Assignment, Biology (MA5678)

February 15, 2023

Contents

1 Requirements **3**
- 1.1 Requirements for Section 1 . . . 3
- 1.2 Requirements for Section 2 . . . 4
  - 1.2.1 Results and their Presentation . . . 4
- 1.3 Requirements for Section 3 . . . 5
  - 1.3.1 Numerical Solutions of a Biological PDE Model . . . 5
  - 1.3.2 Results and their Presentation . . . 5

2 Work Additional to the Standard Package **6**

3 Marking Scheme and Threshold Criteria **7**
- 3.1 Marking Scheme . . . 7
- 3.2 Marking Criteria . . . 7
- 3.3 Requirements for Mathematics Writing . . . 8

4 Generalities on the Logistic Model **9**

5 Simulation of a Stochastic Process **9**
- 5.1 Stochastic Nonlinear Logistic Equation . . . 10
- 5.2 Matlab Functions . . . 10
  - 5.2.1 Simple Code for (6) . . . 10
  - 5.2.2 Vectorised Version of the Simple Code for (6) . . . 11
  - 5.2.3 Another Piece of Code for (6) . . . 11
  - 5.2.4 Nonlinear Logistic Equation . . . 12

6 Advection (Transport) Equation **13**

7 Two Dimensional Heat Equation **15**

8 Insect Dispersal Model **17**

The assignment is a substantial piece of work where you shall implement and discuss finite difference numerical schemes used in biological models OR option pricing. **You may choose freely which of the two topics you do**. Note also, that in each topic there are different, but comparable questions. This is the biological models assignment. From your student number it is focused on the **insect dispersal model**. The assignment contributes to 100% of the final mark for MA5678. There will be a viva of about 45 minutes on your final submission to assess authorship and clarify certain aspects of your submission. The viva itself will not be marked.

The general aim of the assignment is to create MatLab programmes that can be used to simulate the growth of reaction diffusion models and use the programmes to answer questions about the behaviour of the biological populations (animals, chemicals, patterns etc.)

The final submission is due on

To Be Discussed.

Due to other piece of coursework in other blocks, the plan is to meet in Week 24 to assess progress and to determine a reasonable date of submission.
via up-loading files on Wiseflow, **zipped in one folder** with your student number as its 'name', ie 1980745.zip, say. The 'main file' in Wiseflow's speak will be the pdf file of the assignment. Relevant files include the .tex and .pdf files of your report with all the picture files and Matlab codes. There is no maximum length for the submission, but snappy text will be rewarded. The report should be typed in LaTeX (support provided in the computer labs) and the numerics done in Matlab.

The submission of the **standard package** (see the marking criteria) will result in a maximum of an $A-$ grade (70%). There will be suggestions for **additional work** beyond the standard package that will be rewarded up to a maximum of 30% (if you answer more of the additional work it will be capped at 30%).

1 Requirements

For the full assignment you should present your results in **THREE sections**:

1. Section 1 is about getting each of you on speed with the logistic growth model and simple stochastic simulation of populations.
2. Section 2 is concerned with the understanding, control and modify programmes that will be useful to model your particular biological situation in the final part.
3. In Section 3 you shall derive algorithms to solve PDE problems described thereafter to write programmes that will approximate the PDE models of your particular biological situation.

1.1 Requirements for Section 1

The questions you have to answer are the following:

1. Describe the role of r and K in the deterministic logistic equation in Section 4 and determine its analytical solution starting at some initial population X_0.

   You can look in the literature or internet or solve the **separable ODE** by yourself. If you use external sources, reference them properly.

2. Create a Matlab script that

   (a) illustrate on one graph the typical solutions of the deterministic logistic model (2).

   (b) generates 10 runs of solutions in the stochastic case (8) in Section 5.1 for a range of five values of the volatility σ and two different time horizon. You should adapt one of the Matlab functions given in Section 5.2 for the simpler stochastic ODE (3). The values of the volatility is of your choice, but take a wide span of values that produce different behaviour. The last two digits YZ of your student number represent the initial population $X_0 = YZ$.

   (c) Comment every line of your Matlab programme.

The graphs can be made with Matlab, or LaTeX or any other means. But, if you import a picture/graph, I think it is better if it is in postscript (.eps file). The following commands put the file fig.eps into a figure environment (See LaTeX instructions for details of the extra commands like [h], height=3cm etc).

```
\begin{figure}[h]
\begin{center}
\epsfig{file=fig.eps,height=3cm}
\end{center}
\caption{caption text} \label{fig0b}\end{figure}
```

1.2 Requirements for Section 2

For Section 2 you should **write TWO programmes** that will solve

1. a one dimensional time dependent PDE problem,
2. and a two dimensional diffusion time dependent problem.

More precisely, you should do the following.

1. Implement and answer some questions about the numerical scheme for an advection equation (See Section 6).

   Your outputs should be graphical representations of the solutions at different times, different enough so we can see what is happening to the solution.

2. (a) Implement the numerical scheme given in `Heat2D.m` for the two dimensional heat equation for two different initial values giving two qualitatively different results.

   (b) Modify the programme `Heat2D.m` to accommodate any initial condition of your choice. Illustrate your modified programme using three initial conditions of your choice giving three qualitatively different results.

More details are in Sections 6, 7 and 8.

1.2.1 Results and their Presentation

In this section you should **present the results you got from using the programmes using tables and graphs**. In the standard package and work beyond it, you will be asked for some type of results. Do not forget to describe what the tables and pictures represent, as well as highlighting their main features.

There will be a description of the standard package for them, including input and output, and suggested work beyond it. You will be free to organise your programmes inside the requests of the standard package. **If you use pieces of code written by someone else, you should acknowledge your source and explain what that piece of code is doing.** In this section of your write-up, you should make clear what are the input and output of your programmes (how to use them) and describe the algorithms you use. Between the text of the report and the comments in the programmes, **your numerics should be clear and precisely described**.

1.3 Requirements for Section 3

1.3.1 Numerical Solutions of a Biological PDE Model

For Section 3, you are asked to do the following:

1. Derive finite difference numerical algorithms to solve your given PDE models corresponding to insect dispersal (see Section 8). DO NOT HESITATE TO SEEK ADVICE BEFORE IMPLEMENTING A COMPLICATED CODE.

2. Implement your algorithms in Matlab. Typically, you may modify the codes given to you in the context of Section 2.

3. Apply your code to the situations described in Section 8.

   Again, your outputs should be graphical representations of the solutions at different times.

You will be free to organise your programmes inside the constraints of the questions. **If you use pieces of code written by someone else, you should acknowledge your source and explain what that piece of code is doing.** In this section of your write-up, you should make clear what are the input and output of your programmes (how to use them) and describe the algorithms you use. **Your numerics should be clear and precisely described** between the text of the report and the comments in the programmes.

1.3.2 Results and their Presentation

You should **present the results you got from using the programmes using mainly graphs (sometimes tables)**. In the standard package and work beyond it, you will be asked for some type of results. Do not forget to describe what the tables and pictures represent, as well as highlighting their main features.

2 Work Additional to the Standard Package

There will be additional questions with the marks total capped at 30% to reach the maximum of 100%. You may attempt as many additional questions as you like but the total will still be capped at 30%, so you cannot compensate directly short comings in the standard package using the additional work. Those additional questions are as follows:

1. (10 marks) For the advection equation, test the stability as a function of $\rho = \dfrac{\delta t}{\delta x}$. Write an implicit code and test its stability. Compare both codes.

2. (10 marks) Study the stability of the code for the two dimensional heat equation as a function of
$$\rho = \frac{\delta t}{(\delta x)^2} + \frac{\delta t}{(\delta y)^2}.$$

3. (10 marks) For the final insect dispersal model (14), modify `Heat2D.m` to solve the problem if the dispersal occurs in two space dimensions x and y.

4. (twice 10 marks, one on Section 2 and/or one on Section 3) **Warning: this question is left deliberately vague. It is there so you can exercise your freedom to explore new avenues. You will not receive prior feedback on technical details or on the topics you choose.** Do additional numerical work in either or both work in Sections 2 or 3 that is not requested or suggested in the full package (standard package and additional questions). Do not forget to document what you are doing.

   Marks will be given mainly depending on originality, technical competence (ie. how correct is what you are doing?), explanations and exploitation of results.

Remark 2.1. *You may use your own codes as well, but it is an 'interesting exercise'* **to use someone else's code**. *Read (to understand) carefully the input structure because it does not necessarily fit exactly with your problem. This can then lead to wrong inputs, then to error messages that are not easy to decode. This is why we ask you to comment the given codes, to help you really look carefully at them.*

You are advised to sketch your algorithm, and seek feedback on it, before embarking in complex and time consuming coding that may be faulty to start with.

3 Marking Scheme and Threshold Criteria

3.1 Marking Scheme

In total, for the standard and additional packages, there will be

1. **25 marks for Section 1** (including the new discussions necessary for the pricing algorithms of your options),
2. **20 marks for the standard package of Section 2**, and
3. **25 marks for the standard package of Section 3**,
4. **30 marks (capped) for additional package questions**, as many as you like, but capped.

For each section, there will be 20% of the marks for writing-up and presentation, taking into account in equal measure 5%:

1. if the text is precise and to the point,
2. if the full width of the pages is used and the referencing properly done,
3. if the mathematics is properly typed and displayed and properly imbedded in sentences,
4. if the Matlab scripts you wrote are properly commented.

3.2 Marking Criteria

To get at least an A-, the following threshold must be reached.

1. The report is error free in term of the contents of the requirements.
2. The report is grammatically correct and free of major spelling errors.
3. The report is concise but every terms and elements are well defined.
4. The report satisfies the requirement for Mathematics writing (see Section 3.3).
5. The Matlab programme is error free and produce correct outputs.
6. The Matlab programme is properly commented.
7. The layout of the report is controlled by an effective use of LaTeX.

For a pass mark D-, the requirements 1-3 and 6 are achieved and the errors that can be present do not invalidate that achievement. Having satisfied one more requirement, and if the errors that can be present do not invalidate that achievement, students will get at least a C-. Having satisfied two more requirements, and if the errors that can be present do not invalidate that achievement, students will get at least a B-.

3.3 Requirements for Mathematics Writing

1. Every mathematical term, including single name of variables or functions, should be in math mode: like ‘the function f’, not ‘the function f’.

2. Sentences must be fully punctuated with commas and full stops including mathematical expression. The display of mathematics is for the ease of reading, but does not change the request for full punctuation.

3. When mathematics is displayed, it should be centered in the middle of the page. LaTeXdoes that naturally using the display commands.

4. Use the full width and length of the page. Do not use only the top and left side of the page.

5. Make proper references for information you use without proof. The list of references should be at the end of the text using the ‘bibliography’ structure.

4 Generalities on the Logistic Model

A population is described in continuous time $t \in [0, \infty)$ by a function $X : [0, \infty) \to \mathbb{R}$. The simpler time evolution of the population X is the **exponential growth** model given by an ODE in X:

$$\dot{X(t)} = rX(t), \tag{1}$$

where $r = b - d$ is a constant representing the difference between the **birth rate** b and **death rate** d of the population X. The **general solution** of (1) is

$$X(t) = e^{rt}X_0,$$

where X_0 is the population at $t = 0$ (now).

A more sophisticated model, based on the **logistic equation**, is

$$\dot{X}(t) = rX(t)\left(1 - \frac{X(t)}{K}\right), \tag{2}$$

where r, K are constants. This model can be viewed as modifying the constant effective growth rate r of (1) by a variable value taking into effect an overcrowding effect, namely

$$r\left(1 - \frac{X(t)}{K}\right).$$

Even if (2) looks much more complicated, it is actually a separable equation (see Year 1 Calculus), that can be solved exactly as required in the assignment.

5 Simulation of a Stochastic Process

A simple typical stochastic process followed by a population S is the following stochastic ODE:

$$dS = \mu S\, dt + \sigma S\, dW \tag{3}$$

where S is the **population**, $\mu = r$ is the **intrinsic growth**, σ is the **volatility** and dW is a (so called) **Wiener process** (the stochastic (probabilistic) element of the SDE (3)). This is a **stochastic version** of the exponential model (1) where $\mu = r$. The details of dW are not our concern here, but its values are given by

$$dW = \epsilon\sqrt{dt}, \quad \epsilon \in N(0, 1). \tag{4}$$

A simple simulation of the process (3) is to use an Euler approximation:

$$S_k - S_{k-1} = S_{k-1}\left(\mu\Delta t + \sigma\epsilon(k-1)\sqrt{\Delta t}\right), \; 1 \leq k \leq N, \tag{5}$$

where Δt is the discrete time-step, $\epsilon(k-1) \in N(0, 1)$ is a random number drawn at every time-step from the normal distribution of mean 0 and standard deviation 1 ('tossing a normal coin' at each time period). The initial value of the population is S_0. The number N of steps of simulation we use is such that $N\Delta t = T$ where T is the **time horizon** of the run.

Re-organising (5), we get

$$S_k = S_{k-1}\left(1 + \mu\Delta t + \sigma\epsilon_k\sqrt{\Delta t}\right),\ k \geq 1. \tag{6}$$

This iteration can be simply implemented in a programme.

Remark 5.1. *Actually, the exact solution of (3) can be* better simulated *by the following (discrete) stochastic iteration process*

$$S_k = S_{k-1}e^{\Delta t(\mu - \frac{\sigma^2}{2}) + \sigma\sqrt{\Delta t}\,\epsilon(k-1)},\ k \geq 1. \tag{7}$$

This last formula (7) is implemented in the programmes given later in Section 5.2. Note that (6) is an Euler type approximation of (7).

5.1 Stochastic Nonlinear Logistic Equation

For the logistic equation (2), there is an equivalent **logistic stochastic ODE**:

$$dX = rX(t)\left(1 - \frac{X(t)}{K}\right)dt + \sigma X(t)\,dW. \tag{8}$$

For the assignment you are required to adapt one of the following Matlab programmes for (3) to get simulations for (8) using the simpler formula similar to (5)

$$X_k - X_{k-1} = rX_{k-1}\left(1 - \frac{X_{k-1}}{K}\right)\Delta t + \sigma X_{k-1}\epsilon(k-1)\sqrt{\Delta t},\ 1 \leq k \leq N. \tag{9}$$

Remark 5.2. *You need to implement (9) in your programme because the equivalent to formula (7) is actually rather more complicated in the case of the stochastic nonlinear logistic equation, so we use the simpler approximation induced by (9).*

5.2 Matlab Functions

In this section we give you various functions you can adapt to create the script asked for Part 1 of the assignment. In the following, μ is the drift and σ the volatility. The initial value of the asset is S_0. The time horizon is T and the number of time steps are given by NSteps.

5.2.1 Simple Code for (6)

In the following piece of code, the number of runs is NRepl. The output is stored in the matrix (array) SPaths.

```
function SPaths=AssetPaths(S0,mu,sigma,T,NSteps,NRepl)
SPaths = zeros(NRepl, 1+NSteps);
SPaths(:,1) = S0;
dt = T/NSteps;
nudt = (mu-0.5*sigma^2)*dt;
sidt = sigma*sqrt(dt);
for i=1:NRepl
   for j=1:NSteps
      SPaths(i,j+1)=SPaths(i,j)*exp(nudt + sidt*randn);
   end
end
```

5.2.2 Vectorised Version of the Simple Code for (6)

This is the vectorised version of the previous code. To determine it, we rewrite (7) as

$$\ln(S_k) - \ln(S_{k-1}) = (\mu - \frac{\sigma^2}{2})\Delta t + \sigma\sqrt{\Delta t}\,\epsilon(k-1),\ k \geq 1, \tag{10}$$

```
function SPaths=AssetPathsV(S0,mu,sigma,T,NSteps,NRepl)
dt = T/NSteps;
nudt = (mu-0.5*sigma^2)*dt;
sidt = sigma*sqrt(dt);
Increments = nudt + sidt*randn(NRepl, NSteps);
LogPaths = cumsum([log(S0)*ones(NRepl,1) , Increments] , 2);
SPaths = exp(LogPaths);
Spaths(:,1) = S0;
```

5.2.3 Another Piece of Code for (6)

This is another Matlab function simulating the values of an asset on $[0, T]$ using a GBM. The output is in Y.

```
function [Y] = geometric_brownian(NSteps,mu,sigma,T)
NSteps = input('number of step:');
T = input('Enter length of interval:');
mu = input('Enter drift:');
sigma = input('Enter volatility:');

t = (0:1:NSteps)'/NSteps;

W = [0; cumsum(randn(NSteps,1))]/sqrt(NSteps);
t = t*T;
W = W*sqrt(T);
X = (mu-(sigma^2)/2)*t + sigma * W;
Y = exp(X);
plot(t,Y);
title(['Sample Path of Geometric Brownian motion
with diffusion cofficient=' num2str(sigma),' Interest rate=' num2str(r) ])
xlabel(['Time'])
```

5.2.4 Nonlinear Logistic Equation

Finally, we give you a simple code for the simple Euler approximation for the Nonlinear Logistic equation (9):

```
x0=10;
NRepl=1;
NSteps=90;
T=100;
sigma=0.1;
r=0.05;
K=20;
x = zeros(NRepl, 1+NSteps);
x(:,1) = x0;
dt = T/NSteps;
sidt = sigma*sqrt(dt);
t=linspace(0,T, NSteps+1);
for i=1:NRepl
   for j=1:NSteps
      x(i,j+1)=x(i,j)*(1+r*(1-(x(i,j))/K)*dt + sidt*randn);
   end
end

plot(t,x)
```

6 Advection (Transport) Equation

The following programme solves the PDE

$$\frac{\partial u}{\partial t} + c\,\frac{\partial u}{\partial x} = 0, \tag{11}$$

on the domain $\mathbb{R} \times [0, \infty]$ with initial condition

$$u(x,0) = f(x). \tag{12}$$

1. Check that

   $$u(x,t) = f(x - ct)$$

   is a solution of (11) satisfying the initial condition (12).

2. The following programme solves numerically (11,12), using finite differences, on a FIXED interval $[x_{\min}, x_{\max}] \subset \mathbb{R}$.

```
% transport.m
function [solution, N, M] = transport(xmin, dx, xmax, dt, tmax, c, f0)
N = ceil((xmax - xmin) / dx);
xmax = xmin + N*dx;
M = ceil(tmax/dt);
k1 = 1 - dt*c/dx;
k2 = dt*c/dx;
solution = zeros(N+1,M+1);
vetx = xmin:dx:xmax;
for i=1:N+1
   solution(i,1) = feval(f0,vetx(i));
end
fixedvalue = solution(1,1);
% this is needed because of finite domain
for j=1:M
   solution(:,j+1) = k1*solution(:,j) + k2*[ fixedvalue ; solution(1:N,j) ];
end
```

   (a) Fully comment the file `transport.m`. Explain and derive how it is constructed.

   (b) Modify the script `TransportPlot.m` to plot solutions of (12) for many times with f of different shapes and smoothness.

The script to run the previous programme for some given time is given as follows.

```
% TransportPlot.m
function TransportPlot(xmin, dx, xmax, times, sol)
subplot(2,2,1)
plot(xmin:dx:xmax, sol(:,times(1)))
axis([xmin xmax -0.1 1.1])
subplot(2,2,2)
plot(xmin:dx:xmax, sol(:,times(2)))
axis([xmin xmax -0.1 1.1])
subplot(2,2,3)
plot(xmin:dx:xmax, sol(:,times(3)))
axis([xmin xmax -0.1 1.1])
subplot(2,2,4)
plot(xmin:dx:xmax, sol(:,times(4)))
axis([xmin xmax -0.1 1.1])
```

7 Two Dimensional Heat Equation

This script solves, with an implicit scheme, the heat equation

$$\frac{\partial u}{\partial t} = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2}, \tag{13}$$

on the domain $[0,1] \times [0,1] \times [0, T_{\max}]$ subject to the initial condition

$$u(x, y, 0) = u_0(x, y), \quad x, y \in [0,1].$$

The following programme `Heat2D.m` plots 2-D slices of the solutions $u(x, y, \hat{t})$ for fixed $\hat{t}$.

To get some understanding of the programme, determine the boundary conditions and run the programme with the following instruction

```
Heat2D(0.0001, 0.05, 0.05, 0.10, [0.02 0.04 0.06 0.08 0.1], 10, [0.2 0.9 0.2 0.9])
```

Questions to be answered are in Section 1.2.

```
function U = Heat2D(dt, dx, dy, Tmax, Tsnap, value, bounds)
% make sure steps are consistent
Nx = round(1/dx);
dx = 1/Nx;
Ny = round(1/dy);
dy = 1/Ny;
Nt = round(Tmax/dt);
dt = Tmax/Nt;
rhox = dt/dx^2;
rhoy = dt/dy^2;
if  rhox + rhoy > 0.5
    fprintf(1,'Warning: bad selection of steps\n');
end
C1 = 1-2*rhox-2*rhoy;
Layers = zeros(2, 1+Nx, 1+Ny);
tpast = 1;
tnow = 2;
iTsnap = Tsnap/dt;
[X, Y] = meshgrid(0:dx:1, 0:dy:1);
% set up initial conditions and plot
Layers(tpast, (1+round(bounds(1)/dx)):(1+round(bounds(2)/dx)), ...
    (1+round(bounds(3)/dy)):(1+round(bounds(4)/dy))) = value;
U = shiftdim(Layers(tpast,:,:));
figure;
surf(X,Y,U);
title('t=0','Fontsize',12);
% Carry out iterations
for t=1:Nt
    for i=2:Nx
        for j=2:Ny
            Layers(tnow,i,j) = C1*Layers(tpast,i,j) + ...
                rhox*(Layers(tpast,i+1,j) + Layers(tpast,i-1,j)) + ...
                rhoy*(Layers(tpast,i,j+1) + Layers(tpast,i,j-1));
        end
    end
    % Plot if required
    if find(iTsnap == t)
        U = shiftdim(Layers(tnow,:,:));
        figure;
        surf(X,Y,U);
        title(['t=', num2str(Tsnap(1)) ],'Fontsize',12);
        Tsnap(1) = [];
    end
    % Swap layers
    tnow = 1+mod(t+1,2);
    tpast = 1+mod(t,2);
end
```

8 Insect Dispersal Model

Let $n(x,t)$ be the density of insects at point $x \in \mathbb{R}$ (one dimensional habitat) and time t. The dispersal equation is

$$n_t = d_0 \left((\frac{n}{n_0})^m n_x \right)_x \tag{14}$$

where m, d_0 and n_0 are positive constants.

1. Determine the density $n(x,t)$ of insects for $t \geq 0$ when a quantity Q is released at the origin $x = 0$ at time $t = 0$. For the numerics, use a finite interval $[-R, R]$. Show the evolution of the density of insects as time progresses.

2. Add a wind factor wn_x to equation (14), where w is a constant representing the speed of the wind. Discuss the effect of w on the solutions (w can also be negative).