

Effective Classroom Teaching of Optimal Control and Optimization using MATLAB

Saket Adhau, Sayli Patil, Dayaram Sonawane

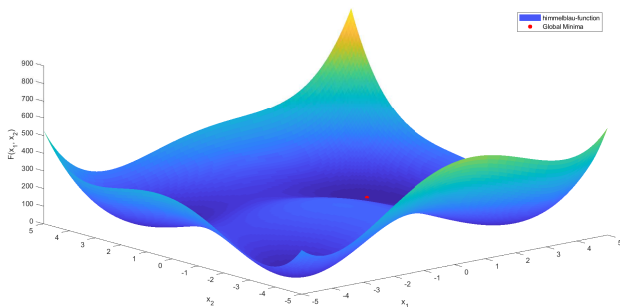
College of Engineering, Pune, India

MATLAB EXPO 2019

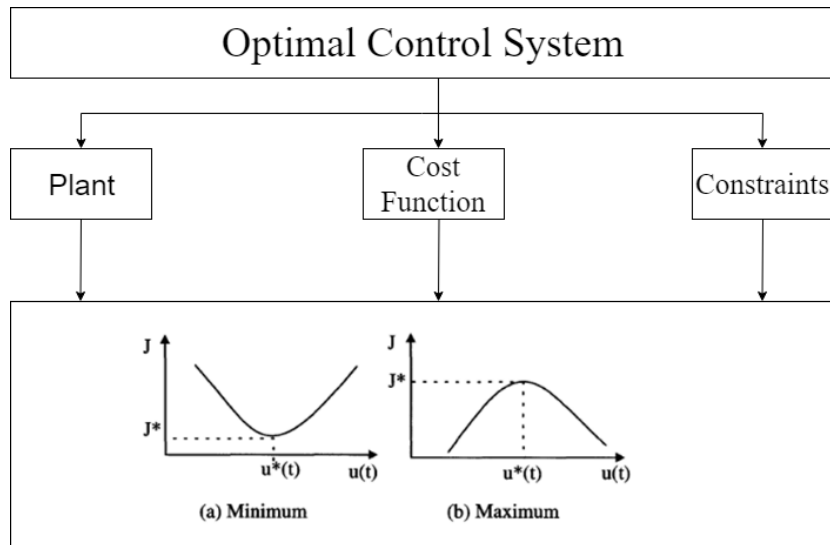
March 23, 2019



- Finding values of the variables that **optimize (minimize or maximize)** the **objective function** while satisfying the **constraints**



Ingredients of Optimal Control Problem

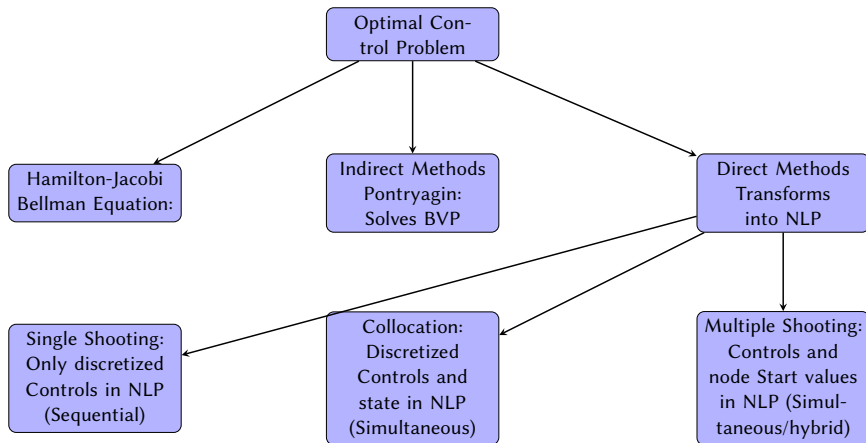


Methods to Solve OCP

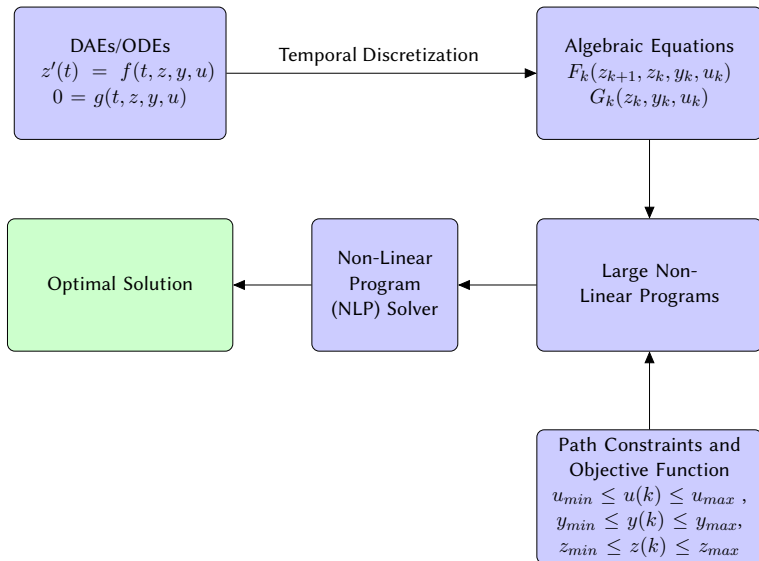
Analytical
Methods –
Pontryagins
maximum principle

Numerical Methods

Numerical Methods to Solve OCP



Optimal Control Problem Flow Chart



Example Problem: Time Optimal Rocket Problem (Time Optimization Problem)

$$\min_U t_f$$

Subject to

$$\dot{s}(t) = v(t); v(\dot{t}) = \frac{(u(t) - 0.02 * v(t)^2)}{m(t)}$$

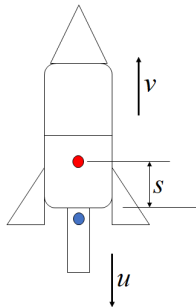
$$\dot{m}(t) = -0.01 * u(t)^2; t \in [0 \quad t]$$

$$s(0) = 0; v(0) = 0; m(0) = 1;$$

$$s(t_f) = 10; v(t_f) = 0$$

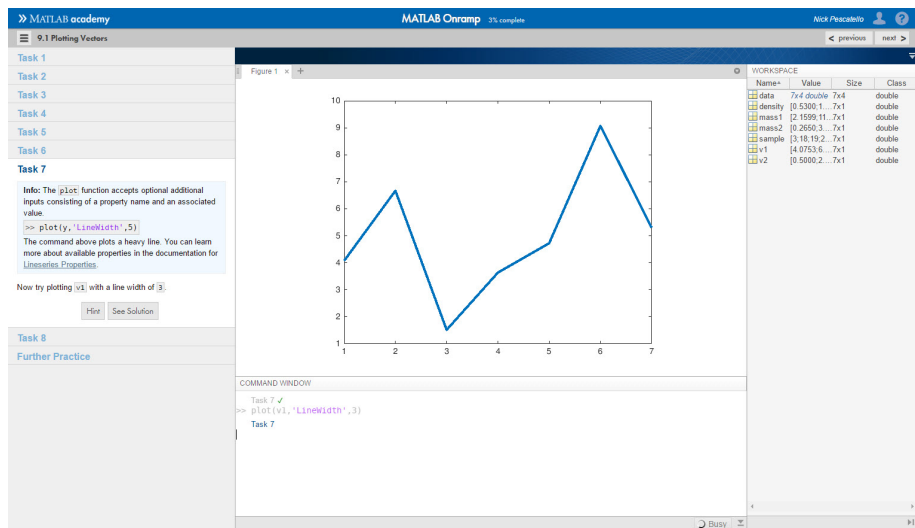
Bounds

$$-0.1 \leq v \leq 1.7; -1.1 \leq u \leq 1.1; 5 \leq T \leq 15$$



Approach: MATLAB and Simulink Onramp

To provide a brief introduction to the MATLAB language and to give students hands-on MATLAB experience via the use of an integrated, web-based version of MATLAB, as shown below.



The screenshot displays the MATLAB Onramp web interface. The top navigation bar includes the MATLAB academy logo, the course title "MATLAB Onramp" with a 3% completion indicator, and the user name "Nick Pescatello". The main content area is divided into several sections:

- Task List:** A vertical list of tasks from Task 1 to Task 7. Task 7 is currently selected.
- Task 7 Information:** An "Info" box explaining that the `plot` function accepts optional additional inputs for property names and values. It shows the command `>> plot(y, 'LineWidth', 5)` and notes that this plots a heavy line. A link to "Lineseries Properties" is provided.
- Task Instruction:** A prompt to "Now try plotting v1 with a line width of 3." Below this are "Hint" and "See Solution" buttons.
- Figure Window:** A plot titled "Figure 1" showing a blue line with a thickness of 3. The x-axis ranges from 1 to 7, and the y-axis ranges from 1 to 10. The data points are approximately (1, 4), (2, 6.5), (3, 1.5), (4, 3.5), (5, 4.5), (6, 9), and (7, 5.5).
- Workspace:** A table listing variables in the workspace:

Name	Value	Size	Class
data	0.5300, 1...	7x4	double
density	2.1599, 11...	7x1	double
mass1	0.2650, 3...	7x1	double
mass2	3.18, 19.2...	7x1	double
sample	4.0753, 6...	7x1	double
v1	0.5000, 2...	7x1	double
v2			
- Command Window:** Shows the execution of the command `>> plot(v1, 'LineWidth', 3)` and a confirmation message "Task 7 ✓".

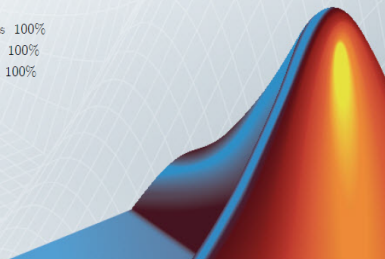
Approach: MATLAB and Simulink Onramp Report

Progress Report

Name: Sayli Patil
Course: MATLAB Onramp
Progress: 100% complete (as of 17-Dec-2018)

Chapters

- | | |
|--|-------------------------|
| 1. Course Overview 100% | 12. Logical Arrays 100% |
| 2. Commands 100% | 13. Programming 100% |
| 3. Vectors and Matrices 100% | 14. Final Project 100% |
| 4. Importing Data 100% | 15. Survey 100% |
| 5. Indexing into and Modifying Arrays 100% | |
| 6. Array Calculations 100% | |
| 7. Calling Functions 100% | |
| 8. Obtaining Help 100% | |
| 9. Plotting Data 100% | |
| 10. Review Problems 100% | |
| 11. MATLAB Scripts 100% | |



Classroom teaching approach: MATLAB Live Script: Interactive approach

Using Conjugate Gradient Method in MATLAB script.

We will try to find optimal solution of function

$$f = x(1) - x(2) + 2 * x(1)^2 + 2 * x(1) * x(2) + x(2)^2;$$

using Conjugate Gradient Method.

Initialize the values

```
1 clear all
2 clc
3 i_max=0;
4 iter=1000;
5 x=[0 0] % Initial Value
6 xi = x' % Transpose
7 iteration=1;
```

Finding the Gradient of the function for the initial value:

```
8 gradf=grad_cost(@costfunc,xi) % Using grad_cost function
9 % which is user defined.
```

Steepest direction

```
10 Si=-gradf
```

Get the function in the form of Lambda

```
11 syms lamda;
12 X=xi+lamda*Si;
```

```
x = 1x2
      0      0
```

```
xi = 2x1
      0
      0
```

```
gradf = 2x1
      1
     -1
```

```
Si = 2x1
     -1
      1
```

```
lamda = 1
```

```
x_new_i =
      (-1)
       1
```

Teaching Optimal Control using Live Script

Check the condition if the solution is optimal

```
15 x_new_i=x_i+ lamda*Si
16 delf_i= grad_cost(@costfunc,x_new_i);
17 if delf_i==0
18     fprintf('Solution is optimum');
19     fprintf('\n');
20     fprintf('Total Number of Iterations = %d\n', imax );
21     return;
22 end
```

Running the loop for finding the Optimal Solution

```
23 for i=1:iter
24     gradf=grad_cost(@costfunc,x_new_i);
25     gradf_old=grad_cost(@costfunc,x_i);
26     Si=(-gradf)+Si*((gradf)*(gradf)*(inv((gradf_old)*(gradf_old))));
27     syms lamda;
28     L=x_new_i+lamda*Si;
29     f=costfunc(L);
30     K=diff(f);
31     lamda=solve(K,lamda);
32     x_new=x_new_i+ lamda*Si;
33     delf_i= grad_cost(@costfunc,x_new);
34     x_i=x_new_i;
35     if delf_i==0
36         fprintf('Solution is optimum');
37         break;
38     end
39     fval_old=costfunc(x_new_i);
40     fval_new=costfunc(x_new);
```

$$z \begin{vmatrix} & -1 \\ & \end{vmatrix}$$
$$Si = 2 \times 1 \begin{vmatrix} & -1 \\ & 1 \end{vmatrix}$$
$$lamda = 1$$
$$x_{new_i} = \begin{pmatrix} -1 \\ 1 \end{pmatrix}$$

Solution is optimum

Teaching Optimal Control using Live Script

Check for stoping criterion

```
41 tol_1=abs((fval_new-fval_old)/fval_old);
42 if(tol_1<1e-6)
43     break;
44 end
45 x_new_i=x_new;
46 end
```

Printing the final values and number of iterations

```
47 imax=i+iteration ;
48 fprintf('\n');
49 fprintf('Solution by ConjugateGradient Method \n');
50 fprintf('\n');
51 fprintf('Total Number of Iterations = %d\n', imax )
52 lamda
53 x_new
54 fval=costfunc(x_new)
55
```

Solution is optimum

Solution by ConjugateGradient Method

Total Number of Iterations = 2

lamda =

$$\frac{1}{4}$$

x_new =

$$\begin{pmatrix} -1 \\ 3 \\ 2 \end{pmatrix}$$

fval =

$$-\frac{5}{4}$$

Assessing the students using MATLAB Grader

MATLAB Grader

Saket Adhau ▾

☰ CONTENTS

Close

[Courses & Content](#) | [LMS Integration](#) | [Documentation & Support](#) ▾

Optimal Control and Linear Model
Predictive Control

Optimal Control and Linear Model Predictive Control

[Edit](#) | [Actions](#) ▾

☰ [Reorder Content](#)

Duration (IST): 14 Mar 2019 - 01 Jun 2019

> Assignment 1 on MPC

Products:

Model Predictive Control Toolbox, Optimization Toolbox

> Assignment 2 on OCP

> State space

Course Description

This course deals with the basics of optimal control and model predictive control which is a special case of optimal control

[ADD ASSIGNMENT](#)

The aim of this course is to give hands-on experience in optimization and model predictive controllers (MPC). The course is recommended for both industrial and academic researchers as well as for masters and Ph.D. students of engineering, computer science, mathematics, and physics.

[Manage People](#)

Assessing the students using MATLAB Grader

MATLAB Grader Saket Adhau ▾

[Courses & Content](#) | [LMS Integration](#) | [Documentation & Support ▾](#)

Optimal Control and Linear Model Predictive Control >

Assignment 2 on OCP Edit | Actions ▾

Visible: 15 Mar 2019 12:00 AM IST | Due: 31 May 2019 12:00 AM IST | Submissions Per Problem: Unlimited

Assignment Description

This assignment is created to check student's understanding of OCP formulation and solving it using MATLAB.

Problems

- Unconstrained optimization method - Conjugate gradient (DRAFT)
- Unconstrained optimization method - BFGS (DRAFT)

[ADD PROBLEM](#)

CONTENTS Close

Reorder Content

- Assignment 1 on MPC
- Assignment 2 on OCP
 - Unconstrained optimization method - Conjugate gradient
 - Unconstrained optimization method - BFGS
- State space

Assessing the students using MATLAB Grader: Report

Problem 10.4

[Edit](#) [Copy](#) [Rescore Solutions](#) [Delete](#)

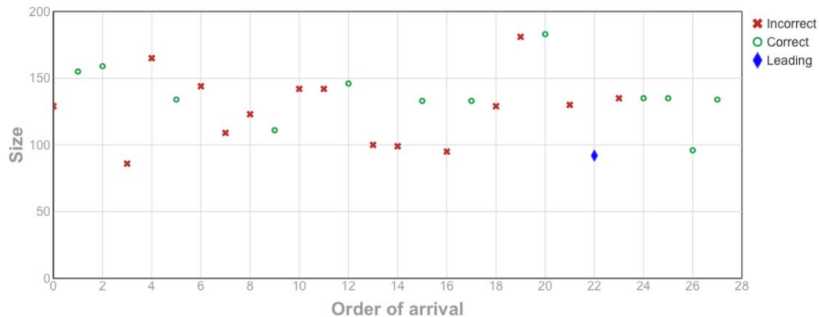
Student Solutions

Search:



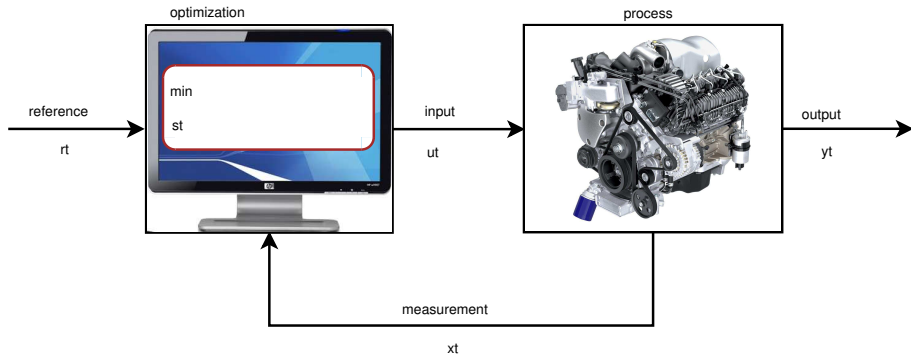
View as:

[List](#) | [Map](#)



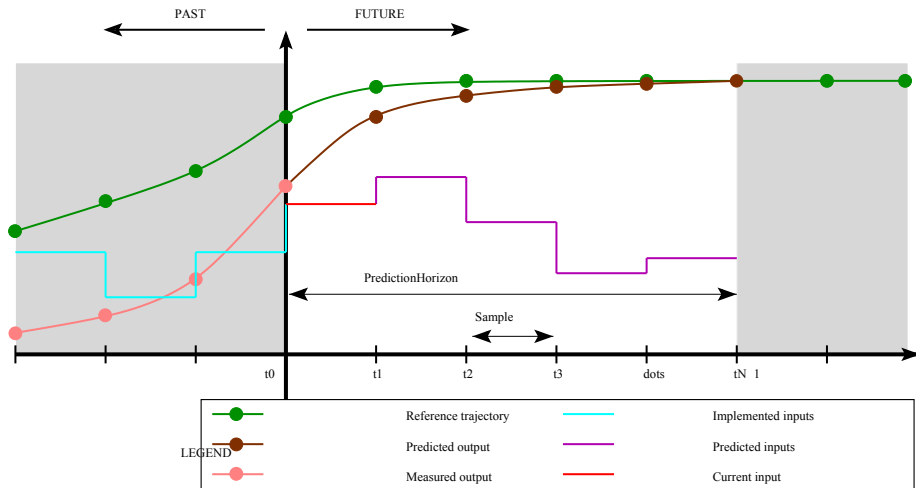
- We have a class of **20** students opting for **Process Modeling and Optimization**.
- As an instructor, it is one challenging task to teach students MATLAB based coding interactively and assessing them individually.
- MATLAB has made this task easier than ever, by introducing MATLAB Live script and MATLAB Grader.

Optimal Control Problem leads to Model Predictive Control (MPC)



Use a dynamical **model** of the process to **predict** its future evolution and choose the “best” control action

Receding Horizon Implementation



Teaching students MPC using Live Editor

Formulating the state space model for MPC

Create a state space model of the plant and set some of the optional model properties.

The State Space model for DC Motor is described as,

$$\begin{bmatrix} \dot{i}_a(t) \\ \dot{\omega}(t) \\ \dot{\theta}(t) \end{bmatrix} = \begin{bmatrix} \frac{-R_a}{L_a} & \frac{-K_b}{L_a} \omega(t) & 0 \\ \frac{-K_t}{J} i_a & \frac{-f}{J} \omega(t) & 0 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} i_a(t) \\ \omega(t) \\ \theta(t) \end{bmatrix} + \begin{bmatrix} \frac{1}{L_a} \\ 0 \\ 0 \end{bmatrix}$$
$$y(t) = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} i_a(t) \\ \omega(t) \\ \theta(t) \end{bmatrix}$$

Define the state space model in matlab script.

```
A = [-Ra/La, -km/La; km/J, -fm/J]
```

```
A = 2x2  
103 x  
    -7.2414    -0.0354  
     1.0524     -0.0000
```

```
B = [1/La; 0]
```

```
B = 2x1  
 862.0690  
         0
```

Teaching students MPC using Live Editor

Create Controller

Create a model predictive controller with a sample time, of 0.0001 second, and with all other properties at their default values.

```
Ts = 0.001;
```

```
MPCobj = mpc(motor,Ts);
```

```
-->Assuming unspecified output signals are measured outputs.  
-->The "PredictionHorizon" property of "mpc" object is empty. Trying PredictionHorizon = 10.  
-->The "ControlHorizon" property of the "mpc" object is empty. Assuming 2.  
-->The "Weights.ManipulatedVariablesRate" property of "mpc" object is empty. Assuming default 0.00000.  
-->The "Weights.ManipulatedVariablesRate" property of "mpc" object is empty. Assuming default 0.10000.  
-->The "Weights.OutputVariables" property of "mpc" object is empty. Assuming default 1.00000.  
    for output(s) y1 and zero weight for output(s) y2
```

Display the controller properties in the Command Window.

```
display(MPCobj)
```

```
MPC object (created on 22-Mar-2019 16:22:13):
```

```
-----  
Sampling time:      0.001 (seconds)  
Prediction Horizon: 10  
Control Horizon:   2
```

```
Plant Model:
```

```
  1 manipulated variable(s) --> | 2 states | --> 2 measured output(s)  
  0 measured disturbance(s) --> | 1 inputs | --> 0 unmeasured output(s)  
  0 unmeasured disturbance(s) --> | 2 outputs |  
                                  -----
```

```
Disturbance and Noise Models:
```

```
Output disturbance model: default (type "getoutdist(MPCobj)" for details)  
Measurement noise model: default (unity gain after scaling)
```

Teaching students MPC using Live Editor

View and Modify Controller Properties

Display a list of the controller properties and their current values.

```
get(MPCobj)
```

```
                Ts: 0.001
PredictionHorizon (P): 10
ControlHorizon (C): 2
                Model: [1x1 struct]
ManipulatedVariables (MV): [1x1 struct]
OutputVariables (OV): [1x2 struct]
DisturbanceVariables (DV): []
                Weights (W): [1x1 struct]
Optimizer: [1x1 struct]
                Notes: {}
UserData: []
                History: 22-Mar-2020 16:22:13
```

The controller is set with default properties, we will modify them according to our purpose.

```
MPCobj.PredictionHorizon = 10;
MPCobj.ControlHorizon = 5;
```

By default, the controller has no constraints on manipulated variables and output variables. Set co

```
MPCobj.MV.Min = 0;
MPCobj.MV.Max = 18;
```

Modify the MPC Properties.

Weights on inputs and outputs state variables.

```
MPCobj.W.ManipulatedVariablesRate = 0.1;
MPCobj.W.OutputVariables = [0 1];
```

Teaching students MPC using Live Editor

Simulating the controller

Time for running the simulation, (10000 control intervals)

```
T = 10000;
```

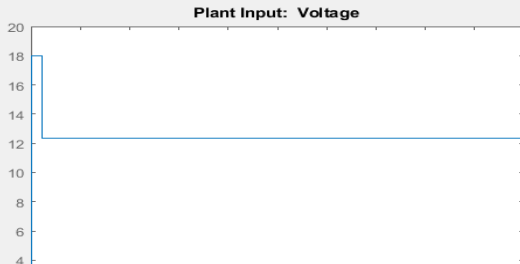
Specify setpoints of 0 and 300 for the Current and the Speed respectively. The setpoint for the Current is ignored because

```
r = [0 0 ; 0 300]
```

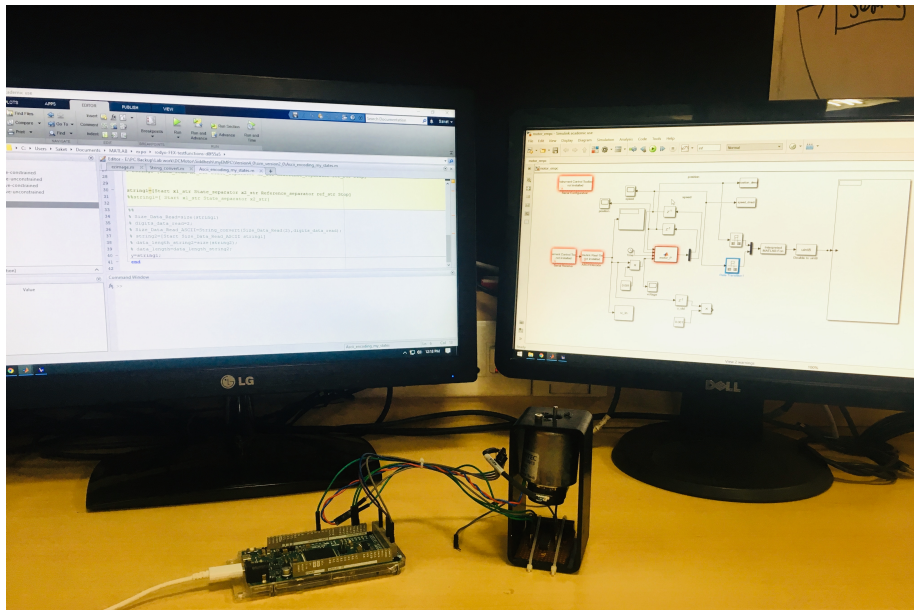
```
r = 2x2
     0     0
     0    300
```

```
sim(MPCobj,T,r)
```

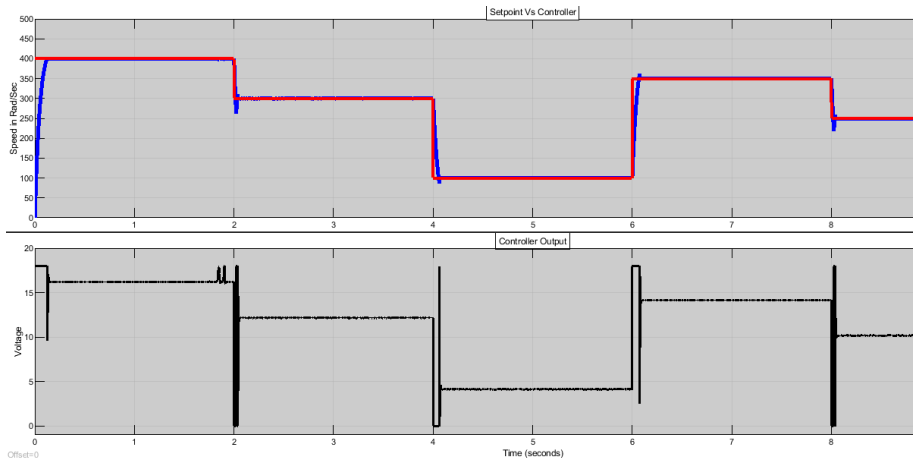
```
-->Converting model to discrete time.
-->Assuming output disturbance added to measured output channel #2 is integrated white noise.
-->Assuming output disturbance added to measured output channel #1 is integrated white noise.
-->The "Model.Noise" property of the "mpc" object is empty. Assuming white noise on each measured output channel.
```



Live Demo using Simulink and Arduino



Live Demo using Simulink and Arduino



The END