MATERIAŁY DYDAKTYCZNE DO PRZEDMIOTU

IDENTIFICATION OF CONTROL SYSTEMS w języku angielskim

Laboratorium: 15 godzin

Wydział Elektrotechniki, Automatyki i Informatyki

Opracował: Józef Ciosmak

Kielce, 2012/2013

CONTENTS

1 MODELS IN THE IDENTIFICATION PROCESS 3

1.1 METHOD OF IDENTIFICATION WITH MODEL TUNING 4
1.2 CONVEREGENCE OF THE MODEL AND THE PLANT 5
1.3 IDENTIFICATION OF FUNCTIONAL MODELS 7

2 IDENTIFICATION OF A DYNAMIC MODEL OF A VEHICLE SUSPENSION SYSTEM 9

3 IDENTIFICATION OF FEEDBACK CONTROL SYSTEMS 11

3.1 MODEL IDENTIFICATION - AN OPEN-LOOP SYSTEM 12
3.2 MODEL IDENTIFICATION - A FEEDBACK SYSTEM 12
3.3 ACTIVE DIRECT IDENTIFICATION 13
3.4 INDIRECT IDENTIFICATION 14

4 ESTIMATION AND VERIFICATION OF MODEL PARAMETERS - *SYSTEM IDENTIFICATION TOOLBOX* 15

4.1 ESTIMATION OF THE MODEL STRUCTURE AND PARAMETERS 15
4.2 MODEL VERIFICATION OF THE MODELS 17

5 IDENTIFICATION OF THE PID CONTROLLER PARAMETERS 18

5.1 IDENTIFICATION IN MATLAB-SIMULINK 18
5.2 SELECTION OF OPTIMAL PARAMETERS OF A PID CONTROLLER 18

6 IDENTIFICATION OF THE MODEL PARAMETERS FOR A HYDRAULIC SYSTEM 20

6.1 A MATHEMATICAL MODEL 20
6.2 METHOD OF IDENTIFICATION 22
6.3 SELECTION OF INPUT SIGNALS. 22
6.4 SELECTION OF PARAMETERS REQUIRED FOR IDENTIFICATION 23
6.5 STAGES OF THE IDENTIFICATION PROCESS – CLASSES 24
6.5.1 Open-loop system identification 24
6.5.2 Closed-loop system identification 28

1 MODELS IN THE IDENTIFICATION PROCESS

Identification is a multistage process. For mechanical systems, identification consists of four stages:
- modeling (creating a physical and mathematical model),
- performing an experiment,
- estimating the parameters and determining the model parameters,
- verifying the results.

These stages are always present in the identification process, irrespective of whether the problem is solved online or offline [1,2].

Identifying a dynamic system by means of a simple model requires finding a model that is excited with the same input signal as the plant and that generates an output as close to the output of the plant as possible. This type of a model is used mainly when disturbances affect the plant output (Fig. 1.1).

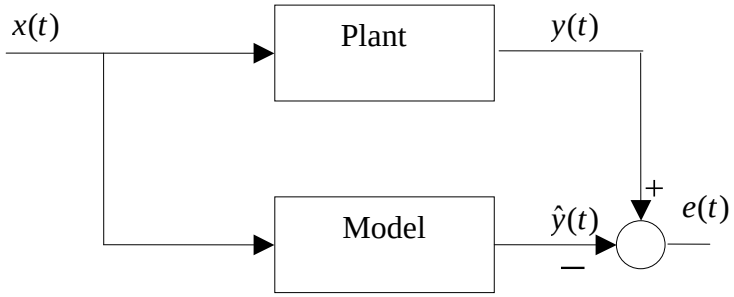


Fig. 1.1 A block diagram of a simple model

When an inverse model is applied, identification of a dynamic system involves finding a model that generates the same input as the plant. The signals have opposite roles: the input signal of the model becomes the output signal of the plant (Fig. 1.2). This can be achieved as long as the model is reversible. Model reversibility is true for linear models and a certain class of non-linear models in which the output can be univocally determined on the basis of the input.

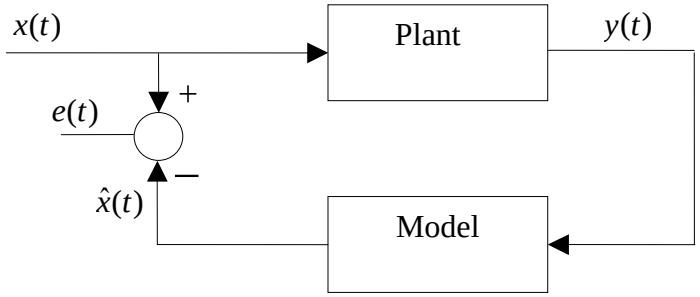


Fig. 1.2 A block diagram of an inverse model

Error estimation is also possible with a ‘generalized’ model shown in Fig. 1.3. However, this type of a model can only be used in the case of inverse models.

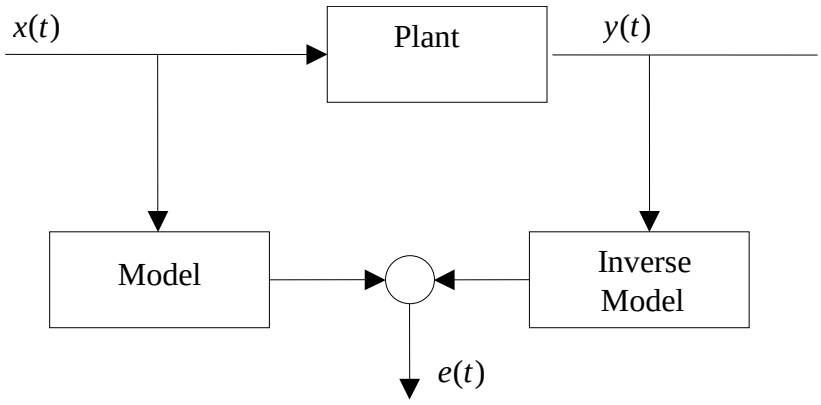


Fig. 1.3 A block diagram of a generalized model

1.1 METHOD OF IDENTIFICATION WITH MODEL TUNING

The development of information technology has contributed to the development of iterative methods, which enable us to find optimal parameters at predetermined initial parameters, taking into account the quality criteria of the model and the plant [3,4].

The model tuning procedure is illustrated in the block diagrams in Figs. 1.4 and 1.5.

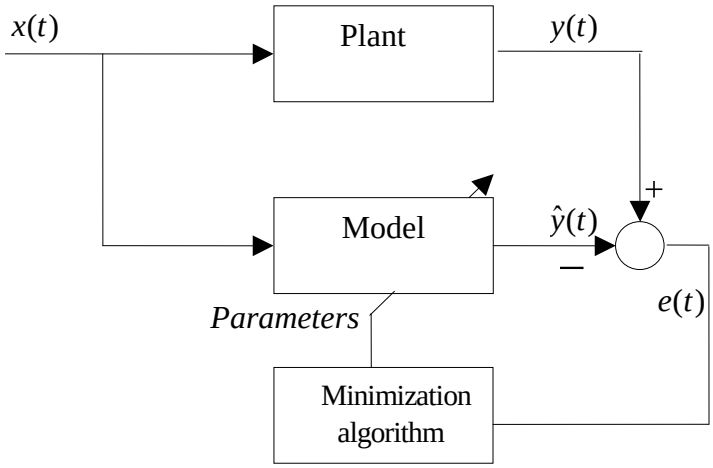


Fig. 1.4 A block diagram of parameter estimation for a simple model (algorithm with model tuning)

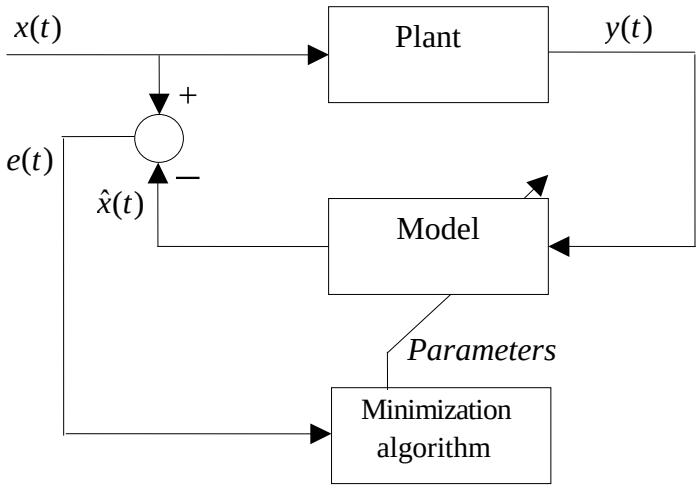


Fig. 1.5 A block diagram of parameter estimation for an inverse model (algorithm with model tuning)

The model tuning procedure comprises:

- assuming a mathematical model (structure),
- selecting a procedure to solve the model equations,
- developing a methodology for an experiment and conducting the experiment,
- assuming the model quality function,
- developing an algorithm for optimization of the quality function,
- selecting the initial points,
- assuming the criterion of tuning completion.

The model form is selected by determining its class and structure. **The class** is defined as a logically justified combination of system properties. A model can exhibit opposite properties, i.e. it can be both linear and non-linear, discrete and continuous, static and dynamic. An example of a class may be a nonlinear discrete and static system.

The structure is a narrower concept. It implies that there may occur several or several dozen different structures within one class.

In practice, determining the model class generally requires determining the model structure with an accuracy to unknown parameters. When the structure of a mathematical model is selected, some additional limitations related to calculability or feasibility are taken into account. It is thus vital that models be as simple as possible.

The method selected to solve equations describing the system behavior depends on the form of the model assumed. The method used to solve the motion equations has a considerable effect on the accuracy of the parameter determination.

If an experiment is correctly planned and the input signals are optimally selected, the model can be tuned more accurately and in a shorter time.

1.2 CONVERGENCE BETWEEN THE MODEL AND THE PLANT

This section deals with the convergence between the model and the plant. By using a quality function, we are able not only to assess the suitability of a mathematical model but also search for a model that is best in a given context. Optimization can refer to both the structure and the parameters. It is essential that the **model quality functions** should satisfy the following conditions [5,6]:

- they must be easy to minimize,
- they must clearly represent the differences between the model and the plant,
- they must be even functions,

- they must minimize the effect of disturbances.

A frequent criterion is the difference between the output signals of the model and the output or input signals of the plant (for an inverse model):

$$e(t) = y(t,\theta) - y_m(t,\hat{\theta}) \tag{1.1}$$

where:

$y(t,\theta)$ - the output signal of the plant,

$y_m(t,\hat{\theta})$ - the output signal of the model.

We also use an error defined as

$$e(t) = \theta(t) - \hat{\theta}(t) \tag{1.2}$$

where:

$\theta(t)$ - time-dependent parameters of the plant,

$\hat{\theta}(t)$ - time-dependent parameters of the model.

The error defined as the difference between signals is more commonly applied. Using this error, we can define the model quality criteria, the most important of which are:

the Integral of Squared Error,

$$ISE = \int_0^\infty e^2 dt \tag{1.3}$$

the Integral of Absolute Error,

$$IAE = \int_0^\infty |e| dt \tag{1.4}$$

the Integral of Time-Weighted Squared Error,

$$ITSE = \int_0^\infty t e^2 dt \tag{1.5}$$

the maximum error,

$$\varepsilon = \max|e| \tag{1.6}$$

and instantaneous criteria (for example, for online identification).

$$\varepsilon = e^2 \tag{1.7}$$

If computers are used, some values of an error are changed into a sequence of discrete values of the error, and the integrals are replaced with appropriate sums.

$$ISE = \sum_{i=1}^{N} e_i^2 \tag{1.8}$$

Optimization of a quality function requires determining its extremum. An extremum (a minimum) is searched for in the case of a function of one or many variables.

The most common techniques used to find a function of one variable are:

- the Fibbonacci serach,
- the golden section search ,
- the dichotomic search.

The methods used to find a function of many variables can be divided into four groups:

- analytical methods,
- gradient methods,
- relaxation methods,
- random search methods.

In practice, the above methods are frequently combined, and the problem requires searching for a minimum of the quality function.

Selecting the initial point is another important step of the identification process. The selection of the initial point generally affects the speed of convergence of the iteration method, and accordingly the calculation time. When the quality function has both global and local minima, and when the initial values of the optimization procedure are not selected properly, the search will be conducted beyond the area of admissible parameters in the direction of the local minimum.

When identification is conducted with methods based on model tuning, two criteria of tuning completion can be used. One terminates tuning when the value of the target function reaches a value that is lower than the predetermined admissible value. The other criterion terminates tuning when the difference between the parameters is lower than the assumed one. The first criterion is a better choice when we expect convergence between the response of the model and the plant to a predetermined signal. The second criterion is selected when we search for convergence between parameters.

1.3 IDENTIFICATION OF FUNCTIONAL MODELS

Identification of functional models does not require any knowledge of the model structure. The model is treated as a "black box" and it defines the relationships between the input signal and the output signal. The most common functional models are frequency and time characteristics. They fully characterize the dynamic behavior of a system. A frequency characteristic is expressed as the ratio of the Fourier transform of the output signal $Y(j\omega)$ to the Fourier transform of the input signal $X(j\omega)$. For a steady state motion of a system, the frequency characteristic is defined using the following expression [7, 8]:

$$H(j\omega) = \frac{Y(j\omega)}{X(j\omega)}. \tag{1.9}$$

A frequency characteristic is a complex quantity with the real part and the imaginary part:

$$H(j\omega) = \mathrm{Re}\{H(j\omega)\} + \mathrm{Im}\{H(j\omega)\} \tag{1.10}$$

A frequency characteristic provides a lot of information about the behavior of a system in motion. The basic data include: amplitude amplification (increase or decrease) and phase shift between the excitation and response. The data are easiest to analyze when the characteristic is represented in polar coordinates

$$H(j\omega) = \left|H(j\omega)\right| e^{j\varphi(\omega)} \tag{1.11}$$

where:

- modulus of the amplitude-frequency characteristic

$$\left|H(j\omega)\right| = \sqrt{(\mathrm{Re}\{H(j\omega)\})^2 + (\mathrm{Im}\{H(j\omega)\})^2} \tag{1.12}$$

- argument of the phase-frequency characteristic

$$\varphi(\omega) = arctg\frac{\mathrm{Im}\{H(j\omega)\}}{\mathrm{Re}\{H(j\omega)\}} \tag{1.13}$$

In the case of multi-input systems, the characteristic is represented as a matrix of characteristics.

Another type of a functional model is the time characteristic (impulse response). The transformation of the input signal into the output signal using an impulse response can be written as

$$y(t) = \int_0^\infty h(\tau)x(t-\tau)d\tau \tag{1.14}$$

where:

$y(t)$, $x(t)$ - input and output signals, respectively, $h(\tau)$ - impulse characteristic.

This characteristic was defined as a response of the system to an excitation in the form of a Dirac impulse

$$y(t) = \int_0^\infty h(\tau)\delta(t-\tau)d\tau = h(t) \tag{1.15}$$

Multi-input and multi-output systems can also be characterized as matrices of time characteristics $h(t)$.

There is an unambiguous correspondence between the frequency characteristic and the impulse response:

$$H(j\omega) = \int_0^\infty h(\tau)e^{-j\omega\tau}d\tau\ . \tag{1.16}$$

The quantities derived are used to determine the properties of linear systems for which the superposition principle is satisfied.

For non-linear systems, the (frequency, time) characteristics are dependent on the amplitude of an input signal. In the case of a system with variable coefficients, the frequency characteristic is also dependent on time. The dynamics of non-linear systems can also be analyzed on the basis of frequency and time characteristics. Qualitative identification of mechanical systems was particularly useful.

The information on the system nonlinearities cannot be lost. It is thus important:

- to analyze the frequency characteristic in steady-state motion, and
- with the assumption that the system is linear, to convert the data from frequency to time domain through a Fourier transform.

To determine the characteristics, we can use one of the following methods:

- the harmonic excitation method,
- the random noise method,
- the impulse excitation method.

The methods differ in the type of the input signal, the way the experiment is conducted and the estimation of the characteristic parameters.

2 IDENTIFICATION OF A DYNAMIC MODEL OF A VEHICLE SUSPENSION SYSTEM

A typical dynamic system described by a linear differential equation is a vehicle suspension system. A block diagram of this system is shown in Fig. 1. A vehicle with the mass *m* is suspended over a road profile by means of two elements with dynamic properties. One of them is a spring, for which the relationship between the force applied to it and its deformation is expressed with a constant coefficient *k*. The other element is the damper for which the resistance force and the displacement rate are related by means of the constant coefficient *B*.

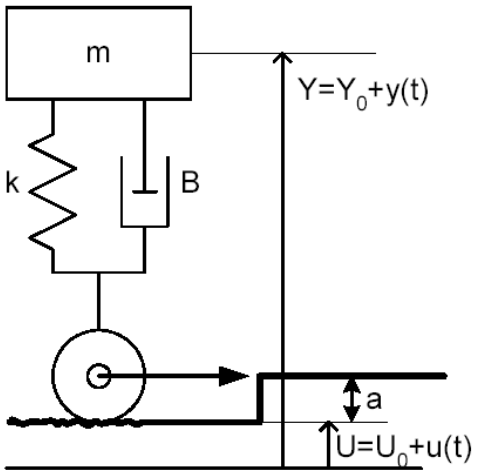


Fig.2.1. A block diagram of a vehicle suspension mechanical system

m – vehicle mass [kg],
k – spring stiffness [N/m],
B – damping coefficient [Ns/m].

After summing all the forces acting within the system and assuming a level of reference, we can write the model of the system from Fig. 2.1 with:

$$m\frac{d^2Y(t)}{dt^2}+B\left(\frac{dY(t)}{dt}-\frac{dU(t)}{dt}\right)+k(Y(t)-U(t))=mg \tag{2.1}$$

The profile and height of the obstacle do not change (let us assume that they are at level U_0). The position of the body does not change either (let us assume it to be at level Y_0). The derivatives are also equal to zero, thus they can be written as:

$$k(Y_0-U_0)=mg \tag{2.2}$$

Substituting $Y=Y_0+y(t)$, $U=U_0+u(t)$ and the formula (2.2) to Eq. (2.1), we obtain a relationship between the changes in the vehicle body position and the changes in the road profile:

$$m\frac{d^2y(t)}{dt^2}+B\frac{dy(t)}{dt}+ky(t)=B\frac{du(t)}{dt}+ku(t) \tag{2.3}$$

Equation (2.3) describes the dynamics of the vehicle suspension.

Since the form of the denominator of the system transfer function is similar to that of the oscillatory system with the following transfer function:

$$G_{osc}(s) = \frac{\varpi_n^2}{s^2 + 2\varpi_n \xi \cdot s + \varpi_n^2} \tag{2.4}$$

where ξ - damping ratio, ω_n – undamped angular frequency [rad/s] ($\omega_n = 2\pi f$), we can compare the parameters of the two systems, and then calculate *B* and *k* from the predetermined values of ξ and ω_n.

After transforming the system transfer function in such a way that the coefficient next to s^2 in the denominator is equal to 1, we have:

$$2\varpi_n \xi = \frac{B}{m},\ \varpi_n{}^2 = \frac{k}{m} \tag{2.5}$$

Despite being imprecise, this approach allows us to approximately estimate parameters *B* and *k* of the suspension system by using the easily interpretable parameters ξ and ω_n.

The response of the plant after it goes over an obstacle 0.05 m high in 60 s is obtained by activating script1. Script1 and all the subsequent scripts are created in the MATLAB program.

script1.m

```
clear all
clc
m=1000
B=20
k=200
a=0.05 % step change in the road profile
licz=[(k/m)^2];
mian=[1, B/m, (k/m)^2];
sys1=tf(licz, mian) %system definition
[y,t]=step(a*sys1);% step response
plot(t,y)
```

The measurement results for the impulse response of the plant are saved in vectors y (plant response) and time t; the response of the plant to be identified can be described with the model

$$G(s) = \frac{K}{s^2 + T \cdot s + T1} \tag{2.6}$$

Identification will require selecting adequate parameters *K*, *T* and *T1* to obtain the best convergence between the response of the plant and that of the model. This will be done by minimizing the response distance according to the least squares criterion *leastsq*. The function looks for the best convergence on the basis of the vector of deviations at the particular points, and it is a MIN function. Here, the identification is based on the output error (Fig. 1.4).

The identification process can be performed in the form of three scripts: ident.m, odp_mod.m, and wsk.m, where **ident.m** is the main script initiating identification.

ident.m

```
global a
x0=[0.01,0.01,0.01]; % initial values of K, T and T1
figure(1)
plot(t,y,t,odp_mod(x0,t)); % graphical representation of the plant and the model before identification
figure(2)
x=leastsq('wsk',x0,[],[],t,y); % convergence
plot(t,y,t,odp_mod(x,t),'r'); % graphical representation of the plant and the model after identification
```

x

The odp_mod.m script is a function generating a response of the model described by a relationship (2.6).

odp_mod.m

```
function ym=odp_mod(x,t) % Here we generate the model response.
global a
K=x(1); T=x(2);T1=x(3);
sys=tf([K],[1 T T1]);
y=step(a*sys,t);
ym=y;
```

The wsk.m function returns the vector of deviations between the response of the plant and that of the model, according to a formula (1.1)

wsk.m

```
function J=wsk(x,t,y) % For least sq we return the deviations, not the sum of squared deviations
ym=odp_mod(x,t);
J=y-ym;
plot(t,y,'b*',t,ym,'r'), drawnow; % online control
```

3 IDENTIFICATION OF FEEDBACK CONTROL SYSTEMS

Identification will be performed for a system described with the following equation:

$$y_i = z^{-1}\frac{B(z^{-1})}{A(z^{-1})}u_i + \frac{C(z^{-1})}{A(z^{-1})}e_i \tag{3.1}$$

with closed-loop feedback control where the controller is the transfer function:

$$R(z^{-1}) = \frac{u_i}{e_i^r} = \frac{Q(z^{-1})}{P(z^{-1})} \tag{3.2}$$

To identify a model in a closed-loop system described with the above equations, the following conditions need to be satisfied:

- order of the polynomial feedback transfer function where:

$$q > n-k \quad \text{or} \quad p > n \tag{3.3}$$

- order **n** and the time delay **k** of the plant are known a priori.

While **q** – order of the numerator of the controller transfer function, **p** – order of the denominator of the controller transfer function.

If the first condition is not satisfied, then identification requires:

- using controllers with changing the coefficients of amplification (passive experiment).

- using an additional identification signal with a mean value equal to zero uncorrelated with the noise being a disturbance (white noise with other parameters) at the input of the plant (active experiment).

With these two assumptions made, we can avoid ambiguity while determining the parameters of the model. If we can select the type of experiment, we should perform an active experiment. This allows us to obtain unbiased estimates of the parameters of the modeled feedback control system for a much wider range of parameters of the plant than those obtained in a passive indication test.

There are two methods of identification of closed-loop systems:

INDIRECT IDENTIFICATION, which involves identifying a model describing a system with feedback; this model is used to calculate the parameters of the model; indirect identification can only be used for steady-state linear feedbacks with known transfer functions.

DIRECT IDENTIFICATION, in which measurement results for the plant inputs and outputs obtained through an identification experiment performed on a system with feedback are analyzed like for a system without feedback.

3.1 MODEL IDENTIFICATION – AN OPEN-LOOP SYSTEM

We will identify a model where the input signal is a sine wave (*Frequency 50 [rad/sec], Sample time 0.1*). The model has a structure: nk=1, na=1, and nb=1. The diagram of the system is shown in Fig. 3.1.

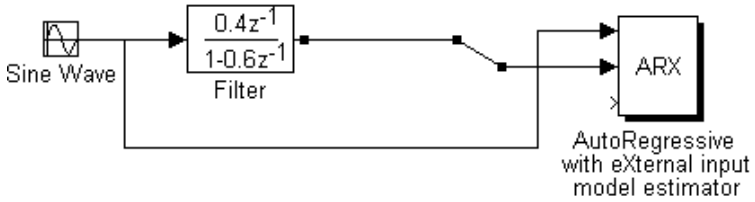


Fig. 3.1 A block diagram of identification with no disturbance (MATLAB-SIMULINK)

Next, we will perform an identical experiment but with a disturbance. Identification will be conducted according to the diagram given in Fig. 3.2.

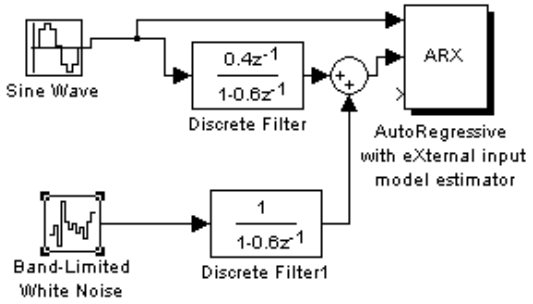


Fig. 3.2. A block diagram of identification with a noise, MATLAB-SIMULINK

3.2 MODEL IDENTIFICATION – A CLOSED-LOOP SYSTEM

Now we will use a closed-loop system with negative feedback (**k=2**) and (**k=5**). The plant will be identified using a passive method. The diagram of the system is shown in Fig. 3.3.

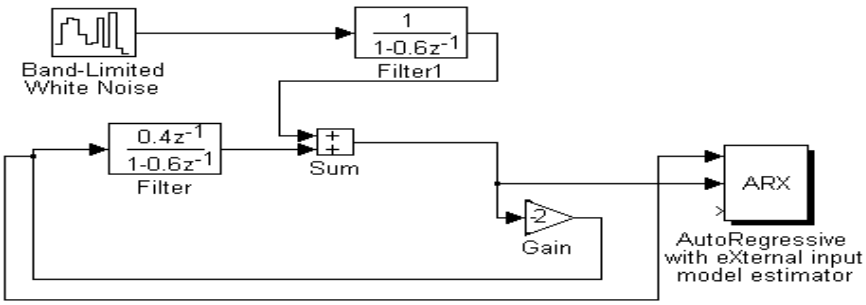


Fig. 3.3. A block diagram of identification of a closed-loop system

In feedback, we have used two different excitations; thus, it is possible to carry out identification of the plant model.

The data are used to determine the correct model coefficients by means of the equation of identification error:

$$\varepsilon_i = y_i + a_1 y_{i-1} - b_0 u_{i-1} \tag{3.1}$$

On the right side of the equation we add an equation of feed forward:

$$mu_{i-1} + mk_r y_{i-1} = 0 \tag{3.2}$$

$$\varepsilon_i = y_i + a_1 y_{i-1} - b_0 u_{i-1} + mu_{i-1} + k_r m y_{i-1} = y_i + (a_1 + k_r m) y_{i-1} - (b_0 - m) u_{i-1} \tag{3.3}$$

which leads to the following system of equations:

$$\begin{cases} \hat{a}(\mathrm{k_r} = 2) = a_1 + k_r m_1 \\ \hat{b}(\mathrm{k_r} = 2) = b_0 - m_1 \\ \hat{a}(\mathrm{k_r} = 5) = a_1 + k_r m_2 \\ \hat{b}(\mathrm{k_r} = 5) = b_0 - m_2 \end{cases} \text{-->} \begin{cases} 0 = a_1 + 2m_1 \\ \beta_0 = b_0 - m_1 \\ 0 = a_1 + 5m_2 \\ \beta_0 = b_0 - m_2 \end{cases} \tag{3.4}$$

By initiating the m-file below in MATLAB we can solve Eqs. (3.4) and obtain model coefficients a_1 and b_0.

```
b02=  %here write the result obtained for gain k=2
b05=  % here write the result obtained for gain k=5
A=[1 0 2 0;0 1 -1 0;1 0 0 5;0 1 0 -1];
B=[0;b02;0;b05];
x=A\B % a1 b0 m1 m2
```

3.3 ACTIVE DIRECT IDENTIFICATION

The next stage is to perform an active experiment with an auxiliary identification signal. We need to consider several cases:

- when signal S_i is correlated with disturbance e (level S_i is equal to level e; level S_i is greater than level e)
- when signal S_i is not correlated with disturbance e (level S_i is equal to level e; level S_i is greater than level e).

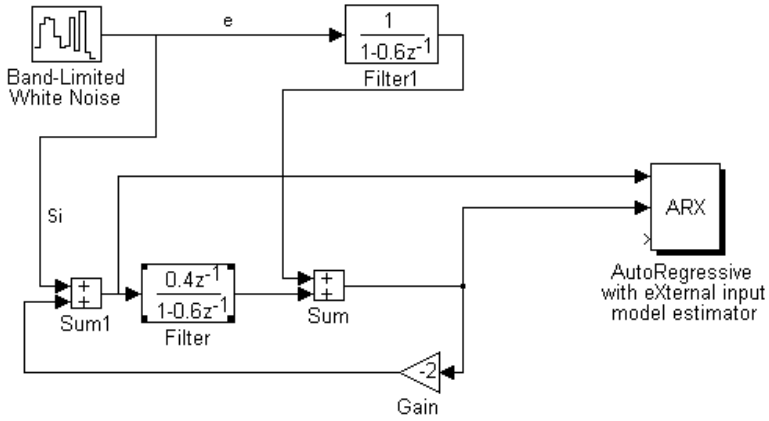


Fig. 3.4. A diagram of a system where signal S_i is correlated with disturbance e

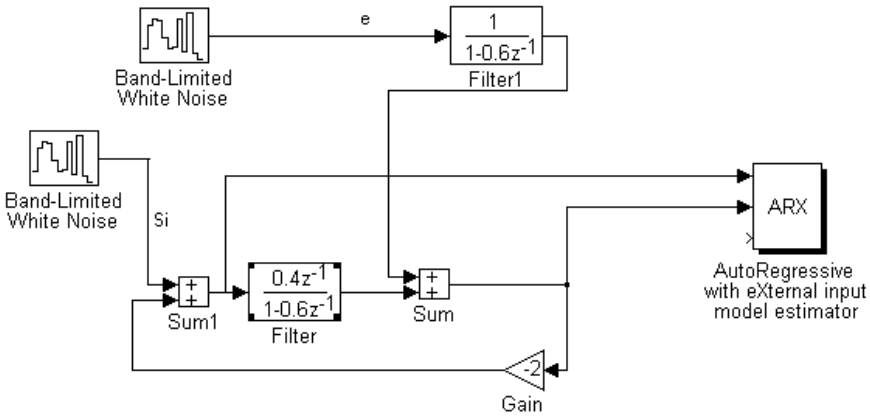


Fig. 3.5. A diagram of a system where signal S_i is not correlated with disturbance e

3.4 INDIRECT IDENTIFICATION

Now we will carry out indirect identification of the plant. Indirect identification involves identifying a model describing the plant with feedback.

$$K_z(z^{-1}) = \frac{\dfrac{b \cdot z^{-1}}{1+az^{-1}}}{1+\dfrac{k_r b z^{-1}}{1+az^{-1}}} = \frac{bz^{-1}}{1+(k_r b+a)z^{-1}} = \frac{b_m z^{-1}}{1+a_m z^{-1}} \tag{3.5}$$

For: $k_r = 2$, b=0.4, and a=-0.6, we obtained $b_m = 0.4$ and $a_m = 0.2$. The system diagram is shown in Fig. 3.6.

Pulse Generator(Amplitude 1, Period 0.1, Pulse Width 50, Phase delay 0)
ARX (na=1, nb=1, nk=1, Sample time 0.1)

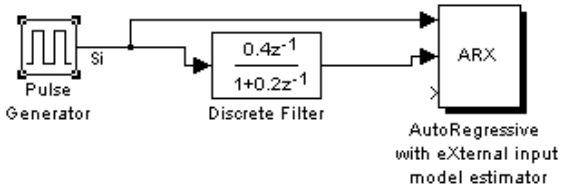


Fig. 3.6. A diagram of identification of the plant

The identification process will lead to the determination of parameters b_m and a_m. To obtain the model parameters b and a, it is necessary to use the relationships (3.5) from which it is clear that: $b = b_m$, and $a = a_m - k_r b$.

4 ESTIMATION AND VERIFICATION OF THE MODEL PARAMETERS - *SYSTEM IDENTIFICATION TOOLBOX*

Before the identification experiment commences, we can obtain some initial information, applying a unit step input to the plant. The response will enable us to determine approximate duration of the transition process, the real delay, and the noise spectrum of the output signal, which can be used to determine the desired discretization step size, or even parameters of the lowpass filter (the **idfilt** command).

If there are no real experiment data available, we can carry out an identification session with MATLAB using data saved in *mat*—files called: dryer2, iddata1, iddata2...iddata8 [9].

4.1 ESTIMATION OF THE MODEL STRUCTURE AND PARAMETERS

```
»load dryer2
```

We load the data into the workspace and enter the command:

```
»who
```

to check the names of the variables loaded: input: *u2*, output: *y2*, time discretization step size *T*= 0.08 s. We combine the data into one matrix:

```
»dane=iddata(y2, u2, 0.08)
```

Let us look at our measurement data. We select 100 central measurement results:

```
»plot(dane(300: 400))
```

We divide the set of the measurement data into two equal parts: a set for estimation called *ze* and a set for verification called *zw*. If the number of all measurements is *N*= 1000, then:

```
»ze=dane(1:500)
```

```
»zw=dane(501:1000)
```

We remove the trend (constant constituent):

```
»ze=detrend(ze)

»zw=detrend (zw)
```

Some information about the plant can be searched for in the impulse response:

```
»impulse(ze, ' sd' , 3)
```

From the impulse response it is evident with a probability of more than 99% that a real delay is of an order of magnitude of two steps. Let us assume that we select an autoregressive ARX model and search for its structure. By means of the arxstruc command, which calculates the loss function V, we search a series of structures of the order of 1:5, with delays from l to 3:

```
»V=arxstruc(ze, zw, struc(1:5,1:5,1:3))
```

The **selstruc** function will select the structure with the smallest loss function, and therefore will theoretically best fit the measurement data. The function allows us to select the optimization criterion, which is entered as a variable e.g. 'AIC' or 'MDL' (default variable is the Akaike criterion):

```
»nn=selstruc(V,'AIC')
```

On the screen we will have *nn* = 4 5 2, which indicates that among the 76 models searched, the model with four poles, four zeros and with real delay *nk = 2* has the smallest loss function. Note the difference in the notation of polynomials *A(q)* and *B(q).* Remember that we do not enter the order of the numerator or denominator; we enter the number of calculated parameters, and in the denominator, one parameter is always known: $a_0 = 1$:

```
»model0=arx(ze,nn)
```

The model is unfavorable. Since the numerator order is equal to the denominator order, the model does not meet the causality condition. Thus, we select:

```
» model1=arx(ze,[4 , 4 , 2 ])
```

But is such a high order of an ARX model necessary? Perhaps a lower order model will also be able to satisfactorily represent the dynamic properties of the plant. We create :

```
»model2=arx(ze, [3 , 3 , 2 ])

»model3=arx(ze, [2 , 2 , 2 ])
```

For comparison purposes, we will try to calculate other models. The **pem** model is a general linear state space model, and its main advantage is that it does not require a priori declaration of the model order. The structure is searched for automatically using the prediction error method.

```
»model4=pem(ze);
```

The calculation result appears on the screen, but it is important to note that the model is not one of the simplest. Let us now check whether model ARMAX with a low (first) order filter will not give good convergence, and let us calculate:

```
»model5=armax(ze,[2 2 1 2 ] )
```

We have calculated 5 models. Now we will verify them.

4.2 MODEL VERIFICATION

Models are compared in the time and frequency domains. First, we check visually the convergence between the model output and the real measurement. For comparison purposes, we select a certain central part of the data set:

```
»compare(zw(200:250),model1,model2,model3,model4,model5)
```

The information in the box comprises the percentage of the convergence between the model and the data, taking into account the integral of squared error. For each model, we can visually check the uncertainty of the model step response by observing the limits of the parameter scatter in the range equal to the triple value of standard deviation:

```
»step(model1, 'sd',3)
»step(model2, 'sd',3)
»step(model3, 'sd',3)
»step(model4, 'sd',3)
»step(model5, 'sd',3)
```

and we can check if the residuals satisfy the prediction error whiteness test. Samples must be within the predetermined range:

```
»resid(zw, model1)
»resid(zw, model2)
»resid(zw, model3)
»resid(zw, model4)
»resid(zw, model5)
```

Now, we will compare the models in the frequency domain. We will calculate the spectral transfer function of the plant using a set of measurement data:

```
»Gp=spa(dane);
```

and compare it with the amplitude-phase characteristics (Bode plots) of the models:

```
»bode(Gp, model1,model2,model3,model4,model5)
```

From the graph it is evident that the models have characteristics fairly similar to the characteristic calculated from real measurement data; however, at higher frequencies, it is not possible to detect all the subtleties of the plant dynamics. The plot shows unfavorable effects of model overdimensioning:

```
»nyquist(Gp,model3)
```

The Nyquista plot shows the spectra transfer function of a selected model and, by means of ellipses, model confidence interval corresponding to the triple value of the standard deviation. The frequency intervals corresponding to a 10-fold change are separated by asterisks. We check if zeros and poles of the transfer function do not coincide or if there are any unstable poles present:

```
»zpplot(model1)
»zpplot(model2)
»zpplot(model3)
```

```
»zpplot(model4)

»zpplot(model5)
```

In the graph we can see that the pair of zeros and poles lie close to each other. Perhaps a lower order model will be sufficient. We repeat the calculations and perform another verification.

5 IDENTIFICATION OF THE PID CONTROLLER PARAMETERS

5.1 IDENTIFICATION IN MATLAB-SIMULINK

The unlimited optimization problems can be solved using the **`fmins('nazwa_funkcji',x0,opcje)`** function (**`fminsearch()`** in version 7 of the program).

The **`nazwa_funkcji`** parameter defines the name of the file determining the target function. This function should be labeled as **`function f=nazwa_funkcji(x)`**, where **`x`** is the vector of decision variables. The **`x0`** parameter contains initial values of the decision variables. The **`opcje`** parameter provides additional options determining, for instance, the halting condition. The **`fmins`** function returns the optimal vector of the parameters.

If SIMULINK is used to determine the function values, the procedure is as follows.

1) Decision variables used in the simulation need to be defined in all the m-files in which they occur as global variables by means of the **`global`** directive, e.g.

```
global a b c
```

2) In the **`nazwa_funkcji`** file, before a simulation is invoked, we need to rewrite the decision variables from vector **`x`** to global variables used in the simulation, for example,

```
a=x(1);
b=x(2);
c=x(3);
```

3) In SIMULINK, the simulation of a model is invoked with the following command

```
sim('nazwa_modelu')
```

4) We must make sure that after the simulation is completed, the necessary results are visible in the MATLAB environment and can be used by the **`nazwa_funkcji`** function. For this purpose we use the **`To Workspace`** block.

5.2 SELECTION OF OPTIMAL PARAMETERS OF A PID CONTROLLER

To determine the optimal parameters of the controller, we use the most general method of plant identification, the so called output error method (Fig. 1.4). The technique involves comparing the plant responses with the model response to a predetermined signal. The model parameters are selected to obtain the highest possible convergence between the output responses of the plant and those of the model

First, we define the concept of the quality of control represented by the target function. Most frequently, control quality is defined by the integral of squared error (1.3) or the integral of absolute error (1.4) e at a unit step input of the setpoint signal.

The criterion defined is dependent on the controller parameters, which play the role of decision variables. For a PID controller in the form $u(t) = K_p e(t) + K_i \int_0^T e(t)dt + K_d \dot{e}(t)$ the decision variables are K_p, K_i and K_d.

We will determine optimal parameters of a PID controller for the plant described with the following transfer function:

$$\frac{k}{T^3 s^3 + 3T^2 s^2 + 3Ts + 1} \tag{5.1}$$

Identification will be made for a target function defined as the integral of squared error (1.3) and the integral of the modulus error (1.4). For this purpose, we will create a universal diagram of a closed-loop system in SIMULINK called **reg.mdl**.

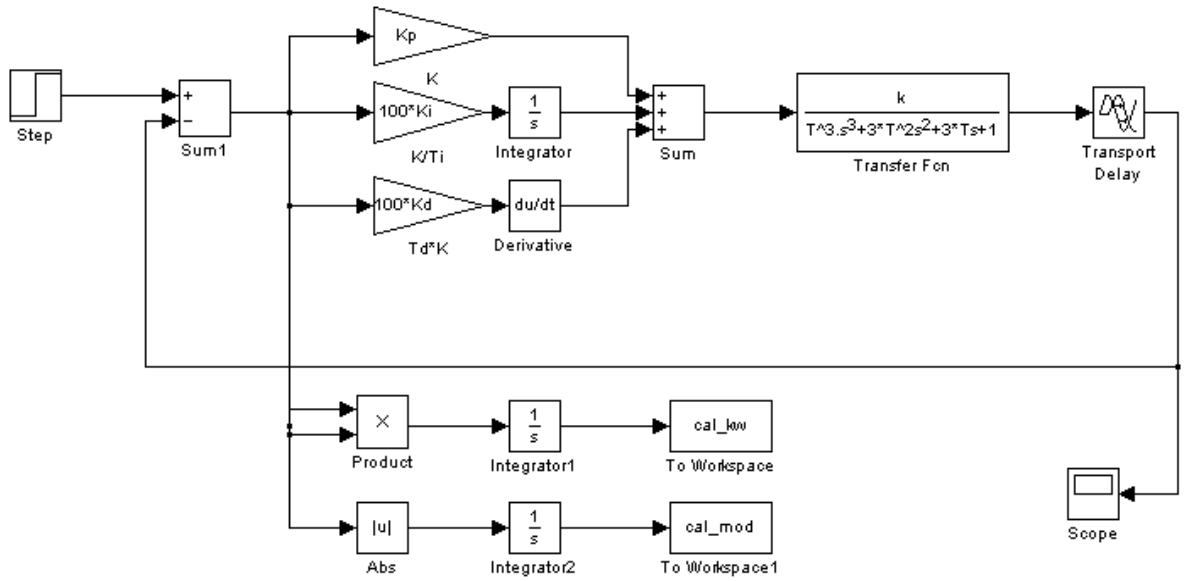


Fig. 5.1 Diagram of a closed-loop system for transfer function (5.1) with a PID controller (*reg.mdl*)

Step: *Step time 1; Initial value 0; Final value 1; Sample time –1;*
Transport Delay: *Time delay To; Initial input 0; Initial buffer size 1024; Pade order 0;*
Integrator, Integrator1, Integrator2: *Initial condition 0.*
To Workspace: *Save format Array.*

To invoke the identification process, we need additional files: **start.m** and **licz.m.** If the target function is to have the form of the integral of the modulus error, in the scripts below, we will use the *cal_mod* variable; if, however, the function is to have the form of the integral of squared error, we will use the *cal_kw* variable.

start.m

```
clear all
global Kp Ki Kd cal_mod
k=1;
T=0.2;
To=0.2;
Kp=1;
Ki=0;
Kd=0;
x=[Kp Ki Kd];
xr=fmins('licz',x)
```

licz.m

```
function i=licz(x)
```

```
global Kp Ki Kd cal_mod
Kp=x(1);
Ki=x(2);
Kd=x(3);
sim reg;
i=cal_mod(length(cal_mod));
```

Identification begins after the **start** command is entered in the MATLAB command line.

By comparing the quality criteria for the particular identification processes, we can establish which of these settings are better. A lower value of the target function implies better identified settings of the controller.

In this experiment, we have shown that it is crucial to define a target function in order to correctly determine the settings of the controller.

6 IDENTIFICATION OF MODEL PARAMETERS FOR A HYDRAULIC SYSTEM

6.1 A MATHEMATICAL MODEL

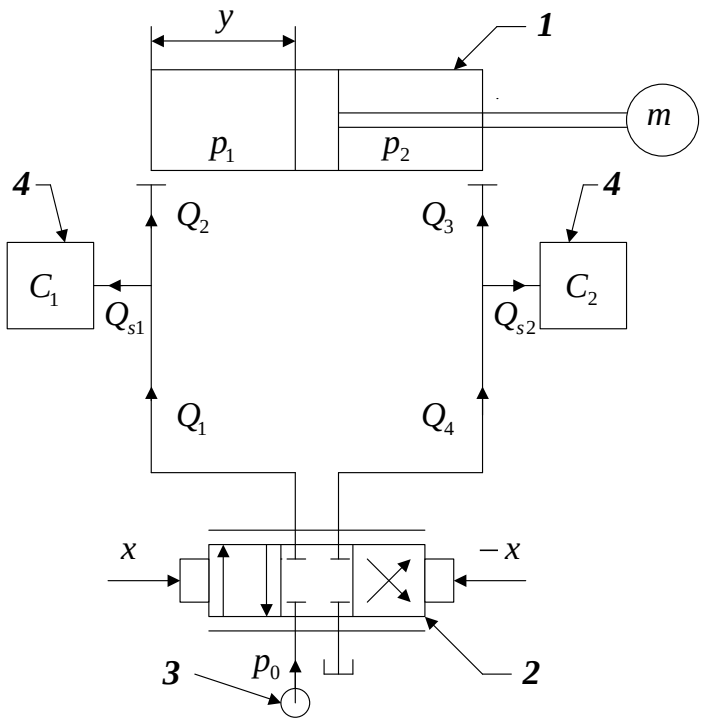


Fig. 6.1. A system with a hydraulic actuator

A mathematical model of a hydraulic system can be written using the following equations:

$$v = \dot{y}\,, \tag{6.1}$$

$$m\dot{v} + Bv = F_1 p_1 - F_2 p_2\,, \tag{6.2}$$

$$Q_1 = \alpha x\sqrt{|p_0 - p_1|}H(x) + \alpha x\sqrt{|p_1|}H(-x)\,, \tag{6.3}$$

$$\frac{V_1}{E_c}\dot{p}_1 = Q_1 - F_1 v - K_v(p_1 - p_2), \tag{6.4}$$

$$Q_4 = -\alpha x\sqrt{|p_2|}H(x) - \alpha x\sqrt{|p_0 - p_2|}H(-x), \tag{6.5}$$

$$\frac{V_2}{E_c}\dot{p}_2 = Q_4 + F_2 v + K_v(p_1 - p_2), \tag{6.6}$$

where: Q_1, Q_4 – rates of the fluid flow, α – coefficient of the rate of flow through the distributor orifice at full flow, p_0 – supply pressure, $H(x)$ – step Heaviside function (1 for $x \geq 0$ and 0 for $x < 0$), V_1 – volume of the vessel with pressure p_1, V_2 – volume of vessel with pressure p_2, E_c – elasticity coefficient, K_v – volumetric loss coefficient, v – piston speed, m – mass of the piston and related elements, B – viscous friction coefficient, p_1, p_2 – pressures on the piston, F_1, F_2 – surface areas of the actuator piston.

In further simulations we will assume the following nominal values of the parameters and the initial conditions of the simulation (Table1 and Table 2).

Table 1. Parameters of the hydraulic system

Symbol	Unit	Value
α	m^5/Ns	$1\cdot10^{-6}$
p_o	N/m^2	$4.6\cdot10^{7}$
K_v	m^5/Ns	$5\cdot10^{-12}$
F_1	m^2	$3\cdot10^{-2}$
F_2	m^2	$3\cdot10^{-2}$
E_c	N/m^2	$1.2\cdot10^{9}$
V_1	m^3	0.016
V_2	m^3	0.016
B	Ns/m	$1\cdot10^{5}$
m	kg	$5\cdot10^{2}$

Table 2. Initial simulation conditions

Quantity	Symbol	Unit	Value
Displacement of the actuator piston	$y(0)$	m	0
Actuator piston speed	$\dot{y}(0)$	m/s	0
Input pressure	$p_1(0)$	N/m^2	$\frac{1}{2}p_o$
Output pressure	$p_2(0)$	N/m^2	$\frac{1}{2}p_o$

6.2 IDENTIFICATION METHOD

Identification will be carried out using the classic output error method. Model parameters are determined according to the diagram presented in Fig. 6.2. The fundamental idea is to use a model with variable parameters. Model identification based on this method requires solving a system of differential equations at each iteration. Finding the right parameters is time-consuming and does not always provide good results.

In this method, an input signal $x(t)$ is supplied simultaneously to the system and the model, then their responses are compared, and the so-called output error $e(t)$ is defined. This error is used to determine the identification quality criterion, which will be in the form of the integral of squared error [10, 11].

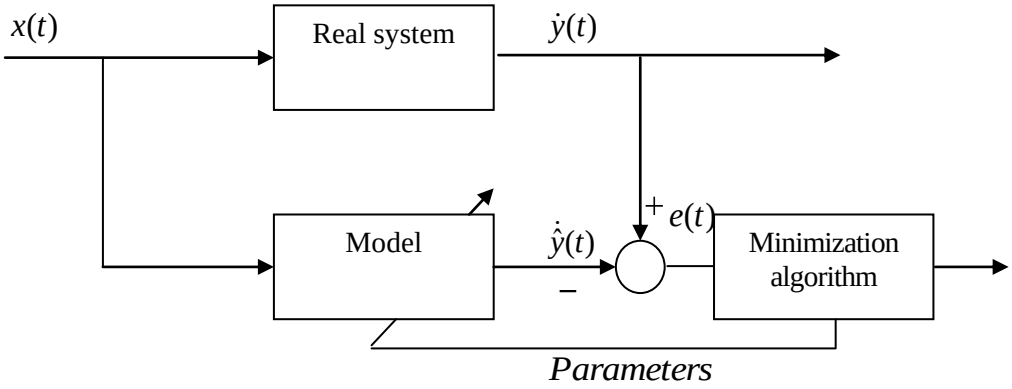


Fig. 6.2 A block diagram of the identification algorithm.

Minimizing the quality criterion, we search for such values of the model parameters, for the assumed structure of the plant, at which the identification quality criterion reaches a minimum.

Identification of many parameters required to build a target function is more accurate when the piston speed is used as the output signal.

6.3 SELECTION OF INPUT SIGNALS

Input signals should ensure sufficient activation of the plant to identify its parameters. In practice, the excitation signals are in the form of step signals or a sum of sine signals varying in frequency. As the identified system is nonlinear, the input signal is assumed to be a sequence of step signals with an increasing amplitude, as shown in Fig. 6.4. With this form of the excitation signal, the system operates in a wide range of variability of the piston speed, which provides us with sufficient information to carry out identification.

If the system to be identified is unstable, we may find it difficult to control the plant when the control is in an open-loop system. In such a case, identification can be performed in an closed-loop system using a properly selected controller. The structure of such an identification system is shown in Fig. 6.3. Because of the system instability, identification will also be performed for a closed-loop system.

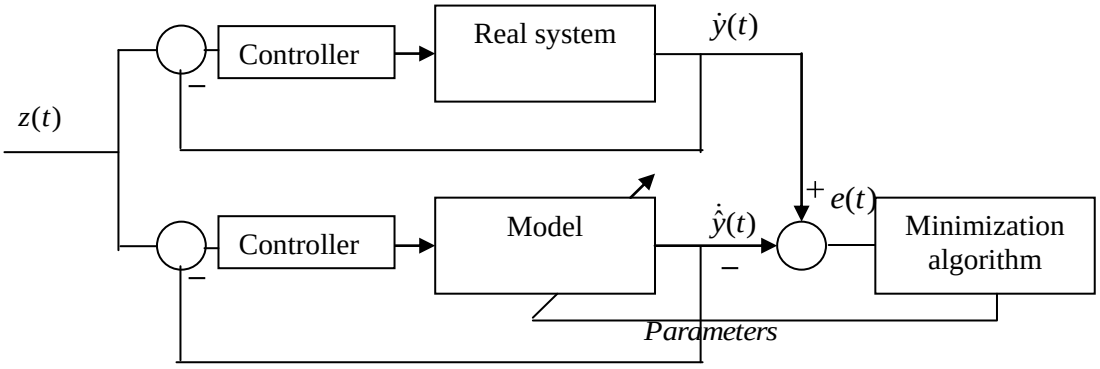


Fig. 6.3 A block diagram of the algorithm for identification in a closed-loop system.

To obtain an excitation of the system similar to that of an open-loop system, we assume a setpoint saw-like signal (Fig. 6.5) and PD-type controllers in the following form:

$$x_r = K*(z-y) - K_d * \dot{y}\,. \tag{6.7}$$

6.4 SELECTION OF PARAMETERS REQUIRED FOR IDENTIFICATION

Another problem is the selection of parameters required for identification. The procedure is called system parametrization. The parameters should be selected in such a way that their identification is unequivocal. In such a case, we say that the system is identifiable. The most natural method of parametrization is to assume that the search parameters are the subsequent parameters of the equations: M, B, F_1, F_2, K_V, E_C, V_1, V_2, p_0, and α. However, even a superficial analysis shows that such a model is not identifiable. We can reduce the number of parameters to be identified. For instance, instead of parameters E_C, V_1 and V_2 it is sufficient to assume parameters $\frac{V_1}{E_C}$ and $\frac{V_2}{E_C}$.

In fact, we can show that the number of parameters can still be reduced if we divide the equations by *m*, which gives us the following parameters: $\theta 1 = \frac{V_1}{E_C m}$, $\theta 2 = \frac{F_1}{m}$, $\theta 3 = \frac{K_V}{m}$, $\theta 4 = \frac{\alpha}{m}$, $\theta 5 = \frac{V_2}{E_C m}$, $\theta 6 = \frac{F_2}{m}$, $\theta 7 = \frac{B}{m}$, and $\theta 8 = p_0$.

With parameters selected in this way, the equations of the model will be as follows:

$$\dot{v} + \theta 7 \cdot v = \theta 2 \cdot p_1 - \theta 6 \cdot p_2 \tag{6.8}$$

$$Q_1 = \theta 4 \cdot x \cdot \sqrt{|p_0 - p_1|} \cdot H(x) + \theta 4 \cdot x \cdot \sqrt{|p_1|} \cdot H(-x) \tag{6.9}$$

$$\theta 1 \cdot \dot{p}_1 = Q_1 - \theta 2 \cdot \dot{y} - \theta 3 \cdot (p_1 - p_2) \tag{6.10}$$

$$Q_4 = -\theta 4 \cdot x \cdot \sqrt{|p_2|} \cdot H(x) - \theta 4 \cdot x \cdot \sqrt{|p_0 - p_2|} \cdot H(-x) \tag{6.11}$$

$$\theta 5 \cdot \dot{p}_2 = Q_4 + \theta 6 \cdot \dot{y} + \theta 3 \cdot (p_1 - p_2) \tag{6.12}$$

The MATLAB `fmins` function, which minimizes the quality criterion, will be used for identification. The function uses the optimization method based on a Nelder-Mead downhill simplex.

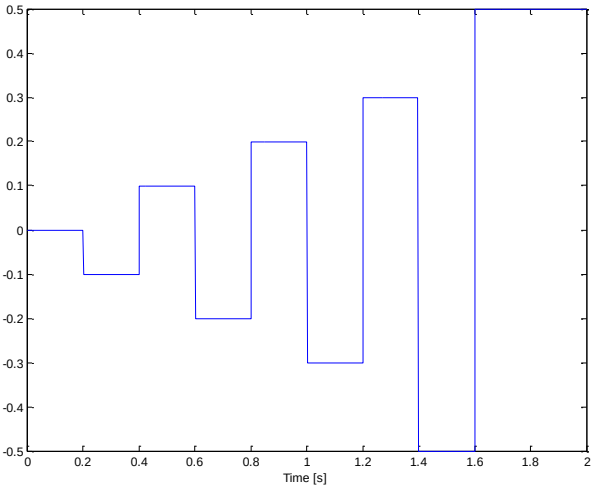


Fig. 6.4 Input signal $x(t)$ for an open-loop system

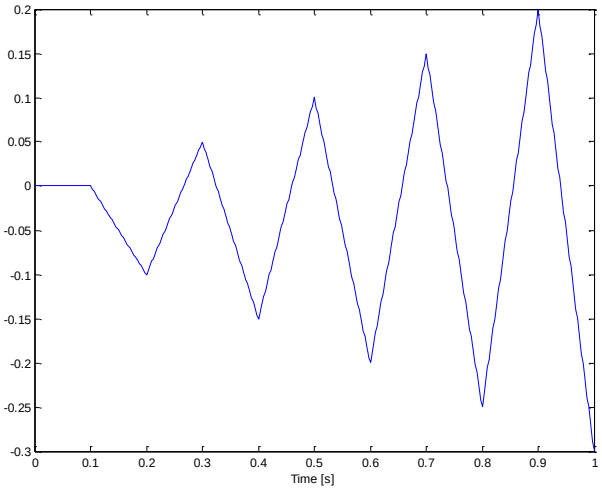


Fig. 6.5 Setpoint signal $z(t)$ for a closed-loop system

6.5 STAGES OF THE IDENTIFICATION PROCESS – CLASSES

Identification will be carried out in an open- or closed-loop system. In either case, there are three major stages:

1. modeling and simulation of the real system (Eqs. 6.1-6.6) to generate signals required for identification (in this case, we need an input signal and output signal (piston speed)),
2. creating an identification function,
3. checking the results.

6.5.1 Open-loop system identification

To model and simulate the system of equations 6.1-6.6 for the input signal shown in Fig. 6.4, we need to build the following functions:

funh.m – a Heaviside function,

syg_zad.m – a function determining the setpoint signal using the data from the krok.m file,

krok.m – a file containing the setpoint signal,
funkrokr.m – a file with the model equations 6.8-6.12,
dane.m – known parameters, as shown in Table 1,
start.m – a start file which defines the necessary variables and the method to solve the differential equations (ode45 – 4-5th order Runge-Kutta methods).

Equations 6.8-6.12 with the calculated parameters were used as the equations of the real system, because Eqs. 6.1-6.6 are equivalent to Eqs. 6.8-6.12.

funh.m

```
function [h]=funh(g)
if g<0
h=0;
else
h=1;
end
```

syg_zad.m

```
N=2;
k=0;
for i=0:0.002:N
k=k+1;
t(k)=i;
xx(k)=krok(i);
end
plot(t,xx)
xlabel('time [s]')
ylabel('distributor plunger displacement')
print -dmeta wykres_xx.wmf
```

krok.m

```
function [y]=krok(t)
if t<=0.2
y=0;
elseif t<=0.4
y=-0.1;
elseif t<=0.6
y=0.1;
elseif t<=0.8
y=-0.2;
elseif t<=1
y=0.2;
elseif t<=1.2
y=-0.3;
elseif t<=1.4
y=0.3;
elseif t<=1.6
y=-0.5;
elseif t<=2
y=0.5;
end
```

funkrokr.m

```
function [Dx]=funkrokr(t,x)
global T1 T2 T3 T4 T5 T6 T7 T8 z

z=krok(t);

Dx=[x(2);
(T2*10^-4)*x(3)-(T6*10^-4)*x(4)-(T7*10^3)*x(2);
-(1/(T1*10^-14))*((T3*10^-14)*x(3)-(T3*10^-14)*x(4)+(T2*10^-4)*x(2)-
(T4*10^-8)*sqrt(x(3))*funh(-z)*z-(T4*10^-8)*sqrt((T8*10^7)-x(3))*funh(z)*z);
-(1/(T5*10^-14))*(-(T3*10^-14)*x(3)+(T3*10^-14)*x(4)-(T6*10^-
4)*x(2)+(T4*10^-8)*sqrt((T8*10^7)-x(4))*funh(-z)*z+(T4*10^-
8)*sqrt(x(4))*funh(z)*z);];
```

dane.m

```
m=5;
B=1;
F1=3;
F2=3;
Kv=5;
alfa=1;
po=4.6;
Ec=1.2;
V1=16;
V2=16;
T1=V1/(Ec*m)
T2=F1/m
T3=Kv/m
```

start.m

```
global  T1 T2 T3 T4 T5 T6 T7 T8 z po
dane;
[t,x]=ode45('funkrokr',[0:0.002:2],[0;0;0.5*po*10^7;0.5*po*10^7]);
yz=x(:,2);
save predkosc t yz;
subplot(3,1,1);
hold on
grid on
plot(t,x(:,1),'LineWidth',2)
xlabel('time [s]')
ylabel('displacement [m]')
subplot(3,1,2);
hold on
```

```
T4=alfa/m
T5=V2/(Ec*m)
T6=F2/m
T7=B/m
T8=po
grid on
plot(t,x(:,2),'LineWidth',2)
xlabel('time [s]')
ylabel('speed [m/s]')
subplot(3,1,3);
hold on
grid on
plot(t,x(:,3),'r','LineWidth',2)
subplot(3,1,3);
plot(t,x(:,4),'g','LineWidth',2)
legend('pressure p1','pressure p2')
xlabel('time [s]')
ylabel('pressure [N/m^2]')
hold off
print -dmeta wykres.wmf
```

To start the simulation, we enter the command
>>start
and to obtain a plot of the setpoint signal, we use the command
>>syg_zad.

Correct operation of the above functions guarantees obtaining the **predkosc.mat** file, which contains data concerning time (t) and actuator piston speed (yz). Additionally, we obtain plots saved in the "wmf" format.

After collecting signals necessary for identification, we can start the identification process. For this purpose, we need to write appropriate functions:

identyfikacja.m – function that initiates the identification process; it contains information on the optimization method (fmins),

index.m – function responsible for saving and identifying the parameters,

parametry.m – function containing the vector of the start parameters and the necessary data.

identyfikacja.m

```
global T1 T2 T3 T4 T5 T6 T7 T8 t1 t2 t3 t4 t5 t6 t7 t8 IND i yz po
i=1;
parametry;
load predkosc t yz;
fmins('index',v)
```

index.m

```
function [ind]=index(v)
global T1 T2 T3 T4 T5 T6 T7 T8 t1 t2 t3 t4 t5 t6 t7 t8 IND i yz po
T1=v(1);
T2=v(2);
T3=v(3);
T4=v(4);
T5=v(5);
T6=v(6);
T7=v(7);
T8=v(8);
[t,x]=ode45('funkrokr',[0:0.002:2],[0;0;0.5*po*10^7;0.5*po*10^7]);
y=x(:,2);
error=(yz-y).^2;
ind=sum(error)
t1(i)=T1;
t2(i)=T2;
t3(i)=T3;
```

```
t4(i)=T4;
t5(i)=T5;
t6(i)=T6;
t7(i)=T7;
t8(i)=T8;
IND(i)=ind;
i=i+1;
save aktualne
```

parametry.m

```
v=[2.5 0.6 0.9 0.18 2.5 0.5 0.18 4.5]
po=4.6;
```

The identification process is initiated by giving the >>identyfikacja command.

The process can take even a dozen minutes. During the identification process, the identification indicator (ind) that appears on the screen will inform us about the error value. Correctly, the indicator should decrease. During the process, all the parameters are saved into the **aktualne.mat** file.

As soon as identification is completed, it is necessary to check and estimate the whole process. We check the parameters and the coincidence of the speed signals (for a system with actual and identified parameters).

The plots can be viewed by executing the following file

sprawdzenie.m

```
clear all;
global po
parametry;
load predkosc t yz;
load aktualne.mat
T1=t1(i-1);
T2=t2(i-1);
T3=t3(i-1);
T4=t4(i-1);
T5=t5(i-1);
T6=t6(i-1);
T7=t7(i-1);
T8=t8(i-1);
[t,x]=ode45('funkrokr',[0:0.002:2],[0;0;0.5*po*10^7;0.5*po*10^7]);
y=x(:,2);
plot(t,yz,t,y);
xlabel('time [s]')
ylabel('speed [m/s]')
print -dmeta yz_y.wmf
plot(t,yz-y);
xlabel('time [s]')
ylabel('difference in speed [m/s]')
print -dmeta roz_yz_y.wmf
plot(t1);
print -dmeta t1.wmf
plot(t2);
print -dmeta t2.wmf
plot(t3);
print -dmeta t3.wmf
plot(t4);
print -dmeta t4.wmf
plot(t5);
```

```
print -dmeta t5.wmf
plot(t6);
print -dmeta t6.wmf
plot(t7);
print -dmeta t7.wmf
plot(t8);
print -dmeta t8.wmf
plot(IND);
print -dmeta blad.wmf
```

The process is initiated by giving the command
>>sprawdzenie

The sprawdzenie.m file will generate plots comparing the appropriate signals and representing the parameter estimation. Additionally, all the plots will be saved in the "wmf" format.

6.5.2 Closed-loop system identification

We perform all the operations in the same order as for an open-loop system.

To model and simulate the system of equations 6.1-6.8 with PD controllers for a setpoint signal (Fig. 6.5), we need to build the following functions:

funh.m – Heaviside function,
syg_zad.m – function determining the setpoint signal based on the information from the krok.m file,
krok.m – file containing the setpoint signal,
funkrokr.m – file with the equations of the model 6.8-6.12 and a PD controller,
dane.m – known parameters, as shown in Table 1,
start.m – start file which determines the necessary variables and the method of solution to differential equations.

funh.m

```
function [h]=funh(g)
if g<0
h=0;
else
h=1;
end
```

syg_zad.m

```
N=1;
k=0;
for i=0:0.002:N
k=k+1;
t(k)=i;
yy(k)=krok(i);
end
plot(t,yy)
xlabel('time [s]')
ylabel('sygna | zadany')
print -dmeta wykres_yy.wmf
```

krok.m

```
function [y]=krok(t)
if t<=0.1
y=0;
elseif t<=0.2
y=-t+0.1;
elseif t<=0.3
y=1.5*t-0.4;
elseif t<=0.4
y=-2*t+0.65;
elseif t<=0.5
y=2.5*t-1.15;
elseif t<=0.6
y=-3*t+1.6;
elseif t<=0.7
```

funkrokr.m

```
function [Dx]=funkrokr(t,x)
global T1 T2 T3 T4 T5 T6 T7 T8 z K Kd

z=K*(krok(t)-x(1))-Kd*x(2);

Dx=[x(2);
(T2*10^-4)*x(3)-(T6*10^-4)*x(4)-(T7*10^3)*x(2);
-(1/(T1*10^-14))*((T3*10^-14)*x(3)-(T3*10^-14)*x(4)+(T2*10^-
4)*x(2)-(T4*10^-8)*sqrt(abs(x(3)))*funh(-z)*z-(T4*10^-
8)*sqrt(abs((T8*10^7)-x(3)))*funh(z)*z);
-(1/(T5*10^-14))*(-(T3*10^-14)*x(3)+(T3*10^-14)*x(4)-(T6*10^-
4)*x(2)+(T4*10^-8)*sqrt(abs((T8*10^7)-x(4)))*funh(-z)*z+(T4*10^-
8)*sqrt(abs(x(4)))*funh(z)*z);];
```

```
y=3.5*t-2.3;
elseif t<=0.8
y=-4*t+2.95;
elseif t<=0.9
y=4.5*t-3.85;
elseif t<=1
y=-5*t+4.7;
end
```

dane.m

```
m=5;
B=1;
F1=3;
F2=3;
Kv=5;
alfa=1;
po=4.6;
Ec=1.2;
V1=16;
V2=16;
T1=V1/(Ec*m)
T2=F1/m
T3=Kv/m
T4=alfa/m
T5=V2/(Ec*m)
T6=F2/m
T7=B/m
T8=po
K=1400
Kd=0.1
```

start.m

```
global  T1 T2 T3 T4 T5 T6 T7 T8 z K Kd po
dane;
[t,x]=ode45('funkrokr',[0:0.002:1],[0;0;0.5*po*10^7;0.5*po*10^7]);
yz=x(:,2);
save predkosc t yz;
subplot(3,1,1);
hold on
grid on
plot(t,x(:,1),'LineWidth',2)
xlabel('time [s]')
ylabel('displacement [m]')
subplot(3,1,2);
hold on
grid on
plot(t,x(:,2),'LineWidth',2)
xlabel('time [s]')
ylabel('speed [m/s]')
subplot(3,1,3);
hold on
grid on
plot(t,x(:,3),'r','LineWidth',2)
subplot(3,1,3);
plot(t,x(:,4),'g','LineWidth',2)
legend('pressure p1',' pressure p2')
xlabel(' time [s]')
ylabel(' pressure [N/m^2]')
hold off
print -dmeta wykres.wmf
```

The simulation is initiated after entering the
>>start command.
To obtain a plot of the setpoint signal, we need to use the
>>syg_zad command.

Correct operation of the above functions guarantees obtaining the **predkosc.mat** file, which contains data in the form of time (t) and the actuator piston speed (yz). Additionally, we will obtain plots saved in the "wmf" format.

The identification process is initiated after the required signals are collected. For this purpose, we need to write correct functions and scripts:

identyfikacja.m – entry function to initiate identification, containing data on the optimization method (fmins),

index.m – function responsible for recording and identifying the parameters,

parametry.m – function containing the vector of entry parameters and the necessary entry data.

identyfikacja.m

```
global T1 T2 T3 T4 T5 T6 T7 T8 t1 t2 t3 t4 t5 t6 t7 t8 IND i yz K Kd
```

```
i=1;
parametry;
load predkosc t yz;
fmins('index',v)
```

index.m

```
function [ind]=index(v)
global T1 T2 T3 T4 T5 T6 T7 T8 t1 t2 t3 t4 t5 t6 t7 t8 IND i yz po
T1=v(1);
T2=v(2);
T3=v(3);
T4=v(4);
T5=v(5);
T6=v(6);
T7=v(7);
T8=v(8);
[t,x]=ode45('funkrokr',[0:0.002:1],[0;0;0.5*po*10^7;0.5*po*10^7]);
y=x(:,2);
error=(yz-y).^2;
ind=sum(error)
t1(i)=T1;
t2(i)=T2;
t3(i)=T3;
t4(i)=T4;
t5(i)=T5;
t6(i)=T6;
t7(i)=T7;
t8(i)=T8;
IND(i)=ind;
i=i+1;
save aktualne
```

parametry.m

```
K=1400;
Kd=0.1;
po=4.6;
v=[2.5 0.6 0.9 0.18 2.5 0.5 0.18 4.5];
```

The identification process is initiated with the >>identyfikacja command.

The process can take up to a dozen minutes. During the identification process, the identification indicator (ind) will appear on the screen, which will inform us about the error values. Correctly, the indicator should decrease. During the process, all the parameters are saved into the **aktualne.mat** file.

After the identification process is completed, we need to check and estimate the entire process. We will check the parameters obtained and the coincidence of the speed signals (for a system with real and identified parameters).

The plots obtained can be saved in the form of the following file.

sprawdzenie.m

```
clear all;
global K Kd po
parametry;
load predkosc t yz;
load aktualne.mat
T1=t1(i-1);
```

Projekt współfinansowany ze środków Unii Europejskiej w ramach Europejskiego Funduszu Społecznego

```
T2=t2(i-1);
T3=t3(i-1);
T4=t4(i-1);
T5=t5(i-1);
T6=t6(i-1);
T7=t7(i-1);
T8=t8(i-1);
[t,x]=ode45('funkrokr',[0:0.002:1],[0;0;0.5*po*10^7;0.5*po*10^7]);
y=x(:,2);
plot(t,yz,t,y);
xlabel('time [s]')
ylabel('speed [m/s]')
print -dmeta yz_y.wmf
plot(t,yz-y);
xlabel(' time [s]')
ylabel('difference in speed [m/s]')
print -dmeta roz_yz_y.wmf
plot(t1);
print -dmeta t1.wmf
plot(t2);
print -dmeta t2.wmf
plot(t3);
print -dmeta t3.wmf
plot(t4);
print -dmeta t4.wmf
plot(t5);
print -dmeta t5.wmf
plot(t6);
print -dmeta t6.wmf
plot(t7);
print -dmeta t7.wmf
plot(t8);
print -dmeta t8.wmf
plot(IND);
print -dmeta blad.wmf
```

The start is initiated with the
>>sprawdzenie command.

The result of the operation of the sprawdzenie.m file will be plots comparing adequate signals and representing the process of parameter estimation.

REFERENCES

1. Bubnicki Z., Identyfikacja obiektów sterowania, PWN, Warszawa, 1974.
2. Maniczak K., Naharski Z., Komputerowa identyfikacja obiektów dynamicznych, PWN, Warszawa, 1983.
3. Soderstrom T., Stoica P., Identyfikacja systemów, PWN, 1997
4. Eykhoff. P.; System Identification – Parameter and State Estimation, John Wiley & Sons, 1974.
5. Ljung L.; System Identification. Theory of the User, Prentice Hall, 1987.
6. Nelles O.; Nonlinear System Identification, Springer Verlag, 2001.
7. Krolikowski A., Rorla D., Identyfikacja obiektów sterowania. Metody dyskretne, Poznań , Wyd. PP 2005
8. Kasprzyk J., Identyfikacja procesów, WPŚ, 2002, ISBN: 83-7335-077-2.
9. Zimmer A., Englot A., Identyfikacja obiektów i sygnałów, PK, Kraków 2005.
10. Janiszowski K., Identyfikacja modeli parametrycznych w przykładach, EXIT, 2002, ISBN: 83-87674-32-X.
11. Cedro L., Identyfikacja układów z napędem hydraulicznym z wykorzystaniem filtrów różniczkujących, Praca doktorska, PŚk 2006.