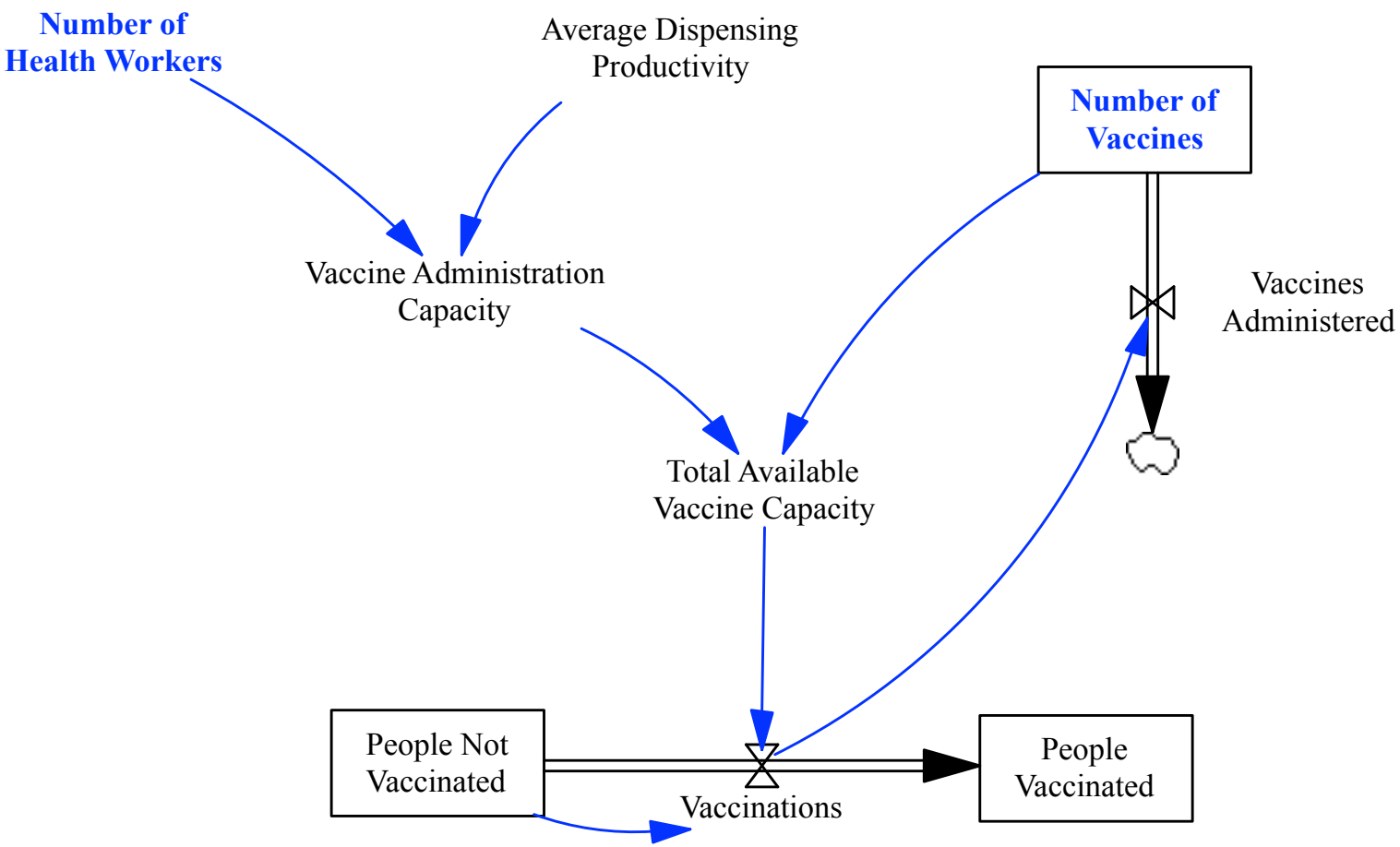
CT561: Systems Modelling & Simulation

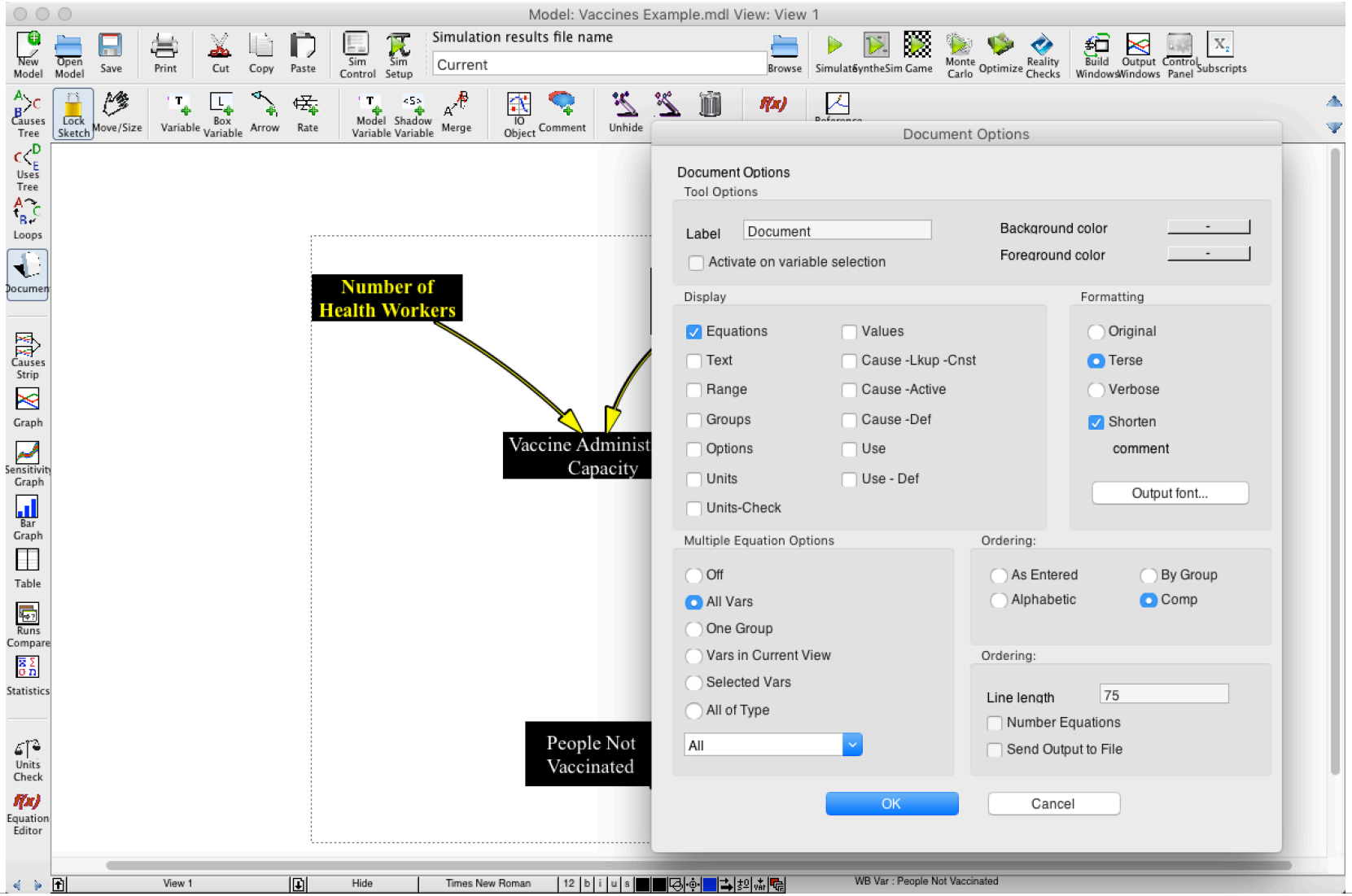
Lecture 9: Exploring Models in R

Dr. Jim Duggan,
School of Engineering & Informatics
National University of Ireland Galway.
https://github.com/JimDuggan/SDMR
https://twitter.com/_jimduggan

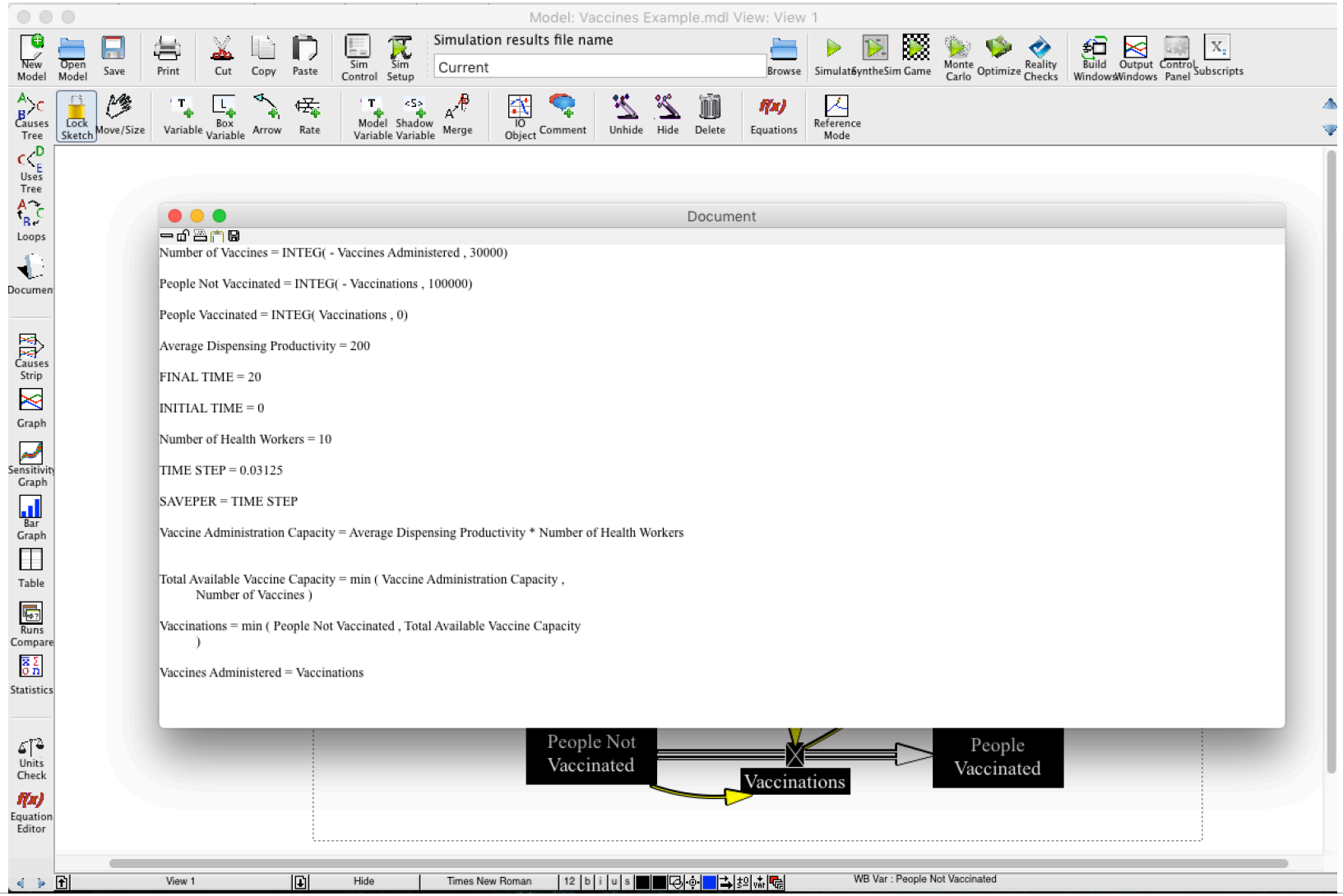
Translation Utility (Vensim – deSolve)



Vensim Document Options



Select equations…



Copy into Rstudio and save (Vacc.txt)

~/Dropbox/R Projects/SDMR - master - RStudio

Go to file/function | Addins

.R | 01 CTrans.R | 02 Vacc.R* | Vaccination.R | Customer.R | SIR.txt | Cust.txt | Vacc.txt

```
Number of Vaccines = INTEG( - Vaccines Administered , 30000)

People Not Vaccinated = INTEG( - Vaccinations , 100000)

People Vaccinated = INTEG( Vaccinations , 0)

Average Dispensing Productivity = 200

FINAL TIME = 20

INITIAL TIME = 0

Number of Health Workers = 10

TIME STEP = 0.03125

SAVEPER = TIME STEP

Vaccine Administration Capacity = Average Dispensing Productivity * Number of Health Workers

Total Available Vaccine Capacity = min ( Vaccine Administration Capacity , Number of Vaccines )

Vaccinations = min ( People Not Vaccinated , Total Available Vaccine Capacity)

Vaccines Administered = Vaccinations
```

20:1 | Text File

Create “compiler” file…

```
source("reader/conv_to_deSolve.R")

output<-sim$translate_vensim("./reader/Vacc.txt")

sim$save_model(output,"./reader/Vaccination.R")
```

```r
library(deSolve)
library(ggplot2)
library(reshape2)
#Displaying the simulation run parameters
START_TIME <- 0.000000
FINISH_TIME <- 20.000000
TIME_STEP <- 0.031250
#Setting aux param to NULL
auxs<-NULL

#Generating the simulation time vector
simtime<-seq(0.000000,20.000000,by=0.031250)
# Writing global variables (stocks and dependent auxs)
stocks <-c( NumberofVaccines = 30000 , PeopleNotVaccinated = 100000 , PeopleVaccinated = 0 )
# This is the model function called from ode
model <- function(time, stocks, auxs){
  with(as.list(c(stocks, auxs)),{
    AverageDispensingProductivity <- 200
    NumberofHealthWorkers <- 10
    VaccineAdministrationCapacity <- AverageDispensingProductivity*NumberofHealthWorkers
    TotalAvailableVaccineCapacity <- min(VaccineAdministrationCapacity,NumberofVaccines)
    Vaccinations <- min(PeopleNotVaccinated,TotalAvailableVaccineCapacity)
    VaccinesAdministered <- Vaccinations
    d_DT_NumberofVaccines  <- -VaccinesAdministered
    d_DT_PeopleNotVaccinated  <- -Vaccinations
    d_DT_PeopleVaccinated  <- Vaccinations
    return (list(c(d_DT_NumberofVaccines,d_DT_PeopleNotVaccinated,d_DT_PeopleVaccinated)))
  })
}
# Function call to run simulation
o<-data.frame(ode(y=stocks,times=simtime,func=model,parms=auxs,method='euler'))
```

Resource on github: /reader

JimDuggan / SDMR

Unwatch 1 | Star 3 | Fork 4

Code | Issues 0 | Pull requests 0 | Projects 0 | Wiki | Pulse | Graphs | Settings

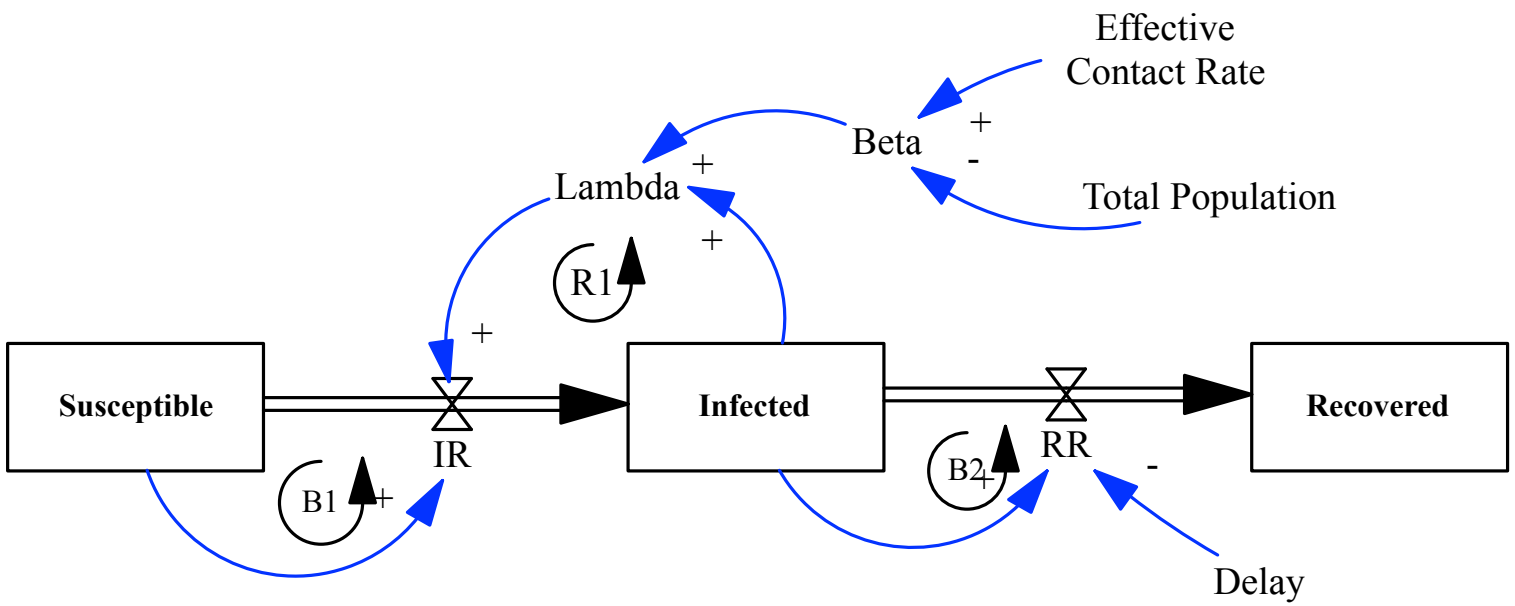
Branch: master | SDMR / reader / | Create new file | Upload files | Find file | History

JimDuggan Adding Vaccination example from lectures — Latest commit 6168d2d 6 minutes ago

..		
01 CTrans.R	Updating reader to filter out blank lines...	3 hours ago
02 Vacc.R	Adding Vaccination example from lectures	6 minutes ago
Current.vdf	Adding lecturing archive from 2015	3 months ago
Cust.txt	Updating reader to filter out blank lines...	3 hours ago
Customer.R	Updating reader to filter out blank lines...	3 hours ago
SIR Vensim.2mdl	Update...	3 months ago
SIR Vensim.mdl	Update README doc...	3 months ago
SIR.R	Updating reader to filter out blank lines...	3 hours ago
SIR.txt	Update README doc...	3 months ago
TestScript.R	Updating test script...	3 months ago
Vacc.txt	Adding Vaccination example from lectures	6 minutes ago

Exploratory Model Analysis – SIR Model

Diffusion - a fundamental process in physical, biological, social and economic settings (Rahmandad and Sterman 2008).



Sensitivity Analysis (library FME)

- Vary key parameters
  - Effective Contacts
  - Recovery Delay
- Perform many simulation runs
- Analyse output

Package 'FME'

February 19, 2015

Version 1.3.2

Title A Flexible Modelling Environment for Inverse Modelling, Sensitivity, Identifiability, Monte Carlo Analysis.

Author Karline Soetaert <karline.soetaert@nioz.nl>, Thomas Petzoldt <thomas.petzoldt@tu-dresden.de>

Maintainer Karline Soetaert <karline.soetaert@nioz.nl>

Depends R (>= 2.6), deSolve, rootSolve, coda

Imports minpack.lm, MASS

Suggests diagram

Description Provides functions to help in fitting models to data, to perform Monte Carlo, sensitivity and identifiability analysis. It is intended to work with models be written as a set of differential equations that are solved either by an integration routine from package deSolve, or a steady-state solver from package rootSolve. However, the methods can also be used with other types of functions.

Overall Idea

R Source code

Run Simulation Function (params)

Model function

Net flow equations go here

Call Model Function
Return Results

Setup Params (FME library)
Call Simulation Function
Display Results

Generating Parameters

```
CE.MIN<-0;            CE.MAX<-7.0
DEL.MIN<-1.0;         DEL.MAX<-10.0
INIT.INF.MIN<-1.0;    INIT.INF.MAX<-25.0;

parRange<-data.frame(
  min=c(CE.MIN, DEL.MIN, INIT.INF.MIN),
  max=c(CE.MAX, DEL.MAX, INIT.INF.MAX)
)

rownames(parRange)<-c("aEffective.Contact.Rate","aDelay","initInfected")

NRUNS<-10
p<-data.frame(RunNo=1:NRUNS,Latinhyper(parRange,NRUNS))
```

parRange data frame & LatinHyper()

```
> parRange
                        min max
aEffective.Contact.Rate   0   7
aDelay                    1  10
initInfected              1  25
>
> p
   RunNo aEffective.Contact.Rate   aDelay initInfected
1      1               5.7435887 4.635804    13.738034
2      2               3.0020634 8.499875    22.389610
3      3               0.8393408 2.684578    17.206845
4      4               2.5906431 7.061717    12.551525
5      5               0.0264442 9.213793    23.679345
6      6               4.2287664 8.193284     8.757535
7      7               6.8971697 4.544439     5.548590
8      8               5.5252311 5.864802     2.606257
9      9               1.8602947 3.583974     7.451173
10    10               3.8608469 1.207800    17.882494
```

Overall idea… *N* simulation runs

```
> p
  RunNo aEffective.Contact.Rate   aDelay initInfected
1     1              5.7435887 4.635804    13.738034
2     2              3.0020634 8.499875    22.389610
3     3              0.8393408 2.684578    17.206845
4     4              2.5906431 7.061717    12.551525
5     5              0.0264442 9.213793    23.679345
```

Run Simulation Function (params)

Model function

Net flow equations go here

Call Model Function
Return Results

The Code (1/3)

```
runsim <- function(rvec){
  # this is the model function
  model <- function(time, stocks, auxs){
    with(as.list(c(stocks, auxs)),{
      aBeta <- aEffective.Contact.Rate / aTotalPopulation
      aLambda <- aBeta * sInfected

      fIR <- sSusceptible * aLambda
      fRR <- sInfected / aDelay

      dS_dt  <- -fIR
      dI_dt  <- fIR - fRR
      dR_dt  <- fRR

      return (list(c(dS_dt,dI_dt,dR_dt),
                   IR=fIR, RR=fRR,Beta=aBeta,Lambda=aLambda,DEL=aDelay,
                   CE=aEffective.Contact.Rate,InitI=initInfected))
    })
  }
```

The Code (2/3)

```
  # setup the individual simulation run
  START<-0; FINISH<-20; STEP<-0.01;
  simtime <- seq(START, FINISH, by=STEP)

  init<-rvec[["initInfected"]]

  a<-c(aTotalPopulation=10000,rvec["aEffective.Contact.Rate"],
       rvec["aDelay"],rvec["initInfected"])

  stocks  <- c(sSusceptible=10000-init,sInfected=init,sRecovered=0)

  o<-data.frame(ode(y=stocks, simtime, func = model,
                    parms=a, method="euler"))
  o$RunNumber<-rvec["RunNo"]
  o
}
```

The Code (3/3)

```
CE.MIN<-0;            CE.MAX<-7.0
DEL.MIN<-1.0;         DEL.MAX<-10.0
INIT.INF.MIN<-1.0;    INIT.INF.MAX<-25.0;

parRange<-data.frame(
  min=c(CE.MIN, DEL.MIN, INIT.INF.MIN),
  max=c(CE.MAX, DEL.MAX, INIT.INF.MAX)
)

rownames(parRange)<-c("aEffective.Contact.Rate","aDelay","initInfected")

NRUNS<-10
p<-data.frame(RunNo=1:NRUNS,Latinhyper(parRange,NRUNS))

out<-apply(p,1,function(x)runsim(x))

df<-rbind.fill(out)
```

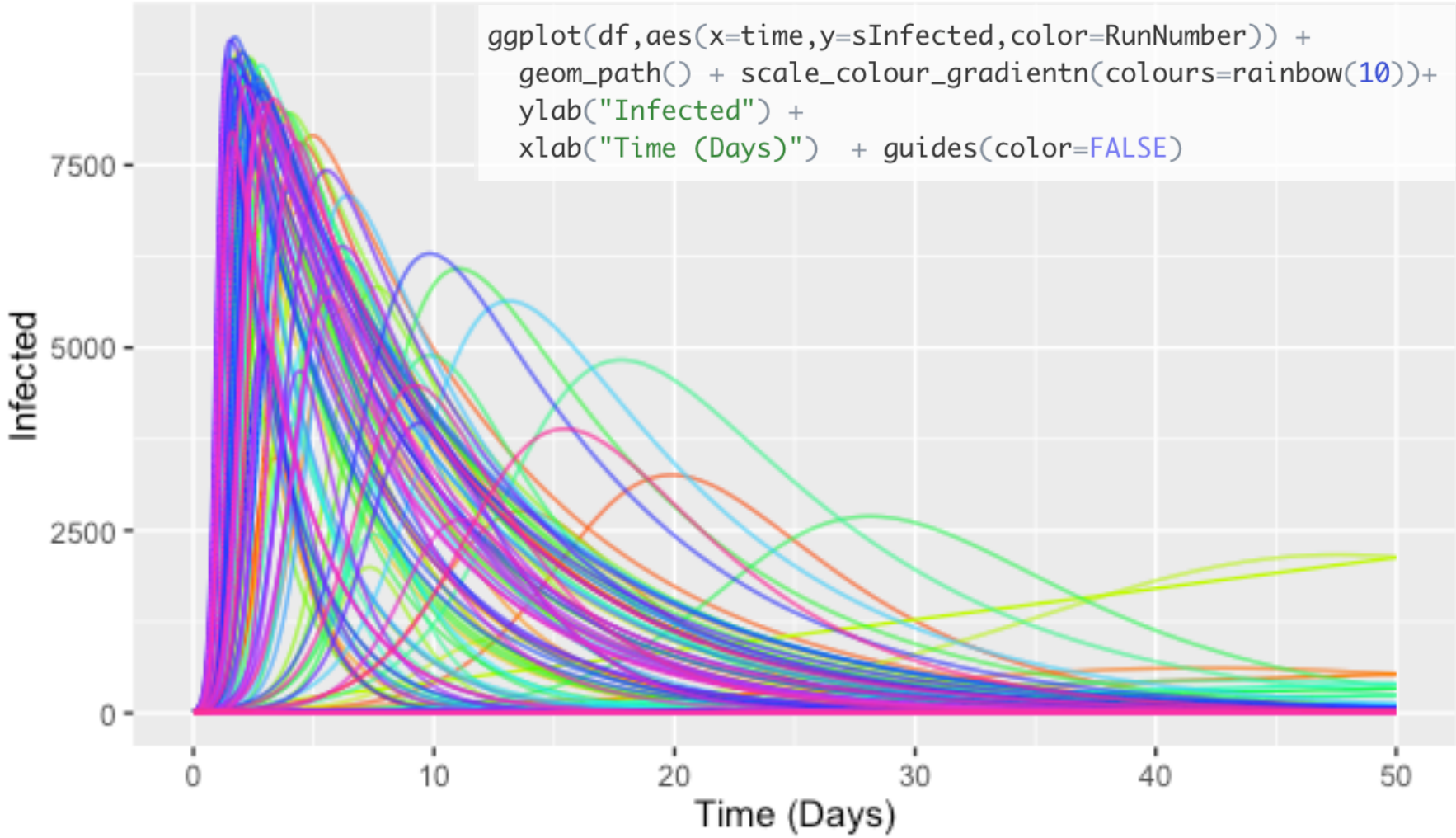
Initial output is a list of data frames: One for each simulation

```
> str(out[[1]])
'data.frame':   5001 obs. of  12 variables:
 $ time        : num  0 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 ...
 $ sSusceptible: num  9991 9991 9991 9990 9990 ...
 $ sInfected   : num  8.65 8.99 9.34 9.71 10.09 ...
 $ sRecovered  : num  0 0.0103 0.021 0.0321 0.0436 ...
 $ IR          : num  35.1 36.4 37.9 39.4 40.9 ...
 $ RR          : num  1.03 1.07 1.11 1.15 1.2 ...
 $ Beta        : num  0.000406 0.000406 0.000406 0.000406 0.000406 ...
 $ Lambda      : num  0.00351 0.00365 0.00379 0.00394 0.00409 ...
 $ DEL         : num  8.41 8.41 8.41 8.41 8.41 ...
 $ CE          : num  4.06 4.06 4.06 4.06 4.06 ...
 $ InitI       : num  8.65 8.65 8.65 8.65 8.65 ...
 $ RunNumber   : num  1 1 1 1 1 1 1 1 1 1 ...
```

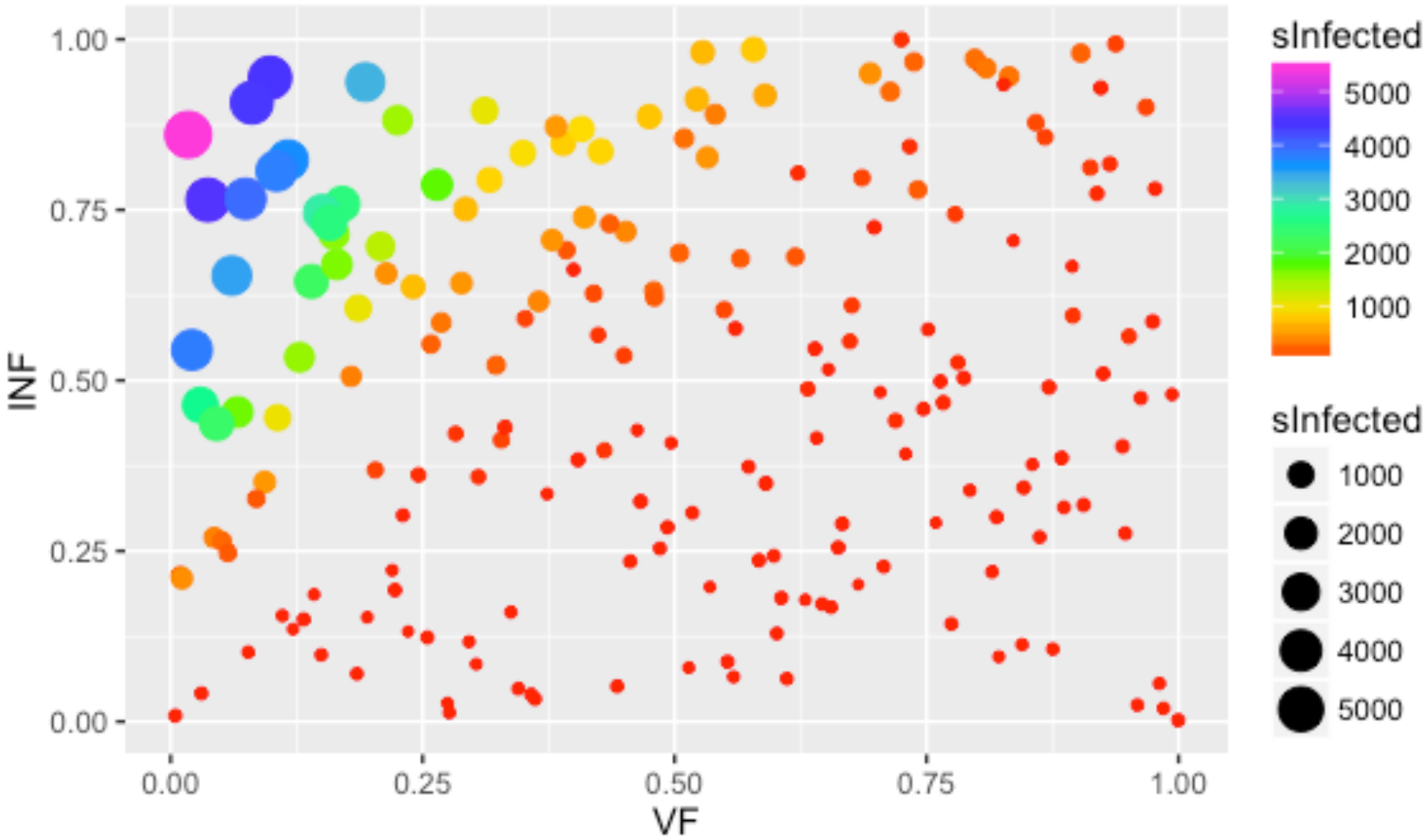
Converted to a single data frame

```
> df<-rbind.fill(out)
>
>
> str(df)
'data.frame':   500100 obs. of  12 variables:
 $ time        : num  0 0.01 0.02 0.03 0.04 0.05 0.06 0.07 0.08 0.09 ...
 $ sSusceptible: num  9991 9991 9991 9990 9990 ...
 $ sInfected   : num  8.65 8.99 9.34 9.71 10.09 ...
 $ sRecovered  : num  0 0.0103 0.021 0.0321 0.0436 ...
 $ IR          : num  35.1 36.4 37.9 39.4 40.9 ...
 $ RR          : num  1.03 1.07 1.11 1.15 1.2 ...
 $ Beta        : num  0.000406 0.000406 0.000406 0.000406 0.000406 ...
 $ Lambda      : num  0.00351 0.00365 0.00379 0.00394 0.00409 ...
 $ DEL         : num  8.41 8.41 8.41 8.41 8.41 ...
 $ CE          : num  4.06 4.06 4.06 4.06 4.06 ...
 $ InitI       : num  8.65 8.65 8.65 8.65 8.65 ...
 $ RunNumber   : num  1 1 1 1 1 1 1 1 1 1 ...
```

Visualise Simulations



Assignment 2: Exploratory Model Analysis



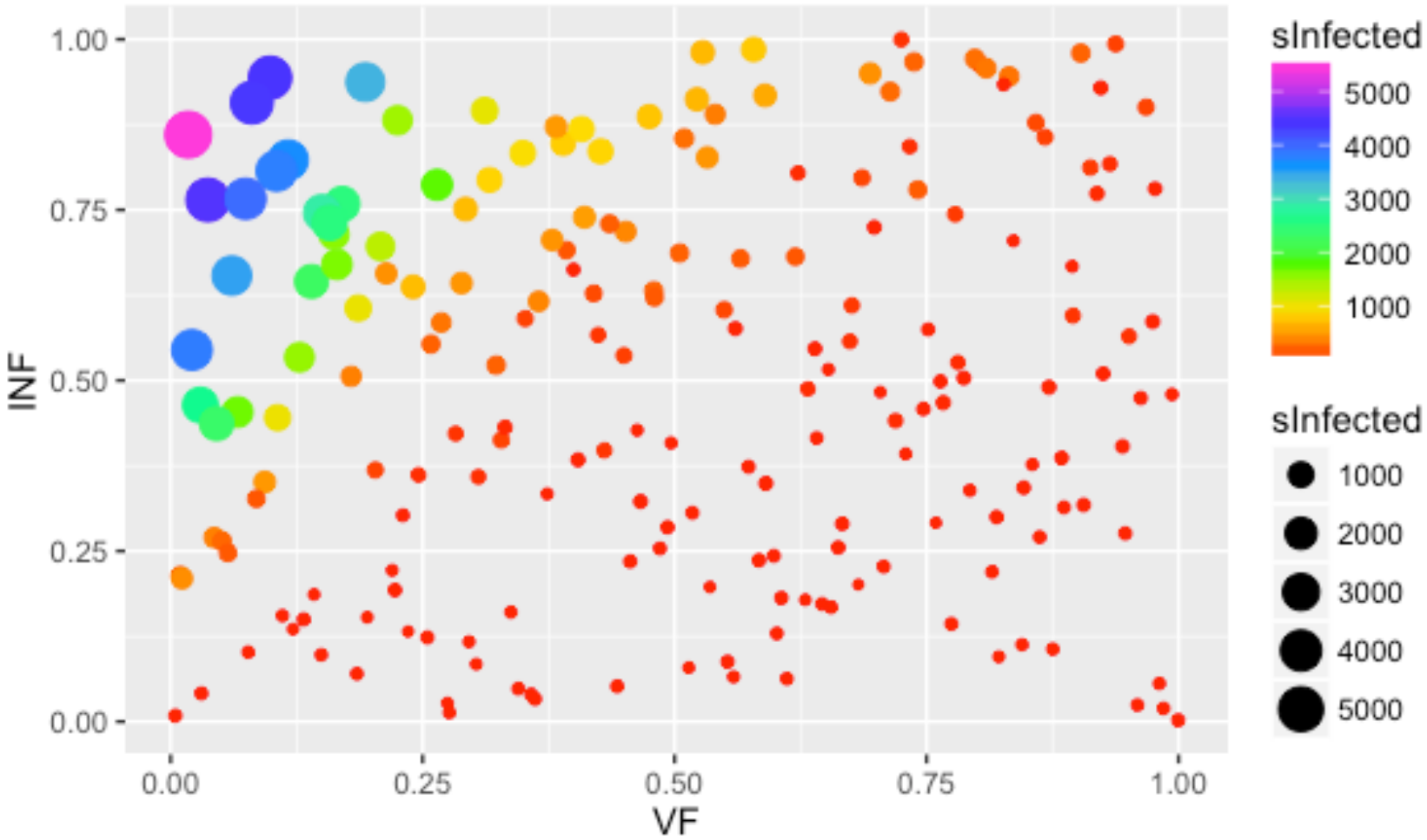
Assignment Information

- Basic SIR Model (N = 10000, Delay = 2)
- Need to add a vaccination flow, and a vaccination fraction [0,1] controlling the flow (no new stocks needed)
- Effective contact rate divided into two elements
  - Contact Rate = 4
  - Infectivity [0,1] probability of passing on infection given contact
  - Effective Contact = Contact Rate * Infectivity
- Run sensitivity for 1000 runs and plot VF v INF for ***Peak Infected Value*** in simulation run.
- set.seed(1234), STEP = 0.01, START = 0, FINISH = 20
- Comment on the results
- Due in Week 1, Semester II.

Sample parameter data (with seed, results should be the same)

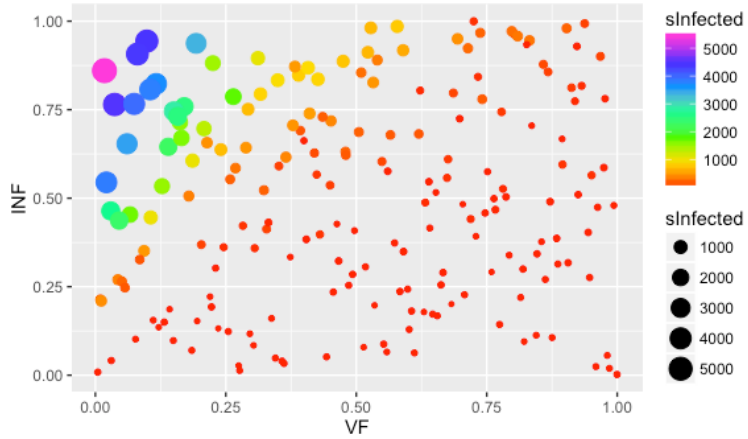
```
> parRange
              min max
aInfectivity    0   1
aVaccFraction   0   1
initInfected    1  25
>
> head(p)
  RunNo aInfectivity aVaccFraction initInfected
1     1   0.03330377     0.3614049    24.500516
2     2   0.31764180     0.9058709    21.698496
3     3   0.36158747     0.2458511    11.693240
4     4   0.92883928     0.9228027     4.162164
5     5   0.48763154     0.6321440    16.931328
6     6   0.50366151     0.7869459    13.646760
```

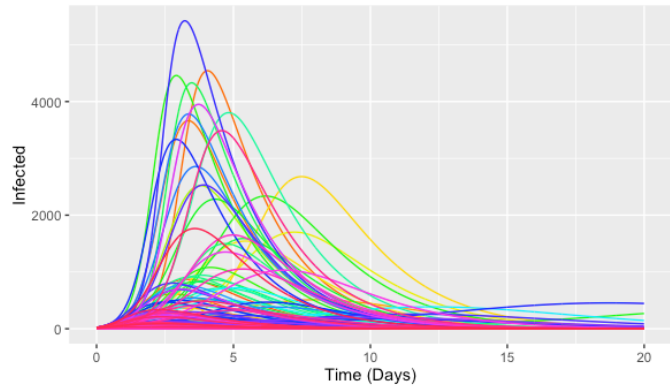
Aim: Produce the following graph for 200 simulations



Data check (first 10 for max infections)

```
> ans[1:10,]
   sInfected        VF        INF
1  24.500516 0.3614049 0.03330377
2  30.314068 0.9058709 0.31764180
3  61.094769 0.2458511 0.36158747
4  44.231923 0.9228027 0.92883928
5  56.223090 0.6321440 0.48763154
6  38.195783 0.7869459 0.50366151
7  36.882682 0.6392185 0.54653833
8   7.852881 0.2960440 0.11702087
9  19.663710 0.5175471 0.30602201
10 25.002103 0.1319748 0.14992817
```





Exam Question (A2015) – part(a)

2. (a) Suppose we have a town with 100,000 (=N) individuals, of which 1% were infectious with influenza, with R_0 = 2 and D=2 days (where D is the recovery delay)

- Calculate the force of infection λ
- Calculate the Herd Immunity Threshold
- Plot the Herd Immunity Threshold where the values of R_0 range from 1 to 10 (in steps of 1)

(6)

Exam Question (A2015) – part(b)

(b) Design a stock and flow model (with equations) to simulate the spread of seasonal influenza. Assume that the value for R_0 is 2, and that the average recovery delay is 2 days. The model should have the following features:

- Its core structure should be based on the *Susceptible – Exposed - Infected – Recovered* model.
- Exposed people are not infectious, but become infections after a first order delay of 5 days. The inflow into the *Exposed* stock is the *infection rate*.
- 1% of people who are infected become hospitalised for a period of 10 days, before recovering. This feature must be captured in the stock and flow model.

(8)

Exam Question (A2015) – part(c)

(c) Show how the model would need to be modified in order to represent different cohorts, for example, males and females in the population, and young, adults and elderly.

(6)