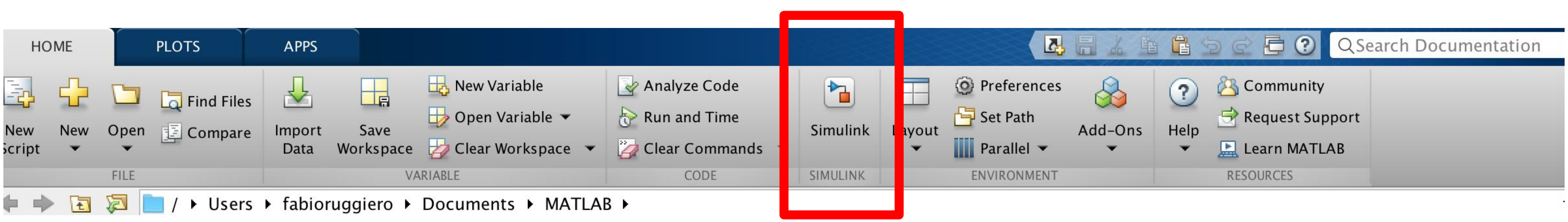
INTRODUCTION TO SIMULINK

PRISMA Lab
Dipartimento di Ingegneria Elettrica e Tecnologie dell'Informazione
Università degli Studi di Napoli Federico II
www.prisma.unina.it

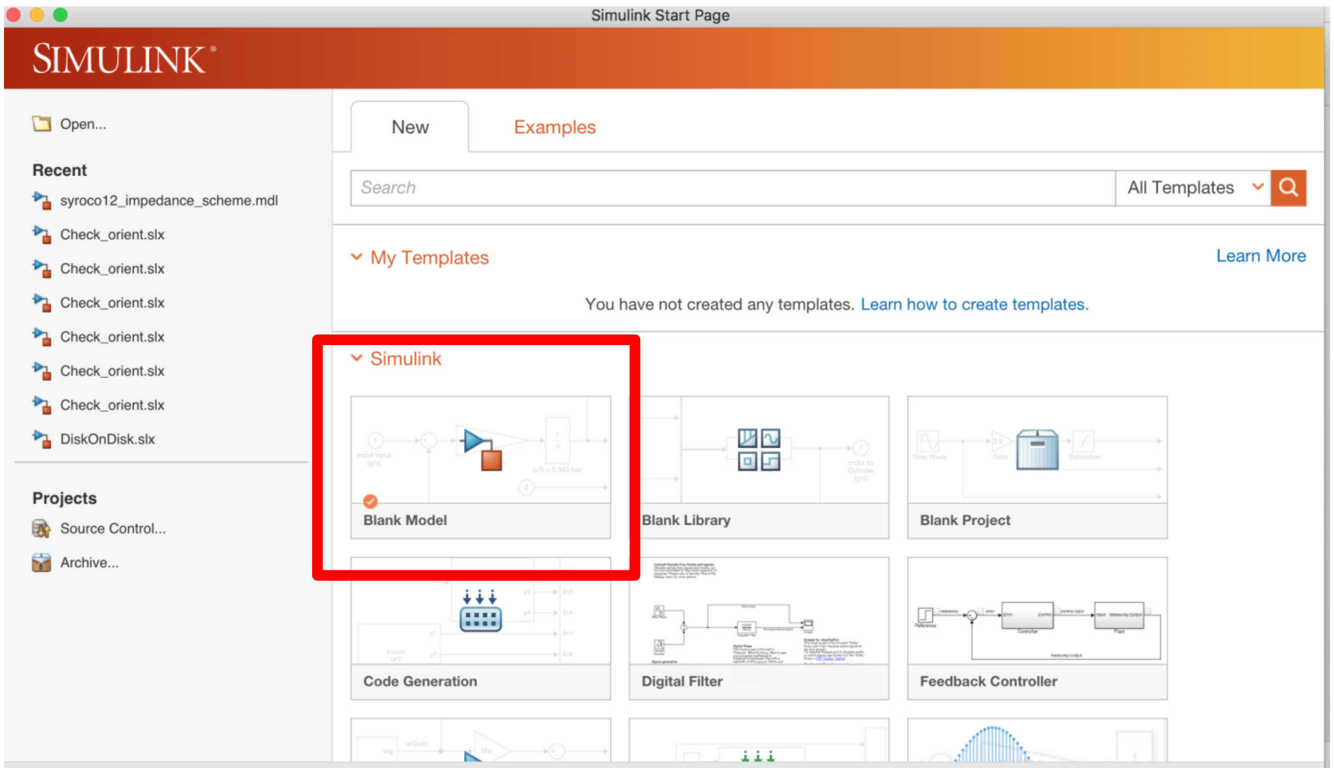
Simulink / 1

- Introduction
  - It is a software, mostly graphic, for the modeling, simulation and analysis of dynamic systems
  - Strictly integrated with Matlab
- Start
  - Type `simulink` on Matlab's command prompt
  - Click on the icon

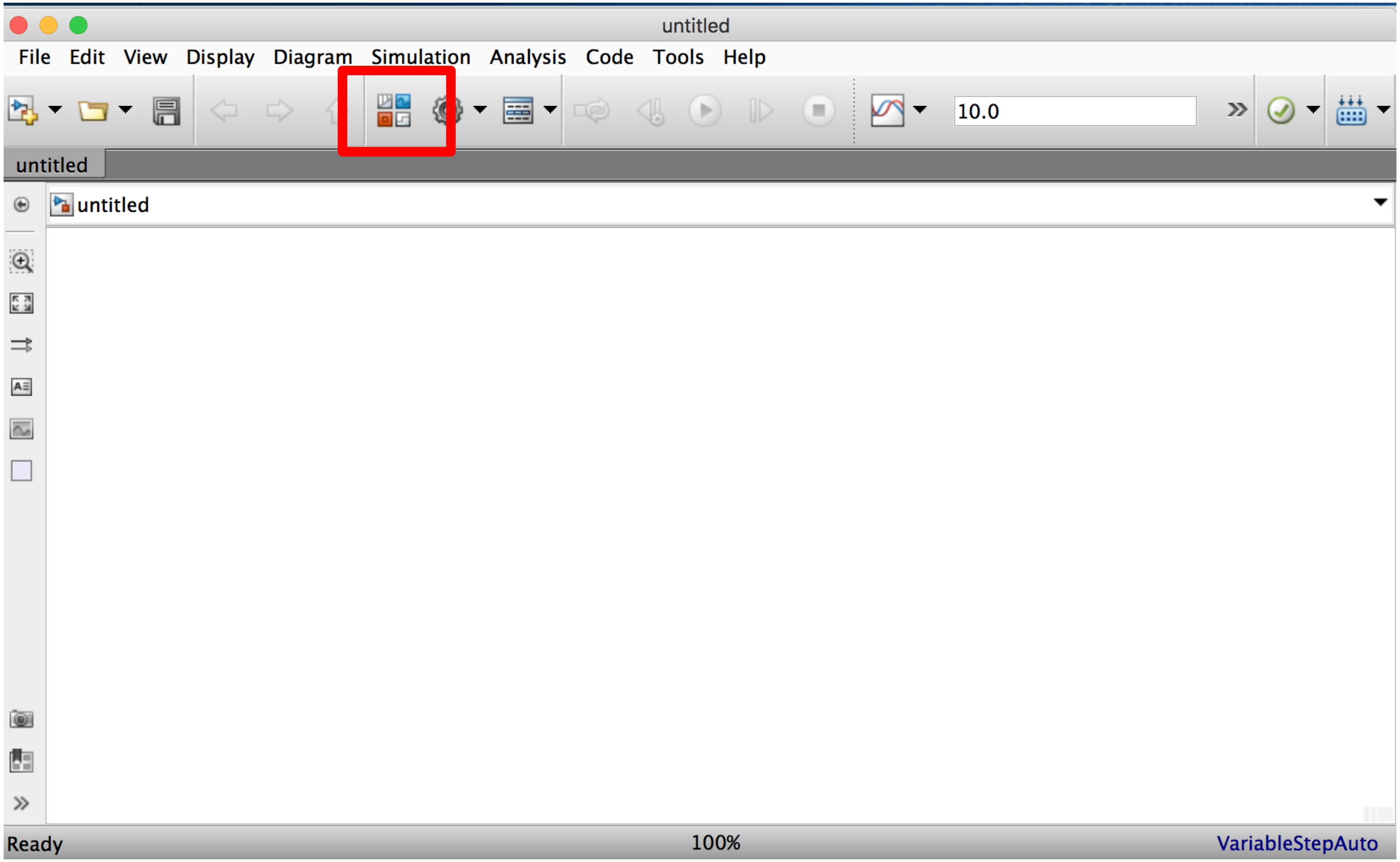


Simulink / 2

- Create a new model
  - Click on the icon `Blank Model` to generate a new file

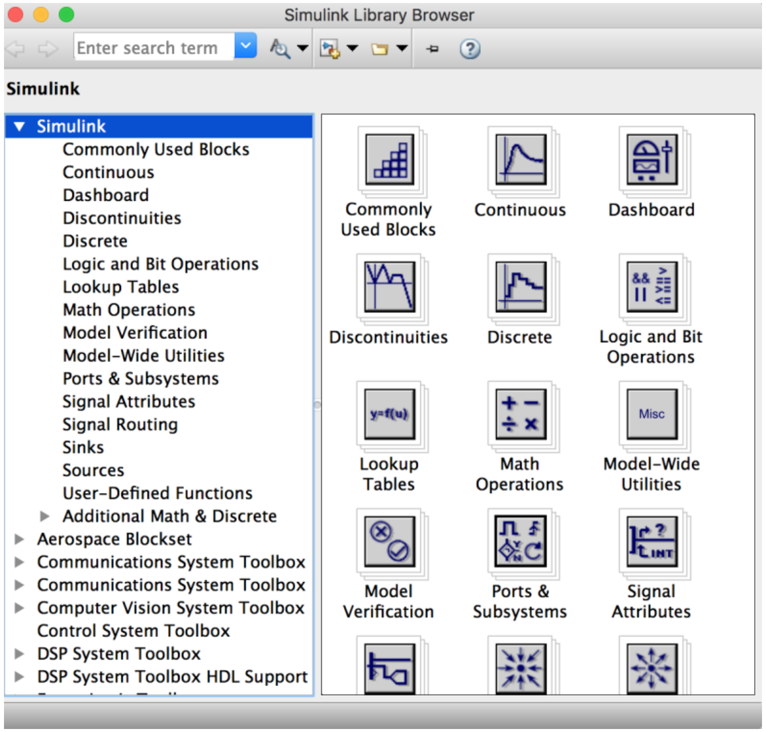


- Click on the selected icon to open the models' library

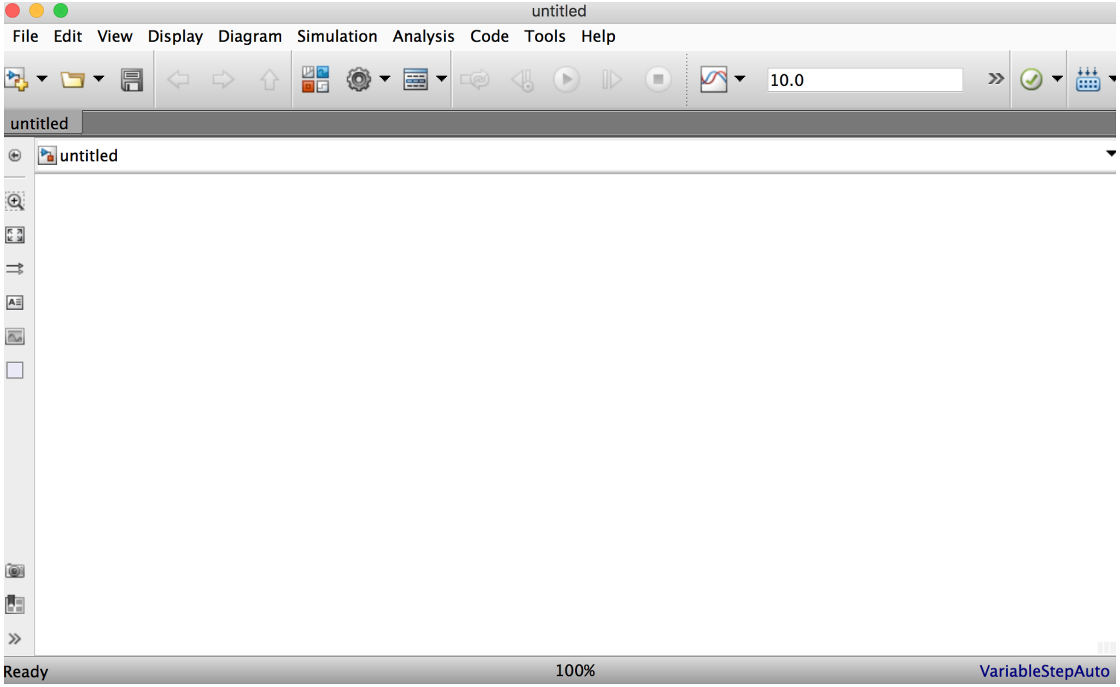


Simulink / 4

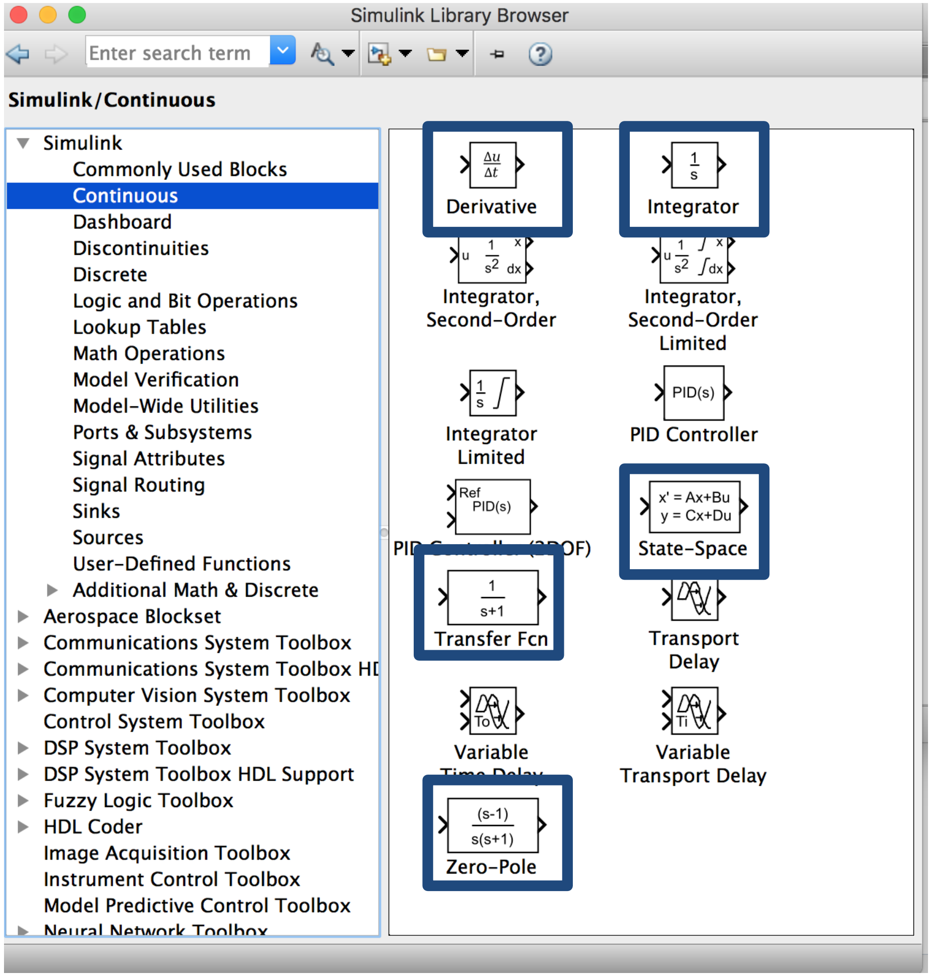
Library



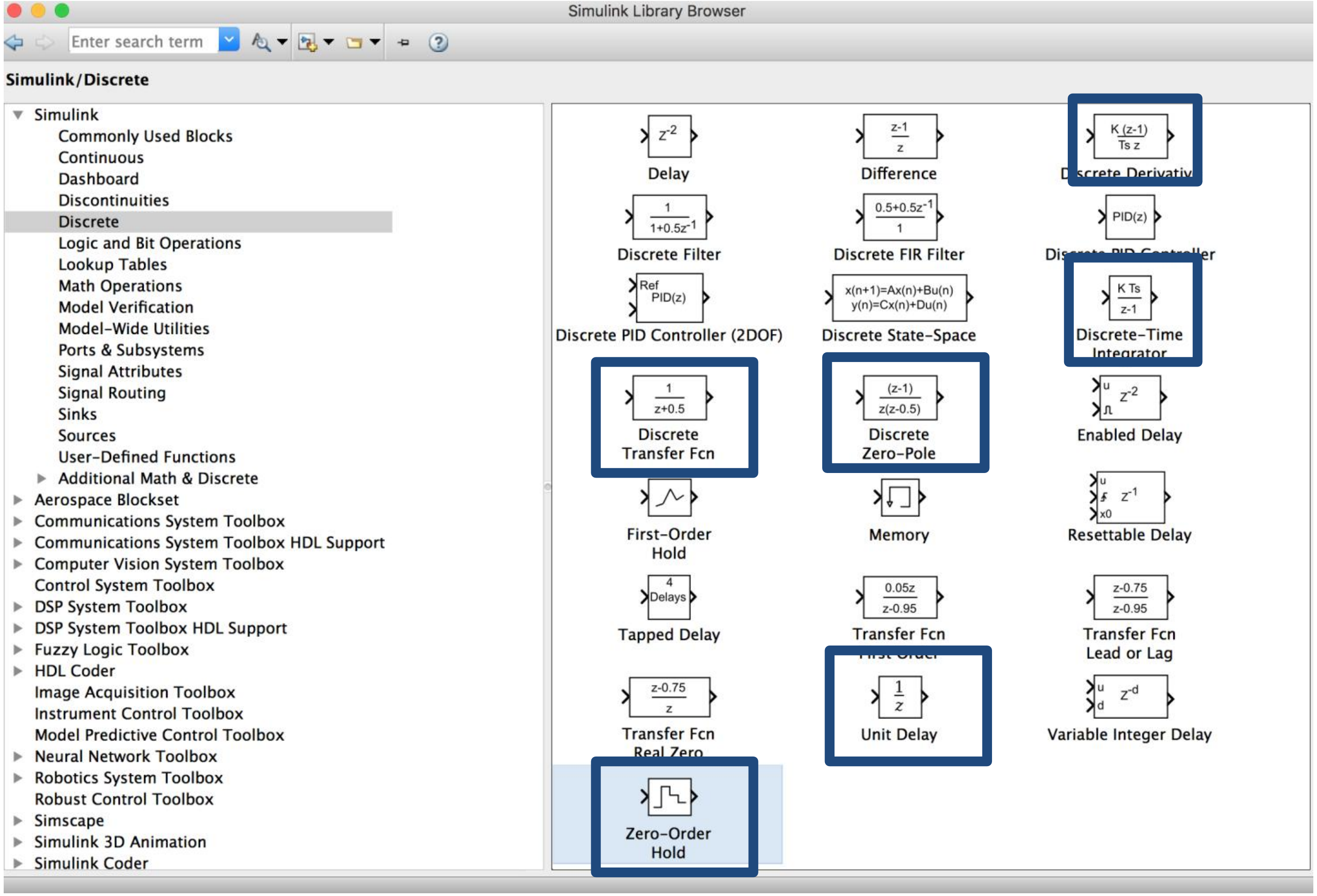
Working Window



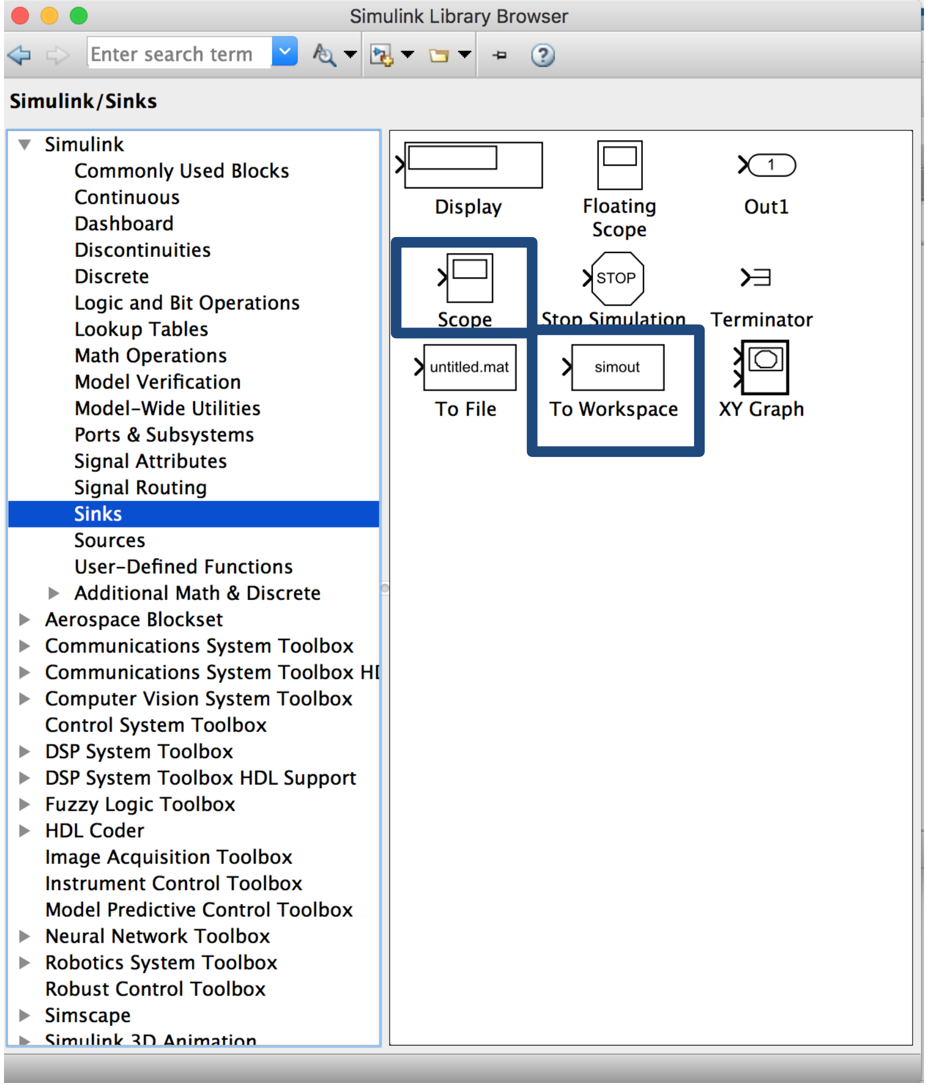
- Simulink models' library
  - Continuous time



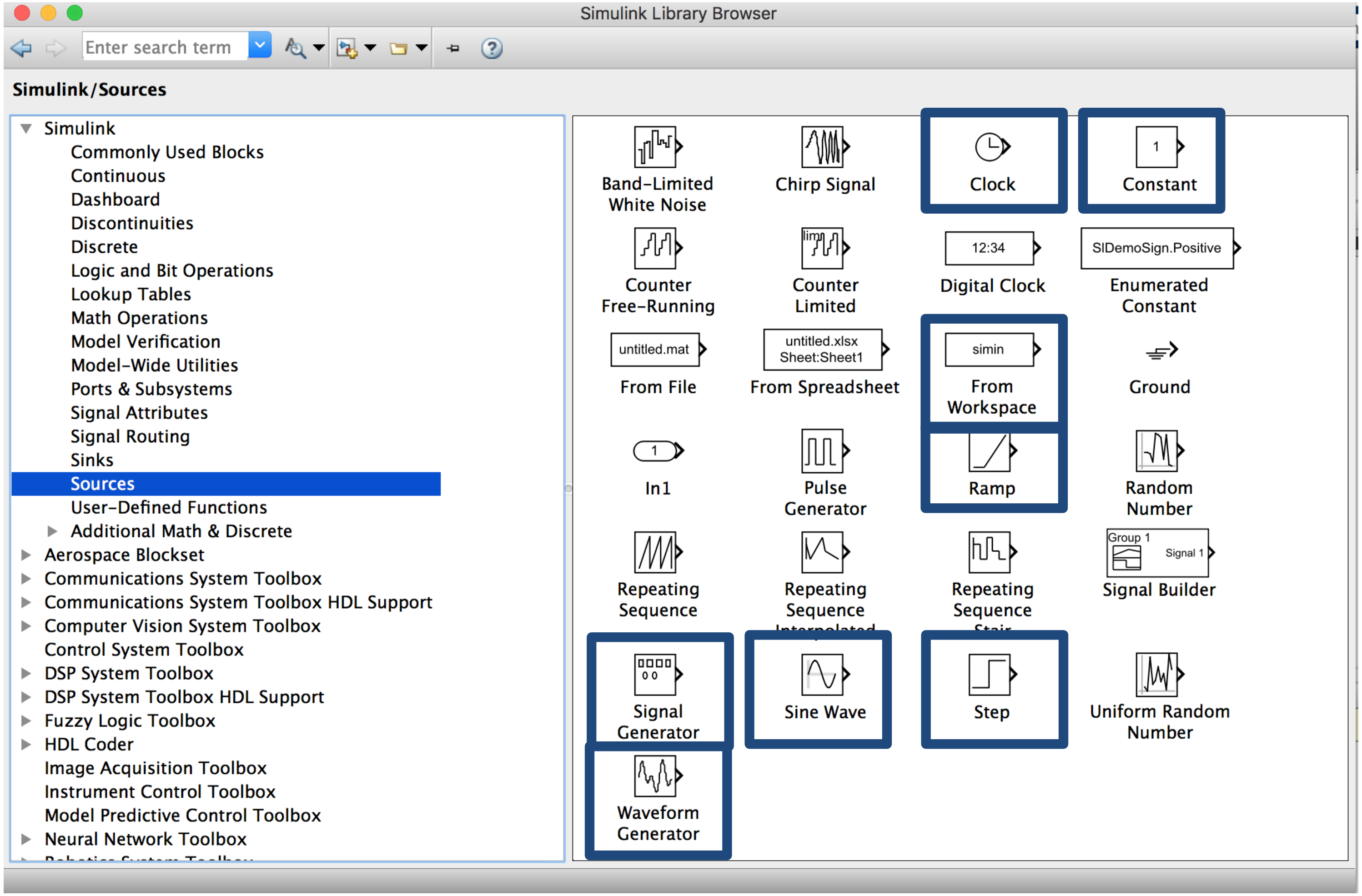
- Discrete time



- Signals visualization

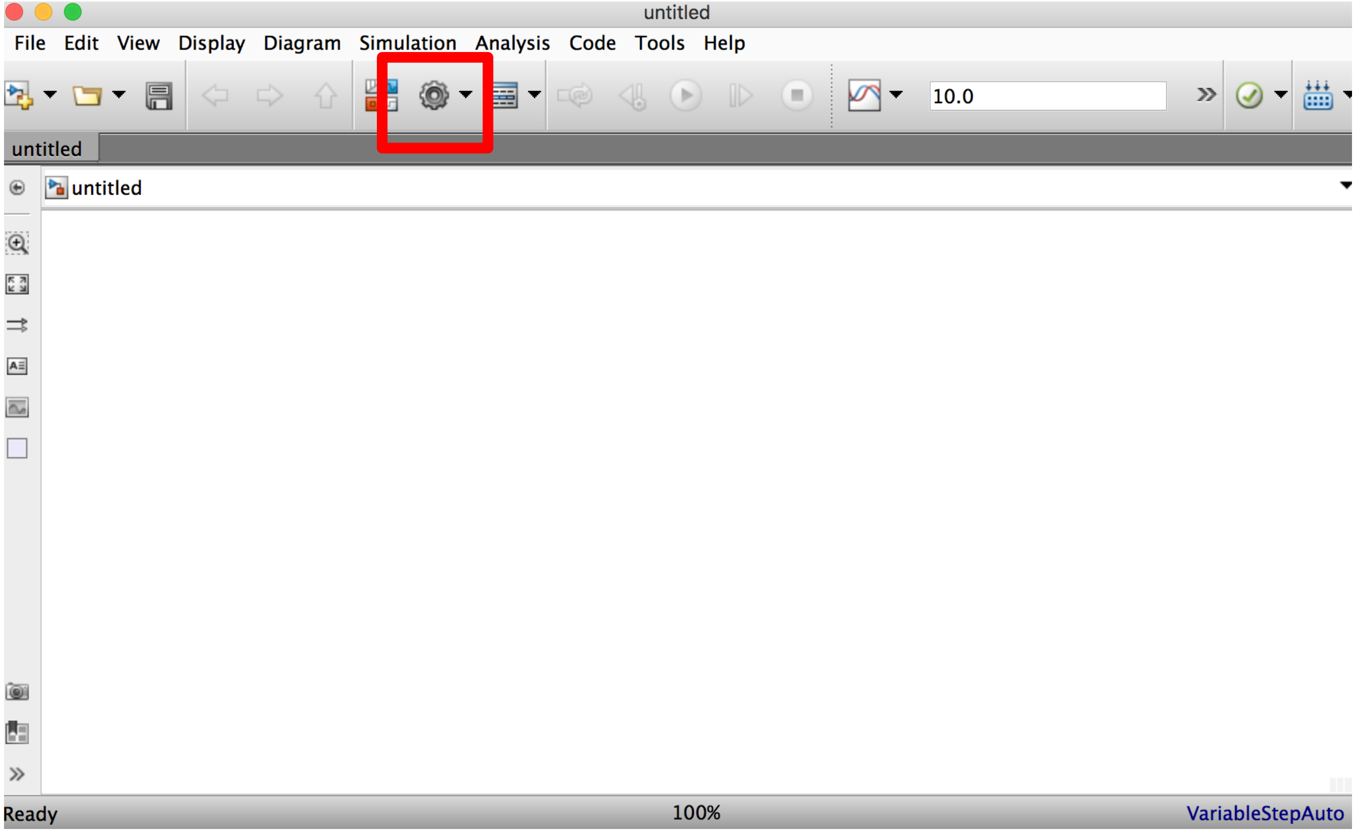


- Signals generation

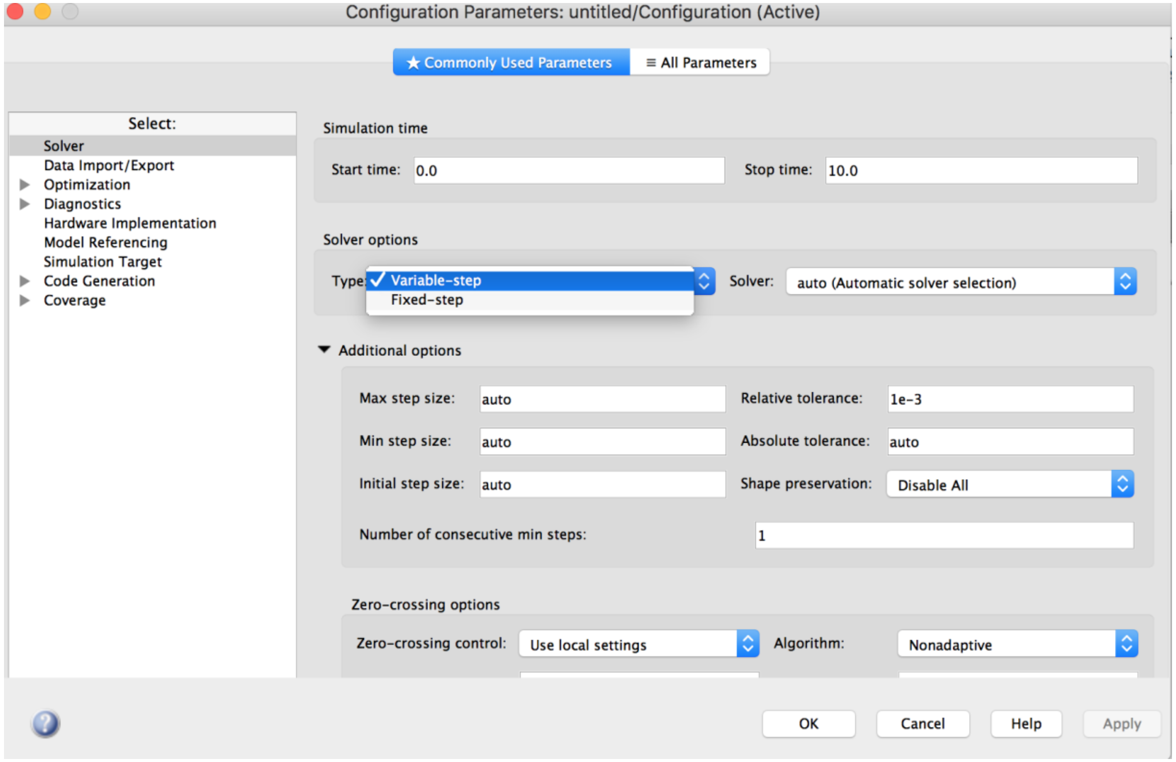


- Graphic creation of the model
  - Drag the desired block from the library to the work window, or right-click with the mouse and click *`Add block to model [model title]`*
  - Some blocks have inputs only or outputs only, others have both inputs and outputs
    - Inputs and outputs can be connected by "pulling" a line between an output and an input while holding down the left mouse button
    - New versions of Simulink allow fast connections between blocks with hotkeys, dependent on the operating system, and quick alignment functions

- The setting of simulation configurations
  - Press the icon with the gear symbol



- Choice of the `solver`
  - *`Variable-step`* continuous time systems
  - *`Fixed-step`* discrete time systems (also set the *`fundamental sample time`*)



- Blocks settings
  - For each block within the work window, a double click with the left mouse button opens the window for setting the options of the relevant block
  - Each block has a variety of settings
  - See the documentation for hints and examples

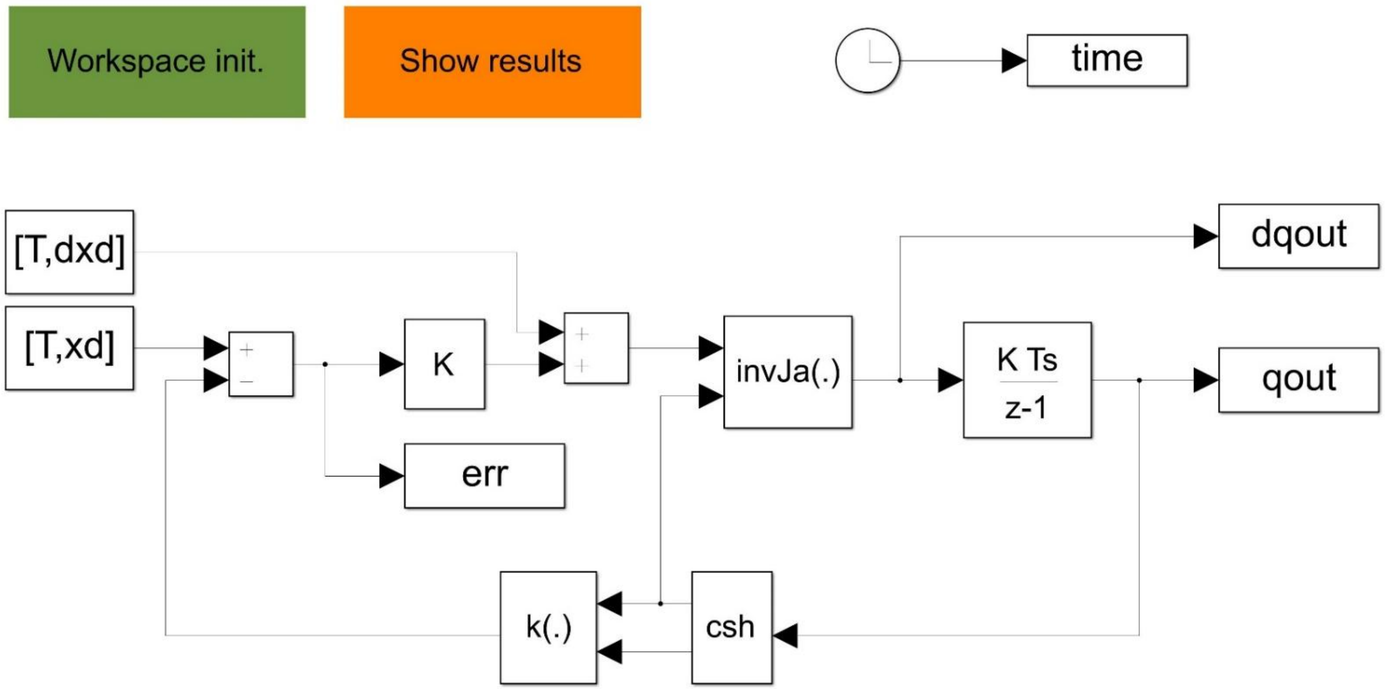
- Call a Simulink model from Matlab
  - Use `sim` command within a Matlab script (`.m` *extension*)
  - See the documentation, and related examples, to get output data
  - Ex.
    - `simout = sim ('name_of_simulink_file');`

Getting Started with Simulink

- A lot of tutorial videos are available on the Mathworks website
  - https://www.mathworks.com/videos/getting-started-with-simulink-part-1-building-and-simulating-a-simple-simulink-model-1508442030520.html?s_tid=vid_pers_recs
  - https://www.mathworks.com/videos/getting-started-with-simulink-part-2-adding-a-controller-and-plant-to-the-simulink-model-1508442594866.html

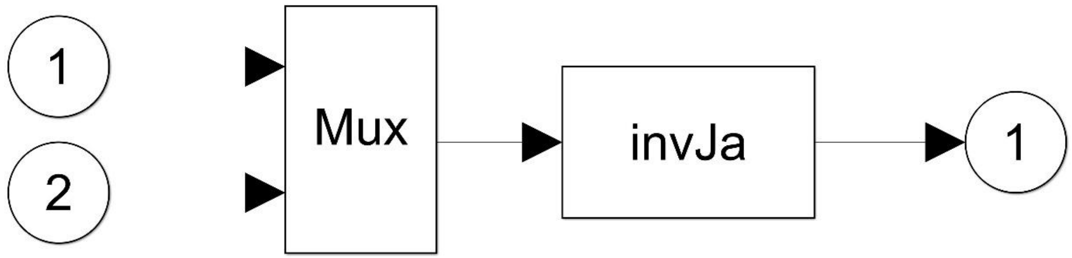
Example / 1

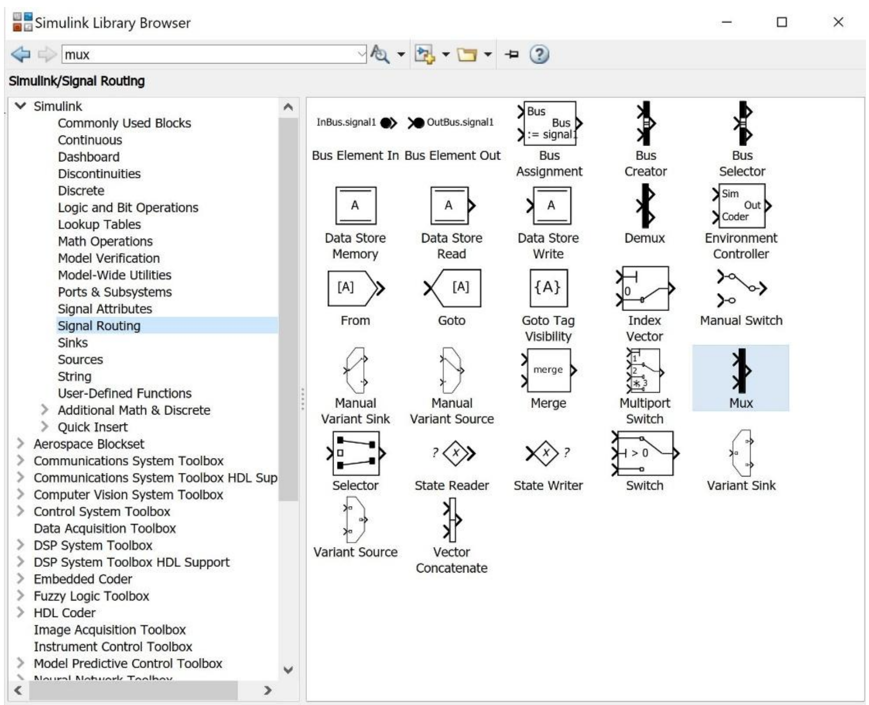
- Inverse kinematic through Jacobian inverse
- The integrator can be ragarded as a simplified robot model (kinematic control)

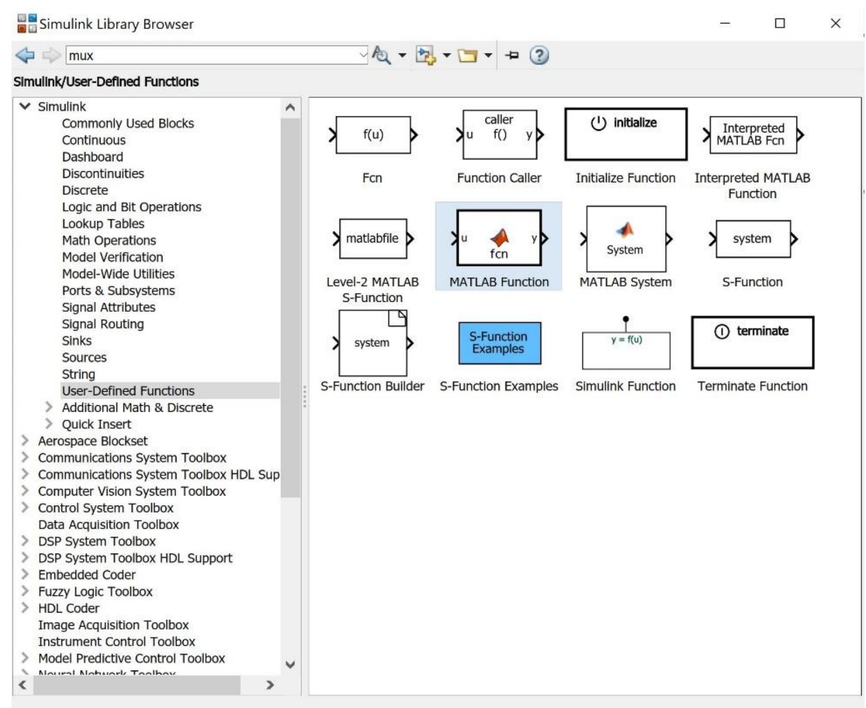


Example / 2

- InvJa(.)







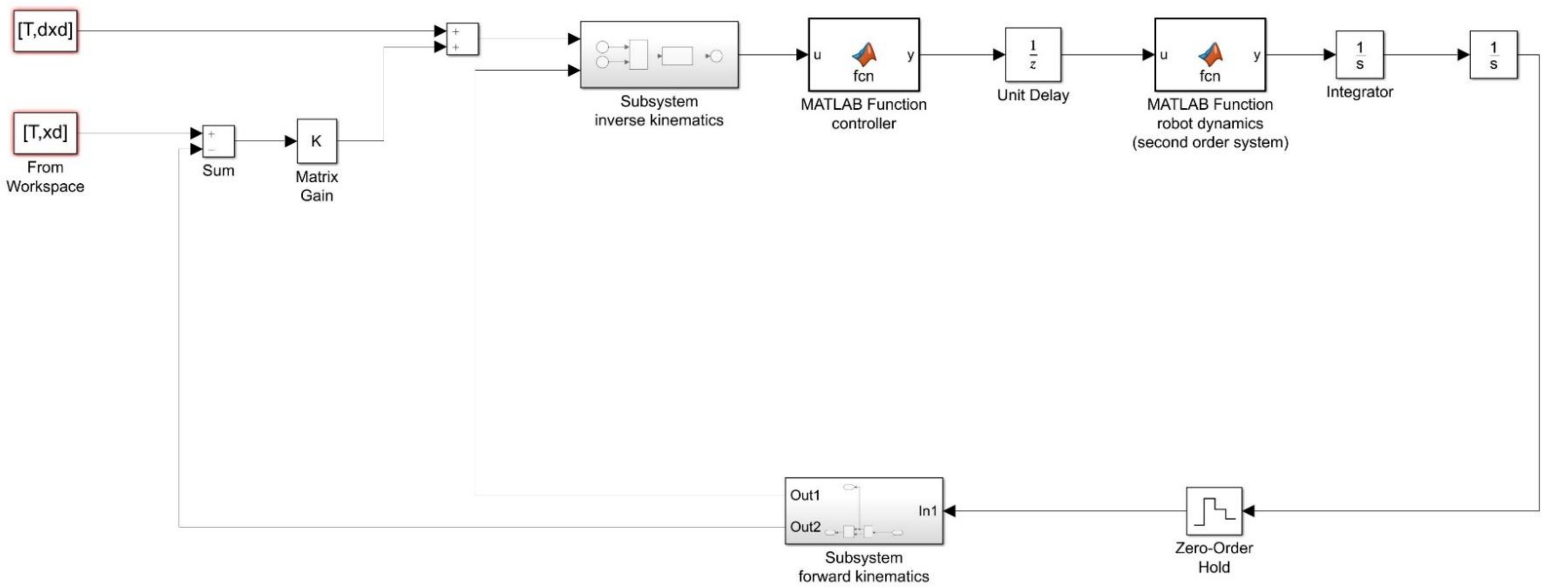
- Matlab functions to implement InvJa

```
function dq = invJa(w)
% Compute inv(Ja(q))*u
dq = J_a(w(4:9))\w(1:3); %Note: the operator 'a\b' is equivalent to 'inv(a)*b'
```

```
function Ja = J_a(u)
% Compute the analytic Jacobian for the three-links planar arm
Ja = [zeros(2,3);ones(1,3)];
Ja(1:2,3) = [-u(6); u(5)];
Ja(1:2,2) = [-u(4); u(3)] + Ja(1:2,3);
Ja(1:2,1) = [-u(2); u(1)] + Ja(1:2,2);
```

Example / 4

- Kinematic control is satisfactory only when not requiring too fast motions
- From a general perspective, the manipulator can be regarded as a sampled data system



Robotics System Toolbox™

- Robotics System Toolbox™ provides tools and algorithms for designing, simulating, and testing manipulators, mobile robots, and humanoid robots
  - https://www.mathworks.com/products/robotics.html
- For manipulators, the toolbox includes algorithms for collision checking, trajectory generation, forward and inverse kinematics, and dynamics using a rigid body tree representation
  - https://www.mathworks.com/help/robotics/ug/rigid-body-tree-robot-model.html
- In the command window type
  - doc-> Robotics System Toolbox