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

Optimal Control System

Plant

Cost Function

Constraints

(a) Minimum

(b) Maximum
Methods to Solve OCP

Analytical Methods – Pontryagins maximum principle

Numerical Methods
Numerical Methods to Solve OCP

Indirect Methods
- Pontryagin: Solves BVP

Direct Methods
- Transforms into NLP

Indirect Methods
- Hamilton-Jacobi Bellman Equation:

Direct Methods
- Multiple Shooting: Controls and node Start values in NLP (Simultaneous/hybrid)

Indirect Methods
- Collocation: Discretized Controls and state in NLP (Simultaneous)

Direct Methods
- Single Shooting: Only discretized Controls in NLP (Sequential)
Optimal Control Problem Flow Chart

DAEs/ODEs
\[ z'(t) = f(t, z, y, u) \]
\[ 0 = g(t, z, y, u) \]

Temporal Discretization

Algebraic Equations
\[ F_k(z_{k+1}, z_k, y_k, u_k) \]
\[ G_k(z_k, y_k, u_k) \]

Optimal Solution

Non-Linear Program (NLP) Solver

Large Non-Linear Programs

Path Constraints and Objective Function
\[ u_{min} \leq u(k) \leq u_{max}, \]
\[ y_{min} \leq y(k) \leq y_{max}, \]
\[ z_{min} \leq z(k) \leq z_{max} \]
Example Problem: Time Optimal Rocket Problem (Time Optimization Problem)

$$\min_U t_f$$

Subject to

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

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

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

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

Bounds

$$-0.1 \leq v \leq 1.7; \quad -1.1 \leq u \leq 1.1; \quad 5 \leq T \leq 15$$
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.
Progress Report

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

Chapters
1. Course Overview 100%
2. Commands 100%
3. Vectors and Matrices 100%
4. Importing Data 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%
12. Logical Arrays 100%
13. Programming 100%
14. Final Project 100%
15. Survey 100%
Using Conjugate Gradient Method in MATLAB script.

We will try to find optimal solution of function

\[ f = x(1) - x(2) + 2 \times x(1)^2 + 2 \times x(1) \times x(2) + x(2)^2; \]

using Conjugate Gradient Method.

Initialize the values

```matlab
clear all
c1c
% kmx=0;
iter=1000;
x=[0 0] % Initial Value
xi = x' % Transpose
iteration=1;
```

Finding the Gradient of the function for the initial value:

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

Steepest direction

```matlab
Si=-gradf
```

Get the function in the form of Lambda

```matlab
syms lamda;
X=xi+lamda*Si;
```
Check the condition if the solution is optimal

```matlab
x_new_i = xi + lambda*Si

delf_i = grad_cost(@costfunc,x_new_i);
if delf_i == 0
    fprintf('Solution is optimum');
    fprintf('
');
    fprintf('Total Number of Iterations = %d\n', imax);
    return;
end
```

Running the loop for finding the Optimal Solution

```matlab
for i=1:iter
    gradf = grad_cost(@costfunc,x_new_i);
    gradf_old = grad_cost(@costfunc,xi);
    Si = (-gradf + Si * ((gradf') * (gradf) * (inv((gradf_old') * (gradf_old))));
    sym l = lambda;
    l = x_new_i + lambda*Si;
    f = costfunc(l);
    K = diff(f);
    lambda = solve(K, lambda);
    x_new = x_new_i + lambda*Si;
    delf_i = grad_cost(@costfunc,x_new);
    xi = x_new_i;
    if delf_i == 0
        fprintf('Solution is optimum');
        break;
    end
end
fval_old = costfunc(x_new_i);
fval_new = costfunc(x_new);
```
Check for stopping criterion

tol_1=abs((fval_new-fval_old)/fval_old);
if(tol_1<1e-6)
    break;
end
x_new=x_new;
end

Printing the final values and number of iterations

imax=i+iteration;
fprintf('n');
fprintf('Solution by ConjugateGradient Method n');
fprintf('n');
fprintf('Total Number of Iterations = %d
', imax)
lambda
x_new
fval=costfunc(x_new)

Solution is optimum

Solution by ConjugateGradient Method

Total Number of Iterations = 2
lambda -
\[
\begin{pmatrix}
\frac{1}{4} \\
-1 \\
\frac{3}{2} \\
\frac{5}{4}
\end{pmatrix}
\]
Optimal Control and Linear Model
Predictive Control

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

Products:
Model Predictive Control Toolbox, Optimization Toolbox

Course Description

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

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.
Assignment 2 on OCP

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)
Assessing the students using MATLAB Grader: Report

Problem 10.4

Student Solutions

Search:
Enter student's last name or solution ID

View as:
List  |  Map

Size

Order of arrival
Case Study:

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

PAST FUTURE

Measured output
Predicted output
Reference trajectory Implemented inputs
Predicted inputs
Current input

t1 t2 t3 dots tN 1
PredictionHorizon
t0
Sample

LEGEND

Reference trajectory
Predicted output
Measured output
Implementated inputs
Predicted inputs
Current input
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,

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

\[
y(t) = 
\begin{bmatrix}
1 & 0 & 0 \\
0 & 1 & 0 \\
0 & 0 & 1
\end{bmatrix}
\begin{bmatrix}
i_a(t) \\
\omega(t) \\
\theta(t)
\end{bmatrix}
\]

Define the state space model in matlab script.

\[
A = \begin{bmatrix}
-Ra/La, -km/La; km/J, -fm/J
\end{bmatrix}
\]

\[
A = 2\times2 \\
10^3 \times \\
-7.2414 \quad -0.0354 \\
1.0524 \quad -0.0000
\]

\[
B = \begin{bmatrix}
1/La; \theta
\end{bmatrix}
\]

\[
B = 2\times1 \\
862.0690 \quad \theta
\]
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.ManipulatedVariables" 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)


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)
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];
```
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

\[ r = [0 \ 0 \ ; \ 0 \ 300] \]

\[ r = 2x2 \]

\[ \begin{bmatrix} 0 & 0 \\ 0 & 300 \end{bmatrix} \]

\[ \text{sim(MPCobj,T,r)} \]

-->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.

---

![Plant Input: Voltage](image-url)
Live Demo using Simulink and Arduino
Live Demo using Simulink and Arduino

Setpoint Vs Controller

Controller Output

Saket Adhau (COEP)
Optimal Control
March 23, 2019
The END