MTH 420/520 Homework 2. **Yesh Godse**
Date: April 21, 2019
Prof. M. Peszynska Due: April 22, 2019

MTH 420 students turn in 1-2.
MTH 520 students turn in 1-3.
Extra credit is marked.
Show enough work and enough of `MATLAB code` to demonstrate this is your own hard work, but be concise.
Sloppy incomplete work, or lengthy algebraic calculations, or core dump of MATLAB output with errors will not receive much credit.

Problem 1. (A) Find the Euler-Lagrange equations for the functional $J(u) = \int_0^1 4(u')^2 - e^x u$. Find the extremal in $V = \{v \in C^2[0,1] : v(0) = 0\}$. What could this model?
(B) The problem $-\varepsilon u'' + 9u = 1, u(0) = 0 = u(1)$ models diffusion and reaction, with a constant source equal 1, diffusivity $\varepsilon > 0$, and reaction constant 9. Its solution is the minimizer of a certain functional $J(u)$. Find $J(u)$.
(C) Find the solution $u(x)$ to (B), call it $u_\varepsilon(x)$, and plot (choose $\varepsilon = 1, 0.1, 0.01$, and more). Describe what happens as $\varepsilon \to 0$. (Pay attention to the boundary layer). Why?
This problem is known as the singularly-perturbed problem.
Hint for C: the solution is $u_\varepsilon(x) = A + B\exp(r_1 x) + C\exp(r_2 x)$, with appropriately chosen r_1, r_2, A, B, C. If you use this Ansatz, please justify.

Solution - A:. According to the class notes, we can define the Euler-Lagrange equations for a function $J(u)$ as the following function $\delta J(u)$:

$$\delta J(u) = \frac{d}{dt}(J(u+tv))\bigg|_{t=0} \tag{1}$$

From definition 9 of the class notes, we also know that the general form of Euler-Lagrange equations for the functional $J(u) = \int_a^b L(x, u(x), u'(x))dx$ over $u \in C^2[0,1]$) is:

$$\mathcal{L}_u - \frac{d}{dx}\mathcal{L}_{u'} = 0$$

Applying this to our functional $J(u)$, we derive the following Euler-Lagrange equation.

$$\delta J(u) = -e^x - \frac{d}{dx}(8u') = 0 \implies -8u'' = e^x \tag{2}$$

For thoroughness, we will now arrive at the same solution without relying on the general form of the Euler-Lagrange equation defined in the class notes. Instead, we will deduce the differential equation for the minimizer $u(x)$ by calculating the first variation $\frac{d}{dt}J(u+tv)\Big|_{t=0}$.
To begin, we define the following:

$$\frac{d}{dt}J(u+tv)\bigg|_{t=0} = \lim_{t\to 0}\frac{J(u+tv) - J(u)}{t} = 0$$

We proceed step by step by first evaluating $J(u+tv)$:

$$J(u+tv) = \int_0^1 4(u'+tv')^2 - e^x(u+tv)$$

$$= \int_0^1 \Big[4((u')^2 + 2u'tv' + t^2(v')^2) - e^x(u) - tve^x\Big]dx$$

By substituting this and $J(u)$ into $\frac{1}{t}(J(u+tv) - J(u))$, we get

$$\frac{1}{t}(J(u+tv) - J(u)) = 4 * \int_0^1 \Big[2u'v' + t(v')^2\Big]dx + \int_0^1 e^x v dx$$

Taking the limit as $t \to 0$, integrating by parts, and factoring out a constant multiplier of 8, and setting the expression equal to 0 in accordance with (1) gives us

$$8\int_0^1 (u'' - e^x)v + u'(1)v(1) = 0$$

As explained in the course notes and clear upon observation, we know that testing this equation with some v that happens to satisfy $v(1) = 0$ let's us deduce the Neumann condition that $u'(1) = 0$ and conclude the following boundary value problem. Note that this is the same result as (2).

$$-8u'' - e^x = 0 \ , \ u(0) = 0 \ , \ u'(1) = 0 \tag{3}$$

Solving this boundary value problem (BVP) is done by isolating u'' and integrating both sides twice. This gives

$$u = -\frac{e^x}{8} + Ax + B \ , \ u(0) = 0 \ , \ u'(1) = 0$$

Using the boundary conditions to determine constants A and B gives us the final solution

$$u(x) = \frac{1}{8}(ex - e^x + 1) \tag{4}$$

Solution - B. Because the solution to $-\epsilon u'' + 9u = 1$ is a minimizer of $J(u)$, we know that it will be described in the Euler-Lagrange equations for $J(u)$. Working backwards, we can deduce $J(u)$ by building the Euler-Lagrange equations of $J(u)$. The two components of this equation are $\mathcal{L}_u$ and $\mathcal{L}_{\sqcap'}$. Following the general form of Euler-Lagrange equations given in the class notes, we can try out the following for these components:

$$\mathcal{L}_u = 9u - 1$$
$$\mathcal{L}_{u'} = \epsilon u'$$

To complete the functional $J(u)$ we simply integrate this Lagrangian $\mathcal{L}(u, u')$ in [0,1]:

$$J(u) = \int_0^1 \Big[\frac{\epsilon}{2}(u')^2 + \frac{9}{2}u^2 - u\Big]dx \tag{5}$$

Solution - C. We can rewrite the differential equation describing the minimizer $u(x)$ to (5) as follows:

$$u'' = \frac{1 - 9u}{-\epsilon},\ u(0) = u(1) = 0$$

Integrating both sides of this equation twice and simplifying gives the following equation for u with the same set of initial conditions:

$$u(x) = C_1 e^{-\frac{3}{\sqrt{\epsilon}}} + C_2 e^{\frac{3}{\sqrt{\epsilon}}} + \frac{1}{9},\ u(0) = u(1) = 0$$

After substituting the initial conditions and solving for constants C_1 and C_2 we get the final solution $u_\epsilon(x)$:

$$u(x) = \frac{e^{\frac{3}{\sqrt{\epsilon}}} - 1}{9(e^{\frac{-3}{\sqrt{\epsilon}}} - e^{\frac{3}{\sqrt{\epsilon}}})} * e^{-\frac{3}{\sqrt{\epsilon}}} - \frac{e^{\frac{-3}{\sqrt{\epsilon}}} - 1}{9(e^{\frac{-3}{\sqrt{\epsilon}}} - e^{\frac{3}{\sqrt{\epsilon}}})} e^{\frac{3}{\sqrt{\epsilon}}} + \frac{1}{9} \tag{6}$$

This can be simplified, but it follows the given pattern for $u_\epsilon(x)$ given in the hint to part B. The following MATLAB code plots $u_\epsilon(x)$ for $\epsilon \in [1, .1, .01, .001, .0001]$ in the same figure. The parameter *uSol* is the function $u_\epsilon(x)$ expressed in MATLAB's symbolic math toolbox. y is a symbolic variable that represents ϵ

```
fplot([subs(uSol, y, 1) subs(uSol, y, .1) subs(uSol, y, .01) ...
    subs(uSol, y, .001) subs(uSol, y, .0001)],[0,1]);
```

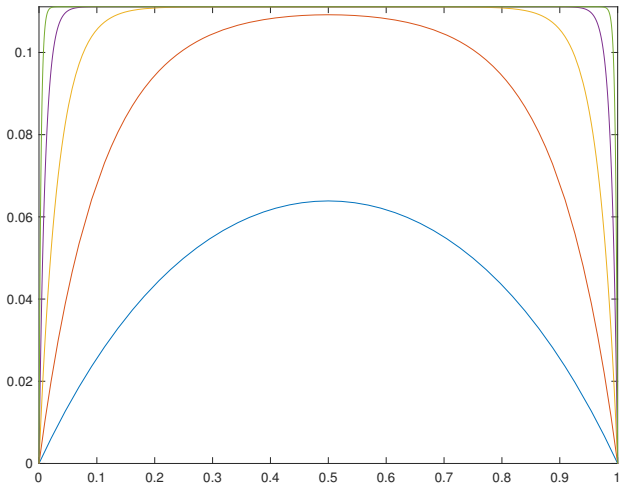

FIGURE 1. Plot of $u_\epsilon(x)$ for varying values of $\epsilon \to 0$

Clearly, as $\epsilon \to 0$, the shape of $u_\epsilon(x)$ more closely resembles a square wave with an aplitude of $1/9$. This makes sense, as an infinitely small ϵ drives the two non-constant terms of $u_\epsilon(x)$ to 0.

Problem 2. (A) Set up a (simplified) network corresponding to OSU campus. Buildings correspond to the nodes (vertices), and the sidewalks or roads are the edges of the network. Assign lengths l_j to the edges, and set the conductivities $c_j = 1/l_j$. Include as nodes: Reser stadium, Valley Library, Kidder Hall, a bus stop on Monroe, parking lot behind Burt Hall, and at least three more buildings.
(B) Set up an equilibrium problem of crowd modeling, solving for density of people x_i on Wednesday April 17 at 11:30am, at every node i. (Assume Fick's law and that people prefer to disperse rather than aggregate) Assume $x_{busstop} = 0$, and $x_{parking} = 0$. (This means that a bus connection and Uber of infinitely large capacity take people instantaneously away from these locations). Also, assume $x_{library} = 100$. (All 100 seats in the library are always taken). Use MATLAB to solve (B). Use `scatter` to show the people density on a "map" (of course you must assign some (simplified) location $(long_i, latt_i)$ to each node. You do not have to print in color, but if you don't, please upload the color picture to CANVAS. (Use `title` to label the graph with your name.) Compare to the case without Uber.
Hint: see

```
>> long=[0,1,1,0]'; latt=[0,0,1,1]'; people=[10,3,2,0]';
>> scatter(long,latt,100+people,people,'filled'); axis([ -3 3 -3 3])
>> print -dpng mymap.png
```

Extra: Push yourself to create a compelling problem, with many nodes, edges, and interesting modeling assumptions on boundary conditions and sources. Be creative.

Solution - A. The following MATLAB code constructs a map of OSU including Memorial Union, Kelly Engineering Center, Dearborn Hall, Graf all, and the required locations in problem 2. This code is purely for the visual and has no impact on the equilibrium problem set up in the solution to part B.

```
G = graph(Adjacency); % 'Adjacency' is an adjacency matrix
x_graph = [0  10 3 8 10 6 6 3 0];
y_graph = [12 12 8 9 11 7 6 6 0];
plot(G,'XData',x_graph,'YData',y_graph,'EdgeLabel',G.Edges.Weight)
title('Map of OSU')
```

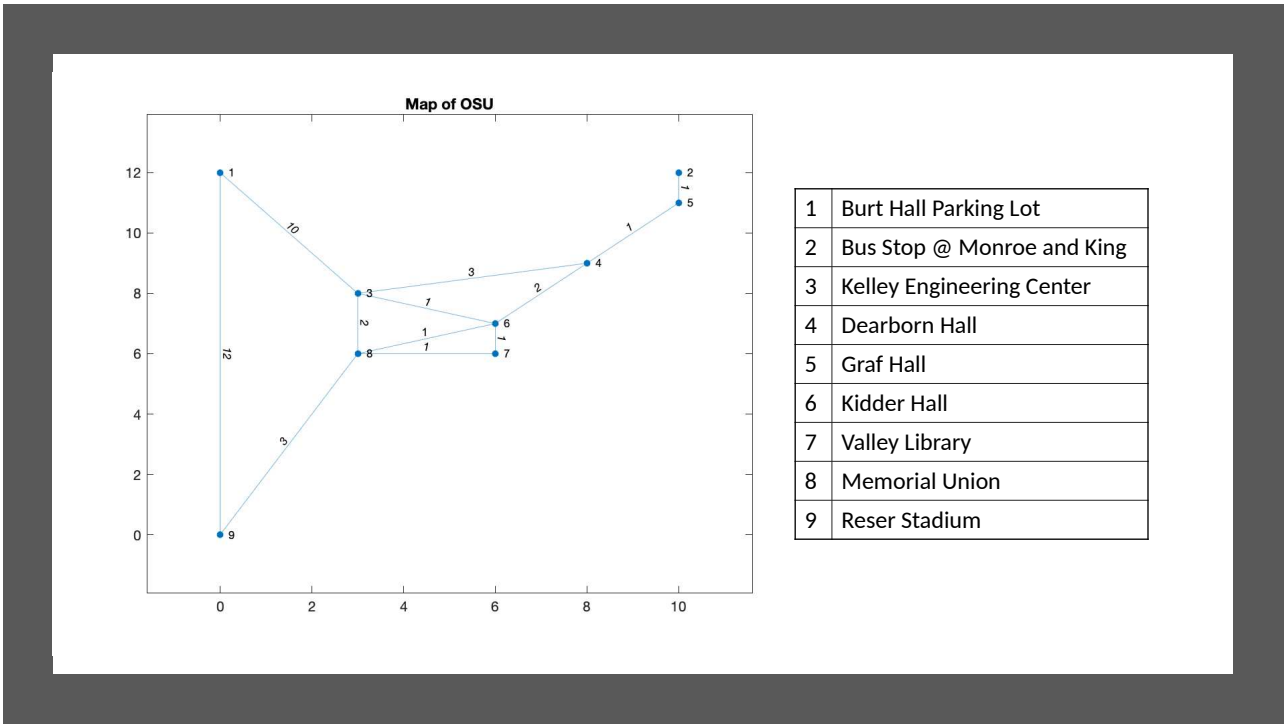

FIGURE 2. Simplified Map of OSU

Solution - B. We will now proceed to set up an equilibrium problem of crowd modeling with the network from part A. To do so, we first identify our given and unknown information.

In this problem there are 9 locations, which we can call m_i, $i = 1...9$. Along with these locations are external forces f_i. There are 11 edges connecting these 9 locations, so we can define our displacements x_i, elongations e_j, diffusion constants c_j, and internal forces y_j for $j = 1...11$. We are given m_i, fi, and c_j. We need to find unknowns x_i, e_j, and y_j to solve the equilibrium problem.

We define e_j for some edge j between locations m_{i_1} and m_{i_2} as the difference of the corresponding displacements x_{i_1} and x_{i_2}. We define the internal forces y_j as $-\frac{1}{c_j}e_j$. Finally, we define the external forces f_i for each node as the sum of internal forces y_j that feed into or feed out of the particular location m_i. By setting up these equations as such, we get a linear system of equations that can be further reduced to form the final equilibrium problem relating our principle known quantity, f_i with our unknown quantity, x_i. The following MATLAB code uses an intermediate matrix A along with the given conductivities C to calculate K in $Kx = f$.

```
%% Create Equilibrium Model of the Network Defined Above
l = [10 5 12 1 3 2 1 2 1 1 3];
c = l.^(-1);
C = diag(c);
A = [-1 0 1 0 0 0 0 0 0;...  % 11x9 matrix that is shown abridged
```

```
    0  1 0 0 -1 0 0 0 0;... % here to avoid unnecessary detail
  ...0 0 0 0 0 0 1 -1 0;...
    0 0 0 0 0 0 0 1 -1;];
K = A'*C*A;
```

In this equilibrium problem we are given the boundary conditions that x_0 and $x_1 = 0$. We are also given that Valley Library is always filled with 100 people, so $x_7 = 100$. Because the two edges interacting with the library node have conductivities of 1, we can set their initial populations to 50. This can be verified by going through the equations describing how f_i is related to multiple y_j values. For a less compelling problem where the library has only one edge going out, checking with these equations is unnecessary. For a more compelling problem where the edges going out of the library have conductivities other than 1, or there are much more than two edges, going through these equations is most likely unavoidable.

The following MATLAB code constructs the reduced problem after these calculations, and graphs a scatterplot of the solved equilibirum problem overlaid over the initial map in figure 3 below.

```
K_reduced = K([3:6 8:9], [3:6 8:9]);
x_1 = 0;
x_2 = 0;
x_7 = 100;
Externals = [0 0 0 x_7/2 x_7/2 0]';
x = K_reduced \ Externals;
people = cat(1, [x_1 x_2 x(1) x(2) x(3) x(4) x_7 x(5) x(6)])
scatter(x_graph',y_graph',people+100,people,'filled');
```

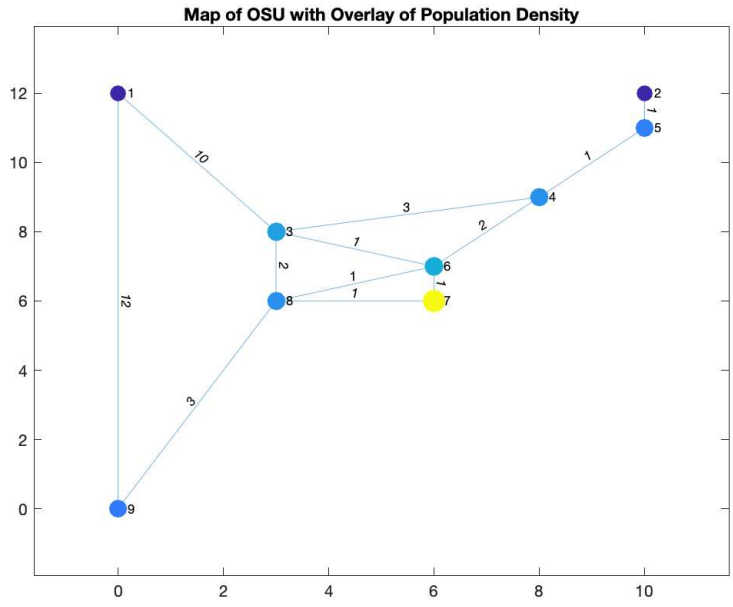

FIGURE 3. Scatter-plot of population density at different locations

This code segment also calculates the following vector for the population at each location. As expected, there is a gradual decrease in population as we go from the library to the sinks at the Burt Hall parking lot and the bus stop.

```
people =
     0   0   36.1446   32.5301   27.1084   40.9639   100.0000   31.9149   25.5319
```

If we modify the network by removing the Uber stop at Burt hall, which is node 1, we simply modify A and recompute the reduced equilibrium problem. The rest of the code stays the same. Doing this produces the following population map of OSU and output vector.

```
people =
         0   0   42.0000   36.0000   30.0000   44.0000   100.0000   31.9149   25.5319
```

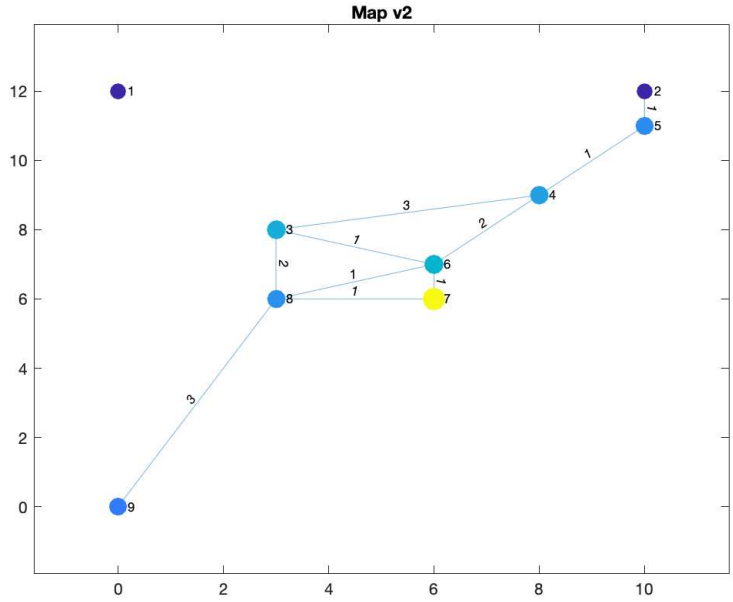

FIGURE 4. Scatter-plot of population density at different locations, without node 1

Problem 3. Solve the time dependent problem for $x(t)$ extending the equilibrium model from Pbm 2 to $\frac{dx}{dt} + Kx = 0$, which would show the "flow of people" on campus leaving after a game. Use as initial condition $x_i(0) = 0$ at all nodes except at Reser where $x_{Reser}(0) = 45000$. If you must, exclude the condition at $x_{library}$.
This can be done in two ways: with `ode45` as in HW1, or with discrete time stepping where

$$x(t + \Delta t) = x(t) - \Delta t K x(t). \tag{7}$$

Here your Δt should be small enough for stability (since you are using explicit time-stepping). In fact, choose it to be $\Delta t < \lambda_{max}(K)$. (What happens if this is significantly violated?).
Report on the solution with an `*.mp4` movie (upload to CANVAS) or a sequence of properly labelled images to be uploaded to a website whose link you would share.
Extra: use implicit time stepping where $x(t + \Delta t) = x(t) - \Delta t K x(t + \Delta t)$.

Solution. The following MATLAB code solves the differential equation $\frac{dx}{dt} = -Kx$ with `ode45`.

```
tspan = [0 10];
x0 = zeros(1,9);
x0(9) = 45000; % set reser stadium to 45000
[t, x_t] = ode45(@(t,x_t) -K*x_t, tspan, x0);
figure();
plot(t,x_t,'-');
```

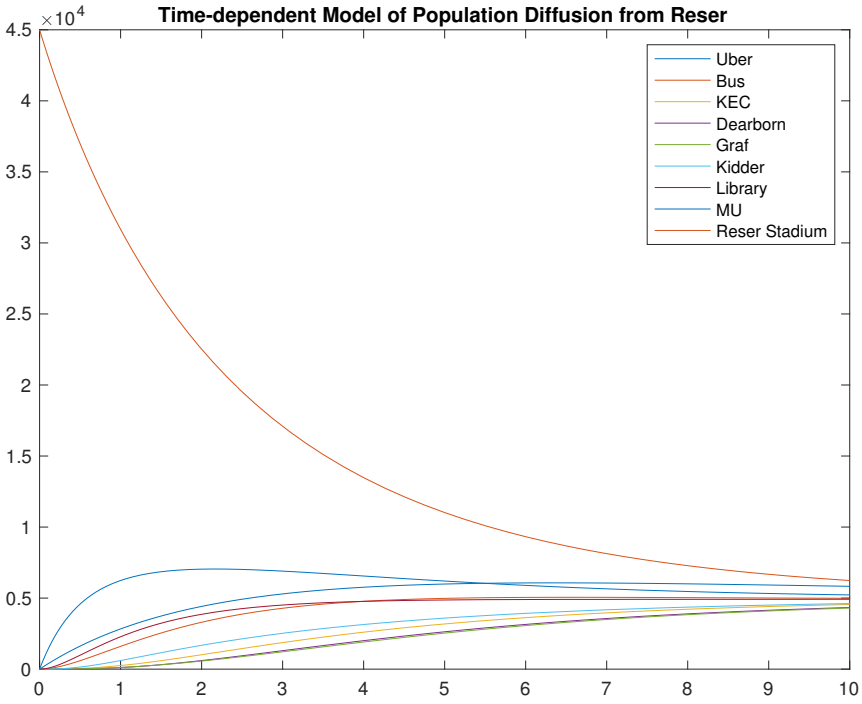

FIGURE 5. Plot of population over time for each location in the dynamic model

It produces the following figure.

The following MATLAB code generates a movie that shows how a scatterplot of population for each location changes as time increases.

```
timesteps = 125;
v = VideoWriter('populationDrain');
open(v);
for j = 1:timesteps
    plot(G,'XData',x_graph,'YData',y_graph,'EdgeLabel',G.Edges.Weight)
    scatter(x_graph',y_graph',x_t(j,:)+100,x_t(j,:),'filled');
    frame = getframe(gcf);
    writeVideo(v,frame);
end
close(v);
```

The movie is uploaded to CANVAS. Note that in the above solution the library is not fixed at 100.