Matlab programs for "Population heterogeneity and consequences for herd immunity to SARS-CoV-2" by Tom Britton, Frank Ball and Pieter Trapman

June 10, 2020

1 Introduction

The following is a list of all matlab files used in the code. They are described in detail below.

R0.m
multiSIRtau.m
backpgf.m
backpgfroot.m
effectR0linearinf.m
effectR0linearroot.m
jacco1.m
jaccomultiactivity.m
multiseirjackoactivitydet.m
multiseirdydt.m
multiseirjackoactivitydetlockdown2.m
multiseirdydtB.m
multiseirjackoactivitydetlockdown3.m
multiseirdydtC.m
multiseirjackoactivitydetlockdown.m
multiseirdydtA.m seirlockdownplots.m seirlockdownplots1.m seirlockdownplots2.m

2 Code for tables

The values of h_D in Table 1 and the fractions infected in Tables 2 and S1 are obtained as follows.

For the *activity structure* model, set vecpi=[.25 .5 .25], pactivity=[.5 1 2], lambda=pactivity'*pactivity and call [inf totinf]=effectR0linearinf(vecpi,lambda,Rzero), where Rzero is the desired value of R_0.

For the *age structure* model, call [lambda, vecpi]=jacco1 and then

[inf totinf]=effectR0linearinf(vecpi,lambda,Rzero),

where Rzero is the desired value of R_0.

For the *age & activity structure* model, call

[lambda, vecpi]=jaccomultiactivity(pactivity, activitypi),

where pactivity=[.5 1 2] and activitypi=[.25 .5 .25],

and then [inf totinf]=effectR0linearinf(vecpi,lambda,Rzero), where Rzero is the desired value of R_0.

function x=R0(vecpi,lambda)

Calculates R_0 for a multitype SIR epidemic having infection rate matrix lambda assuming all infectives have mean infectious period 1. vecpi contains the fractions of the population of the different types. Note that R_0 is the same for the corresponding SEIR epidemic.

function tau=multiSIRtau(vecpi,lambda)

Calculates the final fractions infected of the different types in a multitype SIR epidemic (with $R_0 > 1$) having infection rate matrix lambda assuming all infectives have mean infectious period 1. vecpi contains the fractions of the population of the different types. The program first calculates the extinction probability vector of the associated backward branching process, the elements of which are 1 minus the final fraction infected. Note that the final fractions infected are the same in the corresponding SEIR epidemic.

function f=backpgf(x,vecpi,lambda)

Calculates the joint PGF of the backward branching process associated with the multitype SIR epidemic described in *multiSIRtau.m.*

function f=backpgfroot(x,vecpi,lambda)

Calculates the extinction probability equation of the backward branching process associated with the multitype SIR epidemic described in *multiSIRtau.m.*

function [inf totinf]=effectR0linearinf(vecpi,Lambda,Rzero)

Considers a multitype SIR epidemic having infection contact rate matrix Lambda (this is the matrix A in the supplementary information) assuming all infectives have mean infectious period 1. (vecpi contains the fractions of the population of the different types.) It first calculates the value of the scalar lambda0 so that the epidemic with infection rate matrix lambda0*Lambda has R_0 equal to the desired Rzero. It then determines the value of the scalar lam, so that if the epidemic with infection rate matrix Lam1=lam*Lambda is run to its conclusion and then the epidemic with infection rate matrix lambda0*Lambda is run on the remaining susceptible population, the latter has $R_0 = 1$. (Thus α_* in the paper is given by lam/lambda0.) The vector inf gives the fractions of the different types that are infected by the first epidemic and totinf is the total fraction infected by the first epidemic. This program is used to produce the h_D values in Table 1 and Table 2 of the paper.

function y=effectR0linearroot(x,vecpi,Lambda,Rzero)

This program gives the function, $f(x)$ say, whose root is required to find the scalar lam described in *effectR0linearinf.m.*

function [lambda, vecpi]=jacco1

This program calculates the infection contact rate matrix Lambda (this is the matrix A in the supplementary information) and the vector giving the fractions of the population in different age groups from the data in Wallinga et al (2006).

function [lambda, vecpi]=jaccomultiactivity(pactivity, activitypi)

This program calculates, for the model with differentactivity levels, the infection contact rate matrix Lambda (this is the matrix A in the supplementary information) and the vector vecpi giving the fractions of the population in different age groups from the data in Wallinga et al (2006). pactivity is a vector containing the different activity levels and activitypi is a vecctor containing the fractions of the population having the different activity levels. If the age types are denoted by (A1, A2, A3, A4, A5, A6) and the activity types by (L, M, H) then the 18 types in e.g. vecpi are ordered (A1L, A1M, A1H, A2L, A2M, A2H,..., A6L, A6M, A6H).

3 Code for figures

3.1 Figures in the main paper

To produce the figures in the main paper run the script file *seirlockdownplots.m*, which calls the following m.files (some indirectly) in addition to some described above.

function [tdet ydet]=multiseirjackoactivitydet(gamma, sigma, pactivity, activitypi, Rzero, epsilon, tend)

Numerically solves ODEs for multitype SEIR model for age & activity structure model. gamma = 1/(mean infectious period), sigma = 1/(mean latent period), pactivity is a vector containing the different activity levels and activitypi is a vector containing the fractions of the population having the different activity levels. Rzero is the desired R_0. epsilon is the fraction of the different types that are initially infected. More precisely, it is assumed that for each type, a fraction 1-epsilon are susceptible and a fraction epsilon are latent (exposed). The ODE solver produces a solution over the time interval [0, tend].

tdet is a vector giving the time points that the solution is given. ydet is a matrix, the ith row of which gives the solution at the ith time point. The second dimension of ydet is 54 as there are 18 types in this model. The first 18 columns give the fractions of the population that are in the different classes and susceptible, the second 18 columns give the fractions of the population that are in the different classes and exposed, and the final 18 rows give the fractions of the population that are in the different classes and infected. Note that these have a different interpretation from the corresponding quantities in the supplementary information, which give the corresponding fractions of the class rather than of the population.

The percentaged out material at the end of the code produces the total fraction infected and the cumulative total fraction infected during the epidemic.

function dydt=multiseirdydt(t,y,lambda,gamma,sigma,k)

This function (called by multiseirjackoactivitydet.m) gives the right-hand sides of the system of ODEs for a deterministic multitype SEIR model with k types, all having mean infectious period 1/gamma and mean exposed (latent) period 1/sigma. lambda is the infection rate matrix. y is a vector of dimension 3*k. The first k entries give the fraction of the population infected of the different types, the next k the fraction exposed and the last k the fraction infected.

function [tdet ydet]=multiseirjackoactivitydetlockdown2(gamma, sigma, pactivity, activitypi, Rzero, tlockdown1, alphalockdown1, tlockdown2, alphalockdown2, epsilon, tend)

This is the equivalent code to multiseirjackoactivitydet.m but incorporating lockdown, details of which are given in multiseirdydtB.m below.

function dydt=multiseirdydtB(t,y,lambda,gamma,sigma,k,tlockdown1, alphalockdown1, tlockdown2, alphalockdown2)

This function is the equivalent of multiseirdydt.m to include lockdown. tlockdown1 and tlockdown2 are the start and end of lockdown. lambda gives the infection rates before lockdown. During lockdown, all infection rates are multiplied by alphalockdown1 and after lockdown they are multiplied by alphalockdown2.

3.2 Figures in the supplementary material

To produce the Figures S1 and S2, run the script file *seirlockdownplots1.m*; to produce Figure S3, run the script file *seirlockdownplots2.m.* These script files call the following m.files (some indirectly),in addition to some described above.

function [tdet ydet]=multiseirjackoactivitydetlockdown3(gamma, sigma, pactivity, activitypi, Rzero, tlockdown1, alphalockdown1, tlockdown2, tlockdown3, epsilon, tend)

This is the equivalent code to multiseirjackoactivitydet.m but incorporating lockdown followed by gradual relaxation of lockdown, details of which are given in multiseirdydtC.m below.

function dydt=multiseirdydtC(t,y,lambda,gamma,sigma,k,tlockdown1, alphalockdown1, tlockdown2, tlockdown3)

This function is the equivalent of multiseirdydt.m to include lockdown followed by gradual relaxation of lockdown. lambda gives the infection rates before lockdown. Lockdown starts at time tlockdown1, when all infection rates are multiplied by alphalockdown1, and is released gradually (i.e. linearly) between times tlockdown2 and tlockdown3, so that at time tlockdown3 all infection rates are back at their pre-lockdown values.

function [tdet ydet R0eff]=multiseirjackoactivitydetlockdown(gamma, sigma, pactivity, activitypi, Rzero, tlockdown, alphalockdown, epsilon, tend)

This is similar code to *multiseirjackoactivitydet.m* but incorporating indefinite lockdown which starts at time tlockdown when all infection rates are multiplied by alphalockdown. As well as saving the solution to the ODEs in (tdet,ydet) at each of the time points in tdet it saves in R0eff the effective R_0 (incorporating disease-induced immunity) if lockdown was fully released (i.e. all rates set back to their original values) at that time.

function dydt=multiseirdydtA(t,y,lambda,gamma,sigma,k,tlockdown, alphalockdown)

This function is the equivalent of *multiseirdydt.m* to include indefinite lockdown, which starts at time tlockdown when all infection rates are multiplied by alphalockdown.