

# MATLAB Introduction Course: Lecture 5

Øivind K. Kjerstad

31. October 2014

# TOC

1 Symbolic math

2 Simulink

# Symbolic Math Toolbox

- MATLAB can do symbolic math!
- Don't do nasty calculations by hand!

	Advantages	Disadvantages
Symbolic	<ul style="list-style-type: none"><li>→ Analytical solutions</li><li>→ Intuitive</li></ul>	<ul style="list-style-type: none"><li>→ May lack solution</li><li>→ Can be overly complicated</li></ul>
Numeric	<ul style="list-style-type: none"><li>→ Always a solution</li><li>→ Accurate</li><li>→ Easy to code</li></ul>	<ul style="list-style-type: none"><li>→ Harder to extract understanding</li><li>→ Num. methods can fail</li><li>→ Computational time</li></ul>

# Symbolic Variables

- Symbolic variables are a type, like double or char
- To make symbolic variables, use `sym`
  - ▶ `>> a=sym('1/3')`
  - ▶ `>> b=sym('4/5')`
  - ▶ `>> mat=sym([1 2;3 4])`
    - ★ fractions remain as fractions
  - ▶ `>> c=sym('c','positive')`
    - ★ can add tags to narrow down scope
    - ★ see `help sym` for a list of tags
- Or use `syms`
  - ▶ `>> syms x y real`
    - ★ shorthand for `x=sym('x','real');` `y=sym('y','real');`

# Symbolic Expressions

- Multiply, add, divide as you would normally
  - ▶ `>> d=a*b`
    - ★ gives `>> d = 4/15`
- Expand expressions
  - ▶ `>> expand((a-c)^2)`
    - ★ gives `>> ans = 1/9-2/3*c+c^2`
- Factor expressions
  - ▶ `>> factor(1/9-2/3*c+c^2)`
    - ★ gives `>> ans = 1/9*(3*c-1)^2`

# Cleaning up Symbolic Statements

- `pretty(...)`
  - ▶ makes it look nicer
- `collect(...)`
  - ▶ collects terms
- `simplify(...)`
  - ▶ simplifies expressions
- `subs('c^2',c,5)`
  - ▶ Replaces variables with numbers or expressions. To do multiple substitutions pass a cell of variable names followed by a cell of values
- `subs('c^2',c,x/7)`
  - ▶ gives `>> ans = 1/49*x^2`

# More Symbolic Operations

- Matrix symbolics

- ▶ `>> mat = sym(' [a b;c d]')`
- ▶ `>> mat2 = mat*[1 2;4 -2]`
  - ★ computes the matrix product
- ▶ `>> D = det(mat)`
  - ★ computes the determinant
- ▶ `>> mat_i = inv(mat)`
  - ★ computes the inverse

- You can access symbolic matrix elements normally

- ▶ `>> mat_i(1,2)=-b/(a*d-b*c)`

# Exercises

## Exercise a

The equation of a circle of radius  $r$  centered at  $(a,b)$  is given by:

$$(x - a)^2 + (y - b)^2 = r^2$$

- Use `solve` to solve this equation for  $x$  and then for  $y$

## Exercise b

Use `int` to do the following integral symbolically:

$$\int_a^b x e^x dx$$

and then compute the value by `substituting` 0 for  $a$  and 2 for  $b$



# Solution

## Exercise a

```
>> syms a b r x y
>> solve('(x-a)^2+(y-b)^2=r^2','x')
>> solve('(x-a)^2+(y-b)^2=r^2','y')
```

## Exercise b

```
>> syms a b x
>> Q=int(x*exp(x),a,b)
>> subs(Q,{a,b},{0,2})
```

# TOC

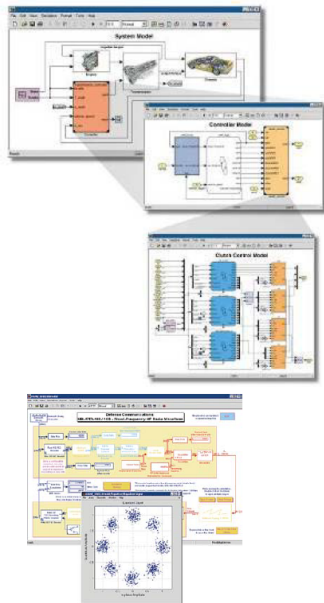
1 Symbolic math

2 Simulink

# What is Simulink

Simulink is a software package for modeling, simulating, and analyzing dynamical systems

- Graphical programming
- Nonlinear simulation
- Hybrid (continuous and discrete) models
- Asynchronous (non-uniform sampling) simulation
- Fully integrated with MATLAB, MATLAB toolboxes and blocksets



# What to use it for

Design, implement, and test:

- Dynamical systems
- Control systems
- Communications systems
- Signal Processing systems
- Embedded systems
- +++

Anything that is time-varying!

## NTNU usage

Most of the control development at NTNU is done using MATLAB/Simulink

# Bell Helicopter Develops the First Civilian Tiltrotor, Using Model-Based Design



The BA609, flying in airplane mode.

## Challenge

To design and build the BA609, the first and fastest commercially available tiltrotor aircraft in the world

## Solution

Use Model-Based Design with MATLAB, Simulink, and Real-Time Workshop software to model, simulate, test, and verify designs

## Results

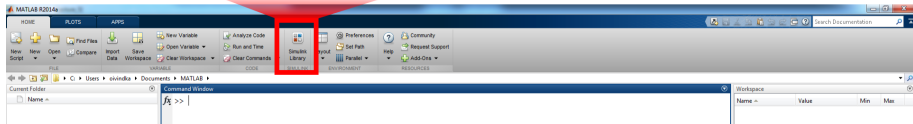
- Full collaboration with suppliers via Simulink models
- Flight control system code generated automatically from models
- 40% improvement in design and development time
- Flawless first flight, which went exactly like the simulation

**"Simulations and a rapid, iterative approach enabled us to minimize the unknowns and ensure that we had established enough margin that when we ran into a surprise we could continue to have a safe flight test program—and run it with unprecedented efficiency."**

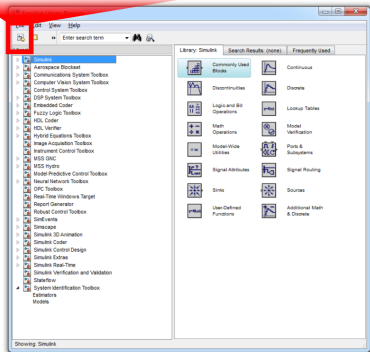
David King  
Bell Helicopter

# Getting started

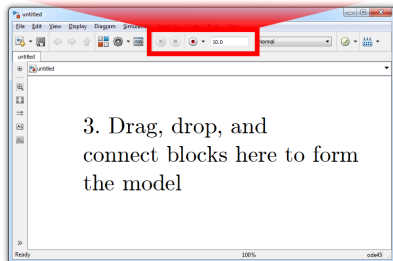
## 1. Start Simulink



## 2. Create a new model

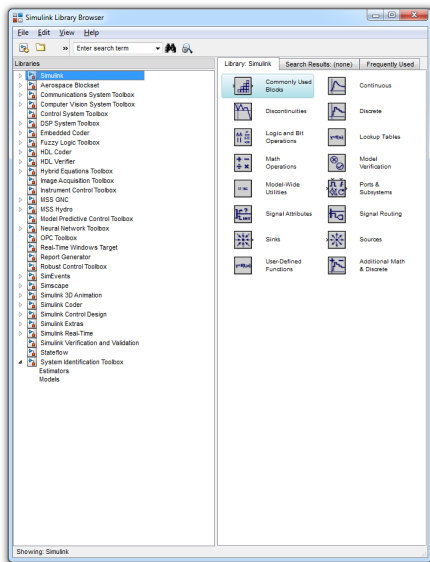


## 4. Run the model in a given interval



3. Drag, drop, and connect blocks here to form the model

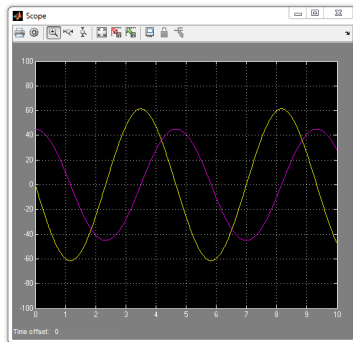
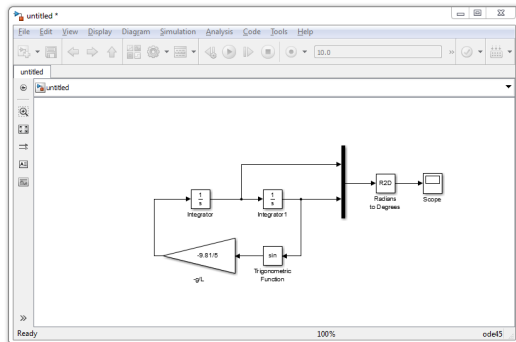
# The Library Browser



- The Library Browser contains various blocks that you can put into your model
- Find blocks by searching or browsing
- Not always intuitive

# A simple example: the pendulum revisited

$$\ddot{\theta} + \frac{g}{L} \sin(\theta) = 0$$





# Behind the curtain: solvers

## Model configuration pane

The image shows the MATLAB Model Configuration pane for an untitled model. The pane is titled "Configuration Parameters: untitled/Configuration (Active)". It contains several sections for configuring the simulation:

- Simulation time:** Start time: 0.0, Stop time: 10.0
- Solver options:**
  - Type: Variable-step (dropdown menu)
  - Max step size: Variable-step (dropdown menu)
  - Min step size: auto
  - Initial step size: auto
  - Number of consecutive min steps: 1
- Tasking and sample time options:**
  - Tasking mode for periodic sample times: Auto (dropdown menu)
  - Automatically handle rate transition for data transfer
  - Higher priority value indicates higher task priority
- Zero-crossing options:**
  - Zero-crossing control: Use local settings (dropdown menu)
  - Algorithm: Nonadaptive (dropdown menu)
  - Time tolerance:  $10 \times 128 \times \text{eps}$
  - Signal threshold: auto
  - Number of consecutive zero crossings: 1000

The Solver options section is expanded, showing a list of solvers:

- ode45 (Dormand-Prince)
- discrete (no continuous states)
- ode15i (Dormand-Prince)
- ode23 (Bogacki-Shampine)
- ode113 (Adams)
- ode15s (stiff/NDF)
- ode23s (stiff/Mod. Rosenbrock)
- ode23t (mod. stiff/Trapezoidal)
- ode23tb (stiff/TR-BDF2)

The MATLAB interface in the background has a red arrow pointing to the Configuration icon in the toolbar.

# Useful tips and blocks

Large Simulink models  
may get really messy!

To avoid this use:

- Subsystems to modularize
- Goto and From blocks
- Color code
- Labels

Most blocks are straight  
forward to use

However..



Send data wirelessly with a given tag



Receive wireless data with a given tag



Trigonometric functions: sin, cos, tan, arcsin, arccos, arctan, hyperbolic



Common math functions



Select a set of signals within a multi-signal wire



send data to MATLAB workspace



Code a function directly in Simulink

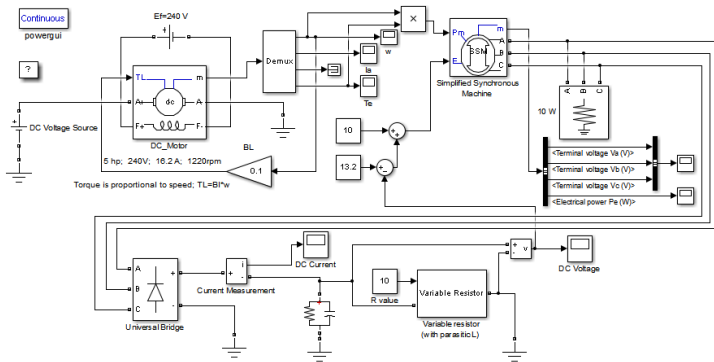


Use a MATLAB function

# Mathworks Simulink demos

Mathworks have some nice demonstration examples of how to use Simulink. Check them out at:

<http://www.mathworks.se/help/simulink/examples.html>



Copyright 2006-2013 The MathWorks, Inc.

# Marine applications: The MSS toolbox

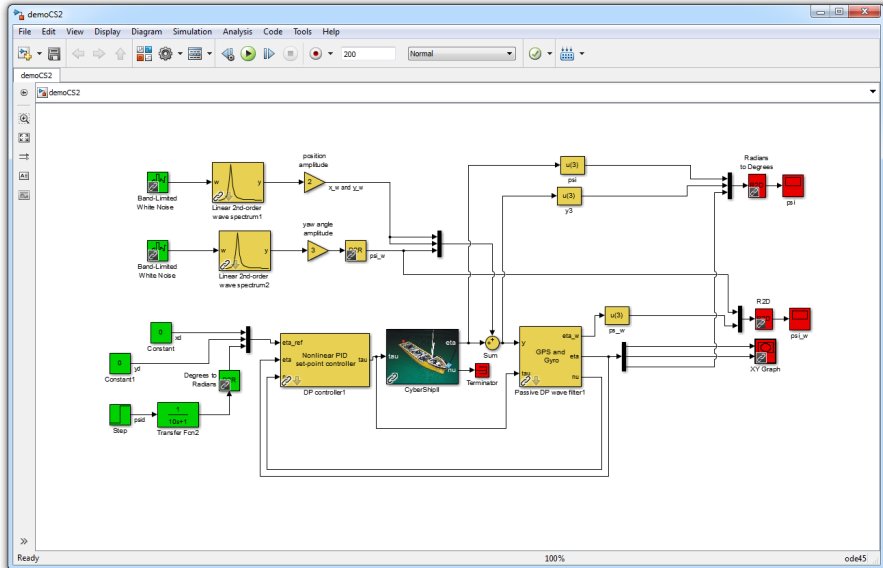
## The Marine Systems Simulator

- MATLAB/Simulink library
- NTNU initiative
- Freeware
- Useful for
  - ▶ guidance, navigation and control
  - ▶ process hydrodynamic data
  - ▶ identification of radiation-force models and fluid memory effects

Get it at

<http://www.marinecontrol.org>

# Example: Dynamic positioning



>> THE END