



Wireless Communications

Introduction to MATLAB & Simulink

Lab Assistant: Davide Scazzoli
(davide.scazzoli@polimi.it)



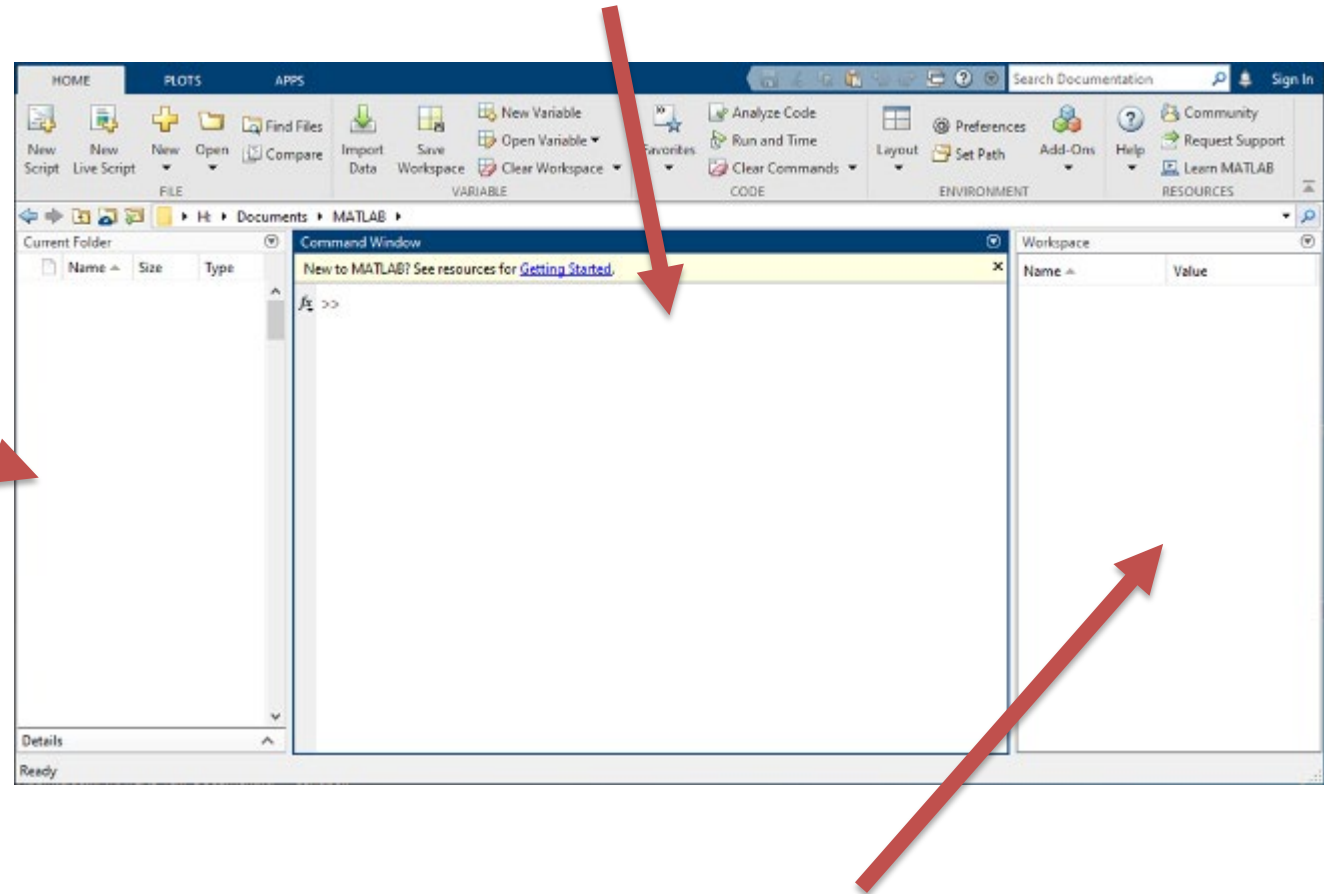
POLITECNICO
MILANO 1863



MATLAB Introduction

Console: executes code line by line

File browser:
browse files
such as scripts
and functions



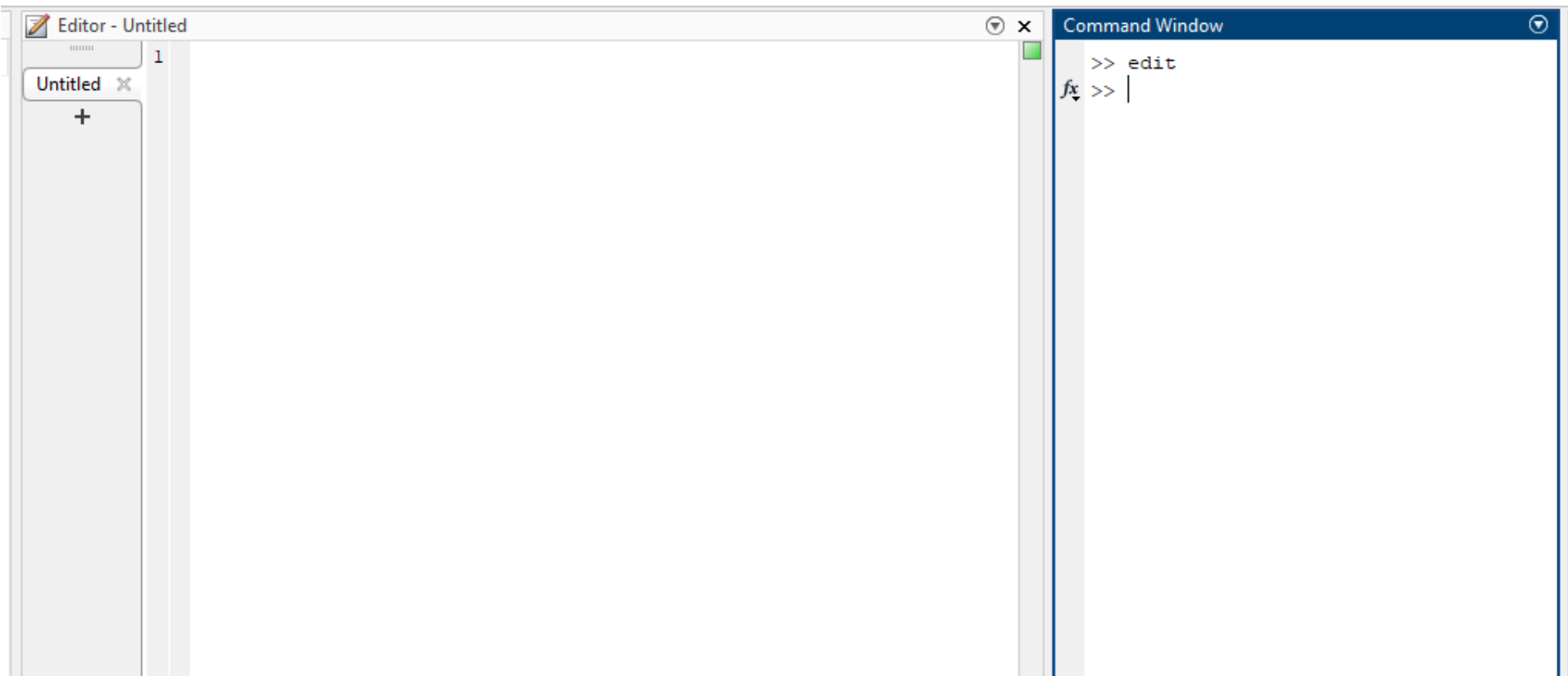
Workspace: browse variables and their value





MATLAB Introduction

Editor: for creating scripts and functions, opens when you open a file or by typing **edit** in the console





MATLAB Introduction

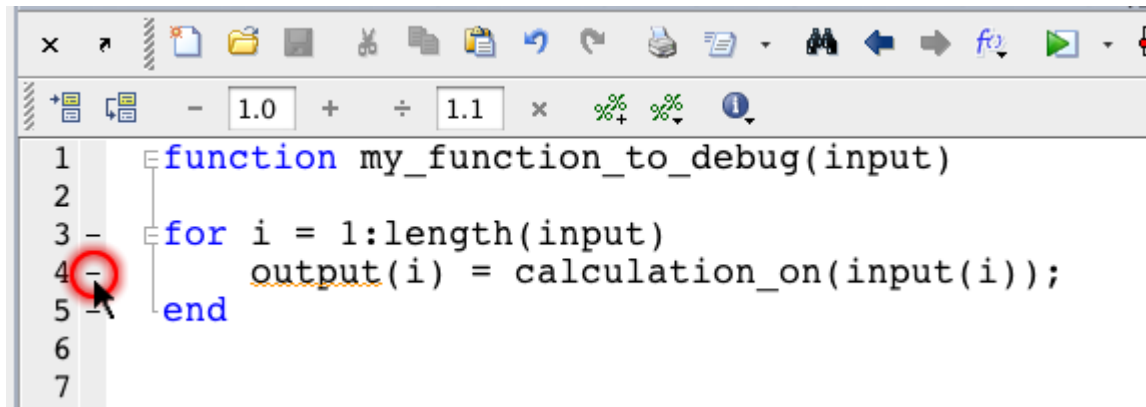
1. After running a script or command all the variables are left in the workspace. These leftover variables may interfere with your next simulations. If you want to clear them type **clear all**.
2. If your console is full of messages and you want to clear it you can type **clc**.
3. If there are many windows open you can close them all by typing **close all**.





MATLAB Introduction

You can set up breakpoints in the editor to examine the state of the variables at any point in your code



The image shows a screenshot of the MATLAB editor window. The code being edited is a function named `my_function_to_debug`. The code is as follows:

```
1 function my_function_to_debug(input)
2
3 for i = 1:length(input)
4     output(i) = calculation_on(input(i));
5 end
6
7
```

A red circle highlights the minus sign in the line number column next to line 4, indicating that a breakpoint has been set at this location. The MATLAB toolbar and calculator are visible at the top of the window.



MATLAB Introduction

You can declare structures of variables to very quickly pass data to a function

```
%% Define constants and geometry
Pars.fc=1e9; %Carrier frequency
Pars.c = physconst('LightSpeed');
Pars.lambda = Pars.c/Pars.fc;

%Define geometry of the problem (xyz coordinates)
G.BSPos=[0,0,25]; % 25m is a typical height for a ma
G.V1PosStart=[70,-100,1.5]; %Start position fo Vehic
G.V1PosEnd=[70,100,1.5]; %End position for vehicl
G.V2PosStart=[200,-50,1.5];
G.V2PosEnd=[10,-50,1.5];
G.I1Pos=[10,-210,1.5];
G.I2Pos=[-150,100,1.5];

% Calculate distances as sqrt((x1-x2)^2+(y1-y2)^2)
G.T1=sqrt(sum((G.V1PosEnd(1,1:2) -...
    G.V1PosStart(1,1:2)).^2));
G.T2=sqrt(sum((G.V2PosEnd(1,1:2) -...
    G.V2PosStart(1,1:2)).^2));
G.DistV1Start=sqrt(sum((G.V1PosStart(1,1:2) - ...
    G.BSPos(1,1:2)).^2));
G.DistV2Start=sqrt(sum((G.V2PosStart(1,1:2) - ...
    G.BSPos(1,1:2)).^2));
```

CreateScenarioAndVisualize(G, Pars);





MATLAB Introduction

The help function returns a short description for a function. If nothing is returned then the function is part of a package that is not installed!

```
>> help atan2
atan2  Four quadrant inverse tangent.
      atan2(Y,X) is the four quadrant arctangent of the elements of X and Y
      such that  $-\pi \leq \text{atan2}(Y,X) \leq \pi$ . X and Y must have compatible sizes.
      In the simplest cases, they can be the same size or one can be a
      scalar. Two inputs have compatible sizes if, for every dimension, the
      dimension sizes of the inputs are either the same or one of them is 1.

      See also atan, atan2d.

      Documentation for atan2
      Other functions named atan2

>> |
```





SIMULINK Introduction

Simulink[®] is a software package for modeling, simulating, and analyzing dynamic systems.

It supports linear and nonlinear systems, modeled in continuous time, sampled time, or a hybrid of the two.

Systems can be multirate, i.e., have different parts that are sampled or updated at different rates.

Simulink[®] is integrated with MATLAB[®], providing immediate access to an extensive range of tools that let's you to develop algorithms, analyze and visualize simulations, create batch processing scripts, customize the modeling environment, and define signal, parameter, and test data.





SIMULINK Introduction

Simulink provides a Graphical User Interface for building models as block diagrams, using click-and-drag mouse operations.

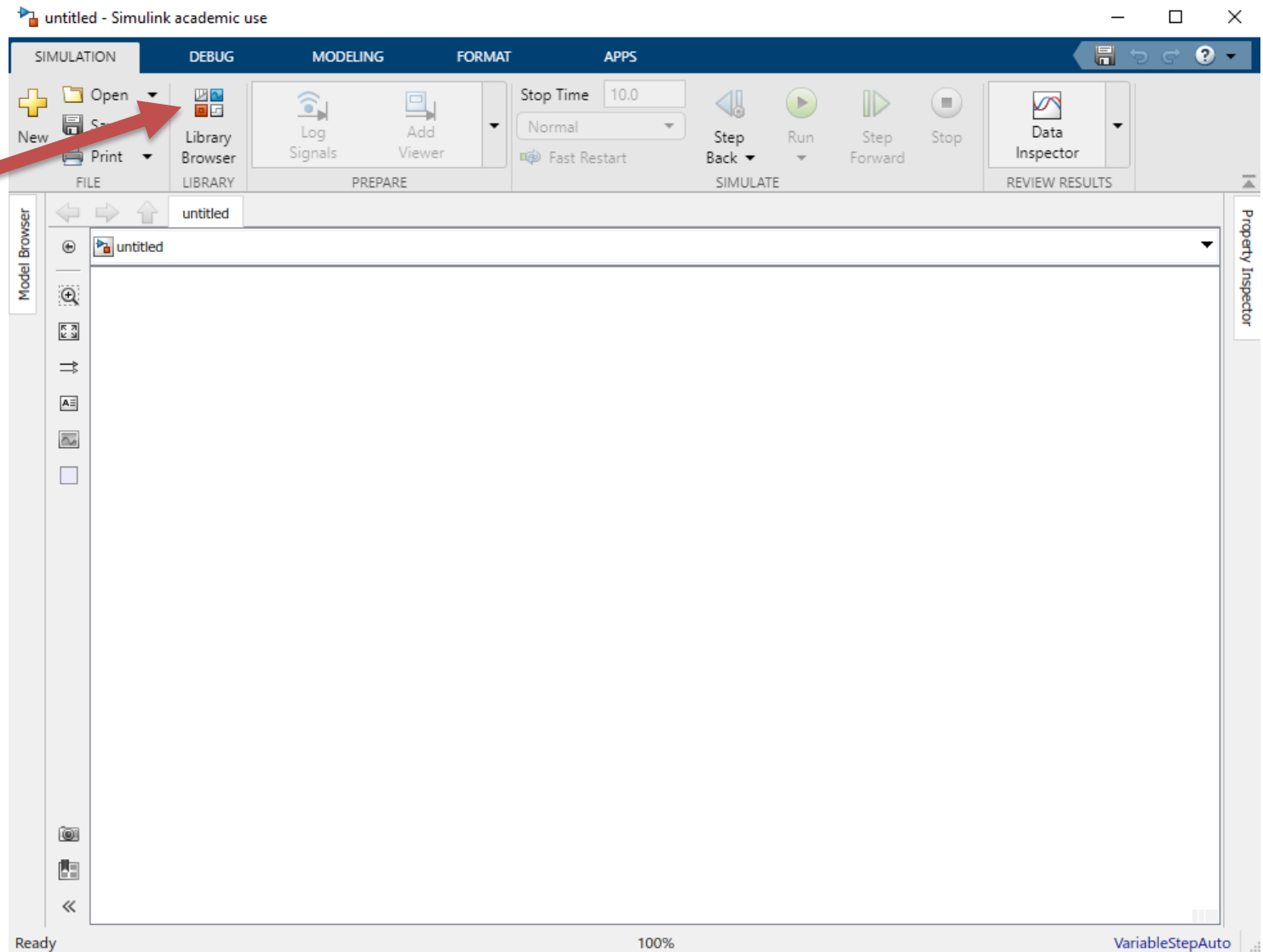
Basic elements:

1. **BLOCKS:** used to generate, modify, combine, output and display signals.
2. **LINES:** used to transfer signals from one block to another.



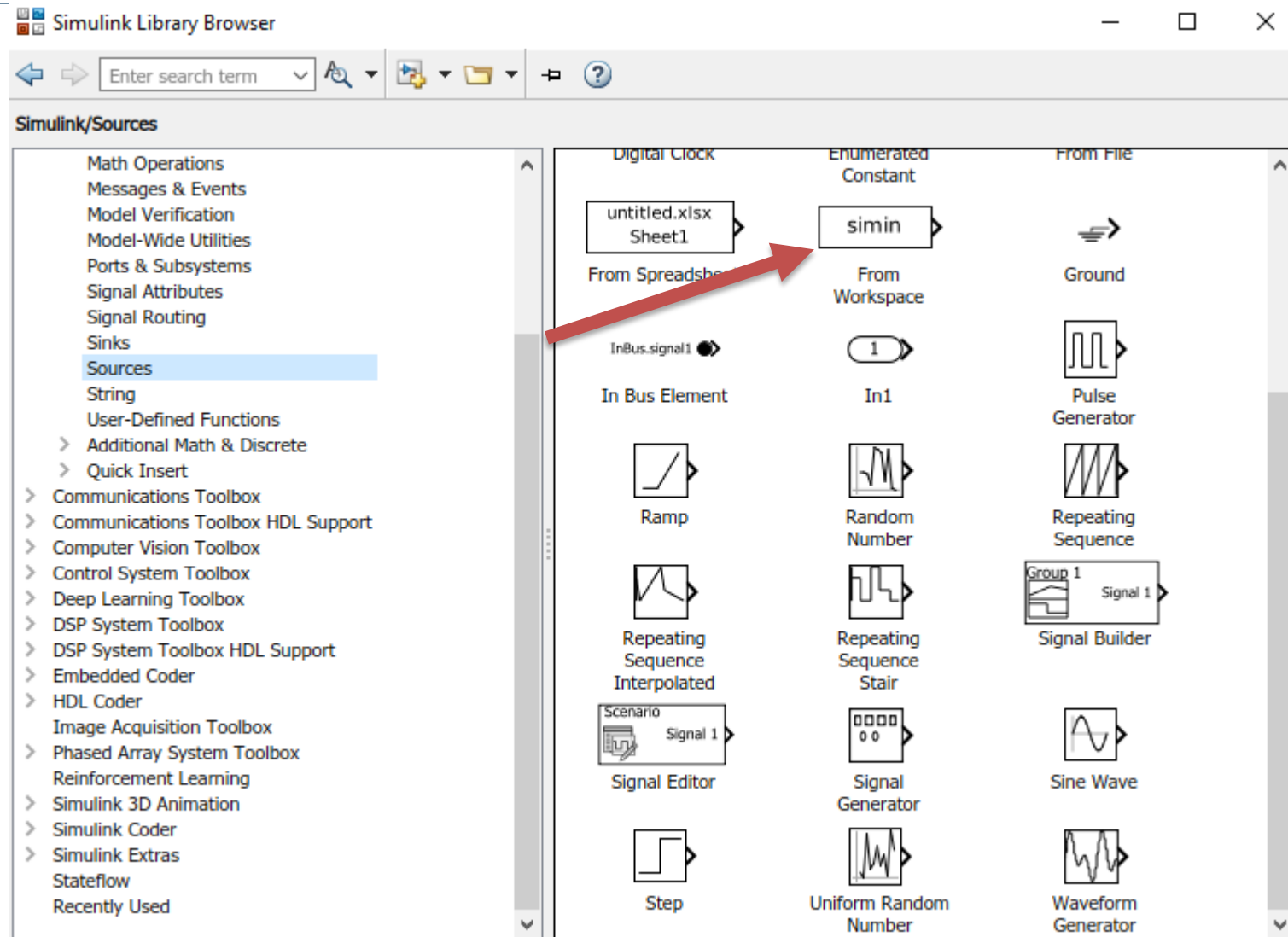
SIMULINK Introduction

Open the
Library
Browser to
drag and drop
blocks in your
model
window



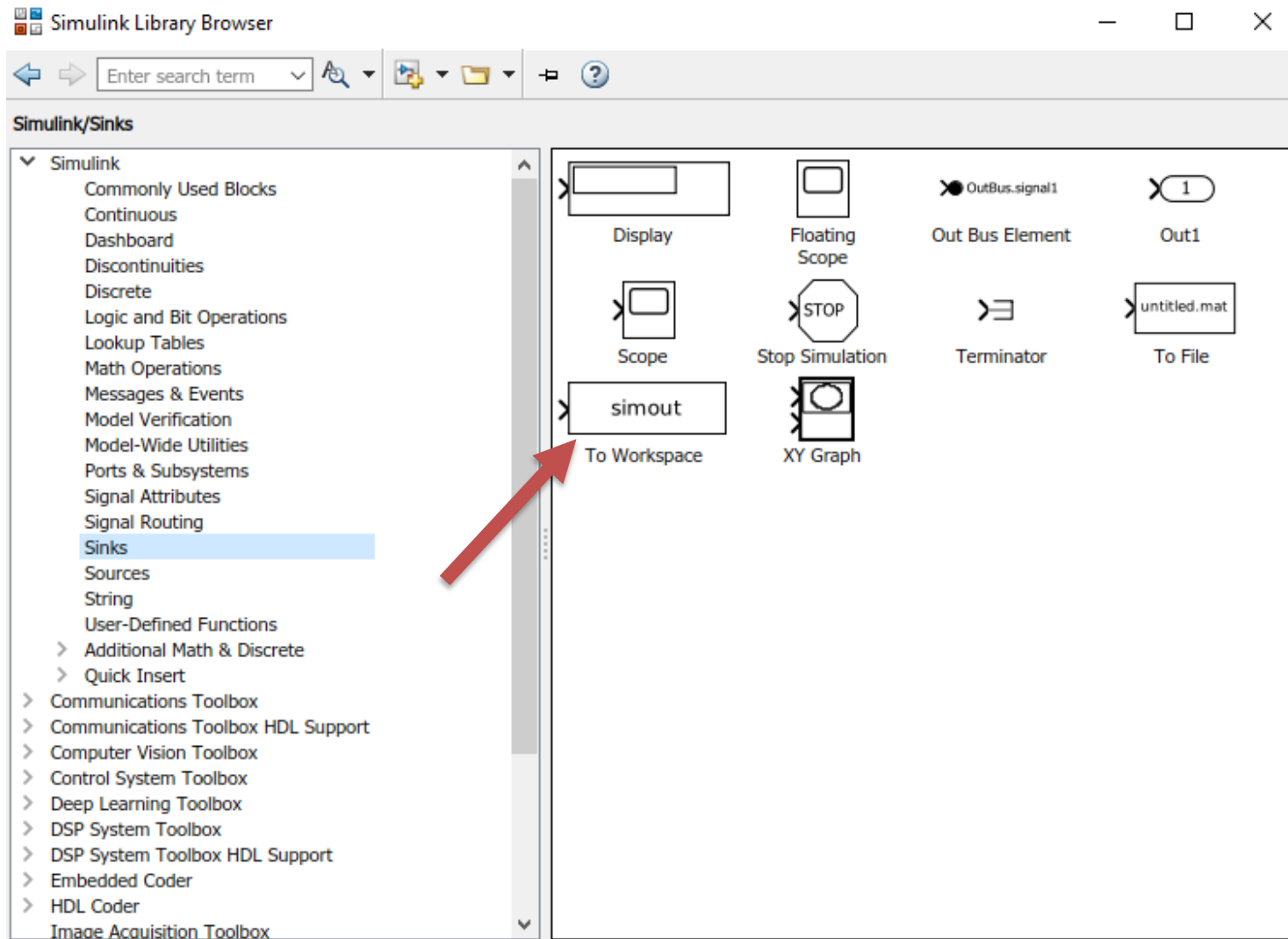
SIMULINK Introduction

Sources
create signals
such as
sinusoids,
random bits
or take from
matlab
workspace
variables



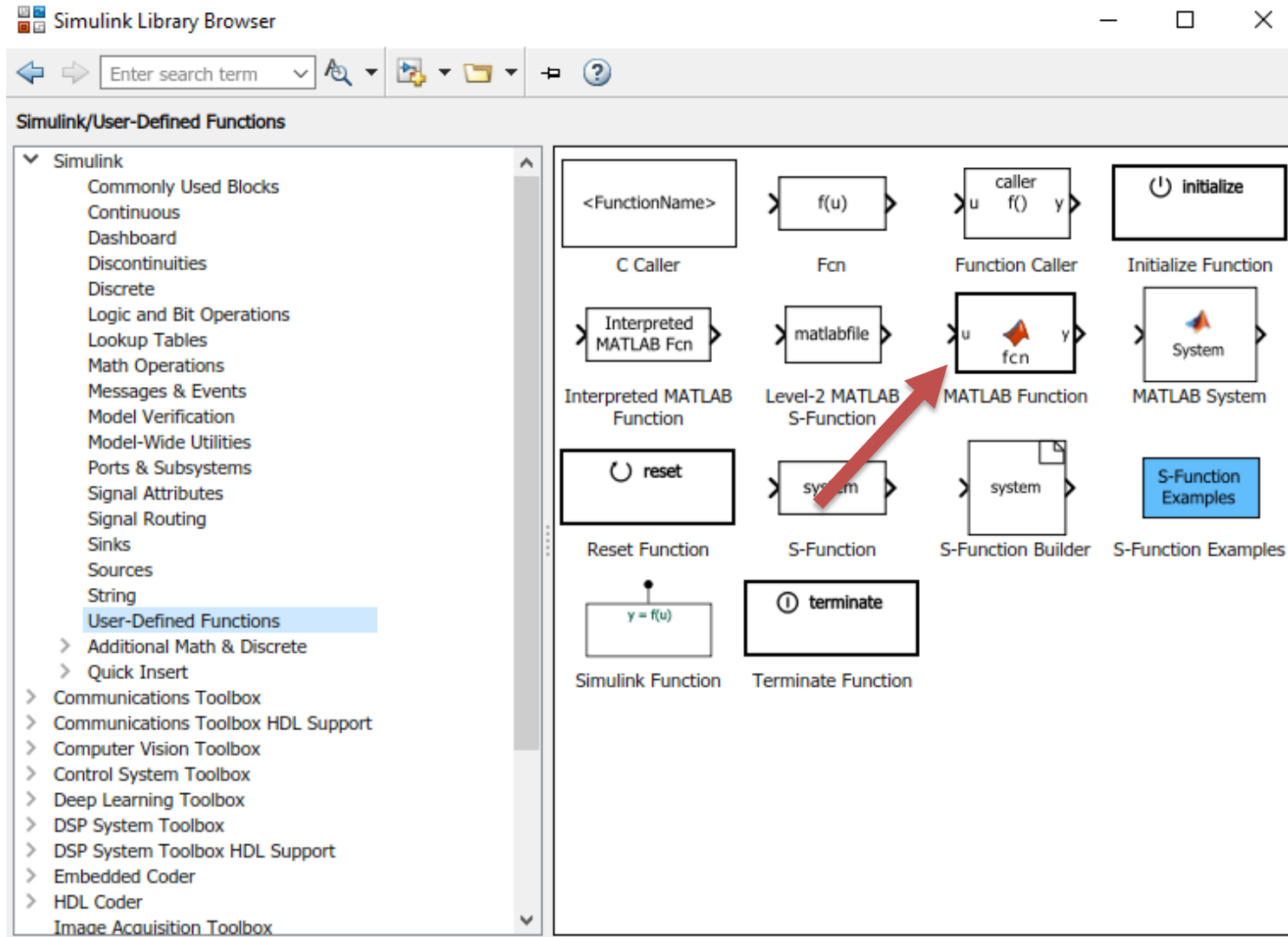
SIMULINK Introduction

Sinks do the opposite, taking signals as input for various types of visualization or return them to the matlab workspace as variables



SIMULINK Introduction

It is possible to use MATLAB defined functions in Simulink using the appropriate block





SIMULINK Introduction

- ▶ Every Simulink® block is considered to have a **sample time**.
- ▶ **Discrete-time blocks**: allows to specify their sample time via a Sample Time parameter.
- ▶ **Continuous-time blocks**: have an infinitesimal sample time called continuous sample time.





SIMULINK Introduction

The most frequently used libraries for analyzing communication systems are

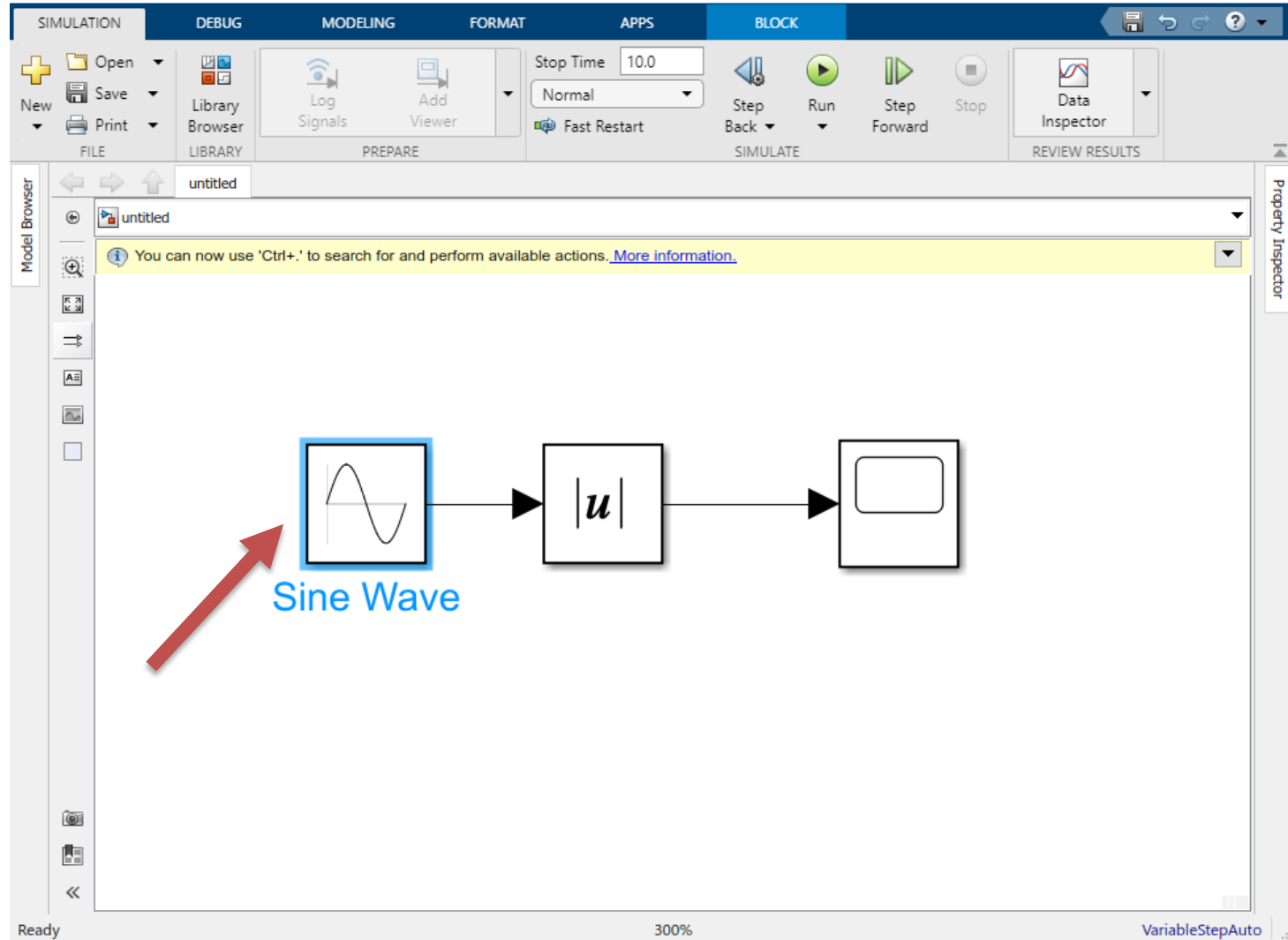
- ▶ Simulink: math operations, sinks, sources, portes & subsystems, etc.
- ▶ Communication System Toolbox: channels, modulation, error detection and correction, equalizers, interleaving, synchronization, etc.
- ▶ DSP System Toolbox: filtering, math functions, quantizers, signal management, estimation, transforms, statistics, etc.

Custom blocks can be also created graphically by drawing a block diagram representing the blocks' behavior.




SIMULINK Introduction

Let's make a
simple
example:
one sinusoid
of 1hz
frequency,
calculate the
abs and
visualize



SIMULINK Introduction

To edit block parameter double click a block. For the sine wave block the following pop-up will display:



Block Parameters: Sine Wave

Sine Wave

Output a sine wave:

$$O(t) = \text{Amp} * \sin(\text{Freq} * t + \text{Phase}) + \text{Bias}$$

Sine type determines the computational technique used. The parameters in the two types are related through:

$$\text{Samples per period} = 2 * \pi / (\text{Frequency} * \text{Sample time})$$
$$\text{Number of offset samples} = \text{Phase} * \text{Samples per period} / (2 * \pi)$$

Use the sample-based sine type if numerical problems due to running for large times (e.g. overflow in absolute time) occur.

Parameters

Sine type: Time based

Time (t): Use simulation time

Amplitude: 1

Bias: 0

Frequency (rad/sec): $2 * \pi$

Phase (rad): 0

Sample time: 0

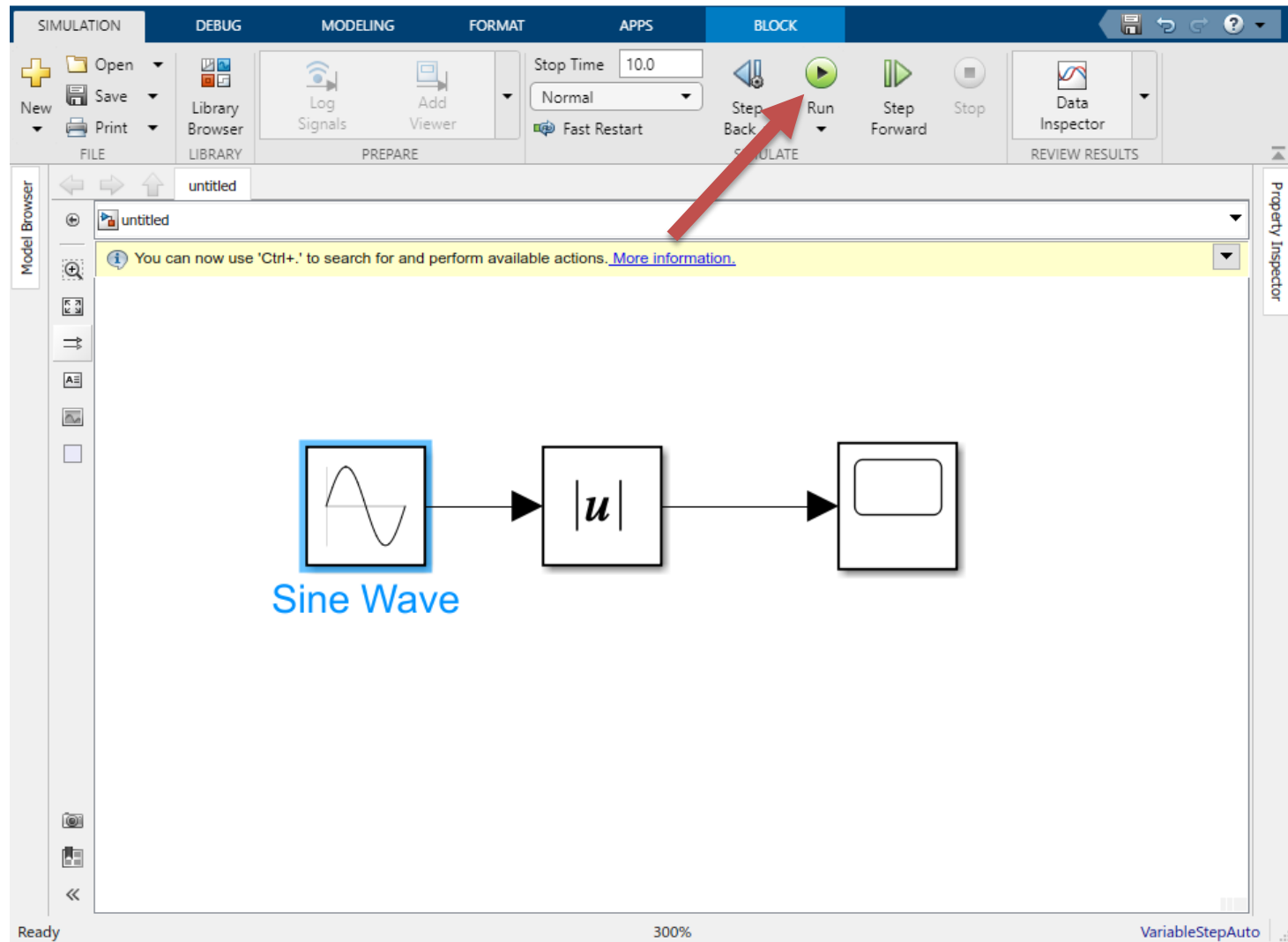
☒ Interpret vector parameters as 1-D

OK Cancel Help Apply



SIMULINK Introduction

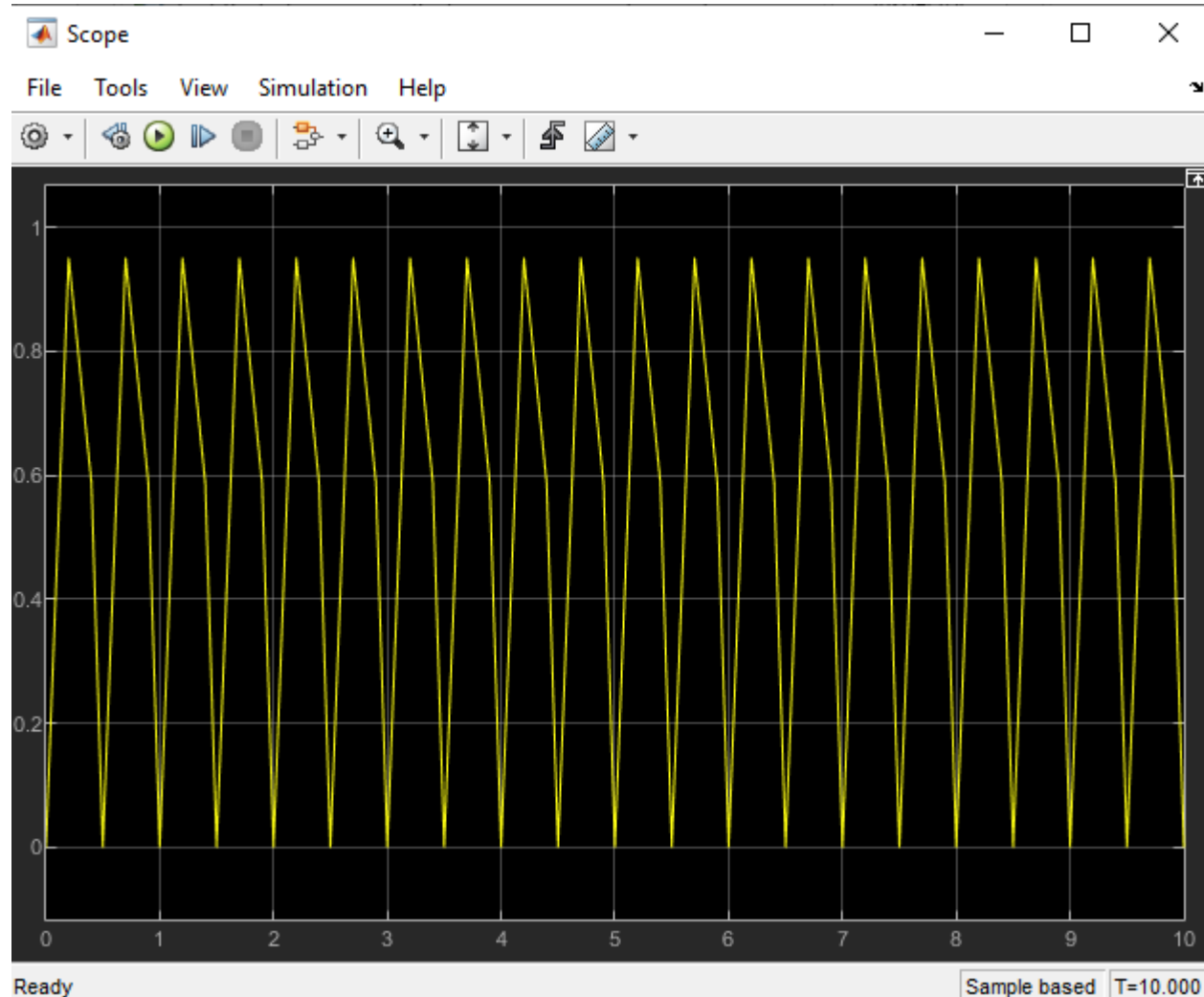
Let's run it





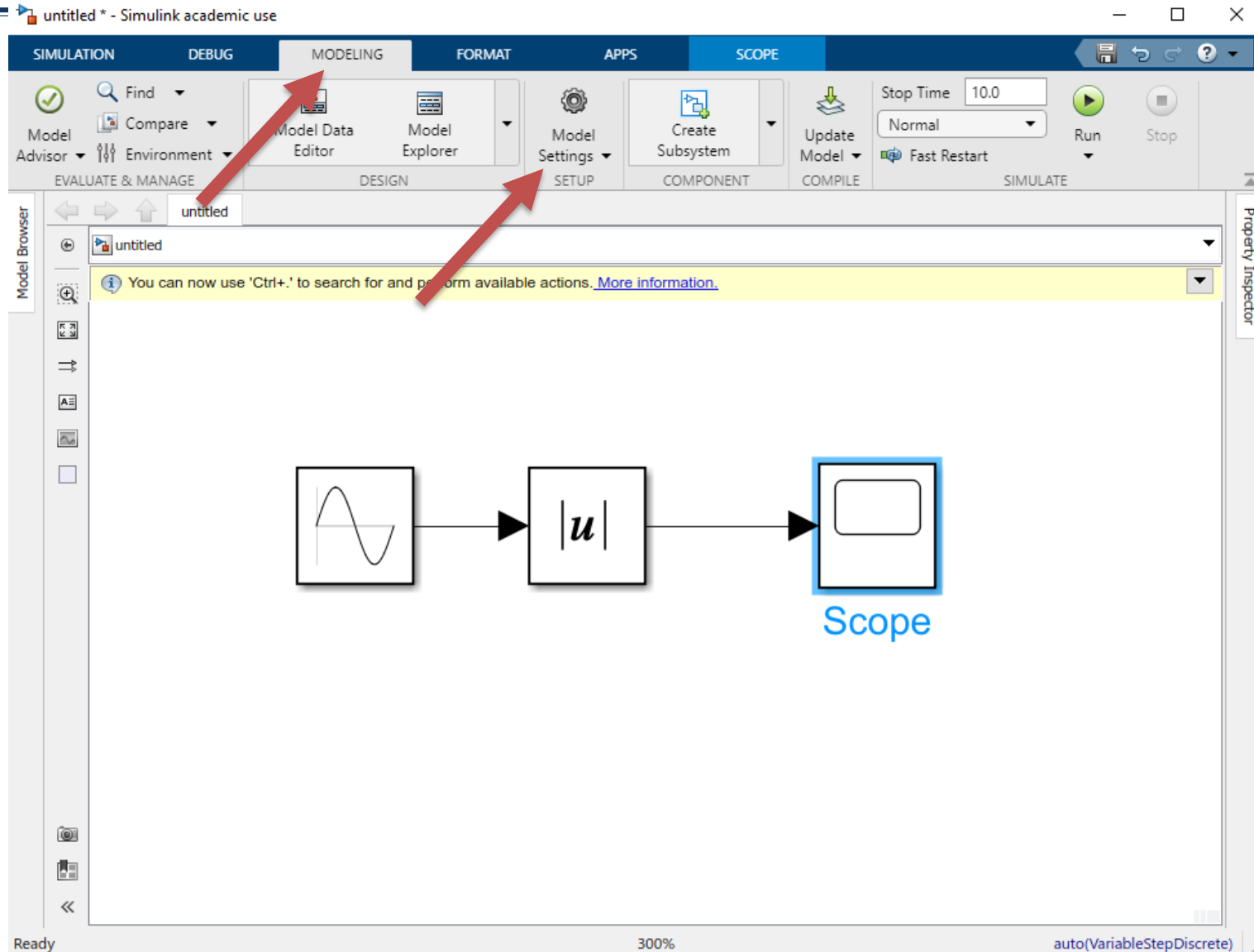
SIMULINK Introduction

Opening the scope
via double click we
get a very bad
looking sinusoid



SIMULINK Introduction

We can modify simulation parameters to enhance the quality of the sinusoid



SIMULINK Introduction

Force an appropriate step size for the fixed step size solver

Configuration Parameters: untitled/Configuration (Active)

Search

- Solver
- Data Import/Export
- Math and Data Types
- Diagnostics
- Hardware Implementation
- Model Referencing
- Simulation Target

Simulation time

Start time: 0.0 Stop time: 10.0

Solver selection

Type: Fixed-step Solver: auto (Automatic solver selection)

▼ Solver details

Fixed-step size (fundamental sample time): 1e-4

Tasking and sample time options

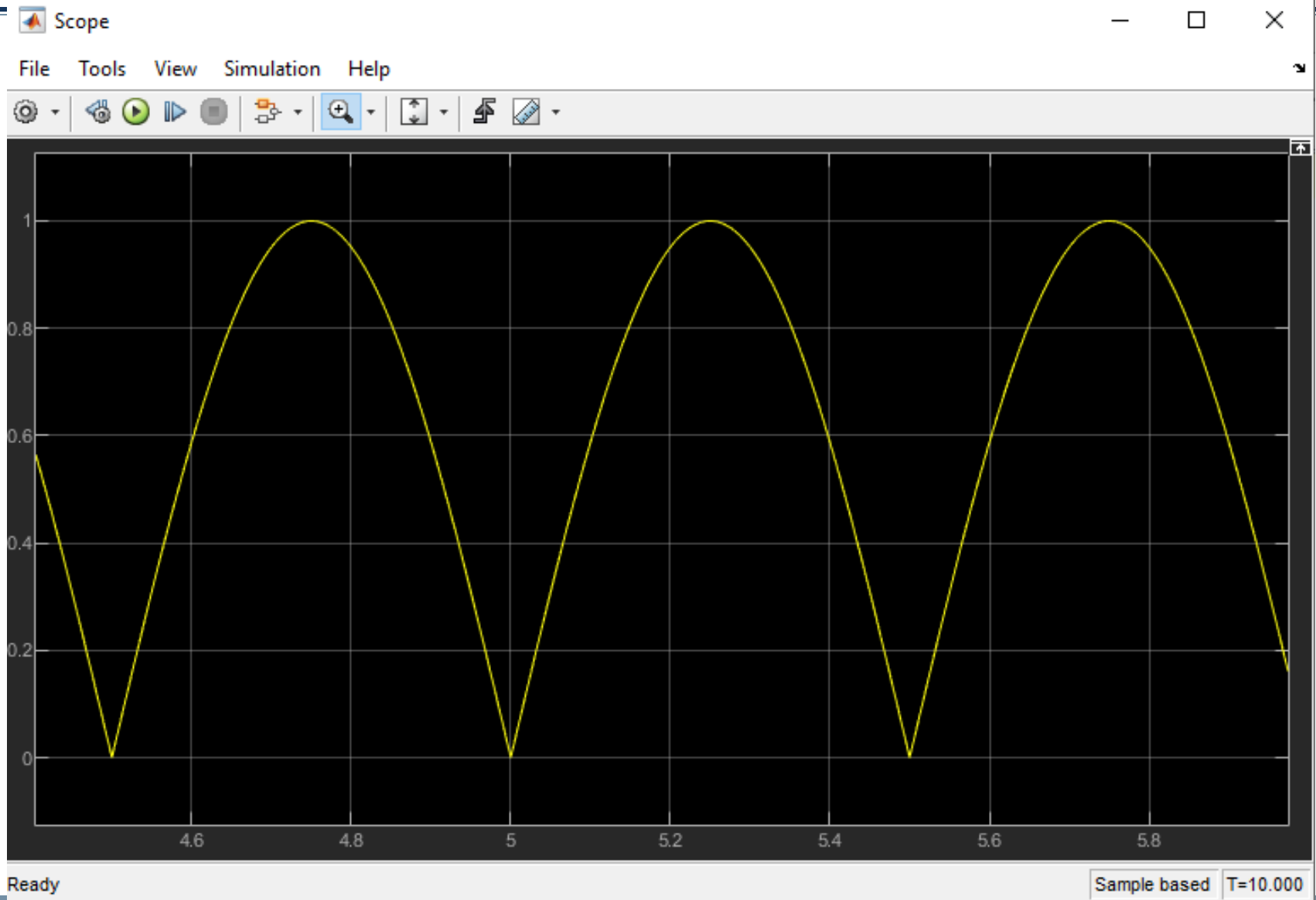
Periodic sample time constraint: Unconstrained

- ☐ Treat each discrete rate as a separate task
- ☐ Allow tasks to execute concurrently on target
- ☐ Automatically handle rate transition for data transfer
- ☐ Higher priority value indicates higher task priority

OK Cancel Help App



SIMULINK Introduction



Thank you!
Questions?

