Kom
verder

SAXION

Solving 2nd-order DEs in Matlab and in Simulink

Ronald Tangelder, Jan de Wilde

Kom verder

Contents

- Solving 2nd-order differential equations numerically with Matlab
- A closer look at some Simulink library elements
  - A Sine Wave
  - A Constant
  - A Mathematical Function
- Solving 2nd-order differential equations with Simulink
- Creating subcircuits in Simulink
- Using a vector in a gain block

Solving Second-order DEs numerically with Matlab

- Matlab is not able to solve second-order DEs directly. Matlab can only solve first order DEs of the form:

$$\frac{dy}{dt} = f(t, y)$$

- However, Matlab can work with more DEs at the same time without any problem, since Matlab uses matrices and arrays.

- **We can rewrite second-order DEs to a system of two first order DEs.** (Think of linear algebra: a system of linear equations).

Kom verder

SAXION

Rewriting of a 2^{nd}-order DE

Given are the DE and the boundary conditions:

$$a\frac{d^2y}{dt^2} + b\frac{dy}{dt} + cy = f(t) \qquad y(0) = y_0 \quad y'(0) = y'_0$$

which is equal to:
$$\frac{d^2y}{dt^2} = -\frac{b}{a}\frac{dy}{dt} - \frac{c}{a}y + \frac{f(t)}{a} \qquad y(0) = y_0 \quad y'(0) = y'_0$$

Introduce new variables:
$$\boxed{z_1(t) = y(t) \qquad z_2(t) = \frac{dy}{dt}}$$

Take the derivatives of these new variables:

$$\frac{dz_1}{dt} = \frac{dy}{dt} \qquad \Rightarrow \qquad \frac{dz_1}{dt} = z_2$$
$$\frac{dz_2}{dt} = \frac{d^2y}{dt^2} \qquad \Rightarrow \qquad \frac{dz_2}{dt} = -\frac{c}{a}z_1 - \frac{b}{a}z_2 + \frac{f(t)}{a}$$

Matrix form of a 2nd–order DE

- These equations can be written into matrix form

$$\frac{d^2y}{dt^2} = -\frac{b}{a}\frac{dy}{dt} - \frac{c}{a}y + \frac{f(t)}{a} \qquad y(0) = y_0 \quad y'(0) = y'_0$$

$$\frac{dz_1}{dt} = \frac{dy}{dt} \quad \Rightarrow \quad \frac{dz_1}{dt} = z_2$$

$$\frac{dz_2}{dt} = \frac{d^2y}{dt^2} \quad \Rightarrow \quad \frac{dz_2}{dt} = -\frac{c}{a}z_1 - \frac{b}{a}z_2 + \frac{f(t)}{a}$$

$$\begin{bmatrix} \frac{dz_1}{dt} \\ \frac{dz_2}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{c}{a} & -\frac{b}{a} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{f(t)}{a} \end{bmatrix}$$

Example of a 2nd–order DE

$$a\frac{d^2y}{dt^2} + b\frac{dy}{dt} + cy = f(t) \qquad y(0) = y_0 \quad y'(0) = y'_0$$

$$\boxed{\begin{bmatrix} \frac{dz_1}{dt} \\ \\ \frac{dz_2}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \\ -\frac{c}{a} & -\frac{b}{a} \end{bmatrix} \begin{bmatrix} z_1 \\ \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \\ \frac{f(t)}{a} \end{bmatrix}}$$

$$\frac{d^2y}{dt^2} + 2\frac{dy}{dt} + 3y = cos(4t) \qquad y(0) = y_0 \quad y'(0) = y'_0$$

$$\boxed{\begin{bmatrix} \frac{dz_1}{dt} \\ \\ \frac{dz_2}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \\ -3 & -2 \end{bmatrix} \begin{bmatrix} z_1 \\ \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \\ cos(4t) \end{bmatrix}}$$

Implementation in Matlab

- The built-in function `ode45` can handle matrices. We only have to evaluate the right side of the DEs. This is done in a Matlab function.

$$\frac{d^2y}{dt^2} + 2\frac{dy}{dt} + 3y = cos(4t)$$

$$y(0) = y_0 \quad y'(0) = y'_0$$

$$\boxed{\begin{bmatrix} \frac{dz_1}{dt} \\ \frac{dz_2}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{c}{a} & -\frac{b}{a} \end{bmatrix} \begin{bmatrix} z_1 \\ z_2 \end{bmatrix} + \begin{bmatrix} 0 \\ \frac{f(t)}{a} \end{bmatrix}}$$

```
function dzdt = righthandside(t,z)
% input    t: time
%          z: boundary condition in a column vector
% output dzdt: derivatives in a column vector

% the derivates have to be placed in a column vector
dzdt = zeros(2,1)

% the coefficients of the second-order DE
a = 1
b = 2;
c = 3;
f = cos(4*t);

% choose either alternative 1 or alternative 2!
% both will give precisely the same result

% alternative 1: component format
dzdt(1) = z(2);
dzdt(2) = -(c/a)*z(1) - (b/a)*z(2) + f/a

% alternative 2: matrix format
A = [0 , 1 ; -c/a, -b/a];
dzdt = A*z + [0 ; f/a];

end % function
```

Example of a 2^{nd}–Order DE from the mechanical domain

- Example: driven mass-spring system with friction.

$$\frac{d^2y}{dt^2} = \cos(4t) - \frac{b}{m}\frac{dy}{dt} - \frac{k}{m}y \qquad y(0) = y_0[m] \qquad y'(0) = v_0[m/s]$$

- Approach: split the second-order DEs in two first order DEs.
  - Substitute $\dfrac{dy}{dt} = v$
  - The result is two first-order DEs

$$\frac{dy}{dt} = v$$

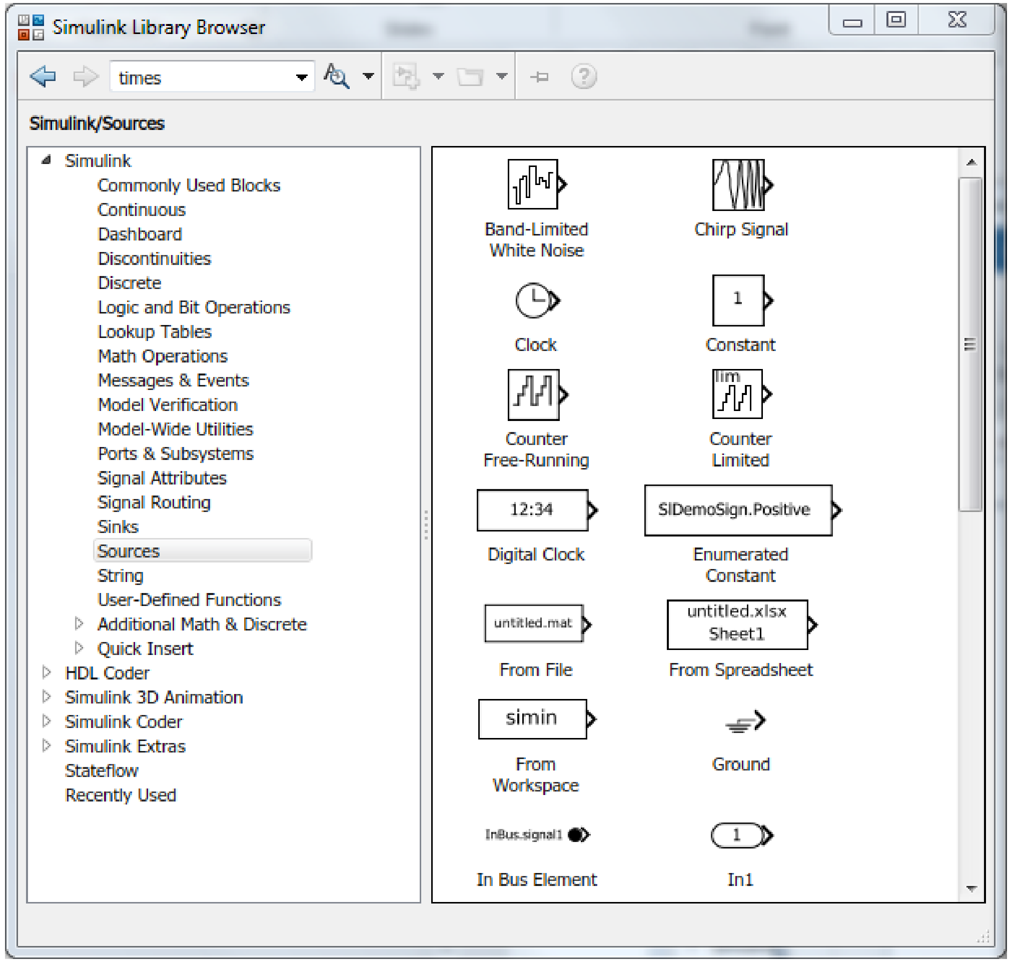
$$\frac{dv}{dt} = \cos(4t) - \frac{b}{m}v - \frac{k}{m}y$$

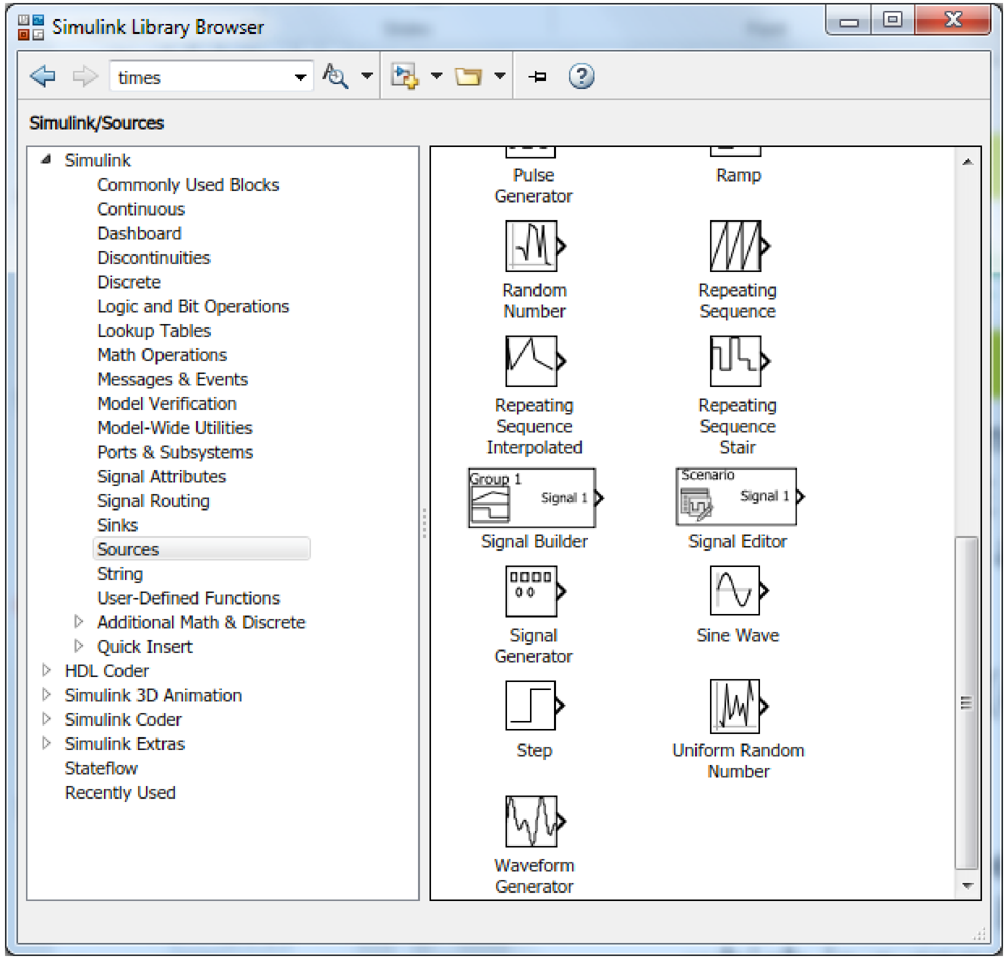
$$\begin{bmatrix} \frac{dy}{dt} \\ \frac{dv}{dt} \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ -\frac{k}{m} & -\frac{b}{m} \end{bmatrix} \begin{bmatrix} y \\ v \end{bmatrix} + \begin{bmatrix} 0 \\ \cos(4t) \end{bmatrix}$$

Contents

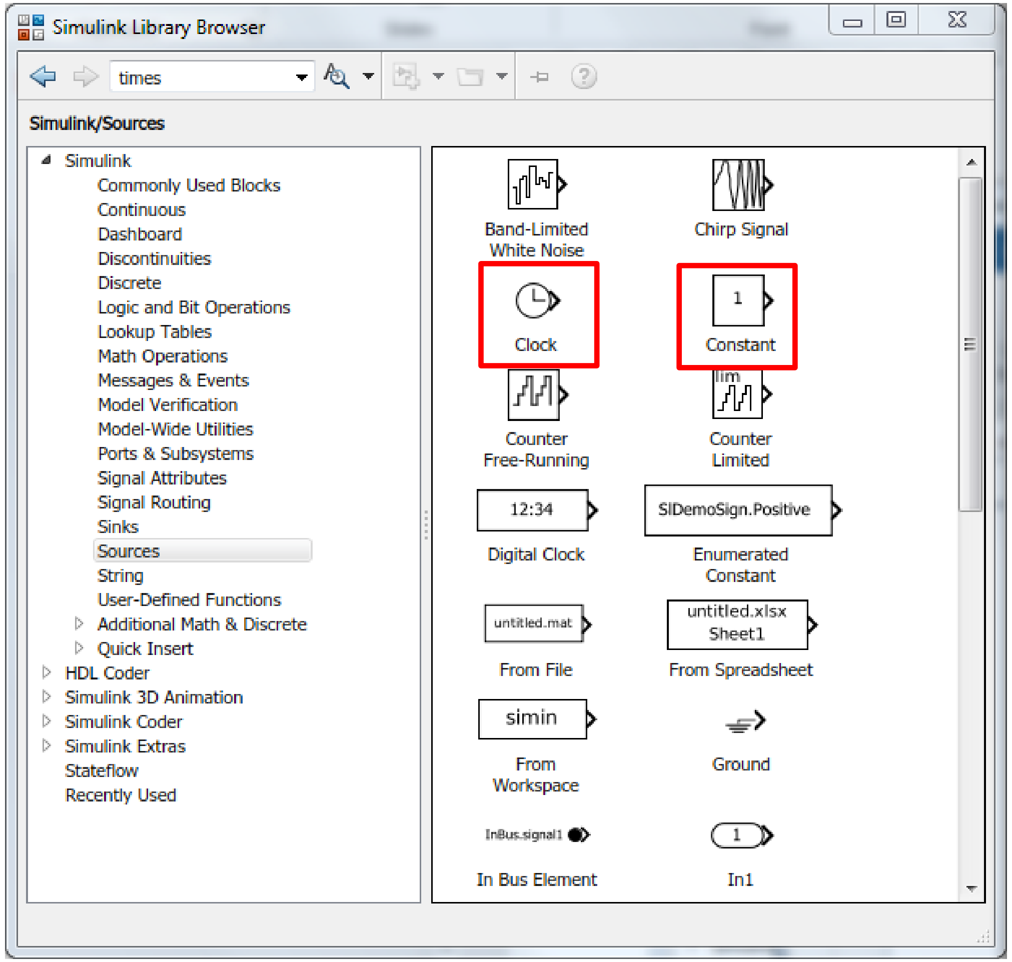
- Solving 2nd-order differential equations numerically with Matlab
- A closer look at some Simulink library elements
  - A Sine Wave
  - A Constant
  - A Mathematical Function
- Solving 2nd-order differential equations with Simulink
- Creating subcircuits in Simulink
- Using a vector in a gain block

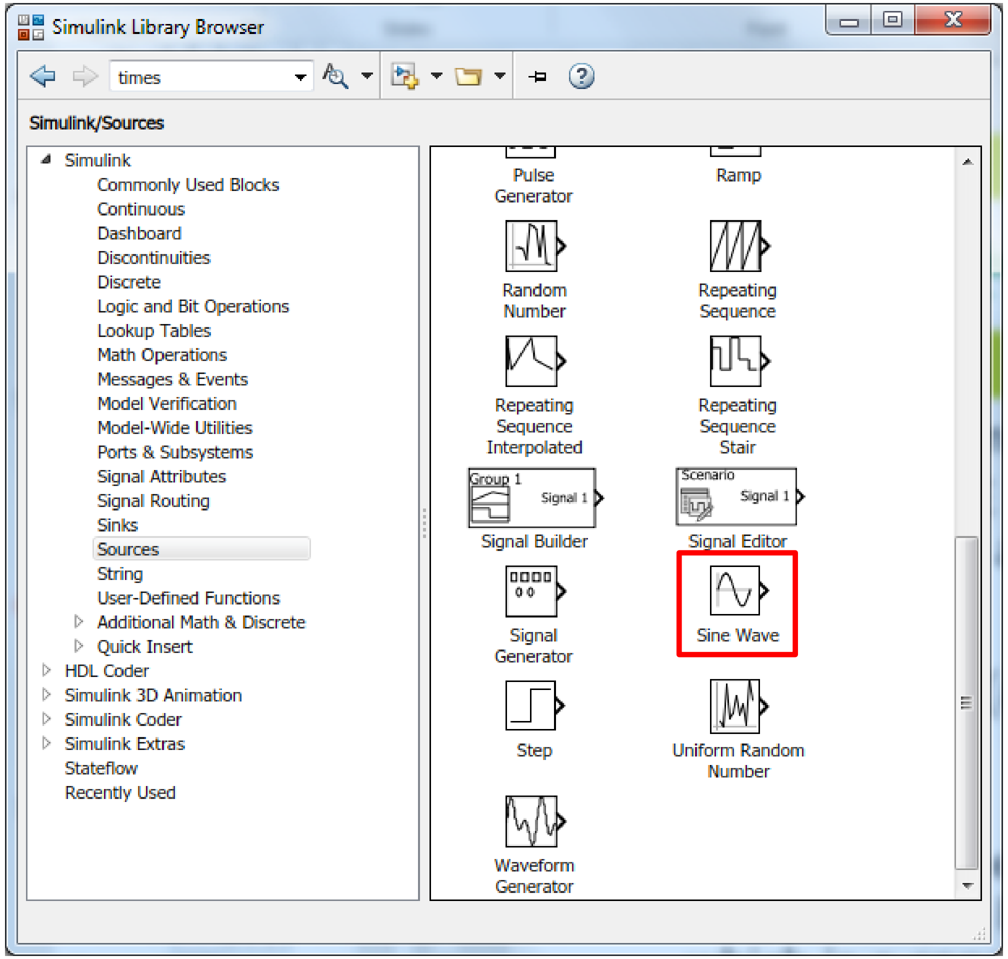
Simulink/Sources





Simulink/Sources





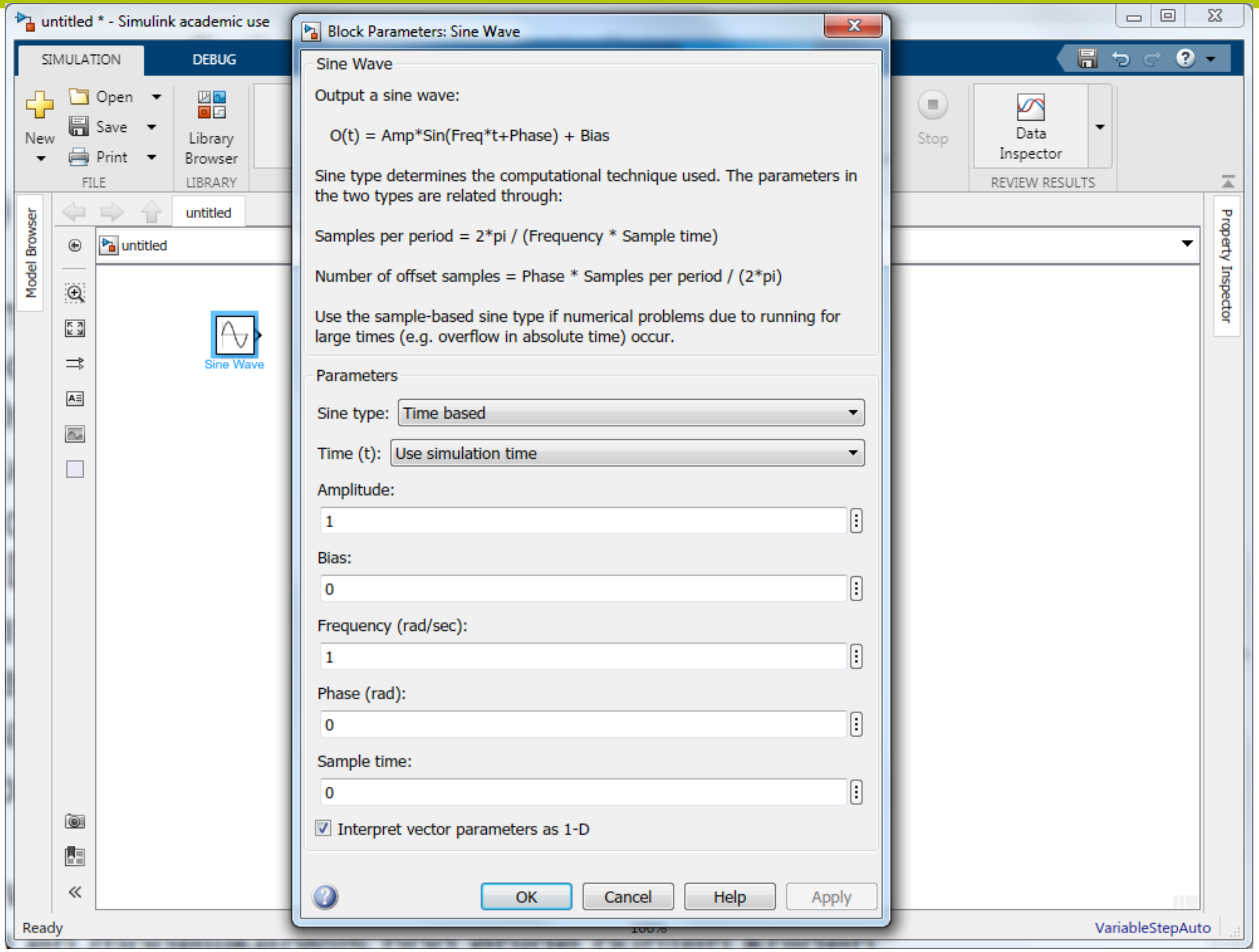
Kom verder

SAXION

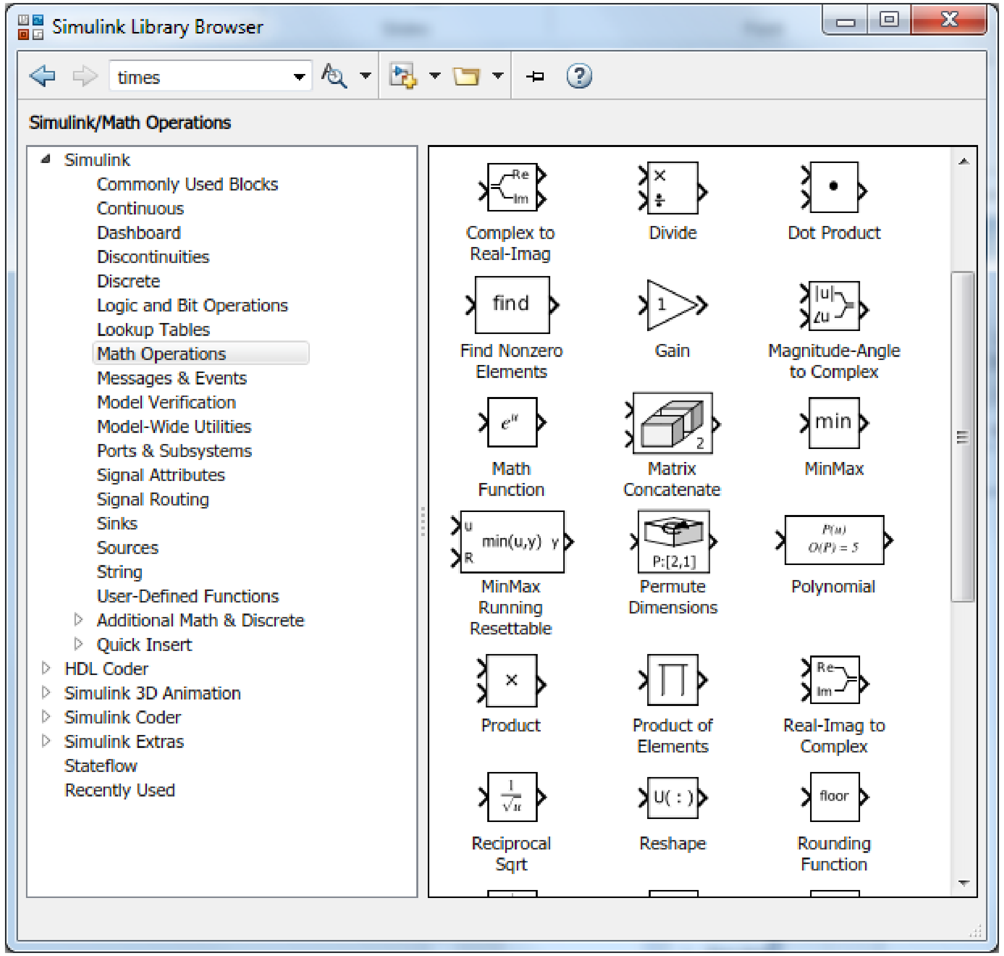
Using a Sine Wave

Sine wave can be fine-tuned by setting the *Amplitude*, *Frequency*, *Phase* and *Bias* (i.e. *Offset*).

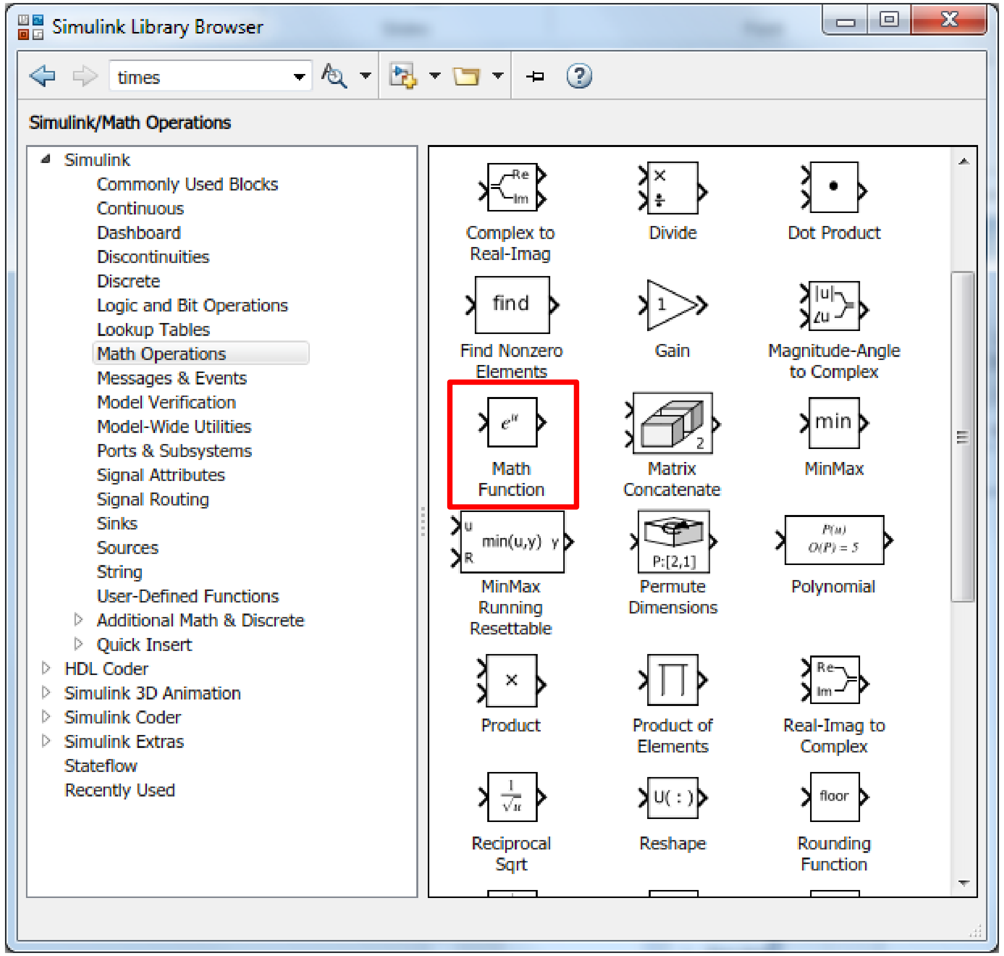
Also, you can choose time as external input, or let it being equal to the simulation time



Simulink/Math Operations



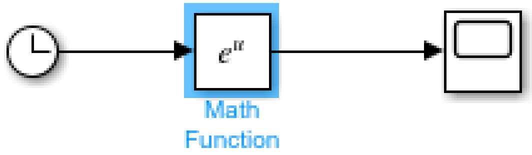
Simulink/Math Operations

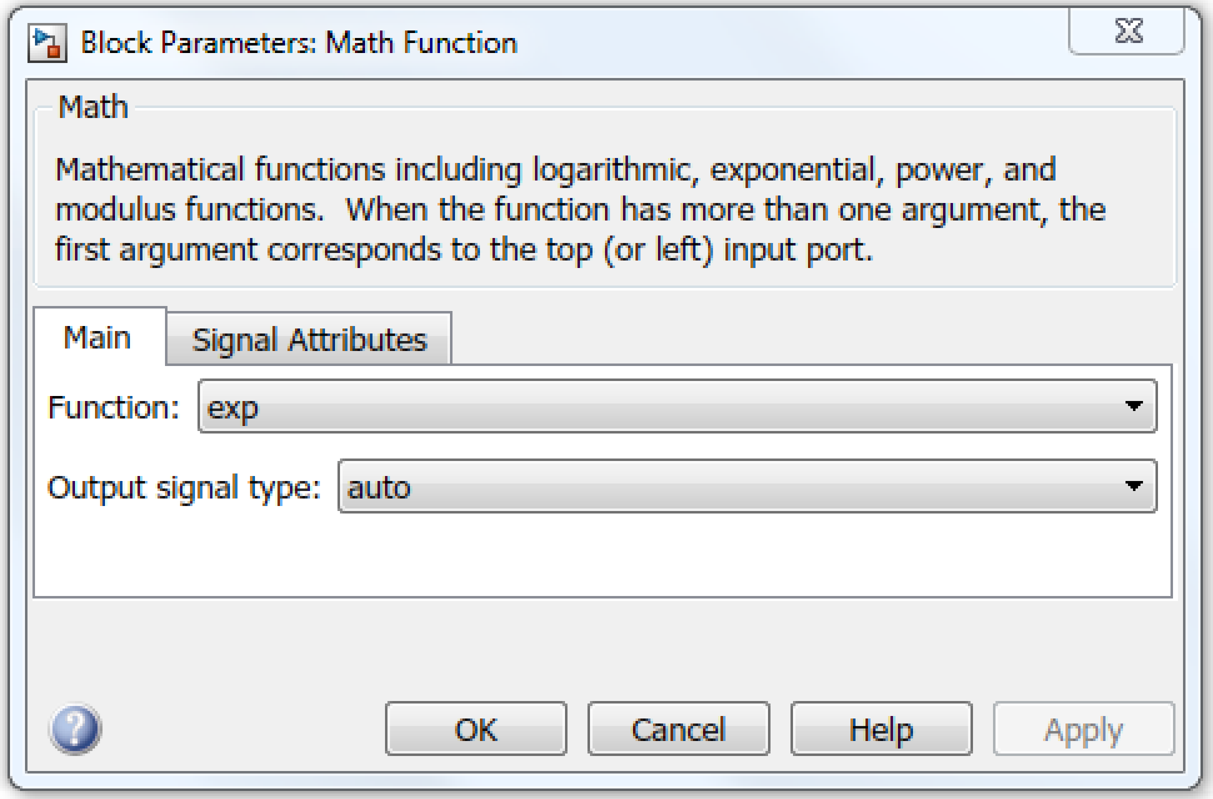


Using a mathematical function (1)

Basic example

- Use the `Clock` from `sources`.
- Use the `Math Function` from `Math Operations` (in this case the default exponential function).
- Use the `Scope` from `Commonly Used Blocks` or `Sinks`.
- Run the simulation for 2 time-units.

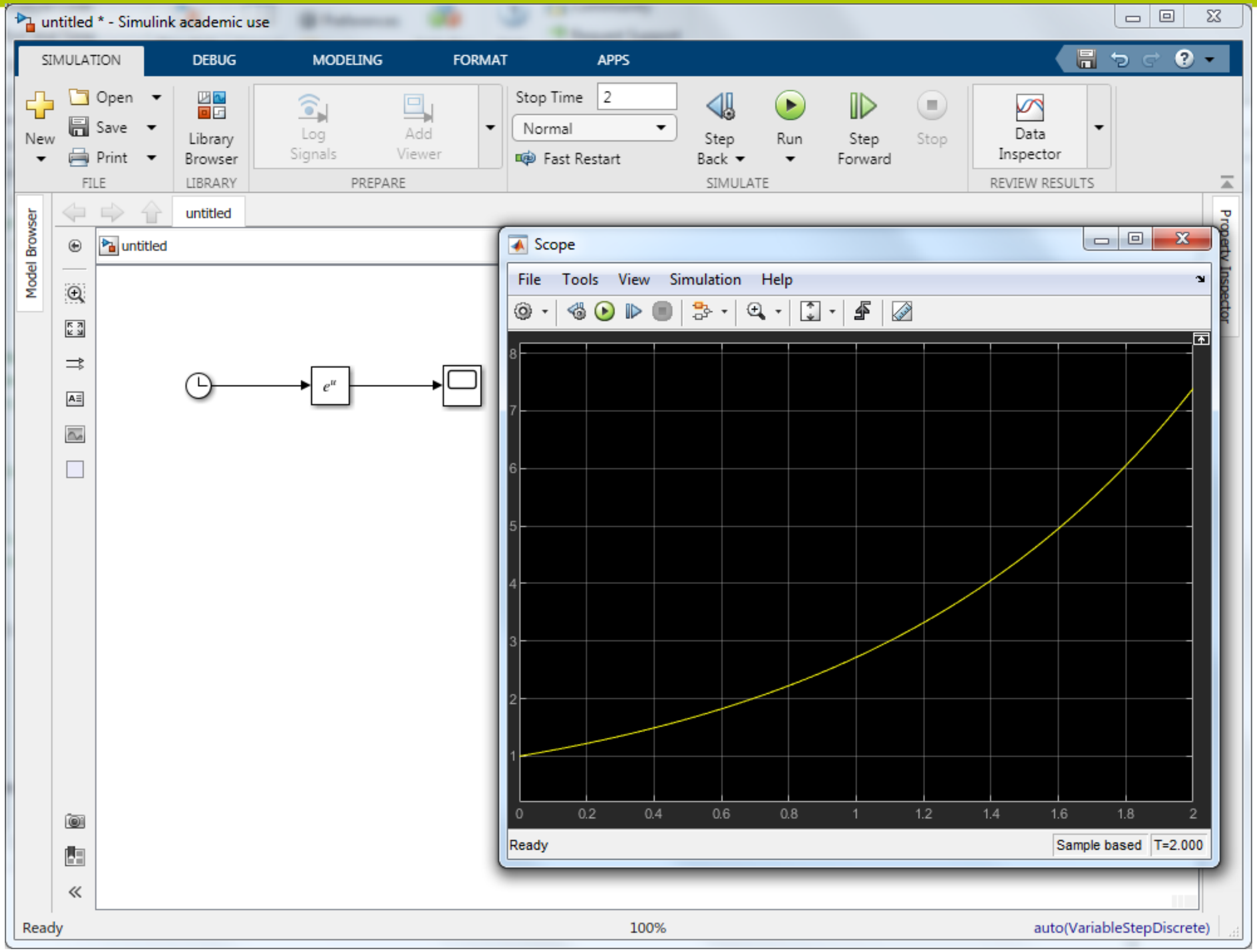




Using a mathematical function (2)

Basic example

- Use the `Clock` from `sources`.
- Use the `Math Function` from `Math Operations` (in this case the default exponential function).
- Use the `Scope` from `Commonly Used Blocks` or `Sinks`.
- Run the simulation for 2 time-units.



Contents

- Solving 2^{nd}-order differential equations numerically with Matlab
- A closer look at some Simulink library elements
  - A Sine Wave
  - A Constant
  - A Mathematical Function
- Solving 2^{nd}-order differential equations with Simulink
- Creating subcircuits in Simulink
- Using a vector in a gain block

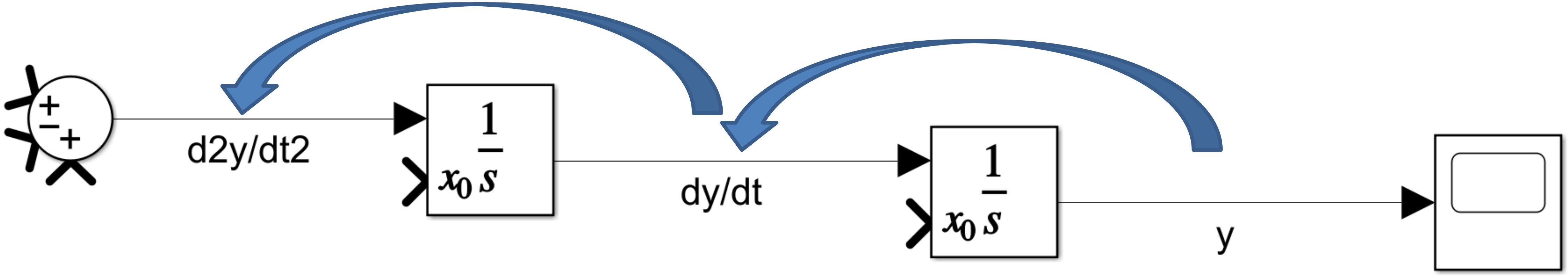
$$a\frac{d^2y(t)}{dt^2}+b\frac{dy(t)}{dt}+cy(t)=f(t),$$
$$y(0)=P\ \&\ y'(0)=Q$$

Given are a 2nd-order DE and its boundary conditions:

$$2\frac{d^2y}{dt^2}-6\frac{dy}{dt}+8y=2e^t \qquad y(0)=2 \quad y'(0)=-5$$

In standard form: $\frac{d^2y}{dt^2}=3\frac{dy}{dt}-4y+e^t \qquad y(0)=2 \quad y'(0)=-5$

A sketch of a possible Simulink implementation could look like the block scheme below (you still need to add the *gains*, the *initial conditions* and the *exponential function* (see previous slide) to complete it)



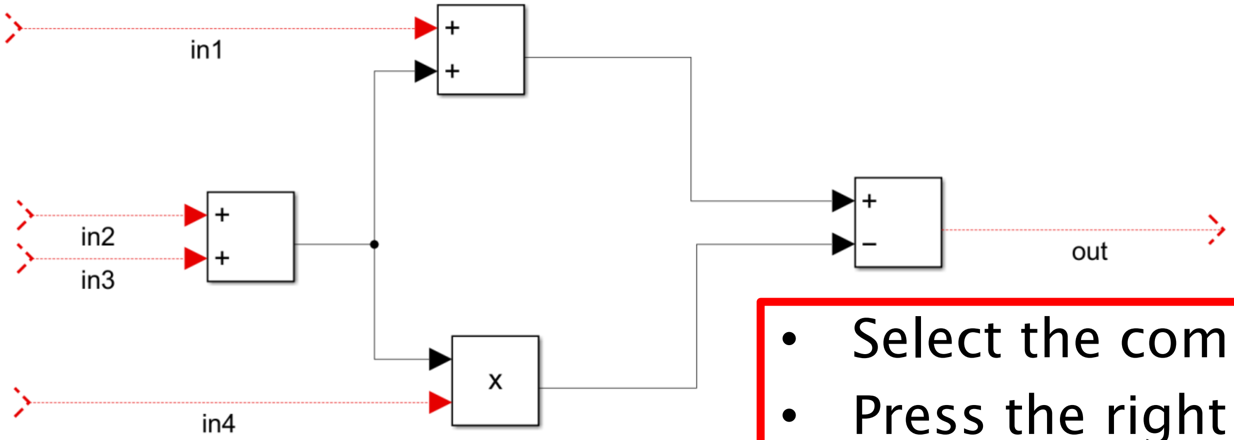
Kom verder

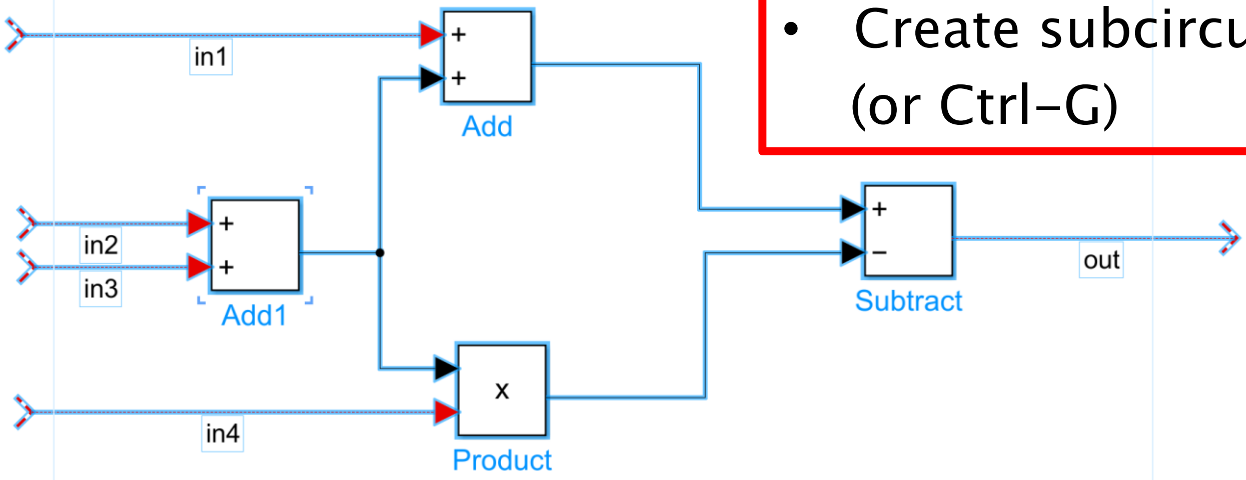
SAXION

Contents

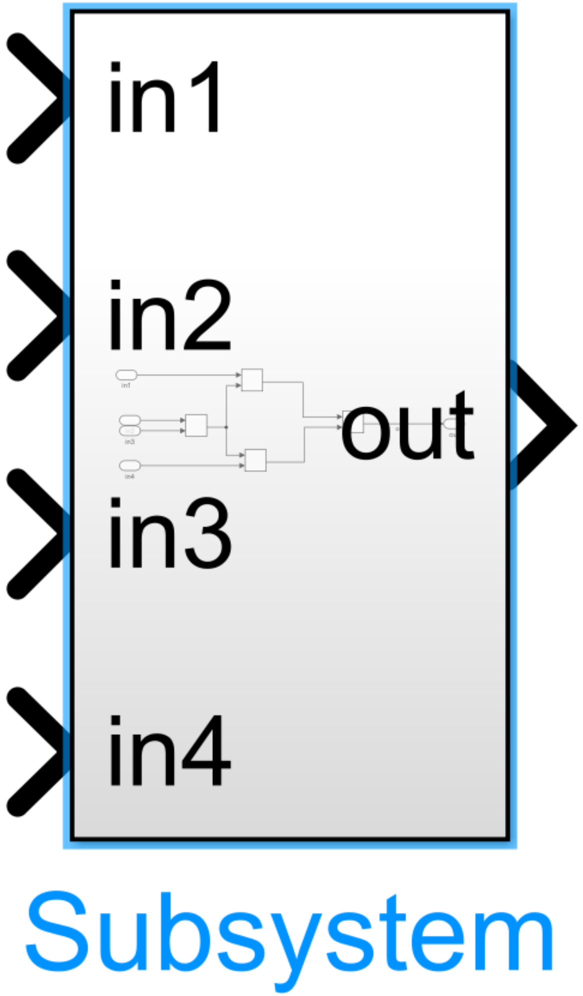
- Solving 2nd-order differential equations numerically with Matlab
- A closer look at some Simulink library elements
  - A Sine Wave
  - A Constant
  - A Mathematical Function
- Solving 2nd-order differential equations with Simulink
- Creating subcircuits in Simulink
- Using a vector in a gain block

Creating a subsystem





- Select the components
- Press the right mouse button
- Create subcircuit from selection (or Ctrl-G)



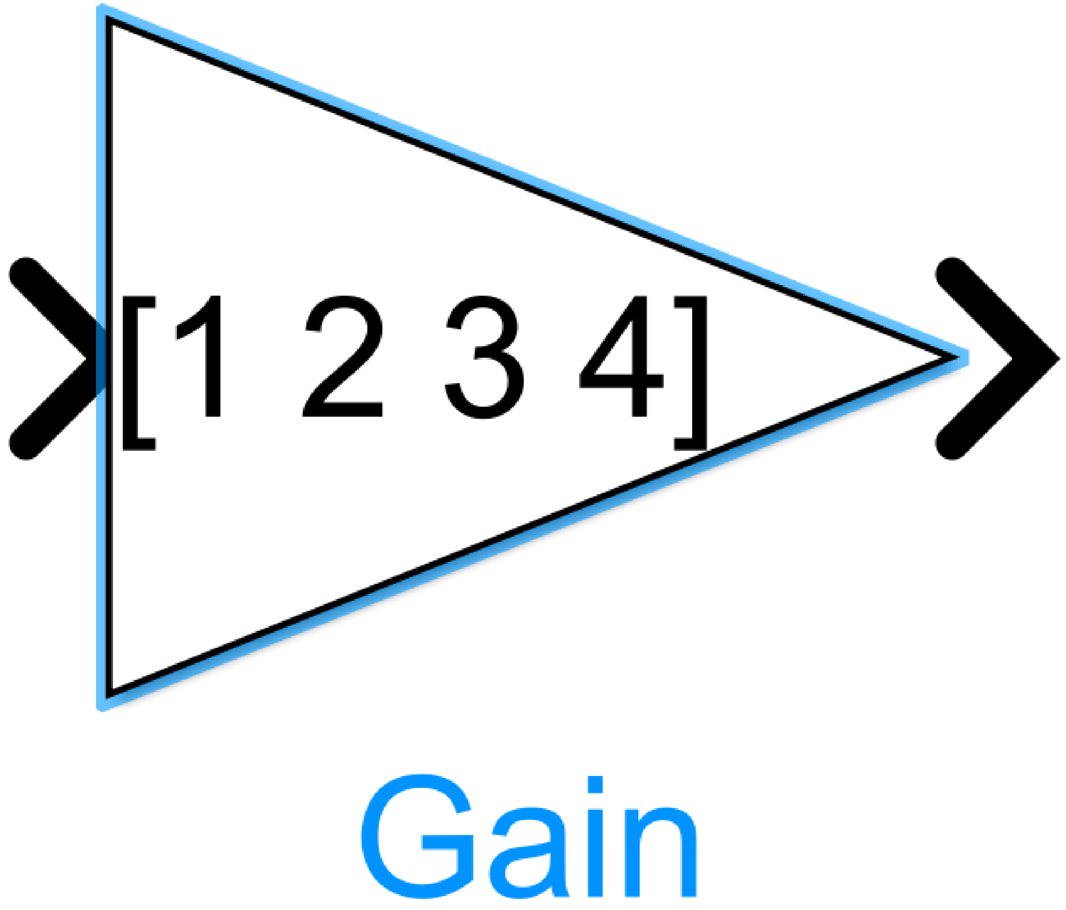
Kom verder

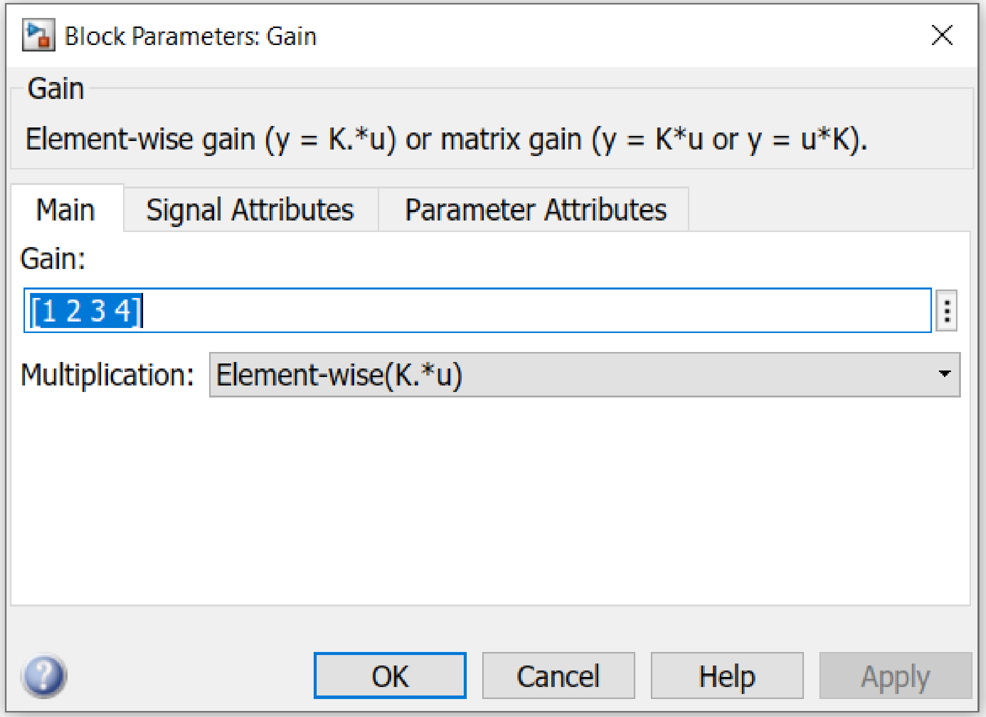
SAXION

Contents

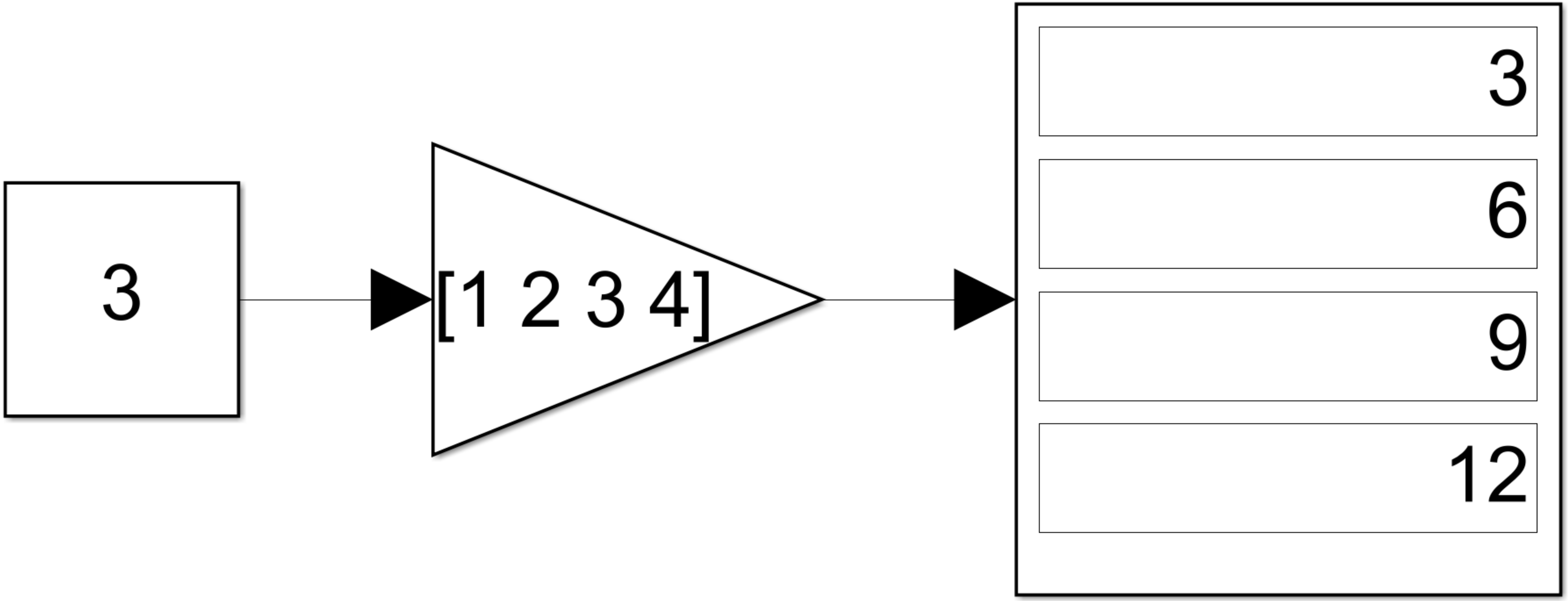
- Solving 2nd-order differential equations numerically with Matlab
- A closer look at some Simulink library elements
  - A Sine Wave
  - A Constant
  - A Mathematical Function
- Solving 2nd-order differential equations with Simulink
- Creating subcircuits in Simulink
- Using a vector in a gain block

Using a vector in a gain block (1)





Using a vector in a gain block (2)



Questions