A PYTHON 3 SCRIPT THAT CAN BE USED TO FIT A THEORETICAL CURVE TO THE OBSERVED CURVE OF AN INFECTIOUS DISEASE OUTBREAK SUCH AS COVID-19

THEORY BEHIND THE MODELLING USED HERE

This procedure is best described as a rule based stochastic modelling fit. The concept is that an infected individual can only infect a fixed proportion of uninfected people that they come into contact with. (This proportion is signified in the program by the variable P. So a value of P=2 means that the infected person can infect only 2 other uninfected people.) These infected and un-infected people exist in a defined space measuring 100 people by 100 people i.e. the simulation takes place within a dimensionally defined space of 10,000 people. Interactions with uninfected people occur randomly through out the space so for example an infected person at the coordinates 10,20 can infect an uninfected person at coordinates 20,10. But if that person has already been infected then that 'opportunity for propagation of the infection' is missed. The program picks the coordinates of the person to be infected next entirely at random; this gives the model a two dimensional spacial aspect of infected people moving randomly through a group of 10,000 people and infecting them if they have not already been infected. So at first the random numbers hit upon uninfected persons with a high degree of 'success' so the curve of cases per day rises steeply. Then as the simulation progresses the probability of an infected person encountering an uninfected person falls and the cases per day curve peaks and then starts to descline.

The Script :

The script is written in Python 3 using the Thonny IDE. However it is best run in the Linux Terminal and not in an IDE; it runs faster that way.. (Note that multiline comments are bracketed by ''' and single line comments are preceded by a # character). The comments are placed strategically so that an explanation is available for the subsequent lines of the script.)

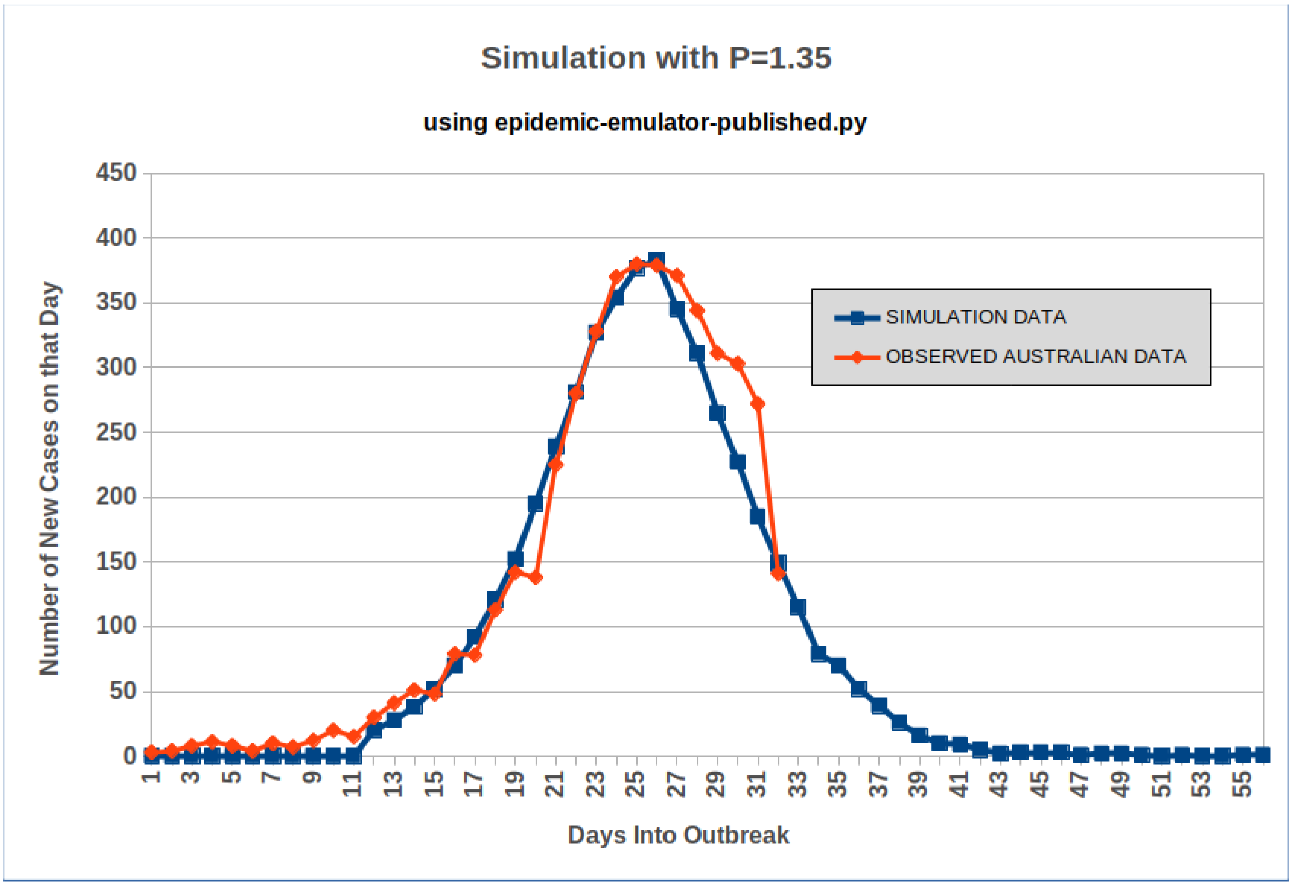
```
print ('Epidemic Infection Propagation Simulation')
print('Tom Hartley : Version 1.00 : April 2020')
print('email medlabstats@iinet.net.au')

import random

N=100
```

```
RUNNINGTOTAL = 0

# Build an empty array 100 by 100 full of zeros --- represents a population of
10,000 persons
a = [[ 0 for i in range(N)] for j in range(N)]

# To print out the whole array uncomment the next line
#print(a);

# To print the top left item in the array uncomment the next line
#print(a[0][0])

# To print the bottom right item in the array uncomment the next line
#print (a[99][99])

# To print the first row of the array uncomment the next line
#print(a[0])
print
('-----------------------------------------------------------------------------')

'''
Assign a value to PR which represents how many primary cases you with which you
wish to ‘infect’ the population of 10,000 people. If you want fine detail eg.
something close to daily counts then set this to a low value. If you are dealing
with a situation where the counts are gathered as weekly, fortnightly or monthly
counts as in a slowing propagating epidemic set this larger.
'''
PR = 20

'''
CYCLE 0 IS THE PRIMARY INFECTION CYCLE
Infect the array with PR primary cases signified by a '1'
Because this is random some rows will have no 1's others may have more than one
1. Also if a cell has already got an infected 1 in it then no additional 1 will
be added to the array so sometimes the starting infected rate may be less than
(PR/100)%
'''

cycle = 0
i=0
j=0
infected = [0] * 50
infected[cycle] = 0

'''
This WHILE loop picks x and y coordinates in the array at random and puts a 1
into that cell - but only if it is currently containing a 0.
'''

while i<PR:
    k = random.randint(0,99)
    j= random.randint(0,99)
    if a[k][j] == 0:
        a[k][j] =1
        infected[cycle] = infected[cycle] + 1
    # print ('      ', i, j)
    # print(a[i])
    i = i+1

print
('-------------------------------------------------------------------------------')

print('NUMBER INFECTED AT DAY ZERO  = ', infected[cycle])
RUNNINGTOTAL = RUNNINGTOTAL + infected[cycle]
print('Running Total = ', RUNNINGTOTAL)

#--------------------------------------------------------------------------------
’’’
INFECTION CYCLES  - For every primary infected person P more get the disease.
These secondary cases signified by the current value of 'cycle'. Because this is
random some rows will have no 1's others may have more than one 1. Also if a
cell has already got an infected ‘value’ in it then no additional value will be
added to the array so sometimes the secondary array infection rate will be less
than P. P is the propagation factor for this simulation. It can be changed
before each run. Similarly the number of DAYS that the simulation runs for can
be adjusted as required
’’’

DAYS = 45
m = 1

# Best fit to Australian Covid-19 data has P=1.35
P=1.35

’’’
This WHILE loop runs for as many times as set in DAYS and it progressively tries
to place the current value of ’cycle’ into randomly selected positions in the
array that are currently set at 0. So if it finds a value not equal to 0 then it
misses out on assigning a value there (this is accommodates the fact that you
cannot be infected more than once)
’’’

while m<=DAYS:

    i=0
    j=0
    q=infected[cycle]
    cycle = cycle + 1
    infected[cycle] = 0

    while i<= (P*q):
        k = random.randint(0,99)
        j = random.randint(0,99)
        if a[k][j] == 0 :
            a[k][j] = cycle
            infected[cycle] = infected[cycle] + 1

        i = i+1
    print('NUMBER INFECTED BY DAY NUMBER ', cycle, " = ", infected[cycle])
    RUNNINGTOTAL = RUNNINGTOTAL + infected[cycle]
    print('Running Total = ', RUNNINGTOTAL)
    m = m + 1
print
('----------------------------------------------------------------------------')

'''
The raw data of the simulation are saved into an Excel compatible .csv file in
the same directory as this script is located. You can open that in Excel or
LibreCalc to do the curve plotting and comparisons with actual epidemic data.
'''
t = open("data.csv", "w")
i = 0

while i<DAYS:
    t.write(str(infected[i]))
    t.write("\n")
    i = i + 1

t.close()

'''
If you want to see the whole array and how the cycle numbers have gone into the
array then uncomment the next line
'''
# print(a)
```

EVALUATION OF THE PERFORMANCE OF THE MODELLING

(1) THE FIT TO OBSERVED AUSTRALIAN DATA AS AT 3rd May 2020.

FIGURE 1.



The actual data used in this fit are shown in Table 1. Day 1 = 1/3/2020

TABLE 1

SIMULATION DATA	OBSERVED AUSTRALIAN DATA
0	3
0	4
0	8
0	11
0	8
0	4
0	10
0	7
0	12

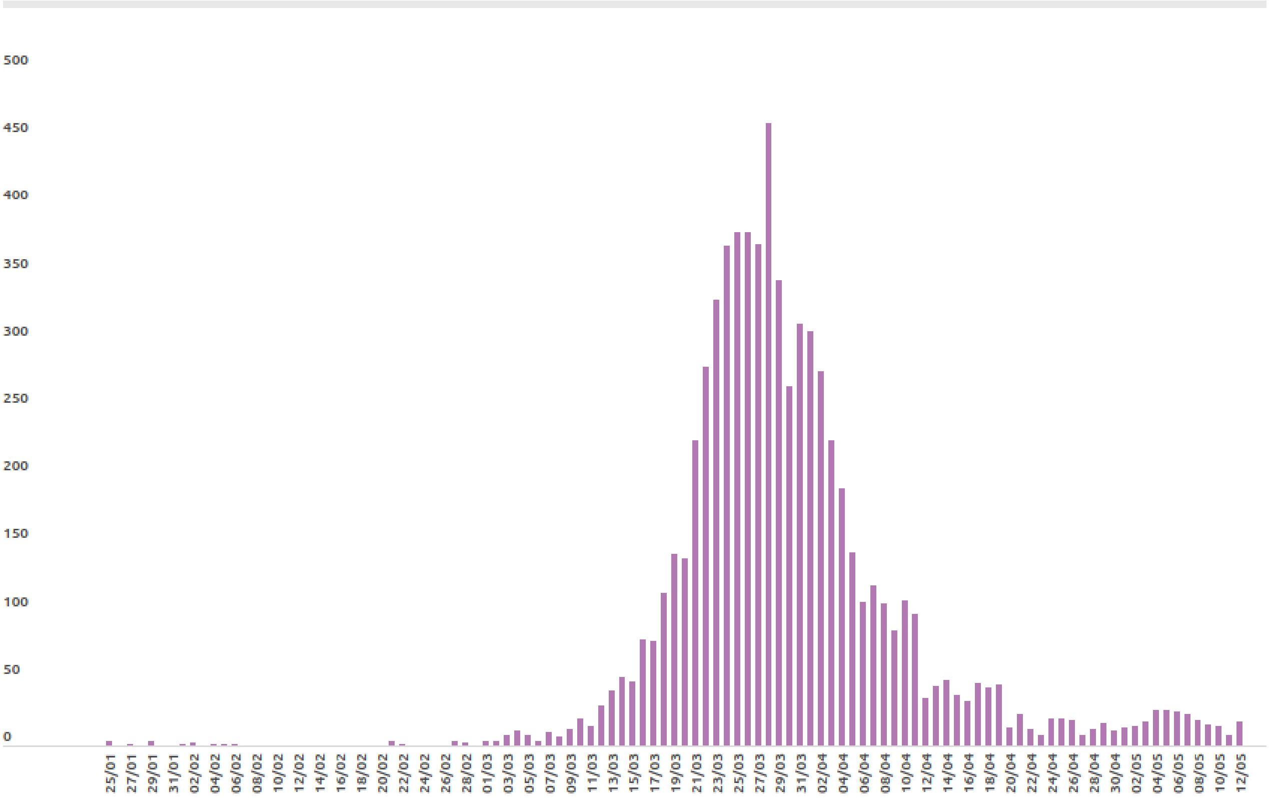
0	20
0	15
20	30
28	41
38	51
52	48
70	79
92	78
121	113
152	142
195	138
239	225
281	280
327	328
354	370
377	380
383	379
345	371
311	344
265	311
227	303
185	272
149	141
115	
79	
70	
52	
39	
26	
16	
10	
9	
5	
2	
3	
3	
3	
1	
2	

2	
1	
0	
1	
0	
0	
1	
1	

In my opinion the Australian data had two outlier events … the count on the 27/3/20 was abnormally high at 460 and was followed two days later on the 30/3/20 by an abnormally low count of 266 (see www.covid19data.com.au for the raw data histogram … https://infogram.com/1p7ve7kjeld1pebz2nm0vpqv7nsnp92jn2x shown below, Figure 2). These two points have been left out; the reason being that there was very probably a reporting error on those two days – it was almost as though there was an over reporting on the 27/3/20 which was 'corrected' by an under reporting on the 29/3/20.)

FIGURE 2.

New daily confirmed COVID-19 cases in Australia (since first case)



The simulation program produces counts immediately but as can be seen from the curve in Figure 1 there was a long lead up before the counts became significantly visible above the baseline. So the simulation data were ‘time shifted’ by 11 days (to 12/3/20) at which point the simulated data curve was an excellent overlay on the actual data curve. In Table 1 this time shift can be seen as the highlighted ‘zeros’.

Visually the fit is not as good on the declining side of the observed curve and this can be expected. As the epidemic progressed it moved into a ‘new clusters phase’ caused by events where members of an infected cluster moved into ‘new’ discrete uninfected populations such as when the Diamond Princess cruise ship passengers were allowed to move unchecked into the general population. In the general population there were also small rapid clusters developing such as in NW Tasmania. These counts got incorporated into the tail of the observed data and consequently distorted it. The important area of fit was in the rising portion of the curve where the epidemic was ‘allowed’ to progress unchecked because there were really no effective national control measures in play. Social distancing was not really enacted until 23/3/20 when the daily case count was 330 a figure very close to the peak of 380 on that was reached on the 25/3/20.

Finally what does the P=1.35 mean ? It means that a group of 100 infected people can infect 135 others. In my modelling the measurement interval is a day whereas epidemiologists use the term R_0 (which we read as R Zero) which they measure across a ‘serial interval’ which for Covid 19 has been reported to be between 2.3 to 3.9.

This extract from the paper by Majumder and Mandl

(https://papers.ssrn.com/sol3/Delivery.cfm/SSRN_ID3525949_code3251439.pdf?abstractid=3524675&mirid=1&type=2)

provides a good explanation of R_0 and the type equation that epidemiologist use in their non-stochastic modelling :

> *“The model itself can be defined by the following single equation:*

$$I = \left[\frac{R_0}{(1+d)^t}\right]^t$$

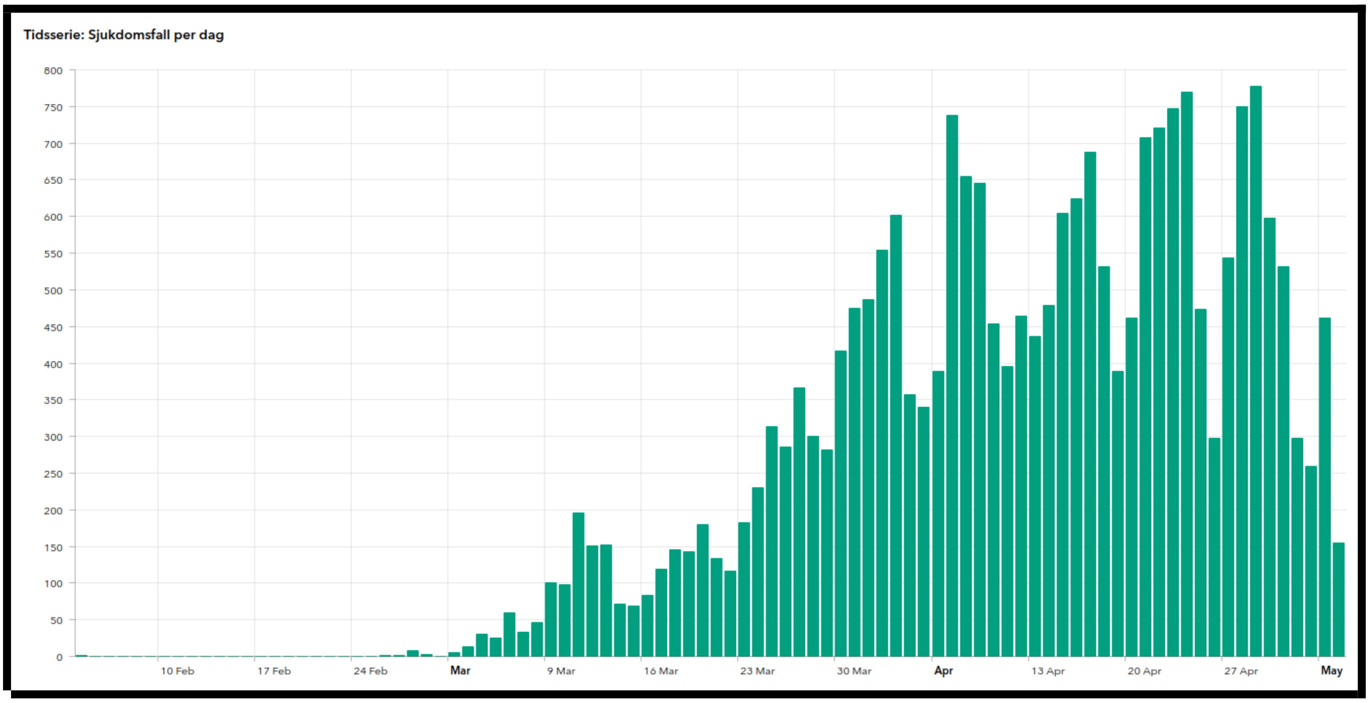
> *Here, t is the number of serial intervals that have passed at time of model parameterization and I is incidence at serial interval t. Meanwhile, d is a discount factor that takes into account reductions in transmissibility over time due to the natural depletion of susceptible individuals in the affected population and any public interventions that may impact disease spread over time. Because the serial interval associated with 2019–nCoV has not yet been established, we referenced mean serial interval lengths (l) from the related SARS-Coronavirus and MERS-Coronavirus [range: 6–10 days] to parameterize our model. For this range of serial interval lengths, modeled R_0 estimates varied from 2.0 to 3.1 when using data from December 8, 2019 through January 26, 2020. Estimates for d were 0 for all serial interval lengths."*

The report by Imperial College London Covid-19 Response Team (Natsuko Imai, Anne Cori, Ilaria Dorigatti, Marc Baguelin, Christl A. Donnelly, Steven Riley, Neil M. Ferguson. Transmissibility of 2019-nCoV. Imperial College London (25-01-2020), doi: https://doi.org/10.25561/77148.) shows that as you reduce the serial interval length then the value of R_0 falls. Their reported range for R_0 was 1.5 to 3.5 which put my simulation value of 1.35 in the right vicinity for a shorter interval range. I was reluctant to resort to reducing the granularity of the Australian data to weekly means or even running means across the past seven days, however, as will be seen in the next simulated fit to the Swedish data, resorting to a 'running weekly mean' was essential if there was to be a good fit of the simulation !

(2) SIMULATED FITS TO THE SWEDISH OUTBREAK DATA :

The Swedish data are an interesting set mainly because the Swedes have taken a much more relaxed approach to control measures such as social distancing and as a community they appear to be more comfortable with their death rates. Their relaxed approach is apparent when you look at the histograms (Figure 3). Initially when I saw the repeated peaks and troughs I assumed that these were evidence of successive waves of infections. But then I noticed that all these troughs appeared at around weekends and I have highlighted them with yellow backgrounds! Clearly the opportunities for data collection and/or generation must have been lower at weekends. Two reasons come to mind – (i) maybe some testing centres closed at weekends or (ii) some testing labs closed at weekends. Either or a combination of both would affect the collection of cases per day data. In light of this observation some massaging of the data to smooth these aberrations seemed justified. The most obvious one was to take the moving averages of the previous seven days so as to reduce the 'weekend' effect.

FIGURE 3



Information taken from :

- **https://experience.arcgis.com/experience/09f821667ce64bf7be6f9f87457ed9aa**
- **https://www.worldometers.info/coronavirus/country/sweden/**

By hovering my cursor over this histogram I was able to retrieve data and case count for each element in this histogram; these data are shown in Table 2.

TABLE 2 : The Swedish Data as at 12th May 2020.

Day Number Day of Week		Date	Daily Cases Count	Running 7 Day Mean Daily Count	SIMULATED FIT TO 2nd Peak with P=1.21 and Population=160*160 =25,600	SIMULATED FIT TO 1st Peak with P=1.35 and Population=52*52 =2,704	SUM OF SIMULATED FITS TO 1st and 2nd PEAKS
1	Tuesday	2/3/20	5	0	0	0	0
2	Wed	3/3/20	13	0	0	0	0
3	Thurs	4/3/20	30	0	0	0	0
4	Fri	5/3/20	25	0	0	19	19
5	SAT	6/3/20	59	0	0	24	24
6	SUN	7/3/20	33	0	0	33	33
7	Mon	8/3/20	46	30	0	40	40
8	Tuesday	9/3/20	101	44	0	52	52
9	Wed	10/3/20	98	56	0	63	63
10	Thurs	11/3/20	196	80	0	76	76
11	Fri	12/3/20	151	98	0	85	85
12	SAT	13/3/20	152	111	0	101	101
13	SUN	14/3/20	71	116	0	112	112
14	Mon	15/3/20	69	120	20	115	135
15	Tuesday	16/3/20	83	117	25	112	137

16	Wed	17/3/20	119	120	31	99	130
17	Thurs	18/3/20	145	113	38	89	127
18	Fri	19/3/20	143	112	46	69	115
19	SAT	20/3/20	180	116	56	61	117
20	SUN	21/3/20	134	125	67	47	114
21	Mon	22/3/20	117	132	81	33	114
22	Tuesday	23/3/20	182	146	99	23	122
23	Wed	24/3/20	230	162	120	18	138
24	Thurs	25/3/20	314	186	143	15	158
25	Fri	26/3/20	286	206	166	10	176
26	SAT	27/3/20	366	233	194	9	203
27	SUN	28/3/20	300	256	228	7	235
28	Mon	29/3/20	281	280	265	5	270
29	Tuesday	30/3/20	416	313	301	4	305
30	Wed	31/3/20	475	348	336	5	341
31	Thurs	1/4/20	486	373	374	4	378
32	Fri	2/4/20	554	411	411	5	416
33	SAT	3/4/20	601	445	437	2	439
34	SUN	4/4/20	357	453	457	2	459
35	Mon	5/4/20	340	461	472	2	474
36	Tuesday	6/4/20	389	457	473	2	475
37	Wed	7/4/20	738	495	467	2	469
38	Thurs	8/4/20	655	519	448	0	448
39	Fri	9/4/20	645	532	418	0	418
40	SAT	10/4/20	454	511	374	1	375
41	SUN	11/4/20	395	517	322	1	323
42	Mon	12/4/20	464	534	281	2	283
43	Tuesday	13/4/20	437	541	244	1	245
44	Wed	14/4/20	480	504	213	1	214
45	Thurs	15/4/20	604	497	181	2	183
46	Fri	16/4/20	623	494	144	2	146
47	SAT	17/4/20	688	527	114	3	117
48	SUN	18/4/20	532	547	102	4	106
49	Mon	19/4/20	389	536	82		82
50	Tuesday	20/4/20	462	540	65		65
51	Wed	21/4/20	708	572	65		65
52	Thurs	22/4/20	722	589	54		54
53	Fri	23/4/20	753	608	45		45
54	SAT	24/4/20	777	620	33		33
55	SUN	25/4/20	474	612	29		29
56	Mon	26/4/20	300	599	28		28
57	Tuesday	27/4/20	547	612	18		18
58	Wed	28/4/20	726	614	15		15
59	Thurs	29/4/20	778	622	0		0
60	Fri	30/4/20	598	600	0		0
61	SAT	1/5/20	532	565	0		0
62	SUN	2/5/20	298	540	0		0
63	Mon	3/5/20	258	534	0		0
64	Tuesday	4/5/20	459	521	0		0
65	Wed	5/5/20	637	509	0		0
66	Thurs	6/5/20	730	502	0		0
67	Fri	7/5/20	751	524	0		0
68	SAT	8/5/20	686	546	0		0

69	SUN	9/5/20	509	576	0		0
70	Mon	10/5/20	279	579	0		0
71	Tuesday	11/5/20	410	572	0		0
72	Wed	12/5/20	213	511	0		0

Stochastic modelling requires you to make ‘intelligent’ guesses to get the modelling underway. Two things have to be tweeked until you get a reasonable fit.

The first is the population size (N=100 in the script. NB that the population size is actually N^2 so if you are thinking of a new population size to try then you need to set N to the square root of that number). The second is the propagation rate (P=1.35 in the script shown above).

The population size determines the area under the curve and the propagation rate determines the sharpness of the curve.

If you change N then you must also change these four lines in the program; NB that when N=x the number in these lines is dropped to x minus one (this is because in Python arrays the first item is indexed at 0 and not at 1.)

```
while i<PR:
    k = random.randint(0,99)
     j= random.randint(0,99)
.
.
.
.
.
    while i<= (P*q):
        k = random.randint(0,99)
          j = random.randint(0,99)
```

It is best to work on N first and use a ‘divided difference’ technique to arrive at your first best estimate. Start off with N=100 then if that is too small increase it. If that overshoots then try a value halfway between you first guess and you second guess. So a search may proceed as follows ..

N=100 … too small, try 120
N=120 … to large, try 110
N=110 … to small, try 115
N=115 pretty good so now work on fine tuning P

The same tactic is followed here …

P=1.35 … too wide, try 1.45
P=1.45 … too narrow, try 1.40

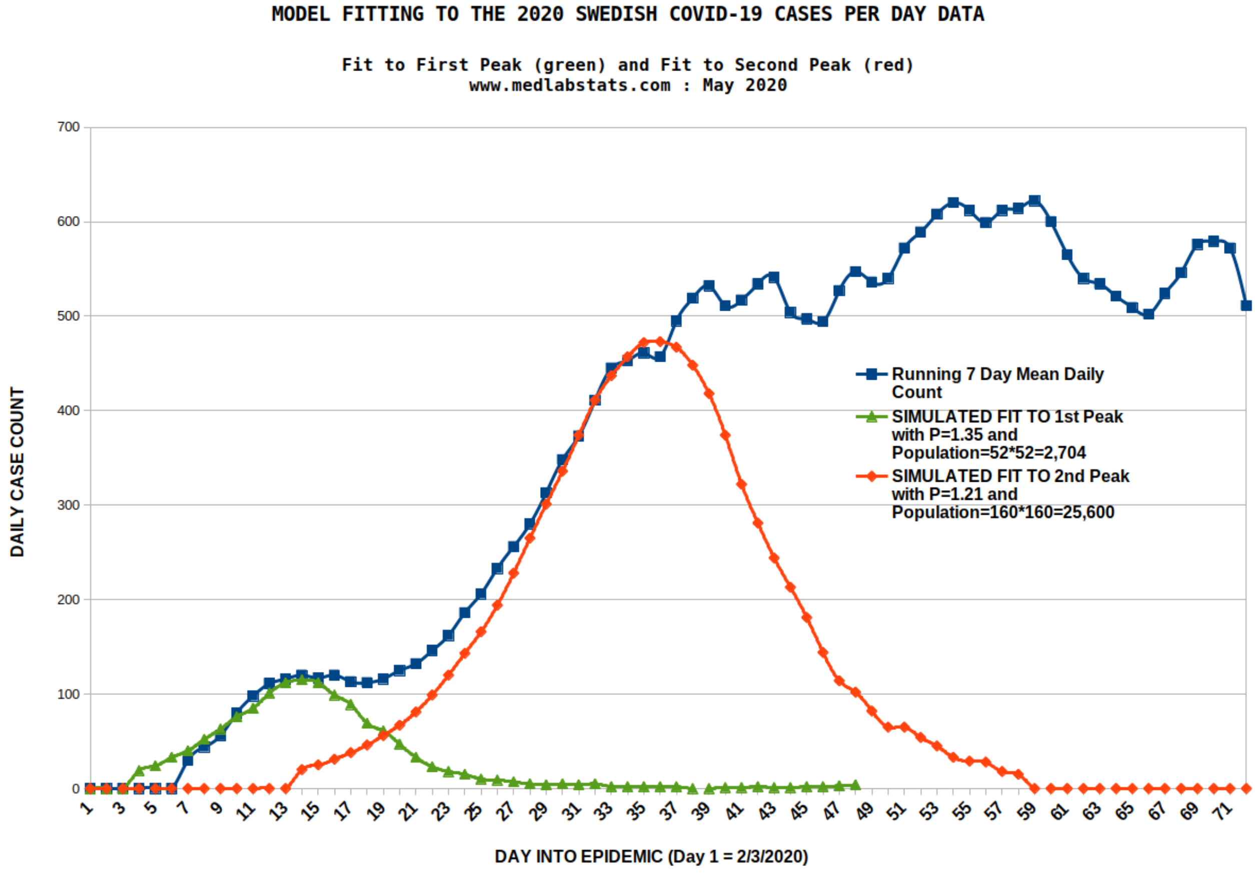
P=1.40 … too wide, try 1.425
and so on ..

Once you have the rising side of the fit visually acceptable then work on the offset in the x-direction. This is done by simply pasting a few zeros into a few cells before the point where you paste in the data from the latest run of the simulation script.

How do you visualise your fit to the observed epidemic data ? Plot the observed epidemic data as a time series line plot in Excel or similar spreadsheet. On the same spreadsheet page you paste in the latest data from your fitting into a separate column. The latest fitting data are written as a column of numbers by the Python script into a file called data.csv in the same directory on your computer as the Python script. Each time you paste over the new fit data the plot will automatically update so watch carefully to see in which direction the fit is moving …. better or worse …. and use that to dictate in which direction you alter the N or P.

Using these techniques I was able to fit two curves to the Swedish data. My result is shown in Figure 4. I fitted the second peak first (red) and then worked on the first peak (green).

FIGURE 4.

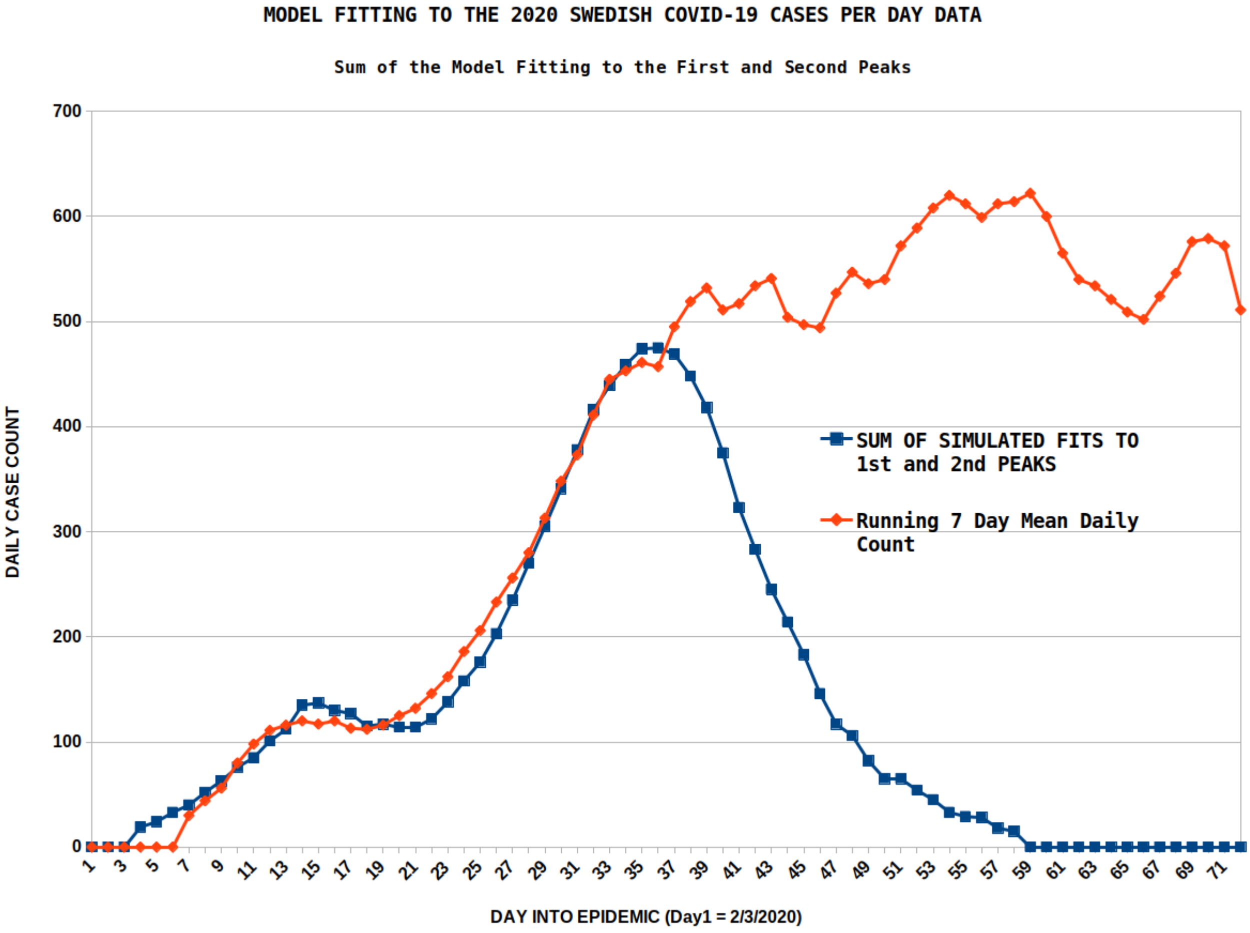


MODEL FITTING TO THE 2020 SWEDISH COVID-19 CASES PER DAY DATA

Fit to First Peak (green) and Fit to Second Peak (red)
www.medlabstats.com : May 2020

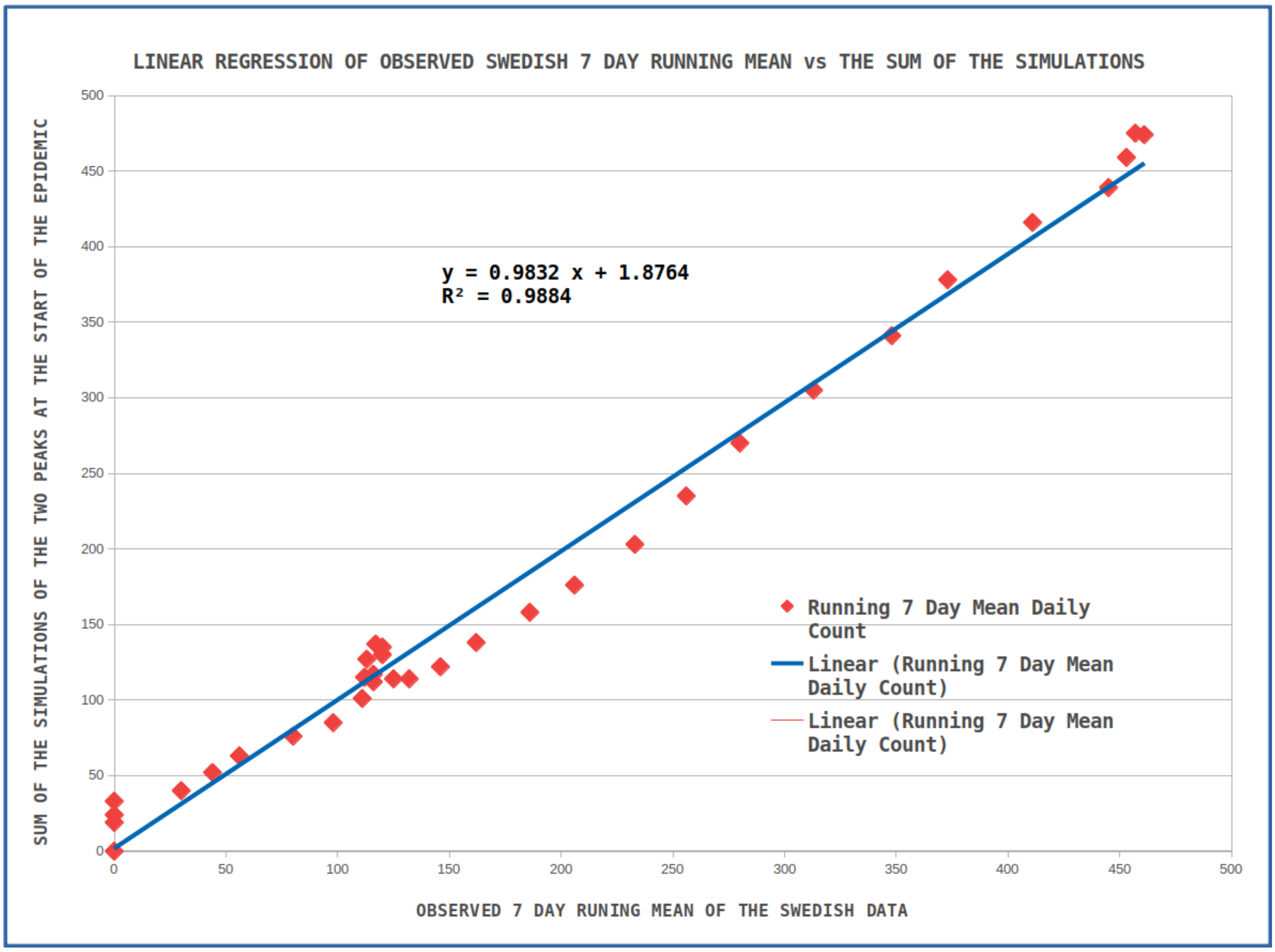
To evaluate the goodness of fit to the overall Swedish epidemic curve I summed my estimates for the first and second peaks and plotted that curve (blue) on the same scale as the actual data curve (red). See Figure 5. As with the Australian data fit I did not expect the modelling to fit the actual data curve once the epidemic had got underway viz. I did not expect the modelling to fit after the 8/4/2020.

FIGURE 5.

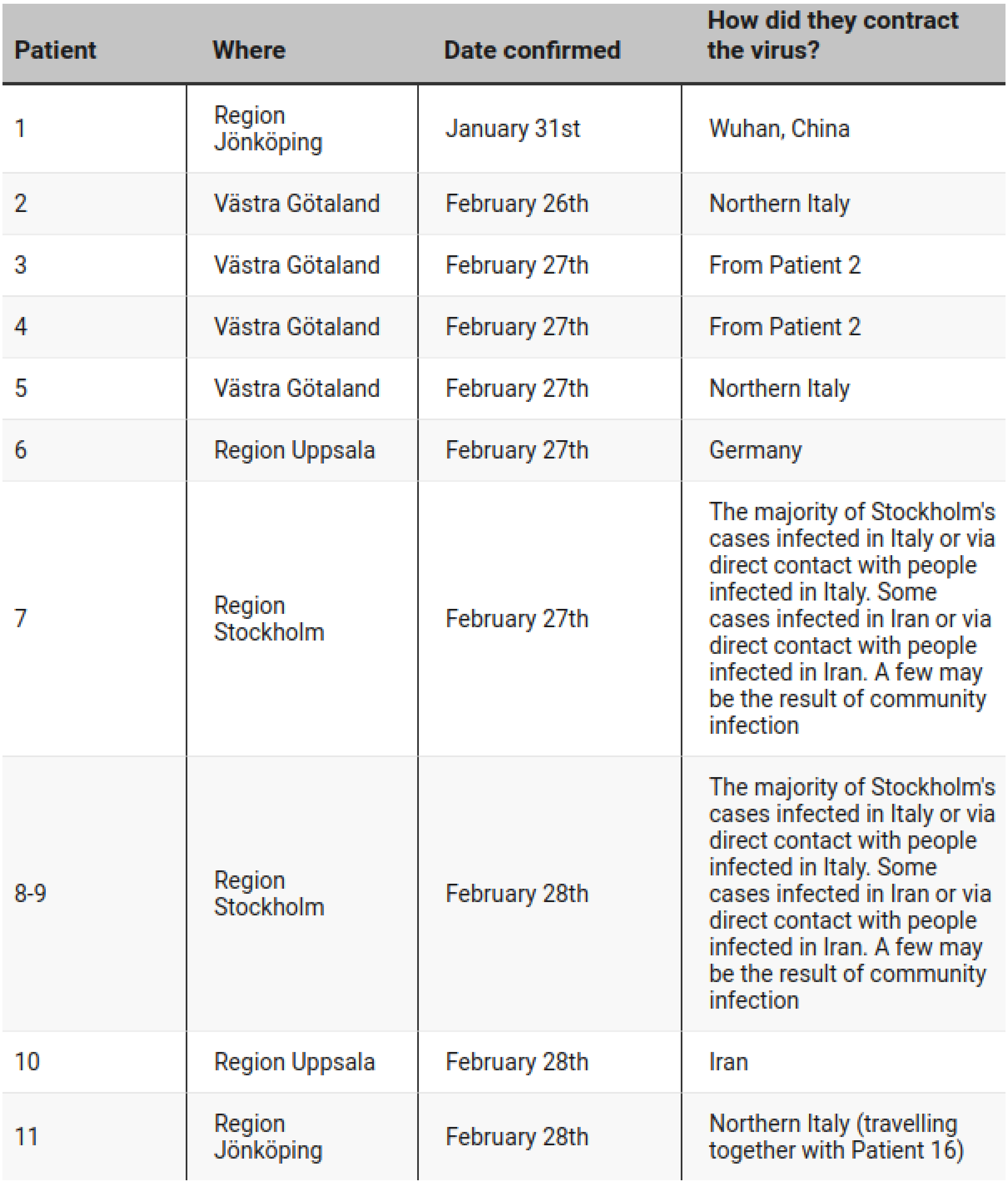


I then made a formal statistical evaluation of my goodness of fit by plotting the ‘Observed 7 Day Running Means’ on the x-axis and my summations from my modelling on the y-axis, Figure 6. In theory if there was perfect agreement between my modelling and the observed data then that graph should have a slope of one and an intercept of zero. In fact it had an intercept of 1.9 which statistically was different from zero. The slope was 0.98 which was statistically significant. It meant that my modelling was underestimating the real data by 2%. I considered this to be a more than adequate outcome from these experiments in rule based stochastic fitting.

FIGURE 6.



Quite why the Swedish data has two discernible peaks is not quite clear to me. It must have been something related to how the epidemic started in Sweden. Superficial investigations of this suggests that the outbreak started within Stockholm first and then spread to the wider community.
(see https://www.thelocal.se/20200303/swedens-first-coronavirus-patient-recovers)



Patient	Where	Date confirmed	How did they contract the virus?
1	Region Jönköping	January 31st	Wuhan, China
2	Västra Götaland	February 26th	Northern Italy
3	Västra Götaland	February 27th	From Patient 2
4	Västra Götaland	February 27th	From Patient 2
5	Västra Götaland	February 27th	Northern Italy
6	Region Uppsala	February 27th	Germany
7	Region Stockholm	February 27th	The majority of Stockholm's cases infected in Italy or via direct contact with people infected in Italy. Some cases infected in Iran or via direct contact with people infected in Iran. A few may be the result of community infection
8-9	Region Stockholm	February 28th	The majority of Stockholm's cases infected in Italy or via direct contact with people infected in Italy. Some cases infected in Iran or via direct contact with people infected in Iran. A few may be the result of community infection
10	Region Uppsala	February 28th	Iran
11	Region Jönköping	February 28th	Northern Italy (travelling together with Patient 16)

MY ESTIMATES FINAL ESTIMATES FOR R_{ZERO} BASED ON THE SIMULATED FITS TO THE AUSTRALIAN AND SWEDISH DATA

In my simulations P is my represntation of R_{ZERO} .

When you rearrange the formula for ‘I’ published by Majumder and Mandl :

$$I = \left[\frac{R_0}{(1+d)^t}\right]^t$$

to calculate their R_{ZERO} you become aware that depending on what epidemiologists assign as the value of t (*t is the number of serial intervals that have passed at time of model parameterization)* greatly affects what value of R_{ZERO} you get :

$$R_{ZERO} = \sqrt[t]{I}\ .\ (1 + d)^t$$

Notice also that in their paper they set d to zero so the second term resolves to 1 and their in their modelling the effective equation is

$$R_{ZERO} = \sqrt[t]{I}$$

Conclusion - their R_{ZERO} is identical to t*th* root of I !

In my modelling I have set t = 1 day.

In the paper by Majumder and Mandl the following can also be deduced :

Serial interval = 6 – 9 days
Incubation period = 3.5 – 5.1 days

Rather than work with ranges I have chosen to used INT(Mean) :

Serial Interval = 7 days

Incubation period = 4 days

So infectious period = 7 – 4 = 3 days.

So I postulate that it is only for a period of three days that my estimate of R_{ZERO} applies. So to get it to match these authors infectious period I needed to raise my estimate to the power 3.

So for the Australian data fit my directly comparable value of R_{ZERO} is 1.35 * 1.35 * 1.35 = 2.46

Majumder and Mandl's estimate is 2.2 – 2.7 which has a mean of 2.45.

We agree exactly.

When I fitted the model to the Swedish data I got P = 1.35 for the first peak and 1.21 for the second peak. So for the Swedish data fit my directly comparable value of R_{ZERO} are :

1.35 * 1.35 * 1.35 = 2.45
and
1.21 * 1.21 * 1.21 = 1.77

Leakage of data points from the first peak into the second peak most probably explains why the R_{ZERO} for the second peak was lower than Majumder and Mandl's lower confidence limit for R_{ZERO} .

MY CONCLUSION

The rule based stochastic simulations I have applied to both the Australian and Swedish data are very good and return values of R_{ZERO} that agree very well with the CDC reports for R_{ZERO} based upon the Wuhan data.