

433-385 Modeling, Analysis and Visualisation

Project 1, 2004

Due date: Monday 13th September, 5pm

This project covers the first half of the course: Dynamical Systems and Financial Analysis. **Please make sure to read the notes on submission and assessment at the end of this document. Watch the 385 page for any corrections.**

A. The Logistic Map [2 marks]

You are required to complete the IDL program *logmap3.pro* as outlined in the lecture and tutorial notes.

Question A1: When plotting and zooming in on the bi-furcation diagram, what effect does the number of iterations have on the regions where bifurcations occur?

B. The Mandelbrot Set [5 marks]

You are required to write the following separate programs as outlined in the lecture and tutorial notes:

(i) *mandel2.pro*: using the ROI function to enlarge a user-defined Region Of Interest and replot the set. The image in the new window should be undistorted, the aspect ratio as per the selection, and the window should have a maximum edge length as specified by the user. The program should prompt the user whether they want to exit, zoom, or redefine the variables (maximum number of iterations, maximum value of $|z|$, maximum window edge length).

Your working program should also compute the magnification of each new plot relative to the initial plot and write this to the image window.

Input file: The following input for *mandel2.pro* should be read in from a file called *mandel2_input.dat* and listed on separate lines: `w_init` = initial window width, `h_init` = initial window height, `maxiter` = initial max number of iterations, `zmax` = initial max value of $|z|$ used in the colour assignment.

Question B1: What effect does the maximum number of iterations and value of $\max(|z|)$ have on the Mandelbrot set as one zooms in by many levels?

(ii) *mandel3.pro*: using the ROI function to define a region of interest, zoom in to a user defined magnification for a certain number of frames and animate the sequence thus obtained.

Your working program should plot the Mandelbrot, allow the user to determine a **square** region to zoom in to using the ROI function (i.e. the “zoom point” $c_{zoom} = a_{zoom} + i b_{zoom}$ is defined as the central point of the ROI). The program should then prompt the user for the final magnification (with respect to the current ROI extent) and number of frames for the

animation. Once this data is entered by the user, automate the zoom process from the current ROI centred at c_{zoom} , using the formula

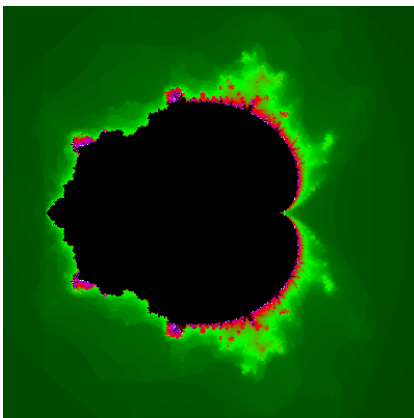
$$\text{Extent new} = \text{extent old} * (\text{extent final}/\text{extent initial})^{(1/n)},$$

create frames for animation (as outlined in Lecture 4) and display the whole sequence using the XINTERANIMATE function. Each image should be square, of the same size, and the set should be undistorted.

Input file: The following input for *mandel3.pro* should be read in from a file called *mandel3_input.dat* and listed on separate lines: `maxiter` = initial max number of iterations, `zmax` = initial max value of $|z|$ used in the colour assignment, `win_edge` = square window width and height.

(iii) *mandel5.pro*: create and explore the Mandelbrot set shown below using the previous programs and the code in the box. There is plenty of interesting action all over the surface of this set for various t -values (shown is $t=2.0$ for reference) and at various magnifications. Find a region and animation sequence (i.e. for a range $[t_{min}, t_{max}]$) that you think is particularly nice. Your final program should compute an animation (with the (a,b) coordinates of the centre of the window displayed on each image) and display using XINTERANIMATE. If you use a non-square window the image should be undistorted.

Input file: The following input for *mande5.pro* should be read in from a file called *mandel5_input.dat* and listed on separate lines: `maxiter` = max number of iterations, `zmax` = max value of $|z|$ used in the colour assignment, `w_init` = window width, `h_init` = window height, `n_frames` = number of animation frames.



```

A three-level depth mapping in (x,y) to compute the set shown
xprev[0:2]=a & yprev[0:2]=b      ; note initialization

xnew = a + it*(xprev[2]^2 - yprev[2]^2) + xprev[1]^2 + xprev[0]^2 $
      - yprev[1]^2 - yprev[0]^2

ynew = b + 2.0*it*xprev[2]*yprev[2] + 2.0*xprev[1]*yprev[1] $
      + 2.0*xprev[0]*yprev[0]

xprev[2]=xprev[1] & yprev[2]=yprev[1]
xprev[1]=xprev[0] & yprev[1]=yprev[0]
xprev[0]=xnew & yprev[0]=ynew

z = complex(xnew,ynew) + c      ; continue usual Mandelbrot

```

Image: Centre at $(a_0, b_0) = (-0.1, 0)$, Extent = 0.5 in both a and b , Time = 2.0

C Predator-Prey Systems

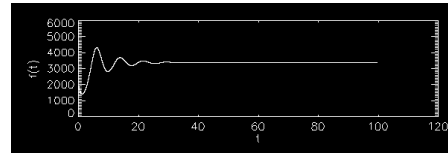
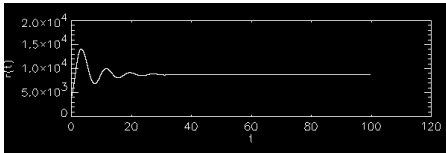
1. [1 mark] Complete the IDL code *rabfox.pro* which solves the system of equations:

$$\frac{dr}{dt} = \alpha r(t) \left(1 - \frac{r(t)}{K}\right) - \beta \frac{r(t)f(t)}{1+r(t)/s} \quad \frac{df}{dt} = -\gamma f(t) + \delta \frac{f(t)r(t)}{1+r(t)/s}$$

for arbitrary values of the parameter set (α , β , γ , δ , s and K) as defined in lectures. Your program should compute the functions $r(t)$ and $f(t)$ from the initial values, open 2 windows corresponding to $r(t)$ and $f(t)$, and plot the results for $r(t)$ and $f(t)$. Finally open a 3rd window and plot the parametric plot in the r - f plane.

Input file: The following input for *rabfox1.pro* should be read in from a file called *rabfox_input.dat* and listed on separate lines: $t_{\max} = \max t$ value, Δt , $r(0)$, $f(0)$, α , β , γ , δ , s , and K .

Test case: for the parameter set ($r(0) = 4000$, $f(0) = 2000$, $\alpha = 1.6$, $\beta = 0.0005$, $\gamma = 1.4$, $\delta = 0.0003$, $s = 10,000$ and $K = 20,000$) the system should converge to $r \rightarrow 8750$ and $f \rightarrow 3375$ (see plots below) for $\Delta t = 0.001$.



2. [2 marks] An ecosystem in a finite area (e.g. an island) comprises solely of rabbits and foxes. Due to a natural disaster the fox population went below a critical number $N_c = 100$, after which the population was so sparse that the foxes became extinct. As a result of this the rabbit population started to increase exponentially towards the carrying capacity of the system, $K=40,000$. To reduce this rabbit plague it was decided to introduce a new fox population to control the rabbit numbers.

Your job is to carry out an analysis of the ecosystem (using *rabfox.pro*) and determine a maximum number of foxes to introduce into system so that the fox population does not go below N_c . For $\Delta t = 0.01$, 0.001 and 0.0001 solve the system out to $t = 100$ years for a number of values of $f(0) = [100 - 2000]$ and make a table of the minimum value of $f(t)$ for each case versus the value of $f(0)$. Be mindful of the accuracy of the solution out to $t=100$ years as you decrease Δt . Data: At the time of fox re-introduction ($t = 0$ years) the number of rabbits was estimated to be 22,000. Long-term observations of the system determined the parameters: ($\alpha = 1.6$, $\beta = 0.0030$, $\gamma = 1.3$, $\delta = 0.0002$, $s = 32,000$ and $K = 40,000$ [time measured in years]).

Question C1: What happens if too many foxes are introduced?

Question C2: To provide a buffer against loss of foxes from unforeseen causes, find the maximum number of foxes $f(0)|_{\max}$ which can be introduced so that the system never falls below $N_c = 100$.

D Financial Analysis

1. [3 marks] For a given set of $\{S(0), \mu$ and $\sigma\}$, write an IDL program *finance1.pro* which simulates the random walk:

$$S(t + \delta t) = S(t)(1 + \mu\delta t + \sigma\phi\sqrt{\delta t}),$$

where the symbols are the same as those defined in the lecture notes. Simulate the random walk over the range $t = [0, T=1]$ years, with $\delta t=1/N$ (where $N =$ number data points). From the random walk for $S(t)$ compute the sequence of returns $R(t)=(S(t+\delta t) - S(t))/S(t)$. Compute estimates for μ and σ from the mean and standard deviation of the return sequence (as if this was the raw data you were given, using the template *NormalDist.pro* given in lecture 9). The output should be a plot of the $S(t)$ data (window 1), the $R(t)$ data (window 2) and the distribution of $R(t)$ (window 3). Print the computed μ and σ (compared with the input values).

The following input for *finance1.pro* should be read in from a file called *finance1_input.dat* and listed on separate lines: $S(0)$, μ , σ , T , N . (beware case sensitivity for “ t ” and “ T ”)

Question D1: How well do the computed μ and σ compared with the input values? For a given fixed T , how does one improve the estimate of drift and volatility in this case?

2. [3 marks] Write an IDL program *finance2.pro* which given the data set $\{S(t=0), S(t=T), E, T, r$ and $\sigma\}$ computes the portfolio profit table (see Lecture 11) for the speculative case (no-hedging) and for the case of “ $\Delta(0)$ -hedging”. Your output to screen should look like this:

```
Scenario parameters:
r = 0.06 : sig = 0.18 : T = 2
S(t=0) = 20.00 : S(T) = 30.00 : Ex = 25.00

Black-Scholes:
d1 = -0.27790692 : d2 = -0.53246537
N(d1) = 0.39054191 : N(d2) = 0.29720186

Portfolio Table:

Holding      Worth (t = 0)   Worth (t = 2)
Call Option   1.22            5.00
Short Position -7.81           -11.72
Cash          6.59            7.43

Hedged Profit = 0.71
Non-Hedged Profit = 3.62
```

Note the rounding to 2 decimal places for dollar amounts, e.g. using the formatting command:

```
print, format= ("Cash1 = ", f6.2, " : Cash2 = ", f4.2) ', Cash1, Cash2
```

The following input for *finance2.pro* should be read in from a file called *finance2_input.dat* and listed on separate lines: $S(t=0)$, $S(t=T)$, E , T , σ and r .

Question D2: Consider the call option defined by $\{S(0)=\$34, E = \$42, T=3 \text{ years}, \sigma = 15\% \text{ pa.}, r = 7\% \text{ pa.}\}$. Find the critical value of the final asset price $S(T)$ above which the both the hedged and non-hedged positions will be profitable.

4. [4 marks] Write an IDL program *finance4.pro* (based on the previous programs), which simulates possible outcomes for a given call option, defined by $\{S(t=0), E, r \text{ and } \sigma\}$. For a given random walk simulation in $t=[0,1]$ (daily steps: $\delta t=1/365$) of the asset price $S(t)$ determine the value $S(T)$ for each simulation and hence calculate the $\Delta(0)$ -hedged profit at $t=T$. By running many such simulations build up a distribution of $\Delta(0)$ -hedged profit for the call option. In window 1 *oplot* all random walks of $S(t)$ as the program proceeds. When the simulations are complete, plot the data set for the $\Delta(0)$ -hedged profits in window 2, determine the distribution (as in *finance1.pro*) and plot this in window 3. From the distribution of profits find the maximum likely $\Delta(0)$ -hedged profit (bin value corresponding to the largest frequency in the distribution) and print this value to the distribution window.

Input file: The following input for *finance4.pro* should be read in from a file called *finance4_input.dat* and listed on separate lines: $S(t=0), E, T, r, \mu, \sigma, \text{number_sims} = \text{number of simulations}, \text{number_bins} = \text{number of bins in the distribution}, \text{random_seed} = \text{a random number to start the random generator}.$

Question D3: Consider the two options defined by:

Option 1: $\{S(t=0)=\$100.00, E=\$110.00, T=1 \text{ years}, r=4\% \text{ p.a.}, \mu=40\% \text{ p.a.}, \sigma=10\% \text{ p.a.}\}$

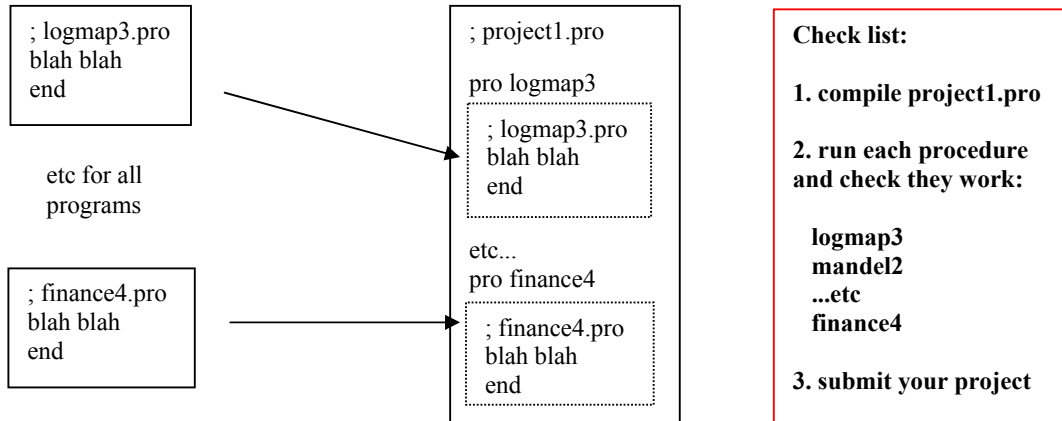
Option 2: $\{S(t=0)=\$220.00, E=\$225.00, T=1 \text{ years}, r=4\% \text{ p.a.}, \mu=20\% \text{ p.a.}, \sigma=20\% \text{ p.a.}\}$

Comment on the profit distributions that you calculate using *finance4.pro* for the two cases (number of simulations = 10000). What is the most likely profit in each case? Which is the better option to consider buying?

Submission Procedure

You must submit a text file *answers1.txt* with tabulated results and written answers to the questions (where asked), plus one program file *project1.pro* which contains the following programs (ready to run) as procedures: *logmap3.pro*, *mandel2.pro*, *mandel3.pro*, *mandel5.pro*, *rabbitfoxes1.pro*, *finance1.pro*, *finance2.pro*, *finance3.pro*, *finance4.pro*.

e.g.



Your program *project1.pro* and procedures contained therein must compile and run on the Computer Science student machines in the lab room (UG 10). Programs with compilation errors receive 0 marks contribution to the final mark for the project.

The files should be submitted using the command **submit 385 1** from one of the student machines in the CS department. It is not necessary to submit any picture files - your codes will be run and checked in the marking process.

Assessment

The project is out of a total of 20 marks and is worth 20% of your final mark.

Projects will be marked on the basis of completeness, correctness, and readability (including documentation).

All assessment in this subject is to be done on an individual basis. Students who submit for assessment the work of other individuals, or who work together excessively, will be penalised by the Department and risk formal prosecution under the University's Disciplinary Regulations. Punishment is also extended to those who supply projects to other students.

Questions and clarification

You may seek assistance from your lab demonstrator during lab classes or by emailing them (see 385 page for the list of email addresses).

Lloyd C.L. Hollenberg

August 15, 2004.