Technisch-wetenschappelijke software / Scientific Software
Assignment 3: pandemic countermeasures

Emil Løvbak, Pieter Appeltans and Wouter Baert

November 8, 2020

1 Introduction

In the first assignment, you designed code for simulating the SIQRD model for a pandemic

$$\begin{cases} \dot{S}(t) &= -\beta \frac{I(t)}{S(t)+I(t)+R(t)} S(t) + \mu R(t) \\ \dot{I}(t) &= \left(\beta \frac{S(t)}{S(t)+I(t)+R(t)} - \gamma - \delta - \alpha\right) I(t) \\ \dot{Q}(t) &= \delta I(t) - (\gamma + \alpha)\, Q(t) \\ \dot{R}(t) &= \gamma \bigcap \big(I(t) + Q(t)\big) - \mu R(t) \\ \dot{D}(t) &= \alpha \big(I(t) + Q(t)\big) \end{cases} . \tag{1}$$

You will now use this code to analyze the pandemic behavior in terms of the model parameters. This assignment consists of two parts. First off, we expect you to update your existing code based on the feedback you receive for your first assignment. Secondly, you will be expected to expand on the existing code to develop some new functionality, making use of the contents of the first four exercise sessions and corresponding lectures.

2 Updating your previous implementation

We expect you to update your existing code, based on our feedback concerning the first assignment. This feedback consists of general observations, which will be available shortly after this assignment is announced, and individual observations, which will be emailed to you once all assignments have been graded. You can also test your implementation by running the simulations used to generate the plots in the first assignment. In all three cases $\beta = 0.5$, $\gamma = 0.2$, $\alpha = 0.005$, $S_0 = 100$ and $I_0 = 5$. For respectively the "no measures", "quarantine" and "lockdown" cases, δ takes the values 0.0, 0.2 and 0.9. Make a note of any modifications or further remarks you wish to make after receiving your feedback and testing your code and add them to your report.

We also expect the following modifications to your existing code:

- Modify your program so that N and T are now read as respectively the first and second command line arguments. What changes do you need to make to your software?

- Write your own solver module to replace the compiled module that was provided for the first assignment. The interface should be the same as that in the provided code, i.e,

  ```
  subroutine solve(A , bx)
          real(wp), dimension(:,:), intent(inout) :: A
          real(wp), dimension(:), intent(inout) :: bx
  ```

  where `wp` can be either IEEE 754 single or double precision. Use SGESV and DGESV to solve the system[1] in single and double precision, respectively.

Document any major changes you need to make to your software in order to implement this additional functionality.

3 Tweaking pandemic measures

We will now write a program which uses your implementation of the SIQRD model to predict which parameters are necessary to achieve a desired output for the pandemic. Specifically, we will consider the following situation: Given a virus outbreak with a given α and γ, and an initial population of susceptible people S_0 and infected people I_0, determine what the maximal acceptable β is, so that the total number of sick people remains under a given threshold at all times. You can interpret this problem as determining which measures are needed to reduce contacts between individuals (decreasing β) in order to avoid overwhelming the nation's health services.

To simplify this problem, we rely on the fact that, for all other parameters fixed, a larger β means more cases. We also assume that every reduction in β has some cost to society, e.g., closing shops causes damage to the economy. Based on these assumptions, we re-state the problem as follows: What value of β causes the maximal number of cases at any moment in time to be equal to the set limit?

In order to answer this question, go to work as follows:

- Write a Fortran subroutine which expects N, T, $\Delta\beta$ and target maximal value for the number of infections as command line arguments. Have your program reads a given initial state and a set of model parameters from a parameter file in the same format as the previous assignment. This program should then solve the problem

  $$F(\beta) = \text{target},$$

  where $F(\beta)$ returns the maximum value of the sum $I_k + Q_k$ over all k, using Newton iteration, where the derivative of $F(\beta)$ is computed using a finite difference approximation, i.e.,

  $$\frac{d}{d\beta}F(\beta) \approx \frac{F(\beta + \Delta\beta) - F(\beta)}{\Delta\beta}.$$

[1] You can find all necessary documentation at `http://www.netlib.org/lapack/explore-html/index.html`.

Pick a suitable stop criterion for the iteration and motivate your choice. Also think about how you can write this code in order to make it easy to adapt, should you want to replace $F(\beta$ with some other quantity to be optimized in the future.

- Now consider the following case. Given a country with a population of 11 million, where 5 people are infected with a virus with the parameters $\beta = 0.3$, $\gamma = 0.2$ and $\alpha = 0.005$. What value of β should the government strive for, in order to ensure that fewer than 20 000 people are infected at any given moment in the $T = 100$ days following the initial infections? Fix $N = 1000$ and for a range of values of $\Delta\beta = 10^{-i}$, with $i = 1, \dots 16$ generate a table with the following columns: $\Delta\beta$, the computed optimal β, the amount of time needed to compute the optimal β and the number of Newton iterations needed to compute the optimal β.

- Compare the tables produced by your implementations of forward Euler, backward Euler and Heun from the previous assignment. What do you see?

- Pick one of the three simulation algorithms, motivated by your conclusions from this and the previous assignment. Now compare the effect of different optimization flags on your timings.

- Run your code under `valgrind`. What do you see? You do not need to spend too much time on fixing any issues you may encounter, but you should discus what the `valgrind` output means.

- (extra) Test your code on different compilers and make a note of any changes you needed to make in order to make the code work on all of them.

4 Analysis of stability

In this question, we will examine the behavior of the SIQRD model in the neighborhood of an equilibrium point, ie. a starting point for which the number of individuals in the different compartments does not change over time. It is clear that

$$\begin{bmatrix}\overline{S}_0 & 0 & 0 & 0 & 0\end{bmatrix}^T \tag{2}$$

is an equilibrium point of (1) for any $\overline{S}_0$, as nobody can get infected because there are no infectious individuals. We now want to study the effect of a small perturbation to this equilibrium point: will this perturbation grow out or will the solution return to the equilibrium? To this end we will linearize (1) around equilibrium point (2). To do so, recall from Assignment 1 that (1) can be written in the general form

$$\dot{x}(t) = f\big(x(t)\big). \tag{3}$$

We will denote the equilibrium point given in (2) as $\bar{x}_0$ and the perturbation as $\eta(t)$ such that $x(t) = \bar{x}_0 + \eta(t)$. Equation 3 can now be rewritten as:

$$\dot{\eta}(t) = f\big(\bar{x}_0 + \eta(t)\big)$$

This system of non-linear differential equations can be linearized using the Taylor expansion of f around $\bar{x}_0$ and only retaining the first two terms

$$\dot{\eta}(t) = f\big(\bar{x}_0\big) + \frac{\partial f}{\partial x}\big(\bar{x}_0\big)\eta(t)$$

in which $\frac{\partial f}{\partial x}\big(\bar{x}_0\big)$ is the Jacobian of f evaluated at $\bar{x}_0$. Furthermore, $f\big(\bar{x}_0\big) = 0$ as $\bar{x}_0$ is an equilibrium point:

$$\dot{\eta}(t) = \frac{\partial f}{\partial x}\big(\bar{x}_0\big)\eta(t),$$

which is a system of linear differential equations in $\eta(t)$ that describes the behavior of (3) around the equilibrium point $\bar{x}_0$. The analytic solution to this linear systems of equations is given by

$$\eta(t) = \sum_{i=1}^{n} c_i e^{\lambda_i t} v_i,$$

with $\lambda_1, \ldots, \lambda_n$ the eigenvalues of $\frac{\partial f}{\partial x}\big(\bar{x}_0\big)$ and $v_1, \ldots, v_n$ the corresponding eigenvectors (assuming that all eigenvalues of $\frac{\partial f}{\partial x}\big(\bar{x}_0\big)$ are semi-simple). This means that if the real part of at least one of the eigenvalues of $\frac{\partial f}{\partial x}\big(\bar{x}_0\big)$ is strictly larger than zero, then the perturbation grows. To predict if the introduction of a small number of infectious individuals will lead to an outbreak, we have to check if at least one eigenvalue of the Jacobian has real part larger than zero.

We thus expect you to do the following:

- Write a Fortran program that reads a set of model parameters from an input file, as already specified in the previous assignment, constructs a Jacobian matrix and computes the eigenvalues of the matrix. Look in the documentation of the Lapack library, to select a suitable routine for doing this. You are allowed to implement your code in either single or double precision.
- Test your implementation for the parameter files on Toledo. Which ones are stable, i.e., have only negative eigenvalues? Which ones are not? We do not expect you to interpret why one case is stable while another is not.

5 Practical details

The deadline for submission on Toledo is **November 25th at 14:00**, and is strict! Do not wait until the last minute, as we will not accept technical issues as an excuse for late submissions. We expect you to submit your code together with a PDF. We expect you to use the report template provided with this assignment. Your report should be named `hw3_lastname_firstname_studentnumber.pdf`, i.e., if your name is John Smith and your student number is r0123456, your file should be called hw3_smith_john_r0123456.pdf. The code and report should be submitted in a ZIP file with the same name. You can either provide a makefile for your code, or you can add the instructions needed for compiling your code to the main file. Please honestly fill in the total amount of time spent on the assignment in your report. This has no influence on your grade, but helps us

in determining the load of the assignments for future years. There is no hard page limit, but it should not be necessary to write more than a few paragraphs of text for each section of the report. Please include any figures or terminal output relevant to your discussion in the report as well. You can post questions about the assignment on the discussion forum on Toledo. Finally, some things to pay attention to:

- Make sure that your report and code are easy to read, eg. add short commentary explaining functionally.
- If you take part in the Dutch course, you can write your report in Dutch.