tf Construct transfer function or convert to transfer function.

Construction:

SYS = tf(NUM,DEN) creates a continuous-time transfer function SYS with numerator NUM and denominator DEN. SYS is an object of type tf when NUM,DEN are numeric arrays, of type GENSS when NUM,DEN depend on tunable parameters (see REALP and GENMAT), and of type USS when NUM,DEN are uncertain (requires Robust Control Toolbox).

SYS = tf(NUM,DEN,TS) creates a discrete-time transfer function with sample time TS (set TS=-1 if the sample time is undetermined).

S = tf('s') specifies the transfer function H(s) = s (Laplace variable).
Z = tf('z',TS) specifies H(z) = z with sample time TS.
You can then specify transfer functions directly as expressions in S or Z, for example,

s = tf('s'); H = exp(-s)*(s+1)/(s^2+3*s+1)

SYS = tf creates an empty tf object.
SYS = tf(M) specifies a static gain matrix M.

You can set additional model properties by using name/value pairs.
For example,
sys = tf(1,[1 2 5],0.1,'Variable','q','ioDelay',3)
also sets the variable and transport delay. Type "properties(tf)" for a complete list of model properties, and type help tf.<PropertyName>
for help on a particular property. For example, "help tf.Variable" provides information about the "Variable" property.

By default, transfer functions are displayed as functions of 's' or 'z'. Alternatively, you can use the variable 'p' in continuous time and the variables 'z^-1', 'q', or 'q^-1' in discrete time by modifying the "Variable" property.

Data format:
For SISO models, NUM and DEN are row vectors listing the numerator and denominator coefficients in descending powers of s,p,z,q or in ascending powers of z^-1 (DSP convention). For example,
sys = tf([1 2],[1 0 10])
specifies the transfer function (s+2)/(s^2+10) while
sys = tf([1 2],[1 5 10],0.1,'Variable','z^-1')
specifies (1 + 2 z^-1)/(1 + 5 z^-1 + 10 z^-2).

For MIMO models with NY outputs and NU inputs, NUM and DEN are NY-by-NU cell arrays of row vectors where
NUM{i,j} and DEN{i,j} specify the transfer function from input j to output i. For example,
H = tf( [-5 ; [1 -5 6]], {{1 -1} ; [1 1 0]})
specifies the two-output, one-input transfer function

[ -5 /(s-1) ]
[ (s^2-5s+6)/(s^2+s) ]

Arrays of transfer functions:
You can create arrays of transfer functions by using ND cell arrays for NUM and DEN above. For example, if NUM and DEN are cell arrays of size [NY NU 3 4], then

SYS = tf(NUM,DEN) creates the 3-by-4 array of transfer functions SYS(:,:,k,m) = tf(NUM(:,:,k,m),DEN(:,:,k,m)), k=1:3, m=1:4. Each of these transfer functions has NY outputs and NU Inputs.

SYS = tf(ZEROS([NY NU k1 k2...])) is used to pre-allocate an array of zero transfer functions with NY outputs and NU Inputs.
Conversion:

SYS = tf(SYS) converts any dynamic system SYS to the transfer function representation. The resulting SYS is always of class tf.

See also tf/exp, filt, tfdata, zpk, ss, frd, genss, USS, DynamicSystem.

Overloaded methods:
DynamicSystem/tf
mfilt.tf
adaptfilt.tf
StaticModel/tf
dfilt.tf

Reference page in Help browser
doc tf
**step**  Step response of dynamic systems.

\[ [Y,T] = \text{step}(SYS) \]  computes the step response \( Y \) of the dynamic system \( SYS \). The time vector \( T \) is expressed in the time units of \( SYS \) and the time step and final time are chosen automatically. For multi-input systems, independent step commands are applied to each input channel. If \( SYS \) has \( NY \) outputs and \( NU \) inputs, \( Y \) is an array of size \( [\text{LENGTH}(T) \ NY \ NU] \) where \( Y(:,;:;j) \) contains the step response of the \( j \)-th input channel.

For state-space models,

\[ [Y,T,X] = \text{step}(SYS) \]

also returns the state trajectory \( X \), an array of size \( [\text{LENGTH}(T) \ NX \ NU] \) for a system with \( NX \) states and \( NU \) inputs.

For identified models (see IDLTI and IDNLMODEL),

\[ [Y,T,X,YSD] = \text{step}(SYS) \]

also computes the standard deviation \( YSD \) of the response \( Y \) (\( YSD \) is empty if \( SYS \) does not contain parameter covariance information).

\[ [Y,...] = \text{step}(SYS,TFINAL) \]  simulates the step response from \( t=0 \) to the final time \( t=\text{TFINAL} \) (expressed in the time units of \( SYS \)). For discrete-time models with unspecified sample time, \( \text{TFINAL} \) is interpreted as the number of sampling periods.

\[ [Y,...] = \text{step}(SYS,T) \]  specifies the time vector \( T \) for simulation (in the time units of \( SYS \)). For discrete-time models, \( T \) should be of the form \( 0:T_{s}:T_{f} \) where \( T_{s} \) is the sample time. For continuous-time models, \( T \) should be of the form \( 0:dT:T_{f} \) where \( dT \) is the sampling period for the discrete approximation of \( SYS \).

\[ [Y,...] = \text{step}(SYS,...,OPTIONS) \]  specifies additional options such as the step amplitude or input offset.

Use \text{stepDataOptions} to create the option set \( OPTIONS \).

When called without output arguments, \( \text{step}(SYS,...) \) plots the step response of \( SYS \) and is equivalent to \( \text{STEPPLT}(SYS,...) \). See \text{STEPPLT} for additional graphical options for step response plots.

See also \text{stepplot}, \text{stepDataOptions}, \text{impulse}, \text{initial}, \text{lsim}, \text{Itiview}, \text{DynamicSystem}, \text{IDLTI}.
**impulse**  
Impulse response of dynamic systems.

**impulse(SYS)** plots the impulse response of the dynamic system SYS. For systems with more than one input, independent impulse commands are applied to each input channel. The time range and number of points are chosen automatically. For continuous-time systems with direct feedthrough, the infinite pulse at t=0 is ignored.

**impulse(SYS,TFINAL)** simulates the impulse response from t=0 to the final time t=TFINAL (expressed in the time units specified in SYS.TimeUnit). For discrete-time models with unspecified sample time, TFINAL is interpreted as the number of sampling periods.

**impulse(SYS,T)** uses the time vector T for simulation (expressed in the time units of SYS). For discrete-time models, T should be of the form Ti:Ts:Tf where Ts is the sample time. For continuous-time models, T should be of the form Ti:dt:Tf where dt is the sampling period for the discrete approximation of SYS. The impulse is always assumed to arise at t=0 (regardless of Ti).

**impulse(SYS1,SYS2,...,T)** plots the impulse response of several systems SYS1,SYS2,... on a single plot. The time vector T is optional. You can also specify a color, line style, and marker for each system, for example:

```
impulse(sys1,'r',sys2,'y--',sys3,'gx')
```

**[Y,T] = impulse(SYS)** When invoked with left-hand arguments, returns the output response Y and the time vector T used for simulation. No plot is drawn on the screen. If SYS has NY outputs and NU inputs, and LT=length(T), Y is an array of size [LT NY NU] where Y(:,:,j) gives the impulse response of the j-th input channel. The time vector T is expressed in the time units of SYS.

**[Y,T,X] = impulse(SYS, ...)** For state-space models, also returns the state trajectory X which is an LT-by-NX-by-NU array if SYS has NX states.

**[Y,T,X,YSD] = impulse(SYS)** The response uncertainty computation returns the standard deviation YSD of the response Y of an identified system SYS. YSD is empty if SYS does not contain parameter covariance information.

See IMPULSEPLOT for additional options for impulse response plots.

**Note:** In discrete time, impulse computes the response to a unit-area pulse of length Ts and height 1/Ts where Ts is the sample time. This pulse approaches the continuous-time Dirac impulse delta(t) as Ts goes to zero.

See also impulseplot, step, initial, lsim, ltiview, DynamicSystem.
lsim  Simulate time response of dynamic systems to arbitrary inputs.

lsim(SYS,U,T) plots the time response of the dynamic system SYS to the input signal described by U and T. The time vector T is expressed in the time units of SYS and consists of regularly spaced time samples. The matrix U has as many columns as inputs in SYS and its i-th row specifies the input value at time T(i). For example,

\[ t = 0:0.01:5; \quad u = \sin(t); \quad \text{lsim}(sys,u,t) \]

simulates the response of a single-input model SYS to the input \( u(t) = \sin(t) \) during 5 time units.

For discrete-time models, U should be sampled at the same rate as SYS (T is then redundant and can be omitted or set to the empty matrix). For continuous-time models, choose the sampling period T(2)-T(1) small enough to accurately describe the input U. lsim issues a warning when U is undersampled and hidden oscillations may occur.

lsim(SYS,U,T,X0) specifies the initial state vector X0 at time T(1) (for state-space models only). X0 is set to zero when omitted.

lsim(SYS1,SYS2,...,U,T,X0) simulates the response of several systems SYS1, SYS2,... on a single plot. The initial condition X0 is optional. You can also specify a color, line style, and marker for each system, for example

\[ \text{lsim}(sys1,'r',sys2,'y--',sys3,'gx',u,t). \]

Y = lsim(SYS,U,T) returns the output history Y. No plot is drawn on the screen. The matrix Y has LENGTH(T) rows and as many columns as outputs in SYS.

For state-space models,

\[ [Y,T,X] = \text{lsim}(SYS,U,T,X0) \]
also returns the state trajectory X, a matrix with LENGTH(T) rows and as many columns as states.

For continuous-time models,

\[ \text{lsim}(SYS,U,T,X0,'zoh') \quad \text{or} \quad \text{lsim}(SYS,U,T,X0,'foh') \]
explicitly specifies how the input values should be interpolated between samples (zero-order hold or linear interpolation). By default, lsim selects the interpolation method automatically based on the smoothness of the signal U.

See LSIMPLOT for additional graphical options for lsim plots.

See also lsimplot, gensig, step, impulse, initial, DynamicSystem.
**initial**  Initial condition response of state-space models.

**initial(SYS,X0)** plots the undriven response of the state-space model SYS (created with SS) with initial condition X0 on the states. This response is characterized by the equations

Continuous time:  \( x = A x , \ y = C x , \ x(0) = x0 \)

Discrete time:  \( x[k+1] = A x[k], \ y[k] = C x[k], \ x[0] = x0 \).

The time range and number of points are chosen automatically.

**initial(SYS,X0,TFINAL)** simulates the time response from \( t=0 \) to the final time \( t=\text{TFINAL} \) (expressed in the time units specified in SYS.TimeUnit). For discrete-time models with unspecified sample time, \( \text{TFINAL} \) is interpreted as the number of sampling periods.

**initial(SYS,X0,T)** uses the time vector \( T \) for simulation (expressed in the time units of SYS). For discrete-time models, \( T \) should be of the form \( Ti:Ts:Tf \) where \( Ts \) is the sample time. For continuous-time models, \( T \) should be of the form \( Ti:dt:Tf \) where \( dt \) is the sampling period for the discrete approximation of SYS.

**initial(SYS1,SYS2,...,X0,T)** plots the response of several systems SYS1,SYS2,... on a single plot. The time vector \( T \) is optional. You can also specify a color, line style, and marker for each system, for example:

```
initial(sys1,'r',sys2,'y--',sys3,'gx',x0).
```

When invoked with left hand arguments,

\[
[Y,T,X] = \text{initial}(SYS,X0)
\]

returns the output response \( Y \), the time vector \( T \) used for simulation, and the state trajectories \( X \). No plot is drawn on the screen. The matrix \( Y \) has \( \text{LENGTH}(T) \) rows and as many columns as outputs in SYS. Similarly, \( X \) has \( \text{LENGTH}(T) \) rows and as many columns as states. The time vector \( T \) is expressed in the time units of SYS.

See also initialplot, impulse, step, lsim, ltiview, DynamicSystem.