Procesamiento de Señales I
TP2: Estimación de parámetros utilizando LS

Sampayo, Sebastián Lucas
Padrón: 93793

Primer Cuatrimestre de 2015

FACULTAD DE INGENIERIA
Universidad de Buenos Aires

Índice

1. Objetivos

Se desea estimar los errores que afectan a un instrumento de medición, más precisamente, dos acelerómetros. Para esto, se cuenta con datos obtenidos de un ensayo al cual se sometieron los acelerómetros. Conociendo la clase de errores que afectan a los acelerómetros, se deberá deducir cómo, a partir del ensayo realizado, estimar los valores de estos errores.

2. Ejercicios

2.1. Ejercicio 1

Indique cómo haría para, a partir de los datos del ensayo, armar un modelo ($b = Hc + v$) que le permita estimar los sesgos y factores de escala para cada uno de los acelerómetros.

En base a la primera prueba de laboratorio, se sabe que la aceleración medida responde a la siguiente fórmula que la relaciona con la aceleración real, para cada eje:

$$A_{medida} = A_{real} + (Error_{escala}) \cdot A_{real} + Error_{sesgo} + Ruido$$

Reordenando:

$$A_{medida} = A_{real} \cdot (1 + Error_{escala}) + Error_{sesgo} + Ruido \tag{1}$$

En el segundo ensayo realizado se obtuvo una serie de puntos correspondientes al ángulo rotado y a la medición de los acelerómetros. Por lo tanto es posible escribir una ecuación por cada punto, formando un sistema de ecuaciones. Este sistema se puede escribir matricialmente de la siguiente manera:

$$\underbrace{\begin{bmatrix} A_{medida}(0) \\ A_{medida}(1) \\ \vdots \\ A_{medida}(N-1) \end{bmatrix}}_{\triangleq \underline{b}} = \underbrace{\begin{bmatrix} A_{real}(0) & 1 \\ A_{real}(1) & 1 \\ \vdots & \\ A_{real}(N-1) & 1 \end{bmatrix}}_{\triangleq H} \cdot \underbrace{\begin{bmatrix} 1 + Error_{escala} \\ Error_{sesgo} \end{bmatrix}}_{\triangleq \underline{c}} + \underbrace{\begin{bmatrix} Ruido(0) \\ Ruido(1) \\ \vdots \\ Ruido(N-1) \end{bmatrix}}_{\triangleq \underline{v}} \tag{2}$$

Donde A_{real} se relaciona con el ángulo θ con la siguiente descomposición vectorial:

$$A_{real\ x}(i) = -g \cdot \sin(\theta(i))$$

$$A_{real\ y}(i) = -g \cdot \cos(\theta(i))$$

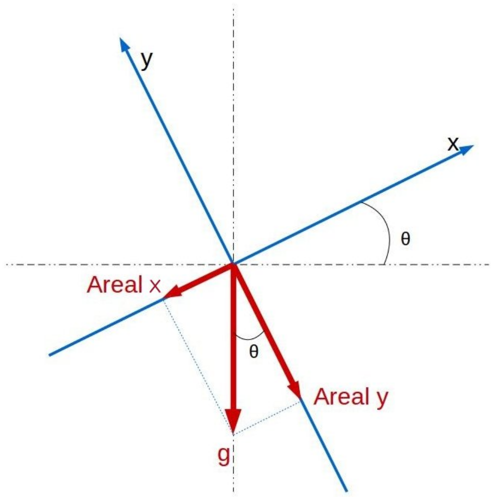

Figura 1: Descomposición vectorial de la aceleración medida

Este sistema corresponde a un modelo lineal donde $\underline{c}$ es una variable determinística pero desconocida y $\underline{v}$ es aleatoria, en particular, ruido blanco gaussiano, con matriz de correlación $R_v = \mathbb{E}[vv^*]$; y se conoce una realización de $\underline{b}$.

$$\underline{b} = H \cdot \underline{c} + \underline{v}$$

En estas condiciones, no existe solución a $\underline{c}$ ya que a pesar de conocer una realización de $\underline{b}$, este vector no pertenece a col(H) y se desconoce el valor de la realización de $\underline{v}$. Por lo tanto lo mejor que se puede hacer, en términos de minimizar el error cuadrático medio, es usar el estimador insesgado de mínima varianza (a.k.a. MVUE, o estimador de Gauss-Markov), que no es otra cosa que la solución por cuadrados mínimos ponderados (WLS) al sistema:

$$\underline{b} \approx H \cdot \underline{c}$$

utilizando $W = R_v^{-1}$ como matriz de producto interno. En otras palabras, este método utiliza la información del ruido para minimizar la diferencia: $\|\underline{b} - H \cdot \underline{c}\|_W^2$

La fórmula de este estimador queda entonces:

$$\hat{\underline{c}} = \left(H^* R_v^{-1} H\right)^{-1} H^* R_v^{-1} \cdot \underline{b}$$

con $(\cdot)^*$ siendo el operador 'transpuesta+conjugada'. Teniendo en cuena que el ruido es blanco y gaussiano:

$$R_v = \mathbb{E}[vv^*] = \sigma_v^2 \cdot I$$

Donde σ_v es la varianza (escalar) del ruido. Entonces queda:

$$\hat{\underline{c}} = \left(H^* (\sigma_v^2)^{-1} H\right)^{-1} H^* (\sigma_v^2)^{-1} \cdot \underline{b}$$

$$\boxed{\hat{\underline{c}} = \left(H^* H\right)^{-1} H^* \cdot \underline{b}}$$

La matriz de covarianza de este estimador resulta ser:

$$\mathbb{E}\left[(\hat{\underline{c}} - \underline{c})(\hat{\underline{c}} - \underline{c})^*\right] = \left(H^* R_v^{-1} H\right)^{-1} \tag{3}$$

$$\boxed{\mathbb{E}\left[(\hat{\underline{c}} - \underline{c})(\hat{\underline{c}} - \underline{c})^*\right] = \sigma_v^2 \left(H^* H\right)^{-1}} \tag{4}$$

donde la componente (1,1) de esta matriz corresponde a la varianza de $\hat{\underline{c}}(1)$ -que es $(1 + \widehat{Error}_{escala})$-; y la componente (2,2) corresponde a la varianza de $\hat{\underline{c}}(2)$ -que es $\widehat{Error}_{sesgo}$-.

Reemplazando con las definiciones de más arriba se puede ver que:

$$H^* H = \begin{bmatrix} A_{real}(0) & A_{real}(1) & \cdots & A_{real}(N-1) \\ 1 & 1 & \cdots & 1 \end{bmatrix} \cdot \begin{bmatrix} A_{real}(0) & 1 \\ A_{real}(1) & 1 \\ \vdots & \\ A_{real}(N-1) & 1 \end{bmatrix}$$

$$= \begin{bmatrix} \displaystyle\sum_{i=0}^{N-1} A_{real}^2(i) & \displaystyle\sum_{i=0}^{N-1} A_{real}(i) \\ \displaystyle\sum_{i=0}^{N-1} A_{real}(i) & \displaystyle\sum_{i=0}^{N-1} 1 \end{bmatrix}$$

Tomando el caso del eje x y reemplazando se tiene:

$$H_x^* H_x = \begin{bmatrix} \displaystyle\sum_{i=0}^{N-1} g^2 \cdot \sin(\theta(i))^2 & \displaystyle\sum_{i=0}^{N-1} -g \cdot \sin(\theta(i)) \\ \displaystyle\sum_{i=0}^{N-1} -g \cdot \sin(\theta(i)) & \displaystyle\sum_{i=0}^{N-1} 1 \end{bmatrix}$$

Y como se puede ver en el archivo 'ensayo.mat', θ barre linealmente N puntos entre 0 y 2π, es decir:

$$\theta(i) = \frac{2\pi i}{N},\ i = 0, ..., N-1$$

Sabiendo que:

$$\sum_{i=0}^{N-1} \sin\left(\frac{2\pi i}{N}\right)^2 = \frac{N}{2}$$

$$\sum_{i=0}^{N-1} \sin\left(\frac{2\pi i}{N}\right) = 0$$

Por lo tanto queda:

$$H_x^* H_x = \begin{bmatrix} g^2 \cdot \dfrac{N}{2} & 0 \\ 0 & N \end{bmatrix}$$

$$(H_x^* H_x)^{-1} = \begin{bmatrix} \dfrac{2}{g^2 N} & 0 \\ 0 & \dfrac{1}{N} \end{bmatrix}$$

Y de la ecuación 4, para cada eje se tiene:

$$\mathbb{E}\left[(\underline{\hat{c}} - \underline{c})(\underline{\hat{c}} - \underline{c})^*\right] = \begin{bmatrix} \dfrac{2\sigma_v^2}{g^2 N} & 0 \\ 0 & \dfrac{\sigma_v^2}{N} \end{bmatrix}$$

Con lo cual, para cada eje se tiene::

$$\mathbb{E}\left[((1 + \widehat{Error}_{escala}) - (1 + Error_{escala}))^2\right] = \boxed{\sigma_{Error_{escala}} = \frac{2\sigma_v^2}{g^2 N}} \tag{5}$$

$$\mathbb{E}\left[(\widehat{Error}_{sesgo} - Error_{sesgo})^2\right] = \boxed{\sigma_{Error_{sesgo}} = \frac{\sigma_v^2}{N}} \tag{6}$$

2.2. Ejercicio 2

Estime los valores de los sesgos y factores de escala, a partir de los datos del ensayo (archivo: ensayo.mat) que se le suministraron. Calcule la varianza del estimador.

Para este ejercicio se escribió un programa de MATLAB cuyo código se encuentra en 'tp2.m' y utiliza la función del archivo 'mvue.m'. En esta aplicación simplemente se implementan las fórmulas obtenidas en la teoría a partir de los datos del ensayo en 'ensayo.mat'. Los resultados obtenidos se presentan a continuación:

$\widehat{Error}_{escala\ x}$	-0.0301
$\widehat{Error}_{escala\ y}$	0.0100
$\widehat{Error}_{sesgo\ x}$	0.0767
$\widehat{Error}_{sesgo\ y}$	-0.0175
$\sigma^2_{Error_{escala\ x}}$	2.6031e-07
$\sigma^2_{Error_{escala\ y}}$	6.6632e-07
$\sigma^2_{Error_{sesgo\ x}}$	1.2499e-05
$\sigma^2_{Error_{sesgo\ x}}$	3.1998e-05

Cuadro 1: Resultados de la ejecución de 'tp2.m'

2.3. Ejercicio 3

Calcule la trayectoria del vehículo. De las posiciones de los cuatro puntos suministrados A,B,C o D (archivo: puntos.mat) ¿A cuál de ellos llega el vehículo?.

En este caso se tomaron los datos de las aceleraciones medidas del archivo 'acel.mat' y luego se integró 2 veces para obtener la posición del vehículo.
Por otro lado, con la técnica de cuadrados mínimos utilizada, se pasó de la ecuación 1 a:

$$A_{medida} = A_{real} \cdot \left(1 + \widehat{Error}_{escala}\right) + \widehat{Error}_{sesgo} \tag{7}$$

De esta forma, se despejó la aceleración real para obtener el estimador:

$$\boxed{\hat{A}_{real} = \frac{A_{medida} - \widehat{Error}_{sesgo}}{\left(1 + \widehat{Error}_{escala}\right)}} \tag{8}$$

para cada eje. En el siguiente gráfico puede verse la trayectoria calculada al integrar la aceleración medida, sin realizar ninguna estimación ('Posición medida'), y la trayectoria obtenida a partir de la estimación de la aceleración real ('Posición real estimada'), así como también los 4 puntos suministrados. Se puede ver que el vehículo pasa por el punto A y finaliza cerca del punto C.

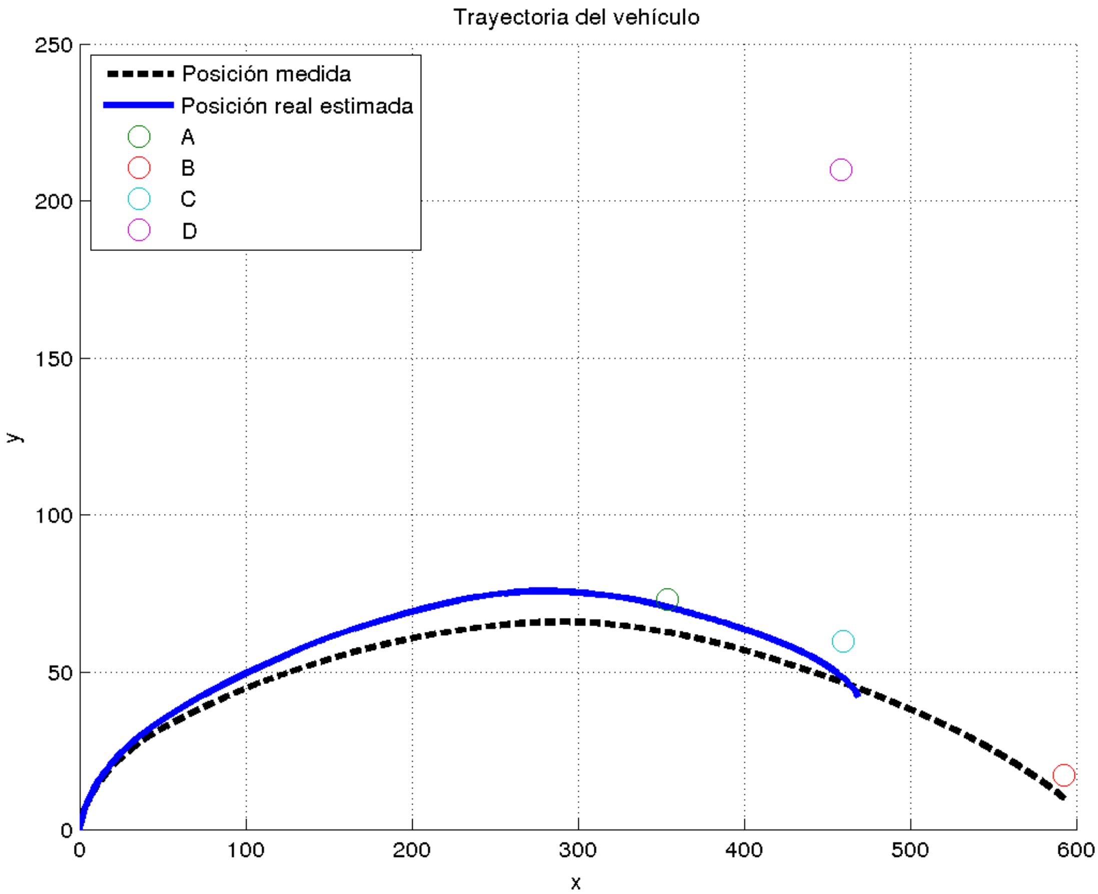

Figura 2: Comparación de trayectorias.

2.4. Ejercicio 4

Suponga que, para resolver el item 2, por limitaciones en la capacidad de cómputo, no puede resolver el problema de estimación LS, utilizando todos los datos del ensayo. Indique cuál sería la cantidad mínima de muestras del experimento (N), que podría usar para distinguir si el vehículo llega al punto A,B,C o D. Si bien no es posible garantizar que para un cierto N, se puede determinar A, B, C o D (¿por qué?), explique (en forma clara y sucinta) cuál es su razonamiento y consideraciones que haya tenido en cuenta.

Una forma de abordar este problema es considerar el caso límite para el cual el error debido a la estimación permite distinguir en que punto se encuentra el vehículo. Esto se puede ver planteando que el error 'radial' en cada instante sea menor a la mitad de la distancia mínima entre dos puntos A, B, C y D.

Tomando las ecuaciones 7 y 8 se puede deducir que el error del estimador de la aceleración real es para el eje x (análogo para el eje y):

$$\Delta \hat{A}_{real\ x}(i) = \hat{A}_{real\ x}(i) \cdot \sigma_{Error_{escala\ x}} + \sigma_{Error_{sesgo\ x}}$$

Este valor se puede acotar al máximo valor absoluto que alcanza $\hat{A}_{real\ x}(i)$, que es 3,3

$$\Delta \hat{A}_{real\ x} \leq \text{máx}\{\hat{A}_{real\ x}(i)\} \cdot \sigma_{Error_{escala\ x}} + \sigma_{Error_{sesgo\ x}}$$

$$\Delta \hat{A}_{real\ x} \leq 3{,}3 \cdot \sigma_{Error_{escala\ x}} + \sigma_{Error_{sesgo\ x}}$$

Luego, para obtener el error en la posición, se integra este valor 2 veces, que al ser constante en el tiempo queda simplemente:

$$\Delta \hat{P}_{real\ x} \leq \left(3{,}3 \cdot \sigma_{Error_{escala\ x}} + \sigma_{Error_{sesgo\ x}}\right) \cdot \frac{T^2}{2} \tag{9}$$

con T el intervalo temporal de integración, que en este caso es de 60 segundos.

De esta manera, el error radial resulta:

$$\Delta \hat{P}_{real} = \sqrt{\left(\Delta^2 \hat{P}_{real\ x}\right)^2 + \left(\Delta^2 \hat{P}_{real\ y}\right)^2} \leq \frac{d_{min}}{2} \tag{10}$$

Manipulando las ecuaciones 5, 6, 9 y 10 se obtiene:

$$\boxed{N \geq \frac{T^4}{d_{min}^2} \left(A_{MAX} \frac{\sqrt{2}}{g} + 1\right)^2 \left(\sigma_{v\ x}^2 + \sigma_{v\ y}^2\right)} \tag{11}$$

Con los valores del ensayo el resultado es $\boxed{N = 2196}$.

3. Código

3.1. Script principal

```matlab
%------------------------------------------------------------------------------%
%Facultad de Ingeniería de la Universidad de Buenos Aires
%Procesamiento de Señales I
%Trabajo Práctico 2:
%   - Estimación de parámetros utilizando LS -
%1° Cuatrimestre de 2015
%
%Sampayo, Sebastián Lucas
%Padrón: 93793
%e-mail: sebisampayo@gmail.com
%
%Script principal - MATLAB
%------------------------------------------------------------------------------%

close all;
clear all;

%------------------------------------------------------------------------------%
%----- Parámetros iniciales -----
%------------------------------------------------------------------------------%
%## TEST ## (Para usar datos generados con "generar_datos_test.m")
test = true; %false: Sin test, true: Con test
if (test)
    generar_datos_test
end
%##
%Nombre del archivo de datos del ensayo realizado con los acelerómetros
%Contiene:
%   tita:   los valores del ángulo rotado ($\theta$). Nx1
%   datos:  los valores de aceleraciones medidas en x e y, respectivamente,
%           correspondientes a cada ángulo ($\theta$). Nx2
test_file_name = 'ensayo.mat';
%## TEST ##
if (test)
    test_file_name = 'ensayo_test.mat';
end
%##
load(test_file_name);

%Nombre del archivo de los puntos A, B, C y D
%Contiene:
%   A, B, C y D.
points_file_name = 'puntos.mat';
load(points_file_name);

%Nombre del archivo de la aceleración medida durante la trayectoria del vehículo
%Contiene:
%   t:      tiempo en segundos
%   Aerr:   la aceleración en los ejes x e y, respectivamente.
acel_file_name = 'acel.mat';
%## TEST ##
```

```
if (test)
    acel_file_name = 'acel_test.mat';
end
%##
load(acel_file_name);

% Posición inicial
initial_position = [0, 0];
%## TEST ##
if (test)
    load('initial_test.mat');
    initial_position = P0;
end
%##

% Velocidad inicial
initial_velocity = [1, 3];
%## TEST ##
if (test)
    initial_velocity = V0;
end
%##

% Varianza del ruido de los acelerómetros
acel_variance = [0.25, 0.64];

% Constante universal de aceleración de la gravedad
g = 9.8; % [m/s^2]
%----------------------------------------------------------------------------%


%----------------------------------------------------------------------------%
%----- Ejercicio 1) -----
%----------------------------------------------------------------------------%
% Indique cómo haría para, a partir de los datos del ensayo, armar un modelo
% (y = Ax + ) que le permita estimar los sesgos y factores de escala para
% cada uno de los acelerómetros.

% y = Ax +
% b = Hc + v
% b := Aceleración medida, Nx1
% H := Matriz de 2 columnas.
%         Columna 1: Aceleración Real. Columna 2: todos 1's.
% c := Vector columna de 2 componentes.
%         Componente 1: (1+Error de Escala). Componente 2: Error de sesgo
% v := Ruido, Nx1.



%----------------------------------------------------------------------------%
%----- Ejercicio 2) -----
%----------------------------------------------------------------------------%
% Estime los valores de los sesgos y factores de escala, a partir de los datos
% del ensayo (archivo: ensayo.mat) que se le suministraron. Calcule la varianza
% del estimador.
```

```
% [x_hat, cov_x_hat] = mvue(y, H, Rv)

% Cantidad de muestras:
N = length(tita);
% Se asume Ruido Blanco Gaussiano, por lo tanto, la matriz de correlación es:
% Teóricamente la matriz identidad multiplicada por la varianza del ruido.
% Sin embargo, como esto utiliza demasiada memoria innecesariamente, lo reduzco
% a un escalar igual a la varianza. Todo da igual, ver las fórmulas.
Rv_x = acel_variance(1); %* diag(ones(N,1));
Rv_y = acel_variance(2); %* diag(ones(N,1));

H = [-g*sin(tita), ones(N,1)];
[c_hat_x, cov_c_hat_x] = mvue(datos(:,1), H, Rv_x);
clear H;

H = [-g*cos(tita), ones(N,1)];
[c_hat_y, cov_c_hat_y] = mvue(datos(:,2), H, Rv_y);
clear H;

% Pasaje de variables de errores obtenidos:
scale_error_x = c_hat_x(1) - 1
scale_error_y = c_hat_y(1) - 1
bias_error_x = c_hat_x(2)
bias_error_y = c_hat_y(2)

scale_error_x_variance = cov_c_hat_x(1,1)
scale_error_y_variance = cov_c_hat_y(1,1)
bias_error_x_variance = cov_c_hat_x(2,2)
bias_error_y_variance = cov_c_hat_y(2,2)
%-----------------------------------------------------------------------------%


%-----------------------------------------------------------------------------%
%----- Ejercicio 3) -----
%-----------------------------------------------------------------------------%
% Calcule la trayectoria del vehículo. De las posiciones de los cuatro puntos
% suministrados A,B,C o D (archivo: puntos.mat) ¿a cuál de ellos llega el
% vehículo?.

% Estimador de Aceleración real:
A_real_x = (Aerr(:,1) - bias_error_x)/(1 + scale_error_x);
A_real_y = (Aerr(:,2) - bias_error_y)/(1 + scale_error_y);

% Calculo la velocidad y la posición a partir de la aceleración medida
% aproximando las integrales con el método del trapecio.
T_step = t(2)-t(1);
integrate = @(x) (T_step*cumtrapz(x));
velocity_x_measured = initial_velocity(1) + integrate(Aerr(:,1));
velocity_y_measured = initial_velocity(2) + integrate(Aerr(:,2));
position_x_measured = initial_position(1) + integrate(velocity_x_measured);
position_y_measured = initial_position(2) + integrate(velocity_y_measured);

% Calculo la velocidad y la posición a partir del estimador de aceleración real
velocity_x_real = initial_velocity(1) + integrate(A_real_x);
velocity_y_real = initial_velocity(2) + integrate(A_real_y);
position_x_real = initial_position(1) + integrate(velocity_x_real);
```

```
position_y_real = initial_position(2) + integrate(velocity_y_real);

figure
hold all
plot(position_x_measured, position_y_measured, 'k--', 'Linewidth', 3)
plot(position_x_real, position_y_real, 'Linewidth', 3)
if (test == false)
    plot(A(1), A(2), 'o', 'MarkerSize', 10)
    plot(B(1), B(2), 'o', 'MarkerSize', 10)
    plot(C(1), C(2), 'o', 'MarkerSize', 10)
    plot(D(1), D(2), 'o', 'MarkerSize', 10)
end
legend('Posición medida', 'Posición real estimada', 'A', 'B', 'C', 'D',...
    'Location', 'NorthWest');
title('Trayectoria del vehículo');
grid on;
xlabel('x');
ylabel('y');

% print('-dpng', 'trayectoria.png');

%## TEST ##
if (test)
    load('pos_real_test.mat');
    plot(Px, Py, 'r--')
%     axis([0 max(Px) min(Py) max(Py)]);
%     print('-dpng', 'trayectoria_test.png');
%     load('acel_real_test.mat');
%     figure
%     hold all
%     plot(Ax)
%     plot(A_real_x)
%     plot(Aerr(:,1))
%     legend('Original', 'Estimada');
%
%     figure
%     hold all
%     plot(Ay)
%     plot(A_real_y)
%     plot(Aerr(:,2))
%     legend('Original', 'Estimada');
end
%##
```

3.2. MVUE

```
%---------------------------------------------------------------------------------%
%Facultad de Ingeniería de la Universidad de Buenos Aires
%Procesamiento de Señales I
%Trabajo Práctico 2:
%   - Estimación de parámetros utilizando LS -
%1° Cuatrimestre de 2015
%
%Sampayo, Sebastián Lucas
%Padrón: 93793
%e-mail: sebisampayo@gmail.com
```

```
%
% Función MVUE (Minimum Variance Unbiased Estimator) - MATLAB
%--------------------------------------------------------------------%
%
%% Función MVUE (Minimum Variance Unbiased Estimator)
%
%  Calcula el estimador insesgado de mínima varianza para un modelo lineal:
%
%       $$  y = Hx + v  $$
%
%  donde 'x' es determinístico pero desconocido mientras que 'v' es ruido
%  aleatorio, y H es conocida.
%
%  Uso:
%    [x_hat, cov_x_hat] = mvue(y, H, Rv)
%
%  donde:
%    y:          es un vector de Nx1
%    H:          es una matriz de NxP
%    Rv:         es la matriz de correlación del ruido 'v', de NxN
%    x_hat:      es el estimador de 'x', de Px1
%    cov_x_hat:  es la matriz de covarianza del estimador 'x_hat', de PxP
%
%  Fórmulas utilizadas:
%
%    $$  x
%    $$  cov_x
%    $$  Rv

function [x_hat, cov_x_hat] = mvue(y, H, Rv)
  x_hat = inv(H'*inv(Rv)*H) * H'*inv(Rv)*y;
  cov_x_hat = inv(H'*inv(Rv)*H);
end
```

3.3. Script adicional

Adicionalmente se codificó una prueba con datos generados a partir del modelo propuesto.

```
%--------------------------------------------------------------------%
% Facultad de Ingeniería de la Universidad de Buenos Aires
% Procesamiento de Señales I
% Trabajo Práctico 2:
%   - Estimación de parámetros utilizando LS -
% 1° Cuatrimestre de 2015
%
% Sampayo, Sebastián Lucas
% Padrón: 93793
% e-mail: sebisampayo@gmail.com
%
% Script para generar datos de test - MATLAB
%--------------------------------------------------------------------%

% close all;
% clear all;


%--------------------------------------------------------------------%
```

```
% Posición inicial
initial_position = [0, 0];

% Velocidad inicial
initial_velocity = [1, 3];

% Varianza del ruido de los acelerómetros
acel_variance = [0.25, 0.64];

% Constante universal de aceleración de la gravedad
g = 9.8; % [m/s^2]
%---------------------------------------------------------------------%
Ekx = -0.0301;
Eky = 0.01;
Esx = 0.0767;
Esy = -0.0175;


%---------------------------------------------------------------------%
%--- Ensayo de acelerómetros, para estimar los errores de escala y sesgo --%
%---------------------------------------------------------------------%

N = 20001;
tita = linspace(0, 2*pi, N)';

A_real_x = -g*sin(tita);
A_real_y = -g*cos(tita);
Vx = normrnd(0, acel_variance(1), [N, 1]);
Vy = normrnd(0, acel_variance(2), [N, 1]);
datos(:,1) = A_real_x * (1 + Ekx) + Esx + Vx;
datos(:,2) = A_real_y * (1 + Eky) + Esy + Vy;

save('ensayo_test.mat', 'tita', 'datos');


%---------------------------------------------------------------------%
%---- Ensayo de trayectoria ----
%---------------------------------------------------------------------%

N = 6000;
t = linspace(0, 60, N)';
T = t(2) - t(1);

%Px = linspace(0,10, N)';
Px = t;
%Py = -Px.*(Px-10);
Py = Px.*(Px-10).*(Px-30).*(Px-60);
%Py = -t.*(t-60);

Vex = diff(Px)/T;
Vey = diff(Py)/T;

Ax = diff(Vex)/T;
Ay = diff(Vey)/T;

N = length(Ax);
```

```
Vx = normrnd(0, acel_variance(1), [N, 1]);
Vy = normrnd(0, acel_variance(2), [N, 1]);

 Aerr(:,1) = Ax * (1 + Ekx) + Esx + Vx;
 Aerr(:,2) = Ay * (1 + Eky) + Esy + Vy;

P0 = [Px(1), Py(1)];

V0 = [Vex(1), Vey(1)];

save('acel_test.mat', 't', 'Aerr');
save('pos_real_test.mat', 'Px', 'Py');
save('initial_test.mat', 'P0', 'V0');
save('acel_real_test.mat', 'Ax', 'Ay');
```

Resultado:

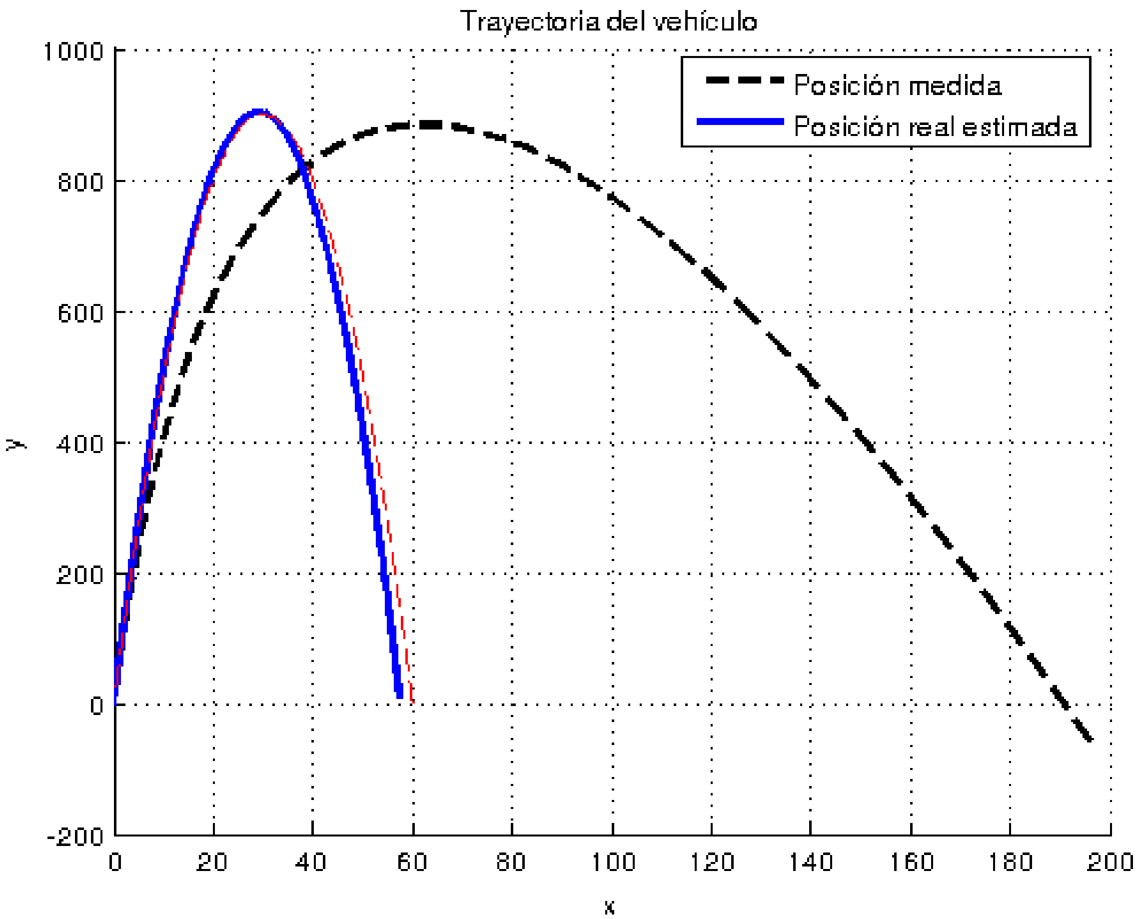

Figura 3: Comparación de trayectorias

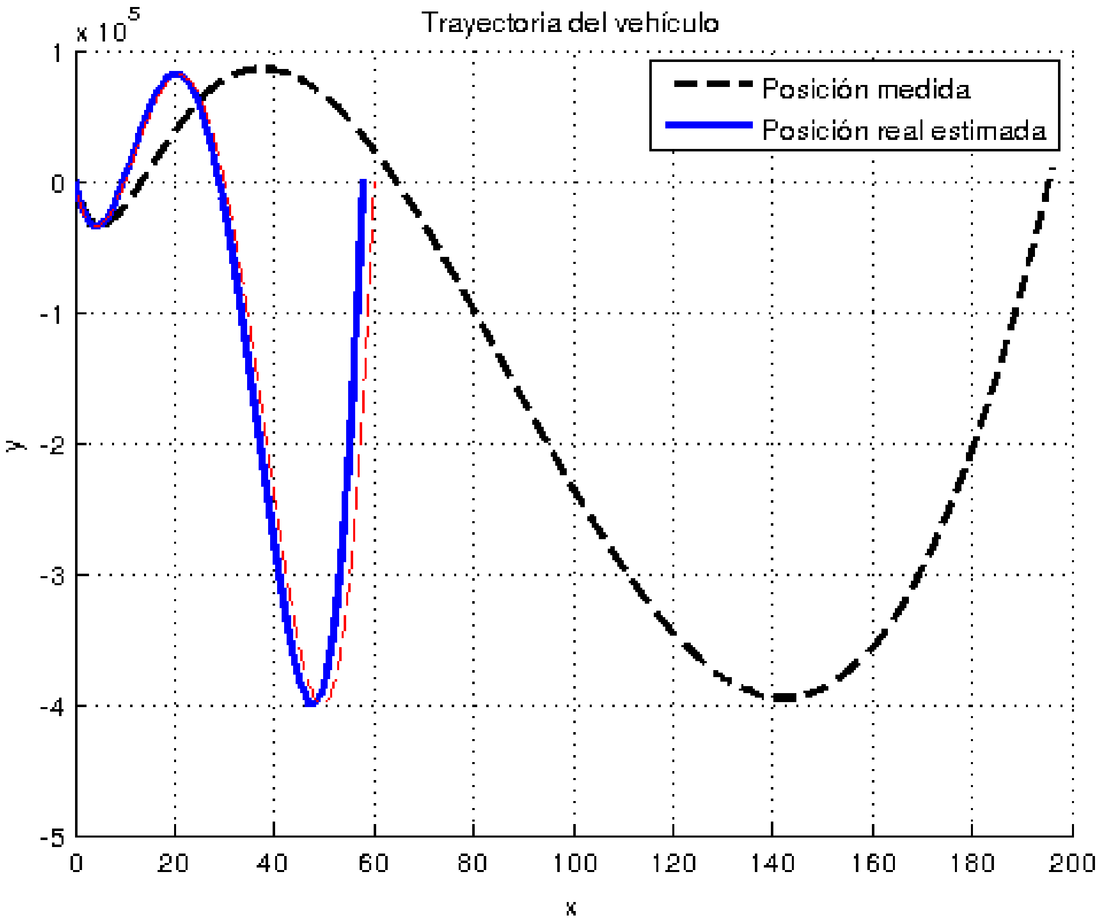

Figura 4: Comparación de trayectorias

```
scale_error_x = -0.0299

scale_error_y = 0.0112

bias_error_x = 0.0746

bias_error_y = -0.0172

scale_error_x_variance = 2.6031e-07

scale_error_y_variance = 6.6632e-07

bias_error_x_variance = 1.2499e-05

bias_error_y_variance = 3.1998e-05
```