CHIME Interactive

Anshul Saxena and Gabriel Odom

4/4/2020

## Setup

We need to load some R packages before we can perform our computing.

library(reticulate)  
library(tidyverse)

The shell script to install the packages we need is in src/install\_python\_packages\_20200403.sh. This creates a conda environment called env\_sflCHIME and activates it. At current, I have been unable to run these commands on Windows.

On Mac, I have a Conda environment created for this project. In the terminal, I type which python and it tells me

which python

/Users/gabrielodom/anaconda3/envs/env\_sflCHIME/bin/python

If it says

/usr/bin/python

(or some other place), then something is wrong.

I specifically want this python environment, so I “point” reticulate to the particular environment that we installed with Anaconda, replete with the version of Python we selected and all associated libraries. (**NOTE**: this line must be executed before any other reticulate functions or Python calls, otherwise it will have no effect.)

use\_python(  
 "/Users/gabrielodom/anaconda3/envs/env\_sflCHIME/bin/python",  
 required = TRUE  
)

We also have some Python libraries to load.

from functools import reduce  
from typing import Tuple, Dict, Any  
import pandas as pd  
import streamlit as st  
import numpy as np  
import altair as alt

IT WORKS!!!!!! (This seriously took a full weekend to figure out, and then I called in the big guns: Tim N. and Athina H. Note that this is not compatible with environments created via Anaconda, at least not by default.)

## Parameters

Now, we want to set our initial parameters. We organise them the same way as on the CHARM app.

### Hospital Parameters

# Regional Population  
S\_default = 2761581.0  
  
# Hospital Market Share  
Penn\_market\_share = 0.15  
  
# Currently Hospitalised Patients (at current hospital?)  
known\_cases = 400.0  
current\_hosp = known\_cases  
  
# Currently Hospitalised Patients  
known\_infections = 1000.0  
initial\_infections = known\_infections

### Spread and Contact Parameters

# Doubling Time  
doubling\_time = 6.0  
  
# Social Distancing (% reduction in social contact)  
relative\_contact\_rate = 0.0

### Severity Rates

# Hospitalisation (% of total infections)  
hosp\_rate = 0.05  
  
# ICU (% of total infections)  
icu\_rate = 0.02  
  
# Ventilated (% of total infections)  
vent\_rate = 0.01

### Severity Times

# Infectious Days  
recovery\_days = 14.0  
  
# Average hospital length of stay (days)  
hosp\_los = 7.0  
  
# Average ICU length of stay (days)  
icu\_los = 9.0  
  
# Average length of stay on a ventilator (days)  
vent\_los = 10.0

## Directly Estimated Parameters

Given the parameters above, we calculate the following disease spread parameters.

### SIR Initial Values

total\_infections = current\_hosp / Penn\_market\_share / hosp\_rate  
print(total\_infections)

## 53333.333333333336

detection\_prob = initial\_infections / total\_infections  
print(detection\_prob)

## 0.01875

Infct = initial\_infections / detection\_prob  
Susc, Infct, Recov = S\_default - Infct, Infct, 0.0  
print(Susc)

## 2708247.6666666665

print(Infct)  
# print(Recov)  
# type(Recov)

## 53333.333333333336

intrinsic\_growth\_rate = 2 \*\* (1 / doubling\_time) - 1  
print(intrinsic\_growth\_rate)

## 0.12246204830937302

**NOTE**: the total infections and initial infections are equal at .

### SIR Transition Rates

# type(Recov)  
# print(Recov)  
  
gamma = 1.0 / recovery\_days  
# mean recovery rate, gamma, (in 1/days).  
print(gamma)

## 0.07142857142857142

beta = (intrinsic\_growth\_rate + gamma) / Susc \* (1.0 - relative\_contact\_rate)  
# Contact rate, beta  
# {rate based on doubling time} / {initial Susc}  
print(beta)

## 7.15926472029741e-08

r\_t = beta / gamma \* Susc   
# Current Reproduction Number  
# r\_t is r\_0 after distancing  
print(r\_t)

## 2.7144686763312222

r\_naught = r\_t / (1.0 - relative\_contact\_rate)  
# Initial Reproduction Number  
print(r\_naught)

## 2.7144686763312222

doubling\_time\_t = 1.0 / np.log2(beta \* Susc - gamma + 1.0)   
# Current Doubling Time  
# doubling time after distancing  
print(doubling\_time\_t)  
  
# type(Recov)

## 5.999999999999998

As an aside, if I want to access any of these values in R (for instance, ), I can call them as py$r\_naught, which yields 2.7144687.

## Model Specification

We can now specify the SIR model at outset and over time.

# type(Recov)  
  
# SIR at Outset  
def sir(y, beta, gamma, N):  
 S, I, R = y  
 Sn = (-beta \* S \* I) + S  
 In = (beta \* S \* I - gamma \* I) + I  
 Rn = gamma \* I + R  
 if Sn < 0:  
 Sn = 0.0  
 if In < 0:  
 In = 0.0  
 if Rn < 0:  
 Rn = 0.0  
  
 scale = N / (Sn + In + Rn)  
 return Sn \* scale, In \* scale, Rn \* scale  
 return S, I, R  
   
# type(Recov)  
  
# Dynamic SIR   
def sim\_sir(S, I, R, beta, gamma, n\_days, beta\_decay = None):  
 N = S + I + R  
 s, i, r = [S], [I], [R]  
 for day in range(n\_days):  
 y = S, I, R  
 S, I, R = sir(y, beta, gamma, N)  
 if beta\_decay:  
 beta = beta \* (1 - beta\_decay)  
 s.append(S)  
 i.append(I)  
 r.append(R)  
   
 s, i, r = np.array(s), np.array(i), np.array(r)  
 return s, i, r  
  
# type(Recov)  
  
# Test Function  
def test\_fun(x):  
 y = x + 2  
 return y  
   
# type(Recov)

Run the model, assuming no decay in contact rate over the next 30 days.

n\_days = 100  
beta\_decay = 0.0  
  
test\_fun(3)  
# type(Susc)  
# type(Infct)  
# type(Recov)

## 5

y = Susc, Infct, Recov  
# print(y)  
# type(y)  
N = Susc + Infct + Recov  
print(N)

## 2761581.0

type(N)

## <class 'float'>

sir(y, beta, gamma, N)

## (2697906.8336139764, 59864.6425764999, 3809.5238095238096)

s, i, r = sim\_sir(  
 Susc, Infct, Recov, beta, gamma, n\_days, beta\_decay = beta\_decay  
)  
print(s)

## [2708247.66666667 2697906.83361398 2686343.96041234 2673429.22349672  
## 2659022.7679715 2642974.92642874 2625126.76021732 2605311.00315433  
## 2583353.49520194 2559075.19804396 2532294.88389722 2502832.58098789  
## 2470513.84129663 2435174.86571532 2396668.47621179 2354870.86233679  
## 2309688.95034253 2261068.14962702 2209000.12873143 2153530.17112599  
## 2094763.57292077 2032870.48703665 1968088.60919197 1900723.15644506  
## 1831143.71933025 1759777.77419417 1687100.91032388 1613624.12985131  
## 1539878.87846321 1466400.71711853 1393712.70765715 1322309.62896367  
## 1252644.05532488 1185115.1273455 1120060.56077308 1057752.11497791  
## 998394.42846001 942126.864422 889027.8220667 839120.86916188  
## 792382.03366 748747.64090821 708122.17682868 670385.77446277  
## 635401.0426362 603019.06743142 573084.51130313 545439.80748248  
## 519928.49882366 496397.80278953 474700.50134251 454696.25989664  
## 436252.47673555 419244.75640269 403557.08986421 389081.81244248  
## 375719.39877742 363378.14311658 351973.76345518 341428.95960563  
## 331672.94818246 322640.99165775 314273.93394389 306517.7512394  
## 299323.12397744 292645.03349948 286442.38541044 280677.66034357  
## 275316.59198092 270327.87155937 265682.87768175 261355.42999538  
## 257321.56515839 253559.33345542 250048.61442472 246770.94990116  
## 243709.39294857 240848.37124165 238173.56355363 235671.788106  
## 233330.90163688 231139.70814234 229087.87633872 227165.86498185  
## 225364.85526161 223676.68956589 222093.81597812 220609.23793595  
## 219216.46853673 217909.48902761 216682.71106579 215530.94237671  
## 214449.35547655 213433.45915973 212479.07248302 211582.30100553  
## 210739.51506843 209947.32992066 209202.58751648 208502.33982856  
## 207843.83353594]

**ERROR**: Error in py\_call\_impl(callable, dots$args, dots$keywords) : TypeError: unsupported operand type(s) for +: 'float' and 'type'  
**SOLUTION**: I can’t have a Python object named R.

We now inspect the results of this model

res\_df <- tibble(  
 days = seq\_len(py$n\_days + 1),  
 Susc = py$s,  
 Infct = py$i,  
 Recov = py$r  
)  
  
ggplot(data = res\_df) +  
 aes(x = days) +  
 geom\_line(aes(y = Susc), colour = "black") +   
 geom\_line(aes(y = Infct), colour = "red") +   
 geom\_line(aes(y = Recov), colour = "green")

