# **Control Design MathScript Functions**

Use the Control Design MathScript functions to design, analyze, and simulate linear controller models using a <u>text-based language</u>. The following is a list of Control Design MathScript classes of functions and commands that LabVIEW MathScript supports.

The LabVIEW Control Design and Simulation Module <u>additional</u> <u>MathScript functions</u>.

The LabVIEW Digital Filter Design Toolkit installs <u>additional MathScript</u> <u>functions</u>.

Class	Description
<u>dynchar</u>	Dynamic characteristics functions
<u>frqrsp</u>	Frequency response analysis functions
<u>construct</u>	Model construction functions
<u>convert</u>	Model conversion functions
<u>connect</u>	Model interconnection functions
<u>reduce</u>	Model reduction functions
<u>ssdesign</u>	State-feedback design functions
<u>ssanals</u>	State-space analysis functions
<u>timeresp</u>	Time response analysis functions
<u>cdsolvers</u>	Equation solver functions
<u>cdutil</u>	Utility functions
<u>info</u>	Model information functions

# Interpreting Frequency and Time Response Data (Control Design and Simulation Module)

The frqrsp and timeresp MathScript classes contain functions that return frequency or time response data. The following explanation describes how to interpret the data these functions return.

These functions return matrices of real numbers. Each column of these matrices represents one input/output pair of the system model. Each row of these matrices represents the response at different frequencies or times. Therefore, each matrix element represents the response of a certain input-output pair at a certain frequency or time. These functions return a list of the frequencies or times used to evaluate the model.

Consider a multiple-input multiple-output (MIMO) model with two inputs three outputs. The first column is the data from input 1 and output 1. The second column is the data from input 1 and output 2. The third column is the data from input 1 and output 3. Because the model has only three outputs, the fourth column is the data from input 2 and output 1. This numbering continues as the column index numbers increase from left to right.

Using this same matrix, the first row is the data from the first frequency or time. The second row is the data from the second frequency or time. This numbering continues as the row index numbers increase from top to bottom.

Note This numbering applies to each output for functions with more than one output, such as the <u>bode</u> function. The bode function has two outputs, **mag** and **phase**. Each output is a separate matrix.

#### **Example: Viewing Step Response Data**

The following code constructs a state-space model with two states, two inputs, and two outputs. The code also returns the step response data from 0 seconds to 1 second in 0.10 second intervals.

```
A = [-9, 8; -1, 0]
B = [-1, 0; 0, 1]
C = [0.70711, 0.70711; -0.70711, 0.70711]
D = 0
SysIn = ss(A, B, C, D)
[Y, T] = step(SysIn, [0:0.10:1])
```

The following figure shows the  ${\bf Y}$  output, which returns the step response for each input-output pair.

Y =				
	0	0	0	0
	-0.043354	0.048673	0.091227	0.048673
	-0.054077	0.070543	0.20228	0.070543
	-0.05097	0.08037	0.31557	0.08037
	-0.042405	0.084786	0.42384	0.084786
	-0.032068	0.08677	0.52438	0.08677
	-0.021553	0.087661	0.61653	0.087661
	-0.011517	0.088062	0.70042	0.088062
	-0.002201	0.088242	0.77657	0.088242
	0.0063337	0.088323	0.84557	0.088323
	0.014104	0.088359	0.90806	0.088359

In the figure above, the first column is the step response of input 1/output 1, the second column is the step response of input 1/output 2, and so on. Each row of this matrix corresponds to a row of the T output, which returns the times at which this function evaluated the model. The following figure shows this output:

C =	
	о
	0.1
	0.2
	0.3
	0.4
	0.5
	0.6
	0.7
	0.8
	0.9
	1.0

Using this information, the step response of input 2/output 1 at 0.2 seconds is 0.20228.

# Deprecated Control Design MathScript Functions and Classes (Control Design and Simulation Module)

The following table lists deprecated MathScript functions/classes and the recommended replacement(s).

Deprecated Function or Class	<b>Replacement Function or Class</b>
c2dm	<u>c2d</u>
cloop	<u>feedback</u>
dbode	<u>bode</u>
ddamp	<u>damp</u>
dimpulse	<u>impulse</u>
dkalman	<u>kalman, kalmd</u>
dlsim	<u>Isim</u>
dreg	reg
freqresp	frqrsp
ss2tf	<u>tf</u>
tf2ss	SS

# cdsolvers (Control Design and Simulation Module, MathScript Class)

Use members of the cdsolvers class to solve continuous and discrete Lyapunov and algebraic Riccati equations.

Function	Description
<u>care</u>	Solves the continuous algebraic Riccati equation
<u>dare</u>	Solves the discrete algebraic Riccati equation
<u>dlyap</u>	Solves the discrete Lyapunov equation
<u>lyap</u>	Solves the continuous Lyapunov equation

# care (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdsolvers</u> class.

#### **Syntax**

X = care(A, B, Q)X = care(A, B, Q, R)X = care(A, B, Q, R, N)[X, eig] = care(A, B, Q)[X, eig] = care(A, B, Q, R)[X, eig, K] = care(A, B, Q, R, N)[X, eig, K] = care(A, B, Q, R)[X, eig, K] = care(A, B, Q, R)[X, eig, K] = care(A, B, Q, R, N)[X, eig, K, res] = care(A, B, Q, R)[X, eig, K, res] = care(A, B, Q, R)[X, eig, K, res] = care(A, B, Q, R)

### Description

Calculates the positive definite, or stabilizing, matrix **X** that solves the following continuous algebraic Riccati equation (CARE): A'X+XA-(XB+N)\*inv(R)\*(B'X+N')+Q = 0. This function also returns the

 $A^{X+XA}-(XB+N)^{*}(R)^{*}(B^{X+N})+Q = 0$ . This function also returns the gain matrix K and the eigenvalues of the matrix (A-BK).

**Examples** 

# Inputs

Name	Description
A	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
Q	Specifies the state weight matrix. <b>Q</b> is a real matrix that is symmetric and positive semi-definite.
R	Specifies the input weight matrix. The default is the identity matrix. <b>R</b> is a symmetric, positive definite real matrix.
Ν	Specifies the state-input cross weight matrix such that ( <b>Q</b> - <b>N</b> *inv( <b>R</b> )* <b>N</b> ') is positive semi-definite. The default is an appropriately sized matrix of zeros. <b>N</b> is a real matrix.

## Outputs

Name	Description
X	Returns the solution to the continuous algebraic Riccati equation. <b>X</b> is a real matrix.
eig	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>eig</b> is a complex vector.
К	Returns the gain matrix such that $\mathbf{K} = inv(\mathbf{R})^*(\mathbf{B}'\mathbf{X}+\mathbf{N}')$ . <b>K</b> is a real matrix.
res	Returns the residual, which is the sum of the absolute value of all the elements in all the matrices you specify. The value of the <b>res</b> output specifies the distance between the solution and zero. <b>res</b> is real scalar.

## Examples

A = [-1, 2; 0, -3] B = [0; 1] Q = [2, 0; 0, 2] R = 1[X, eig, K] = care(A, B, Q, R)

# **Related Topics**

<u>dare</u>

# dare (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdsolvers</u> class.

#### Syntax

- X = dare(A, B, Q) X = dare(A, B, Q, R) X = dare(A, B, Q, R, N) [X, eig] = dare(A, B, Q) [X, eig] = dare(A, B, Q, R) [X, eig, K] = dare(A, B, Q, R, N) [X, eig, K] = dare(A, B, Q, R) [X, eig, K] = dare(A, B, Q, R, N)[X, eig, K, res] = dare(A, B, Q, R)
- [X, eig, K, res] = dare(A, B, Q, R, N)

### Description

Calculates the positive semi-definite, or stabilizing, matrix **X** that solves the following discrete algebraic Riccati equation (DARE):

 $X = A'XA-[A'XB+N]^*inv(B'XB+R)*[A'XB+N]'+Q$ . This function also returns the gain matrix K and the eigenvalues of the matrix (A-BK).

**Examples** 

# Inputs

Name	Description
A	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
Q	Specifies the state weight matrix. <b>Q</b> is a real matrix that is symmetric and positive semi-definite.
R	Specifies the input weight matrix. The default is the identity matrix. <b>R</b> is a symmetric, positive definite real matrix.
Ν	Specifies the state-input cross weight matrix such that ( <b>Q</b> - <b>N</b> *inv( <b>R</b> )* <b>N</b> ') is positive semi-definite. The default is an appropriately sized matrix of zeros. <b>N</b> is a real matrix.

## Outputs

Name	Description
X	Returns the solution to the discrete algebraic Riccati equation. <b>X</b> is a real matrix.
eig	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>eig</b> is a complex vector.
К	Returns the gain matrix such that $K = inv(B'XB+R)^*(B'XA+N')$ . K is a real matrix.
res	Returns the residual, which is the sum of the absolute value of all the elements in all the matrices you specify. The value of the <b>res</b> output specifies the distance between the solution and zero. <b>res</b> is real scalar.

### Examples

A = [0.9, 0.1; 0, -0.25]B = [0; 1] Q = [2, 0; 0, 2] R = 1 [X, eig, K] = dare(A, B, Q, R)

# **Related Topics**

<u>care</u>

# dlyap (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdsolvers</u> class.

# Syntax

X = dlyap(A, Q)

#### Description

Calculates the matrix **X** that solves the discrete Lyapunov Equation: AXA'-X+Q = 0.

Examples

# Inputs

Name	Description
A	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
Q	Specifies a real matrix.

# Outputs

Name	Description
X	Returns the solution to the discrete Lyapunov Equation. <b>X</b> is a real matrix.

## Examples

A = [0.2, 1; 0, -0.25] Q = [2, 0; 0, 2] X = dlyap(A, Q)

# **Related Topics**

<u>lyap</u>

# Iyap (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdsolvers</u> class.

## Syntax

X = lyap(A, Q)X = lyap(A, B, Q)

### Description

Calculates the matrix **X** that solves the general form of the following Lyapunov equation: AX+XB = -Q.

**Examples** 

# Inputs

Name	Description
Α	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is <b>A</b> '. <b>B</b> is a real matrix.
Q	Specifies a real matrix.

# Outputs

Name	Description
X	Returns the solution to the continuous Lyapunov equation. <b>X</b> is a real matrix.

## Examples

A = [1, 1; 0, 2] Q = [2, 0; 0, 2]X = lyap(A, Q)

# **Related Topics**

<u>dlyap</u>

# cdutil (Control Design and Simulation Module, MathScript Class)

Use members of the cdutil class to perform miscellaneous tasks such as sorting complex vectors, constructing digital filters, and changing the direction in which you specify transfer function and filter coefficients.

Function	Description
<u>distdelay</u>	Minimizes transport delay in a system model
<u>dsort</u>	Sorts a complex vector in descending order
<u>esort</u>	Sorts a complex vector based on the real and imaginary parts of each element
<u>filt</u>	Constructs a digital filter in transfer function form
inv	Inverts a system model
<u>randvec</u>	Generates one or two random vectors
<u>transpose</u>	Transposes a multiple-input multiple-output (MIMO) system model

# distdelay (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdutil</u> class.

#### Syntax

SysDel = distdelay(SysIn)
[inputdist, outputdist, transdist] = distdelay(inputdelay, outputdelay, transdelay)
#### Description

Distributes the total delay of a system into input, output, and transport delays such that the transport delays are minimal. This function allocates the majority of the delays as input or output delays. The remaining delays become transport delays.

**Examples** 

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) system model in transfer function, zero-pole-gain, or state-space form.
inputdelay	Specifies the input delay of a model. For continuous models, this value is seconds. For discrete models, this value is in number of samples. <b>inputdelay</b> is a real vector.
outputdelay	Specifies the output delay of a model. For continuous models, this value is seconds. For discrete models, this value is in number of samples. <b>outputdelay</b> is a real vector.
transdelay	Specifies the transport delay of a model. For continuous models, this value is seconds. For discrete models, this value is in number of samples. <b>transdelay</b> is a real matrix.

### Outputs

Name	Description
SysDel	Returns the <b>SysIn</b> model with the total delay distributed to minimize the transport delay.
inputdist	Returns the <b>inputdelay</b> vector with the total delay distributed to minimize the transport delay.
outputdist	Returns the <b>outputdelay</b> vector with the total delay distributed to minimize the transport delay.
transdist	Returns the <b>transdelay</b> matrix with a minimal amount of transport delay.

#### Examples

inputdelay = [1 3 4];outputdelay = [2 3 5]; transdelay = [3 2 3; 4 3 1; 4 1 2]; [inputdist, outputdist, transdist] = distdelay(inputdelay, outputdelay, transdelay);

### **Related Topics**

pade delay2z hasdelay get set totaldelay

# dsort (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdutil</u> class.

## Syntax

[q, i] = dsort(l)

#### Description

Sorts a list of elements in a vector in descending order based on the magnitude of each element.

Examples

## Inputs

Name	Description	
I	Specifies a vector of elements you want to sort. I is a real or complex vector.	

# Outputs

Name	Description
q	Returns the list of elements in <b>I</b> sorted in descending order. <b>q</b> is a real or complex vector.
i	Returns the list of indexes this function used to sort <b>I</b> . <b>i</b> is an integer vector.

### Examples

l = [0.1, 0.2, 0.15, 0.3, 0.25][q, i] = dsort(l)

## **Related Topics**

<u>esort</u>

# esort (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdutil</u> class.

### Syntax

[q, i] = esort(l)

#### Description

Sorts a list of elements in a vector in descending order based on the real parts of each element. If two or more elements have identical real parts, this function sorts those elements in descending order based on the imaginary parts of each element.

**Examples** 

## Inputs

Name	Description	
I	Specifies a vector of elements you want to sort. I is a real or complex vector.	

# Outputs

Name	Description
q	Returns the list of elements in <b>I</b> sorted in descending order. <b>q</b> is a real or complex vector.
i	Returns the list of indexes this function used to sort <b>I</b> . <b>i</b> is an integer vector.

### Examples

l = [-1, -2, 1, 0, 2][q, i] = esort(l)

## **Related Topics**

<u>dsort</u>

# filt (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdutil</u> class.

#### Syntax

SysOutTF = filt(K) SysOutTF = filt(num, den) SysOutTF = filt(num, den, Ts)

#### Description

Converts a digital filter to a discrete transfer function system model. You also can use this function to create a transfer function model from a gain matrix  $\mathbf{K}$ .

**Examples** 

## Inputs

Name	Description
K	Specifies the gain matrix. $\mathbf{K}$ is a real matrix.
num	Specifies the coefficients, in descending order, of the numerator polynomial function of the digital filter. <b>num</b> is a real vector.
den	Specifies the coefficients, in descending order, of the denominator polynomial function of the digital filter. <b>den</b> is a real vector.
Ts	Specifies the sampling time of the discrete transfer function model this function creates. The default value is 1. <b>Ts</b> is a real scalar.

# Outputs

Name	Description
SysOutTF	Returns a discrete linear time-invariant (LTI) transfer function model that has a sampling time of <b>Ts</b> .

### Examples

num = [1, 1]den = [1, -1, 3] Ts = 0.2 SysOutTF= filt(num, den, Ts)

## **Related Topics**

<u>tf</u>

# inv (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdutil</u> class.

#### Syntax

SysInv = inv(SysIn)

## Description

Inverts a system model.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form. The model must neither have delay nor be strictly proper. For state-space models, the <b>D</b> matrix must be non-zero. For transfer function and zero-pole-gain models, the orders of the numerator and denominator polynomial
	functions must be identical.

# Outputs

Name	Description
SysInv	Returns the inverse of <b>SysIn</b> .

### Examples

SysIn = tf([1 2], [3 4]); SysInv = inv(SysIn)

## **Related Topics**

<u>balreal</u>

# randvec (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdutil</u> class.

#### Syntax

vectorX = randvec(covXX)

vectorX = randvec(meanX, covXX)

[vectorX, vectorY] = randvec(covXX, covYY, covXY)

[vectorX, vectorY] = randvec(meanX, covXX, covYY, covXY)

[vectorX, vectorY] = randvec(meanX, meanY, covXX, covYY, covXY)

#### Description

Generates one or more random vectors with specified mean, autocovariance, and cross-covariance information.

Examples
## Inputs

Name	Description
covXX	Specifies the auto-covariance of the <b>vectorX</b> vector. <b>covXX</b> is a real matrix and must be symmetric and positive semi-definite.
meanX	Specifies the mean of the <b>vectorX</b> vector. <b>meanX</b> is a real vector and determines the length of <b>vectorX</b> and the dimensions of <b>covXX</b> . The default value of <b>meanX</b> is 0.
covYY	Specifies the auto-covariance of the <b>vectorY</b> vector. <b>covYY</b> is a real matrix and must be symmetric and positive semi-definite.
meanY	Specifies the mean of the <b>vectorY</b> vector. <b>meanY</b> is a real vector and determines the length of <b>vectorY</b> and the dimensions of <b>covYY</b> . The default value of <b>meanY</b> is 0.
covXY	Specifies the cross-covariance between the <b>vectorX</b> and <b>vectorY</b> vectors. <b>covXY</b> is a real matrix. If <i>n</i> is the length of <b>meanX</b> and <i>m</i> is the length of <b>meanY</b> , <b>covXY</b> must be an <i>n</i> x <i>m</i> matrix.

## Outputs

Name	Description
vectorX	Returns a random vector with mean <b>meanX</b> , auto-covariance <b>covXX</b> , and cross-covariance <b>covXY</b> with the <b>vectorY</b> vector.
vectorY	Returns a random vector with mean <b>meanY</b> , auto-covariance <b>covYY</b> , and cross-covariance <b>covXY</b> with the <b>vectorX</b> vector.

#### Examples

```
eig_xy = abs(randn(1,7));diag_xy = diag(eig_xy);
U_xy = rand(7,7);
overall_cov = U_xy*diag_xy*U_xy';
covXX = overall_cov(1:4,1:4);
covYY = overall_cov(5:7,5:7);
covXY = overall_cov(1:4,5:7);
[vectorX, vectorY] = randvec(covXX, covYY, covXY);
```

# transpose (Control Design and Simulation Module, MathScript Function)

Member of the <u>cdutil</u> class.

#### Syntax

SysTrans = transpose(SysIn)

#### Description

Transposes a multiple-input multiple-output (MIMO) system model. The input-output pairs of the original system model become the output-input pairs of the transposed system model.

**Examples** 

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) system model in transfer function, zero-pole-gain, or state-space form.

## Outputs

Name	Description
SysTrans	Returns the transpose of the <b>SysIn</b> model.

#### Examples

z\_11 = 1;z\_12 = []; z\_21 = [2]; z\_22 = []; p\_11 = 2; p\_12 = 2; p\_21 = [1 3]; p\_22 = [1, 1]; K = [2 0; 1 2]; SysIn = zpk(z\_11, z\_12, z\_21, z\_22, p\_11, p\_12, p\_21, p\_22, K); SysTrans = transpose(SysIn);

# connect (Control Design and Simulation Module, MathScript Class)

Use members of the connect class to connect systems models together in various configurations.

Function	Description
<u>append</u>	Appends system models together
<u>diag</u>	Constructs a system model whose diagonal contains copies of another model
feedback	Connects two system models in a closed-loop configuration
hconcat	Horizontally concatenates two or more system models.
<u>parallel</u>	Connects two system models together in a parallel configuration
<u>series</u>	Connects two system models together in a serial configuration
vconcat	Vertically concatenates two or more system models

# append (Control Design and Simulation Module, MathScript Function)

Member of the <u>connect</u> class.

## Syntax

SysApp = append(SysIn\_1, SysIn\_2, ..., SysIn\_i)

#### Description

Connects system models together to produce an augmented model **SysApp**. The inputs and outputs of the **SysApp** model are the collective inputs and outputs of all input models. The input models must be either continuous models or discrete models with identical sampling times. Refer to the LabVIEW Control Design User Manual for information about appending models together.

**Examples** 

## Inputs

Name	Description
SysIn_i	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form. You must specify at least two models.

## Outputs

Name	Description
SysApp	Returns an LTI model consisting of SysIn_1, SysIn_2, SysIn_i appended together. SysApp is a model in transfer function, zero-pole-gain, or state-space form. If the input models are not of the same form, the following hierarchy determines the form of the SysApp model: state-space>zero- pole-gain>transfer function. For example, if the SysIn_1 model is in state-space form and the SysIn_2 model is in zero-pole- gain form, the SysApp model is in state-space form.

#### Examples

SysIn\_1 = tf([1, 1], [1, -1, 3])SysIn\_2 = zpk([1], [1, -1], 2) SysApp = append(SysIn\_1, SysIn\_2)

## **Related Topics**

feedback parallel series

# diag (Control Design and Simulation Module, MathScript Function)

Member of the <u>connect</u> class.

## Syntax

SysDiag = diag(SysIn, n)

#### Description

Constructs a system model whose diagonal contains  ${\bf n}$  copies of the  ${\bf SysIn}$  model.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) system model in transfer function, zero-pole-gain, or state-space form.
n	Specifies the number of times to append the <b>SysIn</b> model to the diagonal. <b>n</b> is an integer.

## Outputs

Name	Description
SysDiag	Returns a matrix whose diagonal contains <b>n</b> copies of the
	SysIn model. The remaining elements of this matrix are zeros.

#### Examples

```
z_1 = 1;p_1 = 2;
z_2 = [];
p_2 = [1, 1];
K = [2; 1];
SysIn = zpk(z_1, z_2, p_1, p_2, K);
SysDiag= diag(SysIn, 3);
```

## **Related Topics**

<u>append</u>

# feedback (Control Design and Simulation Module, MathScript Function)

Member of the <u>connect</u> class.

#### Syntax

SysClosed = feedback(SysIn\_1, SysIn\_2) SysClosed = feedback(SysIn\_1, SysIn\_2, sign)

#### Description

Connects two system models together to produce a closed-loop model using negative or positive feedback connections. The models must be either continuous models or discrete models with identical sampling times. Refer to the <u>LabVIEW Control Design User Manual</u> for information about connecting models together in a closed-loop configuration.

**Examples** 

## Inputs

Name	De	Description		
SysIn_1	Sp fur	ecifies a linear time-invariant (LTI) model in transfer nction, zero-pole-gain, or state-space form.		
SysIn_2	Sp fur	ecifies a linear time-invariant (LTI) model in transfer action, zero-pole-gain, or state-space form.		
sign	Specifies whether the feedback connections are nega positive. <b>sign</b> is an integer scalar that takes the follow values:			
	-1	(Default) Specifies that you want the feedback connections to be negative.		
	1	Specifies that you want the feedback connections to be positive.		

## Outputs

Name	Description
SysClosed	Returns a closed-loop LTI model consisting of <b>SysIn_1</b> and <b>SysIn_2</b> connected together in a closed-loop configuration. All connections are of the type you specify using the <b>sign</b> input. <b>SysClosed</b> is a model in transfer function, zero-pole-gain, or state-space form. If the input models are not of the same form, the following hierarchy determines the form of the <b>SysClosed</b> model: state-space>zero-pole-gain>transfer function. For example, if the <b>SysIn_1</b> model is in state-space form and the <b>SysIn_2</b> model is in zero-pole-gain form, the <b>SysClosed</b> model is in state-space form.

#### Examples

SysIn\_1 = tf([1, 1], [1, -1, 3]) SysIn\_2 = zpk([1], [1, -1], 1) SysClosed = feedback(SysIn\_1, SysIn\_2, 1)

## **Related Topics**

<u>append</u> parallel <u>series</u>

# hconcat (Control Design and Simulation Module, MathScript Function)

Member of the <u>connect</u> class.

## Syntax

SysHCon = hconcat(SysIn\_1, SysIn\_2, ..., SysIn\_n)

#### Description

Concatenates two or more system models such that the **SysHCon** model contains these models as columns.

Examples

## Inputs

Name	Description
SysIn_n	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form. You must specify at least two models. All models must have the same number of outputs.

## Outputs

Name	Description
SysHCon	Returns the concatenated system model.
#### Examples

SysIn\_1 = ss(eye(2),eye(2),eye(2),0);SysIn\_2 = ss(eye(3),eye(3),[1 0 0;0 0 1],0); SysHCon = hconcat(SysIn\_1, SysIn\_2);

## **Related Topics**

vconcat append

# parallel (Control Design and Simulation Module, MathScript Function)

Member of the <u>connect</u> class.

#### Syntax

SysPar = parallel(SysIn\_1, SysIn\_2)
SysPar = parallel(SysIn\_1, SysIn\_2, inlist\_1, inlist\_2, outlist\_1, outlist\_2)

#### Description

Connects two system models in parallel to produce a model **SysPar** with input and output connections you specify. The input models must be either continuous models or discrete models with identical sampling times. Refer to the <u>LabVIEW Control Design User Manual</u> for information about connecting models together in a parallel configuration.

**Examples** 

## Inputs

Name	Description
SysIn_1	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
SysIn_2	Specifies an LTI model in transfer function, zero-pole-gain, or state-space form.
inlist_1	Specifies the index numbers of the SysIn_1 inputs this function connects to the inputs of the SysIn_2 model. For example, if inlist_1 is [1, 4] and inlist_2 is [3, 5], SysPar connects the first input of SysIn_1 to the third input of SysIn_2 and the fourth input of SysIn_1 to the fifth input of SysIn_2. inlist_1 is an integer vector.
inlist_2	Specifies the index numbers of the <b>SysIn_2</b> inputs this function connects to the inputs of the <b>SysIn_1</b> model. <b>inlist_2</b> is an integer vector.
outlist_1	Specifies the index numbers of the <b>SysIn_1</b> outputs this function sums with the outputs of the <b>SysIn_2</b> model. For example, if <b>outlist_1</b> is [3, 6] and <b>outlist_2</b> is [3, 2], <b>SysPar</b> sums the third output of <b>SysIn_1</b> with the third output of <b>SysIn_2</b> and sums the sixth output of <b>SysIn_1</b> with the second output of <b>SysIn_2</b> . <b>outlist_1</b> is an integer vector.
outlist_2	Specifies the <b>SysIn_2</b> outputs this function sums to the <b>SysIn_1</b> outputs. <b>outlist_2</b> is an integer vector.

## Outputs

Name
SysPar

#### Examples

SysIn\_1 = tf([1, 1], [1 -1, 3])SysIn\_2 = zpk([1], [1, -1], 1) SysPar = parallel(SysIn\_1, SysIn\_2)

## **Related Topics**

append feedback series

# series (Control Design and Simulation Module, MathScript Function)

Member of the <u>connect</u> class.

#### Syntax

SysSer = series(SysIn\_1, SysIn\_2) SysSer = series(SysIn\_1, SysIn\_2, outlist, inlist)

#### Description

Connects two system models in series to produce a model **SysSer** with input and output connections you specify. The input models must be either continuous models or discrete models with identical sampling times. Refer to the <u>LabVIEW Control Design User Manual</u> for information about connecting models together in a serial configuration.

**Examples** 

## Inputs

Name	Description
SysIn_1	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form. This model is first in the series connection.
SysIn_2	Specifies an LTI model in transfer function, zero-pole-gain, or state-space form. This model is second in the series connection.
outlist	Specifies the index numbers of the outputs of the <b>SysIn_1</b> model to connect to the inputs of the <b>SysIn_2</b> model. For example, if <b>outlist</b> is [1, 3] and <b>inlist</b> is [1, 5], <b>SysSer</b> connects the first output of <b>SysIn_1</b> to the first input of <b>SysIn_2</b> and connects the third output of <b>SysIn_1</b> to the fifth input of <b>SysIn_2</b> . <b>outlist</b> is an integer vector.
inlist	Specifies the index numbers of the inputs to the <b>SysIn_2</b> model to connect to the outputs of the <b>SysIn_1</b> model. <b>inlist</b> is an integer vector.

## Outputs

Name	Description
SysSer	Returns an LTI model comprised of the <b>SysIn_1</b> and <b>SysIn_2</b> models connected in series. <b>SysSer</b> is a model in transfer function, zero-pole-gain, or state-space form. If the input models are not of the same form, the following hierarchy determines the form of the <b>SysSer</b> model: state-space>zero-pole-gain>transfer function. For example, if the <b>SysIn_1</b> model is in state-space form and the <b>SysIn_2</b> model is in zero-pole-gain form, the <b>SysSer</b> model is in state-space form.

#### Examples

SysIn\_1 = tf([1, 1], [1 -1, 3])SysIn\_2 = zpk([1], [1, -1], 1) SysSer = series(SysIn\_1, SysIn\_2)

## **Related Topics**

<u>append</u> feedback parallel

# vconcat (Control Design and Simulation Module, MathScript Function)

Member of the <u>connect</u> class.

#### Syntax

SysVCon = (SysIn\_1, SysIn\_2, ..., SysIn\_n)

#### Description

Concatenates two or more system models such that the **SysVCon** model contains these models as rows.

Examples

## Inputs

Name	Description
SysIn_n	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form. You must specify at least two models. All models must have the same number of inputs.

## Outputs

Name	Description
SysVCon	Returns the concatenated system model.

#### Examples

SysIn\_1= ss(eye(2),eye(2),eye(2),0);SysIn\_2 = ss(eye(3),[1 0;0 0;0 1],eye(3),0); SysVCon = vconcat(SysIn\_1, SysIn\_2)

## **Related Topics**

<u>hconcat</u> <u>append</u>

# construct (Control Design and Simulation Module, MathScript Class)

Use members of the construct class to construct linear time-invariant (LTI) system models and to convert between model forms.

Function	Description
<u>drss</u>	Generates a random discrete state-space system model
<u>drtf</u>	Generates a discrete random system model in transfer function form
<u>drzpk</u>	Generates a discrete random system model in zero-pole-gain (ZPK) form
<u>ord1</u>	Constructs the components of a first-order system model
ord2	Constructs the components of a second-order system model
<u>pid</u>	Constructs a proportional-integral-derivative (PID) controller model
<u>rss</u>	Generates a continuous random state-space system model
<u>rtf</u>	Generates a continuous random system model in transfer function form
<u>rzpk</u>	Generates a continuous random system model in zero-pole- gain (ZPK) form
<u>SS</u>	Creates a system model in, or converts a model to, state- space form
<u>tf</u>	Creates a system model in, or converts a model to, transfer function form
<u>zpk</u>	Constructs a system model in, or converts a model to, zero- pole-gain form

# drss (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

#### Syntax

SysOutSS = drss(n) SysOutSS = drss(n, r) SysOutSS = drss(n, r, m)

#### Description

Generates a discrete random linear time-invariant (LTI) system model in state-space form. The model is of an order you specify and has a sampling time of one second. You also can specify the number of inputs and outputs of the model.

**Examples** 

## Inputs

Name	Description
n	Specifies the order of the state matrix <b>A</b> of the <b>SysOutSS</b> model. <b>n</b> is the number of states and the dimensions of <b>A</b> . <b>n</b> is an integer scalar.
r	Specifies the number of outputs from the <b>SysOutSS</b> model. The default value is 1. <b>r</b> is an integer scalar.
m	Specifies the number of inputs to the <b>SysOutSS</b> model. The default value is 1. <b>m</b> is an integer scalar.

#### Outputs

Name	Description
SysOutSS	Returns an <i>n</i> -th-order discrete LTI model in <u>state-space</u> form. This model has <b>n</b> states, <b>r</b> outputs, and <b>m</b> inputs. The sampling time of this model is one second. You can use the d2d function to resample this model.

#### Examples

SysOutSS = drss(4, 3, 2)

## **Related Topics**

<u>rss</u> <u>ss</u> <u>ssdata</u>

# drtf (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

#### Syntax

SysOutTF = drtf(i, j, k, Ts)

#### Description

Generates a discrete random system model in transfer function form. <u>Examples</u>

## Inputs

Name	Description
i	Specifies the order of the model. The default value is 2.
j	Specifies the number of model outputs. The default value is 1.
k	Specifies the number of model inputs. The default value is 1.
Ts	Specifies the sampling time of the model, in seconds. The default value is 1.

## Outputs

Name	Description
SysOutTF	Returns a randomly generated discrete <b>i</b> -th-order system model in transfer function form. This model has <b>j</b> outputs, <b>k</b> inputs, and a sampling time of <b>Ts</b> .
# Examples

i = 1;j = 2; SysOutTF = drtf(i, j);

# **Related Topics**

rss drss tf tfdata rtf

# drzpk (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

### Syntax

SysOutZPK = drzpk(i, j, k, Ts)

### Description

Generates a discrete random system model in zero-pole-gain (ZPK) form.

Examples

# Inputs

Name	Description
i	Specifies the order of the model. The default value is 2.
j	Specifies the number of model outputs. The default value is 1.
k	Specifies the number of model inputs. The default value is 1.
Ts	Specifies the sampling time of the model, in seconds. The default value is 1.

# Outputs

Name	Description
SysOutZPK	Returns a randomly generated discrete <b>i</b> -th-order system model in ZPK form. This model has <b>j</b> outputs, <b>k</b> inputs, and a sampling time of <b>Ts</b> .

# Examples

i = 1;j = 2; SysOutZPK = drzpk(i , j);

# **Related Topics**

rzpk zpk zpkdata rss drss rtf

# ord1 (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

### Syntax

SysOutTF = ord1(K, tau, delay) [num, den] = ord1(K, tau, delay) [A, B, C] = ord1(K, tau, delay) [A, B, C, D] = ord1(K, tau, delay)

### Description

Constructs the components of a first-order system model based on a gain, time constant, and delay that you specify. You can use this function to create either a state-space model or a transfer function model, depending on the output parameters you specify.

#### **Examples**

# Inputs

Name	Description
К	Specifies the gain matrix. $\mathbf{K}$ is a real matrix.
tau	Specifies the time constant, in seconds, which is the time required for the model output to reach 63% of its final value. The default value is 0.
delay	Specifies the response delay of the model, in seconds. The default value is 0.

# Outputs

Name	Description
SysOutTF	Returns a transfer function model with gain <b>K</b> , time constant <b>tau</b> , and the specified <b>delay</b> . This model is single-input single-output (SISO).
num	Returns the coefficients of the numerator polynomial function of the resulting model. <b>num</b> is a real vector.
den	Returns the coefficients of the denominator polynomial function of the resulting model. <b>den</b> is a real vector.
Α	Returns the $n \ge n$ state matrix of the resulting model, where $n$ is the number of states. <b>A</b> is a real matrix.
В	Returns the $n \ge 1$ input matrix of the resulting model. <b>B</b> is a real vector.
С	Returns the 1 x <i>n</i> output matrix of the resulting model. <b>C</b> is a real vector.
D	Returns the direct transmission value of the resulting model. <b>D</b> is a real scalar.

# Examples

K = 0.5;tau = 1.5; SysOutTF = ord1(K, tau);

# **Related Topics**

<u>ord2</u>

# ord2 (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

### Syntax

[A, B, C, D] = ord2(wn, dr) [num, den] = ord2(wn, dr)

### Description

Constructs the components of a second-order system model based on a damping ratio and natural frequency you specify. You can use this function to create either a state-space model or a transfer function model, depending on the output parameters you specify.

#### **Examples**

# Inputs

Name	Description
wn	Specifies the natural frequency of the resulting model. <b>wn</b> is a real scalar.
dr	Specifies the damping ratio of the resulting model. <b>dr</b> is a real scalar.

# Outputs

Name	Description
Α	Returns the $n \ge n$ state matrix of the resulting model, where $n$ is the number of states. <b>A</b> is a real matrix.
В	Returns the $n \ge m$ input matrix of the resulting model, where $m$ is the number of inputs. <b>B</b> is a real matrix.
С	Returns the $r \ge n$ output matrix of the resulting model, where $r$ is the number of outputs. <b>C</b> is a real matrix.
D	Returns the $r \ge m$ direct transmission matrix of the resulting model. <b>D</b> is a real matrix.
num	Returns the coefficients of the numerator polynomial function of the resulting model. <b>num</b> is a real vector.
den	Returns the coefficients of the denominator polynomial function of the resulting model. <b>den</b> is a real vector.

### Examples

dr = 0.5wn = 20 [num, den] = ord2(wn, dr) SysTF = tf(num, den) [A, B, C, D] = ord2(wn, dr) SysSS = ss(A, B, C, D)

# **Related Topics**

<u>damp</u> <u>tf</u> <u>ss</u>

# pid (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

### Syntax

SysOutTF = pid(Kc, Ki, Kd, 'parallel') SysOutTF = pid(Kc, Ti, Td, 'series') SysOutTF = pid(Kc, Ti, Td, 'academic') [num, den] = pid(Kc, Ti, Td)

### Description

Constructs a proportional-integral-derivative (PID) controller model in either parallel, series, or academic form. Refer to the <u>LabVIEW Control</u> <u>Design User Manual</u> for information about these three forms.

**Examples** 

# Inputs

Name	Description
Кс	Specifies the proportional gain of the controller model.
Ki	Specifies the integral gain of a parallel controller. This input adjusts the effect of the error integral on the controller output. The default value is 0, which specifies no integral action.
Kd	Specifies the derivative gain of a parallel controller. This input adjusts the effect of the error derivative on the controller output. The default value is 0, which specifies no derivative action.
Ti	Specifies the integral time constant of a series or academic controller. This input adjusts the effect of the error integral on the controller output. The default value is 0, which specifies no integral action.
Td	Specifies the derivative time constant of a series or academic controller. The default value is 0, which specifies no derivative action.
'parallel', 'series', or 'academic'	Specifies the structure of PID controller model you want to construct. The default value is 'academic'. This input is a string.

### Outputs

Name	Description
SysOutTF	Returns a PID controller model in transfer function form.
num	Returns the coefficients of the numerator polynomial function of the PID controller model. <b>num</b> is a real vector.
den	Returns the coefficients of the denominator polynomial function of the PID controller model. <b>den</b> is a real vector.

# Examples

Kc = 0.5;Ti = 0.25; SysOutTF = pid(Kc, Ti, 'academic');

# **Related Topics**

<u>tf</u> rtf drtf

# rss (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

### Syntax

SysOutSS = rss(n) SysOutSS = rss(n, r) SysOutSS = rss(n, r, m)

### Description

Generates a continuous random linear time-invariant (LTI) system model in state-space form. The model is of an order you specify. You also can specify the number of inputs and outputs of the model.

**Examples** 

# Inputs

Name	Description
n	Specifies the order of the state matrix <b>A</b> of the <b>SysOutSS</b> model. <b>n</b> is the number of states and the dimensions of <b>A</b> . <b>n</b> is an integer scalar.
r	Specifies the number of outputs from the resulting state-space model. The default value is 1. <b>r</b> is an integer scalar.
m	Specifies the number of inputs to the resulting state-space model. The default value is 1. <b>m</b> is an integer scalar.

# Outputs

Name	Description
SysOutSS	Returns an <i>n</i> -th-order continuous LTI model in state-space form. This model has <b>n</b> states, <b>r</b> outputs, and <b>m</b> inputs.

# Examples

n = 4m = 2r = 3 SysOutSS = rss(n, r, m)
## **Related Topics**

<u>drss</u> <u>ss</u>

# rtf (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

#### Syntax

SysOutTF = rtf(i, j, k)

### Description

Generates a continuous random system model in transfer function form. <u>Examples</u>

## Inputs

Name	Description
i	Specifies the order of the model. The default value is 2.
j	Specifies the number of model outputs. The default value is 1.
k	Specifies the number of model inputs. The default value is 1.

#### Outputs

Name	Description
SysOutTF	Returns a randomly generated continuous <b>i</b> -th-order system model in transfer function form. This model has <b>j</b> outputs and <b>k</b> inputs.

# Examples

i = 1;j = 2; SysOutTF = rtf(i, j);

## **Related Topics**

rss drss tf tfdata drtf rzpk drzpk

# rzpk (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

#### Syntax

SysOutZPK = rzpk(i, j, k)

#### Description

Generates a continuous random system model in zero-pole-gain (ZPK) form.

Examples

## Inputs

Name	Description
i	Specifies the order of the model. The default value is 2.
j	Specifies the number of model outputs. The default value is 1.
k	Specifies the number of model inputs. The default value is 1.

#### Outputs

Name	Description
SysOutZPK	Returns a randomly generated continuous <b>i</b> -th-order system model in ZPK form. This model has <b>j</b> outputs and <b>k</b> inputs.

# Examples

i = 1;j = 2; SysOutZPK = rzpk(i , j);

## **Related Topics**

drzpk zpk zpkdata rss drss rtf

# ss (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

#### Syntax

SysOutSS = ss(D) SysOutSS = ss(A, B) SysOutSS = ss(A, B, C) SysOutSS = ss(A, B, C, D) SysOutSS = ss(A, B, C, D, Ts) SysOutSS = ss(SysIn)

#### Description

Constructs a continuous or discrete linear time-invariant (LTI) system model in state-space form. You also can use this function to convert transfer function and zero-pole-gain models to state-space form.

**Examples** 

## Inputs

Name	Description
A	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
D	Specifies an $r \ge m$ direct transmission matrix. The default is an empty matrix. <b>D</b> is a real matrix.
Ts	Specifies the discrete sampling time of the <b>SysOutSS</b> model. The default value is 0, which constructs a continuous model. Specify a non-zero value to construct a discrete model. <b>Ts</b> is a real scalar.
SysIn	Specifies an LTI transfer function or zero-pole-gain model that you want to convert to state-space form.

### Outputs

Name	Description
SysOutSS	Returns an LTI state-space model. This model has $n$ states, $m$ inputs, $r$ outputs, and a sampling time of <b>Ts</b> . This model is single-input single-output (SISO), single-input multiple-output (SIMO), multiple-input single-output (MISO), or multiple-input multiple-output (MIMO), depending on the dimensions of the <b>B</b> and <b>C</b> matrices.

#### Examples

```
% Creates a state-space modelA = eye(2)
B = [0; 1]
C = B'
SysOutSS = ss(A, B, C)
% Converts a zero-pole-gain model to state-space form
z = 1
p = [1, -1]
k = 1
SysIn = zpk(z, p, k)
SysOutSS = ss(SysIn)
```

## **Related Topics**

ssdata tf zpk drss rss minreal

# tf (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

#### Syntax

SysOutTF = tf(K) SysOutTF = tf(num\_11, den\_11) SysOutTF = tf(num\_11, den\_11, Ts) SysOutTF = tf(num\_11, num\_12, ..., num\_ij, den\_11, den\_12, ..., den\_ij, [r, m]) SysOutTF = tf(num\_11, num\_12, ..., num\_ij, den\_11, den\_12, ..., den\_ij, [r, m], Ts) s = tf('s') z = tf('z', Ts) SysOutTF = tf(SysIn)

#### Description

Creates a continuous or discrete linear time-invariant (LTI) system model in transfer function form. You also can use this function to convert zeropole-gain and state-space models to transfer function form.

**Examples** 

## Inputs

Name	Description
К	Specifies the gain. <b>K</b> is a real scalar or a real matrix, depending on whether you want to construct a single-input single-output (SISO) or multiple-input multiple-output (MIMO) model.
num_ij	Specifies the coefficients of the numerator polynomial function, in descending order. The order of the numerator polynomial function must be less than or equal to the order of the denominator polynomial function. The coefficients you specify for the <b>num_ij</b> input apply to the transfer function equation at the <i>i</i> -th input and <i>j</i> -th output of a MIMO model. <b>num_ij</b> is either a real scalar or a real vector, depending on the number of coefficients you specify.
den_ij	Specifies the coefficients of the denominator polynomial function, in descending order. The order of the denominator polynomial function must be greater than or equal to the order of the numerator polynomial function. The coefficients you specify for the <b>den_ij</b> input apply to the transfer function equation at the <i>i</i> -th input and <i>j</i> -th output of a MIMO model. <b>den_ij</b> is either a real scalar or a real vector, depending on the number of coefficients you specify.
r	Specifies the number of outputs from the transfer function. <b>r</b> must be consistent with the number of numerator and denominator polynomial functions you specified. <b>r</b> is an integer scalar.
m	Specifies the number of inputs to the transfer function. <b>m</b> must be consistent with the number of numerator and denominator polynomial functions you specified. <b>m</b> is an integer scalar.
Ts	Specifies the discrete sampling time of the <b>SysOutTF</b> model. The default value is 0, which creates a continuous model. Specify a non-zero value to construct a discrete model. <b>Ts</b> is a real scalar.
SysIn	Specifies the LTI model you want to convert to transfer function form.
's'	Specifies that you want to create the continuous transfer

	function s/1. After you enter this command, you can use LabVIEW MathScript operands on this transfer function to define a zero-pole-gain or transfer function model. For example, after you enter s = tf('s'), entering SysOutZPK = $4*(s+2)/(s+1)$ constructs a zero-pole-gain model with a gain of 4, a zero at -2, and a pole at -1. You also can create a transfer function model. For example, entering SysOutTF = $(3*(s*s*s)+2)/(4*(s*s*s*s)+8)$ constructs the transfer function model $3s^3+2/4s^4+8$ . 's' is a string constant.
'z'	Specifies that you want to create the discrete transfer function $z/1$ . The default sampling time <b>Ts</b> of this model is 1 second. After you enter this input, you can use LabVIEW MathScript operands on this transfer function to define a zero-pole-gain or transfer function model. 'z' is a string constant.

## Outputs

Name
SysOutTF

#### Examples

```
% Creates a continuous MISO transfer function modelnum_11 = 1
num_21 = 2
den_11 = [1, 1]
den_21 = [1, -1]
r = 1
m = 2
SysTF = tf(num_11, num_21, den_11, den_21, [r, m])
% Converts a zero-pole-gain model to transfer function form
z = 1
p = [1, -1]
k = 1
SysIn = zpk(z, p, k)
SysTF = tf(SysIn)
% Creates a discrete transfer function by adding and multiplying z/1 together
z = tf('z', 0.1)
SysOutTF = (9.9E-2*z+9.97)/(z*z-1.99*z+0.996)
```

## **Related Topics**

<u>ss</u> tfdata zpk

# zpk (Control Design and Simulation Module, MathScript Function)

Member of the <u>construct</u> class.

#### Syntax

SysOutZPK = zpk(K) SysOutZPK = zpk(K, Ts) SysOutZPK = zpk(z, p, K) SysOutZPK = zpk(z, p, K, Ts) SysOutZPK = zpk(z\_1, z\_2, ..., z\_ij, p\_1, p\_2, ..., p\_ij, K) SysOutZPK = zpk(z\_1, z\_2, ..., z\_ij, p\_1, p\_2, ..., p\_ij, K, Ts) SysOutZPK = zpk(SysIn)

#### Description

Creates a continuous or discrete linear time-invariant (LTI) system model in zero-pole-gain form. You also can use this function to convert transfer function and state-space models to zero-pole-gain form.

**Examples** 

## Inputs

Name	Description
К	Specifies the gain. <b>K</b> is an $m \ge r$ real matrix, where $m$ is the number of model inputs and $r$ is the number of model outputs.
Ts	Specifies the discrete sampling time of the <b>SysOutZPK</b> model. The default value is 0, which constructs a continuous model. Specify a non-zero value to construct a discrete model. <b>Ts</b> is a real scalar.
z_ij	Specifies the locations of the zeros of the <b>SysOutZPK</b> model. The coefficients you specify for the <b>z_ij</b> input apply to the zero- pole-gain equation at the <i>i</i> -th input and <i>j</i> -th output of a MIMO model. <b>z</b> is a complex scalar or complex vector, depending on whether you want the model to be SISO or MIMO.
p_ij	Specifies the locations of the poles of the <b>SysOutZPK</b> model. The coefficients you specify for the <b>z_ij</b> input apply to the zero- pole-gain equation at the <i>i</i> -th input and <i>j</i> -th output of a MIMO model. <b>p</b> is a complex scalar or complex vector, depending on whether you want the model to be SISO or MIMO.
SysIn	Specifies an LTI model you want to convert to zero-pole-gain form.

# Outputs

Name	Description
SysOutZPK	Returns an LTI zero-pole-gain model with zeros at <b>z</b> , poles at <b>p</b> , gain <b>K</b> , and sampling time <b>Ts</b> . The <b>SysOutZPK</b> model is SISO, single-input multiple-output (SIMO), multiple-input single-output (MISO), or MIMO depending on the dimensions of <b>K</b> .

#### Examples

% Creates a continuous zero-pole-gain modelz\_1 = 1 p\_1 = 2 z\_2 = [] p\_2 = [1, 1] K = [2; 1] SysOutZPK = zpk(z\_1, z\_2, p\_1, p\_2, K)

% Converts a transfer function model to zero-pole-gain form num = 1 den = [1, -1] SysInTF = tf(num, den) SysOutZPK = zpk(SysInTF)
## **Related Topics**

<u>ss</u> <u>tf</u> zpkdata

# convert (Control Design and Simulation Module, MathScript Class)

Use members of the convert class to convert a continuous system model to a discrete model, convert a discrete model to a continuous model, and resample a discrete model. You also can use members of this class to incorporate delays into a system model.

Function	Description
<u>c2d</u>	Converts a continuous system model to a discrete model
<u>d2c</u>	Converts a discrete system model to a continuous one
<u>d2d</u>	Resamples a discrete system model
<u>delay2z</u>	Incorporates delays into a discrete system model
<u>pade</u>	Incorporates delays into a continuous system model by using Pade approximation
<u>ss2ss</u>	Applies a state transformation to a system model

# c2d (Control Design and Simulation Module, MathScript Function)

Member of the <u>convert</u> class.

#### Syntax

SysDis = c2d(SysCon, Ts) SysDis = c2d(SysCon, Ts, method)

SysDis = c2d(SysCon, Ts, 'prewarp', w)

[SysDis, Ic] = c2d(SysCon, Ts)

[SysDis, Ic] = c2d(SysCon, Ts, method)

[SysDis, Ic] = c2d(SysCon, Ts, 'prewarp', w)

#### Description

Converts a continuous system model to a discrete model using the **method** you specify. This function also returns the initial condition conversion matrix. Refer to the <u>LabVIEW Control Design User Manual</u> for information about these conversion methods.

**Examples** 

## Inputs

Name	Descriptio	n
SysCon	Specifies a transfer fun	continuous linear time-invariant (LTI) model in ction, zero-pole-gain, or state-space form.
Ts	Specifies th a real scala	ne sampling time of the discrete model <b>SysDis</b> . <b>Ts</b> is ar.
<b>method</b> Specifies the method this function uses to convert the model to the <b>SysDis</b> model. <b>method</b> is a string that of the following values:		
	'zoh'	(Default) Converts the model using the Zero- Order-Hold method.
	'tustin'	Converts the model using Tustin's method.
	'forward'	Converts the model using the Forward Rectangular method.
	'backward'	Converts the model using the Backward Rectangular method.
	'foh'	Converts the model using the First-Order-Hold method.
	'matched'	Converts the model using the Matched Pole-Zero method.
	'prewarp'	Converts the model using the Prewarp method. If you enter this option, you also must specify the prewarp frequency <b>w</b> as an input to this function.
	'z transform'	Converts the model using the Z-Transform method.
w	Specifies th default valu	ne prewarp frequency, in radians/second. The lie is 4. <b>w</b> is a real scalar.

## Outputs

Name	Description
SysDis	Returns a discrete system with sampling time <b>Ts</b> in transfer function, zero-pole-gain, or state-space form.
IC	Returns the initial condition conversion matrix, which this function uses to discretize the continuous initial conditions vector. <b>Ic</b> is a real matrix.

### Examples

SysCon = ss(-1, 1, 2, 0) [SysDis, Ic] = c2d(SysCon, 0.1, 'foh')

## **Related Topics**

<u>d2c</u> <u>d2d</u>

# d2c (Control Design and Simulation Module, MathScript Function)

Member of the <u>convert</u> class.

#### Syntax

SysCon = d2c(SysDis) SysCon = d2c(SysDis, method) SysCon = d2c(SysDis, 'prewarp', w)

#### Description

Converts a discrete system model to a continuous model using the **method** you specify. Refer to the <u>LabVIEW Control Design User Manual</u> for information about these conversion methods.

**Examples** 

## Inputs

Name	Description	
SysDis	Specifies or state-s	a discrete model in transfer function, zero-pole-gain, pace form.
method	<b>hod</b> Specifies the method this function uses to convert the <b>SysD</b> model to the <b>SysCon</b> model. <b>method</b> is a string that can tak the following values:	
	'zoh'	(Default) Converts the model using the Zero-Order- Hold method.
	'tustin'	Converts the model using Tustin's method.
	'prewarp'	Converts the model using the Prewarp method. If you enter this option, you also must specify the prewarp frequency <b>w</b> as an input to this function.
w	Specifies the prewarp frequency, in radians/second. The default value is 4. <b>w</b> is a real scalar.	

## Outputs

Name	Description
SysCon	Returns a continuous model in transfer function, zero-pole-
	gain, or state-space form.

#### Examples

A = [0.5, 0.25; 0, 0.1] B = [1, 0]' C = [-1, 0] SysDis = ss(A, B, C, 0, 0.1) SysCon = d2c(SysDis, 'prewarp', 2)

## **Related Topics**

<u>c2d</u> <u>d2d</u>

# d2d (Control Design and Simulation Module, MathScript Function)

Member of the <u>convert</u> class.

### Syntax

SysOutDis = d2d(SysInDis, Ts)

#### Description

Resamples a discrete system model. Resampling involves converting a discrete model to another discrete model with a different sampling time.

Examples

## Inputs

Name	Description
SysInDis	Specifies a discrete model in transfer function, zero-pole-gain, or state-space form.
Ts	Specifies the sampling time this function uses to resample the discrete model <b>SysInDis</b> . The default value is 1. <b>Ts</b> is a real scalar.

## Outputs

Name	Description
SysOutDis	Returns the discrete model <b>SysInDis</b> with new sampling time <b>Ts</b> .

#### Examples

A = [0.5, 0.25; 0, 0.1]B = [1, 0]'; C = [-1, 0] SysInDis = ss(A, B, C, 0, 0.1) SysOutDis = d2d(SysInDis, 0.2)

## **Related Topics**

<u>c2d</u> <u>d2c</u>

# delay2z (Control Design and Simulation Module, MathScript Function)

Member of the <u>convert</u> class.

#### Syntax

SysDel = delay2z(SysDis)

#### Description

Incorporates delays into discrete system models by adding poles at the origin. You must specify the delay using the <u>set</u> function.

#### Examples

## Inputs

Name	Description
SysDis	Specifies a discrete linear time-invariant (LTI) model in transfer
	function, zero-pole-gain, or state-space form.

## Outputs

Name	Description
SysDel	Returns a discrete LTI model with the total delay converted into poles at the origin. <b>SysOutDis</b> is in the same form as the <b>SysDis</b> model.

#### Examples

SysDis = ss(-1, 1, 2, 0, 0.1)SysDis = set(SysDis, 'inputdelay', 4) SysDel = delay2z(SysDis)

## **Related Topics**

hasdelay pade totaldelay get set

## pade (Control Design and Simulation Module, MathScript Function)

Member of the <u>convert</u> class.

#### Syntax

SysDel = pade(SysCon, order) SysDel = pade(delay, order) [num, den] = pade(delay, order) [A, B, C, D] = pade(delay, order)

#### Description

Incorporates time delays into a system model using the Pade approximation method, which converts all residuals. You must specify the delay using the <u>set</u> function. You also can use this function to calculate coefficients of numerator and denominator polynomial functions with a specified delay.

**Examples** 

## Inputs

Name	Description
SysCon	Specifies a continuous linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
delay	Specifies the delay time, in seconds, to incorporate into the numerator and denominator polynomial functions.
order	Specifies the order of the Pade approximation polynomial functions. A higher order results in a more accurate approximation of the delay but also increases the order of the resulting model. A large order can make the model too complex to be useful. <b>order</b> is an integer scalar.

## Outputs

Name	Description
SysDel	Returns an LTI model in transfer function, zero-pole-gain, or state-space form, with delay incorporated. If you specified <b>SysCon</b> as the first input to this function, <b>SysDel</b> is <b>SysCon</b> with delay incorporated. If you specified <b>delay</b> as the first input, <b>SysDel</b> is a system model of pure delay.
num	Returns the coefficients of a numerator polynomial function with delay incorporated.
den	Returns the coefficients of a denominator polynomial function with delay incorporated.
A	Returns an $n \ge n$ state matrix with delay incorporated, where $n$ is the number of states. <b>A</b> is a real matrix.
В	Returns the $n \ge m$ input matrix with delay incorporated, where $m$ is the number of inputs. <b>B</b> is a real matrix.
С	Returns the $r \ge n$ output matrix with delay incorporated, where $r$ is the number of outputs. <b>C</b> is a real matrix.
D	Returns the $r \ge m$ direct transmission matrix with delay incorporated. <b>D</b> is a real matrix.

#### Examples

SysCon = zpk(1, 3.2, 6)SysCon = set(SysCon, 'inputdelay', 6, 'outputdelay', 1.1) SysDel = pade(SysCon, 2)

delay = 1.2 order = 3 [num, den] = pade(delay, order)
#### **Related Topics**

c2d delay2z get set hasdelay

# ss2ss (Control Design and Simulation Module, MathScript Function)

Member of the <u>convert</u> class.

#### Syntax

SysSimSS = ss2ss(SysInSS, T)

#### Description

Applies a transformation using a transformation matrix you specify. <u>Examples</u>

## Inputs

Name	Description		
SysInSS	Specifies a linear time-invariant (LTI) system model in state- space form.		
Т	Specifies the transformation matrix this function uses to transform the <b>SysInSS</b> model. <b>T</b> must be an $n \ge n$ real matrix, where $n$ is the number of model states.		

#### Outputs

Name	Description
SysSimSS	Returns the LTI state-space model resulting from the transformation.

#### Examples

SysInSS = ss([1, 1; -1, 2], [1, 2]', [2, 1], 0)T = ones(2, 2) - eye(2) SysSimSS = ss2ss(SysInSS, T)

## **Related Topics**

<u>balreal</u> <u>canon</u>

# dynchar (Control Design and Simulation Module, MathScript Class)

Use members of the dynchar class to calculate properties related to the dynamics of a given system model.

Function	Description			
<u>covar</u>	Returns the covariance of the outputs and/or states of a system model			
<u>damp</u>	Calculates the damping ratios and natural frequencies of system model poles			
<u>dcgain</u>	Calculates the DC gain of a system model			
<u>rlocfind</u>	Creates an interactive Evans, or root locus, plot that you use to vary the feedback gain to place poles in specific locations			
<u>norm</u>	Calculates the H-infinity or the H-2 norm of a system model			
<u>pole</u>	Returns the locations of system model poles			
<u>pzmap</u>	Plots the poles and zeros of a system model			
<u>iopzmap</u>	Plots the poles of a control system			
<u>rlocus</u>	Creates an Evans, or root locus, plot of a system model			
<u>totaldelay</u>	Returns the total delay in a system model			
zero	Returns the locations of system model zeros			

# covar (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

#### Syntax

R = covar(SysIn, Q) [R, S] = covar(SysIn, Q)

#### Description

Returns the covariance of the outputs when Gaussian white noise excites the **SysIn** system model. This function also returns the covariance of the states if the **SysIn** model is in state-space form.

**Examples** 

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
Q	Specifies the Gaussian white noise intensity matrix if the input model is continuous. If the input model is discrete, $\mathbf{Q}$ specifies the Gaussian white noise covariance matrix. $\mathbf{Q}$ must be a real matrix that is symmetric and positive semi-definite.

# Outputs

Name	Description
R	Returns the covariance of the outputs when Gaussian white noise excites the <b>SysIn</b> model. <b>R</b> is a real matrix.
S	Returns the covariance of the states when Gaussian white noise excites the <b>SysIn</b> model. This function returns <b>S</b> only if <b>SysIn</b> is in state-space form. <b>S</b> is a real matrix.

#### Examples

SysIn = ss([-1, 0; 1, -2], [1, 0]', [0, 1], 0)Q = 0.1 [R, S] = covar(SysIn, Q)

## **Related Topics**

<u>dlyap</u> lyap

# damp (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

#### Syntax

[dr, wn, p] = damp(SysIn)

#### Description

Returns the damping ratios and natural frequencies of the poles of a system model.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

# Outputs

Name	Description			
dr	Returns the damping ratio of each pole of the <b>Sysin</b> model. The <i>n</i> -th element of this vector corresponds to the <i>n</i> -th element of the <b>wn</b> and <b>p</b> vectors. <b>dr</b> is a real vector.			
wn	Returns the natural frequency of each pole of the <b>Sysin</b> model. The <i>n</i> -th element of this vector corresponds to the <i>n</i> -th element of the <b>dr</b> and <b>p</b> vectors. <b>wn</b> is a real vector.			
р	Returns the poles of the <b>SysIn</b> model. The <i>n</i> -th element of this vector corresponds to the <i>n</i> -th element of the <b>dr</b> and <b>wn</b> vectors. <b>p</b> is a complex vector.			

#### Examples

SysIn = tf([1], [1, 2, 1])[dr, wn, p] = damp(SysIn)

## **Related Topics**

dsort esort pole pzmap zero

# dcgain (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

#### Syntax

DC = dcgain(SysIn)

#### Description

Calculates the DC gain of the **SysIn** system model. The DC gain is the ratio of the outputs to the inputs of a model after all transients decay.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

# Outputs

Name	Description
DC	Returns the DC gain of the <b>SysIn</b> model. <b>DC</b> is a two- dimensional array where the <i>ij</i> -th element corresponds to the DC gain of the model due to the <i>i</i> -th output and <i>j</i> -th input. <b>DC</b> is a real matrix.

#### Examples

SysIn = tf([1], [1, 1])DC = dcgain(SysIn)

## **Related Topics**

<u>norm</u>

# iopzmap (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

#### Syntax

iopzmap(SysIn)
iopzmap(SysIn, attributes)

#### Description

Creates a plot of the poles and zeros of each input-output pair of a system model.

Examples

## Inputs

Name	Description				
SysIn	Specifies a linear time-invariant (LTI) model in transfer				
	runction, zero-pole-gain, or state-space form.				
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies that the plot is blue, marks points with circles, and uses solid lines. <b>attributes</b> is a string that can take a combination of the following values:				
	'b'	Colors the plot blue.			
c' Colors the plot c		Colors the plot cyan.			
	'g'	Colors the plot green.			
	'k'	Colors the plot black.			
	'm'	Colors the plot magenta.			
	'r'	Colors the plot red.			
	'y'	Colors the plot yellow.			
	'.'	Marks points with dots.			
	'0'	Marks points with circles.			
	'X'	Marks points with crosses.			
	'+'	Marks points with plus signs.			
	'*'	Marks points with asterisks.			
	'-'	Uses solid lines.			
	':' Uses dotted lines.				
	'' Uses dashed and dotted lines.				
	'' Uses dashed lines.				

#### Examples

z1 = [1] z2 = [2] p1 = [-1, -2] p2 = [-3, -4] K = [1 2] SysIn = zpk(z1, z2, p1, p2, K) iopzmap(SysIn, 'y.')

## **Related Topics**

acker place pole pzmap rlocus zero
# norm (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

#### Syntax

normval = norm(SysIn) normval = norm(SysIn, type) [normval, w] = norm(SysIn) [normval, w] = norm(SysIn, type)

#### Description

Calculates the H-infinity or the H-2 norm of a system model. Examples

## Inputs

Name	Description	
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.	
type	Specifies the type of norm you want to calculate. <b>type</b> is an integer that can take the following values:	
	2	(Default) Specifies that this function calculates the H-2 norm, or   H   2, of the <b>SysIn</b> model.
	inf	Specifies that this function calculates the H-infinity norm, or   H  , of the <b>SysIn</b> model.

## Outputs

Name	Description
normval	Returns the H-infinity or H-2 norm of the <b>SysIn</b> model, depending on the value you specify for the <b>type</b> input. The 2- norm is infinite for unstable models and for state-space models whose <b>D</b> matrix does not equal zero. <b>normval</b> is a real scalar.
w	Returns the frequency, in radians per second, at which this function evaluates the H-infinity norm. This output is undefined if you calculate the H-2 norm of the model. <b>freq</b> is a real scalar.

## Examples

SysIn = tf([1], [1, 1]) [normval, w] = norm(SysIn, inf)

## **Related Topics**

pole damp dcgain

# pole (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

## Syntax

q = pole(SysIn)

#### Description

Returns the locations of the closed-loop poles of a system model. Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

## Outputs

Name	Description	
q	Returns the closed-loop pole locations of the <b>SysIn</b> model. <b>q</b> is a complex vector.	

#### Examples

SysIn = ss([-1, 0; 1, -2], [1, 0]', [0, 1],0)q = pole(SysIn)

## **Related Topics**

damp place pzmap zero

# pzmap (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

## Syntax

pzmap(SysIn, attributes)
[q, z] = pzmap(SysIn)

#### Description

Creates a plot of system model poles and zeros. You also can use this function to return the locations of the poles and zeros of a model. If you do not specify an output, this function creates a plot.

Examples

## Inputs

Name	Des	scription	
SysIn	Spe	ecifies a linear time-invariant (LT	T) model in transfer
	Tune	ction, zero-pole-gain, or state-sp	Dace Iorm.
attributes	Spe colo that line follo	ecifies valid plot attributes. Orde or, point-style, and line-style. Fo : the plot is blue, marks points w s. <b>attributes</b> is a string that can owing values:	r the plot attributes by r example, 'bo-' specifies <i>i</i> th circles, and uses solid take a combination of the
	'b'	Colors the plot blue.	
	'C'	Colors the plot cyan.	
	'g'	Colors the plot green.	
	'k'	Colors the plot black.	
	'm'	Colors the plot magenta.	
	'r'	Colors the plot red.	
	'y'	Colors the plot yellow.	
	'.'	Marks points with dots.	
	'0'	Marks points with circles.	
	'X'	Marks points with crosses.	
	'+'	Marks points with plus signs.	
	'*'	Marks points with asterisks.	
	'-'	Uses solid lines.	
	י.י י	Uses dotted lines.	
	''	Uses dashed and dotted lines.	
	''	Uses dashed lines.	

## Outputs

Name	Description
q	Returns the locations of the poles of the <b>SysIn</b> model. <b>q</b> is a complex vector.
Z	Returns the locations of the zeros of the <b>SysIn</b> model. <b>z</b> is a complex vector.

#### Examples

SysIn = zpk([1], [-1, -2], 0.5) pzmap(SysIn)

## **Related Topics**

acker place pole rlocus zero

# rlocfind (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

#### Syntax

rlocfind(SysInSISO)
K = rlocfind(SysInSISO)
[K, q] = rlocfind(SysInSISO)
rlocfind(SysInSISO, p)
K = rlocfind(SysInSISO, p)
[K, q] = rlocfind(SysInSISO, p)

#### Description

Creates an interactive Evans, or root-locus, plot of closed-loop poles of a single-input single-output (SISO) system model. You can specify pole locations and use the interactive plot to adjust the feedback gain to place model poles in these locations.

**Examples** 

## Inputs

Name	Description
SysInSISO	Specifies a SISO linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
р	Specifies the locations where you want to place the closed-loop poles. <b>p</b> is a complex vector.

## Outputs

Name	Description
К	Returns the feedback gain value that places the closed-loop poles in the locations the <b>q</b> output returns.
q	Returns the locations of the closed-loop poles of the <b>SysInSISO</b> model when the feedback gain is <b>K</b> . <b>q</b> is a complex vector.

#### Examples

num = [1]den = [1, 5, 6] SysInSISO = tf(num, den) p = [-2.5 - 3.6322i, -2.5 + 3.6322i] [K, q] = rlocfind(SysInSISO, p)

## **Related Topics**

<u>pole</u> <u>rlocus</u>

# rlocus (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

#### Syntax

rlocus(SysInSISO) [roots, gains] = rlocus(SysInSISO)

#### Description

Creates an Evans, or root-locus, plot of closed-loop poles of a singleinput single-output (SISO) system model as the feedback gain increases from zero to infinity.

Examples

## Inputs

Name	Description
SysInSISO	Specifies a linear time-invariant (LTI) SISO model in transfer function, zero-pole-gain, or state-space form.

## Outputs

Name	Description
roots	Returns the locations of the closed-loop roots of the <b>SysIn</b> model. <b>roots</b> is a complex matrix.
gains	Returns the gain values this function uses to create the root locus plot. <b>gains</b> is a real vector.

#### Examples

num = [1]den = [1, 5, 6] SysInSISO = tf(num, den) rlocus(SysInSISO)

## **Related Topics**

<u>issiso</u> pole rlocfind

# totaldelay (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.
#### Syntax

totdel = totaldelay(SysIn)

#### Description

Returns the total delay present in a system model. The total delay includes all input, output, and transport delays.

Examples

## Inputs

Name	Description	
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,	
	zero-pole-gain, or state-space form.	

# Outputs

Name	Description
totdel	Returns the total delay present in the model. For continuous models, the delay is in seconds. For discrete models, the delay is an integer multiple of the sampling time. <b>totdel</b> is a real matrix.

#### Examples

Sys = ss(-1, 1, 2, 0, 0.1)SysIn = set(Sys, 'inputdelay', 4, 'outputdelay', 2) totdel = totaldelay(SysIn)

## **Related Topics**

delay2z pade get hasdelay set

# zero (Control Design and Simulation Module, MathScript Function)

Member of the <u>dynchar</u> class.

#### Syntax

[z, K] = zero(SysIn)

#### Description

Calculates the zeros and gain of a system model.

Examples

## Inputs

Name	Description	
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,	
	zero-pole-gain, or state-space form.	

## Outputs

Name	Description
z	Returns the zeros of the <b>SysIn</b> model. <b>z</b> is a complex vector.
к	Returns the steady-state gain matrix of the <b>SysIn</b> model. <b>K</b> is a complex vector.

#### Examples

SysIn = zpk([1], [1, 2, 1], 2)z = zero(SysIn)

#### **Related Topics**

pole pzmap <u>zpk</u>

# frqrsp (Control Design and Simulation Module, MathScript Class)

Use members of the frqrsp class to analyze a system model in the frequency domain.

Function	Description
<u>allmargin</u>	Calculates all gain and phase margins of a system model
<u>bandwidth</u>	Calculates the bandwidth of a system model
<u>bode</u>	Creates the Bode magnitude and phase plots of a system model
<u>bodemag</u>	Creates a Bode magnitude plot of a system model
<u>evalfr</u>	Calculates the magnitude and phase of a system model at a frequency you specify
<u>margin</u>	Calculates the smallest gain and phase margins of a system model
<u>nyquist</u>	Creates a Nyquist plot of a system model
<u>nichols</u>	Creates a Nichols plot of a system model
<u>sigma</u>	Calculates the singular values of the frequency response of a system model

# allmargin (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

gmf = allmargin(SysInSISO)
[gmf, gm] = allmargin(SysInSISO)
[gmf, gm, pmf] = allmargin(SysInSISO)
[gmf, gm, pmf, pm] = allmargin(SysInSISO)

#### Description

Calculates all gain and phase margins of a single-input single-output (SISO) system model. The gain margins indicate where the frequency response crosses at 0 decibels. The phase margins indicate where the frequency response crosses -180 degrees. Use the <u>margin</u> function to return only the smallest gain and phase margins of a SISO model.

**Examples** 

## Inputs

Name	Description
SysInSISO	Specifies a linear time-invariant (LTI) SISO model in transfer function, zero-pole-gain, or state-space form.

## Outputs

Name	Description
gmf	Returns the gain margin frequencies, in radians/second, of the <b>SysInSISO</b> model. A gain margin frequency indicates where the model phase crosses -180 degrees. The <i>i</i> -th element of this vector corresponds to the <i>i</i> -th element of the <b>gm</b> vector. <b>gmf</b> is a real vector.
gm	Returns the gain margins of the <b>SysInSISO</b> model. <b>gm</b> is a real vector.
pmf	Returns the phase margin frequencies, in radians/second, of the <b>SysInSISO</b> model. A phase margin frequency indicates where the model magnitude crosses 0 decibels. The <i>i</i> -th element of this vector corresponds to the <i>i</i> -th element of the <b>pm</b> vector. <b>pmf</b> is a real vector.
pm	Returns the phase margins of <b>SysInSISO</b> . <b>pm</b> is a real vector.

#### Examples

SysInSISO = tf([1, 1], [1, -1, 3])[gmf, gm, pmf, pm] = allmargin(SysInSISO)

## **Related Topics**

<u>margin</u> <u>bode</u>

# bandwidth (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

band = bandwidth(SysInSISO)
band = bandwidth(SysInSISO, magdrop)

#### Description

Calculates the bandwidth of a single-input single-output (SISO) system model. The bandwidth is the frequency, relative to the DC gain, at which the frequency response magnitude of the model drops below a threshold value you specify.

Examples

#### Inputs

Name	Description
SysInSISO	Specifies a linear time-invariant (LTI) SISO model. <b>SysInSISO</b> is a model in transfer function, zero-pole-gain, or state-space form.
magdrop	Specifies the magnitude drop threshold, in decibels (dB). The default value is -3 dB. <b>magdrop</b> is a real scalar.

## Outputs

Name
band

#### Examples

SysInSISO = tf([1], [1, 1])magdrop = -2 band\_2DB = bandwidth(SysInSISO, magdrop)

## **Related Topics**

<u>dcgain</u>

# bode (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

bode(SysIn) bode(SysIn, attributes) bode(SysIn, attributes, [wmin wmax]) bode(SysIn, attributes, wlist) mag = bode(SysIn) mag = bode(SysIn, [wmin wmax]) mag = bode(SysIn, wlist) [mag, phase] = bode(SysIn) [mag, phase] = bode(SysIn, [wmin wmax]) [mag, phase, wout] = bode(SysIn, wlist) [mag, phase, wout] = bode(SysIn, wlist)

#### Description

Creates the Bode magnitude and Bode phase plots of a system model. You also can use this function to return the magnitude and phase values of a model at frequencies you specify. If you do not specify an output, this function creates a plot.

**Examples** 

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) system model in ransfer function, zero-pole-gain, or state-space form.
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies hat the plot is blue, marks points with circles, and uses solid nes. <b>attributes</b> is a string that can take a combination of the ollowing values:
	b' Colors the plot blue.
	c' Colors the plot cyan.
	g' Colors the plot green.
	k' Colors the plot black.
	m' Colors the plot magenta.
	r' Colors the plot red.
	y' Colors the plot yellow.
	.' Marks points with dots.
	o' Marks points with circles.
	x' Marks points with crosses.
	+' Marks points with plus signs.
	*' Marks points with asterisks.
	-' Uses solid lines.
	:' Uses dotted lines.
	' Uses dashed and dotted lines.
	' Uses dashed lines.
wmin	Specifies the initial frequency, in radians/second, this function uses to calculate the frequency response of the <b>SysIn</b> nodel. The default value is 0.001. You also can use the <b>wlist</b> nput to specify exact frequencies instead of a range. <b>wmin</b> is a real scalar.

wmax	Specifies the final frequency, in radians/second, this function uses to calculate the frequency response of the <b>SysIn</b> model. The default value is 1000. You also can use the <b>wlist</b> input to specify exact frequencies instead of a range. <b>wmax</b> is a real scalar.
wlist	Specifies the frequency values, in radians/second, at which this function calculates the singular values. If you specify only two elements for this input, this function treats those elements as <b>wmin</b> and <b>wmax</b> . Therefore, you must specify at least three elements for the <b>wlist</b> input. <b>wlist</b> is a real vector consisting of three or more elements.

# Outputs

Name	Description
mag	Returns the magnitude(s) of the <b>SysIn</b> model. The <i>n</i> -th element of this parameter corresponds to the <i>n</i> -th element of the <b>phase</b> and <b>wout</b> parameters. <b>mag</b> is a real matrix.
phase	Returns the phase(s), in degrees, of the SysIn model. phase is a real matrix.
wout	Returns the frequencies, in radians/second, at which this function calculates the magnitude and phase. <b>wout</b> is a real vector.

#### Examples

z = 1 p = [-1, -2] k = 0.5 SysIn = zpk(z, p, k) bode(SysIn, 'r')

## **Related Topics**

bodemag rlocus nichols nyquist
# bodemag (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

bodemag(SysIn, attributes) bodemag(SysIn, attributes, [wmin wmax]) bodemag(SysIn, attributes, wlist) [mag, wout] = bodemag(SysIn) [mag, wout] = bodemag(SysIn, [wmin wmax]) [mag, wout] = bodemag(SysIn, wlist)

#### Description

Creates the Bode magnitude plot of a system model. If you do not specify an output, this function creates a plot.

Examples

# Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) system model in ransfer function, zero-pole-gain, or state-space form.
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies hat the plot is blue, marks points with circles, and uses solid nes. <b>attributes</b> is a string that can take a combination of the ollowing values:
	b' Colors the plot blue.
	c' Colors the plot cyan.
	g' Colors the plot green.
	k' Colors the plot black.
	m' Colors the plot magenta.
	r' Colors the plot red.
	y' Colors the plot yellow.
	.' Marks points with dots.
	o' Marks points with circles.
	x' Marks points with crosses.
	+' Marks points with plus signs.
	*' Marks points with asterisks.
	-' Uses solid lines.
	:' Uses dotted lines.
	' Uses dashed and dotted lines.
	' Uses dashed lines.
wmin	Specifies the initial frequency, in radians/second, this function uses to calculate the frequency response of the <b>SysIn</b> nodel. The default value is 0.001. You also can use the <b>wlist</b> nput to specify exact frequencies instead of a range. <b>wmin</b> is a real scalar.

wmax	Specifies the final frequency, in radians/second, this function uses to calculate the frequency response of the <b>SysIn</b> model. The default value is 1000. You also can use the <b>wlist</b> input to specify exact frequencies instead of a range. <b>wmax</b> is a real scalar.
wlist	Specifies the frequency values, in radians/second, at which this function calculates the frequency response of the <b>SysIn</b> model. If you specify only two elements for this input, this function treats those elements as <b>wmin</b> and <b>wmax</b> . Therefore, you must specify at least three elements for the <b>wlist</b> input. <b>wlist</b> is a real vector consisting of three or more elements.

# Outputs

Name	Description
mag	Returns the magnitude(s) of the <b>SysIn</b> model. <b>mag</b> is a real matrix.
wout	Returns the frequencies at which this function calculates the magnitude(s). <b>w</b> is a real vector.

## Examples

z = 1 p = [-1, -2] k = 0.5 SysIn = zpk(z, p, k) bodemag(SysIn, 'g')

## **Related Topics**

bode rlocus nichols nyquist

# evalfr (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

gw = evalfr(SysIn) gw = evalfr(SysIn, w)

#### Description

Calculates the magnitude and phase of a system model at a frequency you specify.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
w	Specifies the frequency, in radians/second, at which this function evaluates the magnitude and phase of the <b>SysIn</b> model. The default value is 0. <b>w</b> is a complex scalar.

## Outputs

Name	Description
gw	Returns the magnitude and phase of the <b>SysIn</b> model at frequency <b>w</b> . <b>gw</b> is a complex matrix.

## Examples

SysIn = ss(-1, 1, 1, 0)gw = evalfr(SysIn, 2i)

## **Related Topics**

<u>bode</u> <u>allmargin</u> <u>bandwidth</u>

# margin (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

margin(SysInSISO)
gm = margin(SysInSISO)
[gm, gmf] = margin(SysInSISO)
[gm, gmf, pm] = margin(SysInSISO)
[gm, gmf, pm, pmf] = margin(SysInSISO)
[gm] = margin(mag, pha, w)
[gm, gmf, ] = margin(mag, pha, w)
[gm, gmf, pm] = margin(mag, pha, w)
[gm, gmf, pm, pmf] = margin(mag, pha, w)

#### Description

Calculates and/or plots the smallest gain and phase margins of a singleinput single-output (SISO) system model. The gain margin indicates where the frequency response crosses at 0 decibels. The phase margin indicates where the frequency response crosses -180 degrees. Use the <u>allmargin</u> function to return all gain and phase margins of a SISO model.

**Examples** 

## Inputs

Name	Description
SysInSISO	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
mag	Specifies the magnitude(s) of the model whose margins you want to calculate. <b>mag</b> is a real vector.
pha	Specifies the phase(s), in radians, of the model whose margins you want to calculate. <b>mag</b> is a real vector.
W	Specifies the frequency, in radians/second, of the model whose margins you want to calculate. <b>w</b> is a real scalar.

# Outputs

Name	Description
gm	Returns the smallest gain margin of the <b>SysInSISO</b> model. <b>gm</b> is a real scalar.
gmf	Returns the gain margin frequency, in radians/second, that corresponds to the <b>gm</b> output. A gain margin frequency indicates where the model phase crosses -180 degrees. <b>gmf</b> is a real scalar.
pm	Returns the smallest phase margin of the <b>SysInSISO</b> model. <b>pm</b> is a real scalar.
pmf	Returns the phase margin frequency, in radians/second, that corresponds to the <b>pm</b> output. A phase margin frequency indicates where the model magnitude crosses 0 decibels. <b>pmf</b> is a real scalar.

## Examples

num = [1]den = [1, 5, 6] SysInSISO = tf(num, den) margin(SysInSISO)

## **Related Topics**

<u>allmargin</u> <u>bode</u>

# nichols (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

nichols(SysIn) nichols(SysIn, attributes) nichols(SysIn, attributes, [wmin wmax]) nichols(SysIn, attributes, wlist) [mag, phase, w] = nichols(SysIn) [mag, phase, w] = nichols(SysIn, [wmin wmax]) [mag, phase, w] = nichols(SysIn, wlist)

#### Description

Creates the Nichols plot of a system model. A Nichols plot shows the magnitude of the frequency response against the phase. You also can use this function to return the magnitudes and phases of a model. If you do not specify an output, this function creates a plot.

**Examples** 

# Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies that the plot is blue, marks points with circles, and uses solid lines. <b>attributes</b> is a string that can take a combination of the following values:
	'b'Colors the plot blue.'c'Colors the plot cyan.'g'Colors the plot green.'k'Colors the plot black.'m'Colors the plot magenta.'r'Colors the plot red.'y'Colors the plot yellow.'.'Marks points with dots.'o'Marks points with circles.'x'Marks points with crosses.'+'Marks points with plus signs.'*'Uses solid lines.'.'Uses dotted lines.''Uses dashed and dotted lines.''Uses dashed lines.
wmin	Specifies the initial frequency, in radians/second, this function uses to calculate the frequency response of the <b>SysIn</b> model. The default value is 1E-8. You also can use the <b>wlist</b> input to specify exact frequencies instead of a range. <b>wmin</b> is a real scalar.

wmax	Specifies the final frequency, in radians/second, this function uses to calculate the frequency response of the <b>SysIn</b> model. The default value is 1000. You also can use the <b>wlist</b> input to specify exact frequencies instead of a range. <b>wmax</b> is a real scalar.
wlist	Specifies the frequency values, in radians/second, at which this function calculates the singular values. If you specify only two elements for this input, this function treats those elements as <b>wmin</b> and <b>wmax</b> . Therefore, you must specify at least three elements for the <b>wlist</b> input. <b>wlist</b> is a real vector consisting of three or more elements.

# Outputs

Name	Description
mag	Returns the magnitude(s) of the <b>SysIn</b> model. The <i>n</i> -th element of this parameter corresponds to the <i>n</i> -th element of the <b>phase</b> and <b>wout</b> parameters. <b>mag</b> is a real matrix.
phase	Returns the phase(s), in degrees, of the SysIn model. phase is a real matrix.
wout	Returns the frequencies, in radians/second, at which this function calculates the magnitude and phase. <b>wout</b> is a real vector.

## Examples

z = 1 p = [-1, -2] k = 0.5 SysIn = zpk(z, p, k) nichols(SysIn)

## **Related Topics**

<u>nyquist</u> <u>bode</u>

# nyquist (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

nyquist(SysIn, attributes)
nyquist(SysIn, attributes, [wmin wmax])
nyquist(SysIn, attributes, wlist)
[re, im, wout] = nyquist(SysIn)
[re, im, wout] = nyquist(SysIn, [wmin wmax])
[re, im, wout] = nyquist(SysIn, wlist)

#### Description

Creates the Nyquist plot of a system model. A Nyquist plot shows the imaginary part of the frequency response against the real part of the frequency response. You also can use this function to calculate the real and imaginary parts of the frequency response. If you do not specify an output, this function creates a plot.

**Examples** 

# Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies that the plot is blue, marks points with circles, and uses solid lines. <b>attributes</b> is a string that can take a combination of the following values:
	'b'Colors the plot blue.'c'Colors the plot cyan.'g'Colors the plot green.'k'Colors the plot black.'m'Colors the plot magenta.'r'Colors the plot red.'y'Colors the plot yellow.'.'Marks points with dots.'o'Marks points with circles.'x'Marks points with crosses.'+'Marks points with plus signs.'*'Uses solid lines.'.'Uses dotted lines.''Uses dashed and dotted lines.''Uses dashed lines.
wmin	Specifies the initial frequency, in radians/second, this function uses to calculate the frequency response of the <b>SysIn</b> model. The default value is 1E-8. You also can use the <b>wlist</b> input to specify exact frequencies instead of a range. <b>wmin</b> is a real scalar.

wmax	Specifies the final frequency, in radians/second, this function uses to calculate the frequency response of the <b>SysIn</b> model. The default value is 1000. You also can use the <b>wlist</b> input to specify exact frequencies instead of a range. <b>wmax</b> is a real scalar.
wlist	Specifies the frequency values, in radians/second, at which this function calculates the frequency response. If you specify only two elements for this input, this function treats those elements as <b>wmin</b> and <b>wmax</b> . Therefore, you must specify at least three elements for the <b>wlist</b> input. <b>wlist</b> is a real vector consisting of three or more elements.

# Outputs

Name	Description
re	Returns the real parts of the frequency response. <b>re</b> is a real matrix.
im	Returns the imaginary parts of the frequency response. <b>im</b> is a real matrix.
wout	Returns the frequencies, in radians/second, at which this function calculates the real and imaginary parts of the frequency response. $w$ is a real vector.
## Examples

z = 1 p = [-1, -2] k = 0.5 SysIn = zpk(z, p, k) nyquist(SysIn)

## **Related Topics**

<u>bode</u> <u>nichols</u> <u>allmargin</u>

# sigma (Control Design and Simulation Module, MathScript Function)

Member of the <u>frqrsp</u> class.

#### Syntax

sigma(SysIn, attributes) sigma(SysIn, attributes, [wmin wmax]) sigma(SysIn, attributes, [wmin wmax], type) sigma(SysIn, attributes, wlist) sigma(SysIn, attributes, wlist, type) [magSV, wout] = sigma(SysIn) [magSV, wout] = sigma(SysIn, [wmin wmax]) [magSV, wout] = sigma(SysIn, [wmin wmax], type) [magSV, wout] = sigma(SysIn, attributes, wlist) [magSV, wout] = sigma(SysIn, attributes, wlist, type)

#### Description

Creates a plot of the singular values of the frequency response of a system model. You also can use this function to calculate the singular values. This function converts transfer function and zero-pole-gain models into state-space models before calculating the singular values. If you do not specify an output, this function creates a plot.

#### **Examples**

# Inputs

Name	Description		
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.		
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies that the plot is blue, marks points with circles, and uses solid lines. <b>attributes</b> is a string that can take a combination of the following values:		
	'b' Colors the plot blue.		
	'c' Colors the plot cyan.		
	'g' Colors the plot green.		
	'k' Colors the plot black.		
	'm' Colors the plot magenta.		
	'r' Colors the plot red.		
	'y' Colors the plot yellow.		
	'.' Marks points with dots.		
	'o' Marks points with circles.		
	'x' Marks points with crosses.		
	'+' Marks points with plus signs.		
	'*' Marks points with asterisks.		
	'-' Uses solid lines.		
	':' Uses dotted lines.		
	'' Uses dashed and dotted lines.		
	'' Uses dashed lines.		
wmin	Specifies the initial frequency, in radians/second, this function uses to calculate the singular values of the <b>SysIn</b> model. The default value is 1E-8. You also can use the <b>wlist</b> input to specify exact frequencies instead of a range. <b>wmin</b> is a real scalar.		

wmax	Specifies the final frequency, in radians/second, this function uses to calculate the singular values of the <b>SysIn</b> model. The default value is 1000. You also can use the <b>wlist</b> input to specify exact frequencies instead of a range. <b>wmax</b> is a real scalar.		
wlist	Specifies the frequency values, in radians/second, at which this function calculates the singular values. If you specify only two elements for this input, this function treats those elements as <b>wmin</b> and <b>wmax</b> . Therefore, you must specify at least three elements for the <b>wlist</b> input. <b>wlist</b> is a real vector consisting of three or more elements.		
type	Specifies the type of singular value decomposition this function performs. <b>type</b> is an integer that takes the following values:		
	) (Default) Specifies that this function calculates the singular value decomposition of <i>H</i> , where <i>H</i> is the <b>SysIn</b> model.		
	Specifies that this function calculates the singular value decomposition of inv( <i>H</i> ).		
	2 Specifies that this function calculates the singular value decomposition of $I + H$ , where $I$ is the identity matrix.		
	Specifies that this function calculates the singular value decomposition of $I + inv(H)$ .		

# Outputs

Name	Description
magSV	Returns the singular value magnitude at different frequencies. The <i>n</i> -th column of this parameter corresponds to the <i>n</i> -th row of the <b>wout</b> parameter. <b>magSV</b> is a real matrix.
wout	Returns the frequencies, in radians/second, at which this function calculates the singular values. The <i>n</i> -th row of this parameter corresponds to the <i>n</i> -th column of the <b>magSV</b> parameter. <b>wout</b> is a real vector.

#### Examples

A = [-0.2, 0;0, -0.4] B = [1, 0;0, 1] C = [1, 0;0, 1] D = [0, 0;0, 0] SysIn = ss(A, B, C, D) sigma(SysIn, 3)

## **Related Topics**

<u>bode</u> nichols nyquist

# info (Control Design and Simulation Module, MathScript Class)

Use members of the info class to obtain information about a system model.

Function	Description		
<u>areequal</u>	Determines whether two models are identical		
<u>aresimilar</u>	Determines whether two models are similar		
<u>get</u>	Returns information about a system model		
<u>hasdelay</u>	Determines whether a system model has delays		
<u>isct</u>	Determines whether a system model is continuous		
<u>isctrb</u>	Determines whether a system model is controllable or stabilizable		
<u>isdt</u>	Determines whether a system model is discrete		
<u>isempty</u>	Determines whether a system model is empty		
<u>isobsv</u>	Determines whether a system model is observable or detectable		
<u>isproper</u>	Determines whether a system model is proper		
<u>issiso</u>	Determines whether a system model is single-input single- output (SISO)		
<u>isstable</u>	Determines whether a system model is stable		
<u>ndims</u>	Returns the number of dimensions of a system model		
<u>set</u>	Sets information to a system model		
<u>size</u>	Returns the size of a system model		
<u>ssdata</u>	Returns information about a state-space system model		
<u>tfdata</u>	Returns information about a transfer function system model		
<u>zpkdata</u>	Returns information about a zero-pole-gain system model		

# areequal (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

## Syntax

equal = areequal(SysIn\_1, SysIn\_2, tol)

## Description

Determines whether two models are identical within a specified tolerance. Examples

## Inputs

Name	Description	
SysIn_1	<b>1</b> Specifies a linear time-invariant (LTI) system in transfer function, zero-pole-gain, or state-space form.	
SysIn_2	2 Specifies an LTI system. This model must be in the same form as the SysIn_1 model.	
tol	Specifies the tolerance. The default value of <b>tol</b> is 0.00000001. <b>tol</b> is a real scalar.	

## Outputs

Name	Description
equal	Returns 1 if the two models are equal. This function returns 0 if
	the two models are not equal. <b>Equal</b> is a boolean.

#### Examples

SysIn\_1 = ss(-1, 1, 2, 0);SysIn\_2 = (1+1.e-10)\*SysIn\_1; areequal(SysIn\_1, SysIn\_2, 1.e-9);

## **Related Topics**

<u>aresimilar</u>

# aresimilar (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

## Syntax

similar = aresimilar(SysIn\_1, SysIn\_2, tol)
[similar, T] = aresimilar(SysIn\_1, SysIn\_2, tol)

#### Description

Determines whether two models are similar. Two state-space models are similar if you can use a similarity transformation matrix to transform the states of one model into the states of the other model linearly and within a certain tolerance. Two transfer function or zero-pole-gain models are similar if, after cancelling pole-zero pairs due to a tolerance, the models are identical.

**Examples** 

## Inputs

Name	Description
SysIn_1	Specifies a linear time-invariant (LTI) system in transfer function, zero-pole-gain, or state-space form.
SysIn_2	Specifies an LTI system. This model must be in the same form as the <b>SysIn_1</b> model.
tol	Specifies the tolerance in determining whether the two models are similar. The default value of <b>tol</b> is 0.00000001. <b>tol</b> is a real scalar.

# Outputs

Name	Description
similar	Returns <b>1</b> if the two models are similar. This function returns <b>0</b> if the two models are not similar. <b>similar</b> is a Boolean.
Т	Returns the similarity transformation matrix this function uses to determine if the two models are similar. If a model is in transfer function or zero-pole-gain form, <b>T</b> contains the number of pole-zero cancellations for each input-output pair. <b>T</b> is a real matrix.

#### Examples

SysIn\_1 = zpk(1, 2, 3, 4, 5, 6,[1 1 1]);SysIn\_2 = (1+1.e-10)\*SysIn\_1; aresimilar(SysIn\_1, SysIn\_2, 1.e-9);

## **Related Topics**

<u>areequal</u>

# get (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

## Syntax

infoval = get(SysIn, infotype)

#### Description

Returns information about linear time-invariant (LTI) system model. You can return numerator and denominator coefficient values from transfer function models. You can return zero and pole locations, as well as a gain value, from zero-pole-gain models. You can return *A*, *B*, *C*, and *D* matrices from state-space models. You can return delay information and the sampling time from all model forms. You can return only one information type at a time.

**Examples** 

## Inputs

Name	Description	
SysIn	Specifies an l state-space f	LTI model in transfer function, zero-pole-gain, or orm.
infotype	Specifies the <b>SysIn</b> model. time. <b>infotyp</b> e	type of information you want to return from the You can specify only one information type at a e is a string that can take the following values:
	'num'	Specifies that you want to return the numerator polynomial coefficients of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in transfer function form.
	'den'	Specifies that you want to return the denominator polynomial coefficients of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in transfer function form.
	'z'	Specifies that you want to return the locations of the zeros of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in zero-pole- gain form.
	'p'	Specifies that you want to return the locations of the poles of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in zero-pole-gain form.
	'k'	Specifies that you want to return the feedback gain the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in zero-pole-gain form.
	'a'	Specifies that you want to return the state matrix of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in state-space form.
	'b'	Specifies that you want to return the input matrix of the <b>SysIn</b> model This option is valid only when the <b>SysIn</b> model is in state-space form.
	'c'	Specifies that you want to return the output matrix of the <b>SysIn</b> model. This option is valid

	only when the <b>SysIn</b> model is in state-space form.
'd'	Specifies that you want to return the direct transmission matrix of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in state-space form.
'ts'	Specifies that you want to return the sampling time of the <b>SysIn</b> model.
'inputdelay'	Specifies that you want to return the input delay of the <b>SysIn</b> model.
'outputdelay'	Specifies that you want to return the output delay of the <b>SysIn</b> model.
'iodelay' or 'transdelay'	Specifies that you want to return the transport delay, or the delay between the inputs and outputs of the <b>SysIn</b> model.

## Outputs

-	
Name	Description
infoval	Returns the value of the information you want to return from the <b>SysIn</b> model. You can specify only one value. The data type of the <b>infoval</b> input depends on the value you specify for the <b>infotype</b> input.
	If <b>infotype</b> is <b>'num'</b> , <b>'den'</b> , <b>'b'</b> , <b>'c'</b> , or <b>'d'</b> , <b>infoval</b> is a real scalar or real vector, depending on the number of elements you specify.
	If <b>infotype</b> is <b>'z'</b> or <b>'p'</b> , <b>infoval</b> is a complex vector.
	If <b>infotype</b> is <b>'a'</b> or <b>'k'</b> , <b>'infoval'</b> is a real scalar or real matrix, depending on the number of inputs, outputs, or states of the model.
	If <b>infotype</b> is <b>'ts'</b> , <b>infoval</b> is a real scalar.
	If <b>infotype</b> is <b>'inputdelay'</b> , <b>infoval</b> is a real vector whose <i>i</i> -th element corresponds to the delay of the <i>i</i> -th input to the <b>SysIn</b> model.
	If <b>infotype</b> is <b>'outputdelay'</b> , <b>infoval</b> is a real vector whose <i>j</i> -th element corresponds to the delay of the <i>j</i> -th output from the <b>SysIn</b> model.
	If <b>infotype</b> is <b>'iodelay'</b> or <b>'transdelay'</b> , <b>infoval</b> is a real matrix whose <i>ij</i> -th element corresponds to the delay between the <i>i</i> -th input and the <i>j</i> -th output of the <b>SysIn</b> model.

#### Examples

SysIn = tf(1, [0.2, 4]) SysOut = set(SysIn, 'num', [.05, 3], 'inputdelay', 0.8, 'ts', 0.1) num = get(SysOut, 'num') SysIn = ss(1, 1, 1, 0, 0.1) SysOut = set(SysIn, 'c', 5) c = get(SysOut, 'c')

## **Related Topics**

<u>set</u> pade hasdelay delay2z

# hasdelay (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

## Syntax

delayed = hasdelay(SysIn)

## Description

Determines whether a system model has delay present.

Examples
## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

## Outputs

Name	Description
delayed	Returns 1 if the <b>SysIn</b> model has delay present. This function returns 0 if the <b>SysIn</b> model does not have delay present. <b>delayed</b> is a Boolean.

### Examples

SysIn = ss(-1, 1, 2, 0, 0.1)SysIn = set(SysIn, 'inputdelay', 4) delayed = hasdelay(SysIn)

## **Related Topics**

delay2z totaldelay get set

# isct (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

iscon = isct(SysIn)

### Description

Determines whether a system model is continuous.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

## Outputs

Name	Description
iscon	Returns 1 if the <b>SysIn</b> model is continuous. This function returns 0 if the <b>SysIn</b> model is discrete. <b>iscon</b> is a Boolean.

### Examples

Ts = 0.1SysIn = ss(-1, 1, 2, 0, Ts) iscon = isct(SysIn)

## **Related Topics**

<u>isdt</u>

# isctrb (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

[control, stabilize] = isctrb(SysInSS, tol)
[control, stabilize] = isctrb(A, B, tol)

#### Description

Determines whether a system model is controllable or stabilizable. A system of order *n* is controllable if the <u>controllability matrix</u> is full rank, meaning the rank of this matrix is equal to *n*. A system is stabilizable if all the unstable eigenvalues are controllable.

**Examples** 

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) system in transfer function, zero-pole-gain, or state-space form. This function converts transfer function and zero-pole-gain models to state- space form.
tol	Specifies the tolerance in determining whether the controllability matrix is rank-deficient. If a diagonal value in the <b>A</b> matrix of <b>SysInSS</b> is less than the value of <b>tol</b> , this function considers that value equal to 0. The default value of <b>tol</b> is 0.0000001. <b>tol</b> is a real scalar.
A	Specifies an <i>n</i> x <i>n</i> state matrix, where <i>n</i> is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.

## Outputs

Name	Description
control	Returns <b>1</b> if the <b>SysInSS</b> model is controllable. This output returns <b>0</b> if the model is not controllable.
stabilize	Returns <b>1</b> if the <b>SysInSS</b> model is stabilizable. This output returns <b>0</b> if the model is not stabilizable.

### Examples

SysInSS = ss(-eye(2), [1 ; 1e-16], [1 1], 0);[control, stabilize] = isctrb(SysInSS, 1e-3);

## **Related Topics**

ctrb ctrbf obsvf obsv isobsv gram

# isdt (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

### Syntax

isdisc = isdt(SysIn)

### Description

Determines whether a system model is discrete.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

## Outputs

Name	Description
isdisc	Returns 1 if the <b>SysIn</b> model is discrete. This function returns 0 if
	the <b>SysIn</b> model is continuous. <b>isdisc</b> is a Boolean.

### Examples

Ts = 0.1SysIn = ss(-1, 1, 2, 0, Ts) isdisc = isdt(SysIn)

## **Related Topics**

<u>isct</u>

# isempty (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

### Syntax

isemp = isempty(SysIn)

### Description

Determines whether a system model is empty.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

## Outputs

Name	Description
isemp	Returns 1 if the <b>SysIn</b> model is empty. This function returns <b>0</b> if
	the <b>SysIn</b> model is not empty. <b>isemp</b> is a Boolean.

### Examples

SysIn = tf(1, [5 6])isemp = isempty(SysIn)

## **Related Topics**

<u>ss</u> tf zpk

# isobsv (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

[observe, detect] = isobsv(SysInSS, tol) [observe, detect] = isobsv(A, C, tol)

#### Description

Determines whether a system model is observable or detectable. A system of order *n* is observable if the <u>observability matrix</u> is full rank, meaning the rank of this matrix is equal to *n*. A system is detectable if all the unstable eigenvalues are observable.

**Examples** 

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) system in transfer function, zero-pole-gain, or state-space form. This function converts transfer function and zero-pole-gain models to state- space form before determining whether the model is observable or detectable.
tol	Specifies the tolerance in determining whether the observability matrix is rank-deficient. If a diagonal value in the <b>A</b> matrix of <b>SysInSS</b> is less than the value of <b>tol</b> , this function considers that value equal to 0. The default value of <b>tol</b> is 0.0000001. <b>tol</b> is a real scalar.
A	Specifies an <i>n</i> x <i>n</i> state matrix, where <i>n</i> is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
# Outputs

Name	Description	
observe	Returns <b>1</b> if the <b>SysInSS</b> model is observable. This output returns <b>0</b> if the model is not observable.	
detect	Returns <b>1</b> if the <b>SysInSS</b> model is detectable. This output returns <b>0</b> if the model is not detectable.	

# Examples

A = eye(2);C = [0 0]; observe = isobsv(A,C);

# **Related Topics**

obsvf obsv gram ctrb ctrbf isctrb

# isproper (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

isprop = isproper(SysIn)

#### Description

Determines whether a system model is proper. Transfer function and zero-pole-gain models are proper if the order of the denominator polynomial function is greater than or equal to the order of the numerator polynomial function. If the order of the denominator is greater than the order of the numerator, the model is strictly proper. State-space models always are proper.

**Examples** 

# Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

# Outputs

Name	Description	
isprop	Returns 1 if the <b>SysIn</b> model is proper. This function returns <b>0</b> if	
	the <b>SysIn</b> model is not proper. <b>isprop</b> is a Boolean.	

#### Examples

num = [1, 1]den = [1] SysIn = tf(num, den) isprop = isproper(SysIn)

# **Related Topics**

<u>tf</u> <u>zpk</u> <u>ss</u>

# issiso (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

siso = issiso(SysIn)

#### Description

Determines whether a system model is single-input single-output (SISO). Examples

# Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

# Outputs

Name	Description
siso	Returns 1 if the <b>SysIn</b> model is SISO. This function returns <b>0</b> if
	the <b>SysIn</b> model is not SISO. <b>siso</b> is a Boolean.

# Examples

D = [1]SysIn = ss([], [], [], D) siso = issiso(SysIn)

# isstable (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

# Syntax

stable = isstable(SysIn)

#### Description

Determines whether a system model is stable.

Examples

# Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

# Outputs

Name	Description
stable	Returns 1 if the <b>SysIn</b> model is stable. This function returns <b>0</b> if
	the <b>Sysin</b> model is not stable. <b>stable</b> is a Boolean.

#### Examples

SysIn = ss(1, 1, 2, 0);stable = isstable(Sys);

# **Related Topics**

<u>isctrb</u> isobsv

# ndims (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

# Syntax

dim = ndims(SysIn)

#### Description

Calculates the number of dimensions of a system model. The number of dimensions also is the order of the model.

Examples

# Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function,
	zero-pole-gain, or state-space form.

# Outputs

Name	Description
dim	Returns the number of dimensions of the <b>SysIn</b> model. <b>dim</b> is an integer scalar.

# Examples

A = [-1, 0.1; 0.2, -2]B = [0; 1] C = [1, 0] D = 0 SysIn = ss(A, B, C, D) dim = ndims(SysIn)

# **Related Topics**

<u>ss</u> tf zpk

# set (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

SysOut = set(SysIn, infotype, infoval)
SysOut = set(SysIn, infotype\_1, infoval\_1, ..., infotype\_n, infoval\_n)

#### Description

Sets information to a linear time-invariant (LTI) system model. You can set numerator and denominator coefficient values to transfer function models. You can set zero and pole locations, as well as a gain value, to zero-pole-gain models. You can set *A*, *B*, *C*, and *D* matrices to state-space models. You can set delay information and sampling time to all model forms. You can set more than one information type and value simultaneously.

**Examples** 

# Inputs

Name	Description	
SysIn	Specifies an I or state-space	LTI model in transfer function, zero-pole-gain, e form.
infotype_n	Specifies the <b>SysIn</b> model. type at a time following valu	type of information you want to set to the You can specify more than one information a. <b>infotype</b> is a string that can take the les:
	'num'	Specifies that you want to set the numerator polynomial coefficients of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in transfer function form.
	'den'	Specifies that you want to set the denominator polynomial coefficients of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in transfer function form.
	'z'	Specifies that you want to set the locations of the zeros of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in zero- pole-gain form.
	'p'	Specifies that you want to set the locations of the poles of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in zero- pole-gain form.
	'k'	Specifies that you want to set the feedback gain the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in zero-pole-gain form.
	'a'	Specifies that you want to set the state matrix of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in state-space form.
	'b'	Specifies that you want to set the input matrix of the <b>SysIn</b> model This option is valid only when the <b>SysIn</b> model is in state-space form.

	11		
	'c'	Specifies that you want to set the output matrix of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in state-space form.	
	'd'	Specifies that you want to set the direct transmission matrix of the <b>SysIn</b> model. This option is valid only when the <b>SysIn</b> model is in state-space form.	
	'ts'	Specifies that you want to set the sampling time of the <b>SysIn</b> model.	
	'inputdelay'	Specifies that you want to set the input delay of the <b>SysIn</b> model.	
	'outputdelay'	Specifies that you want to set the output delay of the <b>SysIn</b> model.	
	'iodelay' or 'transdelay'	Specifies that you want to set the transport delay, or the delay between the inputs and outputs of the <b>SysIn</b> model.	
infoval_n	Specifies the value of the information you want to set to the <b>SysIn</b> model. You can specify more than one value simultaneously. The data type of the <b>infoval</b> input depends on the value you specify for the <b>infotype</b> input.		
	If <b>infotype</b> is <b>'num'</b> , <b>'den'</b> , <b>'b'</b> , <b>'c'</b> , or <b>'d'</b> , <b>infoval</b> is a real scalar or real vector, depending on the number of elements you specify.		
	If <b>infotype</b> is <b>'z'</b> or <b>'p'</b> , <b>infoval</b> is a complex vector.		
	If <b>infotype</b> is <b>'a'</b> or <b>'k'</b> , <b>'infoval'</b> is a real scalar or real matrix, depending on the number of inputs, outputs, or states of the model.		
	If <b>infotype</b> is <b>'ts'</b> , <b>infoval</b> is a real scalar.		
	1		

SysIn model.

If **infotype** is **'outputdelay'**, **infoval** is a real vector whose *j*-th element corresponds to the delay of the *j*-th output from the **SysIn** model.

If **infotype** is **'iodelay'** or **'transdelay'**, **infoval** is a real matrix whose *ij*-th element corresponds to the delay between the *i*-th input and the *j*-th output of the **SysIn** model.
### Outputs

Name	Description
SysOut	Returns an LTI model with the information and value(s) you specified in the <b>infotype</b> and <b>infoval</b> inputs. <b>SysOut</b> is a model in transfer function, zero-pole-gain, or state-space form.

#### Examples

SysIn = tf(1, [0.2, 4]) SysOut = set(SysIn, 'num', [.05, 3], 'inputdelay', 0.8, 'ts', 0.1) SysIn = ss(1, 1, 1, 0, 0.1) SysOut = set(SysIn, 'c', 5)

#### **Related Topics**

get hasdelay pade delay2z

## size (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

### Syntax

numIO = size(SysIn) num = size(SysIn, sizetype)

#### Description

Returns the number of states, inputs, and/or outputs of a system model. <u>Examples</u>

### Inputs

Name	Descri	ption
SysIn	Specifi functio	es a linear time-invariant (LTI) model in transfer n, zero-pole-gain, or state-space form.
sizetype	Specific do not inputs a integer	es the size type you want this function to return. If you specify a value, this function returns both the number of and the number of outputs. <b>sizetype</b> takes the following or string values:
	1	Specifies that this function returns the number of outputs from the model.
	2	Specifies that this function returns the number of inputs to the model.
	'order'	Specifies that this function returns the order of the model, which also is the number of states. This option is valid only for state-space models.

## Outputs

Name	Description
numIO	Returns the number of inputs and outputs of the <b>SysIn</b> model. <b>numIO</b> is an integer vector. This output is valid only if you do not specify a value for the <b>sizetype</b> input.
num	Returns the number of states, inputs, or outputs, depending on the value you specify for the <b>sizetype</b> input. <b>num</b> is an integer scalar. This output is valid only if you specify a value for the <b>sizetype</b> input.

#### Examples

SysIn = rss(4, 5, 6) numIO = size(SysIn) num = size(SysIn, 2)

### **Related Topics**

balreal minreal modred

## ssdata (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

A = ssdata(SysInSS)

[A, B] = ssdata(SysInSS)

[A, B, C] = ssdata(SysInSS)

[A, B, C, D] = ssdata(SysInSS)

[A, B, C, D, Ts] = ssdata(SysInSS)

#### Description

Returns information about a state-space system model.

Examples

#### Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.

## Outputs

Name	Description
Α	Returns the $n \ge n$ state matrix of the <b>SysInSS</b> model, where $n$ is the number of states. <b>A</b> is a real matrix.
В	Returns the $n \ge m$ input matrix of the <b>SysInSS</b> model, where $m$ is the number of inputs. <b>B</b> is a real matrix.
С	Returns the $r \ge n$ output matrix of the <b>SysInSS</b> model, where $r$ is the number of outputs. <b>C</b> is a real matrix.
D	Returns the <i>r</i> x <i>m</i> direct transmission matrix of the <b>SysInSS</b> model. <b>D</b> is a real matrix.
Ts	Returns the sampling time of the <b>SysInSS</b> model. If the value of <b>Ts</b> is <b>0</b> , the <b>SysInSS</b> model is continuous. Otherwise, the model is discrete. <b>Ts</b> is a real scalar.

#### Examples

A = [1, 1; -1, 2]B = [1, 2]' C = [2, 1] D = 0 SysInSS = ss(A, B, C, D) [A, B, C, D, Ts] = ssdata(SysInSS)

## **Related Topics**

<u>SS</u>

## tfdata (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

[num, den, Ts] = tfdata(SysInTF)
[num, den, Ts] = tfdata(SysInTF, iy, iu)
[num, den, delay, Ts] = tfdata(SysInTF)
[num, den, delay, Ts] = tfdata(SysInTF, iy, iu)

#### Description

Returns information about a transfer function system model.

Examples

### Inputs

Name	Description
SysInTF	Specifies a linear time-invariant (LTI) system model in transfer function form.
iy	Specifies the index number of a model output. The default value is 1. <b>iy</b> is an integer scalar.
iu	Specifies the index number of a model input. <b>iu</b> is an integer scalar.

## Outputs

Name	Description
num	Returns the coefficients of the numerator polynomial function, in descending order. <b>num</b> either is a real scalar or a real vector, depending on the number of coefficients.
den	Returns the coefficients of the denominator polynomial function, in descending order. <b>den</b> either is a real scalar or a real vector, depending on the number of coefficients.
delay	Returns the total delay present in the <b>SysInTF</b> model, in seconds.
Ts	Returns the sampling time of the <b>SysInTF</b> model. If the value of <b>Ts</b> is <b>0</b> , the <b>SysInTF</b> model is continuous. Otherwise, the model is discrete.

#### Examples

```
num1 = 1num2 = 2
den1 = [1, 1]
den2 = [1, -1]
r = 1
m = 2
SysInTF = tf(num1, num2, den1, den2, [r, m])
[num2, den2] = tfdata(SysInTF, 1, 2)
```

## **Related Topics**

<u>tf</u>

## zpkdata (Control Design and Simulation Module, MathScript Function)

Member of the <u>info</u> class.

#### Syntax

- [z, p, K, Ts] = zpkdata(SysInZPK)
- [z, p, K, delay, Ts] = zpkdata(SysInZPK)
- [z, p, K, Ts] = zpkdata(SysInZPK, iy, iu)
- [z, p, K, delay, Ts] = zpkdata(SysInZPK, iy, iu)

#### Description

Returns information about a zero-pole-gain system model. <u>Examples</u>

### Inputs

Name	Description
SysInZPK	Specifies a linear time-invariant (LTI) model in zero-pole-gain form.
iy	Specifies the index number of a model output. The default value is 1. <b>iy</b> is an integer scalar.
iu	Specifies the index number of a model input. <b>iu</b> is an integer scalar.

## Outputs

Name	Description
z	Returns the locations of the zeros of the <b>SysInZPK</b> model. <b>z</b> is a complex vector.
р	Returns the locations of the poles of the <b>SysInZPK</b> model. <b>p</b> is a complex vector.
К	Returns the gain of the <b>SysInZPK</b> model. <b>K</b> is a real scalar or real matrix, depending on the number of inputs and outputs of the <b>SysInZPK</b> model.
delay	Returns the total delay present in the <b>SysInZPK</b> model, in seconds.
Ts	Returns the sampling time of the <b>SysInZPK</b> model. If the value of <b>Ts</b> is <b>0</b> , the <b>SysInZPK</b> model is continuous. Otherwise, the model is discrete. <b>Ts</b> is a real scalar.

#### Examples

z1 = 1p1 = 2 z2 = [] p2 = [1, 1] K = [2; 1] SysInZPK = zpk(z1, z2, p1, p2, K) [z2, p2, k2] = zpkdata(SysInZPK, 2, 1)

## **Related Topics**

<u>zpk</u>

# reduce (Control Design and Simulation Module, MathScript Class)

Use members of the reduce class to cancel zero-pole pairs in transfer function and zero-pole-gain system models or to reduce the number of states in state-space models. You also can use members of this class to eliminate inputs and outputs that are uncontrollable or unobservable.

Function	Description
<u>minreal</u>	Returns the minimal realization of a system model
modred	Reduces the order of a state-space system model
<u>remove</u>	Removes inputs, outputs, and/or states from a model
<u>select</u>	Constructs a new system model by selecting inputs, outputs, and/or states that you want to keep from an existing model
<u>sminreal</u>	Calculates the minimal state realization of a state-space system model

## minreal (Control Design and Simulation Module, MathScript Function)

Member of the <u>reduce</u> class.

#### Syntax

SysMin = minreal(SysIn) SysMin = minreal(SysIn, tol)

#### Description

Returns the minimal realization of a system model. For state-space models, a minimal realization removes all uncontrollable and unobservable states. For transfer function and zero-pole-gain models, a minimal realization cancels all matching zero-pole pairs.

#### **Examples**

### Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
tol	Specifies the threshold this function uses to determine the modes to eliminate. For state-space models, <b>tol</b> specifies the controllability and observability thresholds. For transfer function and zero-pole-gain models, <b>tol</b> specifies zero-pole cancellation closeness. The default is 1E-12. <b>tol</b> is a real scalar.
# Outputs

Name	Description
SysMin	Returns the minimal realization of the <b>SysIn</b> model. <b>SysMin</b> is an LTI model in transfer function, zero-pole-gain, or state-space form.

## Examples

z = 1p = [1, -1]k = 1 SysIn = zpk(z, p, k) SysMin = minreal(SysIn)

## **Related Topics**

<u>balreal</u> <u>modred</u> <u>sminreal</u>

# modred (Control Design and Simulation Module, MathScript Function)

Member of the <u>reduce</u> class.

#### Syntax

SysRedSS = modred(SysInSS, states) SysRedSS = modred(SysInSS, states, method)

#### Description

Eliminates state dynamics that you want to ignore in a state-space system model. This function can eliminate the state dynamics by deleting the states or by assuming steady-state conditions for states with fast dynamics. This assumption is a pseudo-steady-state assumption. The pseudo-steady-state assumption does not affect the discrete or continuous system gain of the original system, so the remaining states have the same gain as the full order system.

**Examples** 

## Inputs

Name	Desci	ription
SysInSS	Speci <sup>:</sup> form.	fies a linear time-invariant (LTI) model in state-space
states	Speci <sup>.</sup> elimin	fies the index numbers of the states you want to ate. <b>states</b> is an integer scalar or vector.
method	Speci the <b>Sy</b> follow	fies the method you want to use to reduce the order of <b>/sInSS</b> model. <b>method</b> is a string that can take the ing values:
	'mdc'	(Default) Specifies that this function substitutes the steady-state values of the specified states into the dynamics of the reduced-order model.
	'del'	Specifies that this function eliminates the specified states from the reduced-order model.

# Outputs

Name	Description
SysRedSS	Returns the reduced-order version of the <b>SysInSS</b> model.

#### Examples

A = [-1, 0.1; 0.2, -2] B = [0; 1] C = [1, 0] D = 0 states = 1 SysInSS = ss(A, B, C, D) SysRed = modred(SysInSS, states)

## **Related Topics**

<u>balreal</u> <u>minreal</u> <u>sminreal</u>

# remove (Control Design and Simulation Module, MathScript Function)

Member of the <u>reduce</u> class.

## Syntax

SysRed = remove(SysIn, i, j, k)

#### Description

Constructs a new system model by removing specified inputs and/or outputs from an existing model. You also can remove states from an existing state-space model.

**Examples** 

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
i	Specifies a list of model inputs to remove from the <b>SysIn</b> model. The default value is empty. A value of $-1$ specifies that you want to remove all inputs. <b>i</b> is a real vector.
j	Specifies a list of model outputs to remove from the <b>SysIn</b> model. The default value is empty. A value of $-1$ specifies that you want to remove all outputs. <b>j</b> is a real vector.
k	Specifies a list of model states to remove from the <b>SysIn</b> model. The default value is empty. A value of $-1$ specifies that you want to remove all states. This input is valid only if you specify a state- space model. <b>k</b> is a real vector.

## Outputs

Name	Description
SysRed	Returns the <b>SysIn</b> model without the specified inputs and outputs. If you specified a state-space model for the <b>SysIn</b> input, the <b>SysRed</b> model also lacks the specified states.

#### Examples

SysIn= tf(1, 2, 3, 4, 5, 6, -1, -2, -3, -4, -5, -6, [2 3]);SysRed = remove(SysIn, [2 1], [2 1]);

## **Related Topics**

<u>select</u>

# select (Control Design and Simulation Module, MathScript Function)

Member of the <u>reduce</u> class.

## Syntax

SysSel = select(SysIn, i, j, k) SysSel = select(SysIn, i, j, k, ord)

#### Description

Constructs a new system model by selecting inputs and/or outputs from an existing model. You also can select states from a state-space model.

Examples

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.
i	Specifies a list of model inputs to keep from the <b>SysIn</b> model. The default value is empty. A value of $-1$ specifies that you want to keep all inputs. <b>i</b> is a real vector.
j	Specifies a list of model outputs to keep from the <b>SysIn</b> model. The default value is empty. A value of $-1$ specifies that you want to keep all outputs. <b>j</b> is a real vector.
k	Specifies a list of model states to keep from the <b>SysIn</b> model. The default value is empty. A value of $-1$ specifies that you want to keep all states. This input is valid only if you specify a state- space model. <b>k</b> is a real vector.
ord	Specifies whether the <b>SysSel</b> model orders the selected inputs/outputs/states in the order in which you selected them. To enable this behavior, specify a value of 1 for this input. If you specify a value of 0, the <b>SysSel</b> contains the selected inputs/outputs/states in the order in which they existed in the <b>SysIn</b> model. The default value is 0.

## Outputs

Name	Description
SysSel	Returns the <b>SysIn</b> model with only the specified inputs and outputs. If you specified a state-space model for the <b>SysIn</b> input, the <b>SysSel</b> model also contains the specified states.

#### Examples

SysIn = tf(1, 2, 3, 4, 5, 6, -1, -2, -3, -4, -5, -6, [2 3]);SysSel= select(SysIn, [2 1], [2 1], 1);

## **Related Topics**

remove modred

# sminreal (Control Design and Simulation Module, MathScript Function)

Member of the <u>reduce</u> class.

## Syntax

SysMinSS = sminreal(SysInSS)

#### Description

Calculates the minimal state realization of a state-space system model. This function calculates the minimal state realization based on unconnected (null) rows and columns in the controllability and observability matrices. This function only eliminates inputs or outputs when columns or rows of these matrices are below a tolerance, defined as 2E-16, even though the controllability and observability matrices might be rank deficient.

**Examples** 

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.

## Outputs

Name	Description
SysMinSS	Returns an LTI state-space model that is the minimal state realization of the <b>SysInSS</b> model.

#### Examples

A = 0B = 0 C = 0 D = 1 SysInSS = ss(A, B, C, D)SysMinSS = sminreal(SysInSS)

## **Related Topics**

balreal minreal modred

# ssanals (Control Design and Simulation Module, MathScript Class)

Use members of the ssanals class to calculate properties of a given state-space model, such as observability, controllability, similarity transformations, model balance, and system Grammians.

Function	Description
<u>balreal</u>	Uses Grammians to balance a state-space system model
<u>canon</u>	Transforms a state-space system model into the canonical form
<u>ctrb</u>	Calculates the controllability matrix of a state-space system model
<u>ctrbf</u>	Calculates the controllable staircase of a state-space system model
<u>gram</u>	Calculates the controllability or observability Grammians of a state-space system model
<u>obsv</u>	Calculates the observability matrix of a system model
<u>obsvf</u>	Calculates the observable staircase transformation of a system model
<u>ssbal</u>	Balances a state-space system model using a diagonal similarity transformation

# balreal (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssanals</u> class.

#### Syntax

SysBalSS = balreal(SysIn) [SysBalSS, d] = balreal(SysIn) [SysBalSS, d, T] = balreal(SysIn) [SysBalSS, d, T, Tinv] = balreal(SysIn)

#### Description

Balances a state-space model. A balanced state-space model has identical controllability and observability diagonal Grammians. This function also returns the transformation matrix this function uses to balance the model.

**Examples** 

## Inputs

Name	Description
SysIn	Specifies a linear time-invariant (LTI) system in transfer function,
	zero-pole-gain, or state-space form.
## Outputs

Name	Description
SysBalSS	Returns the balanced LTI state-space model that results when this function applies the transformation matrix <b>T</b> to the <b>SysIn</b> model.
d	Returns the diagonal vector of the observability and controllability Grammians of the <b>SysBaISS</b> model. <b>d</b> is a real vector.
Т	Returns the transformation matrix this function uses to balance the <b>SysIn</b> model. This function calculates the <b>T</b> matrix using the Cholesky decomposition of the controllability and observability Grammians of the <b>SysIn</b> model. <b>T</b> is a real matrix.
Tinv	Returns the inverse of the transformation matrix <b>T</b> . <b>Tinv</b> is a real matrix.

## Examples

SysIn = ss([-1, 0; 0.1, -3], [1, 0]', [0, 1], 0)[SysBalSS, d, T, Tinv] = balreal(SysIn)

## **Related Topics**

gram modred <u>ss</u> ssbal

## canon (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssanals</u> class.

#### Syntax

SysCanSS = canon(SysInSS) SysCanSS = canon(SysInSS, formtype) SysCanSS = canon(A, B, C, D) SysCanSS = canon(A, B, C, D, formtype) [SysCanSS, T] = canon(SysInSS)

[SysCanSS, T] = canon(SysInSS, formtype)

[SysCanSS, T] = canon(A, B, C, D)

[SysCanSS, T] = canon(A, B, C, D, formtype)

#### Description

Transforms a state-space system model to a canonical form using the **formtype** that you specify. This function also returns the transformation matrix that this function uses to transform the model.

**Examples** 

## Inputs

Name	Description	
SysInSS	Specifies a li function, zero converts tran space form b form.	near time-invariant (LTI) system in transfer p-pole-gain, or state-space form. This function sfer function and zero-pole-gain models to state- before transforming the model into canonical
formtype	Specifies the the <b>SysIn</b> mo string that ca	canonical form this function uses to transform odel into the <b>SysCanSS</b> model. <b>method</b> is a In take the following values:
	'modal'	(Default) <b>A</b> is a bi-diagonal matrix. The elements in the main diagonal are the real part of the eigenvalues of <b>A</b> , while elements in the upper secondary or lower secondary diagonal represent the imaginary part of the eigenvalues of <b>A</b> . Therefore, <b>A</b> is strictly diagonal if and only if all eigenvalues are real.
	'companion'	<b>A</b> has the coefficients of the characteristic polynomial of <b>A</b> in its top row.
A	Specifies an states. The c	<i>n</i> x <i>n</i> state matrix, where <i>n</i> is the number of lefault is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an inputs. The d	<i>n</i> x <i>m</i> input matrix, where <i>m</i> is the number of lefault is an empty matrix. <b>B</b> is a real matrix.
С	Specifies an outputs. The	<i>r</i> x <i>n</i> output matrix, where <i>r</i> is the number of default is an empty matrix. <b>C</b> is a real matrix.
D	Specifies an an empty ma	$r \ge m$ direct transmission matrix. The default is a transmission matrix. <b>D</b> is a real matrix.

### Outputs

Name	Description
SysCanSS	Returns an LTI model in state-space form that is the canonical realization of <b>SysInSS</b> .
Т	Returns the transformation matrix this function uses to transform the <b>SysInSS</b> model into the <b>SysCanSS</b> model. <b>T</b> is a real matrix.

## Examples

SysInSS = ss([1, 1; -1, 2], [1, 2]', [2, 1], 0) [SysCanSS, T] = canon(SysInSS, 'modal')

## **Related Topics**

ctrb ctrbf ss ss2ss

# ctrb (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssanals</u> class.

### Syntax

Cb = ctrb(SysInSS) Cb = ctrb(A, B)

#### Description

Calculates the controllability matrix of the **SysInSS** system model. You use the controllability matrix to determine if the given model is controllable. An *n*-th order model is controllable if the controllability matrix is full rank, meaning the rank of the controllability matrix equals *n*.

#### **Examples**

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
A	Specifies an <i>n</i> x <i>n</i> state matrix, where <i>n</i> is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.

## Outputs

Name	Description
Cb	Returns the controllability matrix of the <b>SysInSS</b> model. <b>Cb</b> is a real matrix.

## Examples

SysInSS = ss([1, 1; -1, 2], [1, 2]', [2, 1], 0)Cb = ctrb(SysInSS)

## **Related Topics**

<u>obsvf</u> <u>ctrbf</u> gram

# ctrbf (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssanals</u> class.

#### Syntax

- [As, Bs, Cs, T, l] = ctrbf(SysInSS)
- [As, Bs, Cs, T, l] = ctrbf(SysInSS, tol)
- [As, Bs, Cs, T, l] = ctrbf(A, B, C)
- [As, Bs, Cs, T, l] = ctrbf(A, B, C, tol)

#### Description

Calculates the controllable staircase representation of the **SysInSS** system model. This function also returns a list of controllable states.

Examples

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
A	Specifies an <i>n</i> x <i>n</i> state matrix, where <i>n</i> is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
tol	Specifies the tolerance. If a diagonal value in the <b>A</b> matrix is less than the value of <b>tol</b> , this function considers that value equal to 0. The default value of <b>tol</b> is 0.0001. <b>tol</b> is a real scalar.

## Outputs

Name	Description
As	Returns the controllability staircase state matrix <b>A</b> . The controllable states are in the lower right corner of the matrix. <b>As</b> is a real matrix.
Bs	Returns the controllability staircase input matrix <b>B</b> . The controllable inputs are in the lower right corner of the matrix. <b>Bs</b> is a real matrix.
Cs	Returns the controllability staircase output matrix <b>C</b> . The controllable outputs are in the lower right corner of the matrix. <b>Cs</b> is a real matrix.
Т	Returns the controllability transformation matrix. <b>T</b> is a real matrix.
I	Returns the indexes of the controllable states of the <b>SysInSS</b> model. <b>I</b> is an integer vector.

## Examples

A = [-1, 2; 0, -3]B = [0; 1] C = [1, 0] [As, Bs, Cs] = ctrbf(A, B, C)

## **Related Topics**

ctrb obsvf obsv gram

## gram (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssanals</u> class.

## Syntax

Gr = gram(SysInSS, type)

#### Description

Calculates the controllability or observability Grammians of a stable linear time-invariant (LTI) system model in state-space form.

Examples

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space.
type	Specifies whether this function computes the controllability or observability Grammians of the <b>SysInSS</b> model. <b>type</b> is a string that can take the following values:
	'c' Specifies that this function computes the controllability Grammian.
	'o' Specifies that this function computes the observability Grammian.

## Outputs

Name	Description
Gr	Returns the controllability or observability Grammian of the <b>SysInSS</b> model, depending on the value you specify for the <b>type</b> input. <b>Gr</b> is a real matrix.

## Examples

SysIn = ss([-5, 1; -1, -2], [1, 2]', [2, 1], 0) Gr = gram(SysIn, 'O')

## **Related Topics**

balreal SS Ssbal Ctrbf Ctrb Obsv Obsvf

# obsv (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssanals</u> class.

## Syntax

Ob = obsv(SysInSS) Ob = obsv(A, C)

#### Description

Calculates the observability matrix of a state-space system model. You use the observability matrix to determine if the given system is observable. A model is observable if you can estimate each state of the model by looking at only the output response.

#### **Examples**

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
A	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.

## Outputs

Name	Description
Ob	Returns the observability matrix of the <b>SysInSS</b> model. This model is observable if the observability matrix is full column rank. <b>Ob</b> is a real matrix.
## Examples

A = [1, 1; -1, 2]B = [1, 2]' C = [2, 1] D = 0 SysInSS = ss(A, B, C, D) Ob = obsv(SysInSS)

## **Related Topics**

obsvf ctrb ctrbf gram

# obsvf (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssanals</u> class.

#### Syntax

- [As, Bs, Cs, T, l] = obsvf(SysInSS)
- [As, Bs, Cs, T, l] = obsvf(SysInSS, tol)
- [As, Bs, Cs, T, l] = obsvf(A, B, C)
- [As, Bs, Cs, T, l] = obsvf(A, B, C, tol)

#### Description

Calculates the observable staircase transformation of the **SysInSS** system model. You can use the staircase representation to identify observable and unobservable states of the model. This function also returns a list of observable states.

**Examples** 

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) system in state-space form.
A	Specifies an <i>n</i> x <i>n</i> state matrix, where <i>n</i> is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
tol	Specifies the tolerance. If a diagonal value in the <b>A</b> matrix is less than the value of <b>tol</b> , this function considers that value equal to 0. The default value of <b>tol</b> is 0.0001. <b>tol</b> is a real scalar.

## Outputs

Name	Description
As	Returns the observability staircase state matrix <b>A</b> . <b>As</b> is a real matrix.
Bs	Returns the observability staircase input matrix <b>B</b> . <b>Bs</b> is a real matrix.
Cs	Returns the observability staircase output matrix <b>C</b> . <b>Cs</b> is a real matrix.
Т	Returns the observability transformation matrix. $\mathbf{T}$ is a real matrix.
I	Returns the indexes of the observable states of the <b>SysInSS</b> model. <b>I</b> is an integer vector.

## Examples

A = [-1, 2; 0, -3]B = [0; 1] C = [1, 0] [As, Bs, Cs] = obsvf(A, B, C)

## **Related Topics**

obsv ctrb ctrbf gram

# ssbal (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssanals</u> class.

## Syntax

[SysBalSS, T] = ssbal(SysInSS) [SysBalSS, T] = ssbal(SysInSS, tol)

#### Description

Transforms a state-space system model into a balanced state-space model using a diagonal similarity transformation. This function also returns the transformation matrix this function uses to transform the model.

**Examples** 

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
tol	Specifies the tolerance for the diagonal transformation. <b>tol</b> limits the ill-conditioning of the transformation matrix <b>T</b> . This function estimates this condition using the ratio of maximum and minimum norms of the eigenvalues of <b>T</b> . The default value of <b>tol</b> is inf. <b>tol</b> is a real scalar.

#### Outputs

Name	Description
SysBalSS	Returns the balanced LTI state-space model.
Т	Returns the transformation matrix this function uses to balance the <b>SysInSS</b> model.

## Examples

SysIn = ss([1, 1; -1, 2], [1, 2]', [2, 1], 0)[SysBalSS, T] = ssbal(SysIn)

## **Related Topics**

balreal gram modred SS

# ssdesign (Control Design and Simulation Module, MathScript Class)

Use members of the ssdesign class to calculate controller and observer gains for closed-loop state feedback control or to estimate a state-space model. You also can use these functions to configure and test state-space controllers and state estimators.

Function	Description
<u>acker</u>	Places poles using the Ackermann formula
<u>augstate</u>	Adds states to system model outputs
<u>kalman</u>	Calculates the optimal steady-state Kalman gain for a system model
<u>kalmd</u>	Calculates the optimal steady-state Kalman gain for a discrete system model
<u>dlqr</u>	Calculates the gain that minimizes a linear quadratic cost function for a discrete system model, with weights on the model states
<u>dlqry</u>	Calculates the gain matrix that minimizes a linear quadratic cost function for a discrete system model, with weights on the model outputs
<u>estim</u>	Defines a state estimator
<u>lqr</u>	Calculates the gain matrix that minimizes a linear quadratic cost function for a continuous system model, with weights on the model states
<u>lqrd</u>	Calculates the discrete gain matrix that minimizes a linear quadratic cost function for a continuous system model, with weights on the model states
<u>lqrdy</u>	Calculates the discrete gain matrix that minimizes a linear quadratic cost function for a continuous system model, with weights on the model outputs
lqry	Calculates the gain matrix that minimizes a linear quadratic cost function for a continuous system model, with weights on the model outputs
place	Places poles using the pole placement method
1	

# acker (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[K, q] = acker(A, B, p)
[K, q] = acker(A, B, p, 'K')
[K, q] = acker(SysInSS, p, 'K')
[L, q] = acker(A, C, p, 'L')
[L, q] = acker(SysInSS, p, 'L')
[Ltrans, q] = acker(A', C', p)

#### Description

Places closed-loop control poles using the Ackermann formula and the controllability matrix to determine the controller gain vector, which places the closed-loop poles at specified locations in a system model with full state feedback. You also can use this function to return the estimator gain vector. You can use this function with single-input single-output (SISO), single-input multiple-output (SIMO), and multiple-input single-output (MISO) models. For multiple-input multiple-output (MIMO) models, use the place function.

**Examples** 

## Inputs

Name	Description
Α	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
р	Specifies where you want to place the closed-loop poles. <b>p</b> is a complex vector.
'K'	Specifies that you want this function to design a controller gain vector. <b>'K'</b> is a string constant and is the default value of this argument. You also can enter <b>'c'</b> or <b>'controller'</b> .
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
'L'	Specifies that you want this function to design an estimator gain vector. <b>'L'</b> is a string constant. You also can enter <b>'o'</b> or <b>'observer'</b> .

## Outputs

Name	Description
К	Returns the feedback controller gain vector that produces a closed-loop model such that the locations of the poles are equal to the values you specified in the <b>p</b> vector. <b>K</b> is a real vector.
q	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>q</b> is a complex vector.
L	Returns the feedback estimator gain vector that produces a closed-loop model such that the locations of the poles are equal to the values you specified in the <b>p</b> vector. <b>L</b> is a real vector.
Ltrans	Returns the transpose of <b>L</b> . You then can use the <u>transpose</u> function to transpose <b>Ltrans</b> to obtain <b>L</b> . <b>Ltrans</b> is a real vector.

## Examples

A = [-1, 2; 0, -3]B = [0; 1] p = [-4 + i, -4 - i] [K, q] = acker(A, B, p, 'K')

## **Related Topics**

<u>place</u> <u>estim</u> <u>lqr</u>

# augstate (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

## Syntax

SysAugSS = augstate(SysInSS)

#### Description

Augments a space-state system model by appending states to the outputs of the model. The resulting model returns state values as outputs.

**Examples** 

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.

## Outputs

Name	Description
SysAugSS	Returns an LTI state-space model with states appended to the outputs.

## Examples

SysInSS = ss(-1, 1, 2, 0)SysAugSS = augstate(SysInSS)

## **Related Topics**

<u>feedback</u> <u>ss</u>

# dlqr (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[K, X, eig] = dlqr(A, B, Q, R) [K, X, eig] = dlqr(A, B, Q, R, N)

#### Description

Calculates the optimal steady-state feedback gain matrix **K** that minimizes a quadratic cost function for a discrete linear time-invariant (LTI) state-space system model. The cost function weights the model states. The quadratic cost function *J* is defined as the following equation:  $sum(\mathbf{x}'(k)\mathbf{Q}\mathbf{x}(k) + \mathbf{u}(k)'\mathbf{R}\mathbf{u}(k) + 2\mathbf{x}(k)'\mathbf{N}\mathbf{u}(k), k, 0, +inf)$ , where *k* is discrete time, **u** is the input vector, and **x** is the state vector.

**Examples** 

## Inputs

Name	Description
A	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
Q	Specifies a symmetric, positive semi-definite matrix that penalizes the state vector $\mathbf{x}$ in the cost function. $\mathbf{Q}$ is a real matrix.
R	Specifies a symmetric positive definite matrix that penalizes the input vector <b>u</b> in the cost function. The default is the identity matrix. <b>R</b> is a real matrix.
N	Specifies a matrix that penalizes the cross product between input and state vectors, such that $(\mathbf{Q}-\mathbf{N}^*\mathrm{inv}(\mathbf{R})^*\mathbf{N}')$ is positive semi- definite. The default is an appropriately sized matrix of zeros. <b>N</b> is a real matrix.
### Outputs

Name	Description
К	Returns the gain matrix such that $K = inv(B'XB+R)^*(B'XA+N')$ . K is a real matrix.
X	Returns the symmetric, positive semi-definite (stabilizing) solution to the discrete algebraic Riccati equation. $X$ is a real matrix.
eig	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>eig</b> is a complex vector.

### Examples

A = [0.9, 0.2; 0, 0.8] B = [0; 1] Q = [2, 0; 0, 2] R = 1 [K, X, eig] = dlqr(A, B, Q, R)

## **Related Topics**

dare Iqry dIqry Iqr reg

## dlqry (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[K, X, eig] = dlqry(SysInSS, Q, R)

[K, X, eig] = dlqry(SysInSS, Q, R, N)

[K, X, eig] = dlqry(A, B, C, D, Q, R)

[K, X,,eig] = dlqry(A, B, C, D, Q, R, N)

#### Description

Calculates the optimal steady-state feedback gain matrix **K** that minimizes a quadratic cost function for a linear discrete state-space system model. This cost function weights the model outputs. The quadratic cost function *J* is defined as the following equation:  $sum(\mathbf{y}'(k)\mathbf{Q}\mathbf{y}(k)+\mathbf{u}(k)'\mathbf{R}\mathbf{u}(k)+2\mathbf{y}(k)'\mathbf{N}\mathbf{u}(k), k, 0, +inf)$ , where *k* is discrete time, **u** is the input vector, and **y** is the output vector.

**Examples** 

### Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
Α	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
D	Specifies an $r \ge m$ direct transmission matrix. The default is an empty matrix. <b>D</b> is a real matrix.
Q	Specifies a symmetric, positive semi-definite matrix that penalizes the output vector <b>y</b> in the cost function. <b>Q</b> is a real matrix.
R	Specifies a symmetric positive definite matrix that penalizes the input vector <b>u</b> in the cost function. The default is the identity matrix. <b>R</b> is a real matrix.
N	Specifies a matrix that penalizes the cross product between input and output vectors, such that $(\mathbf{Q}-\mathbf{N}^*\mathrm{inv}(\mathbf{R})^*\mathbf{N}')$ is positive semi-definite. The default is an appropriately sized matrix of zeros. <b>N</b> is a real matrix.

### Outputs

Name	Description
К	Returns the gain matrix such that $K = inv(B'XB+R)^*(B'XA+N')$ . K is a real matrix.
X	Returns the symmetric, positive semi-definite (stabilizing) solution to the discrete algebraic Riccati equation. $X$ is a real matrix.
eig	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>eig</b> is a complex vector.

#### Examples

A = [.9, 0.25; 0, 0.8] B = [0; 1] C = [1, 0] D = 0 Q = 2 R = 1 [K, X, eig] = dlqry(A, B, C, D, Q, R)

## **Related Topics**

<u>dlqr</u> <u>lqr</u> dare lqry

## estim (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

SysEstSS = estim(SysInSS, L) SysEstSS = estim(SysInSS, L, mout) SysEstSS = estim(SysInSS, L, mout, kinp)

#### Description

Defines a state estimator based on a list of known inputs, measured outputs, a linear time-invariant (LTI) state-space system model, and the estimator gain matrix. You can use the <u>acker</u> function to obtain the estimator gain matrix of a single-input single-output (SISO), single-input multiple-output (SIMO), or multiple-input single-output (MISO) LTI statespace model. Use the <u>place</u> function to obtain the estimator gain matrix of a multiple-input multiple-output (MIMO) LTI state-space model.

**Examples** 

### Inputs

Name	Description
SysInSS	Specifies the LTI state-space model for which you want to define a state estimator.
L	Specifies the estimator gain matrix. $L$ is a real matrix.
mout	Specifies the measured outputs from the <b>SysInSS</b> model. <b>mout</b> is an integer vector.
kinp	Specifies the known inputs to the <b>SysInSS</b> model. <b>kinp</b> is an integer vector.

#### Outputs

Name	Description
SysEstSS	Returns the estimator structure this function defines.
3y323133	SysEstSS is a model in state-space form.

#### Examples

A = [1, 1; -1, 2]B = [1, 2]' C = [2, 1] SysInSS = ss(A,B,C,0) L = acker(A', C', [-1 -3])' SysEstSS = estim(SysInSS, L)

### **Related Topics**

lqr reg acker place

## kalman (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

### Syntax

[SysKal, L, P, M, Z] = kalman(SysInSS, Q, R, N)

#### Description

Calculates the optimal steady-state Kalman gain L that minimizes the covariance of the state estimation error. You can use this function to calculate L for continuous and discrete system models.

**Examples** 

### Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
Q	Specifies the auto-covariance matrix of the process noise, if the <b>SysInSS</b> model is discrete. If the model is continuous, <b>Q</b> specifies the auto-intensity matrix of the process noise. <b>Q</b> is a real matrix that is symmetric and positive semi-definite.
R	Specifies the auto-covariance matrix of the measurement noise, if the <b>SysInSS</b> model is discrete. If the model is continuous, <b>R</b> specifies the auto-intensity matrix of the process noise. <b>R</b> is a real matrix that is symmetric and positive definite.
Ν	Specifies the cross-covariance matrix between the process noise and the measurement noise, if the <b>SysInSS</b> model is discrete. If the model is continuous, <b>N</b> specifies the cross- intensity matrix between the process noise and the measurement noise. The default is an appropriately sized matrix of zeros, which specifies the process noise and measurement noise are uncorrelated. <b>N</b> must be valid such that ( <b>Q</b> - <b>N</b> *inv( <b>R</b> )* <b>N</b> ') is positive semi-definite. <b>N</b> is a real matrix.

## Outputs

Name	Description
SysKal	Returns the definition of the Kalman filter with the gain matrix <b>L</b> applied. <b>SysKal</b> is an LTI model in state-space form.
L	Returns the gain matrix that minimizes the covariance of the state estimation error. <b>L</b> is a real matrix.
Р	Returns the steady-state covariance of the estimation error. <b>P</b> is a real matrix.
М	Returns the steady-state innovation gain matrix. This gain matrix weights the difference between the observed and estimated outputs in the updated state estimates for discrete estimators. <b>M</b> is a real matrix.
Z	Returns the steady-state error covariance of the error between the actual states and the updated state estimates in the discrete estimation process. <b>Z</b> is a real matrix.

### Examples

SysInSS = ss(-1, 1, 1, 0)[SysKal, L, P] = kalman(SysInSS, 1, 2)

### **Related Topics**

<u>acker</u> place estim

## kalmd (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[SysKalDisc, L, P, M, Z] = kalmd(SysInSSCont, Q, R, Ts) [SysKalDisc, L, P, M, Z] = kalmd(SysInSSCont, Q, R, N, Ts)

#### Description

Calculates the optimal steady-state Kalman gain **L** that minimizes the covariance of the state estimation error. The input system and noise covariance are based on a continuous system. All outputs are based on a discretized system **SysKalDisc**, which is based on the sample rate **Ts**.

#### **Examples**

### Inputs

Name	Description
SysInSSCont	Specifies a continuous linear time-invariant (LTI) model in state-space form.
Q	Specifies the auto-covariance matrix of the continuous process noise. ${f Q}$ is a real matrix that is symmetric and positive semi-definite.
R	Specifies the auto-covariance matrix of the continuous measurement noise. <b>R</b> is a real matrix that is symmetric and positive definite.
Ts	Specifies the sampling time this VI uses to discretize the <b>SysInSSCont</b> model. <b>Ts</b> is a real scalar.
Ν	Specifies the cross-covariance matrix between the process noise and measurement noise. The default is an appropriately-sized matrix of zeros, which specifies the process noise and measurement noise are uncorrelated. <b>N</b> must be valid such that $(Q-N^*inv(R)^*N')$ is positive semi-definite. <b>N</b> is a real matrix.

## Outputs

Name	Description
SysKalDisc	Returns the definition of the discrete Kalman filter with the gain matrix <b>L</b> applied. <b>SysKalDisc</b> is a discrete LTI model in state-space form.
L	Returns the gain matrix that minimizes the covariance of the state estimation error. $L$ is a real matrix.
Р	Returns the steady-state covariance of the estimation error. <b>P</b> is a real matrix.
Μ	Returns the steady-state innovation gain matrix. This gain matrix weights the difference between the observed and estimated outputs in the state update equation for discrete estimators. <b>M</b> is a real matrix.
Z	Returns the steady-state error covariance of the error between the actual states and the updated state estimates in the discrete estimation process. <b>Z</b> is a real matrix.

### Examples

SysInSSCont = ss(-1, 1, 1, 0)[SysKalDisc, L, P] = kalmd(SysInSSCont, 1, 2, 1)

### **Related Topics**

<u>acker</u> <u>estim</u> <u>kalman</u> <u>place</u>

# Iqr (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[K, X, eig] = lqr(A, B, Q, R) [K, X, eig] = lqr(A, B, Q, R, N)

#### Description

Calculates the optimal steady-state feedback gain matrix **K** that minimizes a quadratic cost function for a linear discrete state-space system model. The cost function weights the model states. The quadratic cost function J is defined as the following equation:

integral( $\mathbf{x}'(t)\mathbf{Q}\mathbf{x}(t)+\mathbf{u}(t)'\mathbf{R}\mathbf{u}(t)+2\mathbf{x}(t)'\mathbf{N}\mathbf{u}(t)$ , *t*, 0, +inf), where *t* is continuous time, **u** is the input vector, and **x** is the state vector.

Examples

### Inputs

Name	Description
A	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
Q	Specifies a symmetric, positive semi-definite matrix that penalizes the state vector $\mathbf{x}$ in the cost function. $\mathbf{Q}$ is a real matrix.
R	Specifies a symmetric positive definite matrix that penalizes the input vector <b>u</b> in the cost function. The default is the identity matrix. <b>R</b> is a real matrix.
N	Specifies a matrix that penalizes the cross product between input and state vectors, such that $(\mathbf{Q}-\mathbf{N}^*\mathrm{inv}(\mathbf{R})^*\mathbf{N}')$ is positive semi- definite. The default is an appropriately sized matrix of zeros. <b>N</b> is a real matrix.

### Outputs

Name	Description
К	Returns the gain matrix such that $\mathbf{K} = inv(\mathbf{R})^*(\mathbf{B}'\mathbf{X}+\mathbf{N}')$ . <b>K</b> is a real matrix.
X	Returns the symmetric, positive semi-definite (stabilizing) solution to the discrete algebraic Riccati equation. $X$ is a real matrix.
eig	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>eig</b> is a complex vector.
### Examples

A = [-1, -2; 0, -4] B = [0; 1] Q = [2, 0; 0, 2] R = 1 [K, X, eig] = lqr(A, B, Q, R)

## **Related Topics**

<u>care</u> <u>dlqr</u> <u>reg</u>

# lqrd (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[K, X, eig] = lqrd(A, B, Q, R) [K, X, eig] = lqrd(A, B, Q, R, N)

#### Description

Calculates the discrete, optimal, steady-state feedback gain matrix **K** that minimizes a quadratic cost function for a continuous linear time-invariant (LTI) system model in state-space form. This function discretizes automatically the continuous system model before calculating **K**. The cost function weights the model states. The quadratic cost function *J* is defined as the following equation:

 $sum(\mathbf{x}'(k)\mathbf{Q}\mathbf{x}(k) + \mathbf{u}(k)'\mathbf{R}\mathbf{u}(k) + 2\mathbf{x}(k)'\mathbf{N}\mathbf{u}(k), k, 0, +inf)$ , where *k* is discrete time, **u** is the input vector, and **x** is the state vector.

**Examples** 

## Inputs

Name	Description
A	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
Q	Specifies a symmetric, positive semi-definite matrix that penalizes the state vector $\mathbf{x}$ in the cost function. $\mathbf{Q}$ is a real matrix.
R	Specifies a symmetric positive definite matrix that penalizes the input vector <b>u</b> in the cost function. The default is the identity matrix. <b>R</b> is a real matrix.
N	Specifies a matrix that penalizes the cross product between input and state vectors, such that $(\mathbf{Q}-\mathbf{N}^*\mathrm{inv}(\mathbf{R})^*\mathbf{N}')$ is positive semi- definite. The default is an appropriately sized matrix of zeros. <b>N</b> is a real matrix.

## Outputs

Name	Description
К	Returns the gain matrix such that $\mathbf{K} = inv(\mathbf{R})^*(\mathbf{B}'\mathbf{X}+\mathbf{N}')$ . <b>K</b> is a real matrix.
X	Returns the symmetric, positive semi-definite (stabilizing) solution to the discrete algebraic Riccati equation. $X$ is a real matrix.
eig	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>eig</b> is a complex vector.

### Examples

A = [-1, -2; 0, -4] B = [0; 1] Q = [2, 0; 0, 2] R = 1 [K, X, eig] = lqrd(A, B, Q, R)

# **Related Topics**

care dlqr dlqry lqrdy lqr

# lqrdy (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[K, X, eig] = lqrdy(A, B, C, D, Q, R, Ts)

[K, X, eig] = lqrdy(A, B, C, D, Q, R, N, Ts)

[K, X, eig] = lqrdy(SysInSS, Q, R, Ts)

[K, X, eig] = lqrdy(SysInSS, Q, R, N, Ts)

#### Description

Calculates the discrete, optimal, steady-state feedback gain matrix **K** that minimizes a quadratic cost function for a continuous linear time-invariant (LTI) system model in state-space form. This function discretizes automatically the continuous system model before calculating **K**. This cost function weights the model outputs. The quadratic cost function *J* is defined as the following equation:

 $sum(\mathbf{y}'(k)\mathbf{Q}\mathbf{y}(k)+\mathbf{u}(k)'\mathbf{R}\mathbf{u}(k)+2\mathbf{y}(k)'\mathbf{N}\mathbf{u}(k), k, 0, +inf)$ , where *k* is discrete time, **u** is the input vector, and **y** is the output vector.

Examples

## Inputs

Name	Description
Α	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
D	Specifies an $r \ge m$ direct transmission matrix. The default is an empty matrix. <b>D</b> is a real matrix.
Q	Specifies a symmetric, positive semi-definite matrix that penalizes the state vector $\mathbf{x}$ in the cost function. $\mathbf{Q}$ is a real matrix.
R	Specifies a symmetric positive definite matrix that penalizes the input vector <b>u</b> in the cost function. The default is the identity matrix. <b>R</b> is a real matrix.
N	Specifies a matrix that penalizes the cross product between input and state vectors, such that ( <b>Q</b> - <b>N</b> *inv( <b>R</b> )* <b>N</b> ') is positive semi-definite. The default is an appropriately sized matrix of zeros. <b>N</b> is a real matrix.
Ts	Specifies the discrete sampling time this function uses to convert the model. <b>Ts</b> is a real scalar.
SysInSS	Specifies an LTI model in state-space form.

## Outputs

Name	Description
К	Returns the gain matrix such that $\mathbf{K} = inv(\mathbf{R})^*(\mathbf{B}'\mathbf{X}+\mathbf{N}')$ . <b>K</b> is a real matrix.
X	Returns the symmetric, positive semi-definite (stabilizing) solution to the discrete algebraic Riccati equation. $X$ is a real matrix.
eig	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>eig</b> is a complex vector.

### Examples

A = [-1, -2; 0, -4] B = [0; 1] Q = [2, 0; 0, 2] R = 1 [K, X, eig] = lqrdy(A, B, Q, R)

# **Related Topics**

care dlqr dlqry lqrd lqr

# Iqry (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[K, X, eig] = lqry(SysInSS, Q, R)

[K, X, eig] = lqry(SysInSS, Q, R, N)

[K, X, eig] = lqry(A, B, C, D, Q, R)

[K, X,,eig] = lqry(A, B, C, D, Q, R, N)

#### Description

Calculates the optimal steady-state feedback gain matrix **K** that minimizes a quadratic cost function for a linear discrete state-space system model. The cost function weights the model outputs. The quadratic cost function *J* is defined as the following equation: integral( $\mathbf{y}'(t)\mathbf{Q}\mathbf{y}(t)+\mathbf{u}(t)\mathbf{R}\mathbf{u}(t)+2\mathbf{y}(t)\mathbf{N}\mathbf{u}(t)$ , *t*, 0, +inf), where *t* is continuous time, **u** is the input vector, and **y** is the output vector.

**Examples** 

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
Α	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
D	Specifies an $r \ge m$ direct transmission matrix. The default is an empty matrix. <b>D</b> is a real matrix.
Q	Specifies a symmetric, positive semi-definite matrix that penalizes the output vector <b>y</b> in the cost function. <b>Q</b> is a real matrix.
R	Specifies a symmetric positive definite matrix that penalizes the input vector <b>u</b> in the cost function. The default is the identity matrix. <b>R</b> is a real matrix.
N	Specifies a matrix that penalizes the cross product between output and state vectors, such that $(\mathbf{Q}-\mathbf{N}^*\mathrm{inv}(\mathbf{R})^*\mathbf{N}')$ is positive semi-definite. The default is an appropriately sized matrix of zeros. <b>N</b> is a real matrix.

## Outputs

Name	Description
К	Returns the gain matrix such that $\mathbf{K} = inv(\mathbf{R})^*(\mathbf{B}'\mathbf{X}+\mathbf{N}')$ . <b>K</b> is a real matrix.
X	Returns the symmetric, positive semi-definite (stabilizing) solution to the discrete algebraic Riccati equation. $X$ is a real matrix.
eig	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>eig</b> is a complex vector.

#### Examples

A = [0.9, 0.25; 0, 0.8] B = [0; 1] C = [1, 0] D = 0 Q = 2 R = 1 [K, X, eig] = lqry(A, B, C, D, Q, R)

# **Related Topics**

dlqr lqr lqry care dare

# place (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

#### Syntax

[K, q] = place(A, B, p) [K, q] = place(A, B, p, 'K') [K, q] = place(SysInSS, p, 'K') [L, q] = place(A, C, p, 'L') [L, q] = place(SysInSS, p, 'L') [Ltrans, q] = place(A', C', p)

#### Description

Calculates the gain that places the closed-loop poles at specified locations in a system model with full state feedback. You can use this function with multiple-input multiple-output (MIMO) models. Use the <u>acker</u> function for single-input single-output (SISO) models.

**Examples** 

## Inputs

Name	Description
Α	Specifies an $n \ge n$ state matrix, where $n$ is the number of states. The default is an empty matrix. <b>A</b> is a real matrix.
В	Specifies an $n \ge m$ input matrix, where $m$ is the number of inputs. The default is an empty matrix. <b>B</b> is a real matrix.
р	Specifies where you want to place the closed-loop poles. <b>p</b> is a complex vector.
'K'	Specifies that you want this function to design a controller gain vector. <b>'K'</b> is a string constant and is the default value of this argument. You also can enter <b>'c'</b> or <b>'controller'</b> .
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
С	Specifies an $r \ge n$ output matrix, where $r$ is the number of outputs. The default is an empty matrix. <b>C</b> is a real matrix.
'L'	Specifies that you want this function to design an estimator gain vector. <b>'L'</b> is a string constant. You also can enter <b>'o'</b> or <b>'observer'</b> .

## Outputs

Name	Description
К	Returns the feedback controller gain vector that produces a closed-loop model such that the locations of the poles are equal to the values you specified in the <b>p</b> vector. <b>K</b> is a real vector.
q	Returns the eigenvalues of the matrix ( <b>A-BK</b> ). These eigenvalues are the closed-loop pole locations. <b>q</b> is a complex vector.
L	Returns the feedback estimator gain vector that produces a closed-loop model such that the locations of the poles are equal to the values you specified in the <b>p</b> vector. <b>L</b> is a real vector.
Ltrans	Returns the transpose of <b>L</b> . You then can use the <u>transpose</u> function to transpose <b>Ltrans</b> to obtain <b>L</b> . <b>Ltrans</b> is a real vector.

### Examples

A = [-1, 2; 0, -3]B = [0, 1; 1, 0] p = [-4 + i; -4 - i] K = place(A, B, p)

## **Related Topics**

<u>acker</u>

# reg (Control Design and Simulation Module, MathScript Function)

Member of the <u>ssdesign</u> class.

### Syntax

SysRegSS = reg(SysInSS, L, K)

#### Description

Constructs a state regulator based on the estimator gain matrix and controller gain matrix of a state-space system model.

Examples

## Inputs

Name	Description
SysInSS	Specifies a linear time-invariant (LTI) model in state-space form.
L	Specifies the estimator gain matrix. You can obtain $L$ using the <u>acker</u> or <u>place</u> function. $L$ is a real matrix.
К	Specifies the controller gain matrix. You can obtain ${f K}$ using the acker or place function. ${f K}$ is a real matrix.

# Outputs

Name	Description
SysReg	Returns the state-space model of a state regulator.

#### Examples

A = [1, 1; -1, 2]B = [1, 2]' C = [2, 1] SysInSS = ss(A, B, C, 0) L = acker(A', C', [-1, -3])' K = acker(A, B, [-0.5, -0.75]) SysRegSS = reg(SysInSS, K, L)
## **Related Topics**

acker estim lqr place ss

# timeresp (Control Design and Simulation Module, MathScript Class)

Use members of the timeresp class to create generic linear simulations and time domain plots for step inputs, impulse inputs, and initial condition responses.

Function	Description
<u>impulse</u>	Creates the impulse response plot of a system model
<u>initial</u>	Creates the initial response plot of a system model
<u>lsim</u>	Creates the linear simulation plot of a system model
<u>step</u>	Creates the step response plot of a system model

# impulse (Control Design and Simulation Module, MathScript Function)

Member of the <u>timeresp</u> class.

### Syntax

impulse(SysIn, attributes)
impulse(SysIn, attributes, t)
[Y, T, X] = impulse(SysIn)
[Y, T, X] = impulse(SysIn, t)

#### Description

Creates an impulse response plot of a system model. You also can use this function to return the impulse response of the model outputs. If the model is in state-space form, you also can use this function to return the impulse response of the model states. If you do not specify an output, this function creates a plot.

**Examples** 

## Inputs

Name	Des	scription	
SysIn	Spe fune	ecifies a linear time-invariant (LT ction, zero-pole-gain, or state-sp	T) model in transfer bace form.
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies that the plot is blue, marks points with circles, and uses solid lines. <b>attributes</b> is a string that can take a combination of the following values:		
	'b'	Colors the plot blue.	
	'C'	Colors the plot cyan.	
	'g'	Colors the plot green.	
	'k'	Colors the plot black.	
	'm'	Colors the plot magenta.	
	'r'	Colors the plot red.	
	'y'	Colors the plot yellow.	
	'.' '	Marks points with dots.	
	'0'	Marks points with circles.	
	'X'	Marks points with crosses.	
	'+'	Marks points with plus signs.	
	'*'	Marks points with asterisks.	
	'-'	Uses solid lines.	
	'.'	Uses dotted lines.	
	''	Uses dashed and dotted lines.	
	''	Uses dashed lines.	
t	Spe info You the [tf]. a re	ecifies a uniformly spaced time v rmation about the initial time, tir can format <b>t</b> as [t0:dt:tf], where time step, and <i>tf</i> is the final time You also can use the <u>linspace</u> f eal vector.	vector that contains me step, and final time. e <i>t0</i> is the initial time, <i>dt</i> is e. You also can format <b>t</b> as function to generate <b>t</b> . <b>t</b> is

# Outputs

Name	Description
Y	Returns the time response of the outputs of the <b>SysIn</b> model due to an impulse input. <b>Y</b> is a real matrix.
Т	Returns the uniformly spaced time vector this function uses to calculate the impulse response and the state trajectories. <b>T</b> is a real vector.
X	Returns the time response of the states of the <b>SysIn</b> model due to an impulse input. <b>X</b> is a real matrix.

## Examples

SysIn = zpk([1], [-1, -2], 0.5) impulse(SysIn, 'rx-', [0:0.1:10])

## **Related Topics**

<u>initial</u> <u>step</u>

# initial (Control Design and Simulation Module, MathScript Function)

Member of the <u>timeresp</u> class.

### Syntax

initial(SysIn, attributes, x0)
initial(SysIn, attributes, x0, t)
[Y, T, X] = initial(SysIn, x0)
[Y, T, X] = initial(SysIn, x0, t)

#### Description

Creates an initial response initial response plot of a system model. For state-space models, the initial response is output of the model due to the initial states. For transfer function and zero-pole-gain models, this function bases the initial conditions on the model outputs. You also can use this function to return the impulse response of the model outputs. If the model is in state-space form, you also can use this function to return the impulse response of a system to return the impulse response of the model outputs, this function to return the model states. If you do not specify an output, this function creates a plot.

**Examples** 

## Inputs

Name	Des	scription	
SysIn	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.		
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies that the plot is blue, marks points with circles, and uses solid lines. <b>attributes</b> is a string that can take a combination of the following values:		
	'b'	Colors the plot blue.	
	'C'	Colors the plot cyan.	
	'g'	Colors the plot green.	
	'k'	Colors the plot black.	
	'm'	Colors the plot magenta.	
	'r'	Colors the plot red.	
	'y'	Colors the plot yellow.	
	'.' '	Marks points with dots.	
	'0'	Marks points with circles.	
	'X'	Marks points with crosses.	
	'+'	Marks points with plus signs.	
	'*'	Marks points with asterisks.	
	'-'	Uses solid lines.	
	י.י י	Uses dotted lines.	
	''	Uses dashed and dotted lines.	
	''	Uses dashed lines.	
x0	Specifies the initial states of the <b>SysIn</b> model if the <b>SysIn</b> model is in state-space form. If the <b>SysIn</b> model is in transfer function or zero-pole-gain form, <b>x0</b> specifies the initial outputs. <b>x0</b> is a real vector.		
t	Spe	ecifies a uniformly spaced time v	vector that contains

information about the initial time, time step, and final time. You can format **t** as [t0:dt:tf], where *tO* is the initial time, *dt* is the time step, and *tf* is the final time. You also can format **t** as [tf]. You also can use the <u>linspace</u> function to generate **t**. **t** is a real vector.

# Outputs

Name	Description
Y	Returns the time response of the outputs of the SysIn model due to the conditions you specify in the $x0$ input. Y is a real matrix.
Т	Returns the uniformly spaced time vector this function uses to calculate the initial response and the state trajectories. <b>T</b> is a real vector.
X	Returns the time response of the states of the <b>SysIn</b> model due to the conditions you specify in the <b>x0</b> input. This function returns <b>X</b> only if the <b>SysIn</b> model is in state-space form. <b>X</b> is a real matrix.

## Examples

SysIn = zpk([1], [-1, -2], 0.5) y0 = 2 initial(SysIn, 'rx-',y0, [0:0.1:10])

## **Related Topics**

<u>lsim</u> step

# Isim (Control Design and Simulation Module, MathScript Function)

Member of the <u>timeresp</u> class.

#### Syntax

lsim(SysIn, attributes, U, t)

lsim(SysIn, attributes, U, t, x0)

lsim(SysIn, attributes, U, t, method)

lsim(SysIn, attributes, U, t, x0, method)

[Y, T, X] = lsim(SysIn, attributes, U, t)

[Y, T, X] = lsim(SysIn, attributes, U, t, x0)

[Y, T, X] = lsim(SysIn, attributes, U, t, method)

[Y, T, X] = lsim(SysIn, attributes, U, t, x0, method)

#### Description

Creates the linear simulation plot of a system model. This function calculates the output of a system model when a set of inputs excite the model, using discrete simulation. If you do not specify an output, this function creates a plot.

**Examples** 

## Inputs

Name	Des	scription		
SysIn	Spe fune	Specifies a linear time-invariant (LTI) model in transfer function, zero-pole-gain, or state-space form.		
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies that the plot is blue, marks points with circles, and uses solid lines. <b>attributes</b> is a string that can take a combination of the following values:			
	'b'	Colors the plot blue.		
	'C'	Colors the plot cyan.		
	'g'	Colors the plot green.		
	'k'	Colors the plot black.		
	'm'	Colors the plot magenta.		
	'r'	Colors the plot red.		
	'y'	Colors the plot yellow.		
	'.' '.	Marks points with dots.		
	'0'	Marks points with circles.		
	'X'	Marks points with crosses.		
	'+'	Marks points with plus signs.		
	'*'	Marks points with asterisks.		
	'-'	Uses solid lines.		
	י.י י	Uses dotted lines.		
	''	Uses dashed and dotted lines.		
	''	Uses dashed lines.		
U	Specifies the inputs that excite the <b>SysIn</b> model. If the <b>SysIn</b> model is discrete, ensure that the time step of the <b>SysIn</b> model matches the time step of the <b>U</b> input. <b>U</b> is a real matrix.			
t	Spe	ecifies a uniformly spaced time v	vector that contains	

	nformation about the initial time, time step, and final time. You can format <b>t</b> as [t0:dt:tf], where <i>t0</i> is the initial time, <i>dt</i> is he time step, and <i>tf</i> is the final time. You also can format <b>t</b> as tf]. You also can use the <u>linspace</u> function to generate <b>t</b> . <b>t</b> is a real vector.
x0	Specifies the initial states of the <b>SysIn</b> model. The <i>i</i> -th element of this input corresponds to the <i>i</i> -th initial state. The default value is all zeros. <b>x0</b> is a real vector.
method	Specifies the method this function uses to convert the <b>SysIn</b> rom continuous to discrete representation. <b>method</b> is a string that can take the following values:
	zoh' (Default) Converts the model using the Zero-Order- Hold method.
	foh' Converts the model using the First-Order-Hold method.

# Outputs

Name	Description
Y	Returns the time response of the outputs of the SysIn model due to the conditions you specify in the <b>x0</b> input. <b>Y</b> is a real matrix.
Т	Returns the uniformly spaced time vector this function uses to calculate the output response and the state trajectories. <b>T</b> is a real vector.
X	Returns the time response of the states of the <b>SysIn</b> model due to the conditions you specify in the <b>x0</b> input. <b>X</b> is a real matrix.

### Examples

z = [1] p = [-1, -2] k = 0.5SysIn = zpk(z, p, k) t = [0:0.1:10] u = sin(0.1\*pi\*t)'lsim(SysIn, u, t)

## **Related Topics**

<u>initial</u>

# step (Control Design and Simulation Module, MathScript Function)

Member of the <u>timeresp</u> class.

### Syntax

step(SysIn, attributes)
step(SysIn, attributes, t)
[Y, T, X] = step(SysIn)
[Y, T, X] = step(SysIn, t)

#### Description

Creates a step response plot of the system model. You also can use this function to return the step response of the model outputs. If the model is in state-space form, you also can use this function to return the step response of the model states. This function assumes the initial model states are zero. If you do not specify an output, this function creates a plot.

**Examples** 

## Inputs

Name	Des	scription	
SysIn	Spe zer	ecifies a linear time-invariant mo p-pole-gain, or state-space form	odel in transfer function, n.
attributes	Specifies valid plot attributes. Order the plot attributes by color, point-style, and line-style. For example, 'bo-' specifies that the plot is blue, marks points with circles, and uses solid lines. <b>attributes</b> is a string that can take a combination of the following values:		
	'b'	Colors the plot blue.	
	'C'	Colors the plot cyan.	
	'g'	Colors the plot green.	
	'k'	Colors the plot black.	
	'm'	Colors the plot magenta.	
	'r'	Colors the plot red.	
	'y'	Colors the plot yellow.	
	'.' '.	Marks points with dots.	
	'0'	Marks points with circles.	
	'X'	Marks points with crosses.	
	'+'	Marks points with plus signs.	
	'*'	Marks points with asterisks.	
	'-'	Uses solid lines.	
	י.י י	Uses dotted lines.	
	''	Uses dashed and dotted lines.	
	''	Uses dashed lines.	
t	Spe info You the [tf]. a re	ecifies a uniformly spaced time v rmation about the initial time, tir can format <b>t</b> as [t0:dt:tf], where time step, and <i>tf</i> is the final time You also can use the <u>linspace</u> f eal vector.	vector that contains me step, and final time. e <i>t0</i> is the initial time, <i>dt</i> is e. You also can format <b>t</b> as function to generate <b>t</b> . <b>t</b> is

# Outputs

Name	Description
Y	Returns the time response of the outputs of the SysIn model due to the step input. Y is a real matrix.
Т	Returns the uniformly spaced time vector this function uses to calculate the step response and the state trajectories. <b>T</b> is a real vector.
X	Returns the time response of the states of the SysIn model due to the step input. This function returns X only if the SysIn model is in state-space form. X is a real matrix.

## Examples

n = 4 r = 2 m = 2 SysIn = rss(n, r, m) step(SysIn, [0:0.01:100])

## **Related Topics**

impulse initial Isim