

COVID19 シミュレーション

梶昌孝

2020年5月6日

1 セットアップ

1.1 使える情報

- 死亡率 $P_{death} = 3\%$
- 感染期間 $D = 10(\text{days})$
- 無症状者の率 $a = 30\%$

日本での死亡率は 2020/4/29 現在 $435/13944 = 3\%$ であった。無症状者の率は武漢帰国者からは 30%, クルーズ船では 20% であった。

1.2 仮定

毎日 $f = 10$ 人の人と濃厚接触する。その時間は $T = 5$ 時間とする。1 時間の濃厚接触で 2% の感染率とする。症状のない $a = 30\%$ の人が感染を広げ、残りの 70% の人は隔離されている。一度感染すると $D = 10$ 日間は感染状態となり、感染力を維持する。その後は回復による免疫獲得、もしくは死亡により、二度と感染しない。死亡も含め総人口 N は一定であり、外部からの流入はない。

2 結果

k 日目の 1 日の感染者の数 $N_{inf-per-day}(k)$ は前日の感染者の数 N_{inf} とまだ感染していない人の数 N_{normal} に依存するとする. ただし, 今回のモデルでは人数は全人口で割って規格化したもの (全人口に対する割合) を使用する. 一定の割合で存在する, 1 人の無症状感染者が 1 日に接触する未感染者の人数の期待値は, (接触する人数) \times (無症状感染者の割合) である. さらに, 未感染者が 1 回の接触で感染しない確率は, $(1 - (1 \text{ 時間当たりの感染率}))^{(\text{接触時間})} = const.$ である. 感染する確率も $1 - (\text{感染しない確率}) = const.$ である. よって, 1 日あたりの感染者は現在の感染者数と, 未感染者の数に比例する.

$$N_{inf-per-day}(k) \propto N_{inf}(k-1) \times N_{normal}(k-1) \quad (1)$$

比例定数は, (濃厚接触の人数) \times (無症状感染者の割合) \times (1 回の接触により感染する確率) である. これは 1 日あたりの基本再生産数 (1 人あたりが感染を広げる人数) と考えることができ, (基本再生産数) / (感染期間) で与えられる. 今回の問題設定では比例定数はおよそ 0.29 で, 感染期間は 10 日なので, 基本再生産数は 2.9 である. 更新アルゴリズムは

1. Compute $N_{inf-per-day}$
2. Update $N_{inf} : N_{inf} = N_{inf} + N_{inf-per-day} - N_{recovered/deth}$
3. Update $N_{normal} : N_{normal} = N_{inf-per-day}$

である. 更新アルゴリズムの計算量は $O(1)$ である.

以下にプログラムの実行結果を示す.

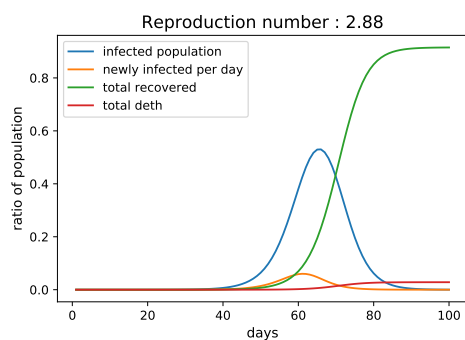


図 1 時間発展 (linear scale)

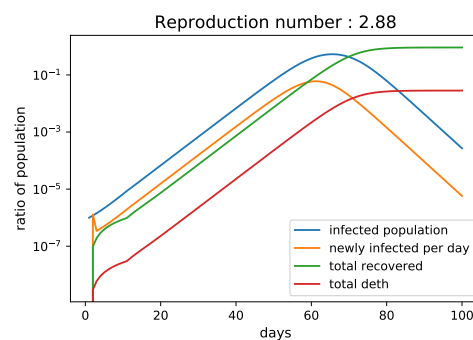


図 2 時間発展 (log scale)

(図 1) と (図 2) は対策をとらなかった場合の感染の広がりを示している. (図 1) からわかる通りに, 感染拡大からおよそ 60 日でピークを迎え, その時の感染者数は人口の過半数である. 最終的な感染者数は, 人口の 9 割を上回る. (図 2) を見ると, 感染拡大からおよそ 60 日間は感染者数は指数関数的に増加していることが分かる. 次に, 対策をとった場合を考える.

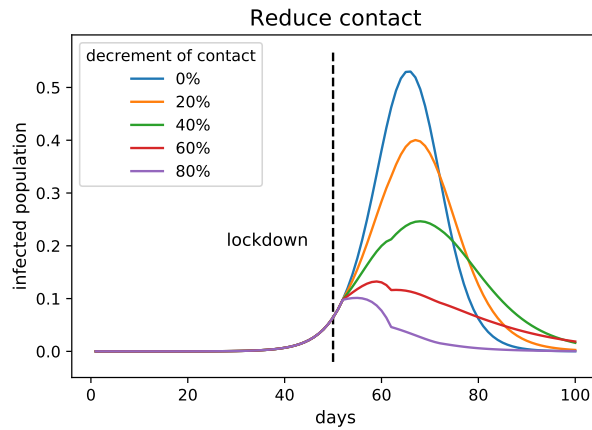


図3 ロックダウンのモデル

(図3)は、感染拡大から50日目でロックダウンを行った場合を想定する。他人との接触をおよそ半分にすると、ピーク時の感染者数を半分に抑えることができる。接触6割減では2か月後も感染者が存在しているのに対して、接触8割減では1か月で収束している。曲線が滑らかでないのは潜伏期間や感染期間が固定であるためで、乱数を使えば滑らかになる。

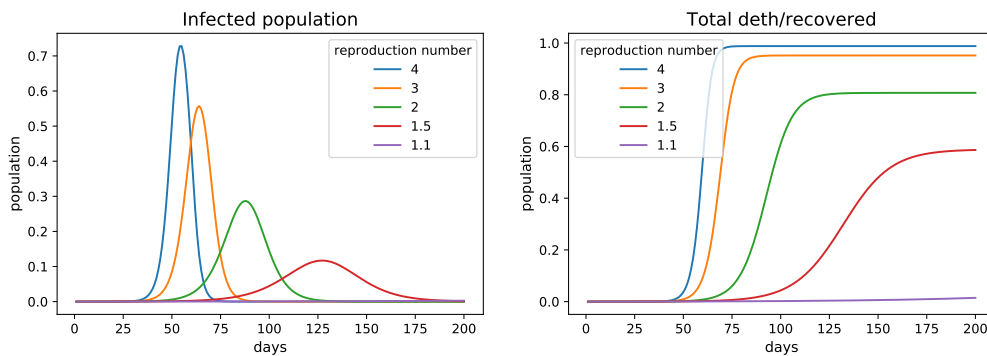


図4 基本再生産数を変えたモデル

(図4)は基本再生産数が感染拡大にどのように影響しているのかを示している。基本再生産数の変更は20日後に行い、早期の対応をした場合のモデルとみることもできる。基本再生産数が2を割ると、感染拡大は非常に緩やかになる。1人が1.1人に感染させるモデルではほとんど感染は拡大しない。

3 ソースコード

以下に今回作成したプログラムを示す.

メインプログラム

```
1 import numpy as np
2 from collections import deque
3 import matplotlib.pyplot as plt
4 from matplotlib.backends.backend_pdf import PdfPages
5
6 # 状況 東京都において,10 日間連続で毎日一人ずつ感染者が見つかる.
7 # 10 人目の感染者が出た時点が 0 日
8
9 pdf = PdfPages('COVID19.pdf')
10 # manual parameter
11 timesteps = 100 # [days]
12 deth_rate = 0.03
13 manual_param = {"contact" : 10,
14                 # Number of people who contact a infected people per day
15                 "asympt_rate" : 0.3, # asymptomatic rate
16                 "inf_rate_per_hour" : 0.02,
17                 # infection rate per 5 hours close contact
18                 "contact_time" : 5} # [hour]
19
20 # initial values
21 infected_per_day_init = deque([1e-7]*10) # initial 10 days
22 infected_init = 1e-6 # 1 / 10,000,00 -- 10 infected in Osaka
23
24 normal = np.empty(timesteps)
25 infected = np.empty(timesteps)
26 recov_deth = np.empty(timesteps)
27 # set initial value
28 normal[0] = 1
29 infected[0] = infected_init
30 recov_deth[0] = 0
31 spd = Spread_per_day(infected_init, infected_per_day_init, **manual_param)
```

```

32 for i in range(timesteps-1):
33     normal[i+1], infected[i+1], recov_deth[i+1] = spd.update()
34
35 days = np.arange(1, timesteps + 1)
36 deth = recov_deth *deth_rate
37 recover = recov_deth *(1 - deth_rate)
38 infected_per_day = np.hstack(([0], np.abs(np.diff(normal))))
39
40 for i in range(2):
41     plt.plot(days, infected, label = "infected population")
42     plt.plot(days, infected_per_day, label = "newly infected per day")
43     plt.plot(days, recover, label = "total recovered")
44     plt.plot(days, deth, label = "total deth")
45
46     plt.title("Reproduction number : {}".format(np.round((spd.reproduction_number), 2)), fontsize = 14)
47     plt.xlabel("days", fontsize = 11)
48     plt.ylabel("ratio of population", fontsize = 11)
49     plt.legend()
50
51     if i == 1:
52         plt.yscale("log")
53     pdf.savefig()
54     plt.show()
55
56
57 #-----
58 # Lock down model
59 plt.figure()
60 lock_down_day = 50
61 reduce_contact_rate = [1, 0.8, 0.6, 0.4, 0.2]
62 for rcr in reduce_contact_rate:
63     # initial values
64     infected_per_day_init = deque([1e-7]*10) # initial 10 days
65     infected_init = 1e-6 # 1 / 10,000,00 -- 10 infected in Osaka
66
67     infected = np.empty(timesteps)
68     # set initial value
69     infected[0] = infected_init

```

```

70     spd = Spread_per_day(infected_init, infected_per_day_init, **manual_param)
71     for i in range(timesteps-1):
72         _, infected[i+1], _ = spd.update()
73         if i == lock_down_day:
74             spd.lock_down(rcr)
75     days = np.arange(1, timesteps + 1)
76     plt.plot(days, infected, label = str(int(np.round(100*(1.0 - rcr)))) + "%")
77
78     plt.title("Reduce contact", fontsize = 14)
79     plt.xlabel("days", fontsize = 11)
80     plt.ylabel("infected population", fontsize = 11)
81     plt.legend(loc = "upper left", title = "decrement of contact")
82     plt.text(28, 0.2, "lockdown", fontsize = 11)
83     ymin = -0.02
84     ymax = 0.57
85     plt.vlines([lock_down_day], ymin, ymax, linestyle='dashed')
86     pdf.savefig()
87     plt.show()
88
89     #-----
90     # Change reproduction number
91     fig = plt.figure(figsize=(13, 4))
92     ax1 = fig.add_subplot(121)
93     ax2 = fig.add_subplot(122)
94     reproduction_list = [4, 3, 2, 1.5, 1.1]
95     timesteps = 200
96     for rep in reproduction_list:
97         infected = np.empty(timesteps)
98         recov_deth = np.empty(timesteps)
99         # set initial value
100        infected[0] = infected_init
101        recov_deth[0] = 0
102        # initial values
103        infected_per_day_init = deque([1e-7]*10)
104        infected_init = 1e-6
105
106        infected = np.empty(timesteps)
107        # set initial value

```

```

108     infected[0] = infected_init
109     spd = Spread_per_day(infected_init, infected_per_day_init, **manual_param)
110     for i in range(timesteps-1):
111         _, infected[i+1], recov_deth[i+1] = spd.update()
112         if i == 20:
113             spd.change_reproduction_number(rep)
114     days = np.arange(1, timesteps + 1)
115     ax1.plot(days, infected, label = str(np.round(rep, 2)))
116     ax2.plot(days, recov_deth, label = str(np.round(rep, 2)))
117
118     ax1.set_title("Infected population", fontsize = 14)
119     ax1.set_xlabel("days", fontsize = 11)
120     ax1.set_ylabel("population", fontsize = 11)
121     ax1.legend(loc = "upper right", title = "reproduction number")
122     ax2.set_title("Total deth/recovered", fontsize = 14)
123     ax2.set_xlabel("days", fontsize = 11)
124     ax2.set_ylabel("population", fontsize = 11)
125     ax2.legend(loc = "upper left", title = "reproduction number")
126     pdf.savefig()
127     plt.show()
128
129     pdf.close()

```

1日あたりの更新アルゴリズム

```

1  # infected_per_day_init should be deque
2  class Spread_per_day:
3      def __init__(self, infected_init, infected_per_day_init,
4                  contact, asympt_rate, inf_rate_per_hour, contact_time):
5          inf_rate = 1 - (1 - inf_rate_per_hour)**contact_time
6          self.inf_factor = contact *asympt_rate *inf_rate    # constant
7          self.reproduction_number = self.inf_factor*10
8                  # infected_factor * infection_period
9          self.infected_per_day = infected_per_day_init
10         self.infected = infected_init
11         self.normal = 1 - infected_init
12         self.recov_deth = 0

```



```
13
14     def lock_down(self, reduce_contact_rate):
15         self.inf_factor *= reduce_contact_rate
16
17     def change_reproduction_number(self, reproduction_num):
18         self.reproduction_num = reproduction_num
19         self.inf_factor = reproduction_num / 10
20
21     def update(self):
22         inf_10days_ago = self.infected_per_day.popleft()
23         inf_new = self.inf_factor * self.normal * self.infected
24         self.infected_per_day.append(inf_new)
25         self.infected += inf_new
26         self.infected -= inf_10days_ago
27         self.recov_deth += inf_10days_ago
28         self.normal -= inf_new
29         return self.normal, self.infected, self.recov_deth
```