McMaster University

MECHTRON 4AX3 , Term I 2023

Assignment 1 Assigments are handed in on paper.

- The derivation of the discrete system matrix by hand
- The printout of your code (dont waist paper, this fits on one sheet used two sided ...
- The printout of the plots.

1) Simple System

We want to simulate the throwing of a ball. (no drag). The ball is thrown at some initial velocity `v0` at some angle `alpha`.

```
v0x=v0 cos(alpha) v0y= v0 sin(alpha)
```

There are no forces in x, and gravity is acting opposing the Y direction. So the differential equation is:

```
x"=0
y"=-g
```

Set up a 4 dimensional matrix `(pos_x,vel_x,pos_y,vel_y)`, by hand. Also find by hand a discrete system that approximates the above with some step size h. (you can use a simple Taylor expansion).

2) System Simulation

We need tools to design and simulate systems. The point of this assignment is to get the tools installed and learn (or practice) the basics.

- Simulator the continuous system using the ode solver or the Runge-Kutta given in my the sample code.
- Simulate the discrete version of the system you computed above.

Eigen

Eigen is a C++ Linear algebra Library. Install Eigen on your computer. This works for most C++ compilers, Linux, OSX and Windows.

- Go over the Eigen tutorial, so you see how to compute with matrix and vector expressions.
- You also will need a tool to visualize your data, plot your data. I use gnuplot (on Linux and OSX there are mangers that make installing all the software easy, e.g. brew on OSX).

Matlab, Octave

Matlab or the free GNU Octave are the common design tools for control engineers. You can either use Matlab on our server, or install yourself the free octave (I use octave in this course).

- There are many tutorials to cover simple LTI systems. We only need very few Matlab commands. Look for example at Tutorial

3) Extra

C++ and Eigen.

- Write a program to simulate a discrete LTI system given by the 3 matrices: A,B,C below:

```
Eigen::MatrixXd A(2, 2);
Eigen::MatrixXd B(1, 2);
Eigen::MatrixXd C(1, 2);

A<<1,1,0,.5;
B=<<0,1;
C=<<1,0;
```

- Ideally we want to write a class for discrete LTI systems. I will give you such code as a solution and do not expect that you know how to do it. (This is not a programming course, but this is actually simple to do and code worth having handy, below is a start).

```
class LTISystem{
public:
 LTISystem(
      const Eigen::MatrixXd& A,
      const Eigen::MatrixXd& B,
      const Eigen::MatrixXd& C
  );
  // Initialize with initial states.
  void init(const Eigen::VectorXd& x0);
  //* Update  with control u
  void update(const Eigen::VectorXd& u);
  /// GET a INFO
  Eigen::VectorXd state() { return stateX; };
  Eigen::VectorXd output();

.......
```