BRNO UNIVERSITY OF TECHNOLOGY

FAKULTY OF MECHANICAL ENGINEERING

INSTITUTE OF AUTOMATION AND COMPUTER SCIENCE

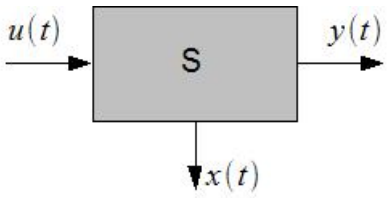
State feedback control design in Matlab through pole placement method

Assoc. Prof. Tomáš Březina, PhD.

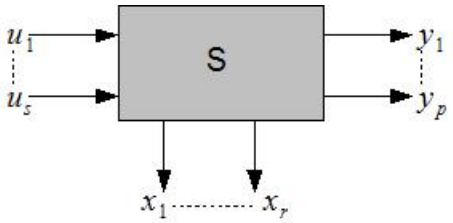
Contents

1. Dynamic system properties description

Dynamic system in general contains r inputs $\mathbf{u}$, m outputs $\mathbf{y}$ and n (inner) states $\mathbf{x}$.



State vector $\mathbf{x}$ is considered as abstract variable which can not be always measured. Input and output vectors $\mathbf{u}$ and $\mathbf{y}$ are variables of certain physical meaning and can be generally measured.



1.1. Basic types of dynamic systems I

Continuous

Continuous linear system of n-th order with r inputs and m outputs works in continuous time t and considers vectors

$$\mathbf{x}(t)=\left[x_1(t),\ldots,x_n(t)\right]^T,\ \mathbf{u}(k)=\left[u_1(t),\ldots,u_r(t)\right]^T,\ \mathbf{y}(t)=\left[y_1(t),\ldots,y_m(t)\right]^T$$

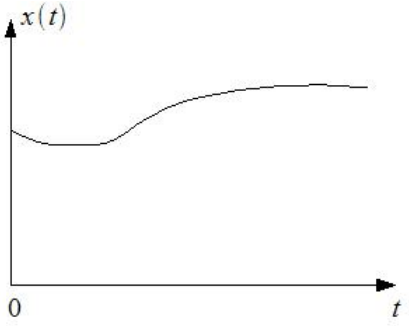
Discrete

Discrete linear system of n-th order with r inputs and m outputs in discrete time $t=kT$, $k=0,1,\ldots$ and considers vectors

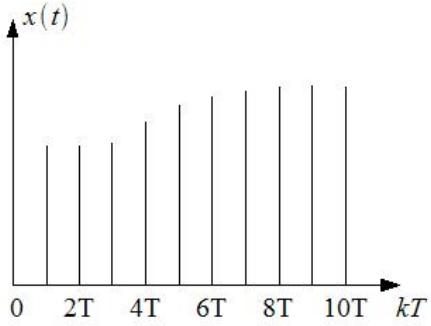
$$\mathbf{x}(k)=\left[x_1(k),\ldots,x_n(k)\right]^T,\ \mathbf{u}(k)=\left[u_1(k),\ldots,u_r(k)\right]^T,\ \mathbf{y}(k)=\left[y_1(k),\ldots,y_m(k)\right]^T$$

in those discrete times only.

Continuous signal



Discrete signal



1.2. Basic types of dynamic systems II

linear

it's mathematical model is represented by linear differential or difference equations

non-linear

it's mathematical model is represented by non-linear differential or difference equations

1.3. Basic types of dynamic systems III

SISO

Systems with single input and single output ($r = m = 1$).

MIMO

Systems with multiple inputs and multiple outputs ($r > 1, m > 1$).

1.4. Basic types of dynamic systems description

Outer system description

- Expresses dynamic properties of relations between input and output of the system
- System is considered as black box with inputs and outputs
- Only the system reaction to input signals is analyzed

Inner system description

- Apart from dynamic properties of relations between input and output of the system also expresses dynamic properties of relations of (inner) states of the system.
- System is considered as white box with inputs and outputs.
- The reaction of the system into the input signal and development of (inner) states is analyzed.

2. Outer system description

2.1. Continuous SISO system

$$a_n y^{(n)}(t) + a_{n-1} y^{(n-1)}(t) + \ldots + a_1 y'(t) + a_0 y(t) = b_m u^{(m)}(t) + b_{m-1} u^{(m-1)}(t) + \ldots + b_1 u'(t) + b_0 u(t)$$

Symbols description

n	Order of the system
a_i , b_j	Constant coefficients (real numbers)
$u(t)$	Input variable
$y(t)$	Output variable

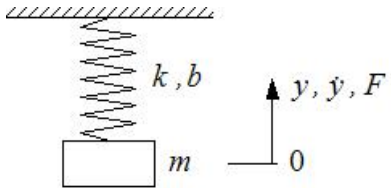
Note.

Initial conditions of the system must be known: $y(0), y'(0), \ldots, y^{(n-1)}(0)$, course of input variable $u(t)$ including its initial conditions $u(0), u'(0), \ldots, u^{(m-1)}(0)$.

Note.

Physical feasibility condition is sufficient condition of cause not preceding the consequence, $m \leq n$

Example



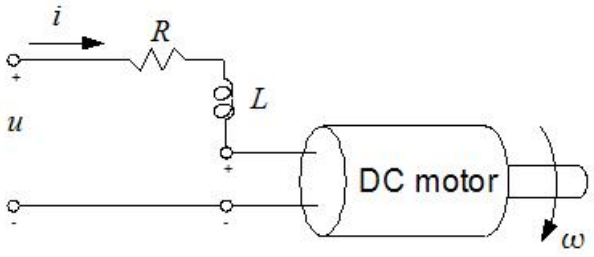
Outer description of deflection and load force in mechanical system made of mass, spring and damper is

$$m\ddot{y} + b\dot{y} + ky = F$$

Symbols description

$y\,[\mathrm{m}]$	deflection
$m\,[\mathrm{kg}]$	mass
$b\left[\mathrm{kg.s^{-1}}\right]$	damping
$k\left[\mathrm{kg.s^{-2}}\right]$	stiffness of spring
$F\,[\mathrm{N}]$	load force

Example



Outer description of relation between armature voltage and angular velocity of the rotor of unloaded electric drive with constant magnetic flux is

$$L\frac{J}{C_e}\omega^{(2)} + R\frac{J}{C_e}\omega' + C_e\omega = u$$

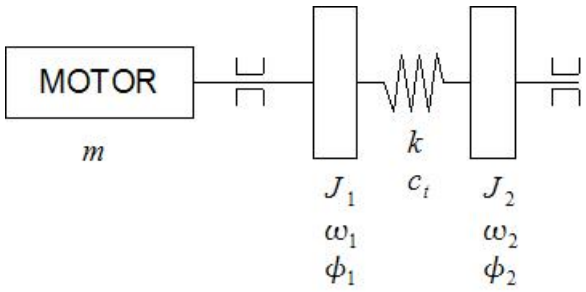
Outer description of relation between armature voltage and angular position of unloaded electric drive with constant magnetic flux is

$$L\frac{J}{C_e}\varphi^{(3)} + R\frac{J}{C_e}\varphi^{(2)} + C_e\varphi' = u$$

Symbols description

$C_e\,[\mathrm{V.s}]$	momentum constant of drive
$L\,[\mathrm{H}]$	armature winding inductance
$R\,[\Omega]$	armature winding resistance
$J\,[\mathrm{kg.m^2}]$	rotor moment of inertia
$\omega\,[\mathrm{s^{-1}}]$	rotor angular velocity
$\varphi\,[-]$	rotor angular position
$u\,[\mathrm{V}]$	armature voltage

Example



Outer description of relation between angular position of shaft on loaded side and torque of motor of electric drive made of two mass system connected with flexible rotor is

$$J_2\varphi_2^{(4)} + c_t\left(\frac{J_2}{J_1}+1\right)\varphi_2^{(3)} + k\left(\frac{J_2}{J_1}+1\right)\varphi_2^{(2)} = \frac{c_t}{J_1}m' + \frac{k}{J_1}m$$

Symbols description

$J_1 \left[\text{kg.m}^2\right]$	rotor moment of inertia on drive side
$J_2 \left[\text{kg.m}^2\right]$	rotor moment of inertia on load side
$\varphi_2 \left[\text{-}\right]$	angular position of rotor on load side
$k \left[\text{kg.s}^{-2}\right]$	stiffness of rotor
$c_t \left[\text{kg.s}^{-1}\right]$	damping factor of rotor
$m \left[\text{N.m}\right]$	Torque of motor

2.2. Discrete SISO system

$$a_n y(k-n) + a_{n-1} y(k-n+1) + \ldots + a_1 y(k-1) + a_0 y(k) =$$
$$b_m u(k-m) + b_{m-1} u(k-m+1) + \ldots + b_1 u(k-1) + b_0 u(k)$$

Symbols description

n	Order of the system
a_i , b_j	Constant coefficients (real numbers)
$u(k-i)$	Input variable in time $t = (k-i)T$
$y(k-i)$	Output variable in time $t = (k-i)T$
T	Sampling period

Example

Outer description of relation between input and output of numerical PI controller is

$$y(k) - y(k-1) = \left[K_p + K_i T\right] u(k) - K_p u(k-1)$$

Symbols description

K_p	controller proportional gain
K_i	controller integral gain
T	sampling period

3. Inner system description

3.1. State model of continuous MIMO system

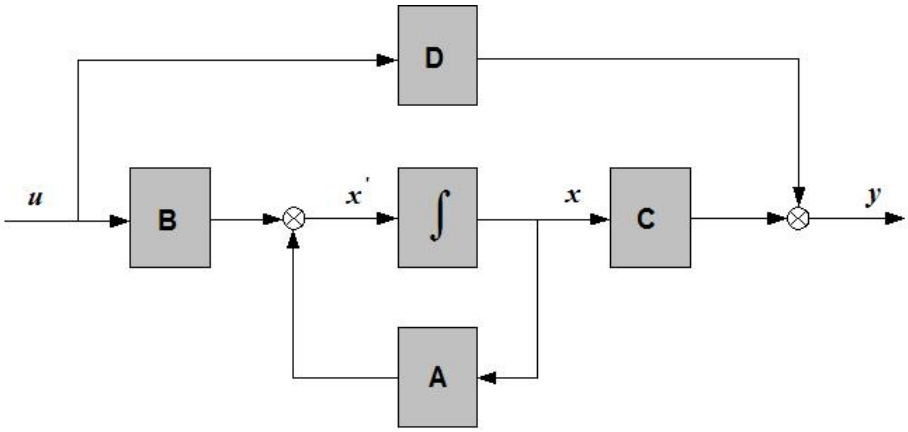
$\mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$ state equation

$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$ output equation

Symbol description

$\mathbf{u}(t)$	Input vector
$\mathbf{y}(t)$	Output vector
$\mathbf{x}(t)$	Vector of system states
A	State matrix
B	Excitation (input) matrix
C	Output matrix
D	Transfer matrix (of direct transfer)

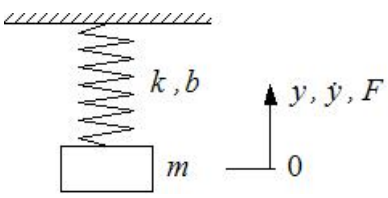
Graphical representation of state model equations



Matlab

```
% A, B, C, D must have concrete values
sys = ss(A, B, C, D)
```

Example



Inner description of relation between deflection and load force in mechanical system made of mass, spring and damper is

$$\dot{y} = v_y$$
$$\dot{v}_y = -\frac{k}{m} y - \frac{b}{m} v_y + \frac{1}{m} F$$

Symbols description

$y\,[\mathrm{m}]$	Deflection
$v_y\,[\mathrm{m.s^{-1}}]$	deflection velocity
$m\,[\mathrm{kg}]$	Mass
$b\,[\mathrm{kg.s^{-1}}]$	Damping
$k\,[\mathrm{kg.s^{-2}}]$	spring stiffness
$F\,[\mathrm{N}]$	load force

State model

$$\begin{bmatrix} y' \\ v_y' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix} \begin{bmatrix} y \\ v_y \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} [F]$$

$$[y] = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} y \\ v_y \end{bmatrix} + [0][F]$$

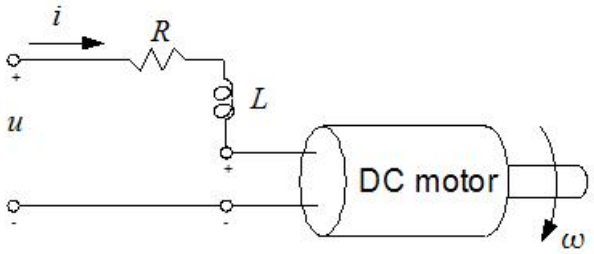
State model matrices are

$$\mathbf{A} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix},\ \mathbf{B} = \begin{bmatrix} 0 \\ 1/m \end{bmatrix},\ \mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix},\ \mathbf{D} = \begin{bmatrix} 0 \end{bmatrix}$$

In Matlab

```
% concrete values
m = 100;  % [kg]
b = 100;  % [kg.s^-1]
k = 1000; % [kg.s^-2]

A=[0 1; -k/m -b/m];
B=[0 1/m]';
C=[1 0];
D=[0];

ss_Mech = ss(A, B, C, D)
```

Example



Inner description of relation between armature voltage and angular velocity of the rotor of unloaded electric drive with constant magnetic flux is

$$\omega' = \frac{C_e}{J} i$$

$$i' = \frac{1}{L}\left(-C_e \omega - Ri + u\right)$$

Inner description of relation between armature voltage and angular position of the same drive

$$\varphi' = \omega$$
$$\omega' = \frac{C_e}{J} i$$
$$i' = \frac{1}{L}\left(-C_e\omega - Ri + u\right)$$

Symbols description

$C_e\,[\mathrm{V.s}]$	momentum constant of drive
$L\,[\mathrm{H}]$	armature winding inductance
$R\,[\Omega]$	armature winding resistance
$J\,[\mathrm{kg.m^2}]$	rotor moment of inertia
$\omega\,[\mathrm{s^{-1}}]$	rotor angular velocity
$\varphi\,[-]$	rotor angular position
$u\,[\mathrm{V}]$	armature voltage
$i\,[\mathrm{A}]$	armature winding current

State model of relation between armature voltage and rotor angular velocity

$$\begin{bmatrix} \omega' \\ i' \end{bmatrix} = \begin{bmatrix} 0 & C_e/J \\ -C_e/L & -R/L \end{bmatrix} \begin{bmatrix} \omega \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 1/L \end{bmatrix} [u] \qquad \text{state equation}$$

$$[\omega] = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} \omega \\ i \end{bmatrix} + [0][u] \qquad \text{output equation}$$

State model matrices are

$$\mathbf{A} = \begin{bmatrix} 0 & C_e/J \\ -C_e/L & -R/L \end{bmatrix},\ \mathbf{B} = \begin{bmatrix} 0 \\ 1/L \end{bmatrix},\ \mathbf{C} = \begin{bmatrix} 1 & 0 \end{bmatrix},\ \mathbf{D} = [0]$$

In Matlab

```
% concrete values

Ce = 2.8;  % [V.s]
J  = 0.1;  % [kg.m^2]
R  = 0.5;  % [Ohm]
L =  5e-3; % [mH]

A = [0 Ce/J; -Ce/L -R/L];
B = [0 1/L]';
C = [1 0];
D = [0];

ss_Mot_2 = ss(A, B, C, D)
```

State model of relation between voltage and angular position

$$\begin{bmatrix} \varphi' \\ \omega' \\ i' \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & C_e/J \\ 0 & -C_e/L & -R/L \end{bmatrix} \begin{bmatrix} \varphi \\ \omega \\ i \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 1/L \end{bmatrix} [u] \quad \text{state equation}$$

$$[\varphi] = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \varphi \\ \omega \\ i \end{bmatrix} + [0][u] \quad \text{output equation}$$

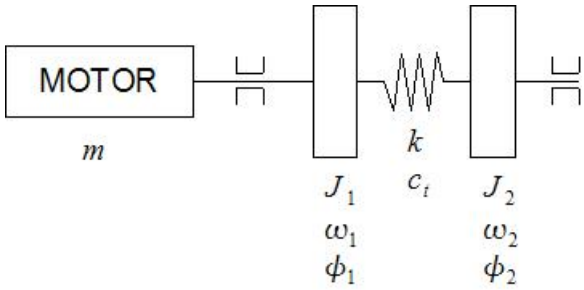
State model matrices are

$$\mathbf{A} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & C_e/J \\ 0 & -C_e/L & -R/L \end{bmatrix}, \ \mathbf{B} = \begin{bmatrix} 0 \\ 0 \\ 1/L \end{bmatrix}, \ \mathbf{C} = \begin{bmatrix} 1 & 0 & 0 \end{bmatrix}, \ \mathbf{D} = [0]$$

In Matlab

```
% concrete values

Ce = 2.8;  % [V.s]
J  = 0.1;  % [kg.m^2]
R  = 0.5;  % [Ohm]
L =  5e-3; % [mH]

A = [0 1 0; 0 0 Ce/J; 0 -Ce/L -R/L];
B = [0 0 1/L]';
C = [1 0 0];
D = [0];

ss_Mot_3 = ss(A, B, C, D)
```

Example



Inner description of relation between angular position of shaft on loaded side and torque of motor of electric drive made of two mass system connected with flexible rotor is

$$\omega_1' = \frac{1}{J_1}\left(-\omega_1 c_t + \omega_2 c_t - \varphi_1 k + \varphi_2 k + m\right)$$
$$\omega_2' = \frac{1}{J_2}\left(\omega_1 c_t - \omega_2 c_t + \varphi_1 k - \varphi_2 k\right)$$
$$\varphi_1' = \omega_1$$
$$\varphi_2' = \omega_2$$

Symbols description

$J_1\left[\text{kg.m}^2\right]$	rotor moment of inertia on motor side
$J_2\left[\text{kg.m}^2\right]$	rotor moment of inertia on load side
$\omega_1\left[\text{s}^{-1}\right]$	rotor angular velocity of motor side
$\omega_2\left[\text{s}^{-1}\right]$	rotor angular velocity of load side
$\varphi_1\left[\text{-}\right]$	rotor angular position on motor side
$\varphi_2\left[\text{-}\right]$	rotor angular position on load side
$k\left[\text{kg.s}^{-2}\right]$	rotor stiffness
$c_t\left[\text{kg.s}^{-1}\right]$	rotor damping factor
$m\left[\text{N.m}\right]$	torque of motor

State model of relation between motor torque and load rotor angular position

$$\begin{bmatrix}\omega_1'\\ \omega_2'\\ \varphi_1'\\ \varphi_2'\end{bmatrix} = \begin{bmatrix}-\frac{c_t}{J_1} & \frac{c_t}{J_1} & -\frac{k}{J_1} & \frac{k}{J_1}\\ \frac{c_t}{J_2} & -\frac{c_t}{J_2} & \frac{k}{J_2} & -\frac{k}{J_2}\\ 1 & 0 & 0 & 0\\ 0 & 1 & 0 & 0\end{bmatrix}\begin{bmatrix}\omega_1\\ \omega_2\\ \varphi_1\\ \varphi_2\end{bmatrix} + \begin{bmatrix}\frac{1}{J_1}\\ 0\\ 0\\ 0\end{bmatrix}[m]$$

$$[\varphi_2] = \begin{bmatrix}0 & 0 & 0 & 1\end{bmatrix}\begin{bmatrix}\omega_1\\ \omega_2\\ \varphi_1\\ \varphi_2\end{bmatrix} + [0][m]$$

State model matrices are

$$\mathbf{A}=\begin{bmatrix} -\frac{c_t}{J_1} & \frac{c_t}{J_1} & -\frac{k}{J_1} & \frac{k}{J_1} \\ \frac{c_t}{J_2} & -\frac{c_t}{J_2} & \frac{k}{J_2} & -\frac{k}{J_2} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix},\ \mathbf{B}=\begin{bmatrix} \frac{1}{J_1} \\ 0 \\ 0 \\ 0 \end{bmatrix},\ \mathbf{C}=\begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix},\ \mathbf{D}=\begin{bmatrix} 0 \end{bmatrix}$$

In Matlab

```
% concrete values

J1 = 1;    % [kg.m^2]
J2 = 1;    % [kg.m^2]
ct = 0.01; % [kg.s^-1]
k  = 1000; % [kg.s^-2]

A = [-ct/J1 ct/J1 -k/J1 k/J1; ct/J2 -ct/J2 k/J2 -k/J2;...
 1 0 0 0; 0 1 0 0];
B = [1/J1 0 0 0]';
C = [0 0 0 1];
D = [0];
ss_Drv = ss(A, B, C, D)
```

Note.

- System dynamics is caused by dependency of the current state of the system on its previous states.
- State mode of continuous MIMO system is often used with $\mathbf{D}=0$ (so called Frobenius canonical form).

Note.

- Several state models exist describing the same relation between input and output (identical behavior of different systems).
- State equation represents the set of linear differential equations with constant coefficients.
- State vector $\mathbf{x}(t)$ is considered as abstract variable which can not always be measured.

  Input and output vectors $\mathbf{u}(t)$ and $\mathbf{y}(t)$ are variable of concrete physical meaning and are generally measurable.
- Outer description can be reached from the inner description by eliminating the system state $\mathbf{x}(t)$,
- Therefore the inner description is mathematically equivalent to outer description.

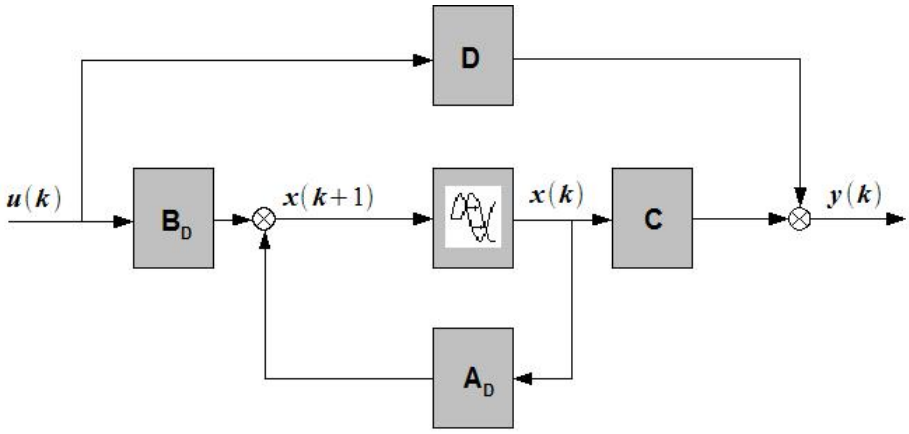
3.2. State model of discrete MIMO system

$$\mathbf{x}(k+1) = \mathbf{A}_D\mathbf{x}(k) + \mathbf{B}_D\mathbf{u}(k) \quad \text{state equation}$$
$$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k) \quad \text{output equation}$$

Symbol description

$\mathbf{u}(k)$	Input vector in time $t = kT$
$\mathbf{y}(k)$	Output vector in time $t = kT$
$\mathbf{x}(k)$	State vector in time $t = kT$
$\mathbf{x}(k+1)$	State vector in time $t = (k+1)T$
T	Sampling period
$\mathbf{A}_D$	State matrix
$\mathbf{B}_D$	Excitation (input) matrix
$\mathbf{C}$	Output matrix
$\mathbf{D}$	Transfer matrix (of direct transfer)

Graphical representation of state model equations



Matlab

```
% Ad, Bd, C, D must have concrete values
sys = ss(Ad, Bd, C, D, T)
```

Example

Inner description of relation between input and output of numerical PI controller is

$$y(k+1) = y(k) + u(k)$$
$$y(k) = K_i T y(k) + \left[K_p + K_i T\right] u(k)$$

Symbols description

K_p	controllers proportional gain
K_i	controllers integral gain
T	sampling period

State model

$$\left[x(k+1)\right] = [1]\left[x(k)\right] + [1][u(k)]$$
$$[y(k)] = \left[K_i T\right]\left[x(k)\right] + [K_p + K_i T][u(k)]$$

State model matrices are

$\mathbf{A}_D = [1]$, $\mathbf{B}_D = [1]$, $\mathbf{C} = \left[K_i T\right]$, $\mathbf{D} = \left[K_p + K_i T\right]$

In Matlab

```
% concrete values
Kp = 6.4;
Ki = 200;
T = 1/300;

Ad = [1];
Bd = [1]';
C = [Ki*T];
D = [Kp+Ki*T];

ss_z_PI = ss(Ad, Bd, C, D, T)
```

4. Favourite tools of linear dynamic system analysis

For simplification of both inner and outer description of linear systems

- Laplace transform – for continuous systems
- Z transform – for discrete systems

4.1. Laplace transform

For simplification of both inner and outer description of linear continuous systems.

Direct (forward)

$$F(s) = L\{f(t)\} = \int_0^\infty f(t)e^{-st}dt\,,$$

Inverse (back)

$$f(t) = L^{-1}\{F(s)\} = \frac{1}{2\pi j}\oint F(s)e^{st}ds$$

Symbols description

$f(t)$	original (subject) – real function in real domain for $t \in \langle 0, \infty)$
$F(s)$	image – complex function in complex variable domain
$s = \alpha + j\omega$	Complex variable ($\alpha = \mathrm{Re}\, s$, $\omega = \mathrm{Im}\, s$)
t	Real variable (time)
L	Operator of direct Laplace transform
$j = \sqrt{-1}$	Imaginary unit

Note.

It is line integral around a closed curve, direct calculation can be difficult

Sufficient conditions of $F(s)$ existence

1. $f(t) = 0$ for $t < 0$
2. $f(t)$ is piecewise continuous
3. $f(t)$ is function of exponential order, that means that exists $M > 0$ a $\alpha_0 \in (-\infty, \infty)$ such as $|f(t)| \le Me^{\alpha_0 t}$, for $t \in \langle 0, \infty)$

Correspondence

$$f(t) \cong F(s)$$

Represents unique assigning of image $F(s)$ to original $f(t)$

Note.

Particular values of (complex) variable s can be shown in complex plane, so called s–plane.

The most important properties of L-transform

- Differentiation of original according to t corresponds to multiplying of image s,
  $$L\left\{\frac{df(t)}{dt}\right\} = sF(s) - f(0)$$
- Integration of original according to t corresponds to division of image s,
  $$L\left\{\int_0^t f(\tau)d\tau\right\} = \frac{1}{s}F(s)$$
- Transform is linear $L\{a_1 f_1(t) + a_2 f_2(t)\} = a_1 L\{f_1(t)\} + a_2 L\{f_2(t)\}$
- …

4.2. Z transform

For simplification of both inner and outer description of linear dicrete systems.

Direct

$$F(z) = Z\{f(kT)\} = \sum_{k=0}^{\infty} f(kT)\, z^{-k}$$

Inverse

$$f(kT) = Z^{-1}\{F(z)\} = \frac{1}{2\pi j}\oint_C F(z)\, z^{k-1} dz$$

Symbols description

kT	Discrete real variable (discrete time)
$f(kT)$	Discrete original (Z original) – real function in discrete real domain for $k = 0, 1, 2, \ldots$
$F(z)$	Discrete image – complex function in complex variable domain
$s = \alpha + j\omega$	Complex variable ($\alpha = \mathrm{Re}\, s$, $\omega = \mathrm{Im}\, s$)

t	Real variable (time)
Z	Operator of direct Z transform
Z^{-1}	Operator of inverse Z transform
T	Sampling period
C	Circle with r diameter, in which all singular points of $F(z)$ function are located

Correspondence

$f(k) \cong F(z)$

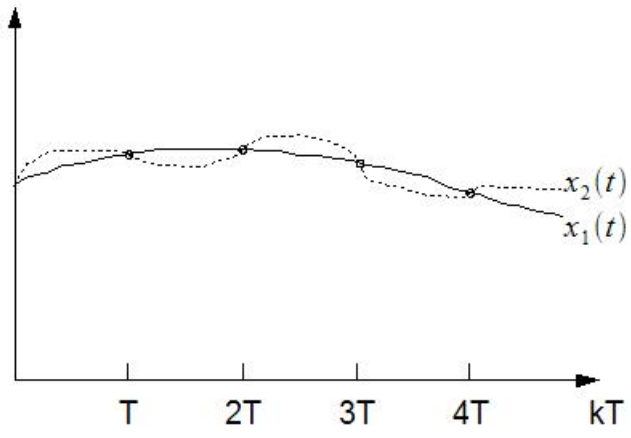
Represents unique assigning of image $F(z)$ to original $f(k)$

Note.

Particular values of (complex) variable z can be shown in complex plane, so called z–plane.

The most important properties of Z transform

- Delay in original for m samples corresponds to multiplication of image z^{-m}
  $Z\{f[(k-m)T]\} = z^{-m}F(z), m \geq 0$
- Transform is linear $Z\{a_1 f_1(t) + a_2 f_2(t)\} = a_1 Z\{f_1(t)\} + a_2 Z\{f_2(t)\}$
- …



4.3. Relation between L transform and Z transform

The relation between particular values of variable s of L–image and variable of Z–image is

$z = e^{sT}$ in other words $s = \frac{1}{T}\ln z$

Note.

Left half plane of s-plane is transformed inside unit circle in z-plane.

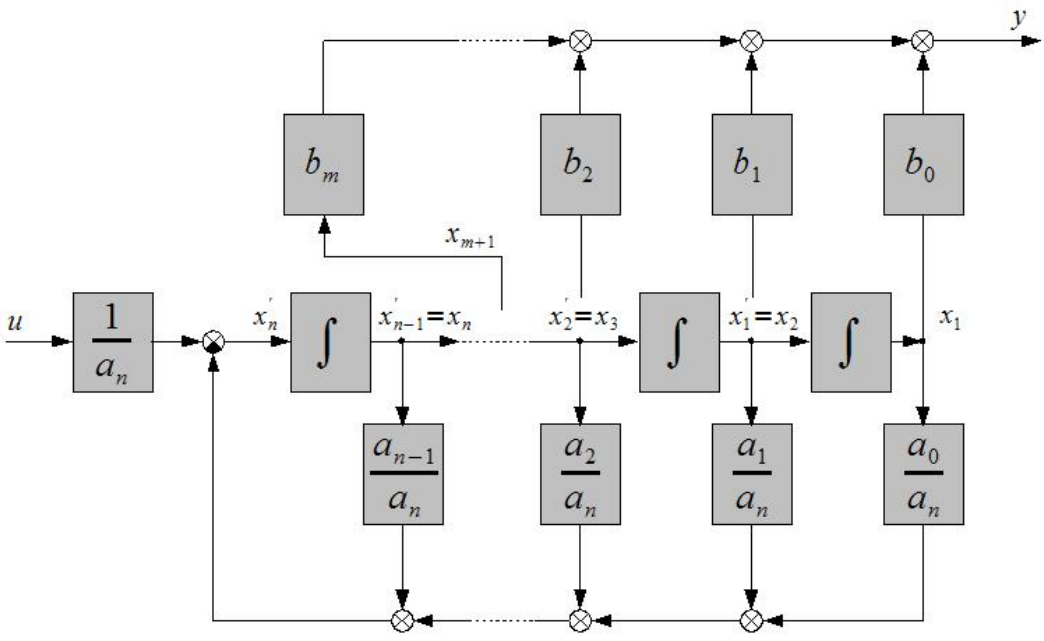
4.4. Continuous system transfer

Characterizes the response of continuous SISO system. It is defined as ration of L-image of output variable towards L-image of input variable (from outer description) with zero initial conditions of the system $y(0) = y'(0) = \ldots = y^{(n-1)}(0) = 0$ and input signal $u(0) = u'(0) = \ldots = u^{(m-1)}(0) = 0$.

$$\left[a_n s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_0\right] Y(s) = \left[b_m s^m + b_{m-1} s^{m-1} + \ldots + b_1 s + b_0\right] U(s)$$

$$F(s) = \frac{Y(s)}{U(s)} = \frac{b_m s^m + b_{m-1} s^{m-1} + \ldots + b_1 s + b_0}{a_n s^n + a_{n-1} s^{n-1} + \ldots + a_1 s + a_0}$$

Graphical representation of system transfer



Matlab

```
% bm, … b1, b0, an, …, a1, a0 must have concrete values

s = tf('s');
sys = (bm * s^m + … + b1 * s + b0) / ...
      (an * s^n + … + a1 *s + a0);
```

Example

Transfer of load force into deflection in mechanical system made of mass, spring and damper

$$F(s) = \frac{Y(s)}{F(s)} = \frac{1}{ms^2 + bs + k}$$

as $m\ddot{y} + b\dot{y} + ky = F$.

Symbols description

$y\,[\mathrm{m}]$	Deflection
$m\,[\mathrm{kg}]$	Mass
$b\left[\mathrm{kg.s^{-1}}\right]$	damping
$k\left[\mathrm{kg.s^{-2}}\right]$	stiffness of spring
$F\,[\mathrm{N}]$	load force

In Matlab

```
% concrete values
m = 100;  % [kg]
b = 100;  % [kg.s^-1]
k = 1000; % [kg.s^-2]

s = tf('s');
tf_Mech = 1 / (m * s^2 + b * s + k)
```

Example

Transfer of armature voltage to angular velocity of the rotor of unloaded electric drive with constant magnetic flux is

$$F(s) = \frac{\Omega(s)}{U(s)} = \frac{1}{L\frac{J}{C_e}s^2 + R\frac{J}{C_e}s + C_e}$$

as $L\frac{J}{C_e}\omega^{(2)} + R\frac{J}{C_e}\omega' + C_e\omega = u$

Transfer of armature voltage to angular position of unloaded electric drive with constant magnetic flux is

$$F(s) = \frac{\Phi(s)}{U(s)} = \frac{1}{L\frac{J}{C_e}s^3 + R\frac{J}{C_e}s^2 + C_e s}$$

as $L\frac{J}{C_e}\varphi^{(3)} + R\frac{J}{C_e}\varphi^{(2)} + C_e\varphi' = u$

Symbols description

$C_e\,[\mathrm{V.s}]$	momentum constant of drive
$L\,[\mathrm{H}]$	armature winding inductance
$R\,[\Omega]$	armature winding resistance
$J\,[\mathrm{kg.m^2}]$	rotor moment of inertia
$\omega\,[\mathrm{s^{-1}}]$	rotor angular velocity
$\varphi\,[-]$	rotor angular position
$u\,[\mathrm{V}]$	armature voltage

In Matlab

```
% concrete values

Ce = 2.8;  % [V.s]
J  = 0.1;  % [kg.m^2]
R  = 0.5;  % [Ohm]
L =  5e-3; % [mH]

s = tf('s');
tf_Mot_2  = 1/(L*J/Ce * s^2 + R*J/Ce * s + Ce)
tf_Mot_3 = 1/(L*J/Ce * s^3 + R*J/Ce * s^2 + Ce * s)
```

Example

Transfer of torque of motor of electric drive made of two mass system connected with flexible rotor to angular position of shaft on loaded side is

$$F(s)=\frac{\Phi_2(s)}{M(s)}=\frac{\frac{c_t}{J_1}s+\frac{k}{J_1}}{J_2 s^4+c_t\left(\frac{J_2}{J_1}+1\right)s^3+k\left(\frac{J_2}{J_1}+1\right)s^2}$$

as $J_2\varphi_2^{(4)}+c_t\left(\frac{J_2}{J_1}+1\right)\varphi_2^{(3)}+k\left(\frac{J_2}{J_1}+1\right)\varphi_2^{(2)}=\frac{c_t}{J_1}m'+\frac{k}{J_1}m$

Symbols description

$J_1\left[\text{kg.m}^2\right]$	rotor moment of inertia on drive side
$J_2\left[\text{kg.m}^2\right]$	rotor moment of inertia on load side
$\varphi_2\left[\text{-}\right]$	angular position of rotor on load side
$k\left[\text{kg.s}^{-2}\right]$	stiffness of rotor
$c_t\left[\text{kg.s}^{-1}\right]$	damping factor of rotor
$m\left[\text{N.m}\right]$	torque of motor

In Matlab

```
% concrete values

J1 = 1;    % [kg.m^2]
J2 = 1;    % [kg.m^2]
ct = 0.01; % [kg.s^-1]
k  = 1000; % [kg.s^-2]

s = tf('s');
tf_Drv = (ct/J1 * s + k/J1)/ ...
         (J2 * s^4 + ct*(J2/J1+1) * s^3 + k*(J2/J1+1) * s^2)
```

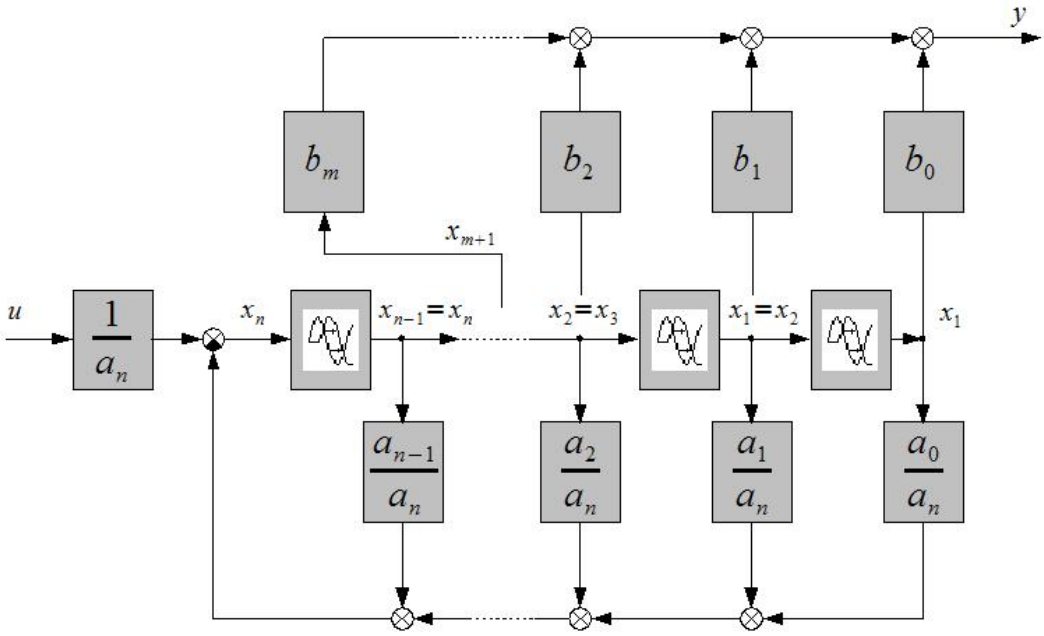
4.5. Discrete system transfer

Characterizes the response of discrete SISO system. It is defined as ration of Z-image of output variable towards Z-image of input variable (from outer description) with zero initial conditions of the system $y(0)=y(-1)=\ldots=y(1-n)=0$ and input signal $u(0)=u(-1)=\ldots=u(1-m)=0$.

$$\left[a_n z^{-n}+a_{n-1}z^{-n+1}+\ldots+a_1 z^{-1}+a_0\right]Y(z)=\left[b_m z^{-m}+b_{m-1}z^{-m+1}+\ldots+b_1 z^{-1}+b_0\right]U(z)$$

$$F(z)=\frac{Y(z)}{U(z)}=\frac{b_m z^{-m}+b_{m-1}z^{-m+1}+\ldots+b_1 z^{-1}+b_0}{a_n z^{-n}+a_{n-1}z^{-n+1}+\ldots+a_1 z^{-1}+a_0}$$

Graphical representation of system transfer



Matlab

```
% bm, … b1, b0, an, …, a1, a0 must have concrete values
%
z = tf('z', T);
sys = (bm * z^-m + … + b1 * z^-1 + b0) / ...
      (an * z^-n + … + a1 * z^-1 + a0);
```

Note.

Discrete transfer function can be expressed in form of fraction of two polynomials in z (more common form in modeling tasks) or polynomials in z^{-1} (more common form in control tasks).

Matlab

```
sys.Variable = 'z';     % transfer as polynomial in z
sys.Variable = 'z^-1';  % transfer as polynomial in z^-1
```

Example

Discrete transfer of numerical PI controller is

$$F_D(z) = \frac{Y(z)}{U(z)} = \frac{\left(K_p + K_i T\right) - K_p z^{-1}}{1 - z^{-1}}$$

as $y(k) - y(k-1) = \left[K_p + K_i T\right] u(k) - K_p u(k-1)$

Symbols description

K_p	controllers proportional gain
K_i	controllers integral gain
T	sampling period

In Matlab

```
% concrete values
Kp = 6.4;
Ki = 200;
T = 1/300;

z = tf('z', T);
Fz = ((Kp+Ki*T) - Kp*z^-1)/(1 - z^-1);
Fz.variable = 'z^-1'
```

4.6. Transfer matrix of system

Characterizes the response of continuous MIMO system with r inputs and m outputs on input signals.

Continuous system

It is matrix $\mathbf{F}(s)$ of type $m \times r$ with elements $f_{i,j}(s)$. Element $f_{i,j}(s)$ contains continuous transfer of j–th input into i-th output.

Discrete system

It is matrix $\mathbf{F}_D(z)$ of type $m \times r$ with elements $(f_D)_{i,j}(z)$. Element $(f_D)_{i,j}(z)$ contains discrete transfer of j–th input into i-th output.

Note.

- Several continuous/discrete transfers can exist describing the same relation between input and output (identical outer behaviour of system).
- Roots of transfer numerator polynomial are called zeros.
- Roots of transfer denominator polynomial are called poles.
- Transfer can be expressed by root elements from zeros and poles.
- Both zeros and poles are real or complex conjugate.

Matlab

```
sys_ = minreal(sys);  % transfers continuous or discrete
                      % transfer or state model into
                      % standard form
zpk(sys)                % Transfer from zeros and poles
```

Example

In Matlab

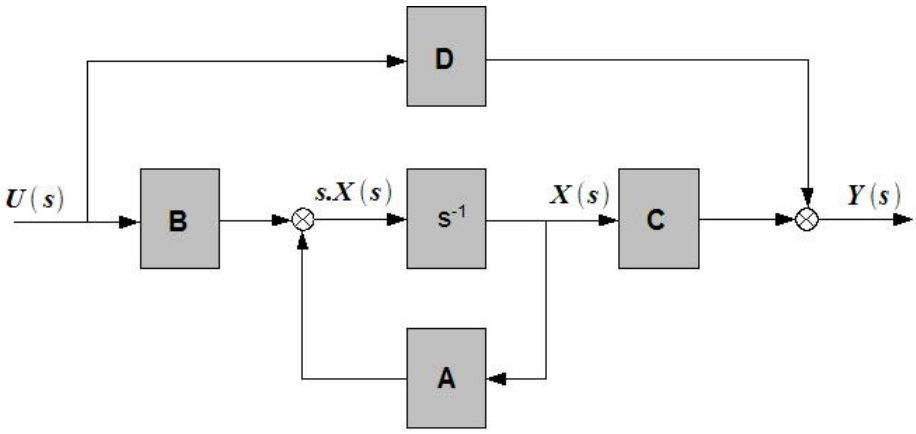
```
tf_Mech
minreal(tf_Mech)

tf_Mot_2
minreal(tf_Mot_2)

tf_Mot_3
minreal(tf_Mot_3)

ss_Mech
minreal(ss_Mech)

ss_Mot_2
minreal(ss_Mot_2)

ss_Mot_3
minreal(ss_Mot_3)

ss_z_PI
minreal(ss_z_PI)
```

4.7. L–image of continuous system inner description

For zero initial conditions of the system $y(0) = y'(0) = \ldots = y^{(n-1)}(0) = 0$ and input signal $u(0) = u'(0) = \ldots = u^{(m-1)}(0) = 0$.

Inner description	L–image of inner description
$\mathbf{x}'(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t)$	$s\mathbf{X} = \mathbf{A}\mathbf{X} + \mathbf{B}\mathbf{U}(s)$
$\mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t)$	$\mathbf{Y}(s) = \mathbf{C}\mathbf{X} + \mathbf{D}\mathbf{U}(s)$

Graphical representation of state and output equations

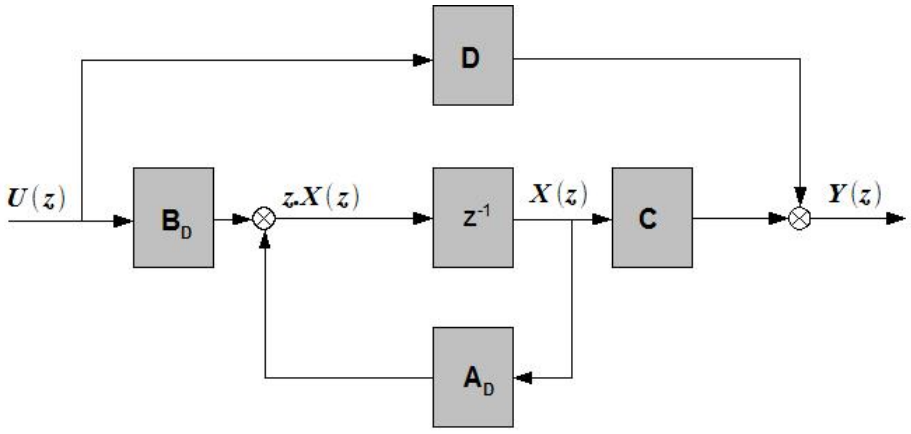


4.8. Z–image of continuous system inner description

For zero initial conditions of the system $y(0) = y'(0) = \ldots = y^{(n-1)}(0) = 0$ and input signal $u(0) = u'(0) = \ldots = u^{(m-1)}(0) = 0$.

Inner description	Z–image of inner description
$\mathbf{x}(k+1) = \mathbf{A}_D\mathbf{x}(k) + \mathbf{B}_D\mathbf{u}(k)$	$z\mathbf{X}(z) = \mathbf{A}_D\mathbf{X}(z) + \mathbf{B}_D\mathbf{U}(z)$
$\mathbf{y}(k) = \mathbf{C}\mathbf{x}(k) + \mathbf{D}\mathbf{u}(k)$	$\mathbf{Y}(z) = \mathbf{C}\mathbf{X}(z) + \mathbf{D}\mathbf{U}(z)$

Graphical representation of state and output equations



5. Relation between inner and outer description

Continuous system

$$\begin{aligned} s\mathbf{X} &= \mathbf{A}\mathbf{X} + \mathbf{B}\mathbf{U}(s) \\ \mathbf{Y}(s) &= \mathbf{C}\mathbf{X} + \mathbf{D}\mathbf{U}(s) \end{aligned} \qquad \mathbf{Y}(s) = \left[\mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}\right]\mathbf{U}(s) = \mathbf{F}(s)\mathbf{U}(s)$$

i.e. transfer matrix $\mathbf{F}(s) = \mathbf{C}(s\mathbf{I} - \mathbf{A})^{-1}\mathbf{B} + \mathbf{D}$

Discrete system

$$\begin{aligned} z\mathbf{X}(z) &= \mathbf{A}_\mathbf{D}\mathbf{X}(z) + \mathbf{B}_\mathbf{D}\mathbf{U}(z) \\ \mathbf{Y}(z) &= \mathbf{C}\mathbf{X}(z) + \mathbf{D}\mathbf{U}(z) \end{aligned} \qquad \mathbf{Y}(z) = \left[\mathbf{C}(z\mathbf{I} - \mathbf{A}_D)^{-1}\mathbf{B}_D + \mathbf{D}\right]\mathbf{U}(z) = \mathbf{F}(z)\mathbf{U}(z)$$

i.e. transfer matrix $\mathbf{F}_D(z) = \left[\mathbf{C}(z\mathbf{I} - \mathbf{A}_D)^{-1}\mathbf{B}_D + \mathbf{D}\right]\mathbf{U}(z)$

Matlab

```
sys_tf = tf(sys);  % transfers state model in sys into
                   % equivalent transfer (transfer matrix)
                   % in sys_tf
sys_ss = ss(sys);  % transfers transfer (transfer
                   % matrix)in sys into equivalent state model
                   % in sys_ss
```

Example

In Matlab

```
tf_Mech
tf(ss_Mech)
minreal(tf(ss_Mech))

tf_Mot_2
tf(ss_Mot_2)
minreal(tf(ss_Mot_2))

tf_Mot_3
tf(ss_Mot_3)
minreal(tf(ss_Mot_3))

ss_Mech
ss(tf_Mech)
minreal(ss(tf_Mech))
```

```
ss_Mot_2
ss(tf_Mot_2)
minreal(ss(tf_Mot_2))

ss_Mot_3
ss(tf_Mot_3)
minreal(ss(tf_Mot_3))

ss_z_PI
ss(tf_z_PI)
minreal(ss(tf_z_PI))
```

Note.

- Poles of continuous system are equal to eigenvalues of state matrix $\mathbf{A}$.
- Poles of discrete system are equal to eigenvalues of state matrix $\mathbf{A}_D$.
- Roots of characteristic equation of the system are equal to eigenvalues of state matrix $\mathbf{A}$, $\mathbf{A}_D$ respectively.
- State model is numerically more stable than transfer for the systems of higher orders.

6. Dynamic system behavior

Typically we study the response of the system for

- Unit step (in time domain)
- Unit impulse (in time domain)
- Harmonic signal (in frequency domain)
  Response is again harmonic signal with the same angular velocity, but with amplitude and phase shifts dependent on angular velocity

Matlab

```
% time domain
step(sys)      % response of continuous or discrete sys into
               % unit step
impulse(sys)   % response of continuous or discrete sys into
               % unit impulse

% frequency domain
bode(sys)      % amplitude and phase characteristics of sys
nyquist(F)     % Nyquist diagram of sys
```

Dynamic system behavior is determined mainly by roots of characteristic equation.

- Continuous system
- Real roots $s_i = \alpha_i$ of characteristic equation determine aperiodic components of the response.
- Complex conjugate roots $s_{i,i+1} = \alpha_i \pm j\omega_i$ of characteristic equation determine oscillative components of the response.
- Components determined by roots with $\alpha_i < 0$ are reduced with increasing time.
- If $\alpha_1 < \alpha_2 < 0$ of two different roots, then component corresponding to α_1 will be reduced faster.
- Discrete system
- Real roots z_i of characteristic equation determine aperiodic components of the response.
- Complex conjugate roots $z_{i,i+1}$ of characteristic equation determine oscillative components of the response.
- Components determined by roots with $|z_i| < 1$ are reduced with increasing time.
- If $|z_1| < |z_2| < 1$ of two different roots, then component corresponding to z_1 will be reduced faster.

Matlab

```
eig(sys)           % characteristic equation roots
                   % of continuous or discrete sys
```

Example

In Matlab

```
% evaluation of velocities of characteristic equation roots of
% systems

eig(ss_Mech)

eig(ss_Mot_2)

eig(ss_Mot_3)

eig(ss_z_PI)
```

6.1. Stability of continuous system

System is stable if its output variable value tends with increasing time towards finite value $|K| < \infty$ (stabilizes on finite value)

$$\lim_{t \to \infty} y(t) = K$$

Matlab

```
dcgain(sys)   % response of continuous and discrete sys into
              % unit step in steady state
```

Necessary and sufficient condition of stability

Continuous system is stable when for all its roots $s_i = \alpha_i + j\omega_i$ of its characteristic equation stands $\alpha_i < 0$ (it is in left half-plane of s-plane).

Discrete system is stable when for all its roots z_i of its characteristic equation stands $|z_i| < 1$ (it is inside the unit circle of z–plane).

Dominant roots of characteristic equation

- of continuous system are those roots of characteristic equation whose real components α_i are of highest values,
- of discrete systems are those roots of characteristic equation whose modules are of highest values.

Dominant roots have essential influence on transition course.
Real dominant root is at least one, complex conjugate dominant roots are at least two.

Correspondence between root of characteristic equation of continuous system and root of the characteristic equation of the same system discretized is

$z = e^{sT}$ $\left(s = \frac{1}{T}\ln z\right)$

Note.

Roots of characteristic equation are usually denoted using damping ratio coefficient d and angular frequency of undamped oscillations ω_0.

For non-oscillation components

$s_1 = -d$, with corresponding root factor $s + d$

For oscillation components

$s_{1,2} = -d\omega_0 \pm j\omega_0\sqrt{1-d^2}$, with corresponding root factor $s^2 + 2d\omega_0 s + \omega_0^2$

Or using time constants

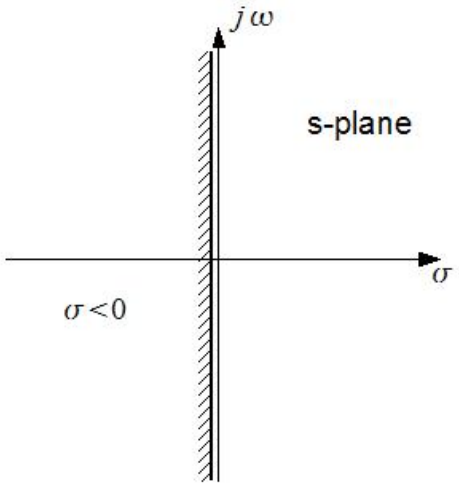
i.e. $T_1 = \frac{1}{\omega_0}$, $T_2 = \frac{2d}{\omega_0}$, with corresponding root factor $T_1^2 s^2 + T_2 s + 1$

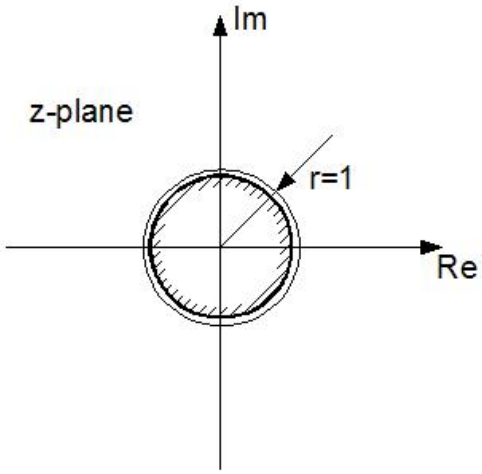
$\omega_0 = \frac{1}{T_1}$, $d = \frac{T_2}{2T_1}$

System speed

System is more stable for:

- (dominant) roots of characteristic equations of continuous system are more distant to the left from imaginary axis of s-plane,
- (dominant) roots of characteristic equations of discrete system inside unit circle are closer to the center of z-plane.

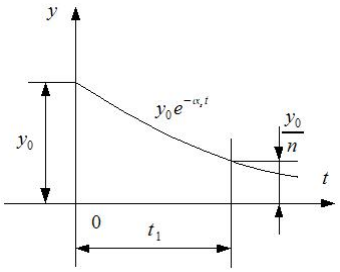


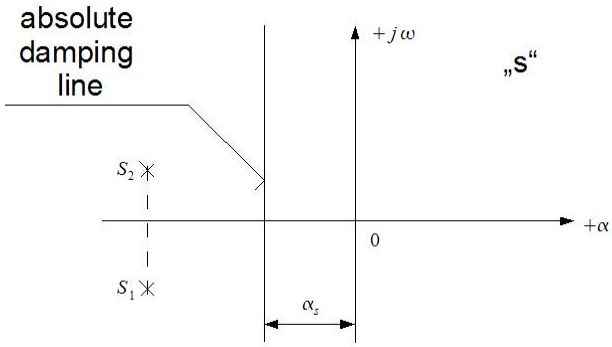


7. Stability degree

7.1. Absolute damping factor

For non-oscillation components the minimal value of damping $|\alpha_s|$ is determined from condition that after time t_1 the initial condition will be n–times smaller.





$$|\alpha_s| = \frac{1}{t_1} \ln n$$

Mathematical details

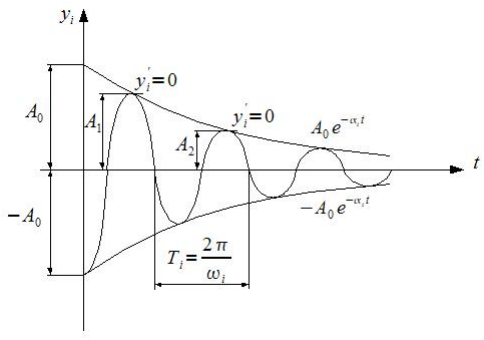
$$\frac{y_0}{n} = y_0 e^{-\alpha_s t_1},$$

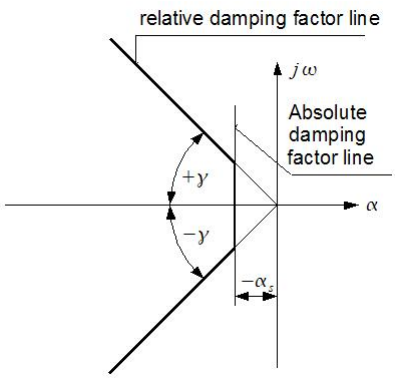
$$\ln n = |\alpha_s| t_1,$$

$$|\alpha_s| = \frac{1}{t_1} \ln n$$

7.2. Relative damping factor

For oscillation components we work with various versions of logarithmic damping decrement LD, which is determined from the condition that during the time of first cycle $T = \frac{2\pi}{\omega}$ the amplitude is reduced n–times.





$$LD \equiv \ln n = 2\pi \frac{\alpha}{\omega}$$

$$d = \frac{\ln n}{\sqrt{4\pi^2 + \ln^2 n}},\ \omega_0 = \frac{2\pi}{T\sqrt{1-d^2}}$$

Mathematical details

$$\gamma = \operatorname{arctg} \frac{\omega}{\alpha}$$

$$A_2 = A_1^{-\alpha_i T_i},\ \frac{A_1}{A_2} = e^{2\pi \frac{\alpha_i}{\omega_i}},\ LD \equiv \ln \frac{A_1}{A_2} = \ln n = 2\pi \frac{\alpha}{\omega} = \frac{2\pi}{\operatorname{tg} \gamma}$$

Line of relative damping factor
$\omega = \operatorname{tg} \gamma \cdot \alpha$

Logarithmic damping decrement
$$LD \equiv \ln \frac{A_1}{A_2} = \ln n = 2\pi \frac{\alpha}{\omega} = \frac{2\pi}{\operatorname{tg} \gamma}$$

$$\gamma = \operatorname{arctg} \frac{2\pi}{\ln \frac{A_1}{A_2}}$$

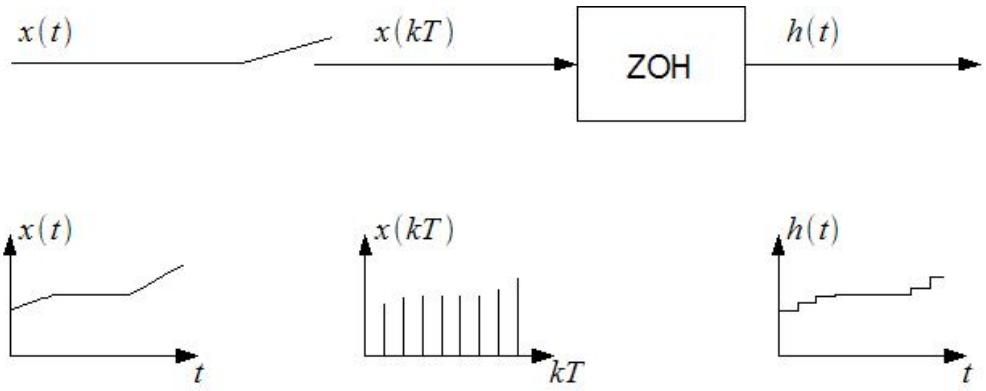
Dominant roots, whose values $|\alpha| \geq |\alpha_s|$ are closest to $|\alpha_s|$ value.

8. Discretization of continuous system

Represents creation of discrete description of continuous system. System excitation is not considered by samples in input variable in isolated times $t = kT$, $k = 0, 1, \ldots$, but we presume its modification by zero order hold (often) or first order hold (less often).

8.1. Zero order hold

Creates continuous output $y(t)$ from input samples $u(k)$ such that it hold on output for one sampling period the value of last received sample $y(t) = u(k)$, $kT \le t < (k+1)T$, $k = 0, 1, \ldots$.



8.2. First order hold

Creates continuous output from input samples by linear interpolation of samples

$$y(t) = u(k) + \frac{t - kT}{T}\left[u(k+1) - u(k)\right],\ kT \le t < (k+1)T,\ k = 0, 1, \ldots .$$

Note.

- Output of discretized continuous system is digital as it uses discrete description. However, the output should be continuous, as in reality it is continuous system with continuous output
- Response course more realistic than the one found using `step`, `impulse` etc can be found by simulation in Matlab–Simulink (using serial connection between hold and continuous system).

Matlab

```
% Discretization of continuous system with sampling period T
sys_d = c2d(sys, T);
sys_d = c2d(sys, T, method);
% method
%    'zoh' with zero order hold in input, transition
%          characteristics (default)
%    'foh' with first order hold in input, transition
%          characteristics
```

```
%    'imp' without hold on input, impulse characteristics
```

Note

- When using `'zoh'`, `'foh'` the response of discrete system corresponds to the transition characteristic of continuous system and doesn't have to precisely correspond to its impulse characteristic and vice versa – for `'imp'` the response of discrete system correspond to impulse characteristic of continuous system and does not have to correspond precisely to its transition characteristic.
- `step` does not show the sequence of samples as sequence of impulses, but as piecewise constant function. That correspond to the processing by zero order hold.

Example

```
ss_Mech_zoh = c2d(ss_Mech, 0.2);
ss_Mech_foh = c2d(ss_Mech, 0.2, 'foh');
ss_Mech_imp = c2d(ss_Mech, 0.2, 'imp');
step(ss_Mech, ss_Mech_zoh)
impulse(ss_Mech, ss_Mech_foh)
impulse(ss_Mech, ss_Mech_zoh)
```

8.3. Sampling period selection

To approximately determine the sampling period one of following formulas can be used

$$T \approx \frac{T_1}{10}$$

$$T \approx \left(\frac{1}{6} \div \frac{1}{15}\right) T_{95}$$

$$T \approx \left(\frac{1}{4} \div \frac{1}{8}\right) T_d$$

$$T \approx \left(\frac{1}{2} \div \frac{1}{4}\right) \sum_i \tau_i$$

Symbol description

T_1	highest time constant of controlled system
T_{95}	time of reaching $95\,\%$ of steady value on transition characteristic
$\sum_i \tau_i$	sum of controlled system time constants
T_d	dead time (delay)

In practice the value of sampling period is considered to be reasonable when control quality does not decrease for more than $15\,\%$.

Example

In Matlab

```
K_inf = dcgain(ss_Mech)
step(ss_Mech/K_inf, ss(0.95), 10)
% T95 from response
[T95/15 T95/6]

K_inf = dcgain(ss_Mot_2);
step(ss_Mot_2/K_inf, ss(0.95), 0.1)
% T95 from response
[T95/15 T95/6]

step(tf(ss_Drv)*s^2, 0.95*tf(0.5))
% T95 from response
[T95/15 T95/6]
```

9. Some properties of systems

9.1. Controllability

Is ability of inputs u influence state variables x.

Continuous system

State $\mathbf{x}(t_1) \neq \mathbf{0}$ of linear system is controllable if there is finite time $t_2 > t_1$ and such input $\mathbf{u}(t)$, which transfers system from state $\mathbf{x}(t_1)$ into state $\mathbf{x}(t_2) = \mathbf{0}$. System is controllable in time t, if each state $\mathbf{x}(t)$ is controllable.

Necessary and sufficient condition of controllability is that controllability matrix $\mathbf{M}_C$ of system of n-th order has rank of n.

Discrete system

State $\mathbf{x}(k_1) \neq \mathbf{0}$ of linear system is controllable if there is such input $\mathbf{u}(k)$, $k = k_1, k_1 + 1, \ldots, k_2$ which transfers system from state $\mathbf{x}(k_1)$ into state $\mathbf{x}(k_2) = \mathbf{0}$. The system is controllable in time k if each state $\mathbf{x}(k)$ is controllable.
Necessary and sufficient condition of controllability is that controllability matrix $\mathbf{M}_C$ of system of n-th order has rank of n.

Matlab

```
Mc = ctrb(sys);  % determination of controllability matrix for
                 % system sys
rank(Mc)         % rank of controllability matrix
```

9.2. Observability

Is ability of outputs $\mathbf{y}$ to give full information about state variables $\mathbf{x}$.

Continuous system

State $\mathbf{x}(t_0)$ of linear system is observable if it can be determined from previous values of output $\mathbf{y}(t)$, $t < t_0$ in finite time interval. System is observable in time t_0 if each state $\mathbf{x}(t_0)$ is observable.
Necessary and sufficient condition of controllability is that observability matrix $\mathbf{M}_O$ of system of n-th order has rank of n.

Discrete system

State $\mathbf{x}(k_0)$ of linear system is observable if it can be determined from previous values of output $\mathbf{y}(k)$, $k < k_0$ in finite time interval. System is observable in time k_0 if each state $\mathbf{x}(k_0)$ is observable

Necessary and sufficient condition of controllability is that observability matrix $\mathbf{M}_O$ of system of n-th order has rank of n.

Matlab

```
Mo = obsv(sys);  % determination of observability matrix for
                 % system sys
rank(Mo)         % rank of observability matrix
```

10. State feedback controller

State theory of control comes from the state description and can be used to control the system for which the conventional control theory does not give sufficient results.

State feedback controller in feedback in multidimensional control circuit enables to formulate overall dynamics of such circuit in a way that control goal

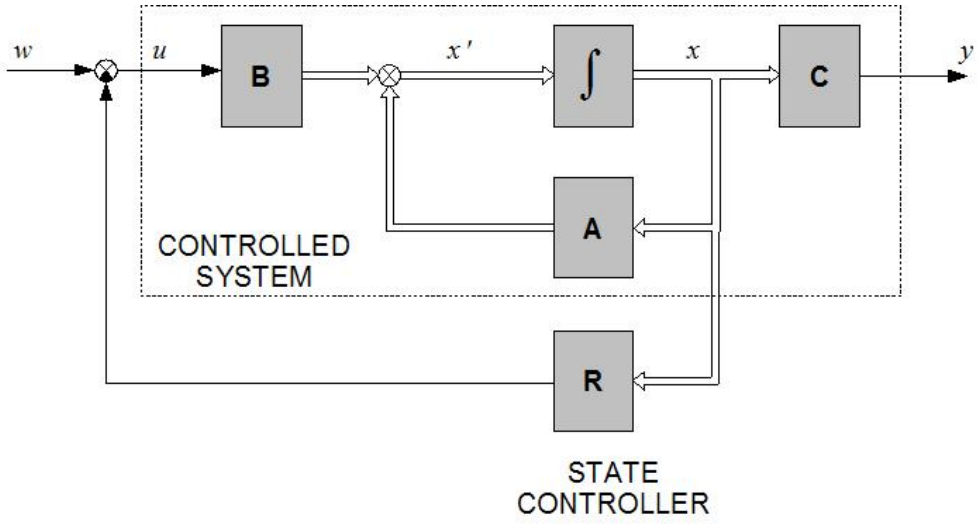
$$\mathbf{y} \rightarrow \mathbf{w}$$

is met in required quality.

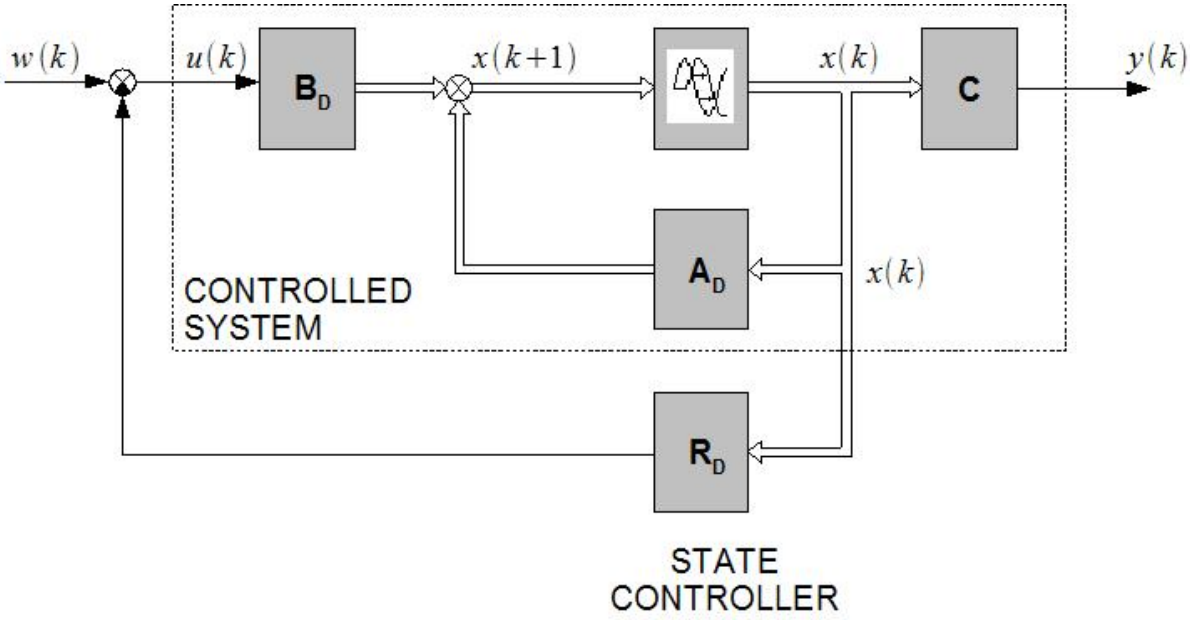
Principle

- Dynamic properties of continuous or discrete system are given by roots of systems characteristic equation, which are equal to eigenvalues of system matrix $\mathbf{A}$ or $\mathbf{A}_D$.
- Suitable choice of gain of state feedback controller can change the position of eigenvalues of system matrix for systems with closed loop

Graphical representation of continuous state feedback controller (SISO system)



Graphical representation of discrete state feedback controller (SISO system)



Mathematical details

Initial system

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$
$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

We search for gain matrix $\mathbf{R}$ of state controller $\mathbf{w}_\mathbf{R} = \mathbf{R}\mathbf{x}$ in closed loop in a way that $\mathbf{u} = \mathbf{w} - \mathbf{w}_\mathbf{R}$.

System with closed feedback

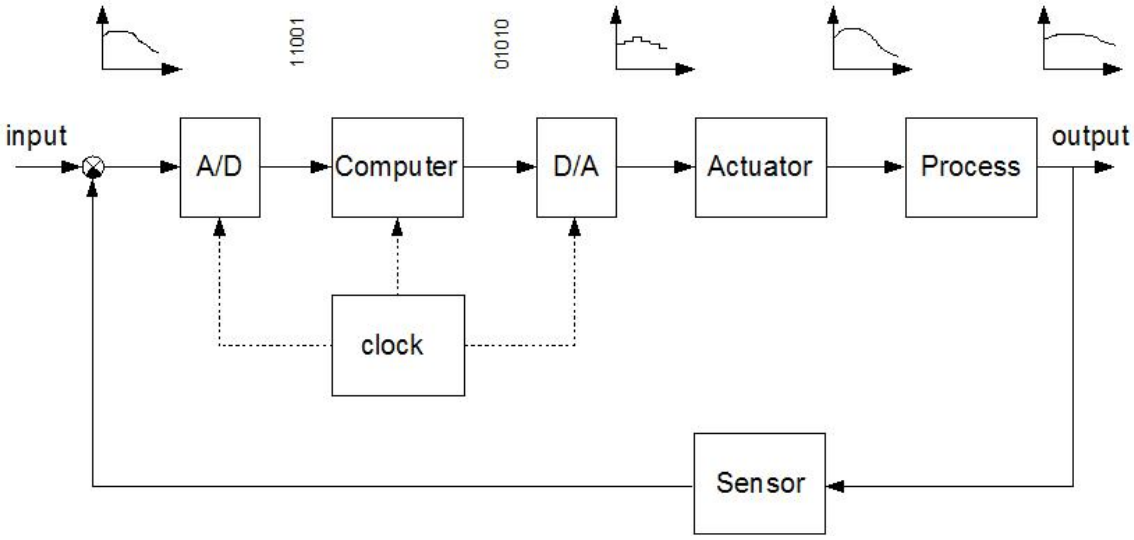
$$\mathbf{x}' = \mathbf{Ax} + \mathbf{Bu} = \mathbf{Ax} + \mathbf{B}\left(\mathbf{w} - \mathbf{w}_R\right) = \mathbf{Ax} + \mathbf{B}\left(\mathbf{w} - \mathbf{Rx}\right) = \left(\mathbf{A} - \mathbf{BR}\right)\mathbf{x} + \mathbf{Bw}$$

$$\mathbf{y} = \mathbf{Cx} + \mathbf{Du} = \mathbf{Cx} + \mathbf{D}\left(\mathbf{w} - \mathbf{w}_R\right) = \mathbf{Cx} + \mathbf{D}\left(\mathbf{w} - \mathbf{Rx}\right) = \left(\mathbf{C} - \mathbf{DR}\right)\mathbf{x} + \mathbf{Dw}$$

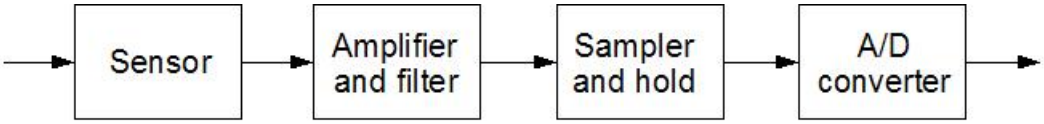
Note

- Gain matrix $\mathbf{R}$ of continuous controller is determined so the state matrix of continuous system with closed feedback $\mathbf{A} - \mathbf{BR}$ would contain required „continuous“ eigenvalues.
- Gain matrix $\mathbf{R}_D$ of discrete controller is determined the same way, so that the state matrix of discrete system (discretized in case of discrete control of continuous system) with closed feedback $\mathbf{A}_D - \mathbf{B}_D\mathbf{R}_D$ would contain required „discrete“ eigenvalues. In case of discrete control of continuous system the „continuous“ eigenvalues p_i are recalculated into discrete z_i using $z_i = e^{s_i T}$. Discrete eigenvalues z_i also depend on sampling period T.
- During design of discrete state controller we first design the continuous one and in next step the discrete controller is designed, which depends also on the choice of sampling period T.

Diagram of digital control and signal types



on computer input



on computer output

Matlab

```
% determination of R matrix for given A, B and given vector
% of eigenvalues cp of matrix A-B*R
R = place(A, B, cp);

% Note
% eigenvalues must be different
```

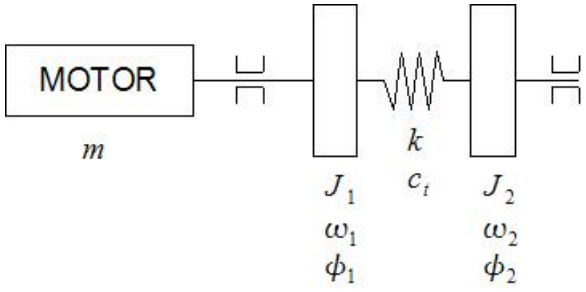
Note

- Structure of state controller is given by state vector $\mathbf{x}$. It depends on systems state model choice.
- To implement state controller it is necessary to have state vector $\mathbf{x}$ available, therefore to implement controller with given structure the values of all variables of state vector must be measured.
- Design ignores the output amplitude in steady state.
- The results must be verified by simulation in Matlab – Simulink, when discrete design is used on discretised system which is in fact continuous with (zero order) hold on input, as their time courses are different. Verification using `step`, etc. may not be suitable.
- To efficiently design state controller in Matlab – Simulink it is useful to use extended state model of controlled system which apart from current outputs shows into the outputs also the states. This affects the matrices of its state description e.g. like this

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$
$$\begin{bmatrix} \mathbf{y} \\ \mathbf{y}_X \end{bmatrix} = \begin{bmatrix} \mathbf{C} \\ \mathbf{I} \end{bmatrix} \mathbf{x} + \mathbf{D}\mathbf{u}$$

Example

Design continuous state controller of electric drive made of two mass system connected by flexible shaft `ss_Drv` (i.e. position controller - φ_2 is the output) in a way that system has dominant poles $p_{1,2} = -d\omega_0 \pm j\omega_0\sqrt{1-d^2}$ and remaining poles $p_3 = -3d\omega_0$, $p_4 = -10d\omega_0$, where $\omega_0 = 1.5$, $d = \frac{1}{\sqrt{2}}$.



State model of relation between motor torque and load rotor angular position is

$$\begin{bmatrix} \omega_1' \\ \omega_2' \\ \varphi_1' \\ \varphi_2' \end{bmatrix} = \begin{bmatrix} -\frac{c_t}{J_1} & \frac{c_t}{J_1} & -\frac{k}{J_1} & \frac{k}{J_1} \\ \frac{c_t}{J_2} & -\frac{c_t}{J_2} & \frac{k}{J_2} & -\frac{k}{J_2} \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \varphi_1 \\ \varphi_2 \end{bmatrix} + \begin{bmatrix} \frac{1}{J_1} \\ 0 \\ 0 \\ 0 \end{bmatrix} [m]$$

$$[\varphi_2] = \begin{bmatrix} 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \omega_1 \\ \omega_2 \\ \varphi_1 \\ \varphi_2 \end{bmatrix} + [0][m]$$

In Matlab

```
% system definition from previous example
% ...
% ss_Drv = ...

% is it meaningful to design controller?
% controllability check
Mc = ctrb(ss_Drv);  % controllability matrix
rank(Mc)            % controllability matrix rank

% rank equals to system order – OK

% roots of characteristic equation
eig(ss_Drv)

% there are 4 roots of systems characteristic equation
w0 = 1.5;
d = 1/sqrt(2);

p1 =    -d*w0+i*w0*sqrt(1-d^2);
p2 =    -d*w0-i*w0*sqrt(1-d^2);
p3 =  -3*d*w0;
p4 = -10*d*w0;

cp = [p1, p2, p3, p4]
R = place(ss_Drv.A, ss_Drv.B, cp) % controller R determination

% setting controlling system
ss_Drv_ =ss((ss_Drv.A - ss_Drv.B*R), ...
  ss_Drv.B, ...
  (ss_Drv.C - ss_Drv.D*R), ...
  ss_Drv.D);
```

```
% comparison of responses of controlled and controlling system
step(ss_Drv, ss_Drv_)  % not well-arranged view
% better for first 10 s of response
step(ss_Drv, ss_Drv_, 10)

% verification of steady state deflection
dcgain(ss_Drv_)

% preparation for Matlab–Simulink simulation
% extended system of controlled system
ss_Drv_st = ss(ss_Drv.A, [ss_Drv.B], eye(4), zeros(4, 1));
% gain matrices as systems with fixed gain
ss_R = ss(R);

% results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The value of steady angular position fi2 for motor torque
%    m = 1 N.m can be set by gain of controlling system input.
% 3. In requirement for jump change of motor torque m = 1 N.m
%    the angular position of rotor on load side is stabilized
%    after 8.6 s on value of fi2 = 13.2 rad.
% 4. For implementation the following must be measured:
%    angular velocity of rotor w1 on drive side, angular
%    velocity of rotor w2 on load side, angular position
%    of rotor fi1 on drive side and angular position of
%    rotor fi2 on load side.
```

Example

Transform continuous state feedback controller for drive control `ss_Drv` proposed in previous example into discrete one.

In Matlab

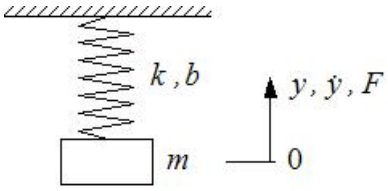
```
% continuous design of R, ss_Drv_ from previous example
% ...
% R = ...
% ss_Drv_ = ...

% estimate of sampling period
K_inf = dcgain(ss_Drv_);
step(ss_Drv_, ss(0.95*K_inf))
T95 = 2.45 % [s] % from response
[T95/15 T95/6]
% lets choose
T = 0.2;  % 0.1  % 0.05  % 0.02
```

```
% discretization of continuous drive model, ZOH on input
ss_Drv_zoh = c2d(ss_Drv, T);

% is it meaningful to design discrete controller?
% controllability check
Mc = ctrb(ss_Drv_zoh);  % controllability matrix
rank(Mc)            % controllability matrix rank

% rank equals to system order – OK

% required roots of characteristic equation of discrete system
% are determined from required roots of characteristic
% equation of continuous system

cp_d = exp(cp*T);
% will sampling period affect stability ?
abs(cp_d)

% root modules are smaller than 1 - OK

% discrete controller R_d determination
R_d = place(ss_Drv_zoh.A, ss_Drv_zoh.B, cp_d)

% setting controlling system
ss_Drv_zoh_ = ss((ss_Drv_zoh.A - ss_Drv_zoh.B*R_d), ...
  ss_Drv_zoh.B, ...
  (ss_Drv_zoh.C - ss_Drv_zoh.D*R_d), ...
  ss_Drv_zoh.D, ...
  T);

% comparison of responses of continuous and discrete
controlling
% system
step(ss_Drv_, ss_Drv_zoh_, 10)
step(ss_Drv_ / dcgain(ss_Drv_), ...
  ss_Drv_zoh_ / dcgain(ss_Drv_zoh_), 10)

% verification of steady state deflection
dcgain(ss_Drv_zoh_)

% preparation for Matlab–Simulink simulation
% extended system of controlled system by discretization
% of continuous system
ss_Drv_st_zoh = c2d(ss_Drv_st, T);
% gain matrix as systems with fixed gain
ss_R_zoh = ss(R_d);
```

```
% Results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The value of steady angular position fi2 for motor torque
%    m = 1 N.m can be set by gain of controlling system input.
% 3. In requirement for jump change of motor torque m = 1 N.m
%    the angular position of rotor on load side is stabilized
%    on value which is higher than for continuous sys.
% 4. For implementation the following must be measured:
%    angular velocity of rotor w1 on drive side, angular
%    velocity of rotor w2 on load side, angular position
%    of rotor fi1 on drive side and angular position of
%    rotor fi2 on load side.
% 5. For more detailed evaluation of reached behavior of
%    controlling system the simulation in Matlab–Simulink
%    must be performed.
```

Example

Design continuous controller of mechanical system `ss_Mech`, made by mass, spring and damper (i.e. position controller - y is the output) in a way that system speed is approximately the same and transition characteristics of deflection does not contain overshoot.



State model of mass deflection and load force is

$$\begin{bmatrix} y' \\ v_y' \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -k/m & -b/m \end{bmatrix} \begin{bmatrix} y \\ v_y \end{bmatrix} + \begin{bmatrix} 0 \\ 1/m \end{bmatrix} [F]$$

$$[y] = \begin{bmatrix} 1 & 0 \end{bmatrix} \begin{bmatrix} y \\ v_y \end{bmatrix} + [0][F]$$

In Matlab

```
% ss_Mech system definition from previous example
% ...

% is it meaningful to design controller?
% controllability check
Mc = ctrb(ss_Mech);  % controllability matrix
rank(Mc)             % controllability matrix rank

% rank equals to system order – OK
```

```
% roots of characteristic equation
eig(ss_Mech)

% Design of required roots of characteristic equation
% there are 2 required roots of characteristic equation
% to avoid overshoot they must be real, e.g.

d = 1.0005;
w0 = 1.4 * 5/d;

cp1 = -d*w0 + i*w0*sqrt(1-d^2);
cp2 = -d*w0 - i*w0*sqrt(1-d^2);

% which leads to response of controlling system
s = tf('s');
step(ss_Mech, tf(1/((s-cp1)*(s-cp2))))
% system speed is convenient

cp = [cp1, cp2]
R = place(ss_Mech.A, ss_Mech.B, cp)   % R controller

% setting controlling system
ss_Mech_ =ss((ss_Mech.A - ss_Mech.B*R), ...
  ss_Mech.B, ...
  (ss_Mech.C - ss_Mech.D*R), ...
  ss_Mech.D);

% comparison of responses of controlled and controlling system
step(ss_Mech, ss_Mech_)

% verification of steady state deflection
dcgain(ss_Mech_)

% preparation for Matlab–Simulink simulation
% extended system of controlled system
ss_Mech_st = ss(ss_Mech.A, ss_Mech.B, eye(2), zeros(2, 1));
% gain matrices as systems with fixed gain
ss_R = ss(R);

% results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The amplitude of steady deflection y for load force
%    F = 1 N can be set by gain of controlling system input.
% 3. for jump of load force F = 1 N the deflection
%    is stabilized after 1.45 s on value of y = 2.0e-4 m.
% 4. For implementation the following must be measured:
%    mass velocity vy and mass deflection y.
```

Example

Transform continuous state feedback controller for mechanical system control `ss_Mech` proposed in previous example into discrete one.

In Matlab

```
% continuous design of ss_Mech from previous example
% ...

% estimate of sampling period
K_inf = dcgain(ss_Mech_);
step(ss_Mech_, ss(0.95*K_inf))
T95 = 0.68 % [s] % from response
[T95/15 T95/6]

% lets choose
T = 0.05;

% discretization of continuous drive model, ZOH on input
ss_Mech_zoh = c2d(ss_Mech, T);

% is it meaningful to design discrete controller?
% controllability check
Mc = ctrb(ss_Mech_zoh);  % controllability matrix
rank(Mc)            % controllability matrix rank

% rank equals to system order – OK

% required roots of characteristic equation of discrete system
% are determined from required roots of characteristic
% equation of continuous system
cp_d = exp(cp*T);
% will sampling period affect stability ?
abs(cp_d)

% root modules are smaller than 1 - OK

% discrete controller R_d determination
R_d = place(ss_Mech_zoh.A, ss_Mech_zoh.B, cp_d)

% setting controlling system
ss_Mech_zoh_ = ss((ss_Mech_zoh.A - ss_Mech_zoh.B*R_d), ...
  ss_Mech_zoh.B, ...
  (ss_Mech_zoh.C - ss_Mech_zoh.D*R_d), ...
  ss_Mech_zoh.D, ...
  T);

% comparison of responses of continuous and discrete
controlling
% system
step(ss_Mech_, ss_Mech_zoh_)
```

```
step(ss_Mech_ / dcgain(ss_Mech_), ...
  ss_Mech_zoh_ / dcgain(ss_Mech_zoh_))

% verification of steadystate deflection
dcgain(ss_Mech_zoh_)

% preparation for Matlab–Simulink simulation
% extended system of controlled system by discretization
% of continuous system
ss_Mech_st_zoh = c2d(ss_Mech_st, T);
% gain matrix as systems with fixed gain
ss_R_zoh = ss(R_d);

% Results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The value of steady deflection y for load force F = 1 N
%    can be set by gain of controlling system input.
% 3. For jump of load force F = 1 N the deflection
%    is stabilized on value which is higher than
%    compared to continuous system.
% 4. for implementation the following must be measured:
%    mass velocity vy and mass deflection y.
% 5. For more detailed evaluation of reached behavior of
%    controlling system the simulation in Matlab–Simulink
%    must be performed.
```

11. Discrete control to finite steps count

("deadbeat" control).

Principle

- Gain matrix $\mathbf{R}_D$ of discrete controller is determined in a way to transfer controlling system from initial state $\mathbf{x}(k)$ to final state $\mathbf{x}(k+n)=0$ in n steps.
- This corresponds to the location of all "discrete" eigenvalues of state matrix of system with closed loop $\mathbf{A}_D-\mathbf{B}_D\mathbf{R}_D$ to the zero of z–plane.

Mathematical details

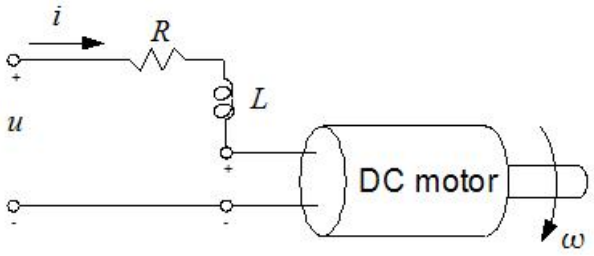
For input $\mathbf{u}=0$ the state equation of closed loop in particular steps

$$\begin{aligned}\mathbf{x}(k+1)&=(\mathbf{A}_D-\mathbf{B}_D\mathbf{R})\mathbf{x}(k)\\ \mathbf{x}(k+2)&=(\mathbf{A}_D-\mathbf{B}_D\mathbf{R})^2\mathbf{x}(k)\\ &\vdots\\ \mathbf{x}(k+N)&=(\mathbf{A}_D-\mathbf{B}_D\mathbf{R})^N\mathbf{x}(k)=0\end{aligned}$$

Last equation is met if the state matrix of system with closed loop is nilpotent, i.e. $(\mathbf{A}_D-\mathbf{B}_D\mathbf{R})^N=\mathbf{0}$ for $N>=n$. All eigenvalues of nilpotent matrix are zero.

Example

Design discrete state feedback controller for control of drive `ss_Mot_3` in finite number of steps.



State model of relation between voltage and angular position is

$$\begin{bmatrix}\varphi'\\ \omega'\\ i'\end{bmatrix}=\begin{bmatrix}0 & 1 & 0\\ 0 & 0 & C_e/J\\ 0 & -C_e/L & -R/L\end{bmatrix}\begin{bmatrix}\varphi\\ \omega\\ i\end{bmatrix}+\begin{bmatrix}0\\ 0\\ 1/L\end{bmatrix}[u] \qquad \text{state equation}$$

$$[\varphi]=\begin{bmatrix}1 & 0 & 0\end{bmatrix}\begin{bmatrix}\varphi\\ \omega\\ i\end{bmatrix}+[0][u] \qquad \text{output equation}$$

In Matlab

```
% system ss_Mot_3 definition from previous example
% ...

% estimate of sampling period from model of the same motor for
% angular velocity on output (ss_Mot_2)
K_inf = dcgain(ss_Mot_3)
% ss_Mot_3 system is not stable
eig(ss_Mot_3)
% ss_Mot_3 is not stable as its characteristic equation
% contains zero root
eig(ss_Mot_2)
% ss_Mot_2 is stable, as its characteristic equation contains
% only stable roots which are identical with roots of ss_Mot_3
% system. System ss_Mot_2 will be used for estimate of
% sampling period
K_inf = dcgain(ss_Mot_2);
step(ss_Mot_2/K_inf, ss(0.95))
T95 = 0.016 % [s] % from response
[T95/15 T95/6]
% lets choose
T = 0.005;

% discretization of continuous drive model, ZOH on input
ss_Mot_3_zoh = c2d(ss_Mot_3, T);

% is it meaningful to design discrete controller?
% controllability check
Mc = ctrb(ss_Mot_3_zoh);  % controllability matrix
rank(Mc)            % controllability matrix rank

% rank equals to system order – OK

% required roots of characteristic equation of discrete system
% for control on finite number of steps are equal to zero.
% However, “place” function requires differing roots as input.
% Therefore
cp1_d = 0;
cp2_d = 1e-1;
cp3_d = 2e-1;
cp_d = [cp1_d, cp2_d, cp3_d]

% discrete controller R_d determination
R_d = place(ss_Mot_3_zoh.A, ss_Mot_3_zoh.B, cp_d)

% setting controlling system
ss_Mot_3_zoh_ = ss((ss_Mot_3_zoh.A - ss_Mot_3_zoh.B*R_d), ...
  ss_Mot_3_zoh.B, ...
  (ss_Mot_3_zoh.C - ss_Mot_3_zoh.D*R_d), ...
  ss_Mot_3_zoh.D, ...
```

```
  T);
% root location check
eig(ss_Mot_3_zoh_)

% evaluation of discrete controlling system response
step(ss_Mot_3_zoh_ / dcgain(ss_Mot_3_zoh_))

% verification of steadystate deflection
dcgain(ss_Mot_3_zoh_)

% evaluation of response in Matlab–Simulink
% preparation for Matlab–Simulink simulation
% extended system of controlled system by discretization
% of continuous system
ss_Mot_3_st = ss(ss_Mot_3.A, ss_Mot_3.B, eye(3), zeros(3, 1));
ss_Mot_3_st_zoh = c2d(ss_Mot_3_st, T);
% gain matrix as system with fixed gain
ss_R_zoh = ss(R_d);
```

```
% Results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The value of steady angular position fi for armature
%    voltage u = 1 V can be set by gain of controlling
%    system input.
% 3. For more detailed evaluation of reached behavior
%    of controlling system the simulation in Matlab–Simulink
%    must be performed.
% 4. Controlling system reaches steady state in 0.03 s,
%    which is double the time of discrete model theory
%    (n*T = 0.015 s) while keeping the value of rotor angular
%    position 7.405e-4 rad.
% 5. For implementation the rotor angular position fi, rotor
%    angular velocity w and armature current i must be
%    measured.
```

12. State control with observer (state estimator)

Principle

State controller with observer consist of

- Observer which reconstructs state variables based on inputs and output of observed system. This enables to save sensors when implementing controller or to work with abstract states which can not be measured during the design of state feedback controller.
- State feedback controller, which does not use states measured directly on the system to be controlled, but uses state reconstructed by observer.

Observer structure

- Observer is represented by model of observed system and has its own feedback from deviation of real and observed (reconstructed) output variable, which is used for correction of its state into the state of observed system.
- Reconstructed state variables forms observer output.

Mathematical details

State model of observed system

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \mathbf{B}\mathbf{u}$$
$$\mathbf{y} = \mathbf{C}\mathbf{x} + \mathbf{D}\mathbf{u}$$

Observer estimates states from observed system. therefore it is based on the state equation of state model of observed system (which defines it dynamics), with input identical to observed system. In the case of state model absolutely precisely describing observed system, the observer could estimate the state only from input of observed system as

$$\hat{\mathbf{x}}' = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u}$$
$$\hat{\mathbf{y}} = \mathbf{I}\hat{\mathbf{x}}$$

But in reality no state model can describe any real system absolute precision. So it is necessary also to consider at least the difference $\Delta\mathbf{y} = \mathbf{y} - \left(\mathbf{C}\hat{\mathbf{x}} + \mathbf{D}\right)$ between the output of observed real system $\mathbf{y}$ and the output of system state model $\mathbf{C}\hat{\mathbf{x}} + \mathbf{D}$ which would be reached by observer without correction. Observer states must be corrected by this difference.
The simplest form of such states corrector is $\Delta\hat{\mathbf{x}} = \mathbf{H}\Delta\mathbf{y}$, where $\mathbf{H}$ is feedback matrix of observer.

So altogether

$$\Delta\hat{\mathbf{x}} = \mathbf{H}\Delta\mathbf{y} = \mathbf{H}\left(\mathbf{y} - \mathbf{C}\hat{\mathbf{x}} - \mathbf{D}\mathbf{u}\right)$$

$$\hat{\mathbf{x}}' = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \Delta\hat{\mathbf{x}} = \mathbf{A}\hat{\mathbf{x}} + \mathbf{B}\mathbf{u} + \mathbf{H}\left(\mathbf{y} - \mathbf{C}\hat{\mathbf{x}} - \mathbf{D}\mathbf{u}\right) = \left(\mathbf{A} - \mathbf{H}\mathbf{C}\right)\hat{\mathbf{x}} + \left(\mathbf{B} - \mathbf{H}\mathbf{D}\right)\mathbf{u} + \mathbf{H}\mathbf{y}$$
$$\hat{\mathbf{y}} = \mathbf{I}\hat{\mathbf{x}}$$

Inputs of observer are then $\mathbf{u}$ and $\mathbf{y}$, reconstructed state $\hat{\mathbf{x}}$ forms the output, i.e..

$$\hat{\mathbf{x}}' = \left(\mathbf{A} - \mathbf{H}\mathbf{C}\right)\hat{\mathbf{x}} + \begin{bmatrix}\mathbf{B} - \mathbf{H}\mathbf{D} & \mathbf{H}\end{bmatrix}\begin{bmatrix}\mathbf{u} \\ \mathbf{y}\end{bmatrix}$$
$$\hat{\mathbf{y}} = \mathbf{I}\hat{\mathbf{x}}$$

i.e.

$$\hat{\mathbf{A}} = \mathbf{A} - \mathbf{H}\mathbf{C},\ \hat{\mathbf{B}} = \begin{bmatrix}\mathbf{B} - \mathbf{H}\mathbf{D} & \mathbf{H}\end{bmatrix},\ \hat{\mathbf{C}} = \mathbf{I},\ \hat{\mathbf{D}} = \mathbf{0},\ \hat{\mathbf{u}} = \begin{bmatrix}\mathbf{u} \\ \mathbf{y}\end{bmatrix}$$

The essence in designing the observer is the design of its feedback matrix $\mathbf{H}$ in a way that the observer is stable and faster than observed system (to be able to observe it) as the whole controlling system.

Note

- Response of the system controlled by state controller with observer will be identical to the response of the system controlled by state controller if $\mathbf{y}$ and $\hat{\mathbf{y}}$ are identical. So the results must be verified by simulation in Matlab – Simulink, where $\mathbf{y}$ is output of continuous system and $\hat{\mathbf{y}}$ is output of discrete one with zero order hold, as their time courses are different. Verification using `step`, etc. is not suitable.

- To efficiently design observer in Matlab – Simulink it is useful to use extended state model of controlled system which apart from current outputs shows into the outputs also the states and further uses additional inputs, one for each state. This affects the matrices of its state description e.g. like this

$$\mathbf{x}' = \mathbf{A}\mathbf{x} + \begin{bmatrix}\mathbf{B} & \mathbf{I}\end{bmatrix}\begin{bmatrix}\mathbf{u} \\ \mathbf{u}_X\end{bmatrix}$$

$$\begin{bmatrix}\mathbf{y} \\ \mathbf{y}_X\end{bmatrix} = \begin{bmatrix}\mathbf{C} \\ \mathbf{I}\end{bmatrix}\mathbf{x} + \begin{bmatrix}\mathbf{D} & \mathbf{0} \\ \mathbf{0} & \mathbf{0}\end{bmatrix}\begin{bmatrix}\mathbf{u} \\ \mathbf{u}_X\end{bmatrix}$$

- Observer feedback matrix $\mathbf{H}$ can be designed analogically to matrix of state controller $\mathbf{R}$.
- Using function `R = place(A, B, cp)`, which calculates matrix $\mathbf{R}$ in a way that matrix $\mathbf{A} - \mathbf{BR}$ has required eigenvalues **cp** must be adjusted to calculation with matrix $\mathbf{A} - \mathbf{HC}$ and eigenvalues **op**.

  As $\mathbf{A} - \mathbf{HC} = \left((\mathbf{A} - \mathbf{HC})^T\right)^T = \left(\mathbf{A}^T - (\mathbf{HC})^T\right)^T = \left(\mathbf{A}^T - \mathbf{C}^T\mathbf{H}^T\right)^T$ it is necessary to use `H = place(A', C', op)'`.
- Feedback matrix $\mathbf{H}_D$ of discrete observer is determined by the same method in a way that state matrix of discrete observer $\mathbf{A}_D - \mathbf{H}_D\mathbf{C}$ using discrete observed system (or discretized in case of discrete observer of continuous system) would contain required "discrete" eigenvalues. In discrete observing of continuous system its own "continuous" eigenvalues p_i are recalculated into discrete ones z_i using $z_i = e^{s_i T}$. Discrete eigenvalues z_i depend on sampling period T.
- When designing discrete state controller with observer we first design the continuous one, then in next step we design discrete controller which further depends on selection of sampling period T.

Example

Design continuous state controller with observer for drive `ss_Drv`.

In Matlab

```
% ss_Drv, R from previous example
% ...
% R = ...
% ss_Drv_ = ...

% is it meaningful to design observer?
% observability check
Mo = obsv(ss_Drv);  % observability matrix
rank(Mo)            % observability matrix rank

% rank equals to system order – OK
```

```
% designing roots of observer characteristic equation
% roots of controlled system are
eig(ss_Drv)
% roots of controlling system are
eig(ss_Drv_)
% required roots of observer characteristic equation
% must ensure observer to be more stable and faster than
% both controlled and controlling system
op1 = -3+j;
op2 = -3-j;
op3 = -8;
op4 = -30;

op = [op1, op2, op3, op4]

% observer feedback matrix determination
H = place(ss_Drv.A', ss_Drv.C', op)'

% preparation for Matlab–Simulink simulation
% extended system of controlled system
ss_Drv_es = ss(ss_Drv.A, [ss_Drv.B, eye(4)], eye(4), zeros(4,
5));
% gain matrices as systems with fixed gain
ss_H = ss(H);

% results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The value of steady angular position fi2 for motor torque
%    m = 1 N.m can be set by gain of controlling
%    system input.
% 3. Transition course is practically the same in both
%    controlling system with or without observer. To evaluate
%    the influence of observer it is necessary to add noise
%    component to system output (to simulate noise
%    of the sensor).
% 4. For implementation only the rotor angular position fi2
%    must be measured.
% 5. For more detailed evaluation of reached behavior
%    of controlling system the simulation in Matlab–Simulink
%    must be performed.
```

Example

Discretize continuous state feedback controller by observer for control of drive `ss_Drv` suggested in previous example.

```
% continuous design of R, H, ss_Drv_ from previous example
% discrete design of R_d, ss_Drv_zoh_ from previous example
% ...
% R = ... (for cp roots)
% ss_Drv_ = ...
% T_ = ...
% H = ... (for roots op)
% ss_Drv_zoh = ...

% is it meaningful to design observer?
% observability check
Mo = obsv(ss_Drv_zoh);  % observability matrix
rank(Mo)            % observability matrix rank

% rank equals to system order – OK

% required roots of characteristic equation of discrete state
% observer are determined from required roots
% of characteristic equation of continuous state observer
op_d = exp(op*T);
% will sampling period affect stability ?
abs(op_d)

% root modules are smaller than 1 - OK

% determination of observer feedback matrix H
H_d = place(ss_Drv_zoh.A', ss_Drv_zoh.C', op_d)'

% preparation for Matlab–Simulink simulation
% extended system of controlled system by discretization
% of continuous system
ss_Drv_es_zoh = c2d(ss_Drv_es, T);
% gain matrix as systems with fixed gain
ss_H_zoh = ss(H_d);

% results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The value of steady angular position fi2 for motor torque
%    m = 1 N.m can be set by gain of controlling system input.
% 3. Transition course is practically the same in both
%    controlling system with or without observer. To evaluate
%    the influence of observer it is necessary to add noise
%    component to system output (to simulate noise
%    of the sensor).
% 4. For implementation only the rotor angular position fi2
```

```
%    must be measured.
% 5. For more detailed evaluation of reached behavior
%    of controlling system the simulation in Matlab–Simulink
%    must be performed.
```

Example

Design continuous state controller with observer of mechanic system `ss_Mech`.

In Matlab

```
% ss_Mech, R from previous example
% ...
% R = ...
% ss_Mech_ = ...

% is it meaningful to design observer?
% observability check
Mo = obsv(ss_Mech);  % observability matrix
rank(Mo)             % observability matrix rank

% rank equals to system order – OK

% designing roots of observer characteristic equation
% roots of controlled system are
eig(ss_Mech)
% roots of controlling system are
eig(ss_Mech_)
% required roots of observer characteristic equation
% must ensure observer to be more stable and faster than
% both controlled and controlling system
op1 = -7;
op2 = -8;

op = [op1, op2]

% observer feedback matrix determination
H = place(ss_Mech.A', ss_Mech.C', op)'

% preparation for Matlab–Simulink simulation
% extended system of controlled system
ss_Mech_es = ss(ss_Mech.A, [ss_Mech.B, eye(2)], eye(2),
zeros(2, 3));
% gain matrices as systems with fixed gain
ss_H = ss(H);

% results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
```

```
% 2. The value of steady deflection y for load force F = 1 N
%    can be set by gain of controlling system input.
% 3. Transition course is practically the same in both
%    controlling system with or without observer. To evaluate
%    the influence of observer it is necessary to add noise
%    component to system output (to simulate noise
%    of the sensor).
% 4. For implementation only the mass deflection y
%    must be measured.
% 5. For more detailed evaluation of reached behavior
%    of controlling system the simulation in Matlab–Simulink
%    must be performed.
```

Example

Discretize continuous state feedback controller by observer for control of drive `ss_Mech` suggested in previous example (deadbeat).

```
% continuous design of R, H, ss_Mech_ from previous example
% discrete design of R_d, ss_Mech_zoh_ from previous example
% ...
% R = ... (for cp roots)
% ss_Mech_ = ...
% T_ = ...
% H = ... (for op roots)
% ss_Mech_zoh = ...

% is it meaningful to design observer?
% observability check
Mo = obsv(ss_Mech_zoh);  % observability matrix
rank(Mo)            % observability matrix rank
% 2
% rank equals to system order – OK

% designing roots of observer characteristic equation
% roots of controlled system are
eig(ss_Mech)
% roots of controlling system are
eig(ss_Mech_)
% required roots of observer characteristic equation
% must ensure observer to be more stable and faster than
% both controlled and controlling system
op1 = -7;
op2 = -8;

op = [op1, op2]
```

```
% required roots of characteristic equation of discrete state
observer are determined from required roots of characteristic
equation of continuous state observer
op_d = exp(op*T);
% will sampling period affect stability ?
abs(op_d)

% root modules are smaller than 1 - OK

% determination of observer feedback matrix H
H_d = place(ss_Mech_zoh.A', ss_Mech_zoh.C', op_d)'

% preparation for Matlab–Simulink simulation
% extended system of controlled system by discretization
% of continuous system
ss_Mech_es_zoh = c2d(ss_Mech_es, T);
% gain matrix as systems with fixed gain
ss_H_zoh = ss(H_d);

% results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The value of steady deflection y for load force
%    F = 1 N can be set by gain of controlling system input.
% 3. Transition course is practically the same in both
%    controlling system with or without observer. To evaluate
%    the influence of observer it is necessary to add noise
%    component to system output (to simulate noise
%    of the sensor).
% 4. For implementation only the mass deflection y
%    must be measured.
% 5. For more detailed evaluation of reached behavior
%    of controlling system the simulation in Matlab–Simulink
%    must be performed.
```

Example

Design discrete state feedback controller with observer for control of drive `ss_Mot_3` designed for finite number of steps (deadbeat).

```
% discrete design R_d, ss_Mot_3_zoh_ from previous example
% ...
% T_ = ...
% R_d = ... (for cp_d roots)
```

```
% is it meaningful to design observer?
% observability check
Mo = obsv(ss_Mot_3_zoh);  % observability matrix
rank(Mo)            % observability matrix rank
% 3
% rank equals to system order – OK

% designing roots of observer characteristic equation
% roots of controlled system are
eig(ss_Mot_3_zoh)
% roots of controlling system are
eig(ss_Mot_3_zoh_)
% required roots of observer characteristic equation
% must ensure observer to be more stable and faster than
% both controlled and controlling system
op1_d = 0;
op2_d = 1e-2;
op3_d = 2e-2;

op_d = [op1_d, op2_d, op3_d]

% determination of discrete observer feedback matrix H
H_d = place(ss_Mot_3_zoh.A', ss_Mot_3_zoh.C', op_d)'

% preparation for Matlab–Simulink simulation
% extended system of controlled system by discretization
% of continuous system
ss_Mot_3_es_zoh = ss(ss_Mot_3_zoh.A, ...
[ss_Mot_3_zoh.B, eye(3)], eye(3), zeros(3, 4), T);
% gain matrix as systems with fixed gain
ss_H_zoh = ss(H_d);

% results discussion
% 1. Controller requirements met. It is meaningful to consider
%    further controller elements design.
% 2. The value of steady angular position fi for armature
%    voltage u = 1 V can be set by gain of controlling
%    system input.
% 3. For more detailed evaluation of reached behavior
%    of controlling system the simulation in Matlab–Simulink
%    must be performed.
% 4. Transition course is practically the same in both
%    controlling system with or without observer.
% 5. For implementation only the rotor angular position fi
%    must be measured.
```