Modelling and Programming IT & Health

Week 3

Deliverables

- Report 1: Assignments 9–12.

Vectors and Matrices

To solve the assignments of this week, it is important that you understand how matrix arithmetic works. In particular, you must understand and use matrix equations when solving Assignments 9 and 10. And, in Assignment 10, you are also required to do matrix arithmetic in MATLAB. Do the following exercises to learn about vectors and matrices and how to use them in MATLAB:

- **C**: Problems 1–2 and 13–16 of Section 9.2.6. First by hand, then check your results using MATLAB.
- **C**: Problem 2 of Section 9.3.4.

Hint: Take particular notice of Sections 1.3–1.4 in **LA** and Sections C.3–C.4 in **P**.

Systems of equations

Assignment 9 is about turning a system of differential equations into a matrix equation. To understand how this works, do the following exercises:

- **C**: Problems 31–32 of Section 9.2.6. (*Hint*: Closely related to Example 2.1 of **LA**.)
- **C**: Problems 1–3 of Section 11.1.4. (*Hint*: Closely related to Example 1 of Section 11.1 in **C**.)

Now, solve Assignment 9.

Dynamical systems

Assignment 10 requires you to simulate the development of your dynamical system over time (using a discrete time model). An analogous problem is to simulate the development of a population over time. The Leslie matrix (see **C**: Section 9.2.5) is a discrete time model for simulating the development of a structured population. This means that the following exercises, which involve the Leslie matrix, can teach you how to solve Assignment 10:

- **C**: Problems 59 and 62 of Section 9.2.6. Do the computations in MATLAB.

Hint: Closely related to Example 17 of Section 9.2.5.

To solve Assignment 10, you also need to know how to do indexing in MATLAB. Work on the following exercises to practice your indexing skills:

- **P**: Exercises 1(e–f) of Section 5.5.

Hint: Confer Sections C.2 and C.4 of **P** to see how indexing works.

Now, solve Assignment 10.

Storing results and looping

Assignments 11 and 12 require you to store simulated system states after each time step. With your solution for Assignment 10, you should be able to compute the state of your dynamical system after one time step. Or after N time steps if you run one of the single-step programs N times. The problem is that every time you take a step by running a single-step program the result from the previous time step gets overwritten. The following exercises show how you can store simulated system states after each time step.

A Leslie matrix is not the only matrix that is useful for modelling population dynamics. We can also use a 2×2 matrix to model the dynamics of two species living in a competitive environment. The following example is adapted from Allman and Rhodes [*Mathematical Models in Biology: An Introduction*. Cambridge University Press, 2004]. Consider a forest that is home to two species of trees: A-trees and B-trees. Let A_t and B_t denote the number of trees of each species in the year t. When a tree dies, a new tree grows in its place, but the new tree might be of either species. The A-trees are relatively long lived, with only 1% dying in any given year, whereas 5% of the B-trees die. However, because they are rapid growers, the B-trees are more likely to succeed in winning a vacant spot left by a dead tree; 75% of all vacant spots go to B-trees, and only 25% go to A-trees. One way to express all this is

$$A_{t+1} = (0.99 + 0.25 \cdot 0.01)A_t + (0.25 \cdot 0.05)B_t \quad (1)$$
$$B_{t+1} = (0.75 \cdot 0.01)A_t + (0.95 + 0.75 \cdot 0.05)B_t \ . \quad (2)$$

Considering this model, do the following:

- Explain the constants in the system of equations (1–2). And rewrite it as a matrix equation.
- Enter the matrix which describes the model into MATLAB. Let us call it `T` for *transition* matrix. And enter the initial tree population in a column vector `x` $= [A_0; B_0] = [10; 990]$.
- Write a MATLAB script which does the following (`pops` is short for populations)

```
pops = [x];
x = T*x;
pops = [pops x];
x = T*x;
pops = [pops x];
⋮
plot(pops');
```

  In place of the vertical dots, insert the two repeated lines 8 more times. Describe how the program works. Make sure you understand how many time steps are taken, what the plot shows, and how simulated results are stored.

- Instead of repeating two lines 10 times, write a `for`-loop (see **P**: Section 8.3) that runs the two lines $N = 10$ times. Increase N and see if the tree population converges toward a *stable distribution*.

Now, solve Assignments 11 and 12.

Curriculum

C Sections 2.1, 4.8, 9.2.5, and pages 702–704.
Discrete Time Models, *Linear Approximation*, and *Systems of Differential Equations*
LA Sections 1.4–1.5 and pp. 17–19. *Linear Mappings* and *Linear Systems of Equations*.
P Chapters 6–8 and Sections C.3–C.4. *Algorithms*, *Conditional Statements*, and *Iterative Statements*.

Read **LA**: Pages 17–19 before **C**: Pages 702–704.

JRF 2009