Problem Set 8
Due: Fri. 5 Dec. 2014 by 9am (beginning of class).

Note: All MATLAB functions mentioned in this homework assignment can be found on the CAAM453 homepage, or come with MATLAB. You can use the MATLAB codes posted on the CAAM453 web-page. If you modify these codes, please turn in the modified code. Otherwise you do not have to turn in printouts of the codes posted on the CAAM453 web-page. Turn in all MATLAB code that you have written and turn in all output generated by your MATLAB functions/scripts. MATLAB functions/scripts must be commented, output must be formatted nicely, and plots must be labeled.

Problem 1 (50 points)
A “double pendulum” consists of a pair of pendula which are connected to each other by a pin joint (see Figure 1). The first pendulum is attached to a fixed point. The two pendula being respectively of lengths \( l_1 \) and \( l_2 \) and of masses \( m_1 \) and \( m_2 \), the position of the two pendula can be represented by two angles \( \theta_1 \) and \( \theta_2 \). The kinetic energy \( T \) and the potential energy \( U \) of the system are written in terms of \( \theta_1(t) \) and \( \theta_2(t) \) as

\[
T = \frac{m_1 + m_2}{2} l_1^2 \dot{\theta}_1^2 + \frac{m_2}{2} l_2^2 \dot{\theta}_2^2 + m_2 l_1 l_2 \dot{\theta}_1 \dot{\theta}_2 \cos(\theta_1 - \theta_2),
\]

\[
U = -(m_1 + m_2) l_1 g \cos(\theta_1) - m_2 l_2 g \cos(\theta_2).
\]

Lagrange’s equations of motion are in the form

\[
\frac{d}{dt} \left( \frac{\partial L}{\partial \dot{\theta}_\alpha} \right) - \frac{\partial L}{\partial \theta_\alpha} = 0, \quad \alpha = 1, 2. \tag{1}
\]

with the Lagrangian of the system \( L = T - U \).

Figure 1: Double pendulum
• Write the two equations of motion in the form
\[
\begin{bmatrix}
  m_{11} & m_{12} \\
  m_{12} & m_{22}
\end{bmatrix}
\begin{bmatrix}
  \dot{\theta}_1 \\
  \dot{\theta}_2
\end{bmatrix}
= \begin{bmatrix}
  f_1 \\
  f_2
\end{bmatrix}
\]
with \( m_{11}, m_{12}, m_{22}, f_1 \) and \( f_2 \) functions of \( \theta_1, \dot{\theta}_1, \theta_2, \) and \( \dot{\theta}_2. \)

• Matrix \( M \) is symmetric and invertible. Write the equations of motion as
\[
\begin{bmatrix}
  \ddot{\theta}_1 \\
  \ddot{\theta}_2
\end{bmatrix}
= \begin{bmatrix}
  \theta_1 \\
  \theta_2
\end{bmatrix} \begin{bmatrix}
  \dot{\theta}_1 \\
  \dot{\theta}_2
\end{bmatrix}
\]
\[
\begin{bmatrix}
  \ddot{\theta}_1 \\
  \ddot{\theta}_2
\end{bmatrix}
= M^{-1} \begin{bmatrix}
  f_1 \\
  f_2
\end{bmatrix}
\]

• Write this system of 2 second order ODE’s as a system of 4 first order ODE’s. For that, introduce
the 4 variables
\[
y_1 = \theta_1, \quad y_2 = \theta_2, \quad y_3 = \dot{\theta}_1, \quad y_4 = \dot{\theta}_2
\]
and write
\[
\begin{bmatrix}
  \dot{y}_1 \\
  \dot{y}_2 \\
  \dot{y}_3 \\
  \dot{y}_4
\end{bmatrix}
= A
\begin{bmatrix}
  y_1 \\
  y_2 \\
  y_3 \\
  y_4
\end{bmatrix}
\]
with \( A \in \mathbb{R}^{4 \times 4}. \)

• Write a MATLAB code that integrates (in time, from \( t = 0 \) to \( t = 60 \)) this system of 4 ODE’s. Use
\( m_1 = m_2 = 1, \theta_1(0) = \theta_2(0) = \pi/4, \dot{\theta}_1(0) = \dot{\theta}_2(0) = 0. \) You can use any method that has
been developed in class or any other method of your choice (explicit, implicit, multistep, one-step). As an output, you should provide two graphs: (i) one graph that draws the position \((x,y)\)
of mass \( m_2 \) as a function of time and (ii) a graph that draws \((\theta_2, \dot{\theta}_2)\) as a function of time.

• The total energy of the system should be conserved over time: \( E = T + V \) should not depend on
time. Provide a graph \( E(t) \) for \( t = 0 \) to \( t = 600. \) Is the energy conserved numerically and why?

Problem 2 (50 points)

You are to write Matlab codes LUfac, Lsolve and Usolve based upon the algorithms presented in class.
LUfac should implement LU-factorization with partial pivoting for stability. Lsolve should implement the
lower triangular solve and Usolve should implement the upper triangular solved needed to obtain
the solution to a given linear system using the output of LUfac.

Honor Code: For this code you are not allowed to search the internet or any other source for an
implementation. You can consult other written textbooks and look a printed descriptions if you wish,
but you may not copy any computer codes.

i. (20 points) Your LUfac code should check \( A \) on input, set a flag and exit if \( A \) is not square.
It should satisfy the following specifications:
% Usage
% [A, p, iflag] = LUfac( A )
%
% input:
% A: the n by n matrix A
% output:
% A: Overwritten with the LU-decomposition of A
% p: a vector containing pivot information
% iflag: error flag
% iflag = 0 The LU Decomposition completed successfully
% iflag = 1 A is not square

ii. (10 points) Your Lsolve code should check input A and b to see that dimensions match up. If not, set a flag and exit.

It should satisfy the following specifications:

% Usage
% [c, iflag] = Lsolve(A, p, b)
%
% Solves L c = b with L in lower triangle of A
% Includes pivoting at each step.
%
% input:
% A: the n by n matrix A overwritten with its LU decomposition computed by LUfac
% Only the strict lower triangle is referenced.
% p: an n-vector containing pivot information
% b: an n-vector containing the right hand side
%
% output:
% c: an n-vector containing the solution
%
% iflag: error flag
% iflag = 0 the solution completed successfully
% iflag = 1 dimensions of A and b are incorrect
iii. (10 points) Your Usolve code should check input A and c to see that dimensions match up. If not, set a flag and exit. During the backsolve, you must test for an exact zero on the diagonal, set a flag and exit if this occurs.

It should satisfy the following specifications:

```matlab
% Usage
% [x, iflag] = Usolve(A, c)
% Solves U x = c with U in upper triangle of A
% input:
% A: the n by n matrix A overwritten with its
% LU decomposition computed by LUfac
% Only the upper triangle is referenced.
% c: an n-vector containing the right hand side
% output:
% x: an n-vector containing the solution
% iflag: error flag
% iflag = 0 the solution completed successfully
% iflag = 1 dimensions of A and c are incorrect
% iflag = k an exact zero is detected in A(k-1,k-1)
% U is singular.
```

iv (10 pts) Run the code testLUfac.m using your LUfac, Lsolve, Usolve. You must use testLUfac.m as given (see assignment page). Do not modify this code. Please turn in the printed output from this run.