; 0 1];
%[m^2 m^2/s; m^2/s m^2/s^2]:
initial covariance matrix
```

Ahora, en función a estos datos, compeltamos la simulación del mundo real de la siguiente manera:



```
%% Linear state model (A amd B) and observation model (C)
A = [1 dt; 0 1];    % Matriz de transición de estado
B = [0.5*dt^2; dt]; % Matriz de control (efecto de la aceleración en posición y velocidad)
% Matriz de observación del sensor de rango (solo observa la posición)
H = [1 0];          % Solo la posición se mide con el sensor de rango
%% Initial condition
% Inicialización de la posición y velocidad
x_real(:, 1) = [parm_RW.p_0; parm_RW.v_0]; % Estado inicial [posición inicial; velocidad inicial]
% Inicialización de la medición del acelerómetro y el sensor de rango
u(1) = parm_RW.u_cmd + parm_RW.sigma_acc * randn(); % Aceleración inicial con ruido
z(1) = H * x_real(:, 1) + parm_RW.sigma_range * randn(); % Medición de rango inicial con ruido
```



```
%% Simulated state (Euler method) and accelerometer/range measurements
```

```
i=2;
```

```
while i<= num_itr+1
```

```
    % Actualización del estado real del sistema (mundo real) con la aceleración constante
```

```
    x_real(:, i) = A * x_real(:, i-1) + B * parm_RW.u_cmd;
```

```
    % Medición ruidosa del acelerómetro (aceleración medida)
```

```
    u(i) = parm_RW.u_cmd + parm_RW.sigma_acc * randn();
```

```
    % Medición ruidosa del sensor de rango (posición medida)
```

```
    z(i) = H * x_real(:, i) + parm_RW.sigma_range * randn();
```

```
    i=i+1;
```

```
end
```

---

Después, se rellenaron los huecos del código dado siguiendo el modelo matemático del filtro de kalman, que maneja la posición y velocidad lineales del robot. Para hacerlo, seguimos los siguientes pasos.

## 1. Predicción

- Se predice el estado actual del sistema y la covarianza de error.
- Ecuación de predicción del estado:

$$\hat{x}_k|k-1 = F \hat{x}_{k-1} + B u_k$$

Donde:

- $F$  es la matriz de transición de estado.
  - $B$  es la matriz de control.
  - $u_k$  es el vector de control (opcional).
- Ecuación de predicción de la covarianza:  
$$P_k|k-1 = F P_{k-1} F^T + Q$$

Donde:

  - $P$  es la matriz de covarianza del error.
  - $Q$  es la covarianza del ruido del proceso.

%% KF state model and observation model

% Matrices de estado (A y B) y modelo de observación (H)

A = [1 dt; 0 1]; % Matriz de transición de estado

B = [0.5\*dt^2; dt]; % Matriz de control

H = [1 0]; % Observación (solo se mide la posición)

% Covarianzas del proceso y del sensor

Q = [0.25\*dt^4, 0.5\*dt^3; 0.5\*dt^3, dt^2] \* parm\_OB.sigma\_acc^2; % Covarianza del ruido del proceso (acelerómetro)

R = parm\_OB.sigma\_range^2; % Covarianza del ruido del sensor (rango)

%% Initial estimate and error covariance

mu(:, 1) = parm\_OB.mu\_0; % Estimación inicial de estado

cov(:, :, 1) = parm\_OB.cov\_0; % Matriz de covarianza inicial

%% Recursive Kalman Filter Algorithm

i=2;

while i<= num\_itr+1

    %--- Predicción ---

    % Predicción del estado siguiente

    mu\_bar(:, i) = A \* mu(:, i-1) + B \* u(i-1); % Predicción del estado usando el control (aceleración)

    % Predicción de la covarianza de error

    cov\_bar(:, :, i) = A \* cov(:, :, i-1) \* A' + Q; % Predicción de la covarianza

    %--- Innovación (residual) ---

    z\_bar(i) = H \* mu\_bar(:, i); % Predicción de la medición de rango

    inn(i) = z(i) - z\_bar(i); % Innovación: diferencia entre la medición real y la predicha



## 2. Actualización (Corrección)

- Se ajusta la predicción utilizando las mediciones observadas para mejorar la estimación.
- Cálculo de la ganancia de Kalman:

$$K_k = P_{k|k-1} H^T (H P_{k|k-1} H^T + R)^{-1}$$

Donde:

- $H$  es la matriz que relaciona el estado con la medición.
- $R$  es la covarianza del ruido de medición.
- Actualización del estado:

$$\hat{x}_k = \hat{x}_{k|k-1} + K_k (z_k - H \hat{x}_{k|k-1})$$

Donde  $z_k$  es la medición observada.

- Actualización de la covarianza:

$$P_k = (I - K_k H) P_{k|k-1}$$

Donde  $I$  es la matriz identidad.

%--- Actualización ---

% Ganancia de Kalman

$S = H * cov\_bar(:, :, i) * H' + R;$  % Varianza residual (S)

$K = cov\_bar(:, :, i) * H' / S;$  % Ganancia de Kalman

% Actualización del estado

$\mu(:, i) = \mu\_bar(:, i) + K * inn(i);$  % Estado corregido

% Actualización de la covarianza del error

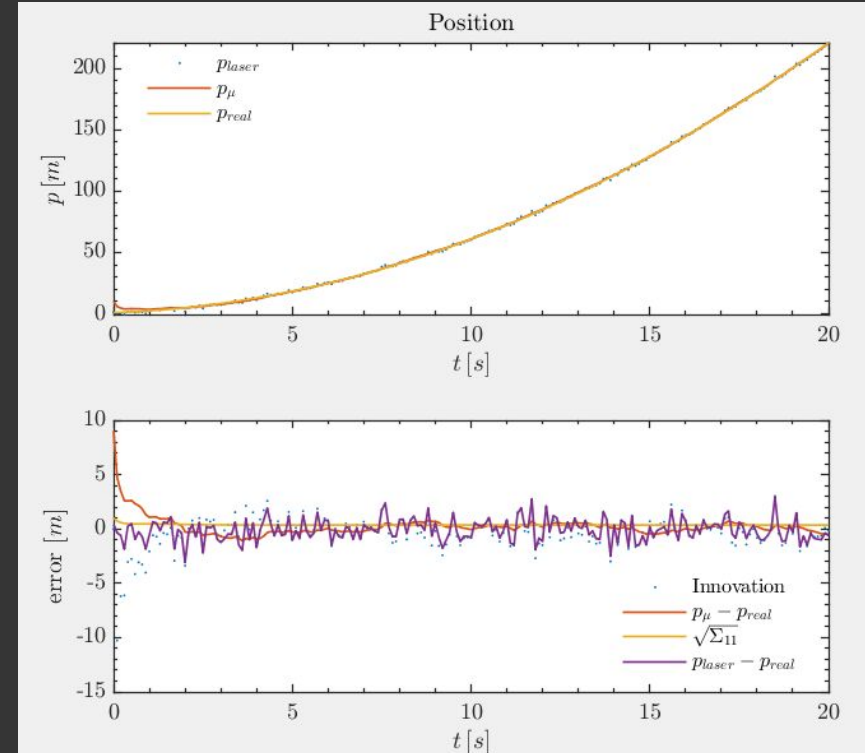
$cov(:, :, i) = (eye(2) - K * H) * cov\_bar(:, :, i);$  % Covarianza corregida

$i=i+1;$

end

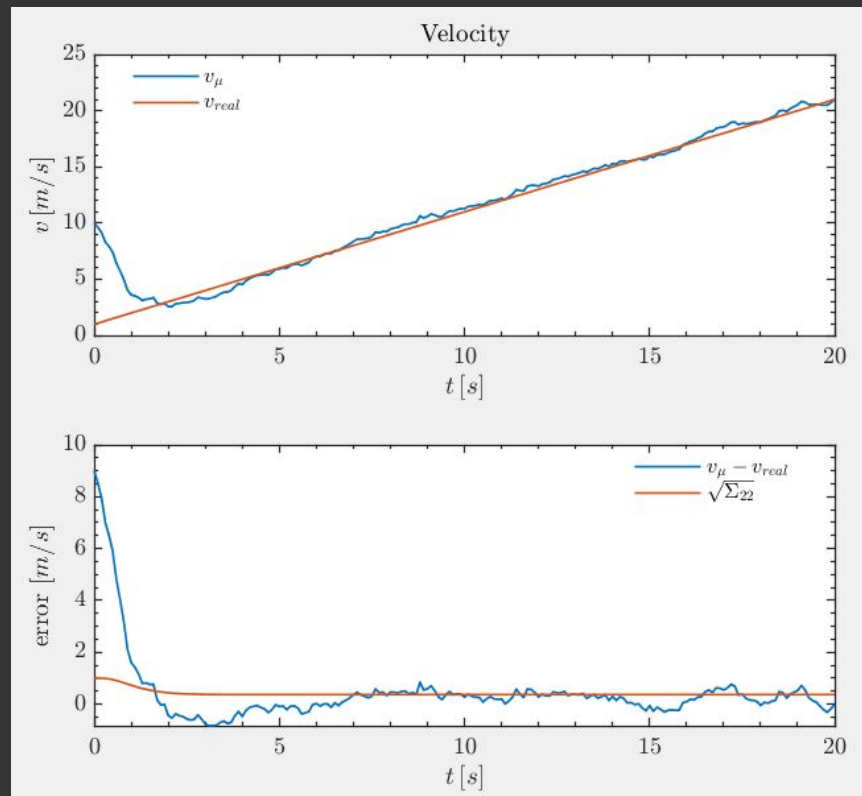
# Resultados. Posición

- **plaser:** Mide la posición del robot obtenida a partir del sensor (mediciones ruidosas).
- **$p_\mu$ :** Estimación de la posición del robot realizada por el filtro de Kalman.
- **preal:** Posición real del robot, basada en el modelo (sin ruido).
- **$p_\mu$ -preal:** Error de estimación de la posición.
- **$\sqrt{\Sigma_{11}}$ :** Raíz cuadrada de la varianza de la posición (covarianza), que representa la incertidumbre de la estimación.
- **plaser-preal:** Diferencia entre la medición del sensor y la posición real.



# Resultados. Velocidad

- $v_{\mu}$ : Estimación de la velocidad realizada por el filtro de Kalman.
- $v_{real}$ : Velocidad real del robot.
- $v_{\mu} - v_{real}$ : Error de estimación de la velocidad.
- $\Sigma_{22}$ : Raíz cuadrada de la varianza de la velocidad.



# Enunciado (EKF)

Se utiliza un sistema de radar terrestre para rastrear una aeronave que vuela a una velocidad y altitud constantes dentro de un plano vertical. La posición de la aeronave se expresa en coordenadas cartesianas, donde la posición horizontal se denota por  $x$  y la altitud por  $y$ . El radar es capaz de proporcionar dos tipos de mediciones: la distancia de alcance oblicua,  $r$ , y el ángulo de elevación,  $\epsilon$  :

$$r = \sqrt{(x - x_r)^2 + (y - y_r)^2}, \quad \gamma = \arctan \frac{y - y_r}{x - x_r},$$

son donde  $x_r$  e  $y_r$  son las coordenadas del radar.

En este problema no hay variable de control. Las variables de estado y de medición son:

$$\mathbf{x} = \begin{bmatrix} x \\ v \\ y \end{bmatrix}$$

$$\mathbf{z} = \begin{bmatrix} r \\ \gamma \end{bmatrix}.$$



# Enunciado (EKF)

Se dan los siguientes valores numéricos.

El intervalo de paso de tiempo constante es  $\Delta t = 0,1$

La posición del radar es  $x_r = y_r = 0$  m.

La posición inicial de la aeronave es  $x_0 = 0$  m,  $y_0 = 1000$  m .

La velocidad prevista de la aeronave es  $v = 100$  m s<sup>-1</sup> .

Se consideran los siguientes tipos de perturbaciones y ruidos.

Las desviaciones estándar de las mediciones del radar son  $\sigma_r = 5$  m,  $\sigma_\gamma = 0.5^\circ$  .

Las desviaciones estándar de la velocidad horizontal y la altitud de la aeronave son  $\sigma_v = 0.02$  m s<sup>-1</sup> ,  $\sigma_y = 0.1$  m .

La media y la covarianza de la creencia inicial son:

$$\mu_0 = \begin{bmatrix} 100 \text{ m} \\ 50 \text{ m s}^{-1} \\ 500 \text{ m} \end{bmatrix}$$

$$\Sigma_0 = \begin{bmatrix} 100 \text{ m}^2 & 0 & 0 \\ 0 & 25 \text{ m}^2 \text{ s}^{-2} & 0 \\ 0 & 0 & 100 \text{ m}^2 \end{bmatrix} .$$





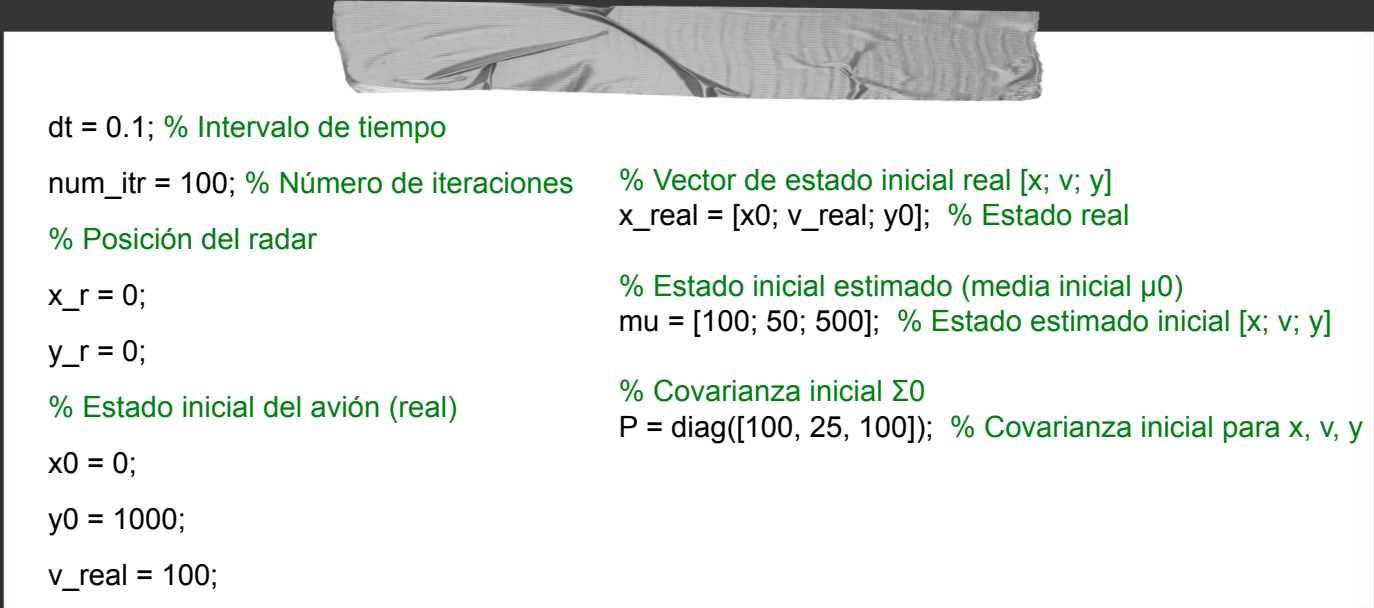
# Tareas

Siguiendo el ejemplo del ejercicio KF, construya el modelo matemático para:

- Simulación del mundo real
- Filtro de Kalman Extendido a bordo
- Representar las figuras pedidas

# Solución

Lo primero que se hizo fue definir las variables dadas en el enunciado



```
dt = 0.1; % Intervalo de tiempo
num_itr = 100; % Número de iteraciones
% Posición del radar
x_r = 0;
y_r = 0;
% Estado inicial del avión (real)
x0 = 0;
y0 = 1000;
v_real = 100;

% Vector de estado inicial real [x; v; y]
x_real = [x0; v_real; y0]; % Estado real

% Estado inicial estimado (media inicial  $\mu_0$ )
mu = [100; 50; 500]; % Estado estimado inicial [x; v; y]

% Covarianza inicial  $\Sigma_0$ 
P = diag([100, 25, 100]); % Covarianza inicial para x, v, y
```



% Covarianza inicial  $\Sigma$

P = diag([100, 25, 100]); % Covarianza inicial para x, v, y

% Ruido del proceso (avión: velocidad y altitud)

Q = diag([0, 0.02^2, 0.1^2]);

% Ruido de las mediciones (radar: rango y ángulo)

R = diag([5^2, (0.5\*pi/180)^2]); % Convertir ángulo a radianes

% Matriz de transición del estado (solo cambia  $x=x+v \cdot dt$ , 'v' e 'y' son ctes)

F = [1 dt 0;

0 1 0;

0 0 1];

% Medidas del radar en función del estado

h = @(x) [sqrt((x(1) - x\_r)^2 + (x(3) - y\_r)^2); % rango

atan2(x(3) - y\_r, x(1) - x\_r)]; % ángulo

—

A continuación, hacemos la simulación del mundo real en base a:

$$\begin{aligned}x_k &= g(x_{k-1}) + \epsilon_k \\z_k &= h(x_k) + \delta_k\end{aligned}$$

---

# Para ello, primero necesitamos definir algunas variables:



```
mu_hist = zeros(3, num_itr); % Para guardar la historia del estado estimado
```

```
P_hist = zeros(3, 3, num_itr); % Para guardar la historia de la covarianza
```

```
x_real_hist = zeros(3, num_itr); % Para guardar la historia del estado real
```

```
innovation_hist = zeros(2, num_itr); % Para guardar la historia de la innovación
```

—

Después, teniendo en cuenta el número de iteraciones elegido anteriormente, actualizamos cada una de las variables siguiendo el modelo matemático:

$$\begin{aligned}x_k &= g(x_{k-1}) + \epsilon_k \\z_k &= h(x_k) + \delta_k\end{aligned}$$

Y, a su vez, ya aplicamos el filtro de kalman extendido para comprobar las diferencias entre lo real y lo filtrado

—

Para hacer el filtro de kalman extendido,  
hemos tenido en cuenta lo siguiente:

$$g(x_{k-1})$$

$$G(\mu_{k-1})$$

$$R$$

$$h(x_k)$$

$$H(\bar{\mu}_k)$$

$$Q$$

---

Una vez explicados los modelos a seguir, el modelo matemático aplicado ha sido el siguiente:

El Filtro de Kalman Extendido maneja el problema no lineal de las mediciones de radar, que involucran la distancia oblicua ( $r$ ) y el ángulo de elevación ( $c$ ).

### Vector de Estado

El vector de estado es:

$$x = [x, v, y]$$

Donde:

- $x$ : posición horizontal
- $v$ : velocidad
- $y$ : posición vertical

---

## Mediciones de Radar

Las mediciones de radar están relacionadas con el estado de manera no lineal:

$$r = \sqrt{(x_r - x)^2 + (y_r - y)^2}$$

$$c = \arctan((y - y_r) / (x - x_r))$$

## Modelo del Sistema

La transición de estado es similar a la del Filtro de Kalman, pero las ecuaciones de medición son no lineales, por lo que se utiliza la linealización a través del Jacobiano (H) para aplicar el EKF.

Entonces, para hacer el filtro de kalman extendido, tendríamos que seguir los siguientes pasos si lo hiciéramos a mano:

Lo que se traduce en código de la siguiente forma:

## 1. Predicción

- Se predice el estado y la covarianza usando las ecuaciones de transición del sistema no lineal.

- Ecuación de predicción del estado:

$$\hat{x}_k|k-1 = f(\hat{x}_{k-1}, u_{k-1})$$

Donde  $f(\cdot)$  es una función no lineal que describe la dinámica del sistema.

- Ecuación de predicción de la covarianza:

$$P_k|k-1 = F_{k-1} P_{k-1} F_{k-1}^T + Q_{k-1}$$

Donde  $F_{k-1}$  es el Jacobiano de la función de transición evaluada en el estado anterior, y  $Q$  es la covarianza del ruido de proceso.

% Predicción del estado y la covarianza (para el estado estimado)

mu\_pred = F \* mu;

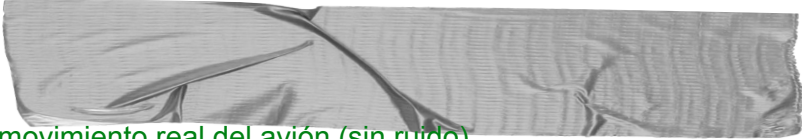
P\_pred = F \* P \* F' + Q;



## 2. Actualización de la Medición

- Las mediciones no lineales se usan para corregir la predicción.
- Jacobiano de la función de medición:  
Se linealiza la relación no lineal entre el estado y las mediciones mediante el Jacobiano  $H_k$ , que es la derivada de la función de medición  $h(\cdot)$ .
- Ecuación de la ganancia de Kalman:  
$$K_k = P_{k|k-1} H_k^T (H_k P_{k|k-1} H_k^T + R_k)^{-1}$$
  
Donde  $R_k$  es la covarianza del ruido de medición.
- Corrección del estado:  
$$\hat{x}_k = \hat{x}_{k|k-1} + K_k (z_k - h(\hat{x}_{k|k-1}))$$
  
Donde  $z_k$  es la medición observada y  $h(\hat{x}_{k|k-1})$  es la medición predicha.
- Corrección de la covarianza:  
$$P_k = (I - K_k H_k) P_{k|k-1}$$

## Lo que se traduce en código de la siguiente manera:



```
for k = 1:num_itr
    % Simulación del movimiento real del avión (sin ruido)
    x_real = F * x_real; % Actualización del estado real

    % Simulación de las mediciones con ruido
    z_real = h(x_real) + sqrt(R) * randn(2, 1);

    % Predicción del estado y la covarianza (para el estado estimado)
    mu_pred = F * mu;
    P_pred = F * P * F' + Q;

    % Jacobiano de la función de medición
    H = [(mu_pred(1) - x_r)/sqrt((mu_pred(1) - x_r)^2 + (mu_pred(3) - y_r)^2), 0, (mu_pred(3) - y_r)/sqrt((mu_pred(1) - x_r)^2 + (mu_pred(3) - y_r)^2);
        -(mu_pred(3) - y_r)/((mu_pred(1) - x_r)^2 + (mu_pred(3) - y_r)^2), 0, (mu_pred(1) - x_r)/((mu_pred(1) - x_r)^2 + (mu_pred(3) - y_r)^2)];
```



% Innovación (diferencia entre la medición real y la predicha)

y\_k = z\_real - h(mu\_pred);

% Covarianza de la innovación

S = H \* P\_pred \* H' + R;

% Ganancia de Kalman

K = P\_pred \* H' / S;

% Actualización del estado estimado y la covarianza

mu = mu\_pred + K \* y\_k;

P = (eye(3) - K \* H) \* P\_pred;

% Guardar la historia

mu\_hist(:, k) = mu;

P\_hist(:, :, k) = P;

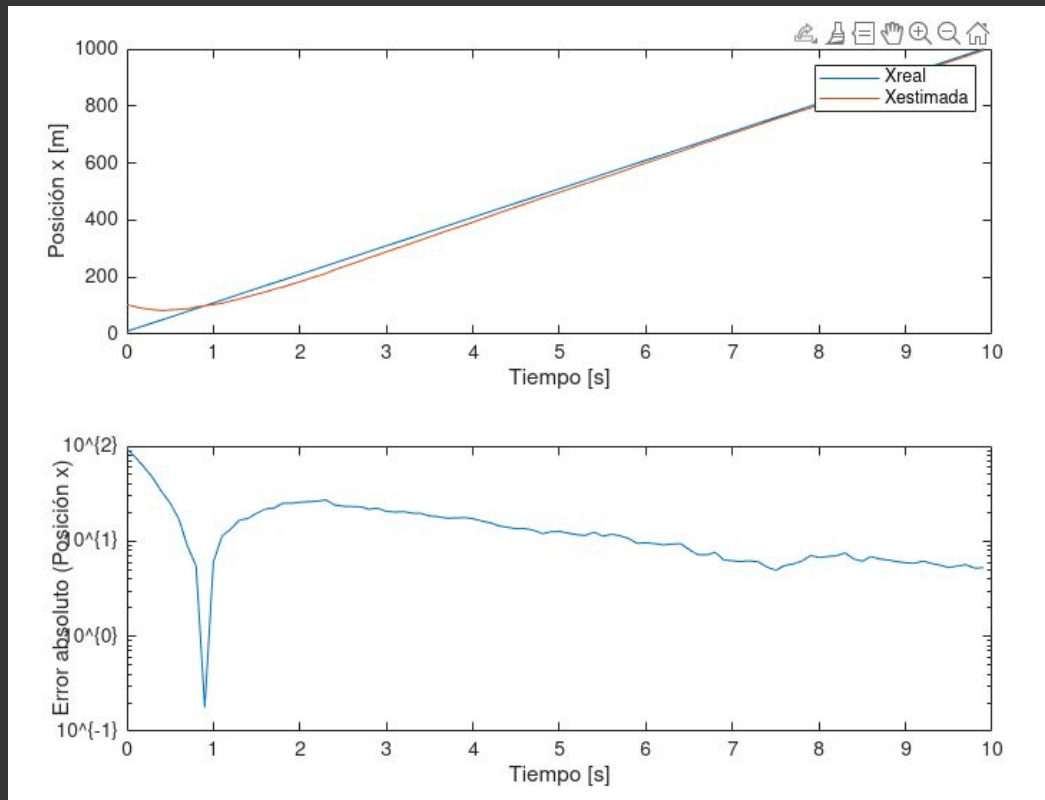
x\_real\_hist(:, k) = x\_real;

innovation\_hist(:, k) = y\_k; % Guardar la innovación

end

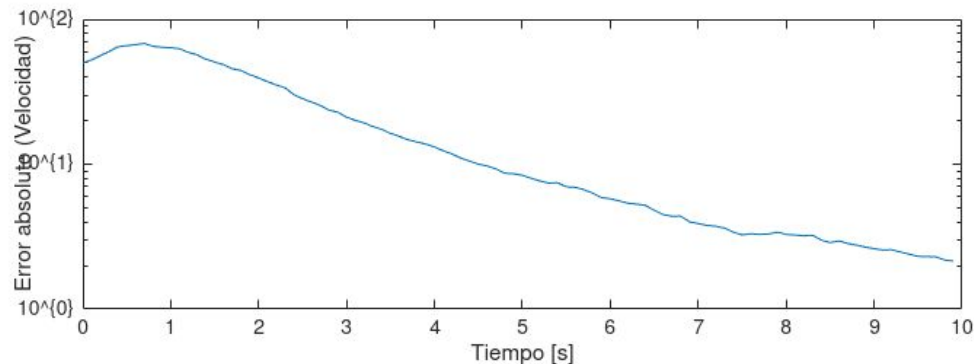
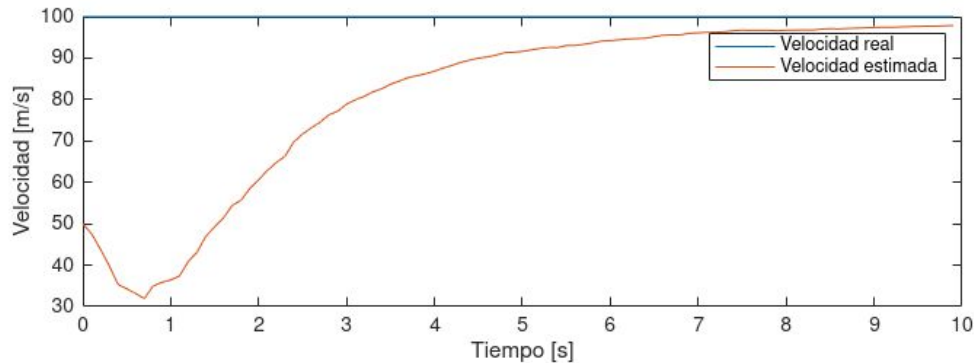
# Resultados (EKF). Posición en x

Gráfica que mide la posición horizontal real respecto a la estimada y su error correspondiente



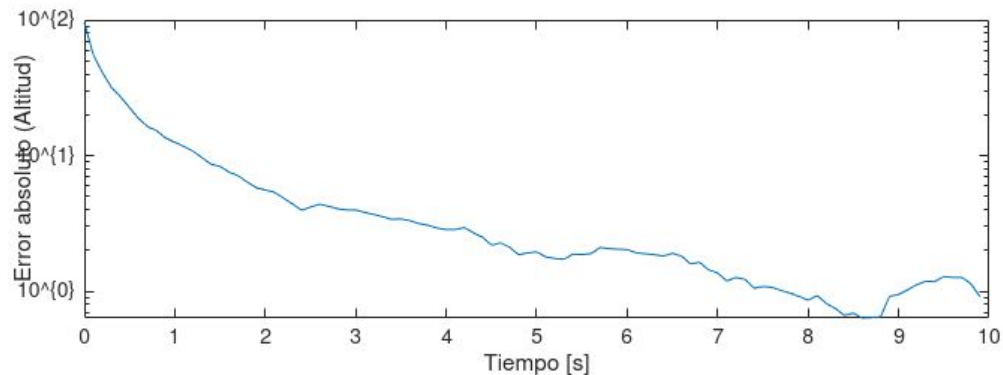
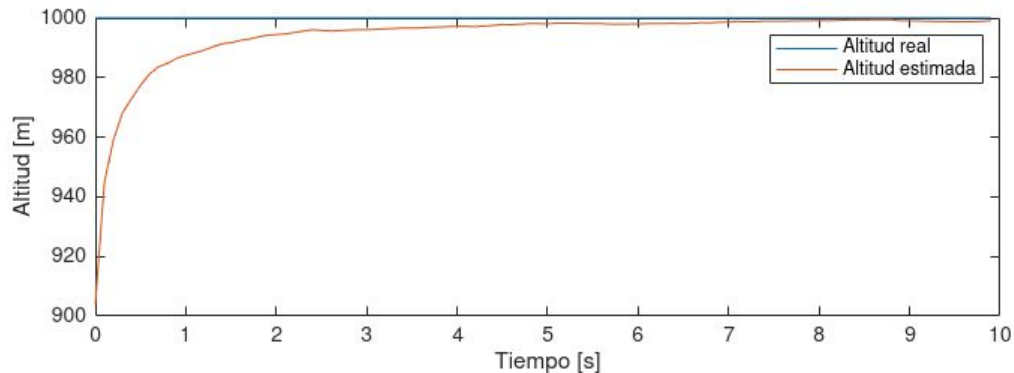
# Resultados (EKF). Velocidad

Gráfica que mide la velocidad real respecto a la estimada y su error correspondiente



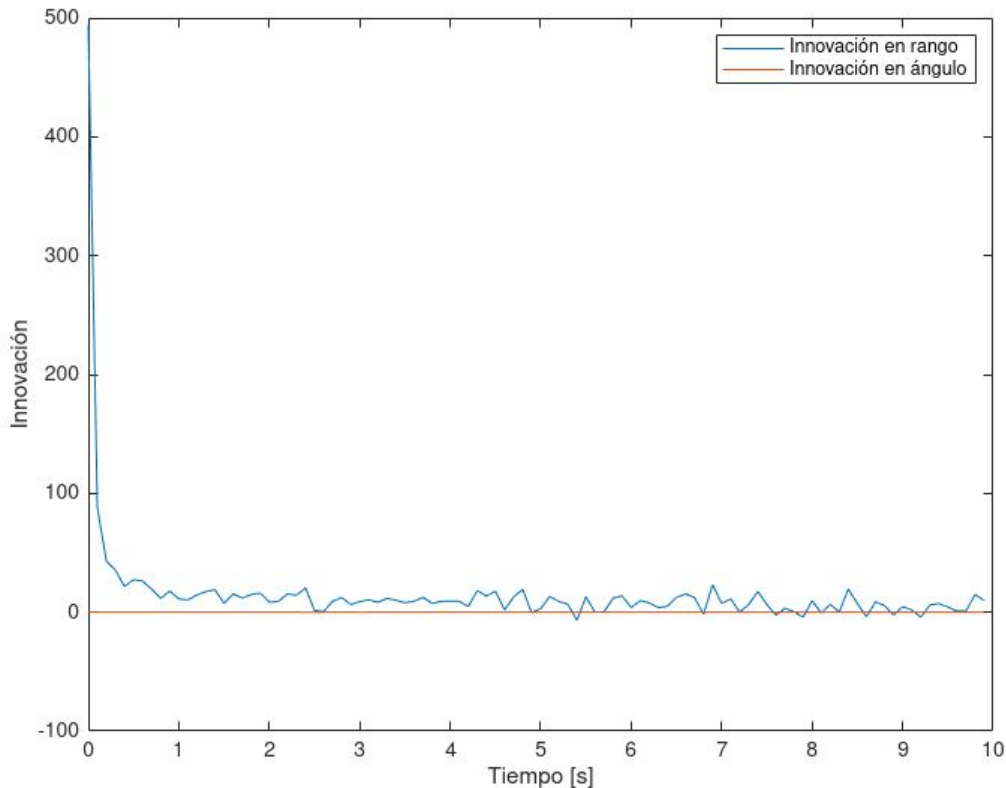
# Resultados (EKF). Posición en y

Gráfica que mide la posición vertical real respecto a la estimada y su error correspondiente



# Resultados (EKF). Innovación

Gráfica que mide la innovación en el rango respecto a la del ángulo



# Resultados (EKF). Varianza

Gráfica que mide la varianza en la posición horizontal, vertical, y en la velocidad

