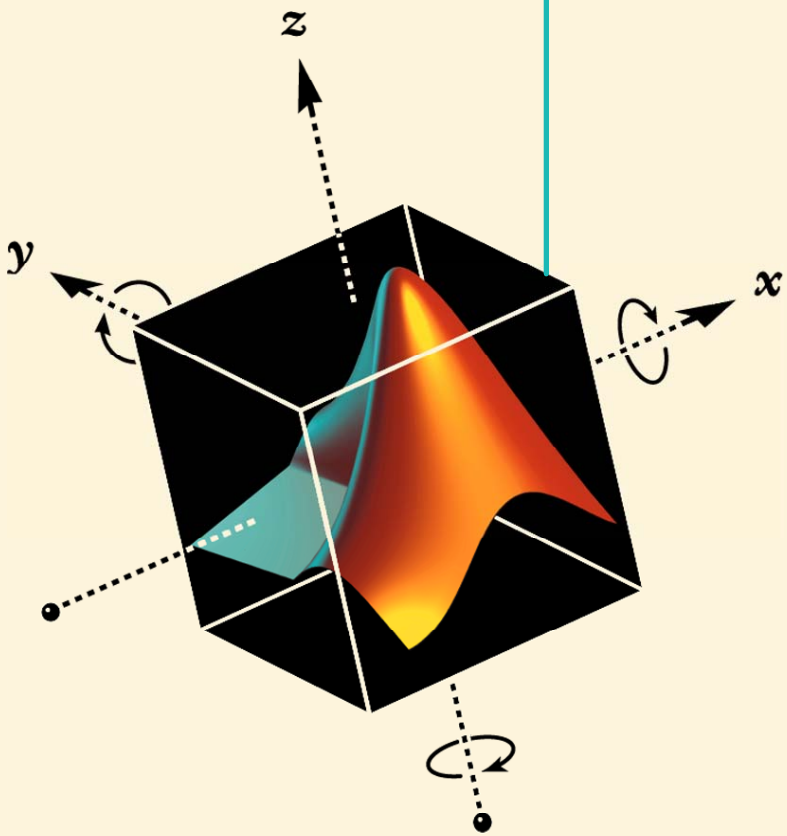
Introduction to Simulink

>> 1997

>> matlab conference



Larry Michaels

michaels@mathworks.com

The MathWorks Inc.

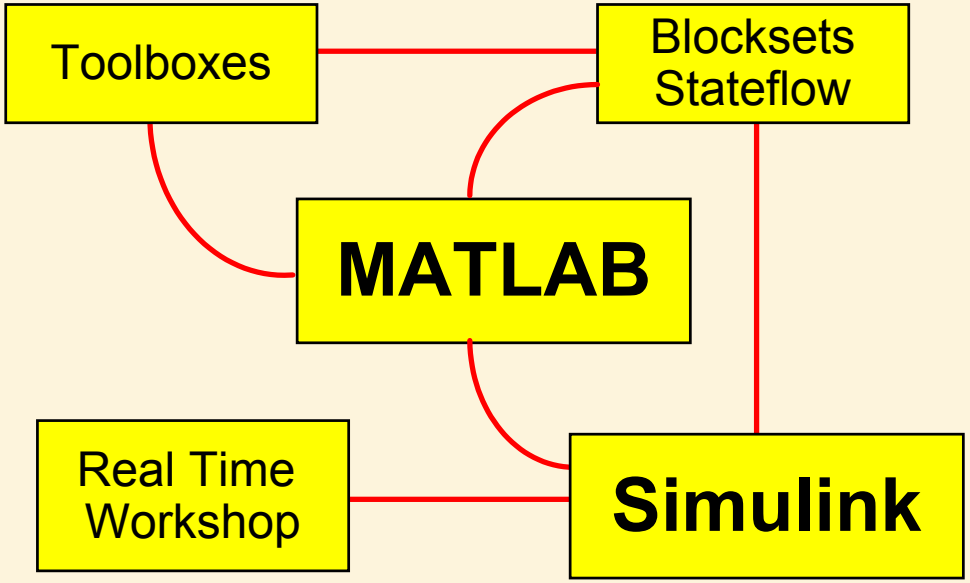
24 Prime Park Way

Natick, MA 01760-1500

USA

What Is Simulink?

Simulink is an interactive, block-diagram-based tool for modeling and analyzing dynamic systems. It is tightly coupled with MATLAB and supported by Blocksets and Extensions.



Copyright © 1984 - 1997 by The MathWorks, Inc.

What Is Simulink?

A simulation tool in MATLAB's Technical Computing Environment

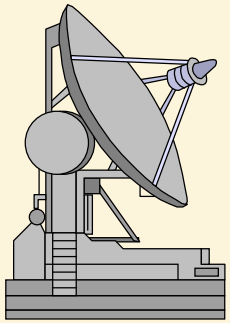
Among other things, this means

- **Block diagram editing**
- **Nonlinear simulation**
- **Continuous and discrete simulation**
- **Asynchronous (non-uniform sampling) simulation**
- **Integration with MATLAB, Extensions, Blocksets & Toolboxes**

Copyright © 1984 - 1997 by The MathWorks, Inc.

What can you model with Simulink?

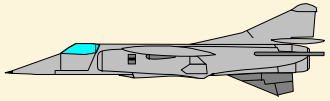
Just about anything you can model mathematically



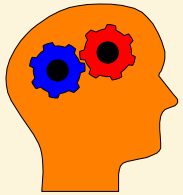
Communications and satellite systems

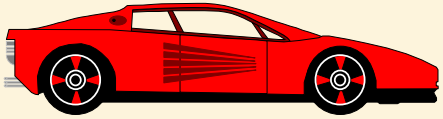
Ship systems

Aircraft and spacecraft dynamics and systems



Biological systems





Automotive systems

Monetary systems

Copyright © 1984 - 1997 by The MathWorks, Inc.

Building & Running a Model

- Creating a model
- Running the simulation
- Analyzing the results

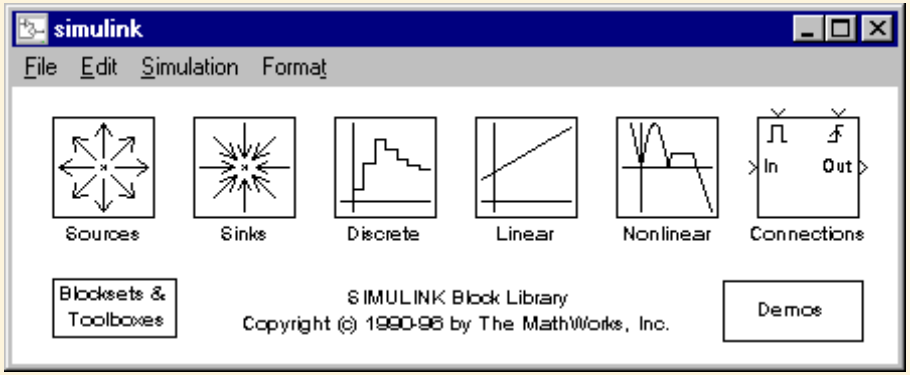
Copyright © 1984 - 1997 by The MathWorks, Inc.

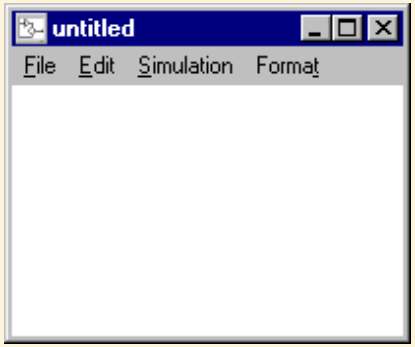
Creating a model

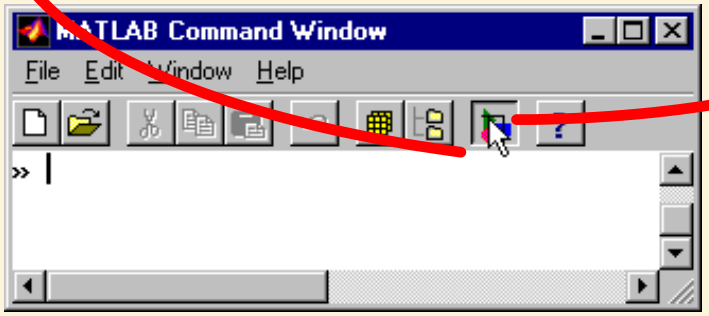
- Editing models
- Annotations
- Setting block dialog parameters
- Creating Subsystems
- Signal Labels
- Saving (MDL File format)

Copyright © 1984 - 1997 by The MathWorks, Inc.

Open a new model window





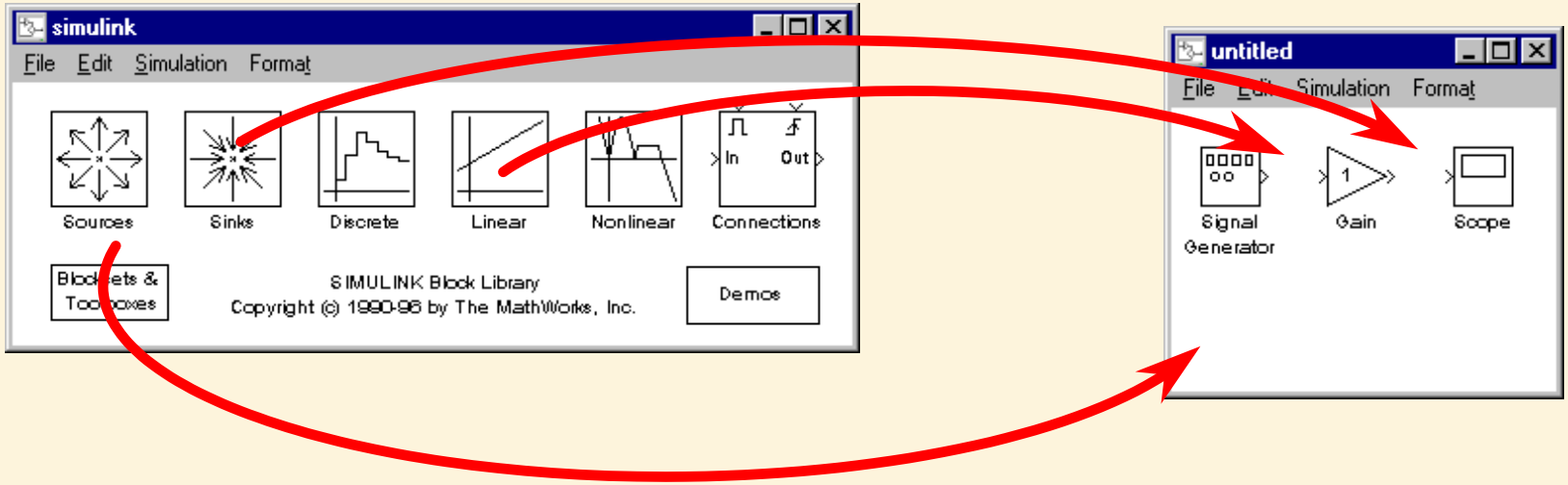


Typing `simulink` will also open a new model window if no other model is open

```
» simulink
```

Copyright © 1984 - 1997 by The MathWorks, Inc.

Collecting blocks

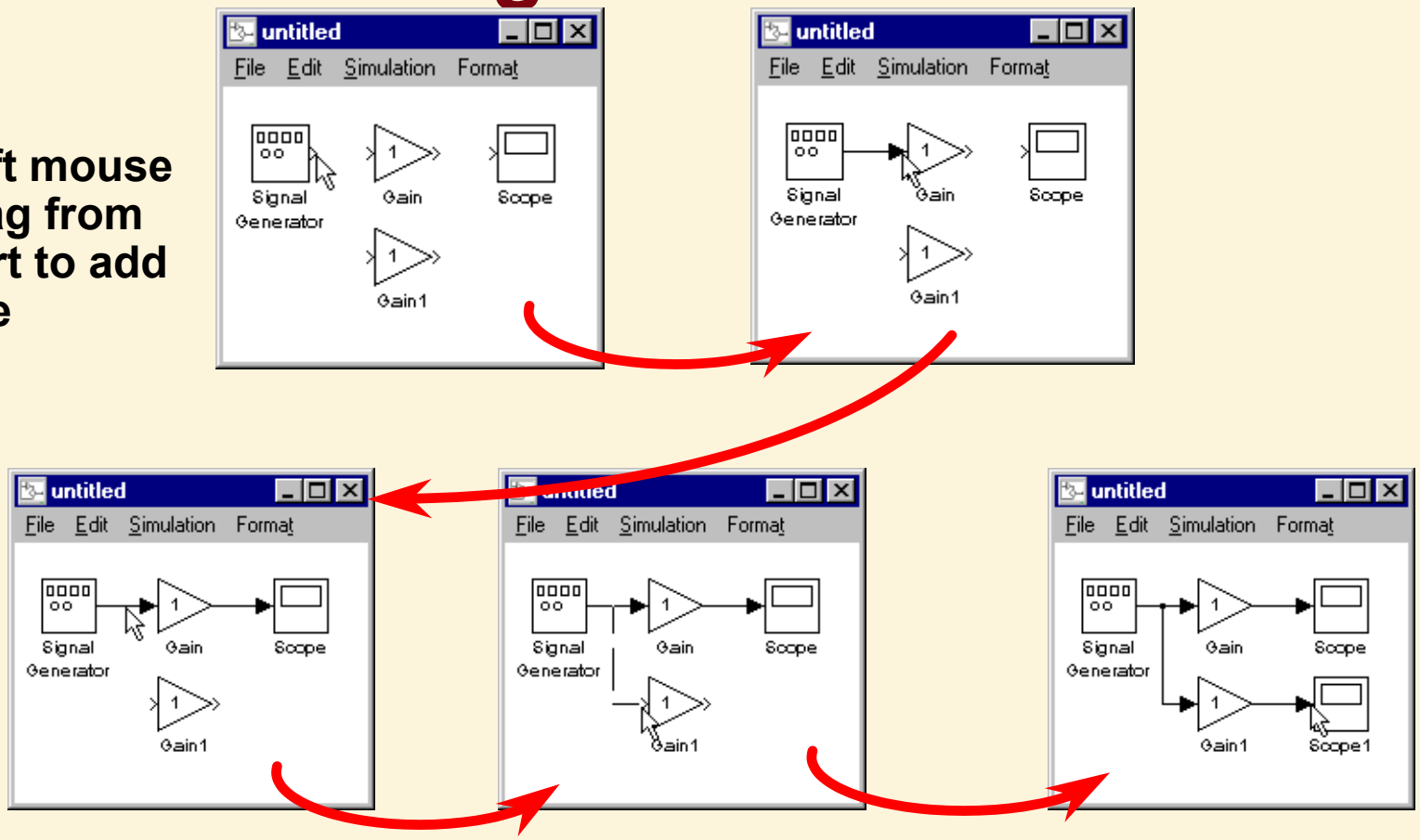


Pull the required blocks into a new SIMULINK block diagram window.

Copyright © 1984 - 1997 by The MathWorks, Inc.

Connecting blocks

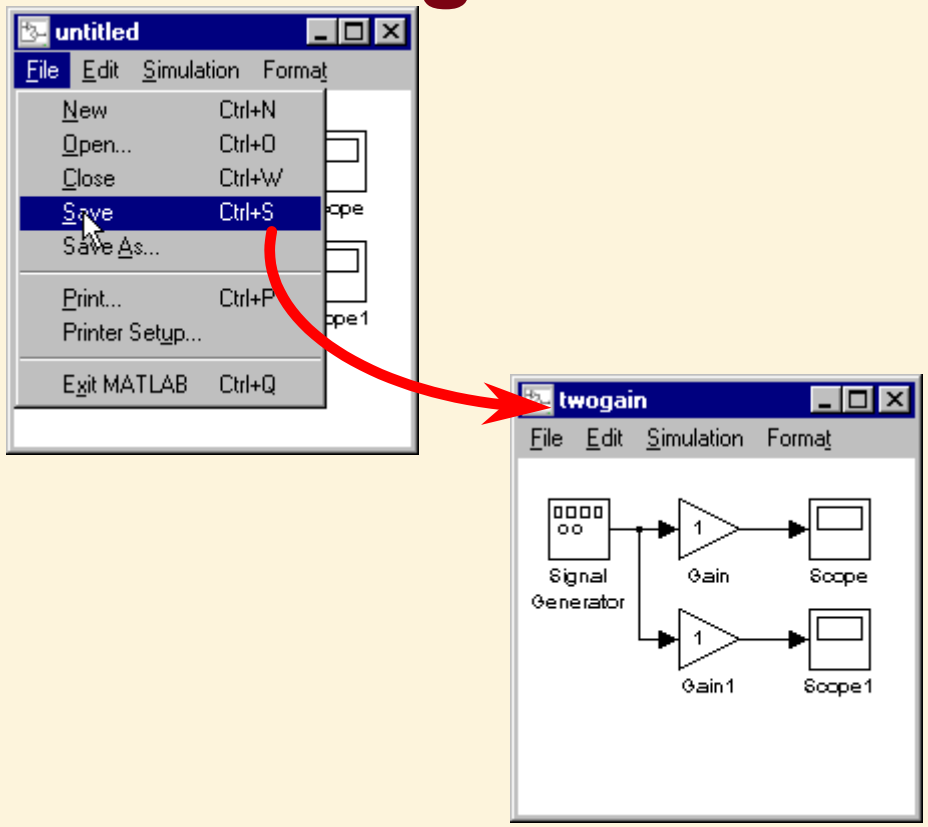
Left mouse drag from port to add line



Right mouse drag to add branch line

Copyright © 1984 - 1997 by The MathWorks, Inc.

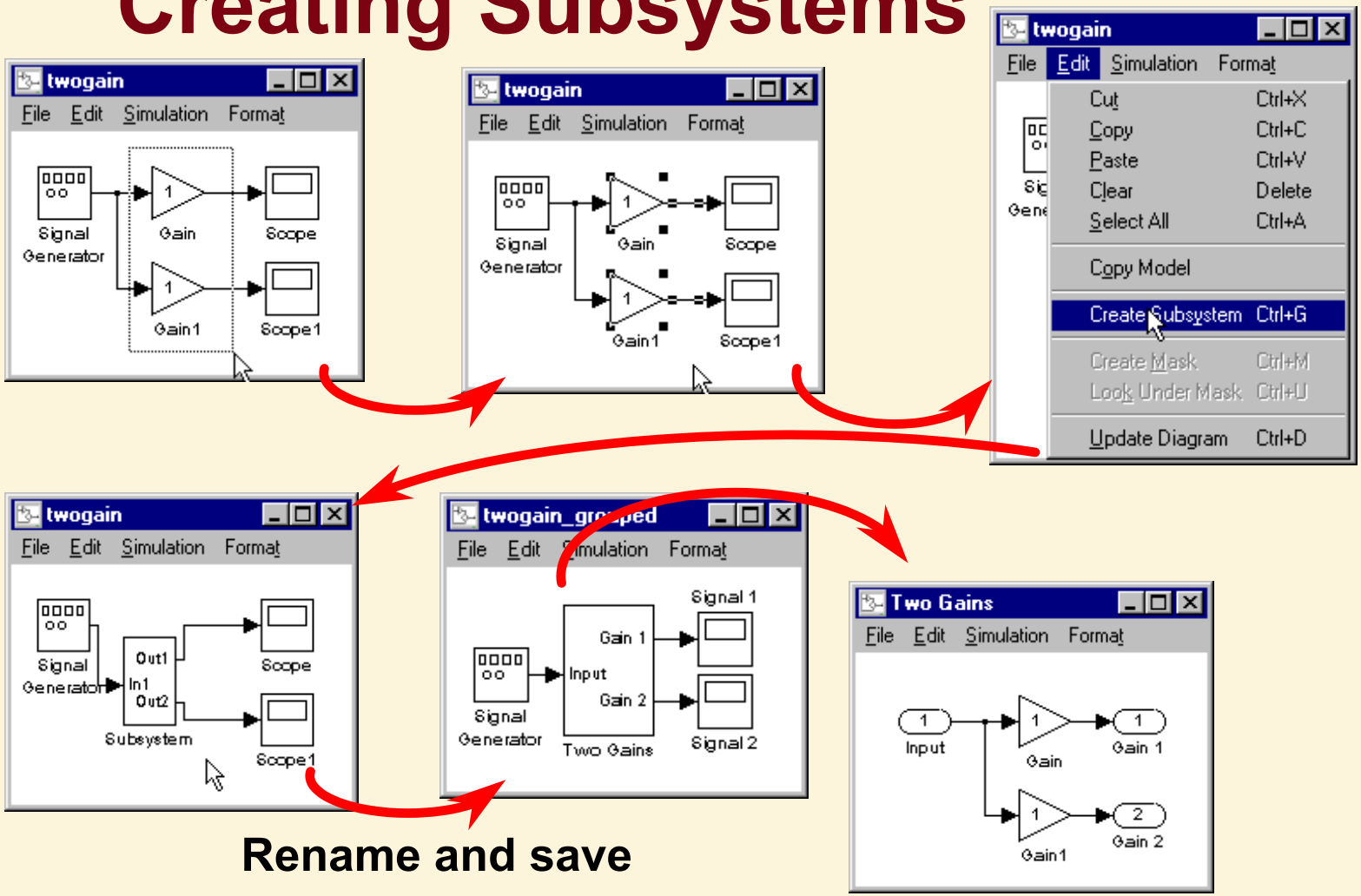
Saving




```
Model {
  BlockDefaults {
  }
  AnnotationDefaults {
  }
  System {
    Block
    }
    Line {
      Branch {
      }
    }
  }
}
```

Copyright © 1984 - 1997 by The MathWorks, Inc.

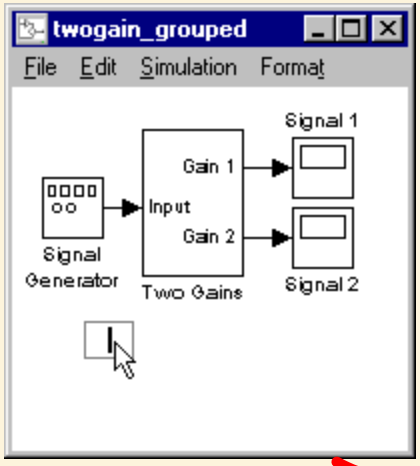
Creating Subsystems



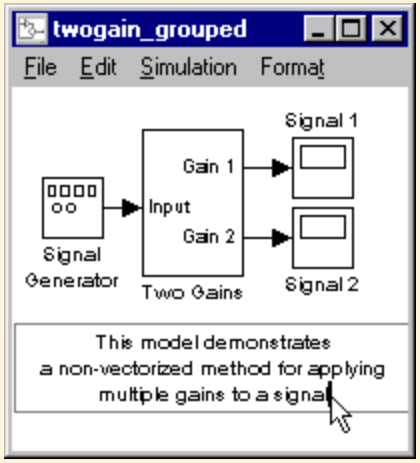
Rename and save

Copyright © 1984 - 1997 by The MathWorks, Inc.

Annotations



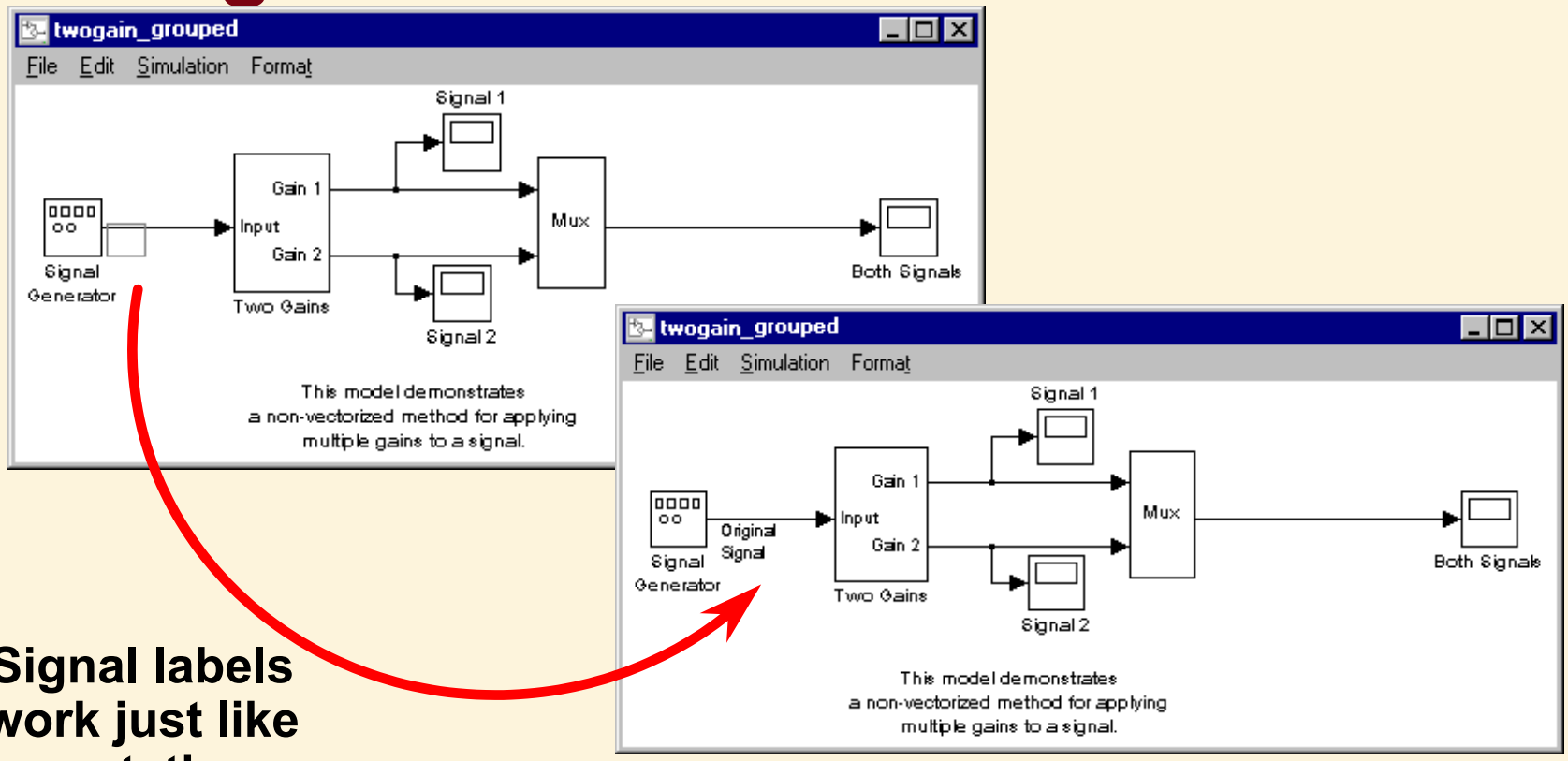
Double click on background



Type in the text

Copyright © 1984 - 1997 by The MathWorks, Inc.

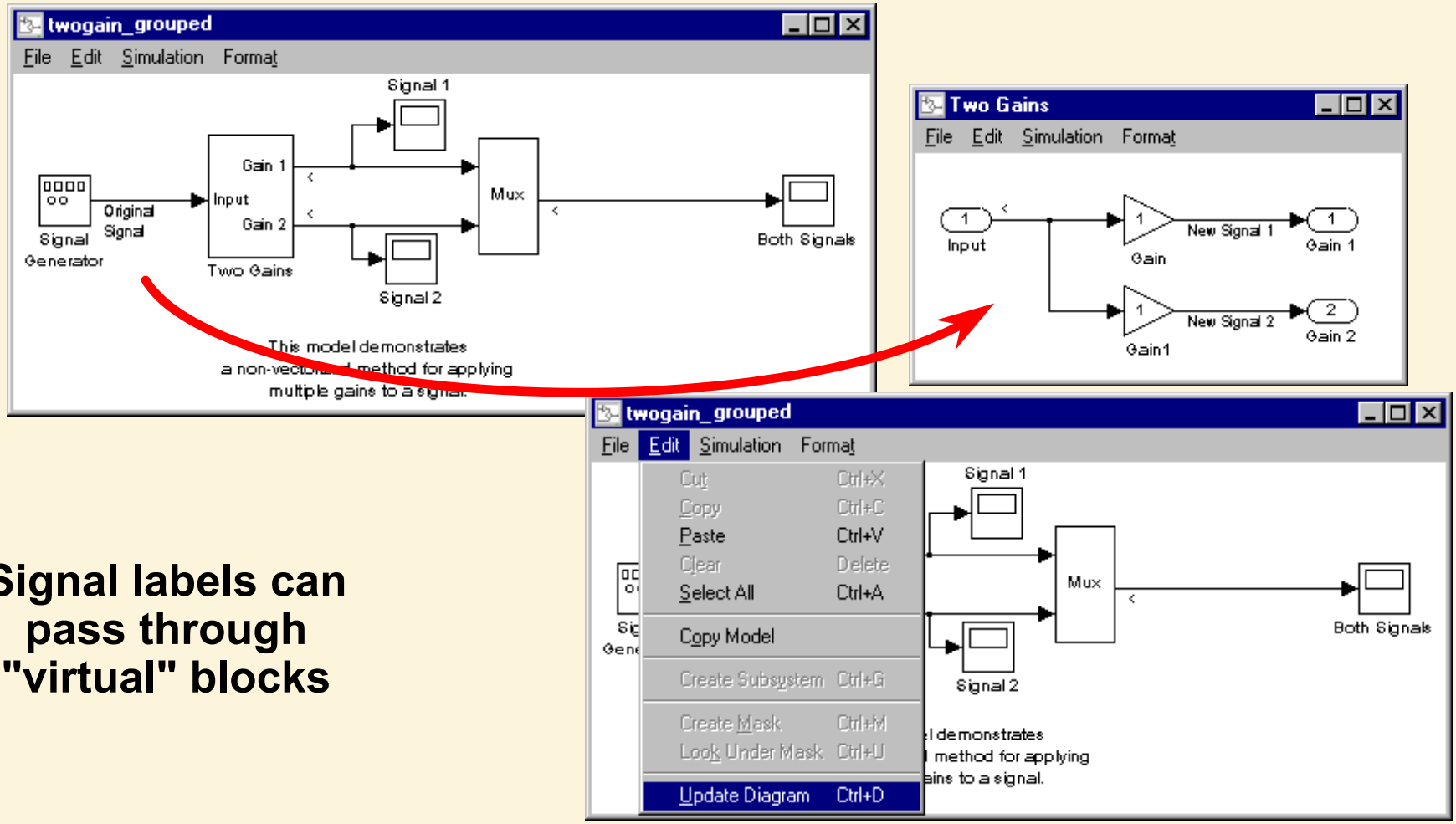
Signal labels



Signal labels work just like annotations

Copyright © 1984 - 1997 by The MathWorks, Inc.

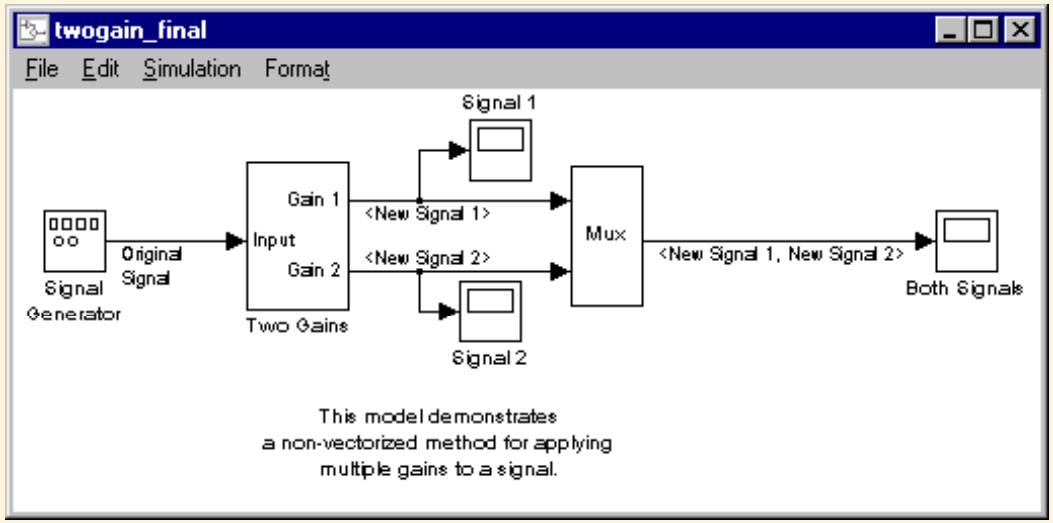
Signal labels

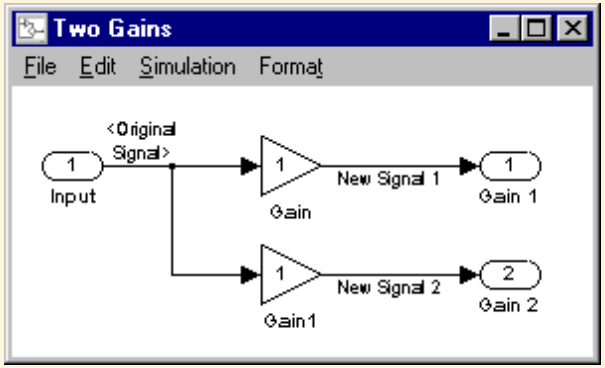


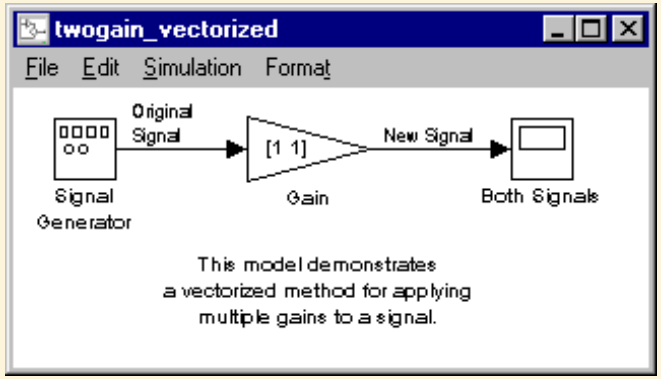
Signal labels can pass through "virtual" blocks

Copyright © 1984 - 1997 by The MathWorks, Inc.

Signal labels







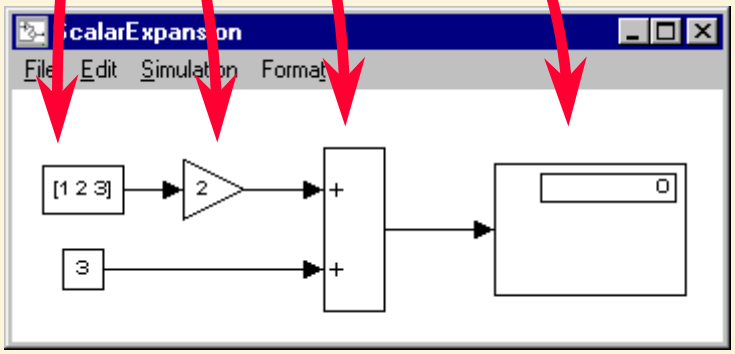
Copyright © 1984 - 1997 by The MathWorks, Inc.

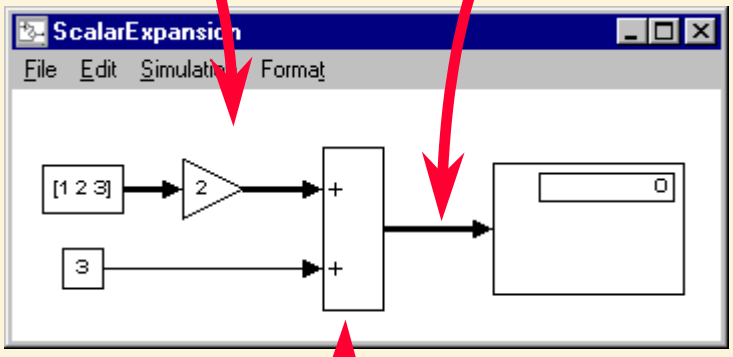
Vector I/O and Scalar Expansion

Vectorized Inputs and Outputs

Scalar expansion of parameter

Wide Vector Lines



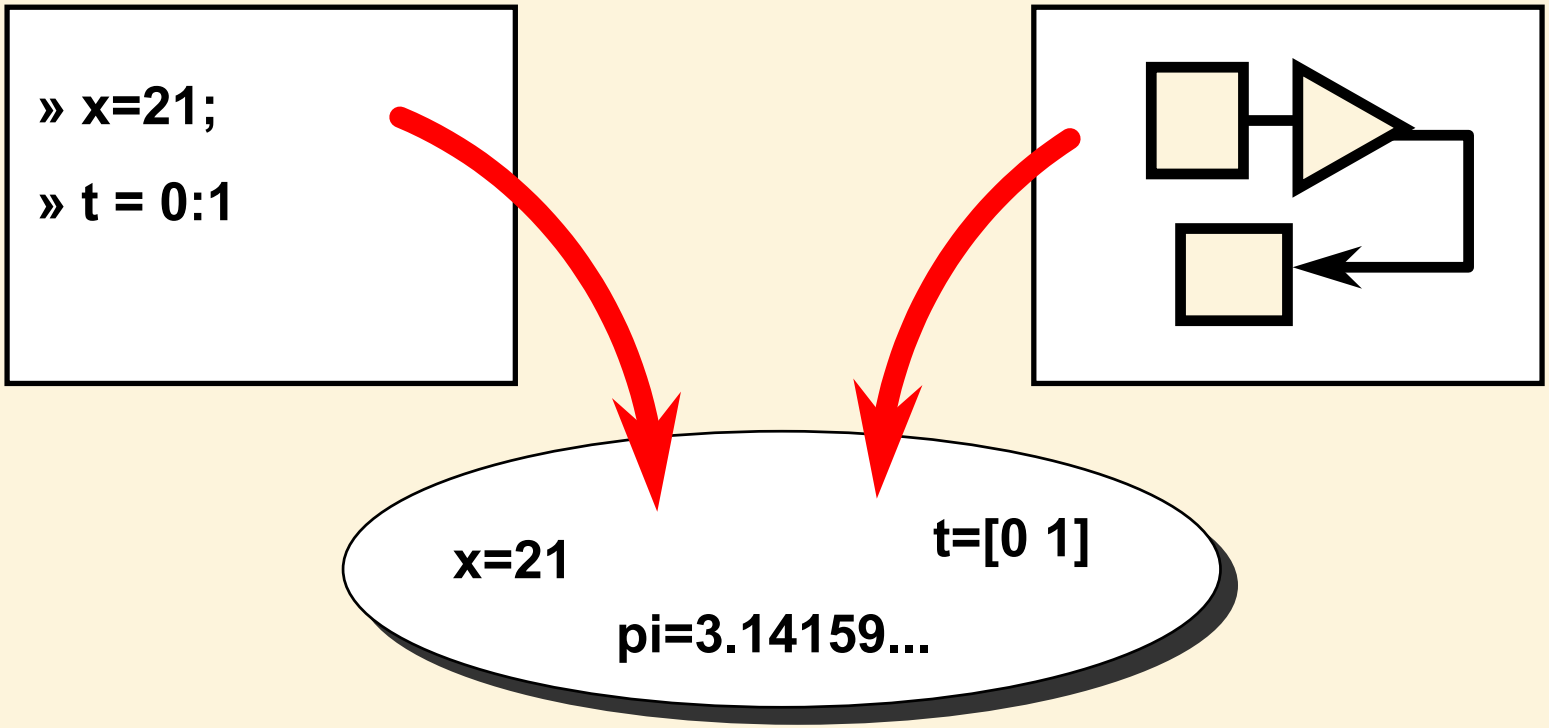


Scalar expansion of inputs

Copyright © 1984 - 1997 by The MathWorks, Inc.

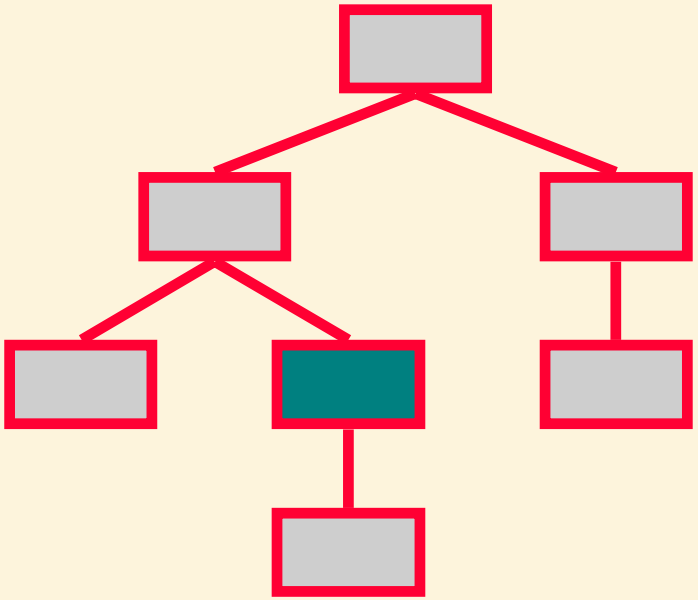
Defining variables

Both MATLAB and Simulink Windows "See" Into the Same Workspace.

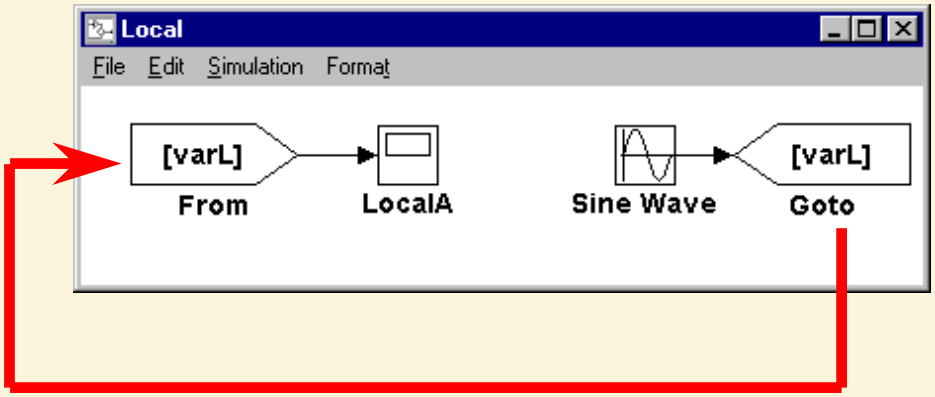


Copyright © 1984 - 1997 by The MathWorks, Inc.

Goto / From blocks

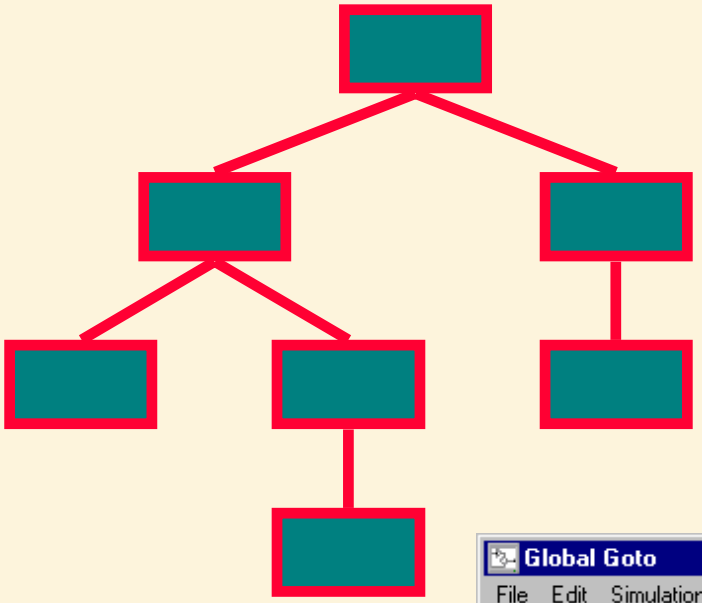


Local [ ] is only on the current window

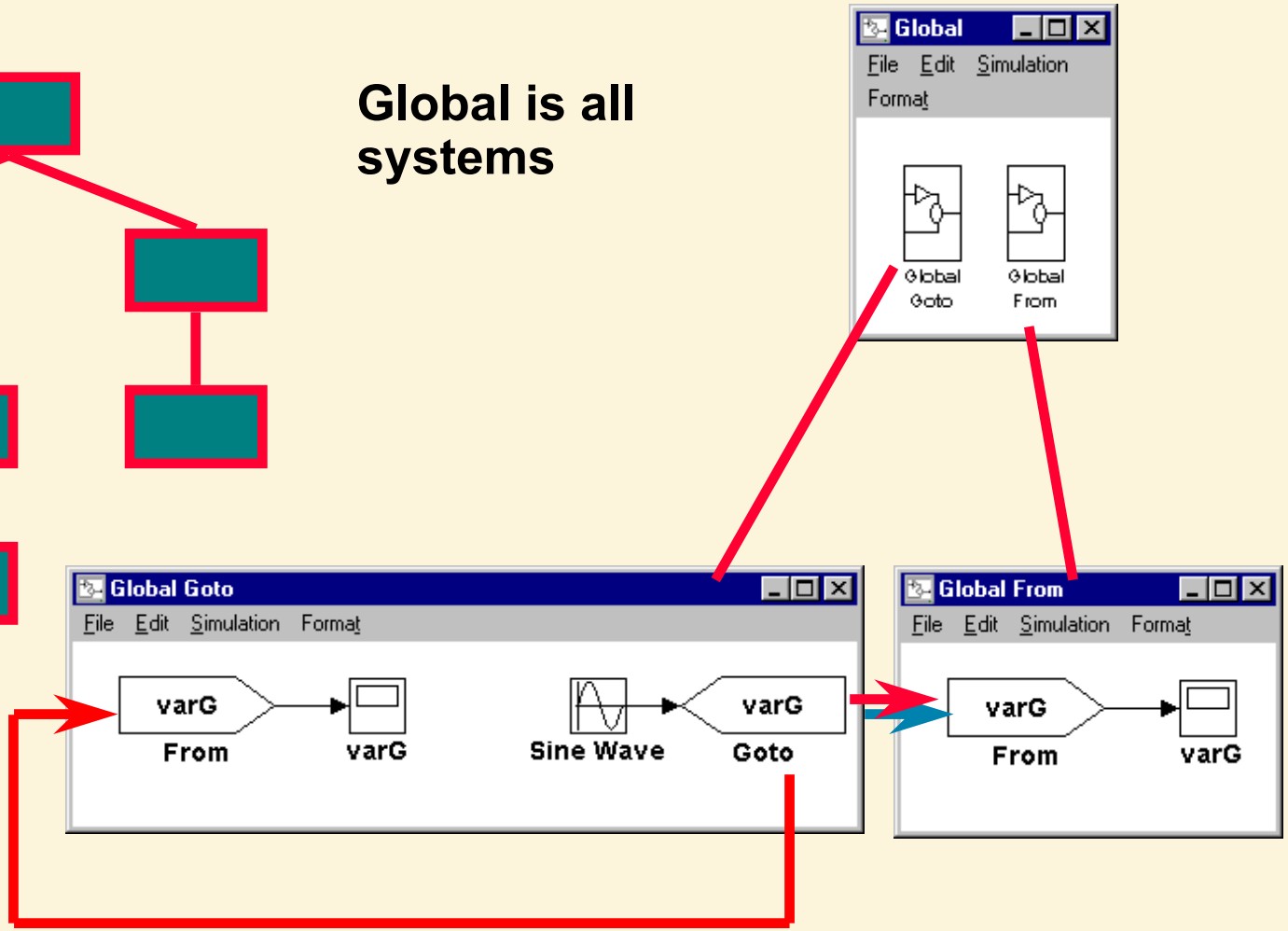


Copyright © 1984 - 1997 by The MathWorks, Inc.

Goto / From blocks



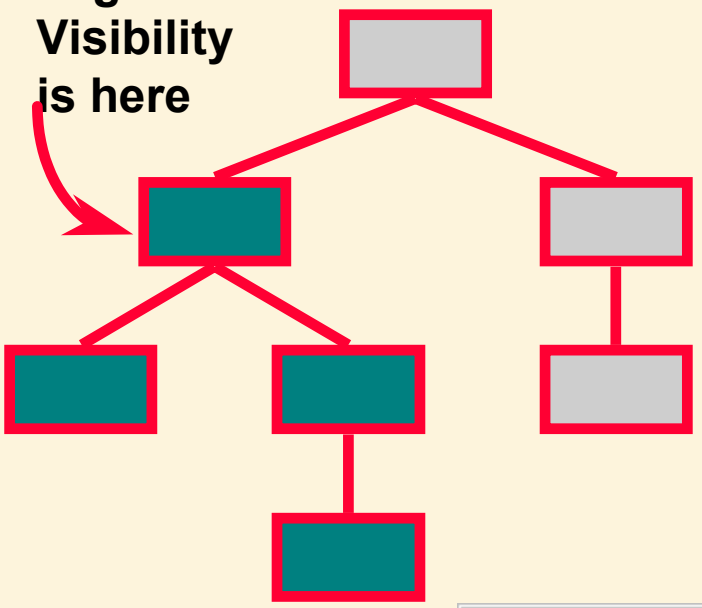
Global is all systems



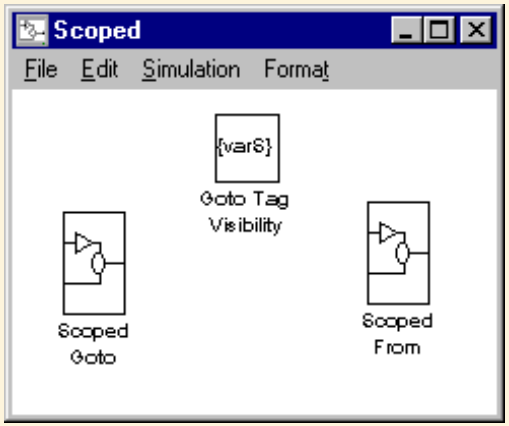
Copyright © 1984 - 1997 by The MathWorks, Inc.

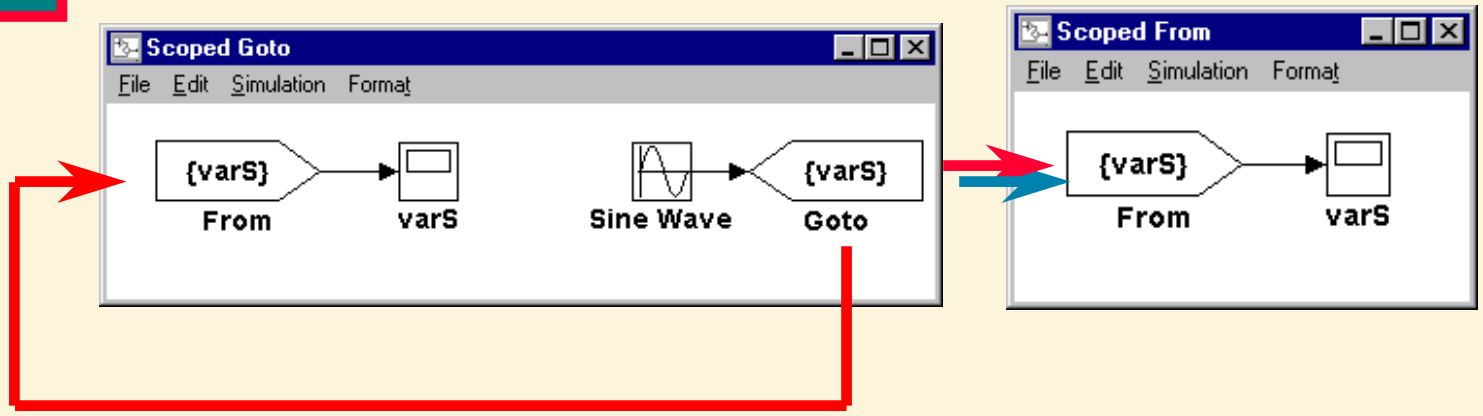
Goto / From blocks

Tag Visibility is here



Scoped {} is all systems beneath the Tag Visibility block





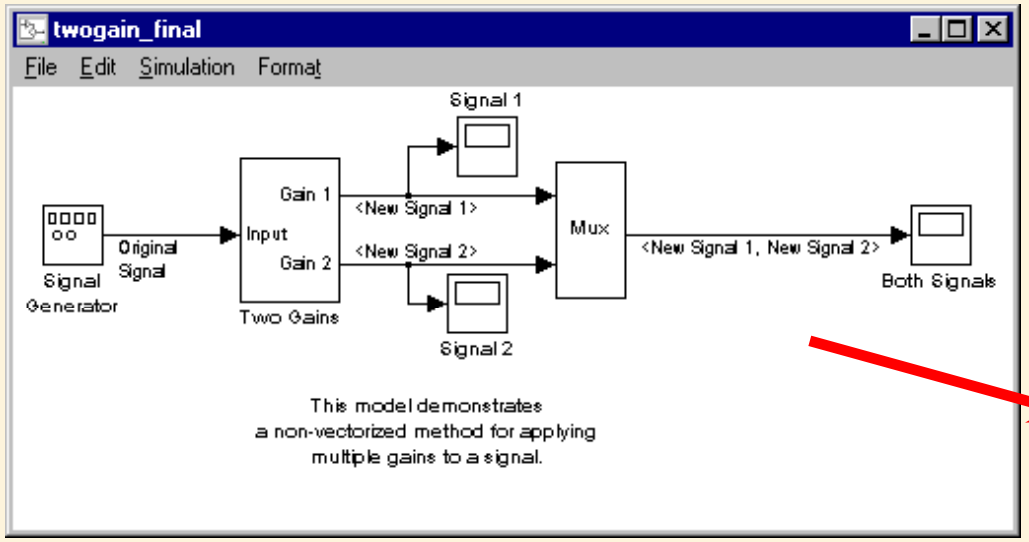
Copyright © 1984 - 1997 by The MathWorks, Inc.

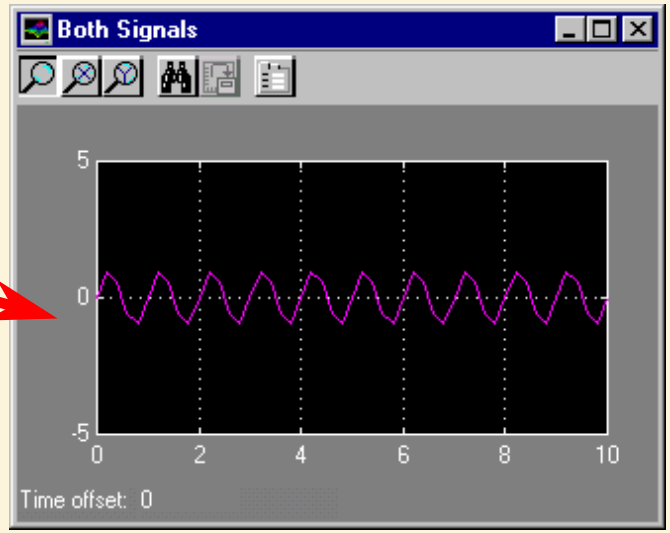
Running a simulation

- Parameters dialog box
- Continuous systems
- Discrete systems
- Getting data into models
- From the command line

Copyright © 1984 - 1997 by The MathWorks, Inc.

Simulating

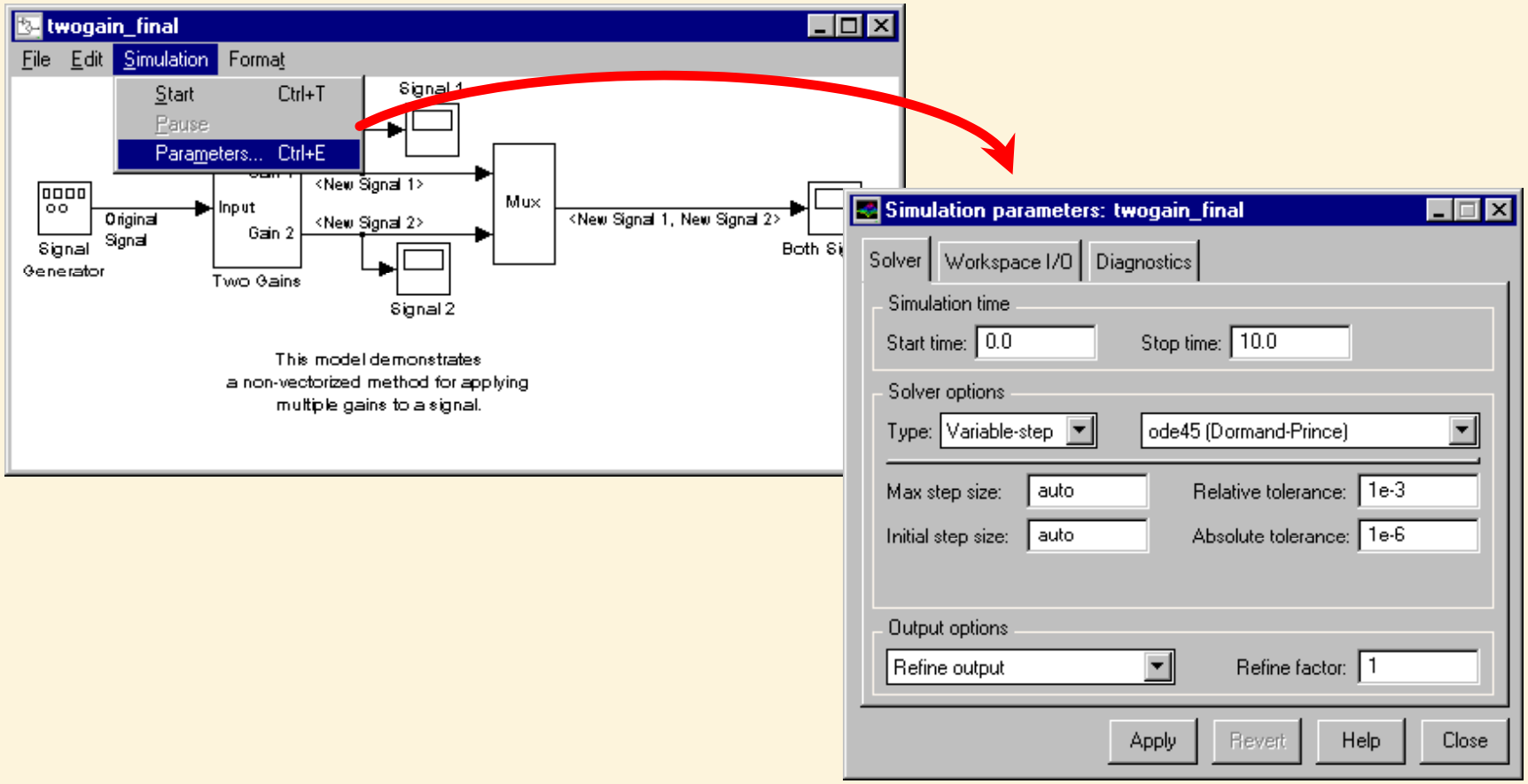




First simulation: jagged!

Copyright © 1984 - 1997 by The MathWorks, Inc.

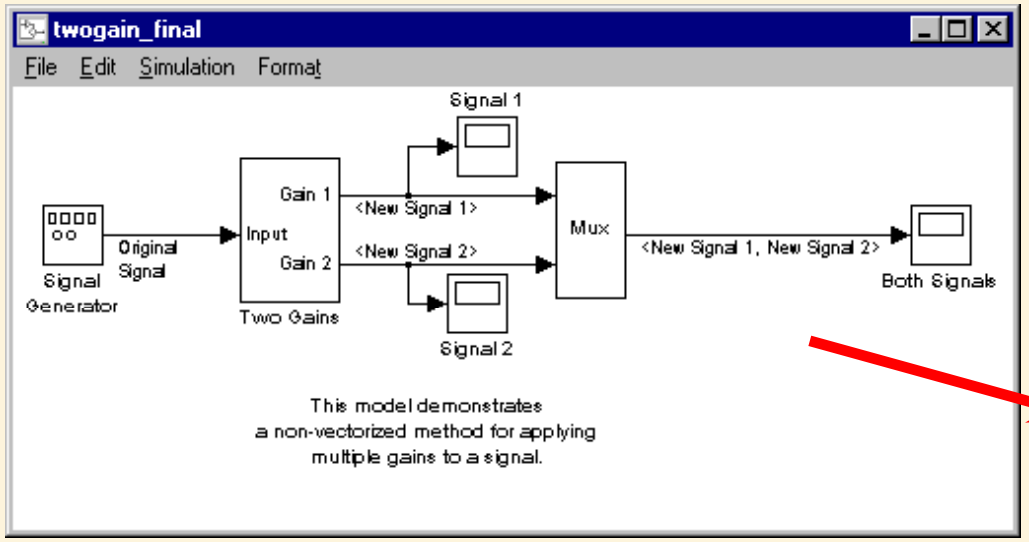
Setting parameters

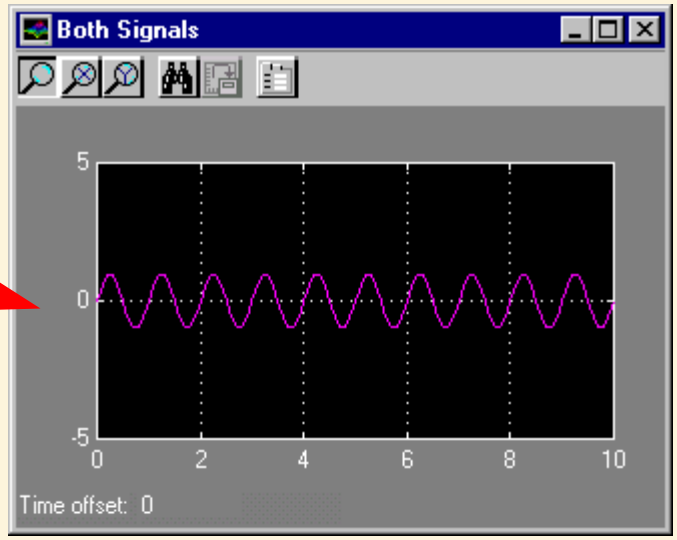


Copyright © 1984 - 1997 by The MathWorks, Inc.

Simulating

Second Simulation: Much Better...

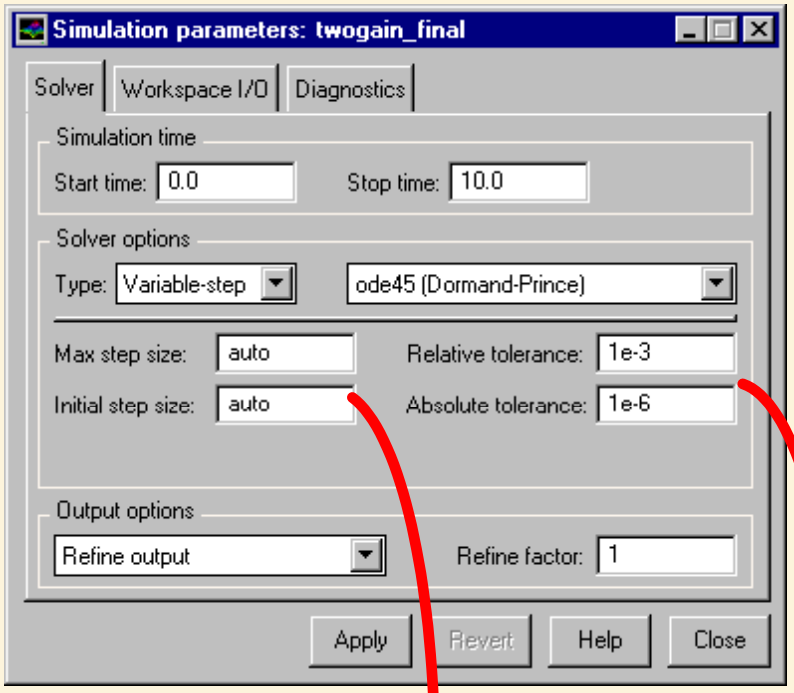




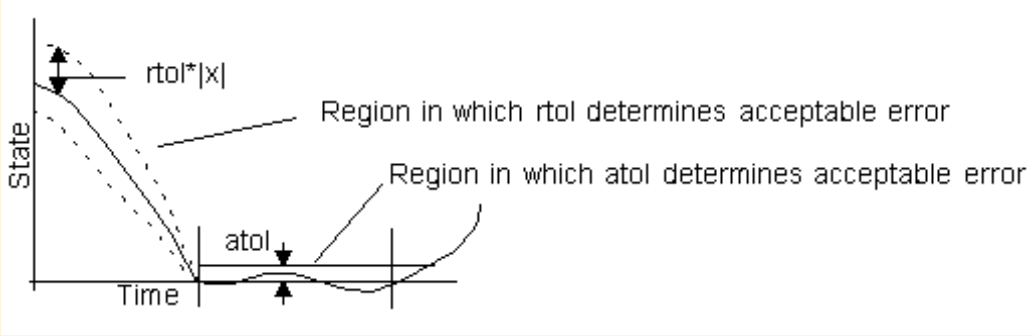
Use:
Variable-step discrete time solver
Max. Step Size = 0.1

Copyright © 1984 - 1997 by The MathWorks, Inc.

Setting parameters



$$h_{\max} = \frac{t_{stop} - t_{start}}{50}$$



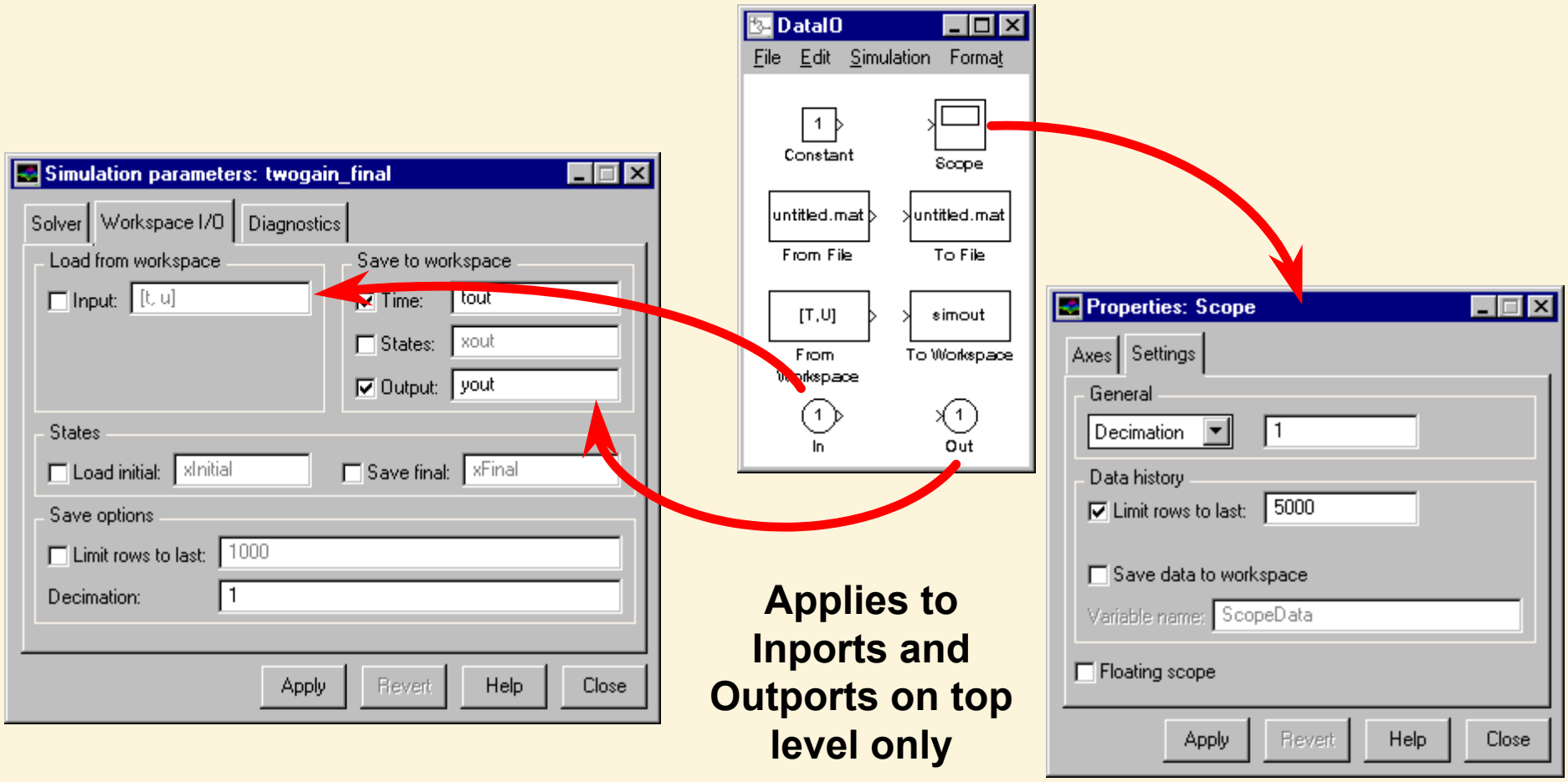
Copyright © 1984 - 1997 by The MathWorks, Inc.

Available Solvers

- **Variable-step solvers**
  - Modify step size during simulation
  - Error control and zero crossing detection
  - ode45, ode23, ode113, ode15s, ode23s, discrete
- **Fixed-step solvers**
  - Same step size throughout simulation
  - No error control or zero crossing detection
  - ode5, ode4, ode3, ode2, ode1, discrete
- **Simulink selects a default solver**
  - ode45 used for models with continuous states
  - Variable-step discrete-time solver used for models with no continuous states

Copyright © 1984 - 1997 by The MathWorks, Inc.

Getting data into and out of models



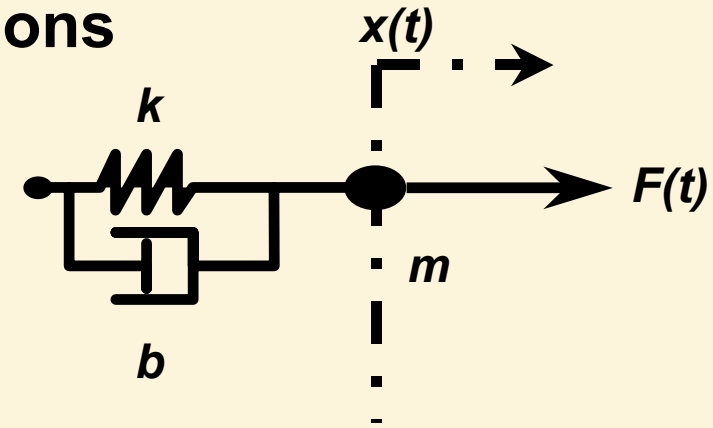
Applies to Inports and Outports on top level only

Copyright © 1984 - 1997 by The MathWorks, Inc.

Continuous models

Ordinary differential equations

$$F = m\ddot{x} + b\dot{x} + kx$$

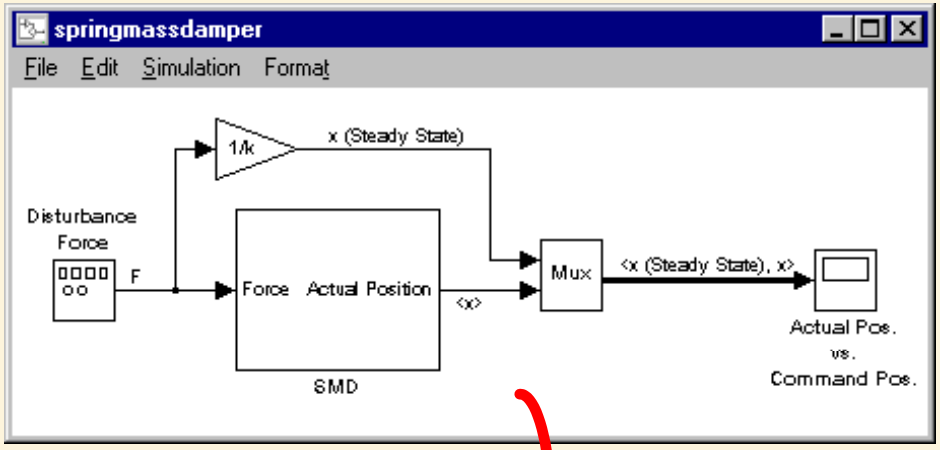


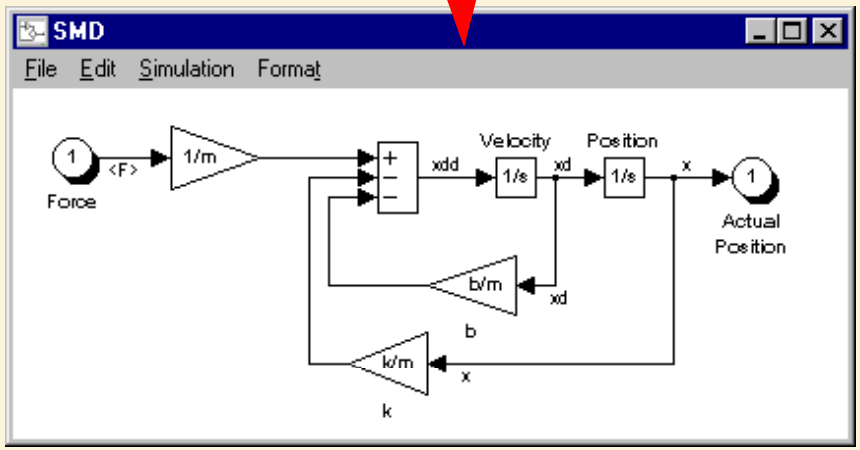
$$\frac{x}{F} = \frac{y}{u} = \frac{1}{ms^2 + bs + k}$$

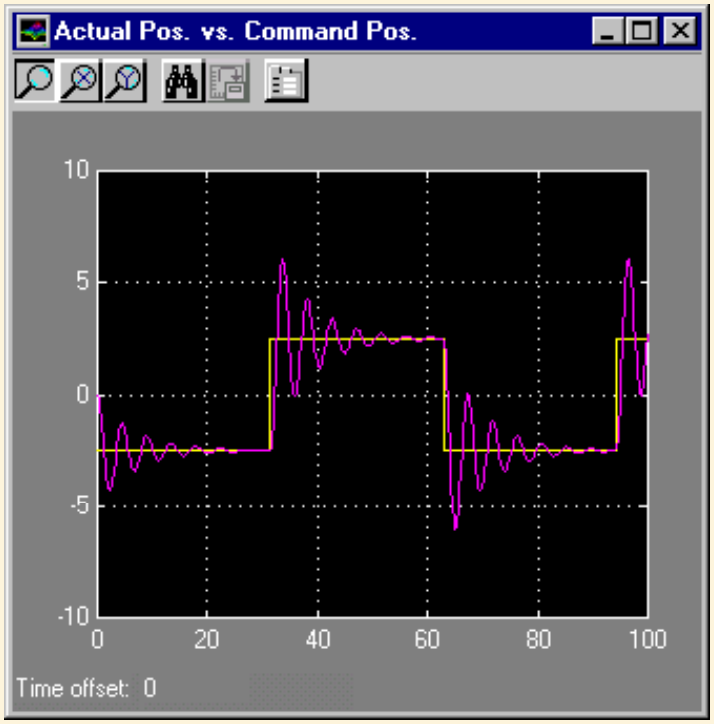
[m b k]

Copyright © 1984 - 1997 by The MathWorks, Inc.

Spring-Mass-Damper

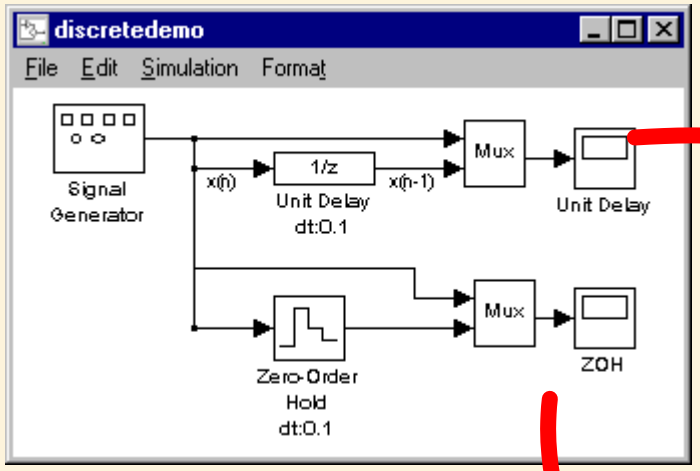


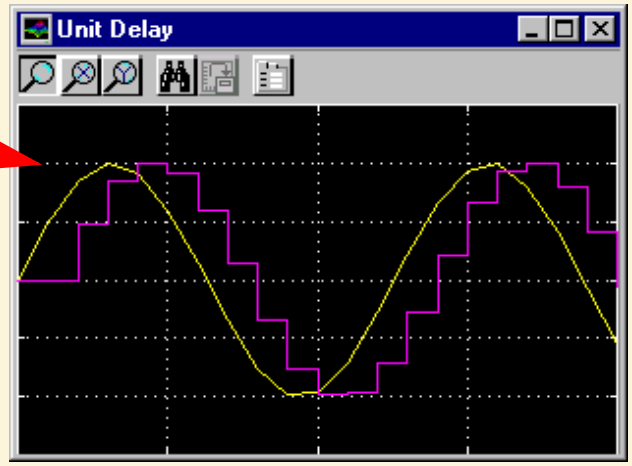


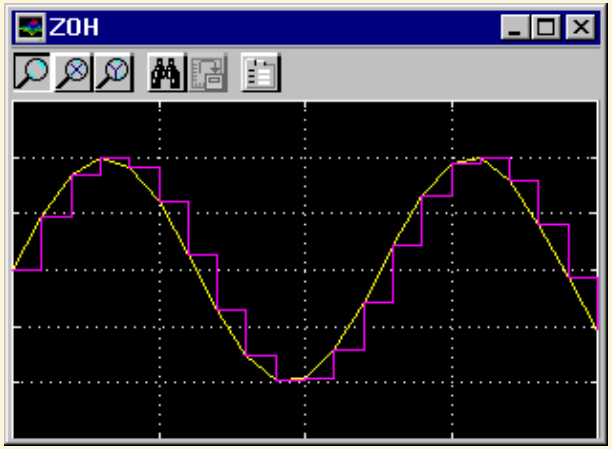


Copyright © 1984 - 1997 by The MathWorks, Inc.

Discrete models







For discrete models, use a fixed-step integrator

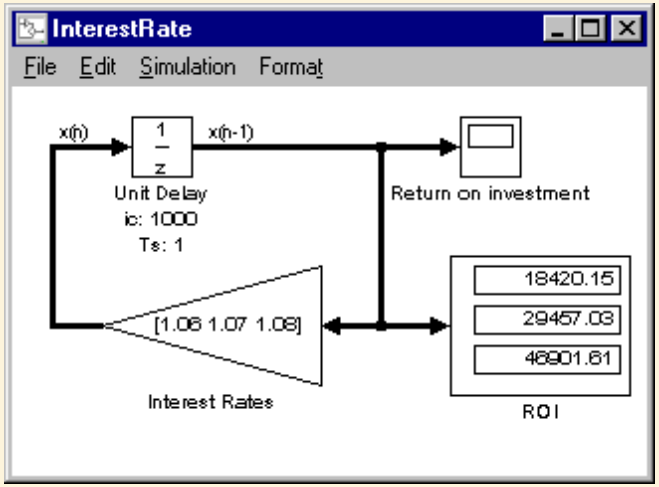
Copyright © 1984 - 1997 by The MathWorks, Inc.

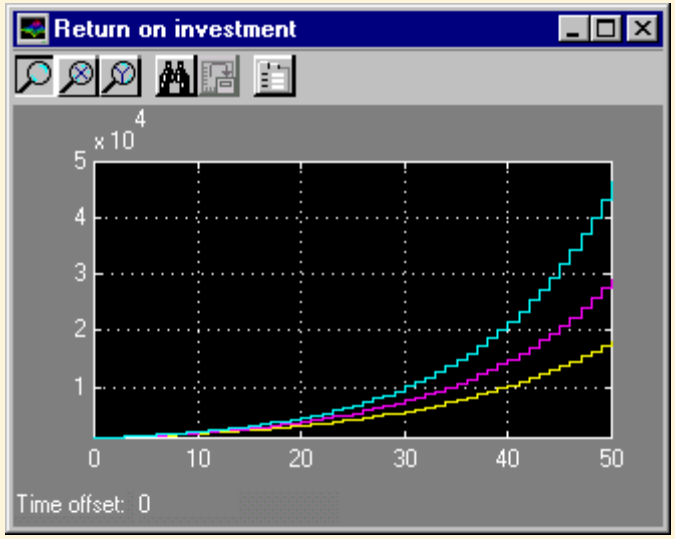
An interest rate example

Let's look at what happens if you gain 6%, 7% and 8% interest on a $1000 investment over 50 years

$$M_n = 1.06\, M_{n-1}$$

$$_0 = 1000$$





WOW!!!

That's a $28,000 difference!

Copyright © 1984 - 1997 by The MathWorks, Inc.

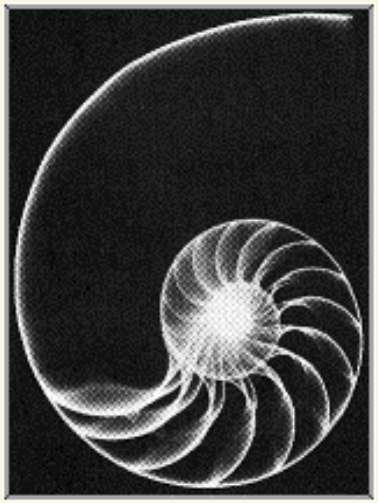
Example: Fibonacci

Generate the Fibonacci sequence with Simulink using unit delay, summing junction, and to workspace blocks, create a system that will generate the first 20 numbers in this series.

$$x_n = x_{n-1} + x_{n-2}$$

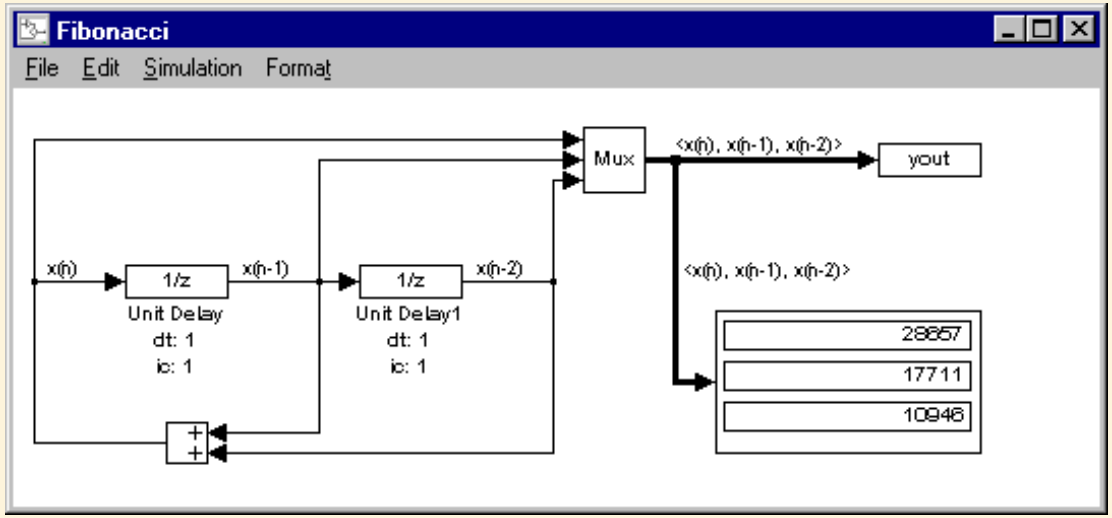
$$x_1 = 1,\ x_2 = 1$$

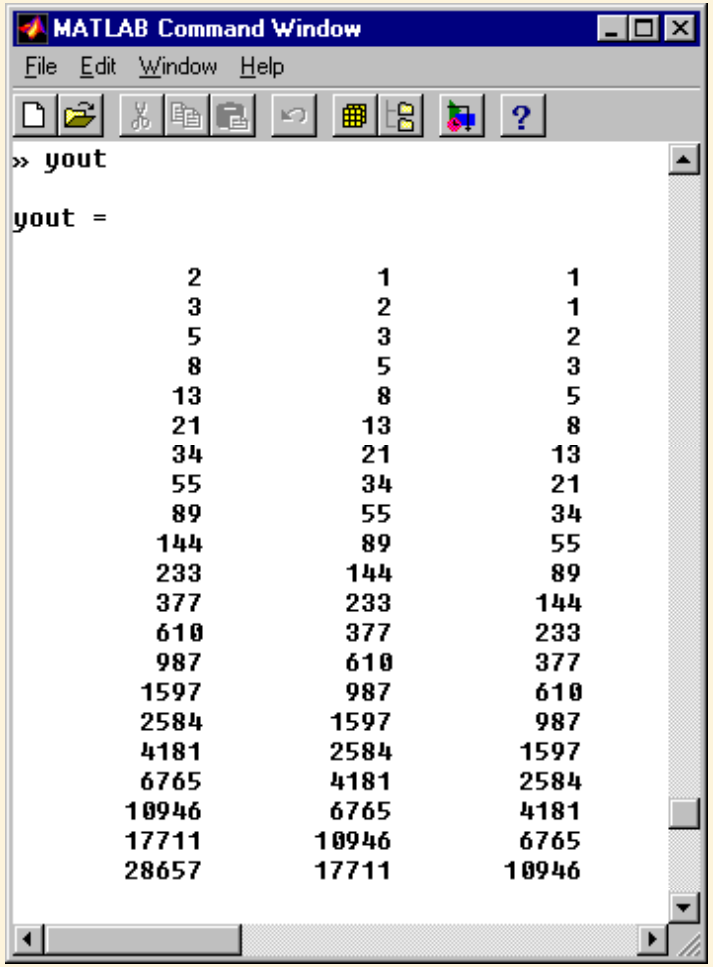
A chambered nautilus is really an example of the golden rectangle which is the Fibonacci sequence.



Copyright © 1984 - 1997 by The MathWorks, Inc.

Solution: Fibonacci



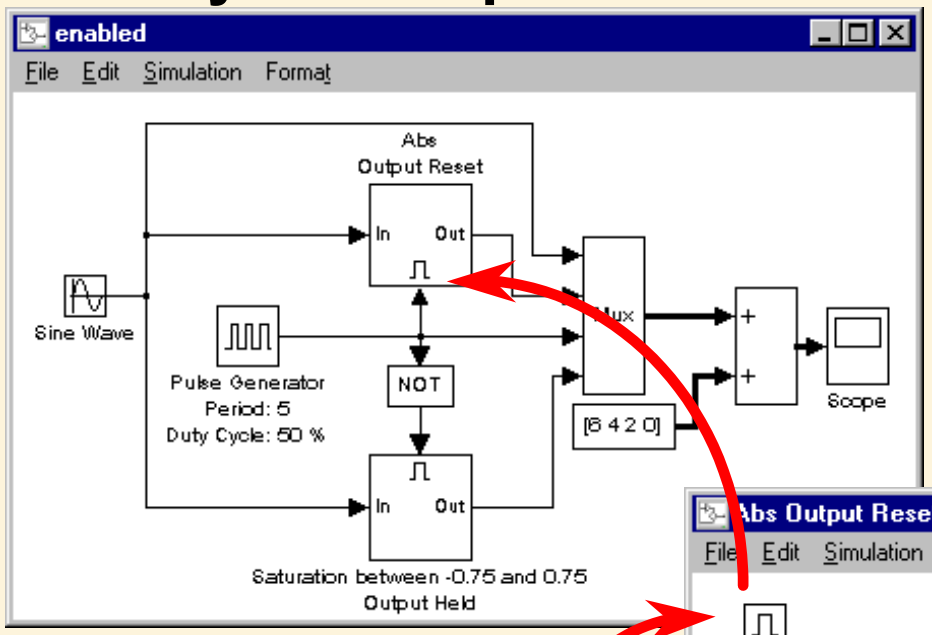


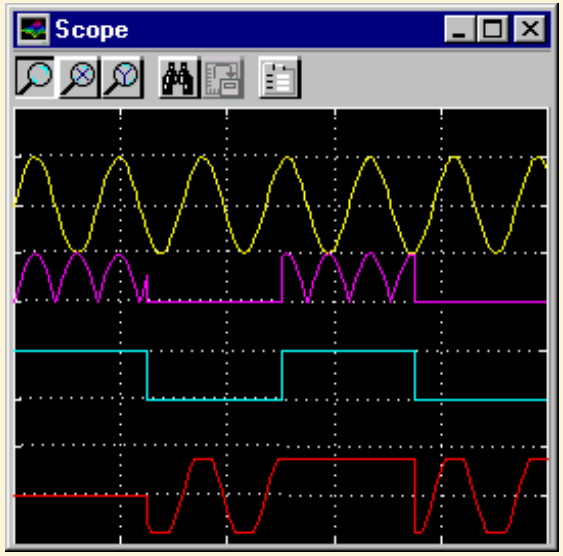
2	1	1
3	2	1
5	3	2
8	5	3
13	8	5
21	13	8
34	21	13
55	34	21
89	55	34
144	89	55
233	144	89
377	233	144
610	377	233
987	610	377
1597	987	610
2584	1597	987
4181	2584	1597
6765	4181	2584
10946	6765	4181
17711	10946	6765
28657	17711	10946

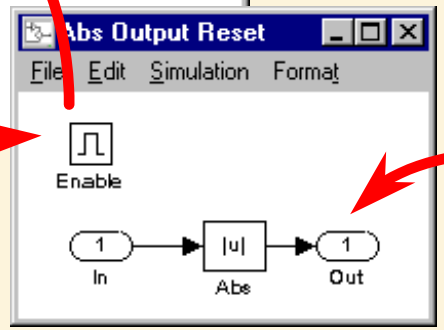
Copyright © 1984 - 1997 by The MathWorks, Inc.

Enabled subsystems

An enabled subsystem is executed at each simulation step for which the control signal to the subsystem is positive.







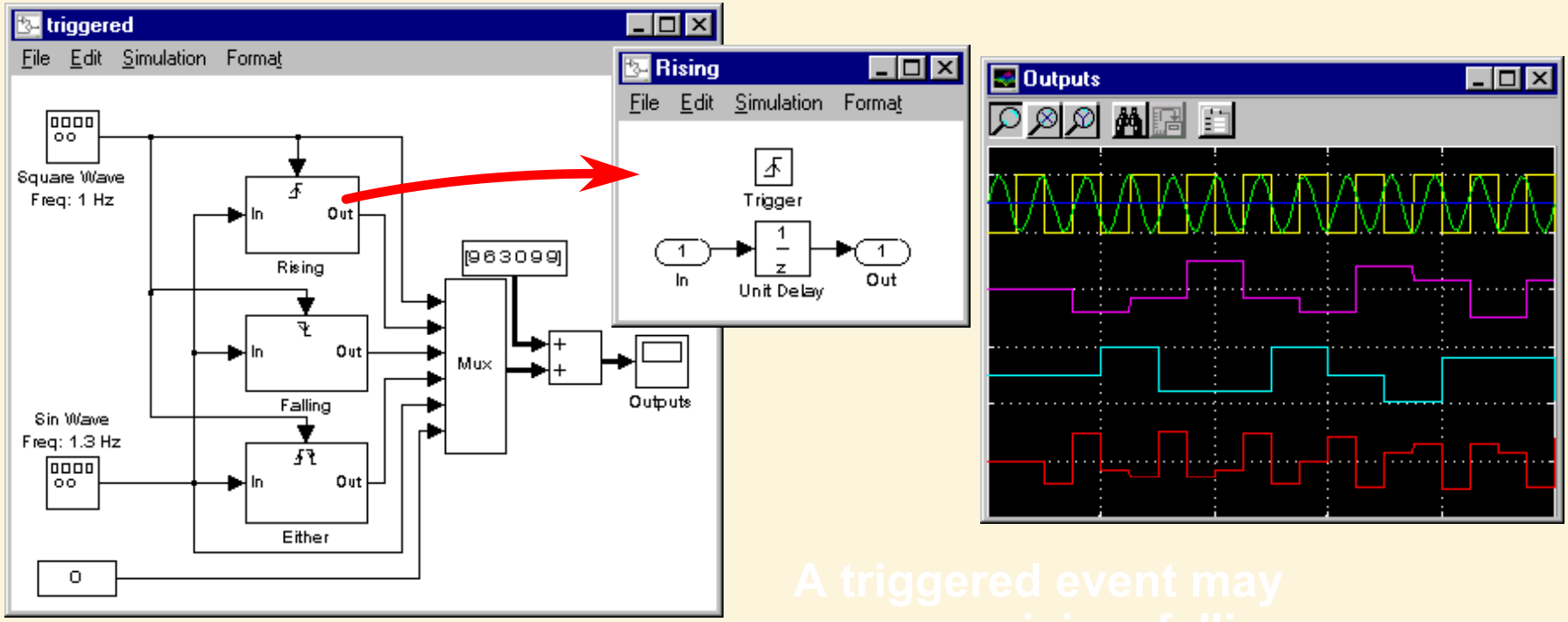
States may be held or reset

Outputs may be held or reset

Copyright © 1984 - 1997 by The MathWorks, Inc.

Triggered subsystems

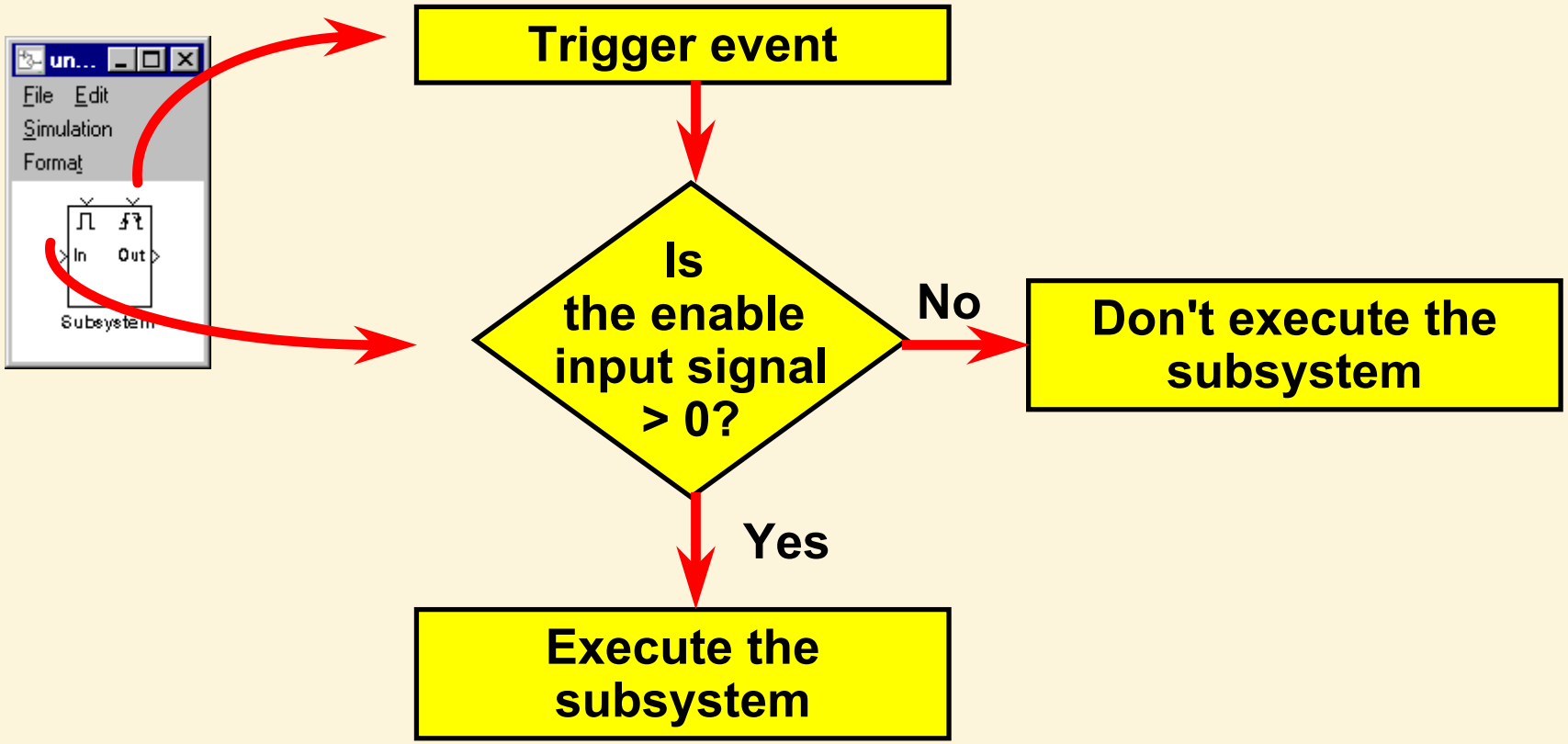
A triggered subsystem executes each time a trigger event occurs. The execution is for one time step only.



A triggered event may occur on rising, falling or both rising and falling edges of signal.

Copyright © 1984 - 1997 by The MathWorks, Inc.

Triggered and enabled subsystems



Copyright © 1984 - 1997 by The MathWorks, Inc.

From the command line

Why simulate from the command line?

- **Automate repeated simulations**
- **Parameter tuning**
- **Parameter Optimization (NCD)**

Copyright © 1984 - 1997 by The MathWorks, Inc.

From the command line

(The most basic method)

```
[t,x,y]=sim(model)
```

$$[t,x,y_1,y_2,\ldots,\ y_n]=\mathrm{sim}(\mathrm{model})$$

- **Simulates the model using all of the model parameters.**
- **The time, state and output vectors are returned from the simulation.**
- **Individual outputs may be obtained.**
- **Outputs are obtained from the outport blocks on the top level of the model.**

Copyright © 1984 - 1997 by The MathWorks, Inc.

From the command line

```
[t,x,y]=sim(model, timespan, options)
```

- options is a data structure that contains all of the solver parameters
- The options structure overrides the simulation parameters settings

```
» options=simget('external_pendulum')

options =

             Solver: 'ode45'
             RelTol: 1.0000e-003
             AbsTol: 1.0000e-006
             Refine: 1
            MaxStep: 0.1000
        InitialStep: 'auto'
           MaxOrder: 5
          FixedStep: 'auto'
       OutputPoints: 'all'
    OutputVariables: ''
            MaxRows: 1000
         Decimation: 1
       InitialState: []
     FinalStateName: 'xFinal'
              Debug: 'off'
              Trace: ''
       SrcWorkspace: 'current'
       DstWorkspace: 'current'
          ZeroCross: 'on'
```

Use `simget` and `simset` to get and set the options structure.

Copyright © 1984 - 1997 by The MathWorks, Inc.

From the command line

```
[t,x,y]=sim(model, timespan, options, ut)
```

- ut allows external inputs to be supplied to the inports on the top level of the model.
- [] in place of timespan or options indicates that the current model settings are used.

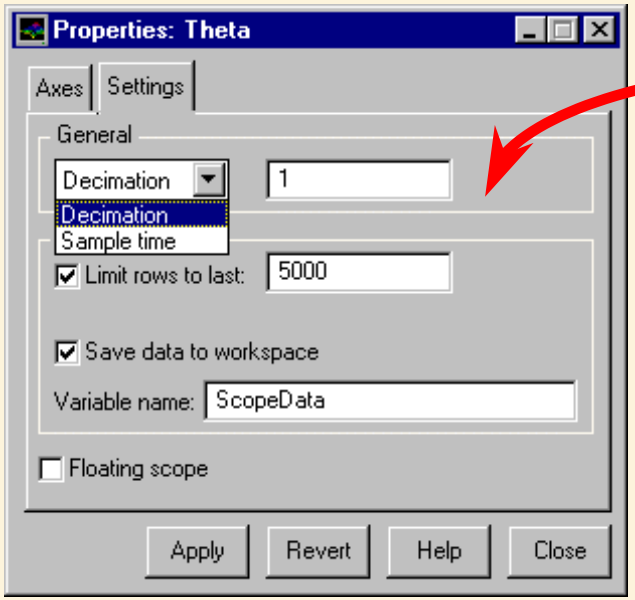
Copyright © 1984 - 1997 by The MathWorks, Inc.

Analyzing results

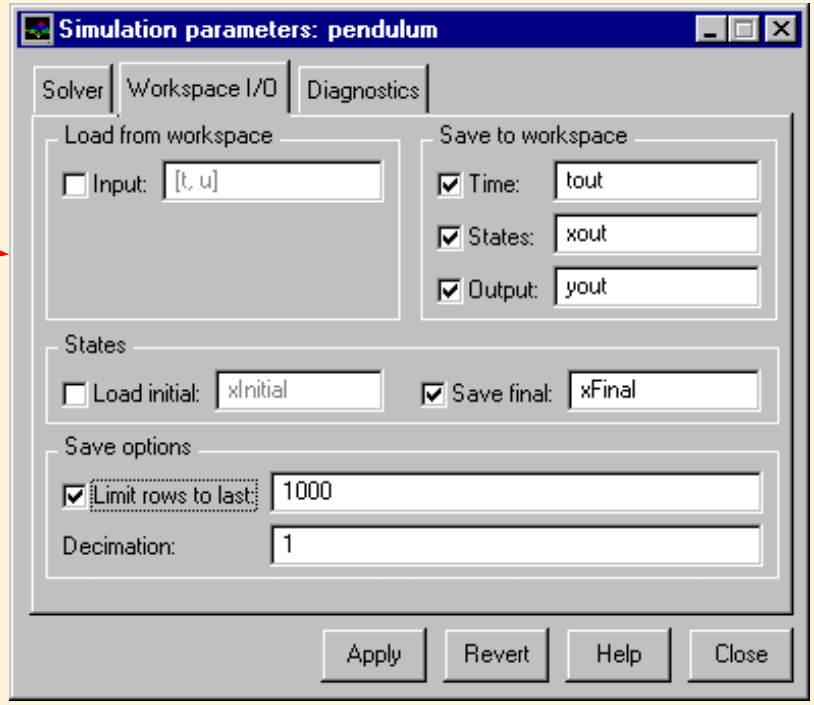
- Getting data out of models
- Floating Scope and Display blocks
- Trim
- Linearization

Copyright © 1984 - 1997 by The MathWorks, Inc.

Getting data out of models



The Scope, ToWorkspace, ToFile and Parameters Dialog all have similar options



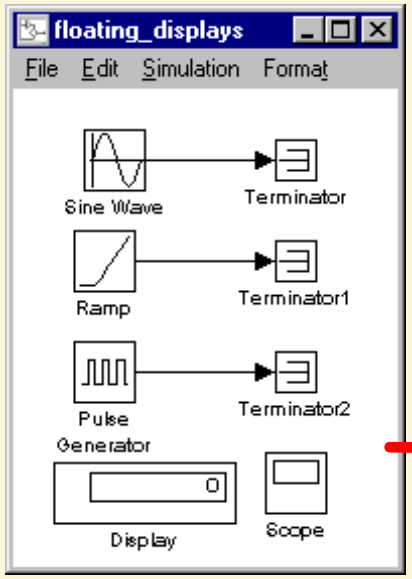
Remember:

These blocks use the data created by the solvers

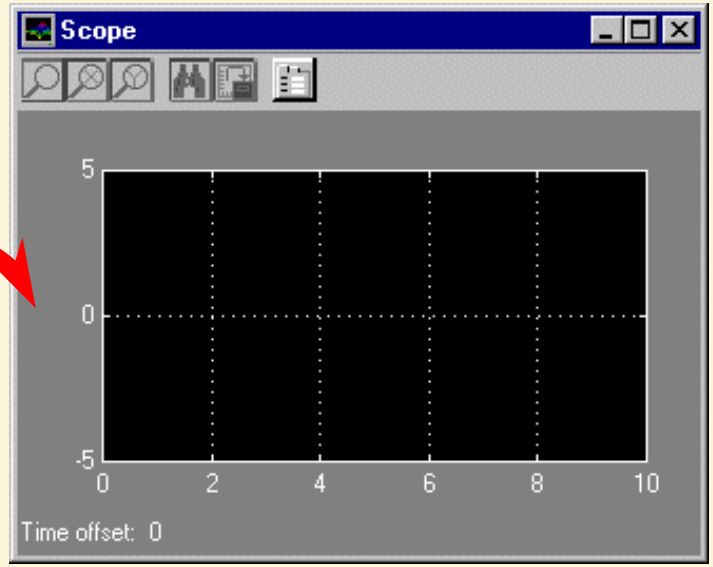
Copyright © 1984 - 1997 by The MathWorks, Inc.

The floating Scope and Display blocks

Selecting any signal will display the value(s) of that signal on any floating scope or display block on that diagram

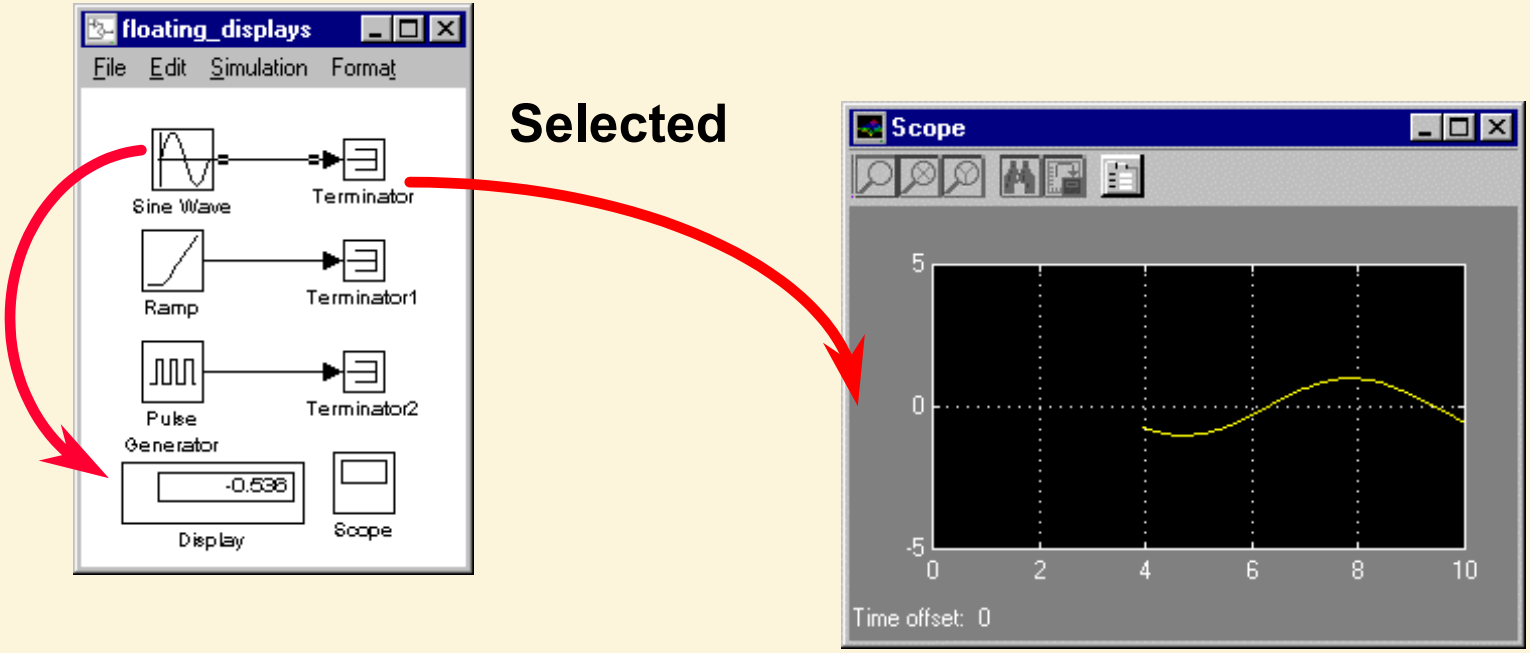


Unselected



Copyright © 1984 - 1997 by The MathWorks, Inc.

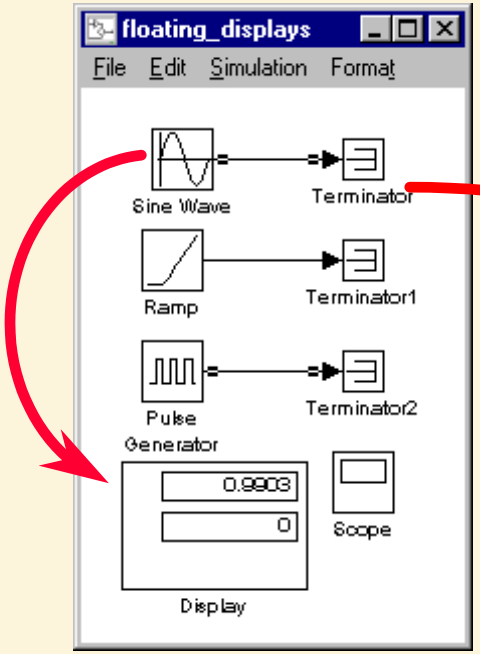
The floating Scope and Display blocks



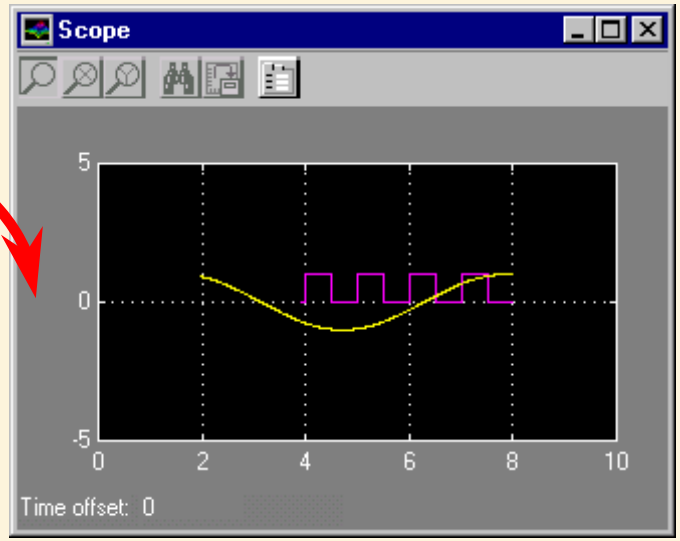
Copyright © 1984 - 1997 by The MathWorks, Inc.

The floating Scope and Display blocks

You can even select multiple signals...



Selected



Copyright © 1984 - 1997 by The MathWorks, Inc.

Trim

- Find the values of states x and inputs u that bring the system to steady state (dx=0)
- The solution is not necessarily unique

```
[x, u, y]=trim('sys')
```

- To provide an initial guess at the solution

```
[x, u, y]=trim('sys', x0, u0, y0)
```

- Individual elements of x,u,y can be constrained
- Individual derivatives can be fixed to non-zero values
- Can also add a fourth output argument to get the state derivatives at trim (`[x,u,y,dx]=…`)

Copyright © 1984 - 1997 by The MathWorks, Inc.

Linearization

- **Use linmod to extract linear models from Simulink systems**
  - Perturbs each state around the operating point to determine the rate of change in the state derivatives and outputs (Jacobians)
- **Use linmod2 for advanced linearization**
  - attempts to balance roundoff and truncation error
  - perturbation levels of each state-space matrix element set individually
  - can detect discontinuities and produce warning messages

Copyright © 1984 - 1997 by The MathWorks, Inc.

Discrete linearization - dlinmod

- **Used for discrete, multi-rate and hybrid systems**
- **One additional input argument, Ts, after the system name**

```
[Ad, Bd, Cd, Dd]=dlinmod('sys',Ts,x,u)
```

- **Use Ts=0 for a continuous model approximation**

Copyright © 1984 - 1997 by The MathWorks, Inc.

Summary

- Block diagram modeling
- Interactive operation
- Continuous/discrete/hybrid systems
- Linear & nonlinear components
- Hierarchical models - subsystems
- Conditional execution
- Command-line operation
- Tightly integrated with MATLAB
- Linearization for Control Design
- Supported by Stateflow, Real-Time Workshop

Copyright © 1984 - 1997 by The MathWorks, Inc.