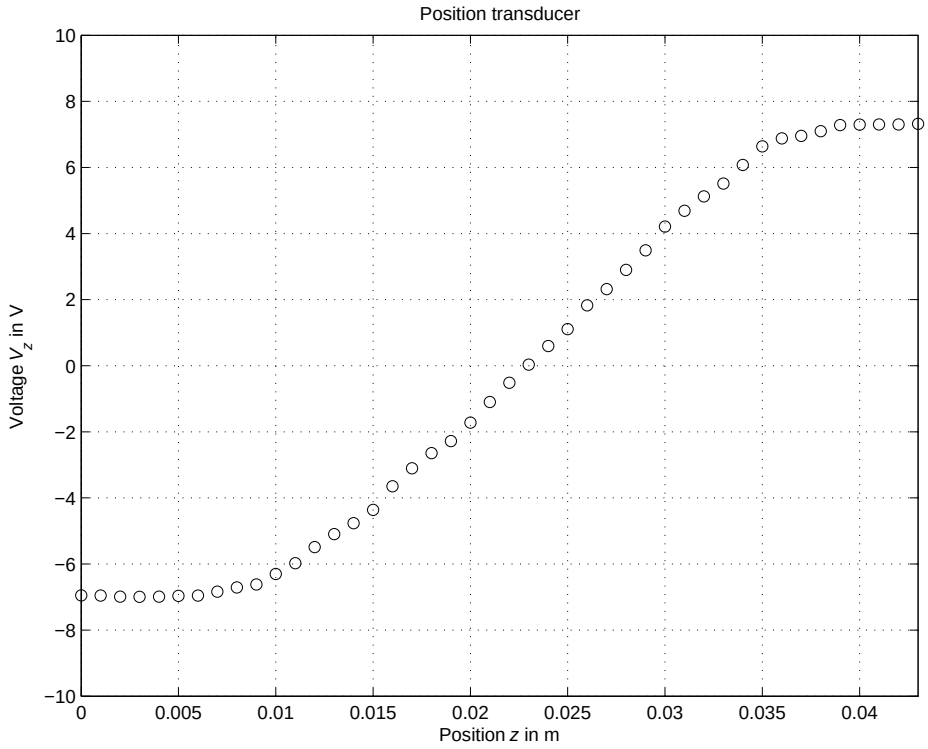
Parametric estimation of static models for a position transducer using the classical (or statistical) approach

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.

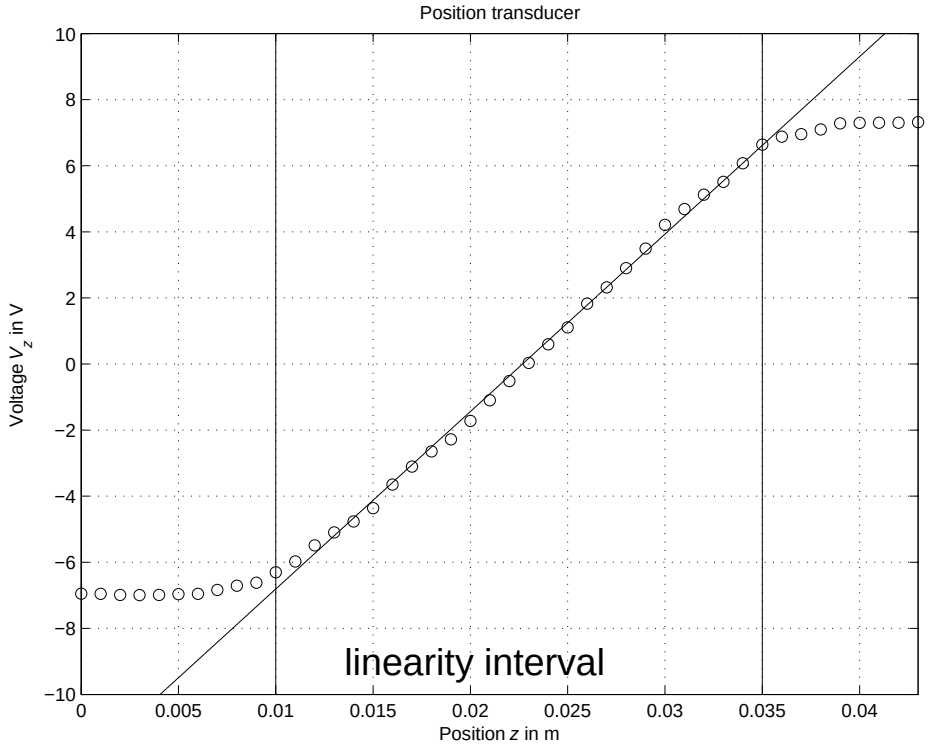
Using the least squares algorithm as estimation method:

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

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:

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

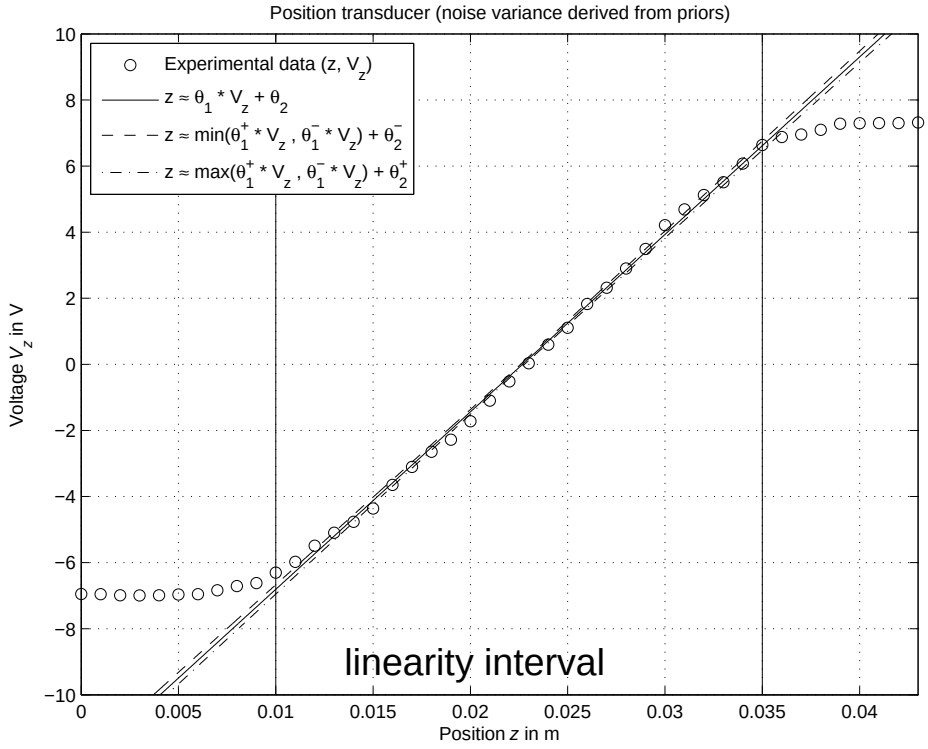
The confidence intervals of parameters $\hat{\theta}_i$, computed for example at 95%, are given by:

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

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:

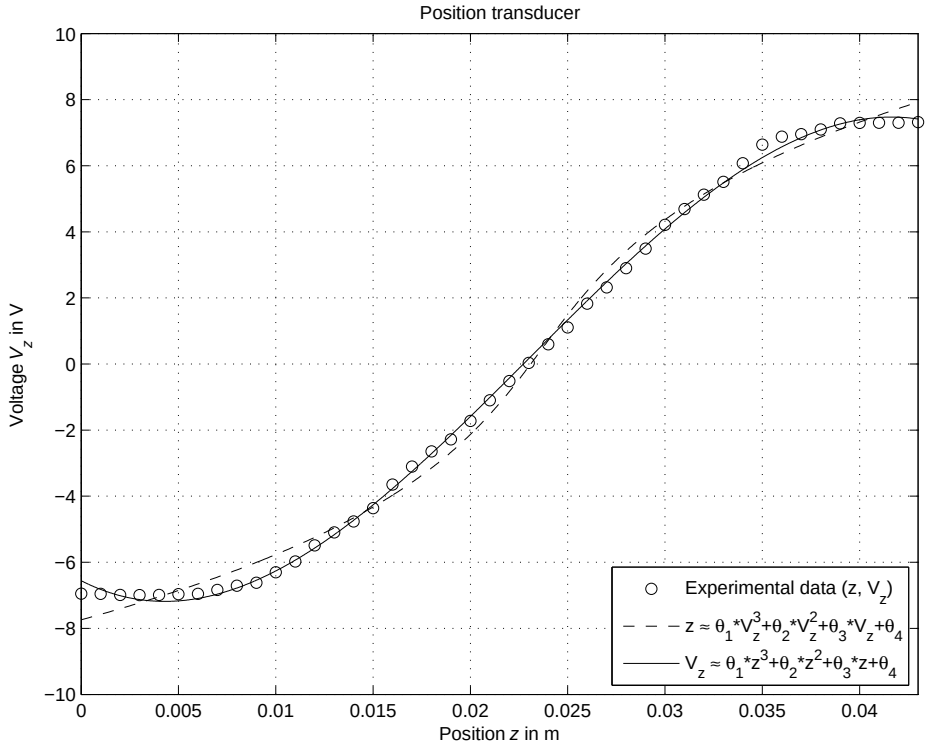
$$y = \Phi \cdot \theta + e^N$$

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)

```
%% Laboratory 1 - Estimation, filtering and system identification - Prof. M. Taragna
% *Parametric estimation of position transducer models*
% *using the classical (or statistical) approach*
%% Introduction
% The program code may be splitted in sections using the characters "%%".
% Each section can run separately with the command "Run Section"
% (in the Editor toolbar, just to the right of the "Run" button). You can do the
% same thing by highlighting the code you want to run and by using the button
% function 9 (F9). This way, you can run only the desired section of your code,
% saving your time. This script can be considered as a reference example.

clear all, close all, clc

%% Procedure
% # Load the file |sensor.mat| containing the position/voltage data
% # Plot the voltage measurements versus the position measurements
% # Estimate the parameters of the linear approximation using least squares
% # Plot the estimated approximation versus the experimental data
% # Evaluate the estimate uncertaity using the prior information only
% # Plot these confidence intervals versus the estimated approximation
% # Evaluate the estimate uncertaity using the experimental data only
% # Plot these confidence intervals versus the estimated approximation
% # Estimate the parameters of polynomial approximations using least squares
% # Plot the estimated approximations versus the experimental data

%% Problem setup

% Step 1: load of data

load sensor
% z  = position measured in meters, without position offset
% Vz = voltage measured in volts

% Step 2: plot of data

figure, plot(z,Vz,'o'),
axis([min(z),max(z),-10,10]), grid, title('Position transducer'),
xlabel('Position{\it z} in m'), ylabel('Voltage{\it V_z} in V')

%% Parametric estimation of a linear model (w.r.t. data) using least squares

% Step3: computation of the parameter estimate

% definition of the linearity interval of the characteristic
i1=11; % z(11) = 0.01
i2=36; % z(36) = 0.035
z_lin=z(i1:i2);
Vz_lin=Vz(i1:i2);
N_lin=length(z_lin);

% parameter estimation by means of least squares algorithm
Phi=[Vz_lin, ones(N_lin,1)];
p=Phi\z_lin; % Form #1: using the "\" operator (more realiable)
Kt=1/p(1)
Vo=-p(2)/p(1)

p_=inv(Phi'*Phi)*Phi'*z_lin; % Form #2: using the pseudoinverse matrix
Kt_=1/p_(1)
Vo_=-p_(2)/p_(1)

% Step 4: graphical comparison of the results

Vz0=linspace(-10,10,1000);
z_hat=p(1)*Vz0+p(2);
figure, plot(z,Vz,'o', z_hat,Vz0,'-', ...
             z_lin(1)*[1,1],[-10,10],'-', z_lin(end)*[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'),
text(0.013,-9,'\fontsize{20} linearity interval')

%% Computation of the parameter confidence intervals (noise variance derived from priors)

% Step 5: computation of the confidence intervals

% Definition of the x% confidence intervals:
% x=95.4 => k=2 ("2 sigma"); x=99.7 => k=3 ("3 sigma")
k_e=2; k_p=2;
noise_max=5e-4;

sigma_e=noise_max/k_e

Sigma_e=sigma_e^2*eye(N_lin);
Sigma_p=inv(Phi'*inv(Sigma_e)*Phi);
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)

% Step 6: graphical comparison of the results

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_hat,Vz0,'-', ...
             z_min,Vz0,'--', z_max,Vz0,'-.', ...
             z_lin(1)*[1,1],[-10,10],'-',z_lin(end)*[1,1],[-10,10],'-')
axis([min(z),max(z),-10,10]), grid,
title('Position transducer (noise variance derived from priors)'),
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} + \theta_2', ...
       'z \approx min(\theta_{1}^{+} * V_{z} , \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')

%% Computation of the parameter confidence intervals (noise variance estimated from data)

% Step 7: computation of the confidence intervals

sigma_e_ml=sqrt((z_lin-Phi*p)'*(z_lin-Phi*p)/N_lin)

Sigma_e=sigma_e_ml^2*eye(N_lin);
Sigma_p=inv(Phi'*inv(Sigma_e)*Phi);
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)

% Step 8: graphical comparison of the results

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_hat,Vz0,'-', ...
             z_min,Vz0,'--', z_max,Vz0,'-.', ...
             z_lin(1)*[1,1],[-10,10],'-',z_lin(end)*[1,1],[-10,10],'-')
axis([min(z),max(z),-10,10]), grid,
```

```
title('Position transducer (noise variance estimated from data)'),
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} + \theta_2', ...
       'z \approx min(\theta_{1}^{+} * V_{z} , \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')

%% Parametric estimation of 3rd order polynomial models (w.r.t. data) using least squares

% Step 9: computation of the parameter estimate

% a) assuming Vz as independent variable, z as variable dependent on Vz =>
%    z = theta(1)*Vz^3 + theta(2)*Vz^2 + theta(3)*Vz + theta(4)
Phi=[Vz.^3, Vz.^2, Vz, Vz.^0];
format shortE, format compact
p3a=Phi\z                       % Form #1: using the "\" operator
z_pol=polyval(p3a,Vz0);

p3a_=polyfit(Vz,z,3)            % Form #2: using the "polyfit" function
format short, format compact

% b) assuming z as independent variable, Vz as variable dependent on z =>
%    Vz = theta(1)*z^3 + theta(2)*z^2 + theta(3)*z + theta(4)
Phi=[z.^3, z.^2, z, z.^0];
p3b=Phi\Vz                      % Form #1: using the "\" operator
z0=linspace(min(z),max(z),1000);
Vz_pol=polyval(p3b,z0);

p3b_=polyfit(z,Vz,3)            % Form #2: using the "polyfit" function

% Step 10: graphical comparison of the results

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)
```