

MATLAB/Simulink Introduction with Corsica Communication Interface

Bill Meyer

Corsica Winter School - ASIPP

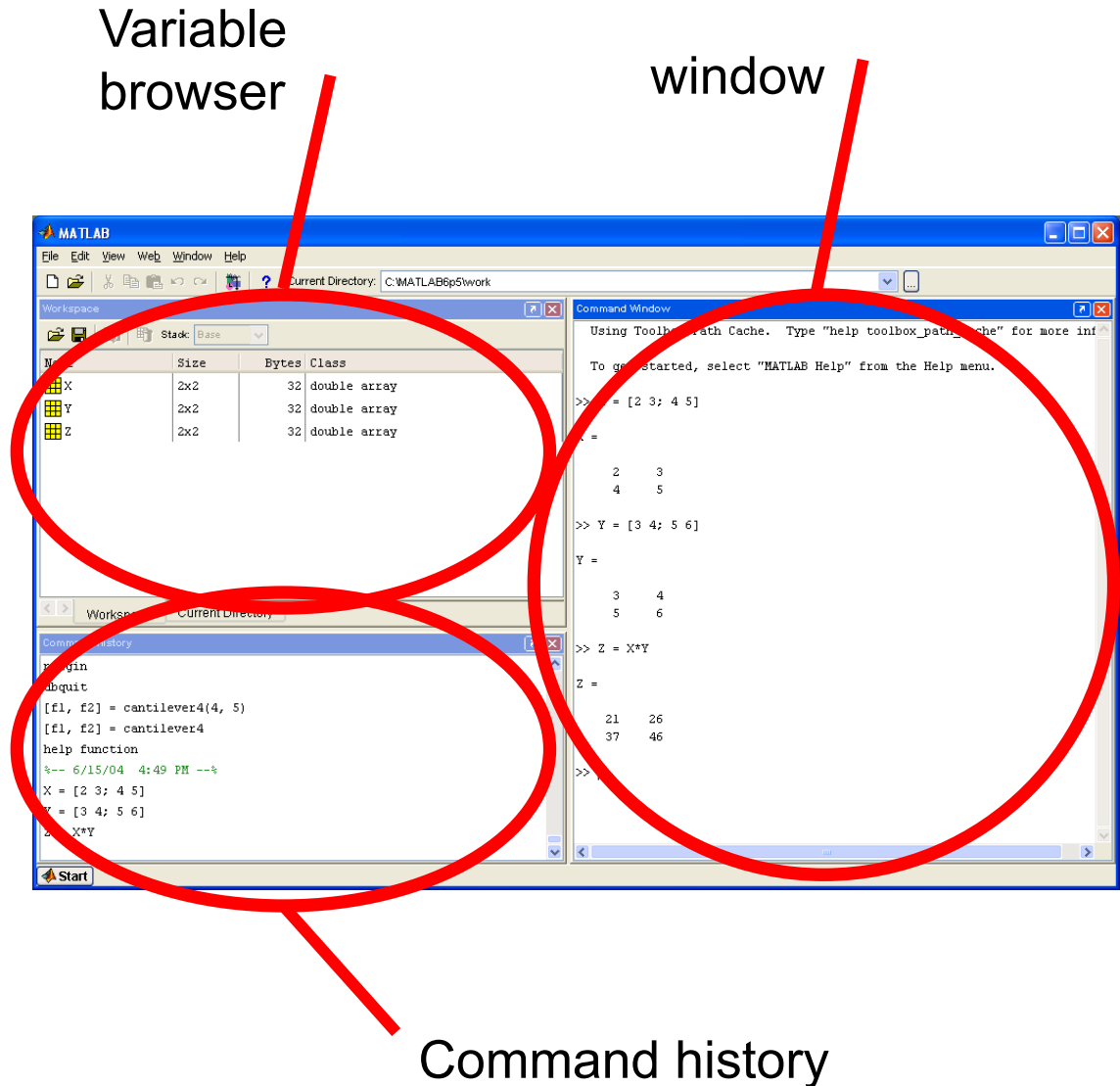
January 26-28, 2016



This work was performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Security, LLC, Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344.

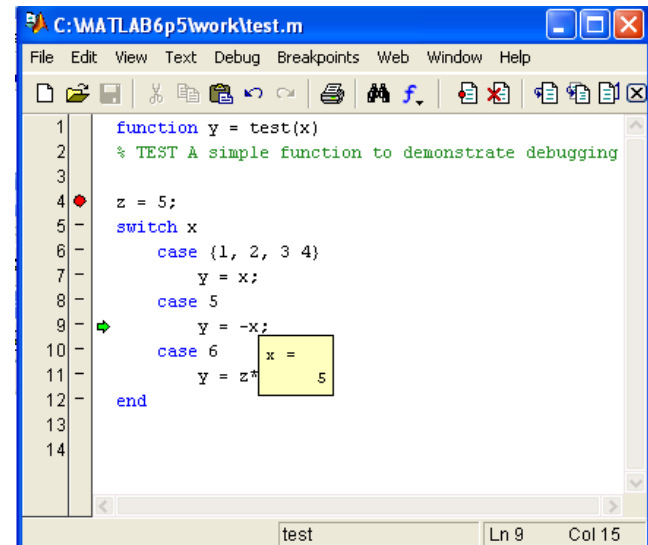
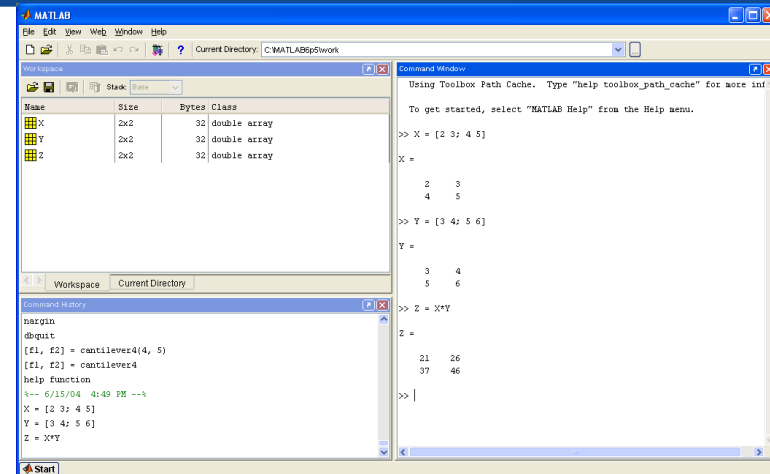
Introduction to Matlab

- Click on the Matlab icon/start menu initialises the Matlab environment:
- The main window is the dynamic command interpreter which allows the user to issue Matlab commands
- The variable browser shows which variables currently exist in the workspace



Matlab Programming Environment

- Matlab (Matrix Laboratory) is a **dynamic, interpreted**, environment for matrix/vector analysis
- Variables are created at run-time, matrices are dynamically re-sized, ...
- User can build programs (in .m files or at command line) using a C/Java-like syntax
- Ideal environment for model building, system identification and control (both discrete and continuous time)
- Wide variety of libraries (toolboxes) available



Numbers and variables and similar in Matlab

- Smallest positive floating point number $2.2251e-308$, and the highest is $1.7977e+308$.
- Spacing of floating point numbers (calculation precision) is $2.2204e-016$.
- $1/0$ gives infinite - *Inf*.
- $0/0$ or $Inf-Inf$ gives *NaN* – (not-a-number).
- Matlab is case sensitive; $a = 10$ is not equal to $A = 10$.
- If the command is concluded with semicolon, the result will not be shown on the screen.
- For decimal numbers, dot is used, for example 2.45.
- Formats: format short, format long, format long e...format.
- *% Comment*.

Numbers and variables and similar in Matlab

- $2.4e-12$ is $2.4 \cdot 10^{-12}$
- pi is the variable with defined name.
- i or j is complex unit (it can be overwritten).
- For trigonometric functions [rad] is used.
- *clear all*, clears all defined variables.
- *close all*, closes all graphical windows.
- *clear all*, *close all*, very usefull combination!
- *clc*, clears the screen, but nothing else.
- CTRL+C stop the execution of the program in Matlab.
- *dir*, current directory.
- *who*, list of all defined variables.

Basic mathematical operation

- $+$, $-$, $*$, $/$,
- $\text{sqrt}(a)$, square root,
- a^b , power,
- $\log(a)$, natural algorithm,
- $\exp(a)$, $\log_{10}(a)$,
- $\text{abs}(a)$, absolute value ,
- $\cos(a)$, $\sin(a)$, $\text{acos}(a)$, $\text{asin}(a)$,
- $\sinh(a)$, $\cosh(a)$, $\tanh(a)$,
- $\text{mod}(a)$, Modulus after division,
- $\text{rem}(a)$ Remainder after division,
- $\text{floor}(a)$, $\text{ceil}(a)$, $\text{round}(a)$, Round towards ...

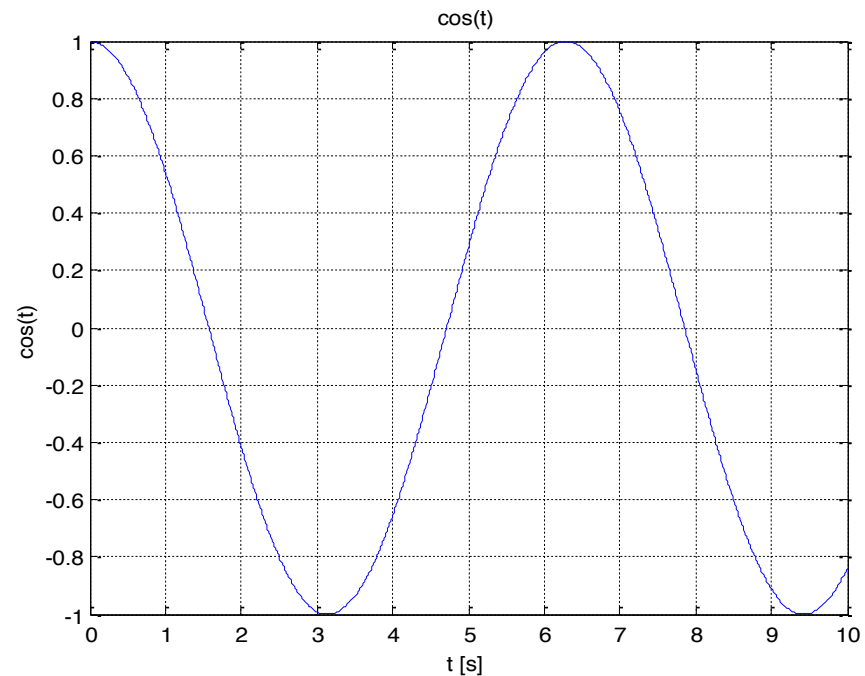
Matrixes and vectors

- $x = [1,2,3]$, vector-row,
- $y=[1;2;3]$, vector-column,
- $x=0:0.1:0.8$, vector $x=[0,0.1,0.2,0.3....0.8]$,
- $A = [1,3,5;5,6,7;8,9,10]$, matrix,
- $A(1,2)$, element of matrix, 1. row, 2. column,
- $A(:,2)$, second column of matrix,
- $A(1,:)$, first row of matrix ,
- $C=[A;[10,20,30]]$ matrix with additional row,
- $A(:,2)=[]$, deleting of second column,
- $B=A(2:3, 1:2)$, part of matrix,
- x' , transpose.

Graphics

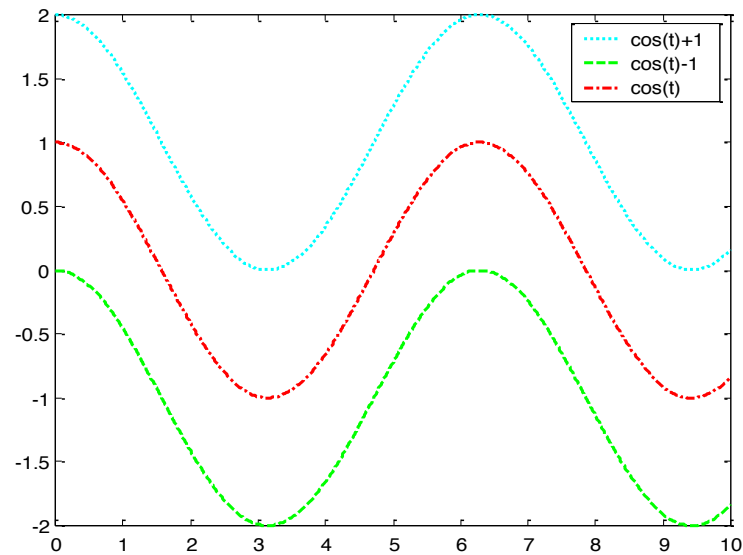
- `t=[0:0.01:10];`
- `figure(1), plot(t,cos(t))`
- `title('cos(t)')`
- `xlabel('t [s]'), ylabel('cos(t)')`
- `grid`

- Copy the figure:
MENU: Edit->Copy Figure



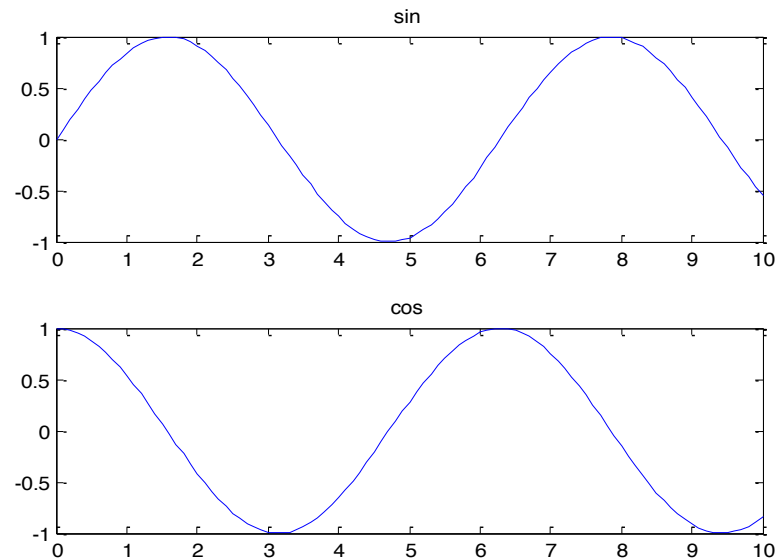
Graphics

- `figure(1), ahandle=plot(t,cos(t)+1,'c',t,cos(t)-1,'-g',t,cos(t),'-.r');`
- `set(ahandle,'LineWidth',[2]);`
- `legend('cos(t)+1','cos(t)-1','cos(t));`



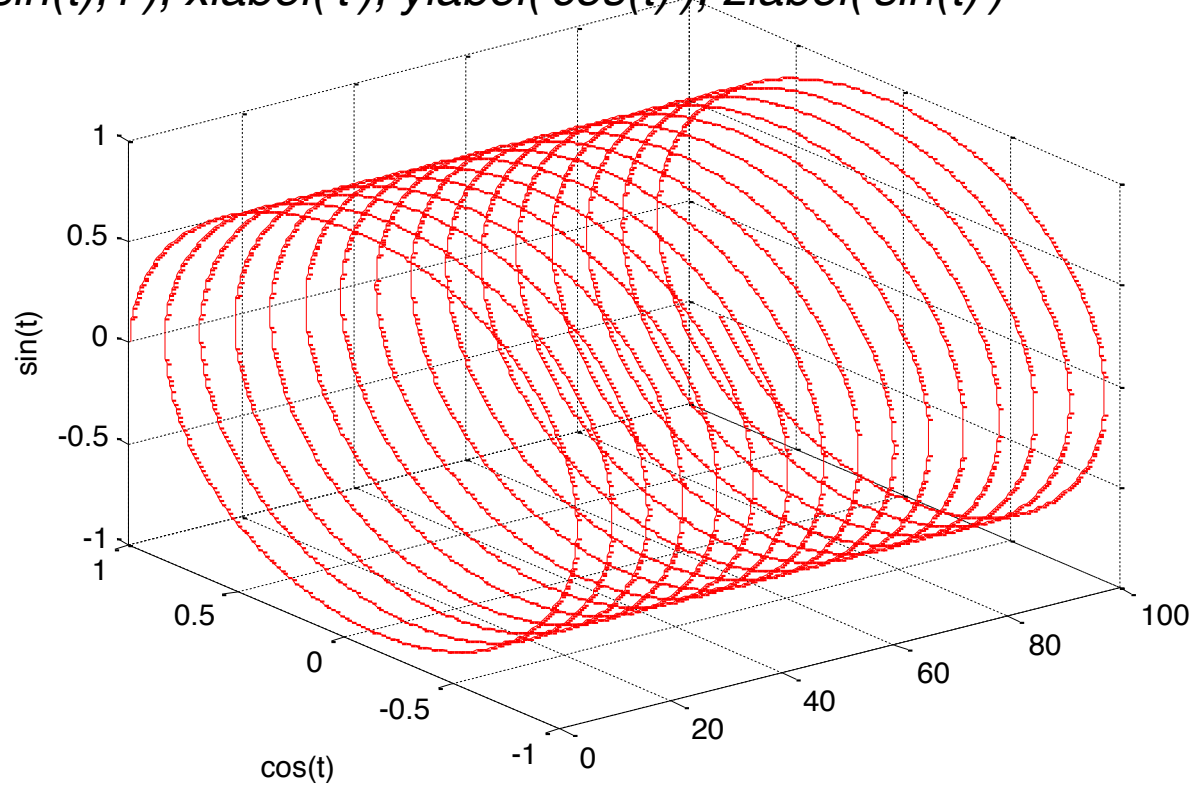
Graphics

- `t=[0:0.1:10];`
- `subplot(2,1,1),plot(t,sin(t)),title('sin')`
- `subplot(2,1,2),plot(t,cos(t)),title('cos')`



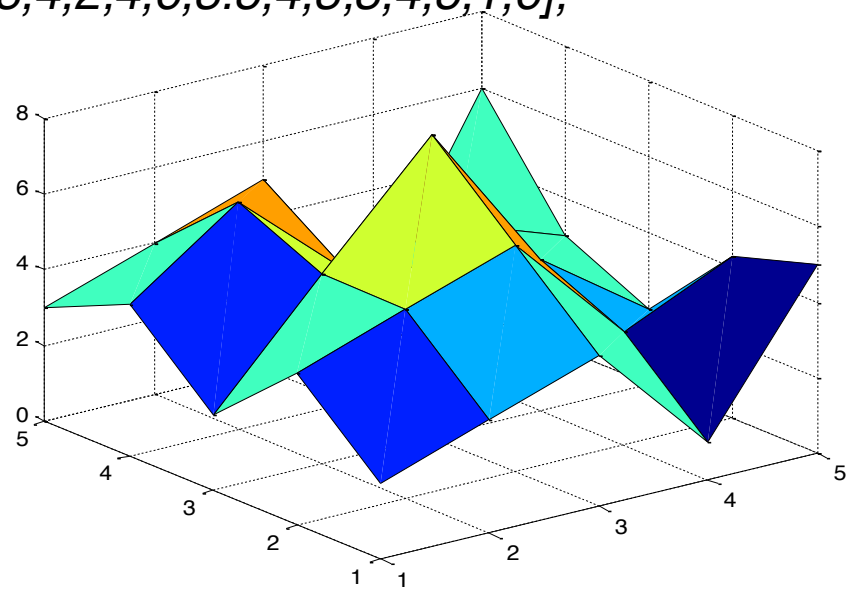
Graphics

- `t=[0:0.02:100];`
- `plot3(t,cos(t),sin(t),'r'), xlabel('t'), ylabel('cos(t)'), zlabel('sin(t)')`
- `grid`



Graphics

- $X=[1,2,3,4,5];$
- $Y=[1,2,3,4,5];$
- $Z=[2,3,4,1,5;4,5,6,3,4.3;2,5,8,4,2;4,6,3.5,4,3;3,4,5,1,6];$
- $\text{surf}(X, Y, Z)$



Program flow

- For loop

```
for I = 1:0.2:N,  
    for J = 1:N,  
        A(I,J) = 1/(I+J-1);  
    end  
end
```

Program flow

- While loop

```
while (a<b)  
  c(a)=sin(a);  
  a=a+1;  
end
```

Program flow

- If-else statement

```
if  $I == J$   
     $A(I,J) = 2;$   
elseif  $abs(I-J) == 1$   
     $A(I,J) = -1;$   
else  
     $A(I,J) = 0;$   
end
```

Help in Matlab

- *help sqrt*, looking for known command,
- *lookfor algorithm*, looking for the key words,
- *help*, help topics are shown,
- *help*, interrogation point in menu, Matlab help windows opens,
- *demo*, window with (many!) examples opens.

M-files

- Matlab files with program and/or definitions.
- Name of the files must be without special characters and spaces.
- Path to the directory with m-file must be set (if it is not *work* directory).
 - `path(„<my directory“,path)`
- To run, type the name of the file in the Command window or run directly from Matlab editor.
 - `$CWD/startup.m` run at startup
- New m-file is created in Matlab menu under *File/New/M-file*

MTConnect.m

```
Terminal
File Edit View Terminal Tabs Help
bash-3.2$ cat MTconnect.m

%
% This assumes that localhost (127.0.0.1) has been tunneled to
% Corsica. The local end of the tunnel is at port 1609
%
host = 'localhost'
port = 1606

cport = corsicaconnect(host,port)
corsicaparser('rpc chameleon simver="MTconnect"',cport)
corsicaparser('rpc real foobar= 0.0',cport)
corsicaparser('rpc real foobar2= 0.0',cport)
sim('SIMconnect')
corsicaparser('rpc.foobar2',cport)
corsicaparser('return')
exit

bash-3.2$ █
```

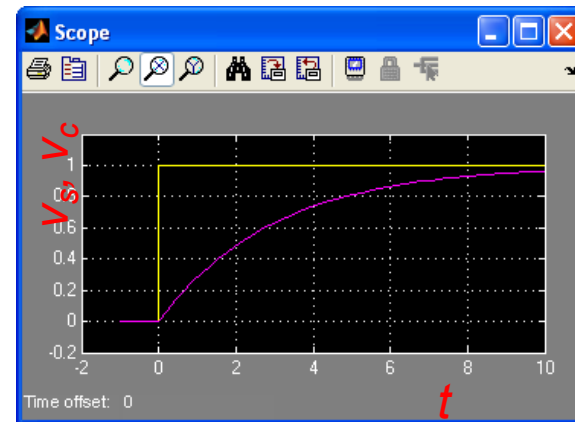
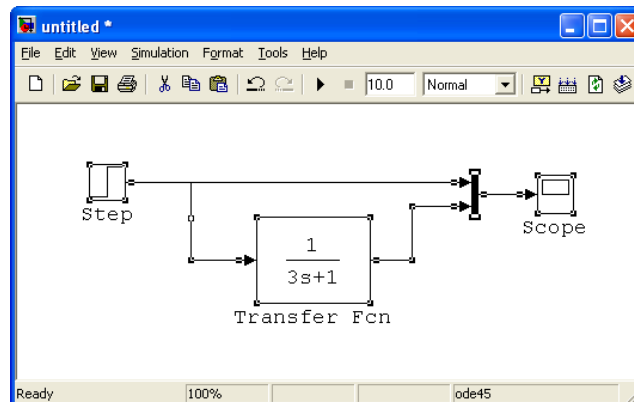
**Corsicaconnect &
corsicaparser part
of corsica checkout**

```
read ignorebegin
#
# This port number (1606) is arbitrary. It can be any number
# in the range 1025-65535 as long as long as it's not already
# in use by another application. Ports numbers <=1024 can only
# be used by processes run as the root user. Whatever is used
# it must match what is connected to from Matlab, or if tunneled,
# the local end of the tunnel must end up at this port.
#
#
# Corsica level_1 test
# MTconnect
rpcinit(1606)
read ignoreend
rpcserver

if ( simver <> "MTconnect") then
<< "Wrong Simulink simulation run: " << simver
endif
```

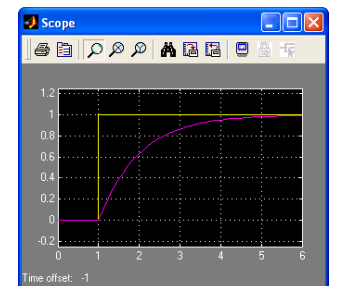
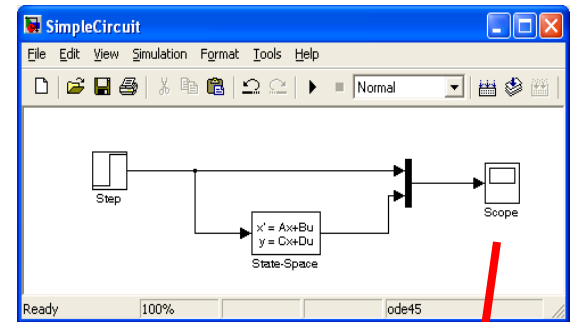
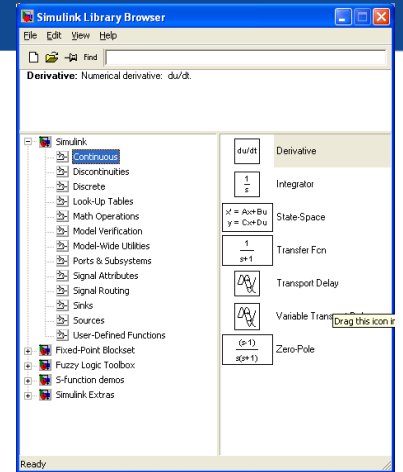
Introduction to Simulink

- Simulink is a graphical, “drag and drop” environment for building simple and complex signal and system dynamic simulations.
- It allows users to concentrate on the structure of the problem, rather than having to worry (too much) about a programming language.
- The parameters of each signal and system block is configured by the user (right click on block)
- Signals and systems are simulated over a particular time.



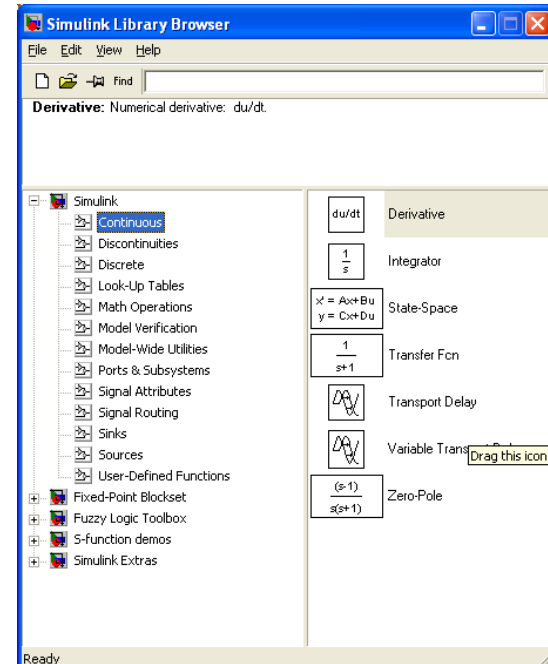
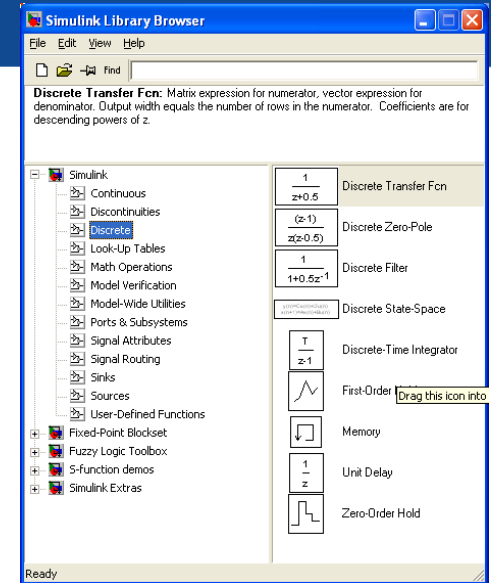
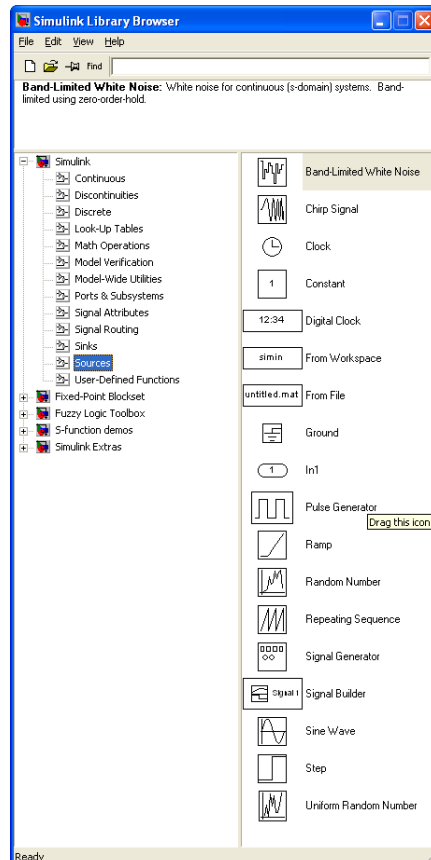
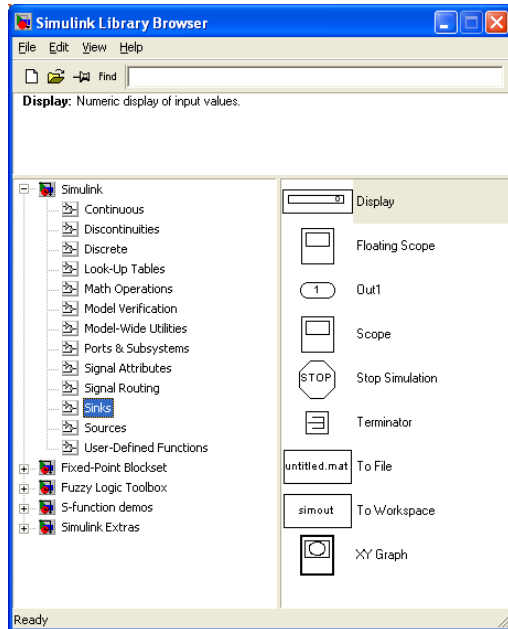
Starting and Running Simulink

- Type the following at the Matlab command prompt
- `>> simulink`
- The **Simulink library** should appear
- Click File-New to create a new **workspace**, and drag and drop objects from the library onto the workspace.
- Selecting **Simulation-Start** from the pull down menu will run the dynamic simulation. Click on the blocks to view the data or alter the run-time parameters



Signals and Systems in Simulink

- Two main sets of libraries for building simple simulations in Simulink:
- **Signals:** Sources and Sinks
- **Systems:** Continuous and Discrete

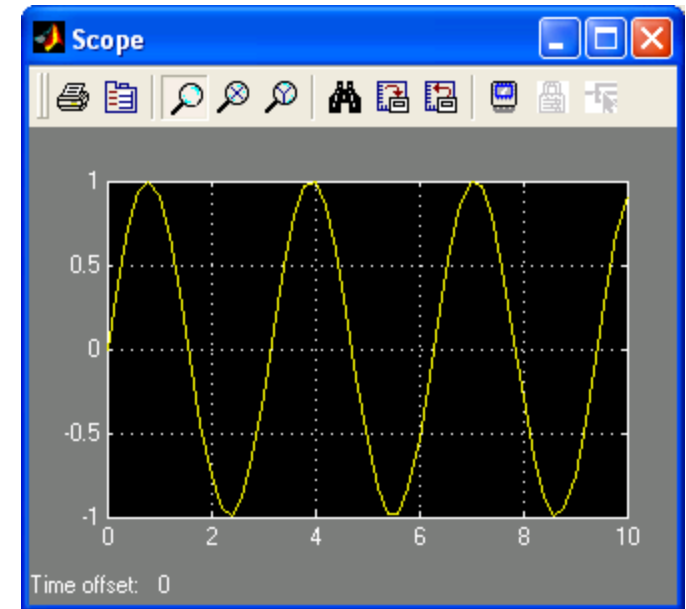
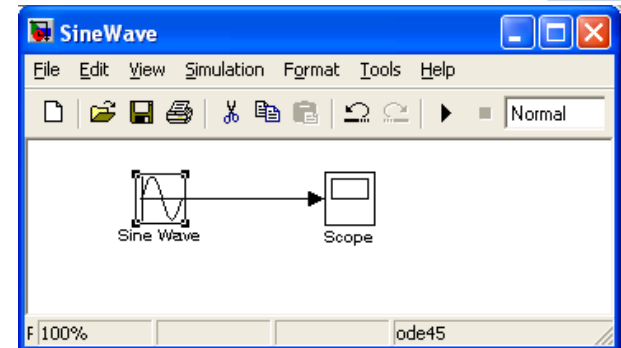


Simulink- libraries

- *Continuous*; integrator, transfer function..
- *Discrete*; discrete transfer function, unite delay, memory..
- *Math operations*; gain, product, sum, trigonometric functions..
- *Sinks*; blocks that have only input, scope, to worspace..
- *Sources*; blocks that have only output, generators, constant,...
- *User defined functions*: S-function, S-function builder,...

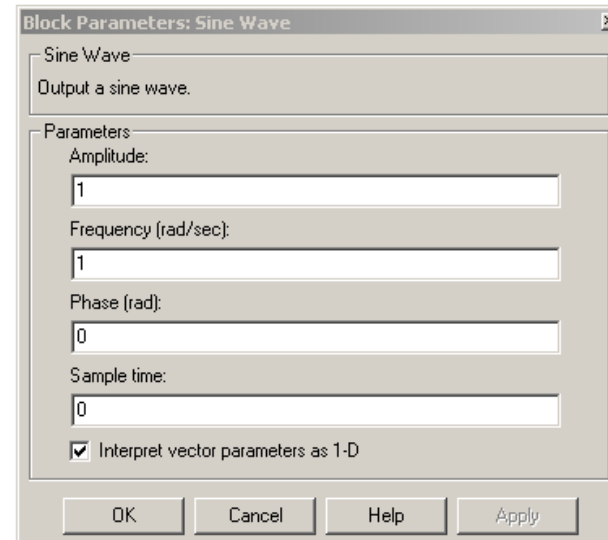
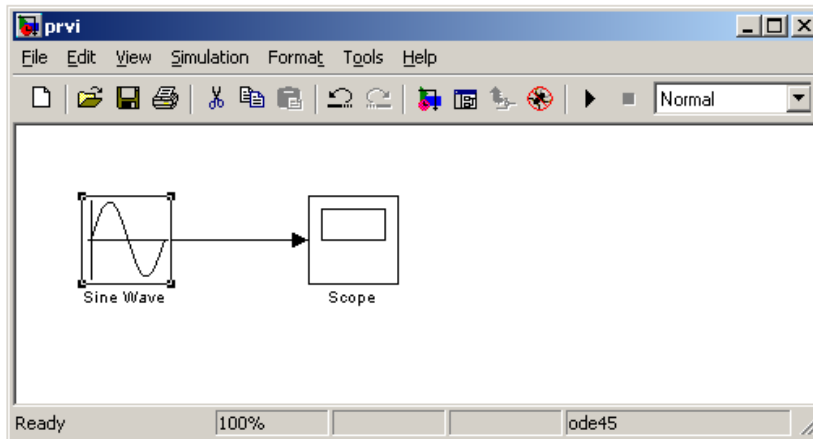
Basic Simulink Example

- Copy “sine wave” source and “scope” sink onto a new Simulink work space and connect.
- Set sine wave parameters modify to 2 rad/sec
- Run the simulation:
 - Simulation - Start
- Open the scope and leave open while you change parameters (sin or simulation parameters) and re-run
- Many other Simulink demos ...



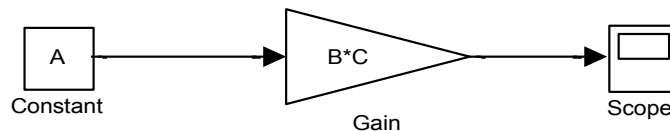
Simulink – creating a model

- Model is created by choosing the blocks from different libraries, dragging them to model window and linking them.
- The parameters of block (shown on picture, sine wave parameters), can be reached with double click on the block.



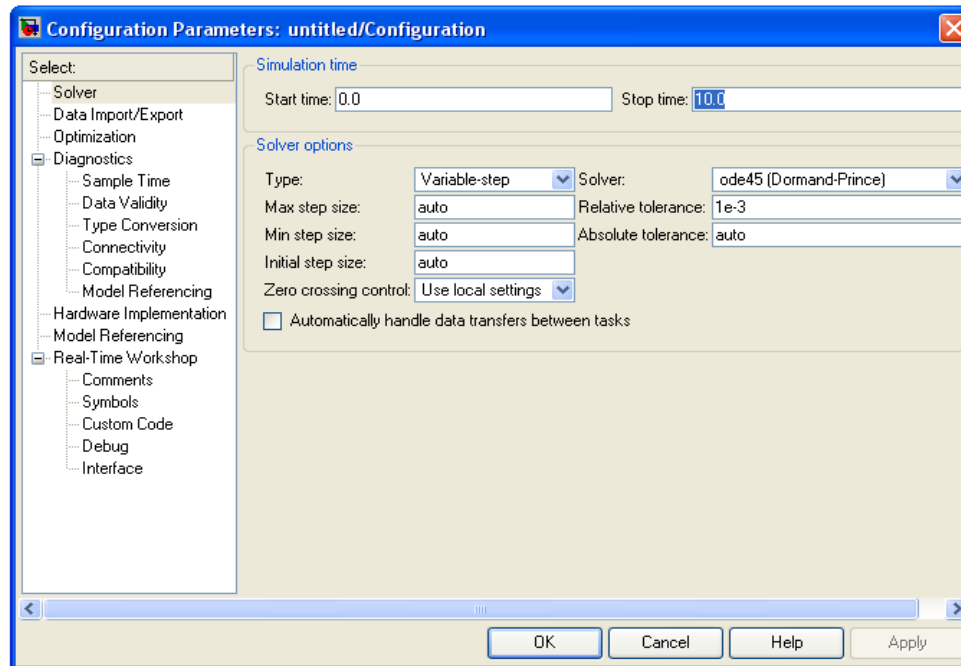
Simulink

- The parameters in the blocks can be variables, that are defined in m-file. M-file must be executed before simulation start, so that variables are defined in Command window, where Simulink can reach them.
- **Never give the same name to Simulink model and m-file!**
- Example: A, B and C must be defined with values in the Command window.



Simulink- configuration parameters

- Numerical solver method, start time, stop time (it can be also set directly)...



Simulink Library libcorsica.mdl

The screenshot displays the MATLAB 7.6.0 (R2008a) environment. The main window shows the workspace with a file list for the current directory. The Command Window shows the execution of the `libcorsica` command. A separate window titled "Library: libcorsica" displays the library's structure, including sub-libraries for Data, Enable, CorsicaSink, and CorsicaSrc.

Current Directory: /home/meyer8/cvsstuff/caltrans/matlab

All Files	Type	Size	Date
caltrans	Folder		1/15/
CVS	Folder		4/9/1
rpc	Folder		2/29/
simulink	Folder		8/18/
#corsicasink.c.1.1	1 File	8 KB	8/17/
#corsicasink.c.1.2	2 File	9 KB	4/2/1
#corsicasource.c.1.1	1 File	8 KB	8/3/1
#Makefile.1.6	6 File	2 KB	3/5/0
#Makefile.1.7	7 File	3 KB	4/2/1
#mexclnt.c.1.2	2 File	7 KB	3/5/0
#mexclnt.c.1.3	3 File	7 KB	4/2/1
circuitstep1.c	C Source	8 KB	3/5/0
circuitstep1.mexglx	MEX-file	118 KB	2/29/
circuitstep2.c	C Source	8 KB	3/5/0
circuitstep2.mexglx	MEX-file	119 KB	2/29/
clnt.c	C Source	3 KB	4/2/1
clnt.o	O File	4 KB	2/29/
corsica.c	C Source	9 KB	3/4/0
corsica.mexglx	MEX-file	120 KB	2/29/
corsica2.mexglx	MEX-file	125 KB	2/29/

Command Window:

```
>> libcorsica
>> |
```

Library: libcorsica Structure:

- > Data
- > Enable
 - CorsicaSink
- > Enable Data
 - CorsicaSrc

CorsicaSrc S-Function

- Insert data from Corsica into simulation

Parameters are 'variable',length, optional (pre) command (corsica),port

Library:libcorsica

File Edit View Format Help

›Data
›Enable
CorsicaSink

›Enable Data
CorsicaSrc

untitled *

File Edit View Simulation Format Tools

›Enable Data
CorsicaSrc

Function Block Parameters: CorsicaSrc

S-Function

User-definable block. Blocks can be written in C, M (level-1), Fortran, and Ada and must conform to S-function standards. The variables t, x, u, and flag are automatically passed to the S-function by Simulink. You can specify additional parameters in the 'S-function parameters' field. If the S-function block requires additional source files for the Real-Time Workshop build process, specify the filenames in the 'S-function modules' field. Enter the filenames only; do not use extensions or full pathnames, e.g., enter 'src src1', not 'src.c src1.c'.

Parameters

S-function name: corsicasource Edit

S-function parameters: 'foobar,1,',0

S-function modules: "

OK Cancel Help Apply

CorsicaSink S-Function

- Send data from simulation into Corsica

The image shows a Simulink workspace with two windows. The top window, titled 'Library:libcorsica', displays two S-function blocks: 'CorsicaSink' and 'CorsicaSrc'. The bottom window, titled 'untitled *', shows a similar workspace with 'CorsicaSrc' and 'CorsicaSink' blocks. A dialog box titled 'Sink Block Parameters: CorsicaSink' is open, showing the configuration for the 'CorsicaSink' block. The 'S-function name' field contains 'corsicasink'. The 'S-function parameters' field contains 'foobar',1,',cport', which is circled in red. A red arrow points from the text 'Parameters are 'variable', length, optional (post) command (corsica), port' to the circled parameters field. The 'S-function modules' field is empty. The dialog box has 'OK', 'Cancel', 'Help', and 'Apply' buttons.

Parameters are 'variable', length, optional (post) command (corsica), port

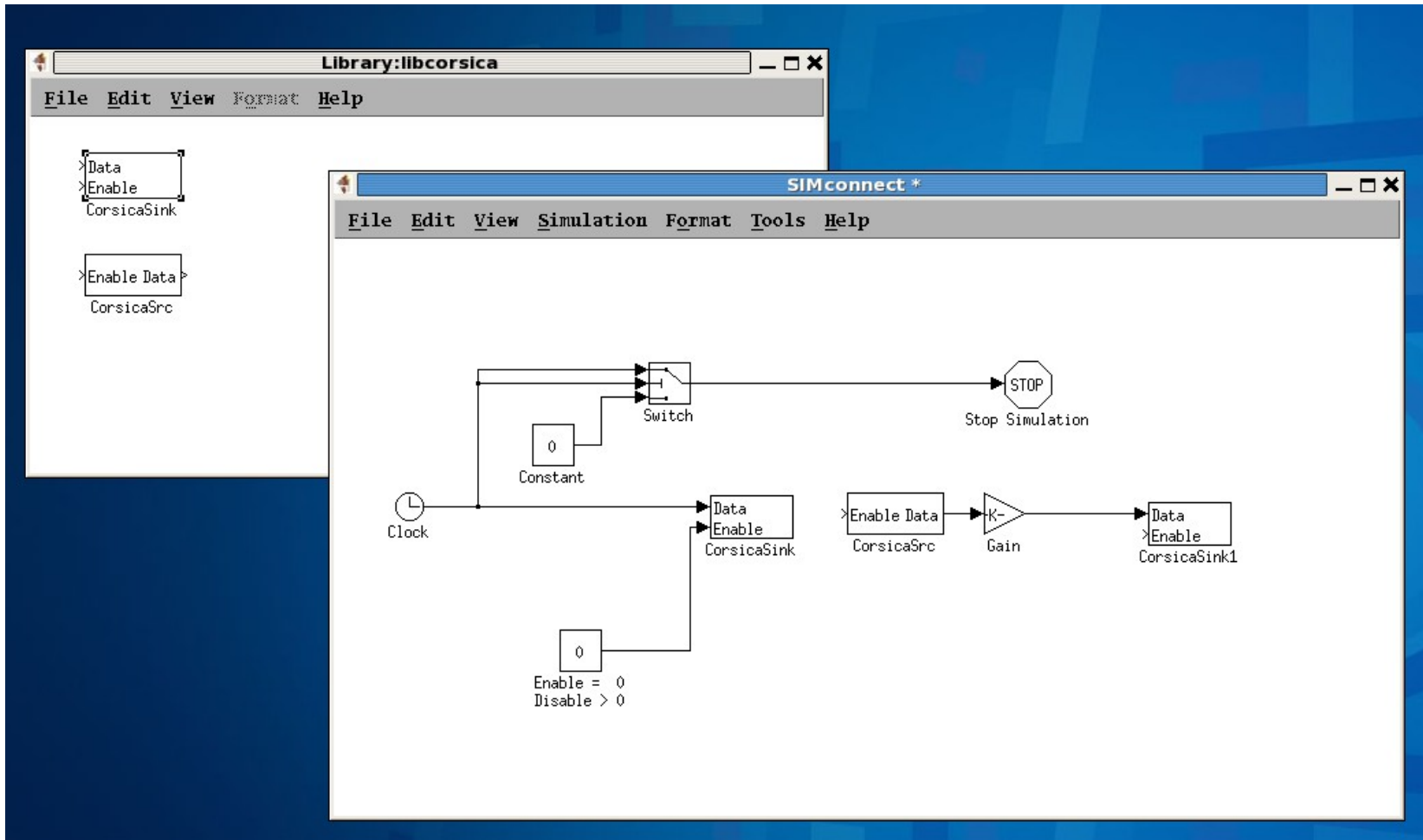
S-function name: corsicasink

S-function parameters: 'foobar',1,',cport'

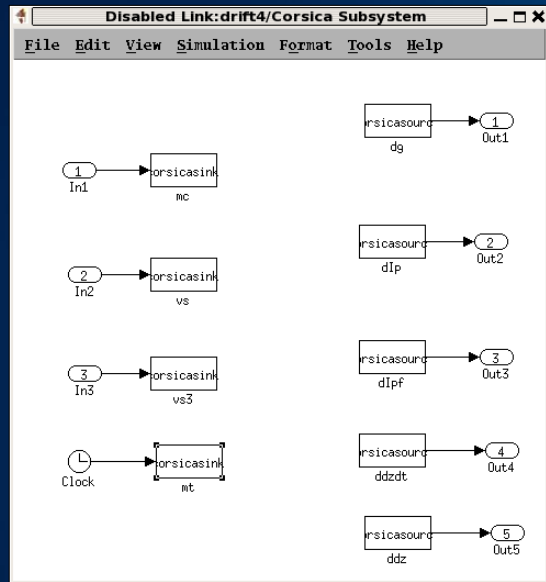
S-function modules: "

SIMConnect

- Checks communication with Corsica



ITER Simulation



MATLAB 7.

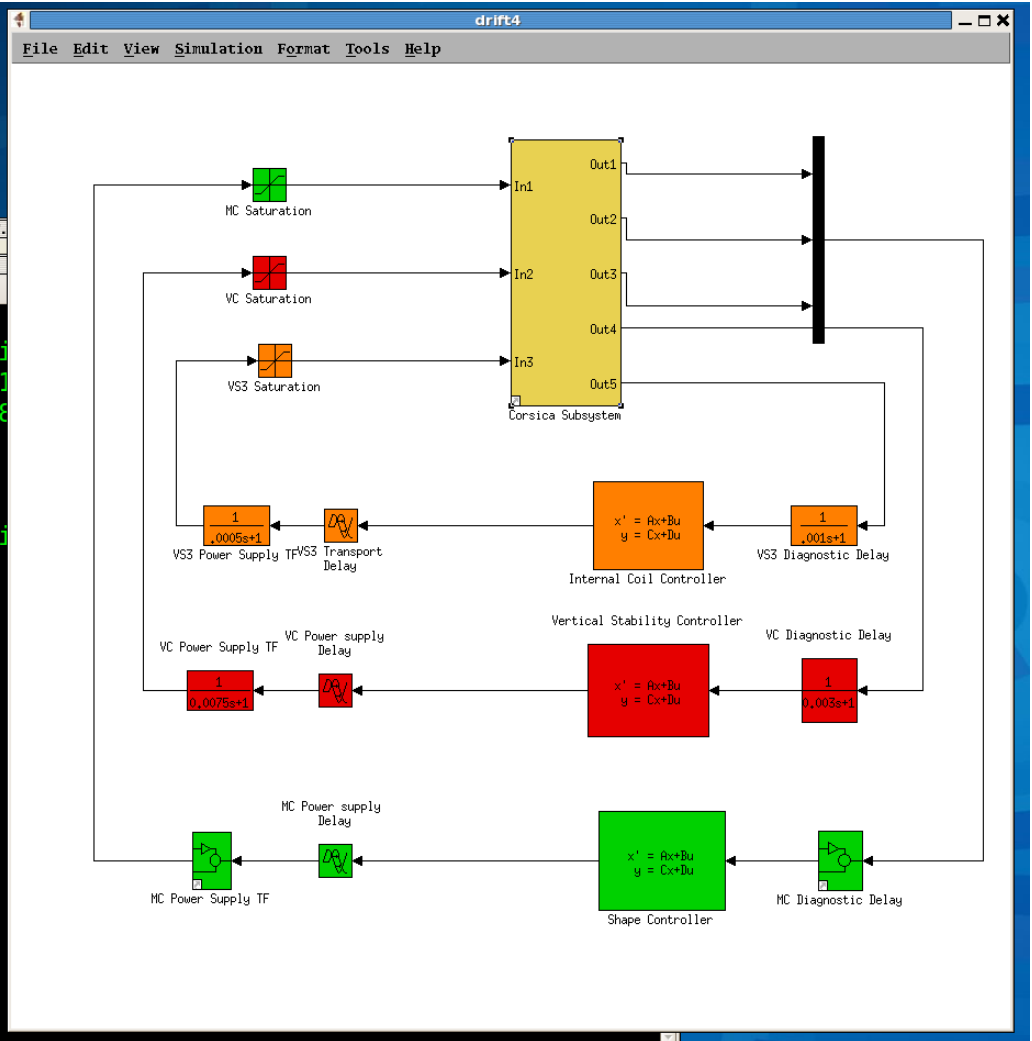
```
mdl
mdl.org
id10011
id20538
2.m
.m
l.orig
```

atlab

```
bash-3.2$
bash-3.2$ cat startup.m

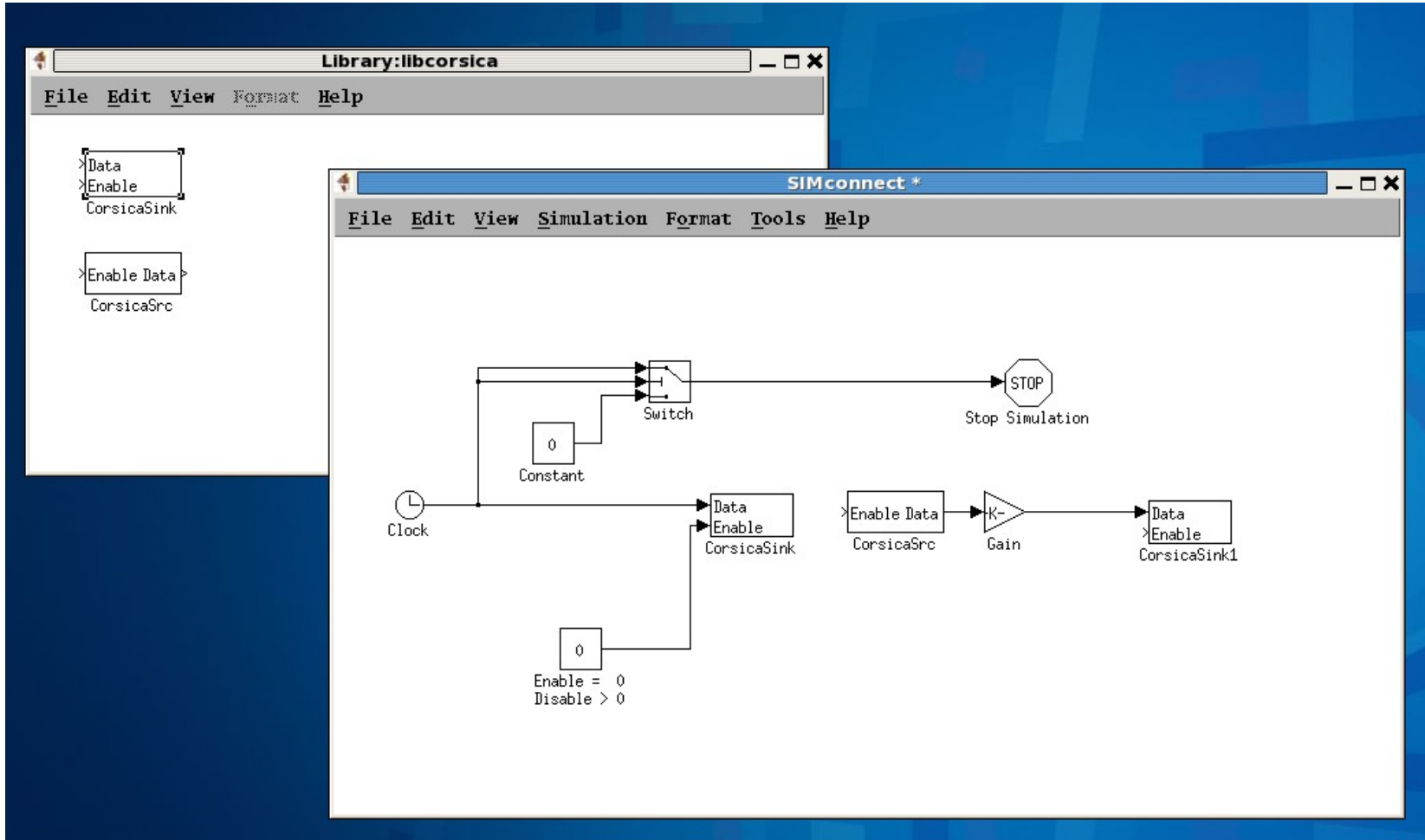
load contia.mat
load lqgp1.mat
bash-3.2$ cat start_sim.m
```

```
corsicaconnect('fepilx',1609)
corsicaparser('mdlInitializeSizes')
bash-3.2$
```



What is the value of CorsicaSink1 input?

- Clock->CorsicaSink(foobar), CorsicaSrc(foobar)->Gain->CorsicaSink1 (foobar2)
- What if CorsicaSrc->Gain->CorsicaSink1 executes before CorsicaSink



Corsica Subsystem Block Priorities

The screenshot illustrates the configuration of Corsica Subsystem blocks. The main window shows a subsystem diagram with five source blocks (In1 to In5) and five sink blocks (mc, vs, vs3, mt, and an unlabeled sink). The 'Sink Block Parameters' window for 'mc' shows the S-function name 'corsicasink' and parameters 'mc', '1', and an empty string. The 'Source Block Parameters' window for 'ddz' shows the S-function name 'corsicasource' and parameters 'ddz', '1', and an empty string. The 'Block Properties' window for 'mt' shows the S-function name 'corsicasink' and parameters 'mt', '1', and 'return'.

Sink Block Parameters: mc

S-Function

User-definable block. Blocks can be written in C, M (level-1), Fortran, and Ada and must conform to S-function standards. The variables t, x, u, and flag are automatically passed to the S-function by Simulink. You can specify additional parameters in the 'S-function parameters' field. If the S-function block requires additional source files for the Real-Time Workshop build process, specify the filenames in the 'S-function modules' field; do not use extensions or full pathnames, e.g., enter 'src1.c'.

Parameters

S-function name: corsicasink

S-function parameters: 'mc', '1', ''

S-function modules: ''

Priority: 1

Tag: ''

Source Block Parameters: ddz

S-Function

User-definable block. Blocks can be written in C, M (level-1), Fortran, and Ada and must conform to S-function standards. The variables t, x, u, and flag are automatically passed to the S-function by Simulink. You can specify additional parameters in the 'S-function parameters' field. If the S-function block requires additional source files for the Real-Time Workshop build process, specify the filenames in the 'S-function modules' field. Enter the filenames only; do not use extensions or full pathnames, e.g., enter 'src1.c', not 'src.c src1.c'.

Parameters

S-function name: corsicasource

S-function parameters: 'ddz', '1', ''

S-function modules: ''

Priority: 3

Tag: ''

Block Properties: mt

S-Function

User-definable block. Blocks can be written in C, M (level-1), Fortran, and Ada and must conform to S-function standards. The variables t, x, u, and flag are automatically passed to the S-function by Simulink. You can specify additional parameters in the 'S-function parameters' field. If the S-function block requires additional source files for the Real-Time Workshop build process, specify the filenames in the 'S-function modules' field. Enter the filenames only; do not use extensions or full pathnames, e.g., enter 'src1.c', not 'src.c src1.c'.

Parameters

S-function name: corsicasink

S-function parameters: 'mt', '1', 'return'

S-function modules: ''

Priority: 2

Tag: ''

Matlab/Corsica Timing

Start.. ..Start
Send MC (Blocked).. ..Trans(.....)
 (Blocked).. ..Calculate local control voltages
 (Blocked).. ..Enter Control Hook
 (Blocked).. ..Copy Diagnostics into RPC Vars
Send VS.. ..Rpcserver (Blocked)
Send VS3.. ..(Blocked)
Send Simulation Time and Return.. ..(Blocked until return received)
Get Gaps (Block).. .. Complete step
 (Blocked).. .. Start next step
 (Blocked).. ..Enter Control Hook
 (Blocked).. ..Copy Diagnostics into RPC Vars
Get rest of diagnostics.. ..Rpcserver (Blocked)
Solve tranfer functions/state models.. ..(Blocked)
Send MC.. ..(Blocked)