SET MEMBERSHIP ESTIMATION THEORY

Exercise: parametric estimation of a position transducer model

Michele TARAGNA

Dipartimento di Automatica e Informatica
Politecnico di Torino

III level Course 01LCPIU
"Experimental modeling: model building from experimental data"

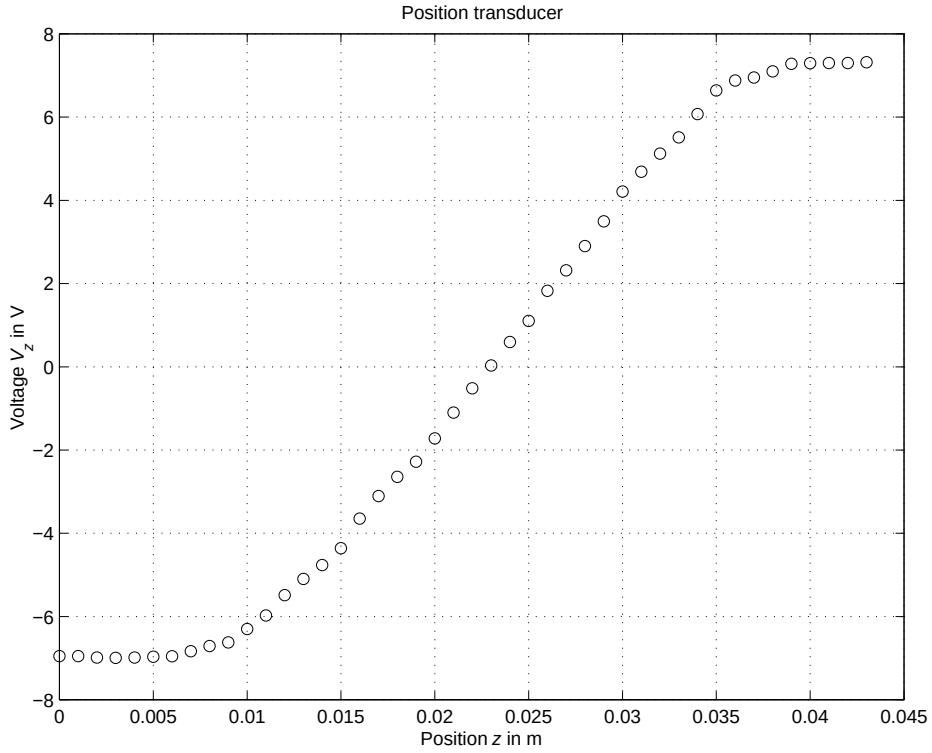
Exercise #1: parametric estimation of a position transducer model using the classical (or statistical) approach

1. System description

The position transducer here considered makes use of an optical sensor build of 2 groups of photodiodes and it is used to evaluate the vertical movements of a metallic body inside a magnetic levitator. It is realized in a differential configuration, in order to discriminate the vertical movements with respect to the lateral ones.

This transducer can be considered as static, since its reply speed, i.e., its time constants are some order of magnitude greater than those of the other parts of the system.

In the figure below, the steady-state output voltage V_z of the transducer (measured in volt) is plotted as a function of the body position z (measured in meters simply by sight). Such a static characteristic becomes much more nonlinear near the boundaries of the transducer action field.



2. Linear approximation of the position-voltage characteristic

The position-voltage characteristic of the transducer shows a nearly linear behaviour in the interval between 1 and 3.5 cm. In such a linearity interval, the characteristic can be described by the following model

$$V_z = K_t \cdot z + V_o$$

where the gain K_t and the offset voltage V_o are unknown constant parameters to be estimated. Since the most relevant error is due to the position measurement (the greatest position error is about 0.5 mm), the model equation can be more suitably rewritten in the following form, where the measurement error e is explicitly taken into account:

$$z = \frac{1}{K_t} \cdot V_z - \frac{V_o}{K_t} + e$$

By considering the N measurements collected in the linearity interval, the following system of linear equations is derived:

$$\begin{aligned} z_1 &= \frac{1}{K_t} \cdot V_{z,1} - \frac{V_o}{K_t} + e_1 \\ z_2 &= \frac{1}{K_t} \cdot V_{z,2} - \frac{V_o}{K_t} + e_2 \\ &\vdots \\ z_N &= \frac{1}{K_t} \cdot V_{z,N} - \frac{V_o}{K_t} + e_N \end{aligned}$$

where $V_{z,i}$ is the voltage provided by the transducer when the position value is z_i.

By introducing the unknown parameters:

$$\theta_1 = \frac{1}{K_t}, \quad \theta_2 = -\frac{V_o}{K_t}$$

the previous equations can be rewritten as:

$$\begin{aligned} z_1 &= V_{z,1} \cdot \theta_1 + \theta_2 + e_1 \\ z_2 &= V_{z,2} \cdot \theta_1 + \theta_2 + e_2 \\ &\vdots \\ z_N &= V_{z,N} \cdot \theta_1 + \theta_2 + e_N \end{aligned}$$

or, in matrix form:

$$\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_N \end{bmatrix} = \begin{bmatrix} V_{z,1} & 1 \\ V_{z,2} & 1 \\ \vdots & \vdots \\ V_{z,N} & 1 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix}$$

i.e., the estimation problem is recast in the standard form:

$$y = \Phi \cdot \theta + e$$

where $y \in \mathbb{R}^N$, $\Phi \in \mathbb{R}^{N \times 2}$, $\theta \in \mathbb{R}^2$ and $e \in \mathbb{R}^N$. Since the unknown is the vector θ, the problem to be solved is overdetermined, because the number of unknowns is smaller than the number of measurement equations taken into account and then, in general, the system of equations does not admit any solution, since the matrix Φ cannot be inverted.

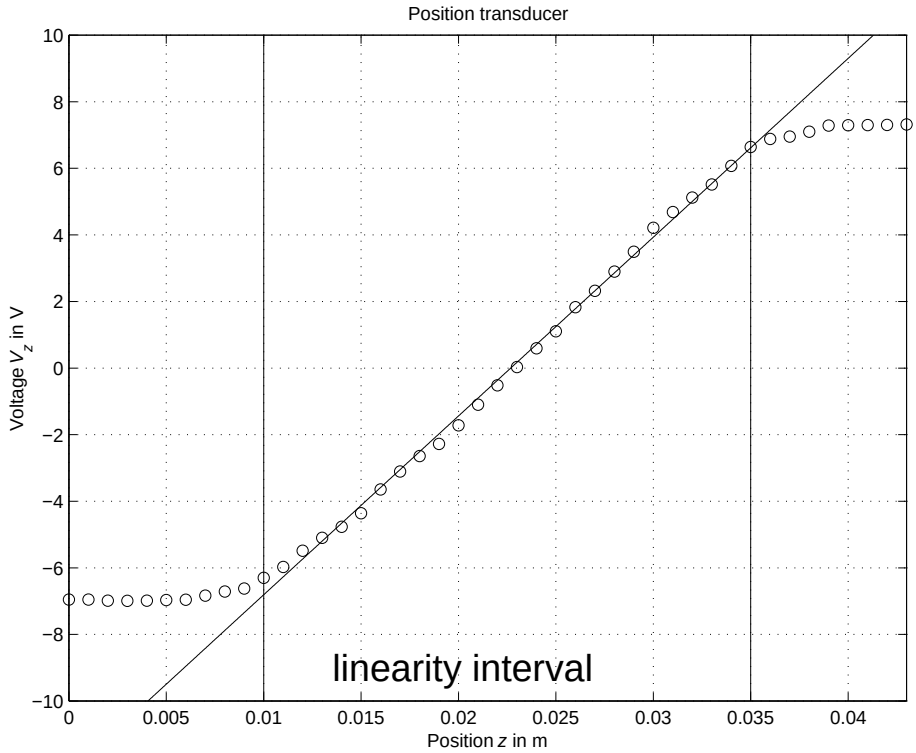
Using the least squares algorithm as estimation method:

$$\hat{\theta} = \left(\Phi^T \cdot \Phi\right)^{-1} \Phi^T \cdot y$$

it results that:

$$\hat{\theta} = \begin{bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \end{bmatrix} = \begin{bmatrix} 0.0019 \\ 0.0227 \end{bmatrix} \quad \Rightarrow \quad \hat{K}_t = \frac{1}{\hat{\theta}_1} = 537.0036\text{V/m}, \quad \hat{V}_o = -\frac{\hat{\theta}_2}{\hat{\theta}_1} = -12.1779\text{V}$$

Under MATLAB, the least squares solution is provided by the operator "\" or the command `mldivide`. Please take a look to the help-on-line by typing `help mldivide`.



3. Evaluation of the confidence intervals of the estimated parameters

a) Case of Gaussian i.i.d. noise with variance that is known or may be obtained from a priori information.

The main error source in the experimental measurements is the measurement of the position z and the greatest error is about 0.5 mm. Assuming that such an error has a normal distribution $\mathcal{N}(0, \Sigma_e)$, the covariance matrix Σ_e can be assumed to be equal to $\Sigma_e = \sigma_e^2 \cdot I_{N\times N}$, where the variance σ_e^2 can be derived according to the following considerations:

- the probability that the outcome of a random variable ξ with normal p.d.f. differs from the mean value $E[\xi]$ no more than k times the standard deviation σ_ξ is equal to:

$$P = P(|\xi - E[\xi]| \leq k \cdot \sigma_\xi) = 1 - \frac{2}{\sqrt{2\pi}} \int_k^\infty e^{-t^2/2} dt$$

and, in particular, it turns out that

k	P
1	68.3%
2	95.4%
3	99.7%

- assuming that the variation range of the measurement error $[-5 \cdot 10^{-4}\text{m}, 5 \cdot 10^{-4}\text{m}]$ corresponds to a sufficiently high-level confidence interval, for example equal to 95%, it turns out that:

$$5 \cdot 10^{-4} = 2 \cdot \sigma_e \quad \Rightarrow \quad \sigma_e = \frac{5 \cdot 10^{-4}}{2} = 2.5 \cdot 10^{-4}$$

The covariance matrix of the Gauss-Markov estimate $\hat{\theta}$ is given by:

$$\Sigma_{\hat{\theta}} = \left(\Phi^T \cdot \Sigma_e^{-1} \cdot \Phi\right)^{-1} = \sigma_e^2 \cdot \left(\Phi^T \cdot \Phi\right)^{-1} = \begin{bmatrix} \sigma^2_{\hat{\theta},11} & \sigma^2_{\hat{\theta},12} \\ \sigma^2_{\hat{\theta},12} & \sigma^2_{\hat{\theta},22} \end{bmatrix} = \begin{bmatrix} 0.0149 & 0.0014 \\ 0.0014 & 0.2405 \end{bmatrix} \cdot 10^{-8}$$

and then the standard deviations of the marginal p.d.f. are:

$$\begin{aligned}\sigma_{\hat{\theta}_1} &= \sqrt{\sigma^2_{\hat{\theta},11}} = 1.2187 \cdot 10^{-5} \\ \sigma_{\hat{\theta}_2} &= \sqrt{\sigma^2_{\hat{\theta},22}} = 4.9043 \cdot 10^{-5}\end{aligned}$$

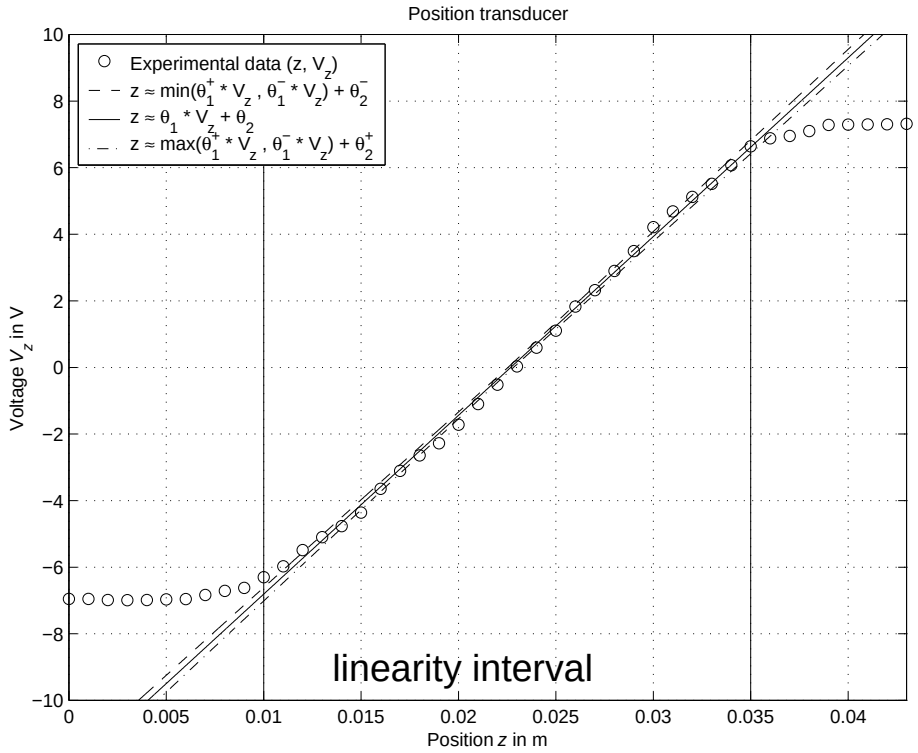
The confidence intervals of parameters $\hat{\theta}_i$, computed for example at 95%, are given by:

$$\begin{aligned}\left[\hat{\theta}_1^-, \hat{\theta}_1^+\right] &= \left[\hat{\theta}_1 - \delta_{\hat{\theta}_1}, \hat{\theta}_1 + \delta_{\hat{\theta}_1}\right] = [0.0018, 0.0019], \text{ with } \delta_{\hat{\theta}_1} = 2 \cdot \sigma_{\hat{\theta}_1} = 2.4374 \cdot 10^{-5} \\ \left[\hat{\theta}_2^-, \hat{\theta}_2^+\right] &= \left[\hat{\theta}_2 - \delta_{\hat{\theta}_2}, \hat{\theta}_2 + \delta_{\hat{\theta}_2}\right] = [0.0226, 0.0228], \text{ with } \delta_{\hat{\theta}_2} = 2 \cdot \sigma_{\hat{\theta}_2} = 9.8086 \cdot 10^{-5}\end{aligned}$$

and then the 95% confidence intervals of $\hat{K}_t$ and $\hat{V}_o$ are equal to:

$$\begin{aligned}\left[\hat{K}_t^-, \hat{K}_t^+\right] &= \left[\frac{1}{\hat{\theta}_1^+}, \frac{1}{\hat{\theta}_1^-}\right] = [530.1, 544.1]\,\text{V/m} \\ \left[\hat{V}_o^-, \hat{V}_o^+\right] &= \left[-\frac{\hat{\theta}_2^+}{\hat{\theta}_1^-}, -\frac{\hat{\theta}_2^-}{\hat{\theta}_1^+}\right] = [-12.39, -11.97]\,\text{V}\end{aligned}$$

In the figure below, the envelope of the static characteristics of models whose parameters $\hat{\theta}$ belong to 95% confidence intervals is plotted.



b) Case of Gaussian i.i.d. noise with unknown variance.

Assuming that $\Sigma_e = \hat{\sigma}_e^2 \cdot I_{N\times N}$, the maximum likelihood estimate of the noise standard deviation is given by:

$$\hat{\sigma}_e = \sqrt{\frac{1}{N}\left(y - \Phi \cdot \hat{\theta}\right)^T \cdot \left(y - \Phi \cdot \hat{\theta}\right)} = 3.5618 \cdot 10^{-4}$$

By proceeding as in the previous case, the 95% confidence intervals of $\hat{K}_t$ and $\hat{V}_o$ are derived and are equal to:

$$\begin{aligned}\left[\hat{K}_t^-, \hat{K}_t^+\right] &= [527.2, 547.2]\,\text{V/m} \\ \left[\hat{V}_o^-, \hat{V}_o^+\right] &= [-12.49, -11.88]\,\text{V}\end{aligned}$$

4. Polynomial approximations of position-voltage characteristic
The overall position-voltage characteristic can be well approximated by means of a polynomial of greater odd order, in order to model the saturation phenomenon occurring near the boundaries of the transducer action field..

For example, the following III order model can be used:

$$z = \theta_1 \cdot V_z^3 + \theta_2 \cdot V_z^2 + \theta_3 \cdot V_z + \theta_4 + e$$

By considering all the N available measurements, the following system of linear equations is derived:

$$\begin{aligned} z_1 &= \theta_1 \cdot V_{z,1}^3 + \theta_2 \cdot V_{z,1}^2 + \theta_3 \cdot V_{z,1} + \theta_4 + e_1 \\ z_2 &= \theta_1 \cdot V_{z,2}^3 + \theta_2 \cdot V_{z,2}^2 + \theta_3 \cdot V_{z,2} + \theta_4 + e_2 \\ &\vdots \\ z_N &= \theta_1 \cdot V_{z,N}^3 + \theta_2 \cdot V_{z,N}^2 + \theta_3 \cdot V_{z,N} + \theta_4 + e_N \end{aligned}$$

or, in the matrix form:

$$\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_N \end{bmatrix} = \begin{bmatrix} V_{z,1}^3 & V_{z,1}^2 & V_{z,1} & 1 \\ V_{z,2}^3 & V_{z,2}^2 & V_{z,2} & 1 \\ \vdots & \vdots & \vdots & \vdots \\ V_{z,N}^3 & V_{z,N}^2 & V_{z,N} & 1 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \\ \theta_3 \\ \theta_4 \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix}$$

i.e., the estimation problem is recast in the standard form:

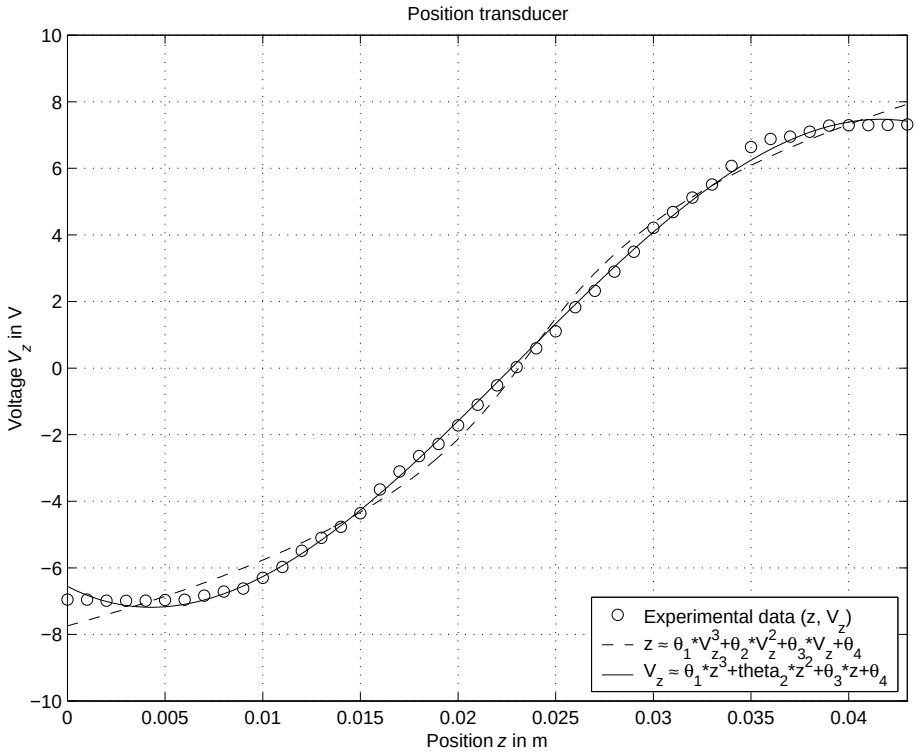
$$y = \Phi \cdot \theta + e^N$$

where $y \in \mathbb{R}^N$, $\Phi \in \mathbb{R}^{N \times 4}$, $\theta \in \mathbb{R}^4$ and $e \in \mathbb{R}^N$.

Using the least squares algorithm as estimation method, it result that

$$\hat{\theta} = \left(\Phi^T \cdot \Phi\right)^{-1} \cdot \Phi^T \cdot y = \begin{bmatrix} 2.4097 \cdot 10^{-5} \\ -3.4590 \cdot 10^{-5} \\ 0.0013 \\ 0.0231 \end{bmatrix}$$

Under MATLAB, such a least squares solution is also provided by the command `polyfit`. Please take a look to the help-on-line by typing `help polyfit`.



5. Main MATLAB commands to take into account

- `mldivide`
- `polyfit`
- `polyval`

Possible solution under MATLAB (file sensor.m)

```
% Parametric estimation of a position sensor model

clear all, close all, pack, clc

% position measured in meters, with a position offset
z_misurata=(0.037:0.001:0.08)';

% position measured in meters, without position offset
z=z_misurata-z_misurata(1);

% output voltage measured in volts
Vz=[-6.9522; -6.9530; -6.9873; -6.9922; -6.9868; -6.9649; -6.9563; -6.8359; ...
    -6.7097; -6.6211; -6.3014; -5.9746; -5.4875; -5.0964; -4.7653; -4.3603; -3.6474; ...
    -3.1056; -2.6439; -2.2799; -1.7218; -1.0987; -0.5180;  0.0326;  0.5958;  1.1052; ...
     1.8264;  2.3193;  2.9003;  3.4942;  4.2122;  4.6871;  5.1238;  5.5129;  6.0731; ...
     6.6401;  6.8797;  6.9531;  7.0979;  7.2823;  7.2941;  7.2998;  7.3008;  7.3191];

save sensore z Vz

plot(z,Vz,'o'), grid, title('Position transducer'),
xlabel('Position{\it z} in m'), ylabel('Voltage{\it V_z} in V'),
print -deps sensore1.eps

%--------------------------------------------------------------------
% Parametric estimation of a linear model using least squares
%--------------------------------------------------------------------

% indices corresponding to the linearity interval of the characteristic
i1=11;
i2=36;

% parametric estimation by means of least squares algorithm
L=[Vz(i1:i2), ones(i2-i1+1,1)];
p=L\z(i1:i2); % by means of the operator "\"
Kt=1/p(1)
Vo=-p(2)/p(1)

p_=inv(L'*L)*L'*z(i1:i2); % computing the pseudoinverse matrix
Kt_=1/p_(1)
Vo_=-p_(2)/p_(1)

% computation of the x% confidence intervals
% x=95 => k=2 ("2 sigma"); x=99.7 => k=3 ("3 sigma")
k_e=2; k_p=2;
rumore_max=5e-4;

% case #1: known variance
sigma_e=rumore_max/k_e
Sigma_e=sigma_e^2*eye(i2-i1+1);
Sigma_p=inv(L'*inv(Sigma_e)*L);
sigma_p=sqrt(diag(Sigma_p));
delta_p=k_p*sigma_p;
p_min=p-delta_p;
p_max=p+delta_p;
Kt_min=1/p_max(1)
Kt_max=1/p_min(1)
Vo_min=-p_max(2)/p_min(1)
Vo_max=-p_min(2)/p_max(1)

% case #2: variance to be estimated
sigma_e_mv=sqrt((z(i1:i2)-L*p)'*(z(i1:i2)-L*p)/(i2-i1+1))
Sigma_e=sigma_e_mv^2*eye(i2-i1+1);
Sigma_p=inv(L'*inv(Sigma_e)*L);
sigma_p=sqrt(diag(Sigma_p));
delta_p=k_p*sigma_p;
p_min=p-delta_p;
p_max=p+delta_p;
```

```
Kt_min=1/p_max(1)
Kt_max=1/p_min(1)
Vo_min=-p_max(2)/p_min(1)
Vo_max=-p_min(2)/p_max(1)

% Comparison of the results

Vz0=linspace(-10,10,1000);
z_hat=p(1)*Vz0+p(2);
z_min=min([p_max(1)*Vz0; p_min(1)*Vz0])+p_min(2);
z_max=max([p_max(1)*Vz0; p_min(1)*Vz0])+p_max(2);

figure, plot(z,Vz,'o',z(i1)*[1,1],[-10,10],'-',z(i2)*[1,1],[-10,10],'-',z_hat,Vz0),
axis([min(z),max(z),-10,10]), grid, title('Position transducer'),
xlabel('Position{\it z} in m'), ylabel('Voltage{\it V_z} in V'),
text(0.013,-9,'\fontsize{20} linearity interval'),
print -deps sensore2.eps

figure, plot(z,Vz,'o', z_min,Vz0,'--', z_hat,Vz0,'-', z_max,Vz0,'-.', ...
             z(i1)*[1,1],[-10,10],'-', z(i2)*[1,1],[-10,10],'-'),
axis([min(z),max(z),-10,10]), grid, title('Position transducer'),
xlabel('Position{\it z} in m'), ylabel('Voltage{\it V_z} in V'),
legend('Experimental data (z, V_z)_{ }^{  }', ...
       'z \approx min(\theta_{1}^{+} * V_{z} , \theta_{1}^{-} * V_{z}) + \theta_{2}^{-}', ...
       'z \approx \theta_{1} * V_{z} + \theta_2', ...
       'z \approx max(\theta_{1}^{+} * V_{z} , \theta_{1}^{-} * V_{z}) + \theta_{2}^{+}', 2),
text(0.013,-9,'\fontsize{20} linearity interval'),
print -deps sensore3.eps

%-----------------------------------------------------------
% 3rd order polynomial model by means of least squares
%-----------------------------------------------------------

% a) z = theta(1)*Vz^3 + theta(2)*Vz^2 + theta(3)*Vz + theta(4)
L=[Vz.^3, Vz.^2, Vz, Vz.^0];
p3a=L\z
z_pol=polyval(p3a,Vz0);

% b) Vz = theta(1)*z^3 + theta(2)*z^2 + theta(3)*z + theta(4)
L=[z.^3, z.^2, z, z.^0];
p3b=L\Vz
z0=linspace(min(z),max(z),1000);
Vz_pol=polyval(p3b,z0);

figure, plot(z,Vz,'o',z_pol,Vz0,'--',z0,Vz_pol,'-'),
axis([min(z),max(z),-10,10]), grid, title('Position transducer'),
xlabel('Position{\it z} in m'), ylabel('Voltage{\it V_z} in V'),
legend('Experimental data (z, V_z)_{ }^{  }', ...
       'z \approx \theta_1*V_z^3+\theta_2*V_z^2+\theta_3*V_z+\theta_4', ...
       'V_z \approx \theta_1*z^3+theta_2*z^2+\theta_3*z+\theta_4',4),
print -deps sensore4.eps

%-----------------------------------------------------------
% 3rd order polymonial model by means of polyfit
%-----------------------------------------------------------
modello_ordine3=polyfit(z,Vz,3);
Vz_modello_ordine3=polyval(modello_ordine3,z0);
```

Exercise #2: parametric estimation of a position transducer model using the set membership approach

1. Linear approximation of the position-voltage characteristic
The position-voltage characteristic of the transducer seen in the exercise #1 shows a nearly linear behaviour in the interval between 1.3 and 3.5 cm. In such a linearity interval, the characteristic can be described by the following model

$$V_z = K_t \cdot z + V_o$$

where the gain K_t and the offset voltage V_o are unknown constant parameters to be estimated. Since the most relevant error is due to the position measurement (the greatest position error is about 0.5 mm), the model equation can be more suitably rewritten in the following form, where the measurement error e is explicitly taken into account:

$$z = \frac{1}{K_t} \cdot V_z - \frac{V_o}{K_t} + e$$

By considering the N measurements collected in the linearity interval, the following system of linear equations is derived:

$$\begin{aligned} z_1 &= \frac{1}{K_t} \cdot V_{z,1} - \frac{V_o}{K_t} + e_1 \\ z_2 &= \frac{1}{K_t} \cdot V_{z,2} - \frac{V_o}{K_t} + e_2 \\ &\vdots \\ z_N &= \frac{1}{K_t} \cdot V_{z,N} - \frac{V_o}{K_t} + e_N \end{aligned}$$

where $V_{z,i}$ is the voltage provided by the transducer when the position value is z_i.

By introducing the unknown parameters:

$$\theta_1 = \frac{1}{K_t}, \quad \theta_2 = -\frac{V_o}{K_t}$$

the previous equations can be rewritten as:

$$\begin{aligned} z_1 &= V_{z,1} \cdot \theta_1 + \theta_2 + e_1 \\ z_2 &= V_{z,2} \cdot \theta_1 + \theta_2 + e_2 \\ &\vdots \\ z_N &= V_{z,N} \cdot \theta_1 + \theta_2 + e_N \end{aligned}$$

or, in matrix form:

$$\begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_N \end{bmatrix} = \begin{bmatrix} V_{z,1} & 1 \\ V_{z,2} & 1 \\ \vdots & \vdots \\ V_{z,N} & 1 \end{bmatrix} \cdot \begin{bmatrix} \theta_1 \\ \theta_2 \end{bmatrix} + \begin{bmatrix} e_1 \\ e_2 \\ \vdots \\ e_N \end{bmatrix}$$

i.e., the estimation problem is recast in the standard form:

$$y = \Phi \cdot \theta + e$$

where $y \in \mathbb{R}^N$, $\Phi \in \mathbb{R}^{N \times 2}$, $e \in \mathbb{R}^N$ and $\theta \in \mathbb{R}^2$ is the unknown.

Using the least squares algorithm as estimation method:

$$\hat{\theta} = A \cdot y, \qquad \text{with } A = \left(\Phi^T \cdot \Phi\right)^{-1} \Phi^T$$

it results that:

$$\hat{\theta} = \begin{bmatrix} \hat{\theta}_1 \\ \hat{\theta}_2 \end{bmatrix} = \begin{bmatrix} 1.8194 \cdot 10^{-3} \\ 2.2791 \cdot 10^{-2} \end{bmatrix} \quad \Rightarrow \quad \hat{K}_t = \frac{1}{\hat{\theta}_1} = 549.62\,\text{V/m}, \quad \hat{V}_o = -\frac{\hat{\theta}_2}{\hat{\theta}_1} = -12.526\,\text{V}$$

2. Evaluation of the estimate uncertainty intervals EUI^∞

The main error source in the experimental measurements is the measurement of the position z and the greatest error is about 0.5 mm. The measurement error e can then be considered as unknown but bounded and, in particular;

$$e \in \mathcal{B}_e^\infty = \left\{\tilde{e} \in \mathbb{R}^N : |\tilde{e}_k| \leq \varepsilon,\ k = 1, \ldots, N\right\}, \quad \text{with } \varepsilon = 5 \cdot 10^{-4}$$

Note that $\mathcal{B}_e^\infty$ is a cube with side length $2 \cdot \varepsilon$ centered in the origin.

The estimate uncertainty intervals EUI^∞ are defined as:

$$EUI_j^\infty = \left[\hat{\theta}_j^m = \min_{\theta \in EUS^\infty} \theta_j, \hat{\theta}_j^M = \max_{\theta \in EUS^\infty} \theta_j\right], \quad j = 1, 2$$

where the estimate uncertainty set EUS^∞ is the image under the linear mapping A of the measurement uncertainty set MUS^∞, which is a cube with side length $2 \cdot \varepsilon$ centered in y:

$$EUS^\infty = A \cdot MUS^\infty = A \cdot (y \oplus \mathcal{B}_e^\infty) = A \cdot \left\{\tilde{y} \in \mathbb{R}^N : |\tilde{y}_i - y_i| \leq \varepsilon,\ i = 1, \ldots, N\right\}$$

Since it can be proven that:

$$\begin{aligned} \hat{\theta}_j^m &= \min_{\theta \in EUS^\infty} \theta_j = \sum_{k=1}^{N} A_{jk} \cdot [y_k - \varepsilon \cdot \text{sign}\,(A_{jk})] \\ \hat{\theta}_j^M &= \max_{\theta \in EUS^\infty} \theta_j = \sum_{k=1}^{N} A_{jk} \cdot [y_k + \varepsilon \cdot \text{sign}\,(A_{jk})] = 2 \cdot \hat{\theta}_j - \hat{\theta}_j^m \end{aligned}$$

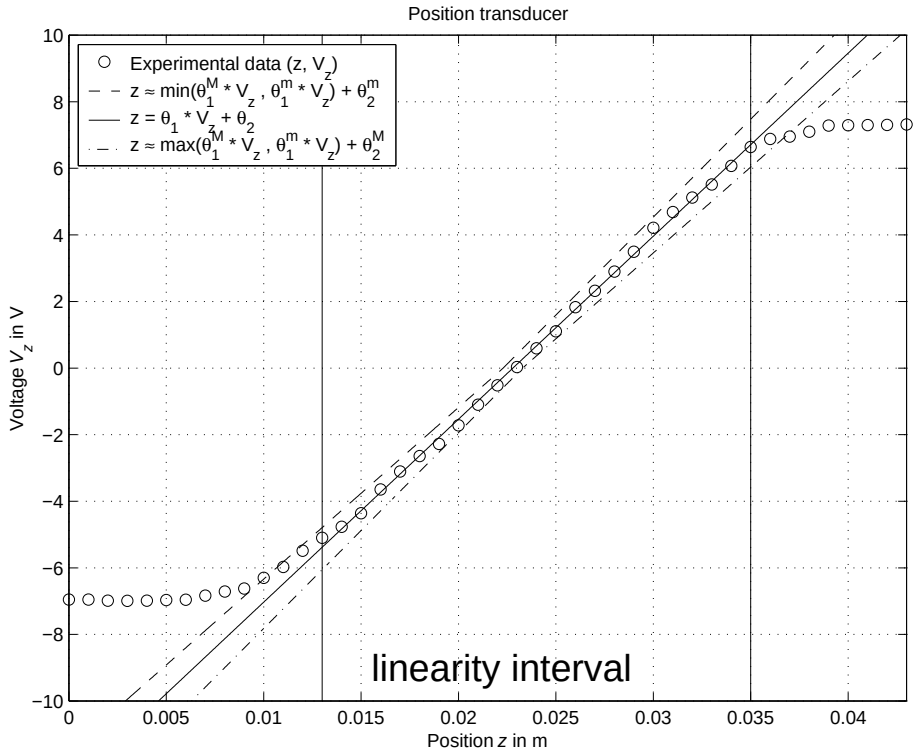
in this case it results that:

$$\begin{aligned} \left[\hat{\theta}_1^m, \hat{\theta}_1^M\right] &= [1.6996 \cdot 10^{-3}, 1.9392 \cdot 10^{-3}] \\ \left[\hat{\theta}_2^m, \hat{\theta}_2^M\right] &= [2.2291 \cdot 10^{-2}, 2.3291 \cdot 10^{-2}] \end{aligned}$$

and then the uncertainty intervals on the estimates $\hat{K}_t$ and $\hat{V}_o$ are equal to:

$$\begin{aligned} \left[\hat{K}_t^m, \hat{K}_t^M\right] &= \left[\frac{1}{\hat{\theta}_1^M}, \frac{1}{\hat{\theta}_1^m}\right] = [515.67, 588.36]\ \text{V/m} \\ \left[\hat{V}_o^m, \hat{V}_o^M\right] &= \left[-\frac{\hat{\theta}_2^M}{\hat{\theta}_1^m}, -\frac{\hat{\theta}_2^m}{\hat{\theta}_1^M}\right] = [-13.703, -11.495]\ \text{V} \end{aligned}$$

In the figure below, it is plotted the envelope of the static characteristics of models whose parameters θ are taken as the extremes of the estimate uncertainty intervals EUI_j^∞, $j = 1, 2$.



3. Evaluation of the feasible parameter uncertainty intervals PUI^{∞}

The feasible parameter uncertainty intervals PUI^{∞} are defined as:

$$PUI_j^{\infty} = \left[\min_{\theta \in FPS^{\infty}} \theta_j, \max_{\theta \in FPS^{\infty}} \theta_j \right] \subseteq EUI_j^{\infty}, \quad j = 1, 2$$

where the feasible parameter set FPS^{∞} is a polytope (i.e., a convex polyhedron) generated by linear inequalities:

$$FPS^{\infty} = \left\{ \tilde{\theta} \in \mathbb{R}^{dim(\tilde{\theta})} : \left| y_i - [\Phi \cdot \tilde{\theta}]_i \right| \leq \varepsilon, \ i = 1, \ldots, N \right\}$$

The extremes of the intervals PUI_j^{∞} are obtained as solutions of linear programming problems. The MATLAB commands `lp` in the form `lp(c,M,b)` or `linprog` in the form `linprog(c,M,b,[],[],[],[],[],optimset(optimset('linprog'),'LargeScale','off'))` allow to solve the problem

$$\min_x c^T \cdot x \quad \text{with the constraint } M \cdot x \leq b$$

It is then necessary to suitably rewrite the inequalities that define FPS^{∞}:

$$\left| y_i - [\Phi \cdot \tilde{\theta}]_i \right| \leq \varepsilon \Leftrightarrow -\varepsilon \leq y_i - [\Phi \cdot \tilde{\theta}]_i \leq \varepsilon \Leftrightarrow \left\{ \begin{array}{l} [\Phi \cdot \tilde{\theta}]_i \leq y_i + \varepsilon \\ -[\Phi \cdot \tilde{\theta}]_i \leq -y_i + \varepsilon \end{array} \right. , \ i = 1, \ldots, N$$

and then it follows that:

$$\begin{aligned} \theta_j^m &= \min_{\theta \in FPS^{\infty}} \theta_j = \min_{M \cdot \theta \leq b} c^T \cdot \theta \\ \theta_j^M &= \max_{\theta \in FPS^{\infty}} \theta_j = - \min_{\theta \in FPS^{\infty}} (-\theta_j) = - \min_{M \cdot \theta \leq b} (-c)^T \cdot \theta \end{aligned}$$

where:

$$M = \left[\begin{array}{c} \Phi \\ -\Phi \end{array} \right], \quad b = \left[\begin{array}{c} y \\ -y \end{array} \right] + \varepsilon, \quad c = j\text{-th column of the identity matrix } I_{2 \times 2}$$

In this case it results that:

$$\begin{aligned} \left[\theta_1^m = \min_{\theta \in FPS^{\infty}} \theta_1, \theta_1^M = \max_{\theta \in FPS^{\infty}} \theta_1 \right] &= [1.7909 \cdot 10^{-3}, 1.8484 \cdot 10^{-3}] \\ \left[\theta_2^m = \min_{\theta \in FPS^{\infty}} \theta_2, \theta_2^M = \max_{\theta \in FPS^{\infty}} \theta_2 \right] &= [2.2596 \cdot 10^{-2}, 2.2807 \cdot 10^{-2}] \end{aligned}$$

and then the uncertainty intervals on the feasible parameters K_t e V_o are equal to:

$$\begin{aligned}\left[K_t^m, K_t^M\right] &= \left[\frac{1}{\theta_1^M}, \frac{1}{\theta_1^m}\right] = [541.01, 558.38]\ \mathrm{V/m} \\ \left[V_o^m, V_o^M\right] &= \left[-\frac{\theta_2^M}{\theta_1^m}, -\frac{\theta_2^m}{\theta_1^M}\right] [-12.735, -12.225]\ \mathrm{V}\end{aligned}$$

4. Evaluation of the central estimate of the parameters

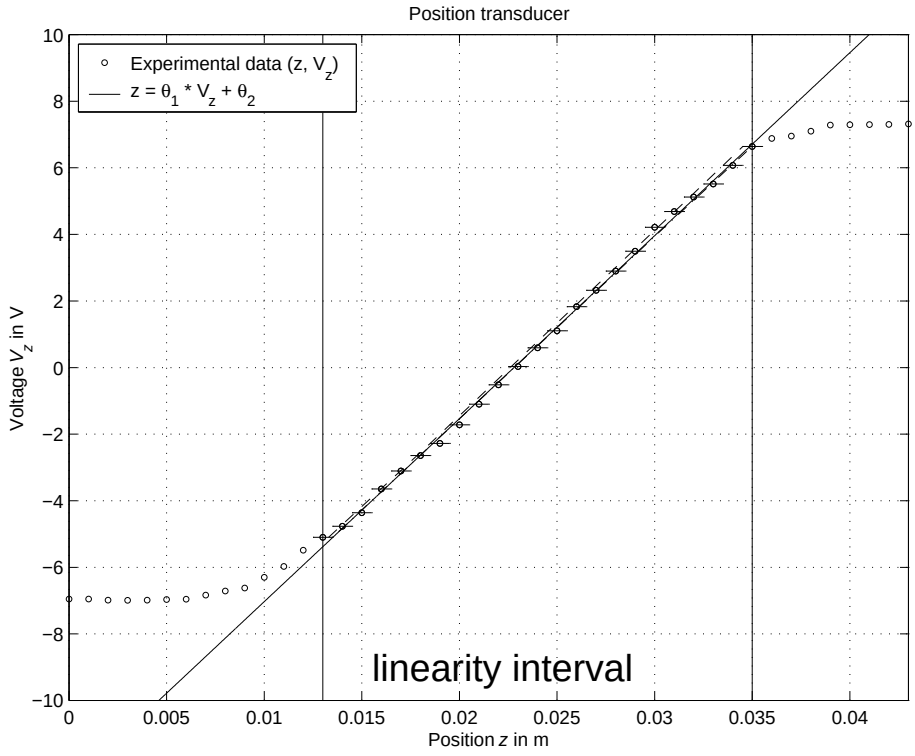
The central estimate of unknown parameter θ is given by:

$$\theta^C = \begin{bmatrix} \theta_1^C \\ \theta_2^C \end{bmatrix} = \begin{bmatrix} \left(\min\limits_{\theta \in FPS^\infty} \theta_1 + \max\limits_{\theta \in FPS^\infty} \theta_1\right) \Big/ 2 \\ \left(\min\limits_{\theta \in FPS^\infty} \theta_2 + \max\limits_{\theta \in FPS^\infty} \theta_2\right) \Big/ 2 \end{bmatrix} = \begin{bmatrix} 1.8196 \cdot 10^{-3} \\ 2.2702 \cdot 10^{-2} \end{bmatrix}$$

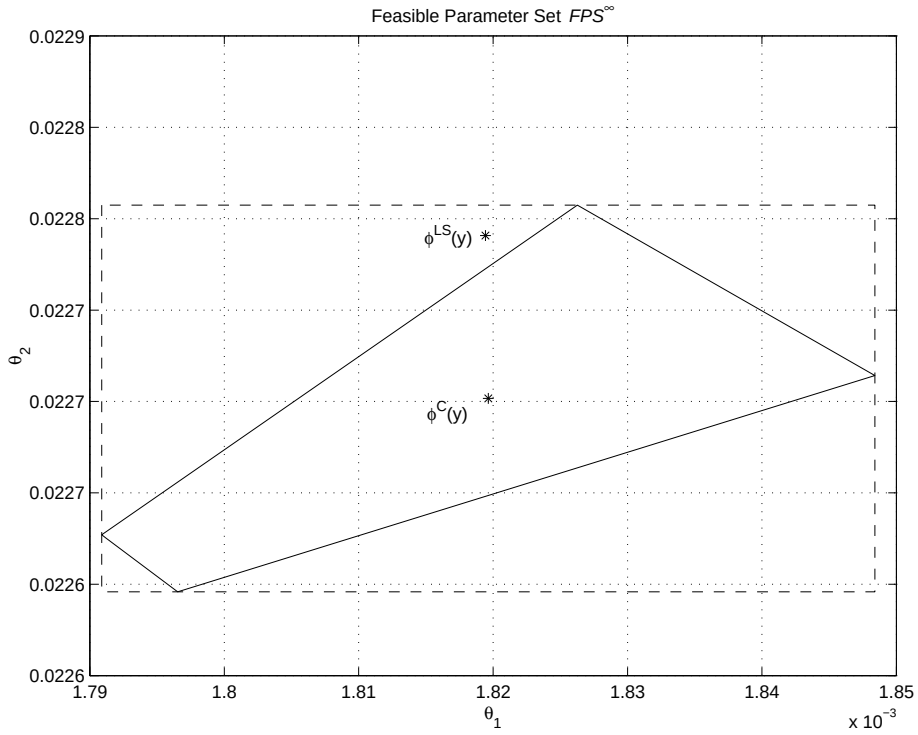
and the corresponding central estimate of the parameters K_t e V_o is equal to:

$$\begin{aligned} K_t^C &= \frac{1}{\theta_1^C} = 549.56\,\mathrm{V/m} \\ V_o^C &= -\frac{\theta_2^C}{\theta_1^C} = -12.476\,\mathrm{V} \end{aligned}$$

In the following figure, the envelope of the static characteristics of models whose parameters θ belong to the feasible parameter set FPS^∞ is plotted.



In the figure below, the feasible parameter set FPS^∞ (continuous line) and the set of estimates given by the extremes of the feasible parameter uncertainty intervals PUI_j^∞, $j = 1, 2$, are plotted.



5. Main MATLAB commands to take into account

- `lp`
- `convhull`

Possible solution under MATLAB (file sensor_sm.m)

```
% Set membership estimation of a position sensor model

clear all, close all, pack, clc
format compact, format short e

load sensor

% bound on the greatest measurement error of the position z
epsilon=5e-4

% indices corresponding to the linearity interval of the characteristic
i1=14; % use both 11 (at first) and 14 (at the end)
i2=36;

z_lin = z(i1:i2);
Vz_lin=Vz(i1:i2);

%------------------------------------------------------------------
% Parametric estimation of a linear model using least squares
%------------------------------------------------------------------

% parametric estimation by means of least squares algorithm
L=[Vz_lin, ones(i2-i1+1,1)];
A=inv(L'*L)*L';

nu_hat=L\z_lin   % by means of the operator "\"
nu_hat_=A*z_lin; % by means of the pseudoinverse matrix

%------------------------------------------------------------------
% Evaluation of the uncertainty intervals EUI in l-infinity norm
%------------------------------------------------------------------

for ind=1:length(nu_hat),
```

```
    nu_min(ind,1)=A(ind,:)*(z_lin-epsilon*sign(A(ind,:))');
end
nu_min
nu_max=2*nu_hat-nu_min

% check

nu_min_=[0;0]; nu_max_=[0;0];
for k=1:length(A),
    nu_min_(1)=nu_min_(1)+A(1,k)*(z_lin(k)-epsilon*sign(A(1,k)));
    nu_min_(2)=nu_min_(2)+A(2,k)*(z_lin(k)-epsilon*sign(A(2,k)));
    nu_max_(1)=nu_max_(1)+A(1,k)*(z_lin(k)+epsilon*sign(A(1,k)));
    nu_max_(2)=nu_max_(2)+A(2,k)*(z_lin(k)+epsilon*sign(A(2,k)));
end
nu_min_
nu_max_
nu_max__=2*nu_hat-nu_min_

EUI=[nu_min, nu_max]
Kt_min=1/EUI(1,2)
Kt_max=1/EUI(1,1)
Vo_min=-EUI(2,2)/EUI(1,1)
Vo_max=-EUI(2,1)/EUI(1,2)

% Comparison of the results

Vz0=linspace(-10,10,1000);
z_hat=nu_hat(1)*Vz0+nu_hat(2);
z_min=min([nu_max(1)*Vz0; nu_min(1)*Vz0])+nu_min(2);
z_max=max([nu_max(1)*Vz0; nu_min(1)*Vz0])+nu_max(2);

figure, plot(z,Vz,'o', z_min,Vz0,'--', z_hat,Vz0,'-', z_max,Vz0,'-.', ...
             z(i1)*[1,1],[-10,10],'-', z(i2)*[1,1],[-10,10],'-'),
axis([min(z),max(z),-10,10]), grid, title('Position transducer'),
xlabel('Position{\it z} in m'), ylabel('Voltage{\it V_z} in V'),
legend('Experimental data (z, V_z)_{ }^{  }', ...
       'z \approx min(\theta_{1}^{M} * V_{z} , \theta_{1}^{m} * V_{z}) + \theta_{2}^{m}', ...
       'z = \theta_{1} * V_{z} + \theta_2', ...
       'z \approx max(\theta_{1}^{M} * V_{z} , \theta_{1}^{m} * V_{z}) + \theta_{2}^{M}', 2),
text(0.015,-9,'\fontsize{20} linearity interval'),
print -deps sensore5.eps

%-----------------------------------------------------------------
% Evaluation of the uncertainty intervals PUI in l-infinity norm
%-----------------------------------------------------------------

% PUI_LP(1,1)=[1; 0]'*lp( [1; 0],[L; -L],[z_lin+epsilon; -z_lin+epsilon]); % Under Matlab 5.3
% PUI_LP(2,1)=[0; 1]'*lp( [0; 1],[L; -L],[z_lin+epsilon; -z_lin+epsilon]); % Under Matlab 5.3
% PUI_LP(1,2)=[1; 0]'*lp(-[1; 0],[L; -L],[z_lin+epsilon; -z_lin+epsilon]); % Under Matlab 5.3
% PUI_LP(2,2)=[0; 1]'*lp(-[0; 1],[L; -L],[z_lin+epsilon; -z_lin+epsilon]); % Under Matlab 5.3

options_old=optimset('linprog'); % the default value for the 'LargeScale' parameter is 'on'
options_new=optimset(options_old, 'LargeScale','off');
PUI(1,1)=[1; 0]'*linprog( [1; 0],[L; -L],[z_lin+epsilon; -z_lin+epsilon], ...
                          [],[],[],[],[],options_new);
PUI(2,1)=[0; 1]'*linprog( [0; 1],[L; -L],[z_lin+epsilon; -z_lin+epsilon], ...
                          [],[],[],[],[],options_new);
PUI(1,2)=[1; 0]'*linprog(-[1; 0],[L; -L],[z_lin+epsilon; -z_lin+epsilon], ...
                          [],[],[],[],[],options_new);
PUI(2,2)=[0; 1]'*linprog(-[0; 1],[L; -L],[z_lin+epsilon; -z_lin+epsilon], ...
                          [],[],[],[],[],options_new);

PUI
Kt_min=1/PUI(1,2)
Kt_max=1/PUI(1,1)
Vo_min=-PUI(2,2)/PUI(1,1)
Vo_max=-PUI(2,1)/PUI(1,2)
```

```
%-----------------------------------------------------------------------
% Approximation of the uncertainty intervals PUI in l-infinity norm
%-----------------------------------------------------------------------

n=16;
theta_vect=[];
for k=0:n-1,
    theta=linprog([sin(2*pi*k/n), cos(2*pi*k/n)],[L; -L],[z_lin+epsilon; -z_lin+epsilon], ...
                  [],[],[],[],[],options_new);
    theta_vect=[theta_vect, theta];
end

%-----------------------------------------------------------------------
% Evaluation of the central estimate
%-----------------------------------------------------------------------

nu_central=[(PUI(1,1)+PUI(1,2))/2; (PUI(2,1)+PUI(2,2))/2]
Kt_central=1/nu_central(1)
Vo_central=-nu_central(2)/nu_central(1)

% Comparison of the results

clear z_min z_max
nu_vect=[];
for k=1:length(Vz_lin),
    % nu=lp([Vz_lin(k); 1],[L; -L],[z_lin+epsilon; -z_lin+epsilon]); % Under Matlab 5.3
    nu=linprog([Vz_lin(k); 1],[L; -L],[z_lin+epsilon; -z_lin+epsilon], ...
               [],[],[],[],[],options_new);
    nu_vect=[nu_vect, nu];
    z_min(k)=[Vz_lin(k), 1]*nu;
    % nu=lp(-[Vz_lin(k); 1],[L; -L],[z_lin+epsilon; -z_lin+epsilon]); % Under Matlab 5.3
    nu=linprog(-[Vz_lin(k); 1],[L; -L],[z_lin+epsilon; -z_lin+epsilon], ...
                [],[],[],[],[],options_new);
    nu_vect=[nu_vect, nu];
    z_max(k)=[Vz_lin(k), 1]*nu;
end

figure, plot(z,Vz,'o'), hold on,
child=get(get(gcf,'Child'),'Child'); set(child(1),'MarkerSize',3);
plot(z_hat,Vz0,'-', z_min,Vz_lin,'--', z_max,Vz_lin,'-.', ...
     z(i1)*[1,1],[-10,10],'-', z(i2)*[1,1],[-10,10],'-'),
plot_dx(z_lin,Vz_lin,epsilon,epsilon*0), hold off,
axis([min(z),max(z),-10,10]), grid on, zoom on, title('Position transducer'),
xlabel('Position{\it z} in m'), ylabel('Voltage{\it V_z} in V'),
legend('Experimental data (z, V_z)_{ }^{  }', ...
       'z = \theta_{1} * V_{z} + \theta_2', 2),
text(0.015,-9,'\fontsize{20} linearity interval'),
print -deps sensore6.eps

k=convhull(nu_vect(1,:),nu_vect(2,:));
figure, plot(nu_vect(1,k),nu_vect(2,k),'-',nu_hat(1),nu_hat(2),'*', ...
   [PUI(1,1),PUI(1,:),PUI(1,2:-1:1)], [PUI(2,2:-1:1),PUI(2,:),PUI(2,2)],'--', ...
   nu_central(1),nu_central(2),'*k'),
grid on, text(nu_hat(1)*0.9975,nu_hat(2),'\phi^{LS}(y)'),
text(nu_central(1)*0.9975,nu_central(2)*0.9997,'\phi^{C}(y)'),
title('Feasible Parameter Set {\it FPS^\infty}'),
xlabel('\theta_{1}'), ylabel('\theta_{2}')
print -deps sensore7.eps

k1=convhull(theta_vect(1,:),theta_vect(2,:));
figure, plot(theta_vect(1,k1),theta_vect(2,k1),'-',nu_hat(1),nu_hat(2),'*', ...
   [PUI(1,1),PUI(1,:),PUI(1,2:-1:1)], [PUI(2,2:-1:1),PUI(2,:),PUI(2,2)],'--', ...
   nu_central(1),nu_central(2),'*k'),
grid on, text(nu_hat(1)*0.9975,nu_hat(2),'\phi^{LS}(y)'),
text(nu_central(1)*0.9975,nu_central(2)*0.9997,'\phi^{C}(y)'),
title('Feasible Parameter Set {\it FPS^\infty}'),
xlabel('\theta_{1}'), ylabel('\theta_{2}')
print -deps sensore8.eps
```