Scientific Software

Fortran 95, Homework Assignment 1

Modelling pandemics

Pieter Appeltans, Emil Loevbak & Wouter Baert October 23, 2020

1 Introduction

This first assignment deals with the theory and concepts encountered in Lectures 1 till 3 and Exercise sessions 1 and 2. More specifically, we will focus on the following concepts:

- basic Fortran 95 syntax: variables, programs, functions, subroutines, modules, loops, if-then statements;
- writing to the standard output stream and print formatting;
- arrays;
- floating point numbers in Fortran 95;
- floating point issues.

2 SIQRD models

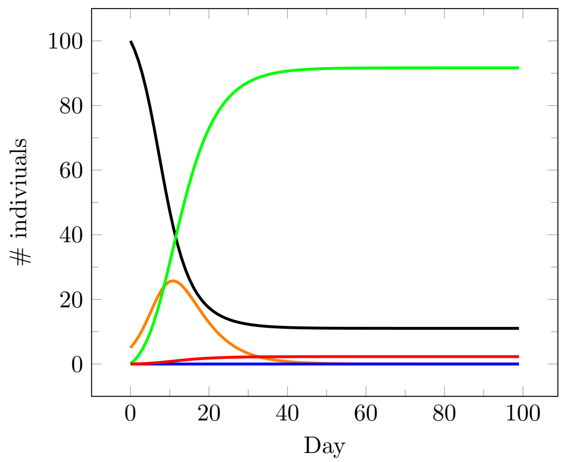
Compartmental models are frequently used to predict the spread of infectious diseases, such as measles, mumps and COVID-19. In such models, the population is partitioned in several compartments, where each compartment contains individuals with the same 'disease characteristics'. Here we will use SIQRD models which consider the following five compartments: **S**usceptible, **I**nfectious, **Q**uarantined (infected but not infectious), **R**ecovered (and immune) and **D**eceased. The evolution of the number of individuals in each compartment is modelled using non-linear ordinary differential equations. Here we will consider the following model based on [1], [2] and [3]:

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

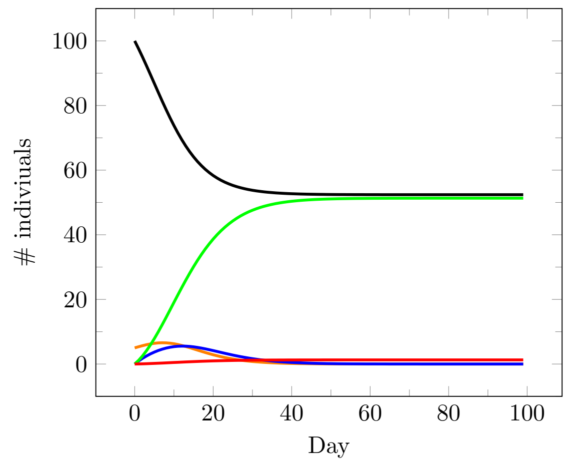
with β the infection rate, μ the rate at which immune people become again susceptible, γ the recovery rate, δ the rate at which infected people get tested and are quarantined and α the death rate. To simulate this SIQRD model we thus have to compute a solution to the set of non-linear ordinary differential equations given in (1) starting from particular initial conditions. Although for some parameter choices there exists an analytic solution, we will resort to numerical methods to solve this problem. These numerical methods are described in the next section. Finally, Figure 1 shows for several parameter choices the predicted dynamics, starting from 100 susceptible and 5 infectious individuals.

3 Solving initial value problems

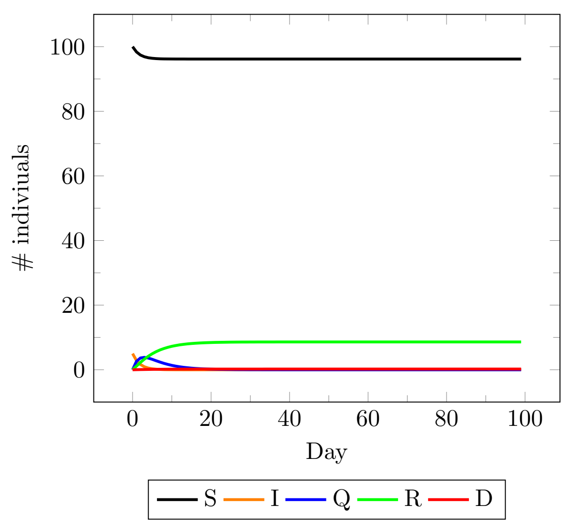
In this section, we will discuss numerical methods to simulate the model given in (1). Simulating this model starting from a particular number of susceptible and infectious individuals is equivalent with



(a) No measures



(b) Quarantining sick people



(c) Lockdown

Figure 1: Simulating (1) in a population of 100 healthy and 5 infectious individuals.

solving an initial value problem (IVP). This section will therefore describe some basic numerical methods to solve such IVPs[1]. In the remainder of this section, we will restrict our attention to time-invariant IVPs of the following general form:

$$\begin{cases} \dot{x}(t) = f(x(t)) & \text{for } t \in [0, T] \\ x(0) = x_0 \end{cases}$$

where we look for a solution $x(t)$ on the interval $[0, T]$. To compute an approximation for this solution, we discretize the interval $[0, T]$ using $N + 1$ equidistant grid points:

$$t_k = \frac{k}{N} T \quad \text{for } k = 0, \ldots, N$$

and compute an approximation for the solution in these grid points

$$x_k \approx x(t_k).$$

Each of the following three subsections discusses a method to compute such an approximation.

3.1 Euler's forward method

Firstly, we will describe Euler's forward method. Starting from the initial point $x_0 = x(0)$, this method approximates the solution in the subsequent grid points using the following formula:

$$x_{k+1} = x_k + \frac{T}{N} f(x_k) \text{ for } k = 1, \ldots N.$$

This is the most basic explicit method to compute a solution for an IVP and was already examined by Leonard Euler in his text book *Institutionum calculi integralis* in 1768. Euler's forward method is a first order method which means that the error at a fixed t is proportional to the step size.

3.2 Heun's method

Secondly, we consider Heun's method, introduced by the German mathematician Karl Heun (1859-1929). In contrast to the previous method, this method is of second order which means that the error at a fixed t decreases quadratically with respect to the step size. The formula to compute the approximate solution x_{k+1} starting from x_k now becomes

$$x_{k+1} = x_k + \frac{T}{N} \left(\frac{1}{2} f(x_k) + \frac{1}{2} f \left(x_k + \frac{T}{N} f(x_k) \right) \right)$$

3.3 Euler's backward method

Finally, we consider Euler's backward method. As Euler's forward method, this method is of first order. However, whereas for Euler's forward method and for Heun's method the approximate solution at time t_{k+1} could directly be calculated from the solution at time t_k, Euler's backward method requires the solution of a system of non-linear equations to obtain x_{k+1}. More specifically, x_{k+1} is found by solving the following non-linear equations in x_{k+1}

$$x_{k+1} = x_k + \frac{T}{N} f(x_{k+1}).$$

[1]For more advanced methods, see the course Numerical Simulation of Differential Equations.

Notice that x_{k+1} now appears both at the right and left hand side of the equation. By bringing x_{k+1} to the right-hand side, we observe that we have to solve a system of non-linear equations of the form

$$\mathbf{0}_5 = G(x_{k+1}) = x_k + \frac{T}{N} f(x_{k+1}) - x_{k+1},$$

with $G : \mathbb{R}^5 \mapsto \mathbb{R}^5$ and $\mathbf{0}_5$ the null vector of length 5. A popular method to solve such systems of non-linear equations is Newton's method. Starting from an initial guess $x_{k+1}^{(0)}$, this method iteratively updates its estimate for the solution as follows:

$$x_{k+1}^{(s+1)} = x_{k+1}^{(s)} - \left(\frac{\partial G}{\partial x}(x_{k+1}^{(s)}) \right)^{-1} G(x_{k+1}^{(s)})$$

with $x_{k+1}^{(s)}$ the estimated solution at iteration s, $\frac{\partial G}{\partial x}\left(x_{k+1}^{(s)}\right)$ the Jacobian of G evaluated at $x_{k+1}^{(s)}$. This iterative refinement of the estimate is repeated until some stopping criterium is fulfilled. Two viable stopping criteria are:

- the forward error $\|x_{k+1}^{(s)} - x_{k+1}^{\star}\|$ (with $x_{k+1}^{\star}$ the solution of the non-linear system, i.e. $G(x_{k+1}^{\star}) = 0$) is sufficiently small;
- the backward error $\|G(x_{k+1}^{(s)})\|$ is sufficiently small.

Remark 1. *To compute*

$$\left(\frac{\partial G}{\partial x}(x_{k+1}^{(s)}) \right)^{-1} G(x_{k+1}^{(s)})$$

we provide a compiled module containing the subroutine `solve`*. This subroutine solves a linear system $Ax = b$ and has the following interface:*

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

Here `bx` *contains the vector b when the subroutine is called and the vector x after completion of the function. Note that* `A` *is also inout, meaning that the contents of the matrix can be altered by the subroutine.*

4 Assignment

4.1 Question 1: exponential growth

In the initial phase of the spread of the disease, i.e. a large number of susceptible individuals and only a few people are immune, the number of infections grows exponential. Therefore, to predict the number of infected individual after a certain number of days we can use the following simplified model:

$$I_d = (1 + r)^d I_0$$

with I_d the predicted number of infections after d days, I_0 the number of infections on day 0 and r the daily infection rate.

- Write a Fortran 95 program that prompts the user for the number of initial infections, the number of days and the daily infection rate (in that order) and reads the user's input from the standard input stream. Write your program in such a way that it is easy to switch between different floating point precision formats.

- Assume that number of infected individuals grows with a daily rate of 0.1 and that the initial number of patients equals 100, now compute how many infections you can expect after 100 days. Do this calculation in both single and double precision[2]? Explain your results. Test your code on `gfortran`, `ifort` and `nagfor`. What do you observe? Are the results the same?
- Repeat the experiment for a daily growth rate of 0.125. What do you observe?

4.2 Question 2: Simulating SIQRD models

Implement Euler's forward method, Euler's backward method and Heun's method in Fortran 95 to simulate the SIQRD model given in (1). The simulation horizon T and the number of grid points $N + 1$ are assumed to be given at compile time. The program should read the system parameters β, μ, γ, α, δ and the initial number of susceptible and infected individuals from the file `parameters.in`. An example file for $\beta = 1.0$, $\mu = 2.0$, $\gamma = 3.0$, $\alpha = 4.0$ $\delta = 5.0$, $S_0 = 100$ and $I_0 = 5$ can be found on Toledo with this assignment. You can initialize the number of quarantined, recovered and deceased people to zero. For each method, store the simulation result in a two dimensional array and write the output in scientific format with 5 significant digits to a file `method_beta_mu_gamma_alpha_delta.out` with `method` the name of the method, beta equal to $\beta * 100$, mu equal to $\mu * 100$, gamma equal to $\gamma * 100$, alpha equal to $\alpha * 100$ and delta equal to $\delta * 100$, in the following format:

$$\begin{array}{cccccc} t_0 & S_0 & I_0 & Q_0 & R_0 & D_0 \\ t_1 & S_1 & I_1 & Q_1 & R_1 & D_1 \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ t_N & S_N & I_N & Q_N & R_N & D_N \end{array}$$

It suffices to implement your code in single precision. Finally, answer the following questions in the provided report template:

- How did you organize your code? What functionality did you put together in a subroutine, function, or module? Why?
- How do you know your implementation is correct? Can you think of some good parameter combinations to check this? What properties of the used model can you use? What properties of the implemented methods can you use?
- Compare the three methods for $\beta = \gamma = 10.0$, $\alpha = 1.0$, $\mu = \delta = 0.0$, $N = 150$, $T = 30$, $S_0 = 100$ and $I_0 = 5$. Generate figures using the provided file `plot.tex`. What do you see?
- Compare the two stopping criteria for Newton's method. Which one do you prefer?

5 Practical information

The deadline for submission on Toledo is **November 3rd at 14:00**. This deadline is strict! Do not wait until the last minute to submit, 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 `hw1_lastname_firstname_studentnumber.pdf`, i.e., if your name is John Smith and your student number is r0123456, your file should be called hw4_smith_john_r0123456.pdf. The code and report should be submitted in a ZIP file with the same name. You an either provide a makefile for your

[2] As defined in the IEEE 754 standard.

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.

References

[1] N. Anand, A. Sabarinath, S. Geetha, and S. Somanath. Predicting the Spread of COVID-19 Using SIR-Model Augmented to Incorporate Quarantine and Testing. *Transactions of the Indian National Academy of Engineering*, 5(2):141–148, 2020.

[2] H. Hethcote, M. Zhien, and L. Shengbing. Effects of quarantine in six endemic models for infectious diseases. *Mathematical biosciences*, 180(1-2):141–160, 2002.

[3] L. Peng, W. Yang, D. Zhang, C. Zhuge, and L. Hong. Epidemic analysis of covid-19 in china by dynamical modeling. *medRxiv*, 2020.