M647 Spring 2012 Practice Problems for Final Exam

The final for M647 will be Tuesday May 8, 10:30 a.m. - 12:30 p.m. in Blocker 122 (the usual classroom). The exam will consist of two parts: Part 1 will not require MATLAB, while Part 2 will require MATLAB. Students will have to turn in Part 1 before starting Part 2, but for Part 2 students will have access to all M-files we've used this semester, from both lecture and homework. Students will be expected to access data files from the course web site.

The final will cover material from the second half of the semester: modeling with ODE, including compartment models, chemical reactions, populations dynamics, classical mechanics (Newtonian, Lagrangian, Hamilitonian), solving ODE in MATLAB (including parameter estimation and confidence intervals), non-dimensionalization, modeling with PDE in one space dimension, and solving PDE in MATLAB.

My office hours during finals will be 2:00-3:00 Wednesday, Thursday, Friday, and Monday.

Part One Problems

1. Consider a community of E. Coli which through mutation has divided itself into three general populations:

C: (*Colicinogenic* cells) Produce (and release) a toxin, colicin, which destroys E. Coli cells, and also produce a colicin-specific immunity protein, which renders the cell immune to the colicin.

R: (*Resistant* cells) Produce the colicin-specific immunity protein, which renders them immune to the colicin, but do not produce the colicin.

S: (*Sensitive* cells) Produce neither the immunity protein nor the colicin.

The production of either the toxin colicin or the colicin specific immunity protein requires a certain amount of effort, and the growth rate of S cells is greater than the growth rate of R cells, which in turn is greater than the growth rate of C cells. Consequently, S can displace R by outproducing it, R can displace C by outproducing it, and C can displace S by killing it with the toxin. The situation has been likened to the game rock–paper–scissors, for which rock crushes scissors, scissors cut paper, and paper covers rock. Write down a system of ODE that models the populations C, R, and S. Your model will involve a number of parameters that you would generally estimate from data. In lieu of this, discuss the relative values you would expect. For example, if you have growth constants r_1, r_2, and r_3, which would you expect to be largest?

2. Consider a simple pendulum as depicted in Figure 1 moving under the influence of gravity and air resistance. Explain why Lagrangian and Hamiltonian mechanics provide inappropriate frameworks for modeling this situation, and use Newtonian mechanics to write down an ODE describing the motion.

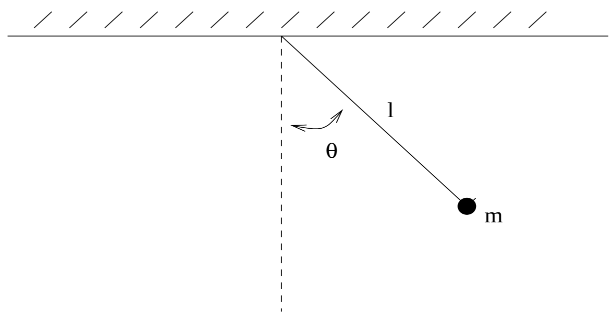

Figure 1: Figure for Problem 2.

3. Consider a pendulum of length l attached to a pivot that rotates with angular velocity ω along an upright circular frame with radius r (see Figure 2). Write down the Lagrangian for this system and also the EulerLagrange equations. Ignore friction and air resistance.

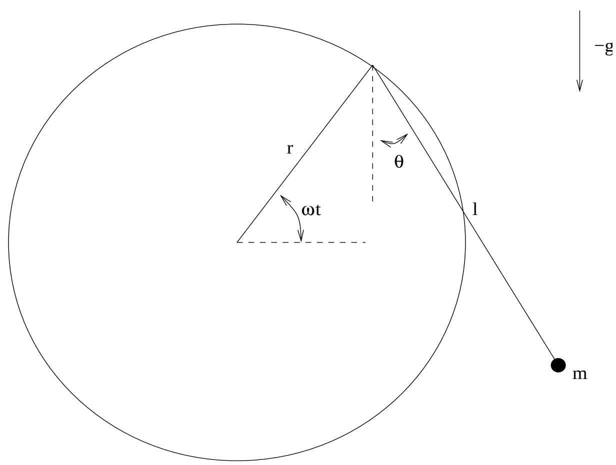

Figure 2: Figure for Problem 3.

4. Consider the mass-spring system depicted in Figure 3. Ignoring friction, write down the Hamiltonian for this system, and the associated Hamilton ODE system.

5. Non-dimensionalize the mutualism model

$$\begin{aligned}\frac{dy_1}{dt} &= r_1 y_1 (1 - \frac{y_1}{K_1}) + b_1 y_1 y_2 \\ \frac{dy_2}{dt} &= r_2 y_2 (1 - \frac{y_2}{K_2}) + b_2 y_1 y_2.\end{aligned}$$

6. A string of length L with constant density ρ is hanging from one end, under the influence of gravity (see Figure 4). Let the x-direction be downward from the string's top end, positioned at $x = 0$, and let $u(x, t)$ denote horizontal displacement from the vertical. Assuming small oscillations, find a PDE for u. Suggest appropriate initial and boundary conditions.

Note. Assume that the force of gravity at each point of the string equals the weight of the part of the string below that point, and is directed tangentially.

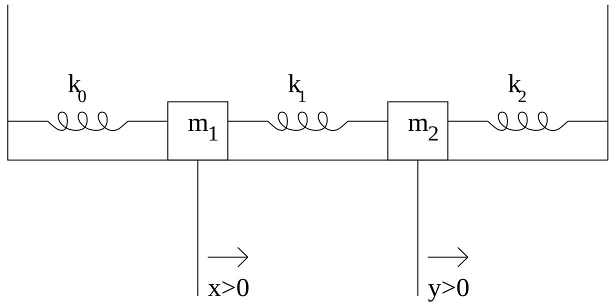

Figure 3: Figure for Problem 4.

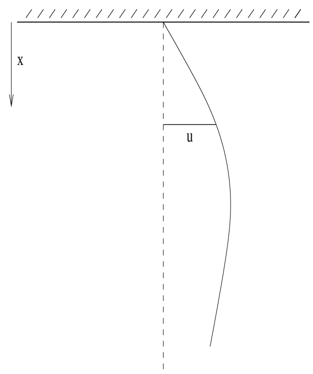

Figure 4: Vertical string.

7. Consider a drum head stretched with tension $T(x, y, t)$, in units force per unit length (i.e., if $\vec{F}$ is force due to tension, $T = |\vec{F}|$), and with mass density $\rho(x, y)$ with units mass per unit area. Assuming small displacement, derive the two-dimensional wave equation,

$$\rho(x, y)u_{tt} = \Big((T(x, y, t)u_x)_x + (T(x, y, t)u_y)_y\Big).$$

Note. Consider a small strip of membrane with sidelengths (approximately) Δx and Δy.

8. Derive the wave equation for two space dimensions using Lagrangian mechanics. Use a square membrane $[0, L] \times [0, M]$, and assume the potential energy is proportional to the deformation for equilibrium, measured by membrane area:

$$P = k\Big(\int_0^L \int_0^M \sqrt{1 + u_x^2 + u_y^2}dxdy - LM\Big).$$

9a. The Navier-Stokes momentum equation is often written in the form

$$(\rho v)_t + (\rho v^2 + p)_x = \mu v_{xx} + f.$$

Show that this is equivalent to our form from class.

9b. Recall from our derivation in class of the Navier-Stokes momentum equation that if the fluid is incompressible then $\Delta x(t)$ is constant in t. Show that in this case we must have

$$v_x = 0.$$

9c. Euler's equations of gas dynamics comprise a system of three equations that are immediate from the continuity equation, the Navier-Stokes momentum equation, and the Navier-Stokes energy equation. In particular, these are the equations you obtain if you ignore effects due to viscosity, body forces, and temperature. Recalling from Assignment 10 that the Navier-Stokes energy equation is

$$\Big[\rho(\frac{v^2}{2}+e)\Big]_t + \Big[\rho v e + \frac{1}{2}\rho v^3 - \kappa(x)T_x + pv - \mu v v_x\Big]_x = 0, \tag{1}$$

write down Euler's equations.

Part Two Problems

1. The Lotka-Volterra model with prey carrying capacity is

$$\begin{aligned}\frac{dy_1}{dt} &= ay_1(1-\frac{y_1}{K}) - by_1y_2 \\ \frac{dy_2}{dt} &= -ry_2 + cy_1y_2.\end{aligned}$$

In this problem, we'll find values for a, b, c, r, and K using the data in *lvdata.m* (available on the course web site).

1a. Use the method of derivative approximation to obtain initial approximations for a, b, c, r, and K. Plot solutions using these parameter estimates along with the data. What does this result suggest about including K in this model?

1b. Part (a) suggests that we cannot obtain a reasonable value for K by derivative approximation, but we can view the maximum observed prey population as a reasonable guess. Repeat Part (a) with this value of K to get corresponding first approximations for a, b, c, and r. Again, plot solutions using these parameters along with the data.

1c. Use nonlinear regression to refine your estimates from Part (b), and plot solutions to your refined model along with the data. When we carried out this fit in class without K, we found $s = 4.5519$. Discuss whether or not including K is justified. Carry this out both with and without scaling.

1d. Find 95% confidence intervals for you values from Part (c). Again, what does your calculation suggest about including K in this model. Carry this out both with and without scaling.

Note. The analysis of this problem doesn't go so smoothly, which is why it's better as a practice problem than an exam problem. Nonetheless, it gives you practice on all the right things.

2. According to Newton's law of cooling, the flux of temperature across a surface is proportional to the temperature difference across the surface. For example, if the temperature at the left endpoint of a heat-conducting cylinder is $u(0,t)$ and the temperature outside the cylinder is T_0, then we expect to have the relation

$$f = -h(u(0,t) - T_0),$$

where the constant h is referred to as the heat transfer coefficient. (Notice that if the temperature inside is greater then the flux will be to the left.) Since the flux can be expressed as $f = -k(x)u_x$, this gives a boundary condition

$$-k(0)u_x(0,t) = -h(u(0,t) - T_0).$$

Assume that for a particular heat-conducting cylinder we have cylinder length $L = 1$ m, thermal diffusivity $k(x) \equiv .1 \ m^2 s^{-1}$, ambient temperature $T_0 = 25^o$ C, and heat transfer coefficient $h = .03 \ m^{-2}s^{-1}$. Suppose the bar is initially cooled uniformly to 0^o C, and that the sides are insulated to that heat only enters or escapes at the ends. Plot a temperature profile after 10 seconds, and also create a mesh plot of your solution for all $t \in [0, 10]$.

3. One general three-species competition model can be written as

$$\begin{aligned}
u_t &= r_1 u(1 - \frac{u + s_1 v + s_2 w}{K_1}) + (b_{11}(x)u_x)_x + (b_{12}(x)v_x)_x + (b_{13}(x)w_x)_x \\
v_t &= r_2 v(1 - \frac{s_3 u + v + s_4 w}{K_2}) + (b_{21}(x)u_x)_x + (b_{22}(x)v_x)_x + (b_{23}(x)w_x)_x \\
w_t &= r_3 w(1 - \frac{s_5 u + s_6 v + w}{K_3}) + (b_{31}(x)u_x)_x + (b_{32}(x)v_x)_x + (b_{33}(x)w_x)_x.
\end{aligned}$$

Solve this system in MATLAB with parameter values $r_1 = .02$, $r_2 = .03$, $r_3 = .05$, $s_1 = 2$, $s_2 = 2$, $s_3 = \frac{1}{2}$, $s_4 = 1$, $s_5 = \frac{1}{2}$, $s_6 = 1$, $K_1 = 20$, $K_2 = 10$, $K_3 = 10$, and

$$B = \begin{pmatrix} 1 & 2 & 1 \\ 5 & 3 & 2 \\ 10 & .8 & 30 \end{pmatrix} \times 10^{-5}.$$

Use no-flux boundary conditions and the initial conditions

$$\begin{aligned}
u(x,0) &= 20(1 + \cos(\pi x)) \\
v(x,0) &= 10(1 + \sin(2\pi x - \frac{\pi}{2})) \\
w(x,0) &= 10(1 + \sin(\pi x - \frac{\pi}{2})).
\end{aligned}$$

Plot a stacked plot of solution profiles for $t = 25$.

Part One Solutions

1. The key to this problem is to recognize that the dynamic is primarily competitive. Set

$$\begin{aligned}
C(t) &= \text{number of colicinogenic cells} \\
R(t) &= \text{number of resistant cells} \\
S(t) &= \text{number of sensitive cells.}
\end{aligned}$$

Then

$$\begin{aligned}\frac{dC}{dt} &= r_1C(1-\frac{C+s_1R+s_2S}{K_1})\\ \frac{dR}{dt} &= r_2R(1-\frac{s_3C+R+s_4S}{K_2})\\ \frac{dS}{dt} &= r_3S(1-\frac{s_5C+s_6R+S}{K_3}).\end{aligned}$$

Here $r_3 > r_2 > r_1$, and we observe that the predation term we expect for the S equation has been subsumed into the expression

$$-r_3S\frac{s_5C}{K_3},$$

altering only the value of the parameter s_5. Since the C-cells produce both the toxin and the immunity protein we might expect them to use more environment than the R and S-cells (effectively, they are converting nutrients into toxins and proteins). I.e., $s_5 > s_6$, $s_3 > 1$, and $s_1 < 1$. Since the R-cells produce the immunity protein, and the S-cells do not, we might expect the R-cells to use more environment than the S-cells. I.e., $s_6 > 1$, $s_4 < 1$, and $s_1 > s_2$.

Also, note that it might not be unreasonable to assume the environment is the same for each cell, so that $K_1 = K_2 = K_3 = K$. (This assumes, for example, that the C-cells don't require a nutrient in addition to the nutrients required by the R and S-cells.) In this case, the numerator for each carrying capacity would be the same, in principle at least, because each cell would use up the same proportion of the shared carrying capacity.

2. Lagrangian and Hamiltonian mechanics are inappropriate because energy is lost to the surrounding air.

The force due to gravity on m acts vertically downward, and must be decomposed into a force $-T$, which is exactly balanced by the rod, and a force F, directed tangentially to the arc of motion (see Figure 5). Observing the right triangle, with hypotenuse of length $-mg$, we have

$$\begin{aligned}\cos\theta &= \frac{T}{mg} \Rightarrow T = mg\cos\theta,\\ \sin\theta &= -\frac{F}{mg} \Rightarrow F = -mg\sin\theta.\end{aligned}$$

We know from our discussion of dimensional analysis that the force due to air resistance is proportional to velocity squared. Measuring distance as arclength, $d = l\theta$, we see that the pendulum's velocity along its arc is $v = l\frac{d\theta}{dt}$, so

$$F_{air} = -k\rho Sl^2\frac{d\theta}{dt}|\frac{d\theta}{dt}|,$$

where the absolute values are taken to get the direction correct. Newton's second law of motion ($F = ma$) becomes (noting $a = l\frac{d^2\theta}{dt^2}$)

$$ml\frac{d^2\theta}{dt^2} = -\,mg\sin\theta - k\rho Sl^2\frac{d\theta}{dt}|\frac{d\theta}{dt}|,$$

which can be written as

$$\frac{d^2\theta}{dt^2} = -\frac{g}{l}\sin\theta - \frac{k\rho Sl}{m}\frac{d\theta}{dt}|\frac{d\theta}{dt}|.$$

It's customary to write

$$b = \frac{k\rho Sl}{m}.$$

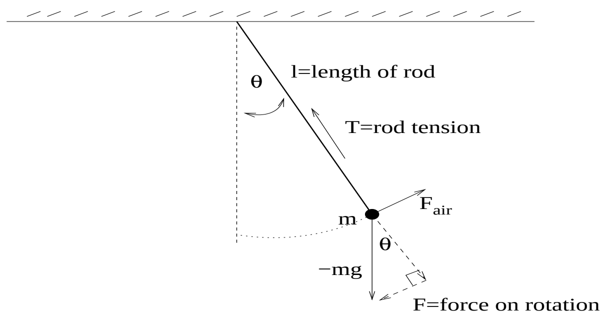

Figure 5: Pendulum motion under the influence of gravity alone.

3. In cartesian coordinates, the position of the mass is (measured from the center of the circle)

$$\begin{aligned} x =& r\cos\omega t + l\sin\theta \\ y =& -l\cos\theta + r\sin\omega t, \end{aligned}$$

and so the kinetic energy of the mass is

$$\begin{aligned} K =& \frac{1}{2}m(\frac{dx}{dt}^2 + \frac{dy}{dt}^2) = \frac{1}{2}m\Big((-r\omega\sin\omega t + l\theta'\cos\theta)^2 + (l\theta'\sin\theta + r\omega\cos\omega t)^2\Big) \\ =& \frac{1}{2}m\Big(r^2\omega^2 + l^2\theta'^2 + 2lr\theta'\omega\sin(\theta - \omega t)\Big). \end{aligned}$$

If we take the bottom of the circle to be our reference (baseline) height for potential energy, we have

$$P = mg(r + y) = mg(r - l\cos\theta + r\sin\omega t).$$

In this case, there is only one generalized coordinate $q = \theta$. In terms of q, the Lagrangian is

$$L(q, q') = \frac{1}{2}m\Big(r^2\omega^2 + l^2q'^2 + 2lrq'\omega\sin(q - \omega t)\Big) - mg(r - l\cos q + r\sin\omega t).$$

The Euler-Lagrange equation in this case is

$$\frac{d}{dt}\frac{\partial L}{\partial q'} - \frac{\partial L}{\partial q} = 0,$$

and we have

$$\begin{aligned} \frac{\partial L}{\partial q} =& mlrq'\omega\cos(q - \omega t) - mgl\sin q \\ \frac{\partial L}{\partial q'} =& ml^2q' + mlr\omega\sin(q - \omega t). \end{aligned}$$

The Euler-Lagrange equation becomes

$$ml^2q'' + mlr\omega\cos(q - \omega t)(q' - \omega) = mlrq'\omega\cos(q - \omega t) - mgl\sin q,$$

which simplifies to

$$q'' - \frac{r\omega^2}{l}\cos(q - \omega t) + \frac{g}{l}\sin q = 0.$$

Clearly, if either r or ω is 0 we obtain the usual equation for an undamped pendulum.

4. In this case the kinetic energy is particularly easy,

$$K = \frac{1}{2}m_1x'^2 + \frac{1}{2}m_2y'^2,$$

and the potential energy is

$$P = \frac{1}{2}k_0x^2 + \frac{1}{2}k_1(y - x)^2 + \frac{1}{2}k_2y^2,$$

so the Lagrangian is

$$L = \frac{1}{2}m_1x'^2 + \frac{1}{2}m_2y'^2 - \frac{1}{2}k_0x^2 - \frac{1}{2}k_1(y - x)^2 - \frac{1}{2}k_2y^2.$$

In order to put this in the context of our notation from class, we'll use the generalized coordinates $(q_1, q_2) = (x, y)$, though this certainly isn't necessary. We have

$$L = \frac{1}{2}m_1q_1'^2 + \frac{1}{2}m_2q_2'^2 - \frac{1}{2}k_0q_1^2 - \frac{1}{2}k_1(q_2 - q_1)^2 - \frac{1}{2}k_2q_2^2.$$

The generalized momentum is

$$\vec{p} = D_{q'}L = (m_1q_1', m_2q_2')$$

(i.e., classical momentum in this case), and we easily solve for q_1' and q_2' in terms of p_1 and p_2 as

$$\begin{aligned} q_1' &= \frac{p_1}{m_1} \\ q_2' &= \frac{p_2}{m_2}. \end{aligned}$$

Note particularly that this provides us with equations for q_1 and q_2, so we only need to find equations for p_1 and p_2. For this, we write our Hamiltonian as

$$\begin{aligned} H &= \vec{p} \cdot \vec{q}' - L \\ &= \frac{p_1^2}{m_1} + \frac{p_2^2}{m_2} - \frac{1}{2}m_1\frac{p_1^2}{m_1^2} - \frac{1}{2}m_2\frac{p_2^2}{m_2^2} + \frac{1}{2}k_0q_1^2 + \frac{1}{2}k_1(q_2 - q_1)^2 + \frac{1}{2}k_2q_2^2 \\ &= \frac{1}{2}\frac{p_1^2}{m_1} + \frac{1}{2}\frac{p_2^2}{m_2} + \frac{1}{2}k_0q_1^2 + \frac{1}{2}k_1(q_2 - q_1)^2 + \frac{1}{2}k_2q_2^2. \end{aligned}$$

(Notice that we could have written this directly as total energy for the system, but it would have missed the point of the material we're reviewing.) Finally, $\vec{p}' = -D_q H$,

and

$$D_q H = (k_0 q_1 - k_1(q_2 - q_1), k_1(q_2 - q_1) + k_2 q_2).$$

This completes our system as

$$\begin{aligned}
\frac{dq_1}{dt} &= \frac{p_1}{m_1} \\
\frac{dq_2}{dt} &= \frac{p_2}{m_2} \\
\frac{dp_1}{dt} &= -k_0 q_1 + k_1(q_2 - q_1) \\
\frac{dp_2}{dt} &= -k_1(q_2 - q_1) - k_2 q_2.
\end{aligned}$$

5. Set

$$\tau = \frac{t}{A}; \quad Y_1(\tau) = \frac{y_1(t)}{C}; \quad Y_2(\tau) = \frac{y_2(t)}{D},$$

where A denotes a constant with dimension T, while C and D denote constants with dimension B (biomass). Our system becomes

$$\begin{aligned}
\frac{dY_1}{d\tau} &= r_1 A Y_1 (1 - \frac{CY_1}{K_1}) + b_1 A D Y_1 Y_2 \\
\frac{dY_2}{d\tau} &= r_2 A Y_2 (1 - \frac{DY_2}{K_2}) + b_2 A C Y_1 Y_2.
\end{aligned}$$

One natural choice is $C = K_1$, $D = K_2$, and $A = 1/r_1$. This gives

$$\begin{aligned}
\frac{dY_1}{d\tau} &= Y_1(1 - Y_1) + \frac{b_1 K_2}{r_1} Y_1 Y_2 \\
\frac{dY_2}{d\tau} &= \frac{r_2}{r_1} Y_2 (1 - Y_2) + \frac{b_2 K_1}{r_1} Y_1 Y_2.
\end{aligned}$$

Generally, this would be written as a three parameter system, for example with $\alpha = \frac{b_1 K_2}{r_1}$, $\beta = \frac{r_2}{r_1}$, and $\gamma = \frac{b_2 K_1}{r_1}$.

6. As in our derivation of the 1-d wave equation in class, we'll apply Newton's second law (this time in the horizontal direction) to a strip of string between x and $x + \Delta x$. First,

$$ma \approx \rho \Delta x u_{tt}.$$

The force at position $x + \Delta x$ in the horizontal direction is

$$F_{x+\Delta x} \approx \rho(L - (x + \Delta x)) g \sin\theta(x + \Delta x, t),$$

where we are assuming that for small displacement the string is almost straight (so we can ignore an arclength integral). Here, θ is the angle between the vertical and the tangent. At position x, the force downward is

$$F_x \approx \rho(L - x) g \sin\theta(x, t),$$

but this is interior to the part of the string we're working with, and we need to consider instead the opposite force caused by the string's being attached at the top (i.e., the force that keeps the string from falling). This is just the negative of F_x. In total,

$$F = F_{x+\Delta x} - F_x = \rho(L - (x + \Delta x))g \sin\theta(x + \Delta x, t) - \rho(L - x)g \sin\theta(x, t).$$

Newton's second law becomes

$$\rho\Delta x u_{tt} \approx \rho(L - (x + \Delta x))g \sin\theta(x + \Delta x, t) - \rho(L - x)g \sin\theta(x, t).$$

Upon dividing by Δx and taking $\Delta x \to 0$, we obtain

$$u_{tt} = g\Big((L - x)\sin\theta\Big)_x.$$

Finally, using the approximation from class,

$$\sin\theta \approx \tan\theta = u_x,$$

we obtain

$$u_{tt} = g\Big((L - x)u_x\Big)_x.$$

In this case, we require two initial conditions and two boundary conditions. For example, the initial conditions would generally by $u(x, 0) = u_0(x)$ and $u_t(x, 0) = w_0(x)$, and one of the boundary conditions would be $u(0, t) = 0$ for $t \geq 0$, since the top location of the string is fixed. The second boundary condition is less clear, but the most natural one is probably $u_x(L, t) = 0$ for $t \geq 0$. This says that the bottom of the string be pointing straight downward.

7. We apply Newton's second law in the vertical direction to the suggested strip, noting first that

$$ma \approx \rho(x, y)\Delta x\Delta y u_{tt}.$$

The force due to tension in the vertical direction along the edge from (x, y) to $(x, y + \Delta y)$ is

$$\int_y^{y+\Delta y} T(x, z, t)\sin\theta(x, z, t)dz \approx T(x, y, t)\sin\theta(x, y, t),$$

while the force due to tension in the vertical direction along the opposite edge from $(x+\Delta x, y)$ to $(x + \Delta x, y + \Delta y)$ is approximately

$$T(x + \Delta x, y, t)\sin\theta(x + \Delta x, y, t).$$

Proceeding likewise on the remaining two sides, we find that the total force is

$$\begin{aligned} F \approx\, & T(x + \Delta x, y, t)\sin\theta_1(x + \Delta x, y, t) - T(x, y, t)\sin\theta_1(x, y, t) \\ & + T(x, y + \Delta y, t)\sin\theta_2(x, y + \Delta y, t) - T(x, y, t)\sin\theta_2(x, y, t) \\ & \approx (T\sin\theta_1)_x + (T\sin\theta_2)_y. \end{aligned}$$

Here θ_1 denotes angle from the horizontal in the x direction and θ_2 denotes angle from the horizontal in the y direction, so that

$$\sin\theta_1(x,y,t) \approx \tan\theta_1(x,y,t) = u_x(x,y,t)$$
$$\sin\theta_2(x,y,t) \approx \tan\theta_2(x,y,t) = u_y(x,y,t).$$

This concludes the derivation.

8. The kinetic energy is

$$K = \int_0^L \int_0^M \frac{1}{2}\rho(x,y)u_t^2 dxdy,$$

so the Lagrangian is

$$L = \int_0^L \int_0^M \frac{1}{2}\rho(x,y)u_t^2 - k\Big(\sqrt{1+u_x^2+u_y^2} - 1\Big) dxdy.$$

The action is

$$A[u] = \int_0^T \int_0^L \int_0^M \frac{1}{2}\rho(x,y)u_t^2 - k\Big(\sqrt{1+u_x^2+u_y^2} - 1\Big) dxdydt.$$

If we denote our domain

$$\Omega = [0,L]\times[0,M]\times[0,T],$$

then it's reasonable to assume A is defined on

$$\mathcal{S} := \{u \in C^2(\Omega) : u|_{\partial\Omega} = u_b = \text{specified}\}.$$

According to Hamilton's Principle, we should have

$$A'[u] = 0.$$

We compute this with the Gâteaux derivative

$$A'[u] = \lim_{\tau\to 0} \frac{A[u+\tau h] - A[u]}{\tau},$$

for any h in the function space

$$\mathcal{S}_0 = \{u \in C^2(\Omega) : u|_{\partial\Omega} = 0\}.$$

We separate out

$$\begin{aligned}
A[u+\tau h] &= \int_0^T \int_0^L \int_0^M \frac{1}{2}\rho(x,y)(u_t+\tau h_t)^2 - k\Big(\sqrt{1+(u_x+\tau h_x)^2+(u_y+\tau h_y)^2} - 1\Big) dxdydt \\
&= \int_0^T \int_0^L \int_0^M \frac{1}{2}\rho(x,y)u_t^2 + \rho(x,y)\tau u_t h_t + \frac{1}{2}\rho(x,y)\tau^2 h_t^2 dxdydt \\
&\quad - k\int_0^T \int_0^L \int_0^M \Big(\sqrt{1+u_x^2+u_y^2} - 1\Big) + \frac{u_x h_x + u_y h_y}{\sqrt{1+u_x^2+u_y^2}} + o(\tau) dxdydt.
\end{aligned}$$

We see that

$$A'[u] = \int_0^T \int_0^L \int_0^M \rho(x,y)u_t h_t - k\frac{u_x h_x + u_y h_y}{\sqrt{1+u_x^2+u_y^2}} dxdydt.$$

Upon integrating the first integrand by parts in t, the second by parts in x, and the third by parts in y, we find

$$\int_0^T \int_0^L \int_0^M \Big[-\rho(x,y)u_{tt} + k\Big(\frac{u_x}{\sqrt{1+u_x^2+u_y^2}}\Big)_x + k\Big(\frac{u_y}{\sqrt{1+u_x^2+u_y^2}}\Big)_y \Big] h dxdydt = 0.$$

As usual, our freedom to choose h ensures that its multiplier in the integrand must be 0. I.e.,

$$-\rho(x,y)u_{tt} + k\Big(\frac{u_x}{\sqrt{1+u_x^2+u_y^2}}\Big)_x + k\Big(\frac{u_y}{\sqrt{1+u_x^2+u_y^2}}\Big)_y = 0,$$

which becomes the wave equation when we assume u_x and u_y are both small and take $\rho(x,y)$ constant.

9. For Part (a), we have

$$\rho_t v + \rho v_t + \rho_x v^2 + 2\rho v v_x = -p_x + \mu v_{xx} + f.$$

We use the continuity equation

$$\rho_t + (v\rho)_x = 0 \Rightarrow \rho_t = -v_x\rho - v\rho_x,$$

so that

$$-\rho v v_x - \rho_x v^2 + \rho v_t + \rho_x v^2 + 2\rho v v_x = -p_x + \mu v_{xx} + f,$$

which is

$$\rho(v_t + v v_x) = \mu v_{xx} - p_x + f,$$

our form from class.

For Part (b), we note that

$$\frac{d}{dt}x(t) = \frac{d}{dt}(x(t) + \Delta x),$$

which is equivalent to

$$v(x(t),t) = v(x(t)+\Delta x, t).$$

Thus,

$$0 = v(x(t)+\Delta x, t) - v(x(t),t) \approx v_x(x(t),t)\Delta x \Rightarrow v_x(x,t) = 0.$$

For Part (c) we have

$$\begin{aligned} \rho_t + (v\rho)_x &= 0 \\ \rho(v_t + v v_x) &= -p_x \\ \Big[\rho(\frac{v^2}{2} + e)\Big]_t + \Big[\rho v(e + \frac{1}{2}v^2 + \frac{p}{\rho})\Big]_x &= 0. \end{aligned}$$

Part Two Solutions

1. First, we express this equation in the linear form

$$\frac{1}{y_1}\frac{dy_1}{dt} = a - \frac{a}{K}y_1 - by_2$$
$$\frac{1}{y_2}\frac{dy_2}{dt} = -r + cy_1.$$

For the first equation, we must fit $Z := \frac{1}{y_1}\frac{dy_1}{dt}$ as a function of two variables, $X = y_1$ and $Y = y_2$. (The second can be analyzed precisely as in class.) We use the MATLAB M-file *lvklinearfit1.m*.

```
%LVKLINEARFIT1: MATLAB script M-file to carry out linear parameter
%estimation for the Lotka-Volterra model with prey carrying capacity
%for the Hudson Bay data
%
lvdata;
%Prey equation
Z = (H(3:21)-H(1:19))./(2*H(2:20)); %Central difference derivative approxi-
mation
X = H(2:20);
Y = L(2:20);
M = [ones(size(Z))' X' Y'];
p = M\Z';
a = p(1)
K = -a/p(2)
b=-p(3)
%Predator equation
Y2 = (L(3:21)-L(1:19))./(2*L(2:20));
p2 = polyfit(X,Y2,1);
c = p2(1)
r = -p2(2)
%
%Plot the result
lvkrhs = @(t,y) [a*y(1)*(1-y(1)/K)-b*y(1)*y(2);-r*y(2)+c*y(1)*y(2)];
[t,y]=ode45(lvkrhs,[0,20],[30.0; 4.0]);
subplot(2,1,1)
plot(t,y(:,1),years,H,'o')
title('Prey population, model and data','FontSize',14)
subplot(2,1,2)
plot(t,y(:,2),years,L,'o')
title('Predator population, model and data','FontSize',14)
```

We find the following values:

$$\begin{aligned}
a &= .4007 \\
b &= .0241 \\
c &= .0234 \\
r &= .7647 \\
K &= -184.8479.
\end{aligned}$$

Clearly, the value for K is unphysical, though a plot of this model suggests our calculations were correct (i.e., the result is not entirely unreasonable). (See Figure 6.)

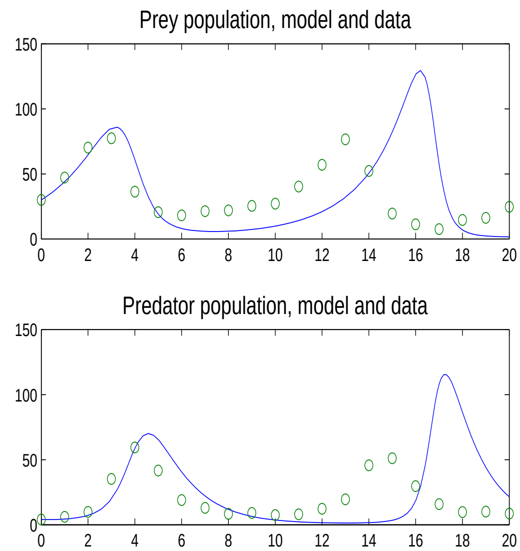

Figure 6: First linear fit.

For Part (b), we set $K := 77.40$ (i.e., the maximum of the data H). In this case, we use the linear form

$$\begin{aligned}
\frac{1}{y_1}\frac{dy_1}{dt} &= a(1 - \frac{y_1}{K}) - by_2 \\
\frac{1}{y_2}\frac{dy_2}{dt} &= -r + cy_1.
\end{aligned}$$

In this case, we consider $Z = \frac{1}{y_1}\frac{dy_1}{dt}$ to be a function of $X = (1 - \frac{y_1}{K})$ and $Y = y_2$. We carry out this fit with *lvklinearfit2.m*.

```
%LVKLINEARFIT2: MATLAB script M-file to carry out linear parameter
%estimation for the Lotka-Volterra model with prey carrying capacity
%for the Hudson Bay data. In this case, we set K to be the
%maximum observed prey population.
%
```

```
lvdata;
K = max(H)
%Prey equation
Z = (H(3:21)-H(1:19))./(2*H(2:20)); %Central difference derivative approxi-
mation
X = 1-H(2:20)/K;
Y = L(2:20);
M = [X' Y'];
p = M\Z';
a = p(1)
b = -p(2)
%Predator equation
Y2 = (L(3:21)-L(1:19))./(2*L(2:20));
X2 = H(2:20);
p2 = polyfit(X2,Y2,1);
c = p2(1)
r = -p2(2)
%
%Plot the result
lvkrhs = @(t,y) [a*y(1)*(1-y(1)/K)-b*y(1)*y(2);-r*y(2)+c*y(1)*y(2)];
[t,y]=ode45(lvkrhs,[0,20],[30.0; 4.0]);
subplot(2,1,1)
plot(t,y(:,1),years,H,'o')
title('Prey population, model and data','FontSize',14)
subplot(2,1,2)
plot(t,y(:,2),years,L,'o')
title('Predator population, model and data','FontSize',14)
```

In this case, we find

$$\begin{aligned} a &= .4302 \\ b &= .0169 \\ c &= .0234 \\ r &= .7646 \\ K &= 77.4. \end{aligned}$$

The solution using these values is plotted against the data in Figure 7.

For Part (c), the nonlinear fit is carried out in *lvknonlinearfit.m.*

```
function lvknonlinearfit
%LVKNONLINEARFIT: MATLAB function M-file that takes an initial
%approximation of parameter values and carries out nonlinear
%regression to obtain best-fit parameter values for the Lotka-Volterra
%system with prey carrying capacity for the Hudson Bay data.
```

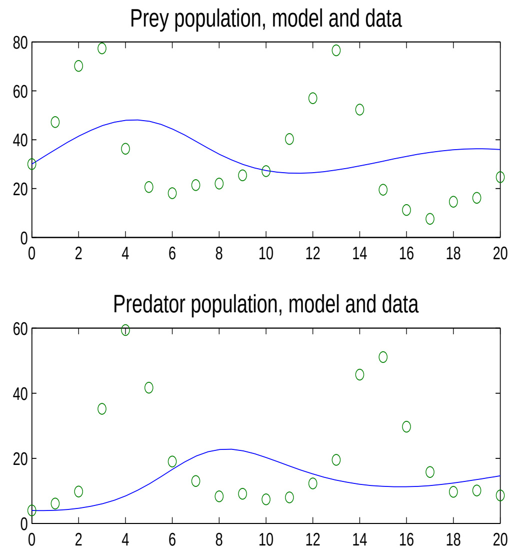

Figure 7: Second linear fit.

```
global years L H;
lvdata
guess = [.4302; .0169; .0234; .7646; max(H)]; %Order: a, b, c, r, K
options = optimset('MaxFunEvals',1e4);
[p,error]=fminsearch(@lverr, guess, options);
a = p(1)
b = p(2)
c = p(3)
r = p(4)
K = p(5)
s = sqrt(error/(2*(length(H)-1)-length(p)))
%
[t,y]=ode45(@lvrhs,[0,20],[H(1); L(1)],[],p);
subplot(2,1,1)
plot(t,y(:,1),years,H,'o')
title('Prey population, model and data','FontSize',14)
subplot(2,1,2)
plot(t,y(:,2),years,L,'o')
title('Predator population, model and data','FontSize',14)
%
function error = lverr(p)
%LVERR: Function defining error function for
%example with Lotka-Volterra equations.
global years L H;
[t,y] = ode45(@lvrhs,years,[H(1);L(1)],[],p); %Notice that we pass
%a parameter vector
```

```
error = norm(y(:,1)-H')^2+norm(y(:,2)-L')^2;
%
function value = lvrhs(t,y,p)
%LVRHS: ODE for example Lotka-Volterra paramter
%estimation example. p(1)=a, p(2) = b, p(3) = c, p(4) = r, p(5)=K
value=[p(1)*y(1)*(1-y(1)/p(5))-p(2)*y(1)*y(2);-p(4)*y(2)+p(3)*y(1)*y(2)];
```

We find values

$$\begin{aligned} a &= .6210 \\ b &= .0288 \\ c &= .0245 \\ r &= .7650 \\ K &= 724.0122, \end{aligned}$$

which give a reasonable fit to the data (see Figure 8), but we find $s = 4.8249$, which is worse than the value we obtained without a carrying capacity. This still suggests we are not justified in including a carrying capacity with this model.

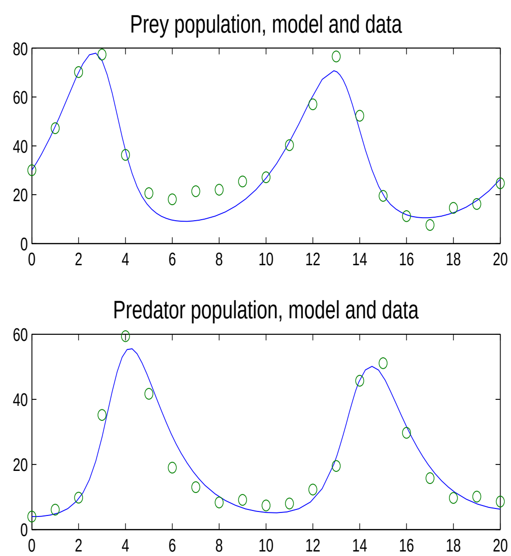

Figure 8: Nonlinear fit.

For Part (d) we proceed in our usual way to find confidence intervals with *lvkconf.m*.

```
function lvkconf
%LVKCONF: MATLAB function M-file for computing 95% confidence
%intervals for parameter estimates of the Lotka-Volterra
%model fit to the Hudson Bay data.
%Parameter values
```

```
%Scaling is not used
lvdata;
y0 = [30.0;4.0];
pbar = [.6210 .0288 .0245 .7650 724.0122];
[t ybar]=ode45(@lvrhs,years,y0,[],pbar);
dely = [H'; L'] - [ybar(:,1); ybar(:,2)];
%Set increment for derivative approximations
h = 1e-6;
[t ybar1]=ode45(@lvrhs,years,y0,[],[pbar(1)+h pbar(2) pbar(3) pbar(4) pbar(5)]);
[t ybar2]=ode45(@lvrhs,years,y0,[],[pbar(1) pbar(2)+h pbar(3) pbar(4) pbar(5)]);
[t ybar3]=ode45(@lvrhs,years,y0,[],[pbar(1) pbar(2) pbar(3)+h pbar(4) pbar(5)]);
[t ybar4]=ode45(@lvrhs,years,y0,[],[pbar(1) pbar(2) pbar(3) pbar(4)+h pbar(5)]);
[t ybar5]=ode45(@lvrhs,years,y0,[],[pbar(1) pbar(2) pbar(3) pbar(4) pbar(5)+h]);
F = [ybar1(:,1)-ybar(:,1) ybar2(:,1)-ybar(:,1) ybar3(:,1)-ybar(:,1) ybar4(:,1)-
ybar(:,1) ybar5(:,1)-ybar(:,1); ...
ybar1(:,2)-ybar(:,2) ybar2(:,2)-ybar(:,2) ybar3(:,2)-ybar(:,2) ybar4(:,2)-ybar(:,2)
ybar5(:,2)-ybar(:,2)]/h;
delp = F\dely
pnew = pbar'+delp
%For 95% confidence interval
q = 2*(length(H)-1)-length(pbar);
l = stut(q,.95);
V = inv(F'*F);
ssq = (norm(dely-F*delp)^2)/q;
s = sqrt(ssq)
error = l*sqrt(ssq*diag(V))
%
function yprime = lvrhs(t,y,p)
a = p(1); b = p(2); c = p(3); r = p(4); K=p(5);
yprime = [a*y(1)*(1-y(1)/K)-b*y(1)*y(2); -r*y(2)+c*y(1)*y(2)];
```

In this case, let's look at a diary file of the output:

```
>>lvkconf
delp =
-0.0482
-0.0001
0.0011
0.0418
474.5652
pnew =
1.0e+03 *
0.0006
0.0000
0.0000
```

```
0.0008
1.1986
s =
4.6523
error =
0.0889
0.0034
0.0029
0.0980
605.6802
```

We notice that the last component of Δp is not small, and this suggests these are not reasonable parameter values. (I.e., the linearization is not justified.) In particular, looking at *pnew*, we see that the suggested value for K is

$$K = 1198.6 \pm 605.6902.$$

Notice that the larger K is, the closer this becomes to the case in which carrying capacity is omitted. Again, we conclude that the model with carrying capacity is not justified for this data.

We now carry out the nonlinear fit with scaling. The analysis is carried out in *lvknonlinearfits.m*.

```
function lvknonlinearfits
%LVKNONLINEARFITS: MATLAB function M-file that takes an initial
%approximation of parameter values and carries out nonlinear
%regression to obtain best-fit parameter values for the Lotka-Volterra
%system with prey carrying capacity for the Hudson Bay data.
%Scaling is used
global years L H w1 w2;
lvdata
w1 = std(H);
w2 = std(L);
guess = [.4302; .0169; .0234; .7646; max(H)]; %Order: a, b, c, r, K
options = optimset('MaxFunEvals',1e4);
[p,error]=fminsearch(@lverr, guess, options);
a = p(1)
b = p(2)
c = p(3)
r = p(4)
K = p(5)
s = sqrt(error/(2*(length(H)-1)-length(p)))
s1 = w1*s
s2 = w2*s
%
```

```
[t,y]=ode45(@lvrhs,[0,20],[H(1); L(1)],[],p);
subplot(2,1,1)
plot(t,y(:,1),years,H,'o')
title('Prey population, model and data','FontSize',14)
subplot(2,1,2)
plot(t,y(:,2),years,L,'o')
title('Predator population, model and data','FontSize',14)
%
function error = lverr(p)
%LVERR: Function defining error function for
%example with Lotka-Volterra equations.
global years L H w1 w2;
[t,y] = ode45(@lvrhs,years,[H(1);L(1)],[],p); %Notice that we pass
%a parameter vector
error = norm(y(:,1)-H')^2/w1^2+norm(y(:,2)-L')^2/w2^2;
%
function value = lvrhs(t,y,p)
%LVRHS: ODE for example Lotka-Volterra paramter
%estimation example. p(1)=a, p(2) = b, p(3) = c, p(4) = r, p(5)=K
value=[p(1)*y(1)*(1-y(1)/p(5))-p(2)*y(1)*y(2);-p(4)*y(2)+p(3)*y(1)*y(2)];
```

An implementation of this file is given below.

```
>>lvknonlinearfits
a =
0.5795
b =
0.0287
c =
0.0256
r =
0.8004
K =
2.1219e+03
s =
0.2381
s1 =
5.0983
s2 =
3.9655
```

In this case

$$
\begin{aligned}
a &= .5795 \\
b &= .0287 \\
c &= .0256 \\
r &= .8004 \\
K &= 2,121.9.
\end{aligned}
$$

Last, let's try to obtain confidence intervals on these estimates. For this, we use *lvkconfs.m.*

```
function lvkconfs
%LVKCONFS: MATLAB function M-file for computing 95% confidence
%intervals for parameter estimates of the Lotka-Volterra
%model fit to the Hudson Bay data.
%Parameter values
%Scaling is used
lvdata;
w1 = std(H);
w2 = std(L);
y0 = [30.0;4.0];
pbar = [.5795 .0287 .0256 .8004 2121.9];
[t ybar]=ode45(@lvrhs,years,y0,[],pbar);
dely = [H'/w1; L'/w2] - [ybar(:,1)/w1; ybar(:,2)/w2];
%Set increment for derivative approximations
h = 1e-6;
[t ybar1]=ode45(@lvrhs,years,y0,[],[pbar(1)+h pbar(2) pbar(3) pbar(4) pbar(5)]);
[t ybar2]=ode45(@lvrhs,years,y0,[],[pbar(1) pbar(2)+h pbar(3) pbar(4) pbar(5)]);
[t ybar3]=ode45(@lvrhs,years,y0,[],[pbar(1) pbar(2) pbar(3)+h pbar(4) pbar(5)]);
[t ybar4]=ode45(@lvrhs,years,y0,[],[pbar(1) pbar(2) pbar(3) pbar(4)+h pbar(5)]);
[t ybar5]=ode45(@lvrhs,years,y0,[],[pbar(1) pbar(2) pbar(3) pbar(4) pbar(5)+h]);
F1 = [ybar1(:,1)-ybar(:,1) ybar2(:,1)-ybar(:,1) ybar3(:,1)-ybar(:,1) ybar4(:,1)-
ybar(:,1) ybar5(:,1)-ybar(:,1)]/h; ...
F2 = [ybar1(:,2)-ybar(:,2) ybar2(:,2)-ybar(:,2) ybar3(:,2)-ybar(:,2) ybar4(:,2)-
ybar(:,2) ybar5(:,2)-ybar(:,2)]/h;
F = [F1/w1;F2/w2];
delp = F\dely
pnew = pbar'+delp
%For 95% confidence interval
q = 2*(length(H)-1)-length(pbar);
l = stut(q,.95);
V = inv(F'*F);
ssq = (norm(dely-F*delp)^2)/q;
s = sqrt(ssq)
s1 = s*w1
```

```
s2 = s*w2
error = l*sqrt(ssq*diag(V))
%
function yprime = lvrhs(t,y,p)
a = p(1); b = p(2); c = p(3); r = p(4); K=p(5);
yprime = [a*y(1)*(1-y(1)/K)-b*y(1)*y(2); -r*y(2)+c*y(1)*y(2)];
```

Finally, the implementation is given below.

```
>>lvkconfs
delp =
-0.0014
-0.0001
0.0000
0.0018
138.1445
pnew =
1.0e+03 *
0.0006
0.0000
0.0000
0.0008
2.2600
s =
0.2381
s1 =
5.0980
s2 =
3.9652
error =
1.0e+03 *
0.0001
0.0000
0.0000
0.0001
5.6043
```

Again, ΔK is unreasonably large, and we conclude that the problems we have with incorporating K are not a result of scaling.

2. We're solving the PDE

$$\begin{aligned} u_t &= k u_{xx} \\ h(u(0,t) - T_0) - k u_x(0,1) &= 0 \\ h(u(L,t) - T_0) + k u_x(L,t) &= 0 \\ u(x,0) &= 25. \end{aligned}$$

See *heat1.m*.

```
function heat1
%HEAT1: MATLAB function M-file that solves the
%heat equation with boudary conditions obtained using
%Newton's law of cooling.
%
m=0; %Recall that m is part of the PDE specification in MATLAB
tvals=0:.02:10; %Time discretization
xvals=0:.01:1; %Space discretization
%Now evolve with in time
u = pdepe(m,@eqn,@initial,@bc,xvals,tvals);
fig=plot(xvals,u(1,:),'erase','xor');
%axis([0 1 0 80])
title(['Time = ',num2str(0),'years'])
pause
for k=2:length(tvals)
set(fig,'xdata',xvals,'ydata',u(k,:));
%axis([0 1 0 80])
title(['Time = ',num2str(tvals(k)),'seconds'])
end
pause
mesh(xvals,tvals,u)
title('Mesh Plot for Temperature')
xlabel('Position x')
ylabel('Time t')
zlabel('Temperature u')
pause
%
surf(xvals,tvals,u)
title('Surface Plot for Temperature')
xlabel('Position x')
ylabel('Time t')
zlabel('Temperature u')
%PDE
function [c,f,s] = eqn(x,t,u,DuDx)
%Define the PDE
k = .1;
c = 1; f = k*DuDx; s=0;
%Initial Condition
function value = initial(x)
value=0;
%Boundary condition
function [pl,ql,pr,qr]=bc(xl,ul,xr,ur,t)
%Define boundary data
```

```
k=.1; h=.03; T0=25;
pl=h*(ul-T0); ql=-1; pr=h*(ur-T0); qr=1;
```

The profile plot is given in Figure 9.

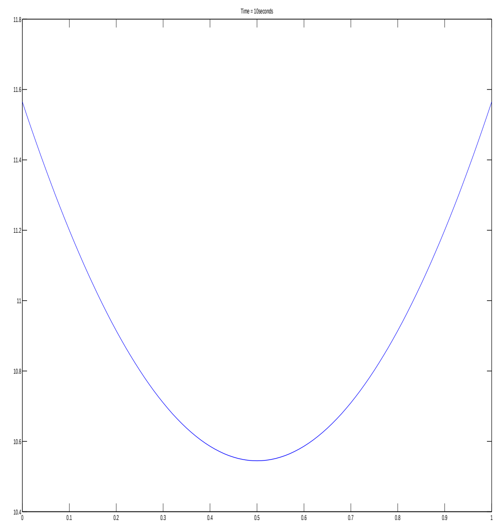

Figure 9: Temperature Profile Plot.

The mesh plot is given in Figure 10.

3. We use the M-file *comppde1.m*.

```
function comppde1
%COMPPDE1: MATLAB function M-file that solves a
%three-species competition model.
%
m=0; %Recall that m is part of the PDE specification in MATLAB
tvals=0:.02:25; %Time discretization
xvals=0:.01:1; %Space discretization
%Now evolve with in time
u = pdepe(m,@eqn,@initial,@bc,xvals,tvals);
subplot(3,1,1)
fig1=plot(xvals,u(1,:,1),'erase','xor');
axis([0 1 min(reshape(u(:,:,1),[],1)) max(reshape(u(:,:,1),[],1))]);
title(['Time = ',num2str(0),'years'])
subplot(3,1,2)
fig2=plot(xvals,u(1,:,2),'erase','xor');
axis([0 1 min(reshape(u(:,:,2),[],1)) max(reshape(u(:,:,2),[],1))]);
title(['Second Competitor'])
subplot(3,1,3)
```

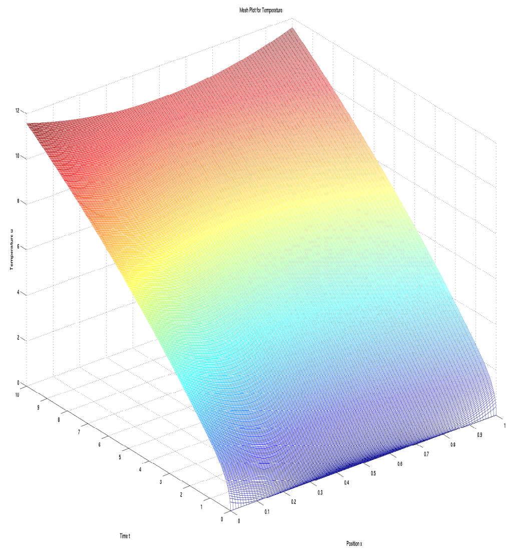

Figure 10: Temperature Mesh Plot.

```
fig3=plot(xvals,u(1,:,3),'erase','xor');
axis([0 1 min(reshape(u(:,:,3),[],1)) max(reshape(u(:,:,3),[],1))]);
title(['Third Competitor'])
pause
for k=2:length(tvals)
subplot(3,1,1)
set(fig1,'xdata',xvals,'ydata',u(k,:,1));
title(['Time = ',num2str(tvals(k)),'seconds'])
subplot(3,1,2)
set(fig2,'xdata',xvals,'ydata',u(k,:,2));
title(['Second Competitor'])
subplot(3,1,3)
set(fig3,'xdata',xvals,'ydata',u(k,:,3));
title(['Third Competitor'])
pause(.1)
end
%PDE
function [c,f,s] = eqn(x,t,u,DuDx)
%Define the PDE
r1 = .02; r2 = .03; r3 = .05;
s1 = 2; s2 = 2; s3 = .5; s4 = 1; s5 = .5; s6 = 1;
K1 = 20; K2 = 10; K3 = 10;
B = 1e-5*[1 2 1; 5 3 2; 10 .8 30];
c = [1;1;1];
f = B*DuDx;
```

```
s=[r1*u(1)*(1-(u(1)+s1*u(2)+s2*u(3))/K1);r2*u(2)*(1-(s3*u(1)+u(2)+s4*u(3))/K2);r3*u(3)*(1-
(s5*u(1)+s6*u(2)+u(3))/K3)];
%Initial Condition
function value = initial(x)
value = [20*(1+cos(pi*x));10*(1+sin(2*pi*x-pi/2));10*(1+sin(pi*x-pi/2))];
%Boundary condition
function [pl,ql,pr,qr]=bc(xl,ul,xr,ur,t)
%Define boundary data
pl = [0;0;0];
ql = [1;1;1];
pr = [0;0;0];
qr = [1;1;1];
```

The initial configuration is given in Figure 11, and the final configuration is given in Figure 12.

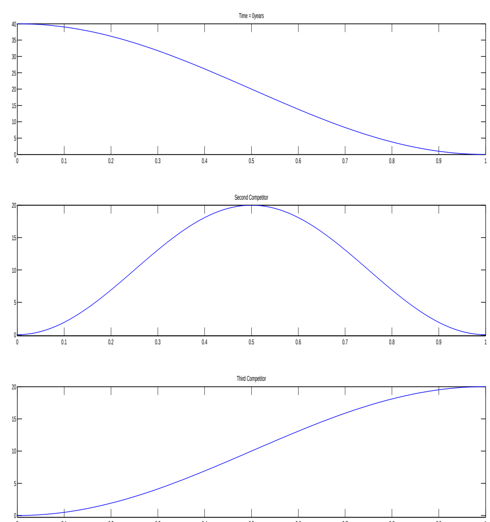

Figure 11: Initial configuration for 3-competitor model.

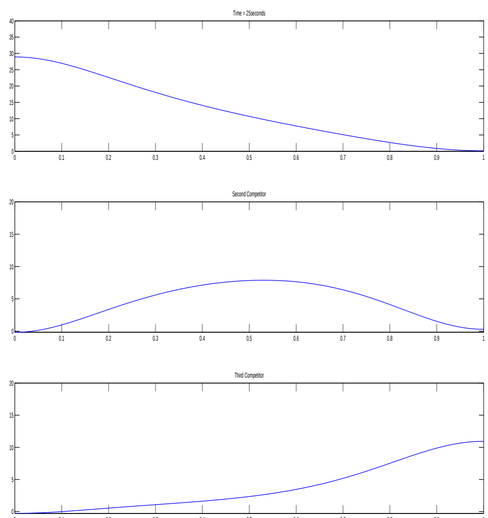

Figure 12: Final configuration for 3-competitor model.