R とパンデミックの数理モデル新型コロナウィルス (COVID-19) 研究を例に

ill-identified

2020-02-29

概要

この原稿は新型コロナウィルス (COVID-19) によって中止に追い込まれた第 84 回 Tokyo.Rのゲリラ投稿である. 色々あったので本来の予定を変えて時事ネタにしてみた. 微分方程式に基づいた 2 種類のモデル, つまり SIR モデルと SEIR モデル, およびマルチエージェントモデルに基づいたシミュレーションで, パンデミック[*1]のメカニズムを解説する. 標準的な SIR/SEIR モデルは比較的簡単なので, ちょっとググるだけでも計算してみたという例がいくつも出てくる. そこで今回はこれらの基本的なモデルの性質についてもう少し深く踏み込んで, モデルから何が分かるかという話をする. さらに, COVID-19 について SEIR を拡張したモデルによる研究例についても R で実装しつつ解説する.

イントロダクション

なぜシミュレーションか. 後で紹介する **?** は次のように書いている.

> 統計モデルに対して, 動学的方程式に基づいた数理モデルはあまり注目されないが, 伝染病流行のダイナミクスのより詳しいメカニズムを得ることができる.

以前にも言ったように, 分析の結果には必ず「ある仮定のもとで」という前置きがあり, それを無視したもののほとんどは, 結局何も言っていないか, 間違えだらけのものになる. ここで重要なのは, 前提が正しいかどうかというよりも, 前提から結果までのロジックが適切かどうか, ということである. この前提と結果の関係をシンプルかつ明確にする方法の 1 つがシミュレーションであるとも言える.

もちろん現実世界の現象をうまく捉えたシミュレーションは価値があるが, たとえ前提が現実を反映していなくとも, 「もしこうだったら?」という思考実験に価値が全くないとは言えない. なぜなら, **現実世界では人間が対策を立てることでいくらでも将来の結果を変えられる可能性があるからである**. つまり, シミュレーションで想定していなかった行動を取ることで結果は良い方にも悪い方にも変えられる. ここで重要なのは将来を予言することではなく, もしこの行動を取ったらどうなるか, について整合的な仮想ができるかである.

技術的な話をするならば, 1 日や 1 週間ごとに国民全員を検査して感染者をカウントするのはほぼ不可能なので, このようなシミュレーションで感染の広がりを予想するしかない. また, 標準的な統計モデル (つまり, 伝統的な回帰モデルや因果推論, 機械学習や「AI」によるもの) の多くは, 伝染病のような個人間で相互に関係する現象を捉えるのは難しい.

読む前から分かっていること

1. 中国の湖北省を中心に新型コロナウィルス (COVID-19) が流行している

[*1] パンデミック (pandemic) の定義はいろいろあり, endemic, epidemic などの類義語とも区別しづらい. 本文では漠然と伝染病の感染者数が増加する状態を指してこう呼んでいる.

2. SIR, SEIR モデルは感染症の流行モデルで最も基本的なもの
3. SIR モデルや SEIR モデルは簡単なので計算だけした例なら既にある
4. 今回はモデルから分かることをもっと踏み込んで話す
5. 一度感染した人が再感染する場合を考慮したモデルがある

これを読んで分かること

1. SIR/SEIR モデルは再感染を考慮しないのでパンデミックが起こるのは 1 度きり
2. 感染予防はパンデミックの規模を軽減したり発生を遅らせたりできる
3. 再感染を考慮するモデルはもっと予想しづらい結果になる
4. SEIR モデルを拡張して COVID-19 の流行を分析した研究がもういくつかある
5. 付録として具体的な数値計算の方法のヒントを書いた
6. お前, 得したな

SIR モデル

? が提案したモデルは, **SIR モデル**と呼ばれる[*2]. 人口全体を **感受性状態** (感染しうる人の数, $S(t)$), **感染状態** ($I(t)$), **回復状態** ($R(t)$) に分解して考えるためだ. このモデルはいたってシンプルで, (1) 一定期間ごとに未感染者から β の割合で感染者が発生し, (2) 感染者は γ の割合で感染から回復する, というものである. このモデルの構造をネットワーク図で表現すると, 図 @ref(fig:diagram-sir) のような一方通行な推移のモデルになるとわかる.

```
diag_sir <- dagify(I ~ S, R ~ I,
                   labels = c("S" = "感受性", "I" = "感染", "R" = "回復"),
                   coords = list(x = c(S = 0, I = .1, R = .2), y = c(S = 0, I = 0, R = 0))
                   ) %>% tidy_dagitty() %>% mutate(label_edge = c("gamma", "beta", NA))
diag_sir %>% ggplot(aes(x = x, y = y, xend = xend, yend = yend)) +
  geom_dag_node(size = 30) +
  geom_dag_edges(edge_width = 2, arrow_directed = grid::arrow(
    length = grid::unit(.1, "npc"), type = "closed")) +
  geom_dag_text(aes(label = label)) +
  geom_label(aes(x = (x + xend)/2, y = (y + yend)/2, label = label_edge), parse = T, size = 10) +
  theme_dag_blank()
```

β 感染力, γ 回復力と呼ばれる. 数式で表すと, 以下のようになる.

[*2] 彼らの提案したモデルは実際にはもう少し複雑だったが, ここでは教科書でよく紹介される単純化した例で紹介する. 教科書によっては, 総人口 N を式に含んでいるものがある. どちらも間違えではない. ここでは $N = 1$ に固定することで人口比率として計算している.

図1　SIR モデルの構造

$$\frac{dS(t)}{dt} = -\beta S(t)I(t),$$
$$\frac{dI(t)}{dt} = \beta S(t)I(t) - \gamma I(t)$$
$$\frac{dR(t)}{dt} = \gamma I(t)$$

数学の問題で言うと SIR モデルは線形常微分方程式 (大学 1,2 年生のうちに習うであろうシンプルな微分方程式の一種) なので, β, γ の大きさを問わず, 似たような形状になることが分かっている.

実際に SIR モデルをもとに計算して感染者の数がどう推移するかを図で表すと, 以下の図 @ref(fig:plot-sir) のようになる.

```
sir <- function(t, y, params){
  with(
    as.list(c(params, y)),
    list(c(
      S = -beta * S * I,
      I = beta * S * I - gamma * I,
      R = gamma * I)
    ))
}
y_init <- c(S = .9999999, I = .0000001, R = 0)
out <- ode(y = y_init, times = 1:100, func = sir, parms = c(beta = 1.0, gamma = .1))
as_tibble(out) %>% mutate_all(as.numeric) %>% pivot_longer(cols = S:R) %>%
  mutate(name = factor(name, levels = c("S", "R", "I"),
                       labels = c("感受性人口", "回復人口", "感染人口"))) %>%
  ggplot() + aes(x = time, y = value, group = name, fill = name) + geom_area() +
  scale_fill_tableau() + thm +
  labs(x = "経過時間", y = "人口割合",
       caption = "※数値は実際の感染者数を予想するものではありません") +
  scale_y_continuous(labels = scales::percent)
```

かなり極端な例だが, 例えば感染者が全人口の 0.00001%(人口 1 億人なら 1 万人), 感染力 100%, 回復力 10% とした場合に, どれくらいの勢いで感染者が増加していくかを見てみる. この場合の感染力や回復力とは一定期間での話なので, 感染力は 1 週間以内に新たに感染者数の何倍くらいの感染者が発生するか (この場合は感染者数と

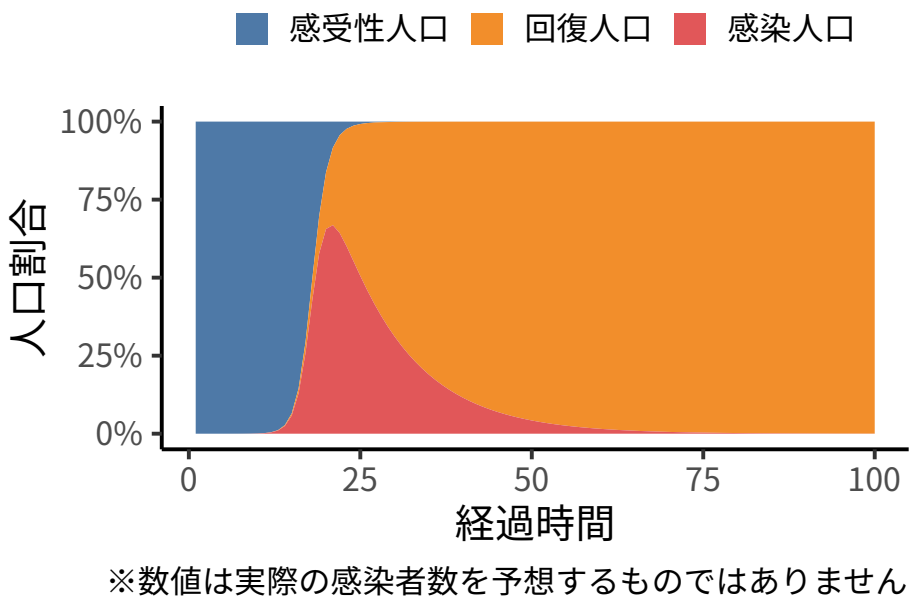


図2　SIR モデルによるパンデミックが起こった場合のシミュレーション結果

同数), 1 週間ごとに何% の確率で回復できるかどうかを表す. 少しわかりづらいので, 感受性人口と回復人口を「**非感染人口**」にまとめてもう一度作図してみる.

```
as_tibble(out) %>% mutate_all(as.numeric) %>% mutate(no_I = 1 - I) %>%
  select(time, I, no_I) %>% pivot_longer(cols = I:no_I) %>%
  mutate(name = factor(name, levels = c(" no_I" , " I" ),
                       labels = c(" 非感染人口" , " 感染人口" ))) %>%
  ggplot() + aes(x = time, y = value, group = name, fill = name) + geom_area() +
  scale_fill_tableau() + thm +
  labs(x = " 経過時間" , y = " 人口割合" ,
       caption = " ※数値は実際の感染者数を予想するものではありません" ) +
  scale_y_continuous(labels = scales::percent)
```

図 @ref(fig:plot-sir2) からわかるように, SIR モデルは**あるタイミングで爆発的に感染者が増加し**, そして (極端な設定であっても) **ピークに達した後は減少する**.

厚生労働省のサイトでも, 以下のような図 @ref(fig:mhlw) が掲載されている. SIR モデルを念頭に入れるなら,「感染拡大防止」は感染力 β が大きくならないように,「重症化防止」は γ を高める (もちろん生存するように) 努力を表していると思われる.

確認のため, SIR モデルのパラメータを変えた場合を見てみよう. 今度は感染力を 50% に下げてみると, 図 @ref(fig:plot-sir-beta) のように, 確かに感染増加のタイミングが遅れ, ピーク時の感染者数も低下している. 国内の保健当局もこのようなモデルに基づいて対策を立てていることがわかる.

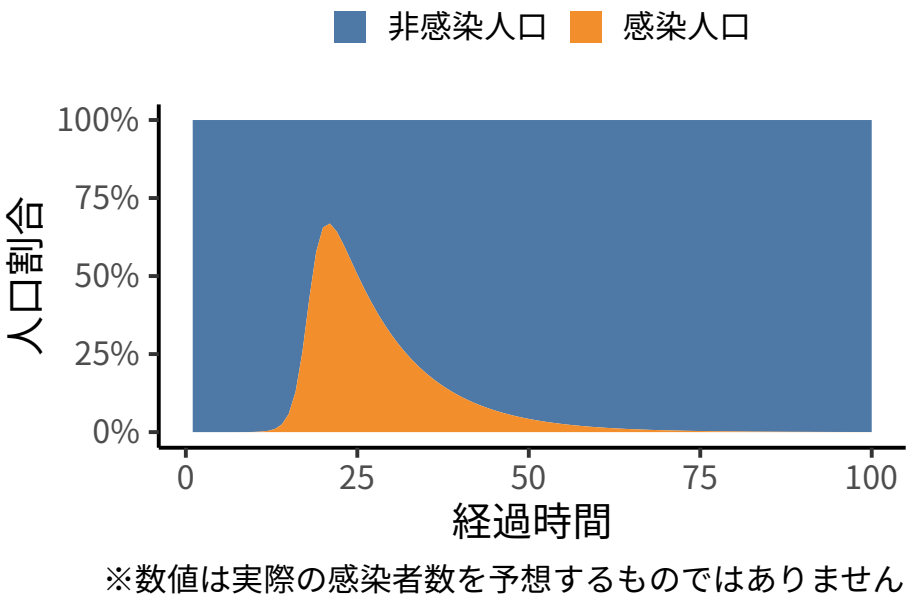


図3　SIR モデルによる非感染者・感染者割合のシミュレーション結果

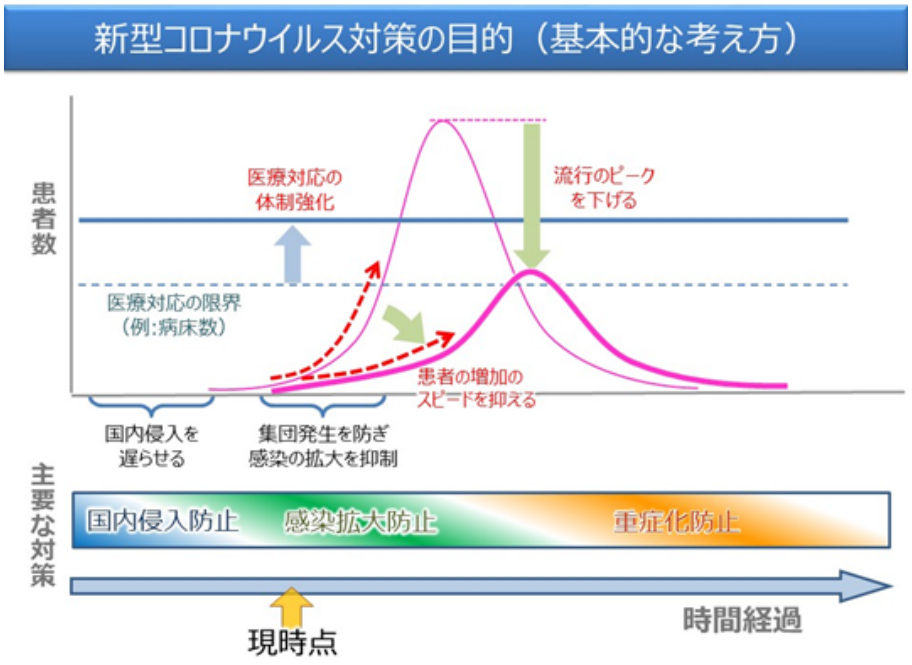


図4　新型コロナウィルス対策の目的 (厚生労働省『新型コロナウイルス感染症について』より)

```
out_low_beta <- ode(y = y_init, times = 1:100, func = sir, parms = c(beta = .5, gamma = .1))
as_tibble(out_low_beta) %>% mutate_all(as.numeric)  %>% pivot_longer(cols = S:R) %>%
  mutate(name = factor(name, levels = c(" S" , " R" , " I" ),
                       labels = c(" 感受性人口" , " 回復人口" , " 感染人口" ))) %>%
  ggplot() + aes(x = time, y = value, group = name, fill = name) + geom_area() +
  scale_fill_tableau() + thm +
  labs(x = " 経過時間" , y = " 人口割合" ,
       caption = " ※数値は実際の感染者数を予想するものではありません" ) +
  scale_y_continuous(labels = scales::percent)
```

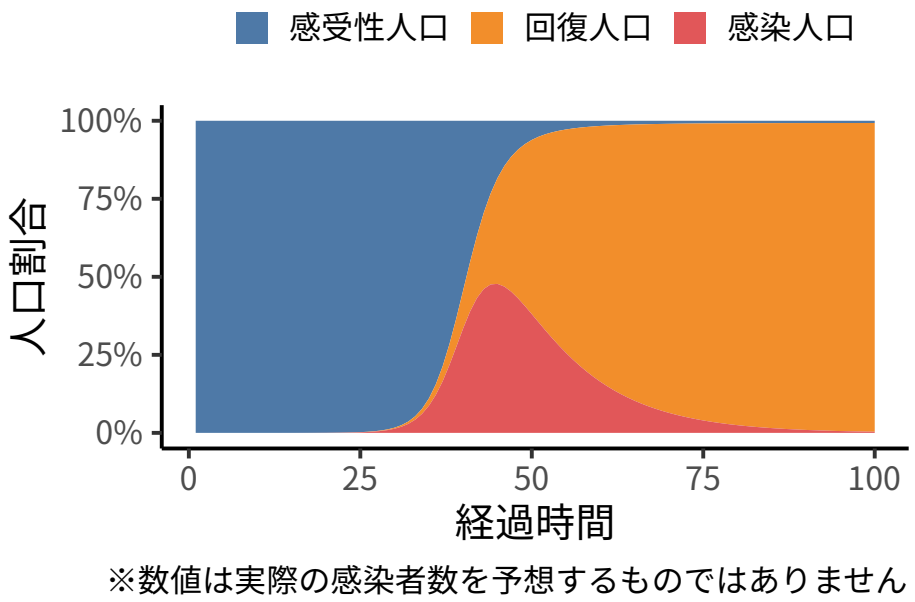


図5　感染率を引き下げた場合のシミュレーション結果

ではパンデミックはどういう条件で発生するのか. 当たり前だが感染者より回復者のほうが増えるスピードが速ければ[*3], 感染者は増えないのでパンデミックとは言えない. ? がこの条件について解説している. 感染初期はほとんど感染者がいないということなので, 感受性人口 $S(t)$ を総人口 N で置き換えられる. この条件をもとに微分方程式を解くと, 感染者人口が以下のように**指数関数**で表せるとわかる.

$$I(t) = I(0)\exp((\beta N - \gamma)t)$$

指数関数は係数 $(\beta N - \gamma)$ がゼロを超えれば急激に増加し, ゼロ以下ならば増加しない性質がある. つまり先ほど示した図は, 係数がゼロを超えた場合である. 式変形するとパンデミックの発生条件は

$$R_0 = \frac{\beta N}{\gamma} > 1$$

[*3] より正確には, 回復人口は回復だけでなく死亡も含む. よって単純に感染力はあるが重症化せずすぐに治るような伝染病ではパンデミックは起こりにくいし, 逆にエボラ出血熱のように死亡率の高い危険な病気でも起こりにくいことになる.

となる. この R_0 は**マルサス係数**[*4]とか**基本再生産数**とか呼ばれる. $R_0 < 1$ ならばそもそもパンデミックは起こらず, $R_0 > 1$ ならばパンデミックが発生しピークに至った後は再び減少する. 一応, 数値計算で確認してみよう. 最初の例とは逆に, $\beta = 0.1, \gamma = 1.0$ として計算した. 感染しにくく回復しやすいという設定である. 結果は感染者数が急激に減って見づらいため, 感染者数のみ図 @ref(fig:compute-sir2) に表す. このようにパンデミックの条件を満たさない場合は感染者数は急激に減少する.

```
out2 <- ode(y = y_init, times = 1:50, func = sir, parms = c(beta = .1, gamma = 1.0))
as_tibble(out2) %>% mutate_all(as.numeric) %>%
  ggplot(aes(x = time, y = I)) + geom_line(size = 1.5) +
   thm + labs(x = "経過時間", y = "人口割合") +
  scale_y_continuous(labels = function(x) scales::percent(x, scale = 6))
```

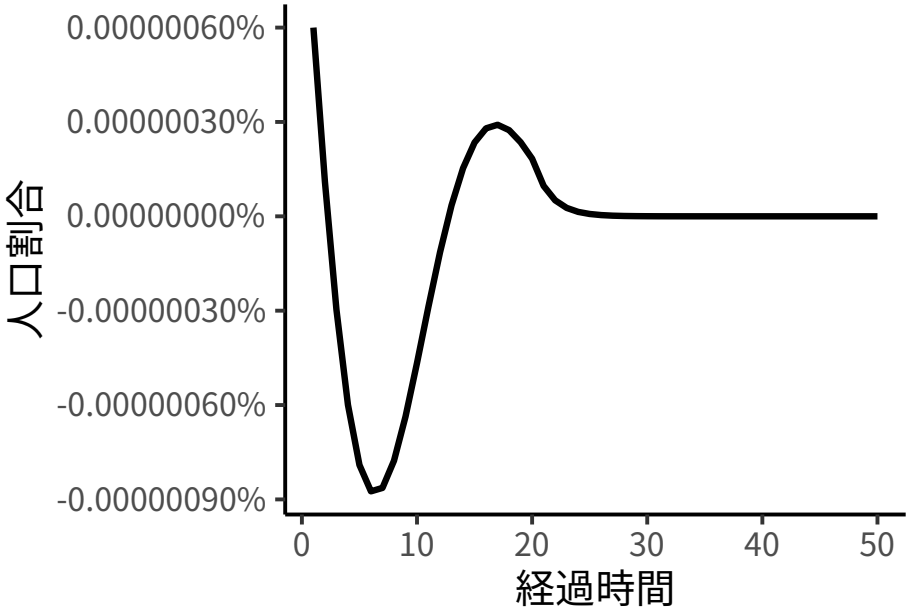


図6　パンデミックが起こらない場合の感染人口の推移

例えば今回のコロナウィルスは, **飛沫感染と接触感染のリスクが高いとされている**[*5]. よって SIR モデルで置き換えると人間同士の距離が近く, より多くの人間と接触があるほど感染力が β が増加するため, パンデミックの条件を満たしやすくなるだろう. よって SIR モデルで置き換えるなら感染者が増加し始める条件は**人口密度が一定の値を超えるかどうか**であるとわかる. よって, 外出を避け繁華街を出歩かないようにすることはパンデミックの抑止に効果があることになる.

さて, 当然の興味として, 次は感染拡大は何日後かとか, 感染者数は最大で何人かとか具体的な数値を知りたいと思うだろう. あるいは死亡率を設定し, 感染者が一定確率で死亡するようなシミュレーションをすれば, この流行が終息するまでに何人死者が出るかという見積もりもできるだろう. しかし, 現時点では国内の感染例はごくわずかである. この数値を元に β, γ, そして死亡率を決めて計算することもできるが, その数値に信頼性があ

[*4] 人口学の最も古典的なマルサスモデルで, 人口が増加する速さを決める要因になるものと同じである.
[*5] 2 月 25 日時点での厚生労働省の発表: https://www.mhlw.go.jp/stf/seisakunitsuite/bunya/0000164708_00001.html

るとは思えない[*6]. **みだりに不安を煽ることも, 安全かどうかとも分からないのに安全だと偽ることも私はしたくない**ため, 具体的な数値の計算はしないことにする.

より現実的な要因を考える

SIR モデルでは感染した人間は一定確率で回復し, **一度回復した人間は二度と感染しない**. また, 新規感染者数は感受性人口と感染人口と感染力 β の積で決まる. そのため図 @ref(fig:plot-sir) のように徐々に回復人口が増えるとともに新規感染者数が減るようになっているので, **ピークは一度きりしかない**. そして実は**感受性人口のうち一定割合の人たちは一度も感染することがない** (?). 実際に, はしかなど一度感染すれば免疫を獲得しほぼ二度とかかることがないとされる病気もあるが, 年をとるほど免疫力が低下することも多い[*7].

つまり, SIR モデルは単純だが, **現実で起こるいろいろな要因を切り捨ててしまっている** [*8].

再感染の可能性を考える

試しに, 一度感染して回復した人たちが, 一定の確率で免疫を失い, 再び感染する可能性が出るモデルにしてみる. これは SIR モデルを少し修正するだけで計算できる. $R(t)$ のうち免疫を失う人の割合を ρ として, 以下のような **SIRS モデル**を考える[*9]. ここでは免疫を失う割合を 1% とした結果を @ref(fig:plot-sirs) に示す.

$$
\begin{aligned}
\frac{dS(t)}{dt} &= -\beta S(t)I(t) + \rho R(t), \\
\frac{dI(t)}{dt} &= \beta S(t)I(t) - \gamma I(t), \\
\frac{dR(t)}{dt} &= \gamma I(t) - \rho R(t)
\end{aligned}
$$

```
diag_sirs <- dagify(I ~ S, R ~ I, S ~ R,
                    labels = c("S" = "感受性", "I" = "感染", "R" = "回復")
                    ) %>% tidy_dagitty() %>%
```

[*6] 単純に見積もろうとする場合, 確率的な統計学上の誤差と, 数値計算上の誤差, 両方の問題があると予想できる. あるいは, 既に感染者数の多い中国での数値を元にシミュレーションすればいいと思いつく人がいるかもしれない. しかし, ここで紹介するモデルの「感染力」とか「回復力」といったパラメータはウイルスの性質だけでなく自然環境や社会環境などさまざまな条件が影響しているため, そのまま日本に適用しても正しい結果が得られるとは限らない.

[*7] 逆に言えばはしかの流行は SIR モデルで予測しやすいということになる. ? は日本ではしかが繰り返し流行する原因として, 免疫人口が 90% を超えていないため, SIR モデルに基づくと流行の再生産係数が 1 を超えているからと主張している. あるいは, 母親からの受動免疫により, 新生児ははしかに対して免疫を持つ可能性がある. これを考慮したモデルは MSIR モデルと呼ばれる. 感染症流行モデルは, 既に判明している感染症の性質に沿ってどういう設定にするかが重要になる.

[*8] 今回は詳しく触れないが, そのほか SIR モデルが切り捨てている要因の一つに, 人口の増減がある. SIR モデルは人口の増減を記述していないため, 人口が一定と仮定していることになる. 今回のように数カ月や 1 年程度の短期間の流行を考えるならばこれで問題ないが, 数十年規模の推移を考えるならば人口の変化も考慮すべきだろう. 例えば人免疫不全ウィルス (HIV) は潜伏期間が 10 年近くあるとされる. また, 今回のコロナウィルスは発生源が中国であった. 地域によって流行が異なることを表現するような多地域モデルの研究も存在する.

[*9] さらに回復人口のステージすら省略し, 感染者人口と感受性人口の 2 つの状態を推移するだけの SIS モデルに簡略化することもできる.

```
  mutate(label_edge = c(" gamma" , " rho" , " beta" ))
diag_sirs %>% ggplot(aes(x = x, y = y, xend = xend, yend = yend)) +
  geom_dag_node(size = 30) +
  geom_dag_edges(edge_width = 2, arrow_directed = grid::arrow(
    length = grid::unit(.1, " npc" ), type = " closed" )) +
  geom_dag_text(aes(label = label)) +
  geom_label(aes(x = (x + xend)/2, y = (y + yend)/2, label = label_edge), parse = T, size = 10) +
  theme_dag_blank()
```

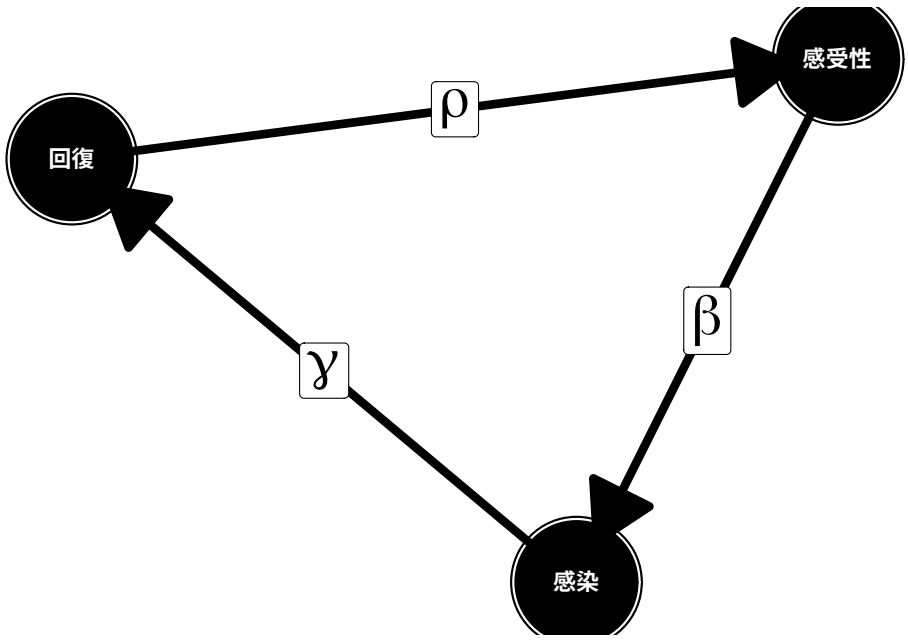


図7　SIRS モデルの構造

```
sirs <- function(t, y, params){
  with(
    as.list(c(params, y)),
    list(c(
      S = -beta * S * I + rho * R,
      I = beta * S * I - gamma * I,
      R = gamma * I - rho * R)
    ))
}
out_sirs <- ode(y = y_init, times = 1:100, func = sirs, parms = c(rho = .01, beta = 1.0, gamma = .1))
as_tibble(out_sirs) %>% mutate_all(as.numeric) %>%
  pivot_longer(cols = S:R) %>%
  mutate(name = factor(name, levels = c(" S" , " R" , " I" ),
```

```
                    labels = c("感受性人口", "回復人口", "感染人口"))) %>%
ggplot() + aes(x = time, y = value, group = name, fill = name) + geom_area() +
scale_fill_tableau() + thm +
labs(x = "経過時間", y = "人口割合",
     caption = "※数値は実際の感染者数を予想するものではありません") +
scale_y_continuous(labels = scales::percent)
```

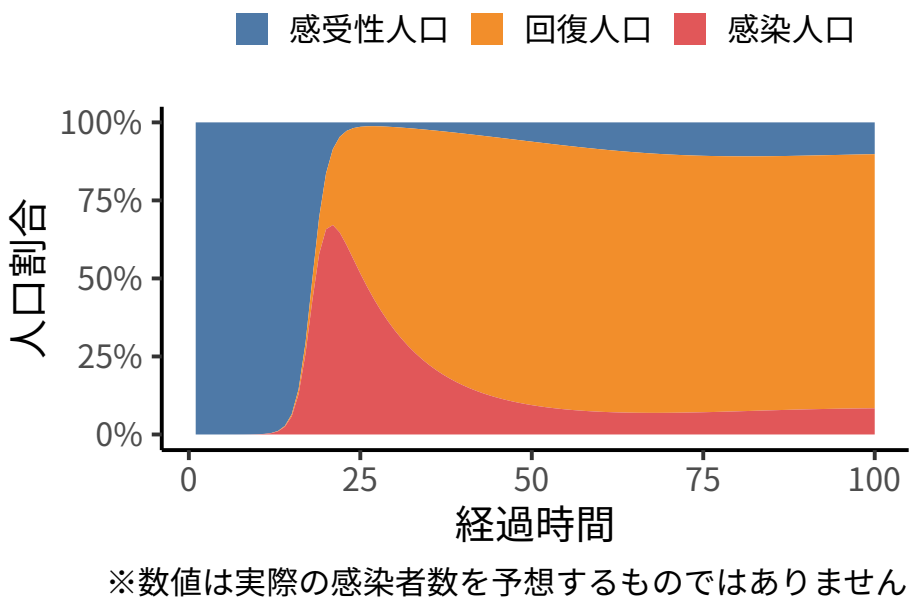


図8　SIRS

SIRS モデルでは, 回復人口から感受性人口に戻る経路があるため, ピークから減少に転じてして時間が経過してからも**常に全人口の 8% 程度が感染者となっている**. もし *rho* が大きければピーク後の感染人口の割合も大きいまま持続する. また, もしある時突然回復人口が全て感受性人口に置き換わった場合, 再びパンデミックが起こりうる. 突然変異したウィルスに対して人々は免疫を持っていないため, パンデミックが発生するメカニズムをこのように表現することもできる.

SEIR モデルの拡張と COVID-19 流行のシミュレーション

例えば SIR モデルと並んでよく紹介されるものに, SEIR モデルがある. E とは感染したが, 発症することも他人に伝染することもない状態, つまり**潜伏期間にある人の数**を表す[*10]. しかし, これは S と I の間にステージを 1 つ挟んだだけであり (図 @ref(fig:diagram-seir)), 正確な感染力や潜伏期間の長さがわからない現状では **SIR との本質的な違いを産まない**ため, ここでは詳しく取り上げない.

[*10] より正確には, **感染待ち期間 (latency period)** と呼ばれる (?).

```
diag_seir <- dagify(
  R ~ I, I ~ E, E ~ S,
  labels = c(" S" = " 感受性", " E" = " 潜伏", " I" = " 感染", " R" = " 回復"),
  coords = list(x = c(S = 0, E =.1,  I = .2, R = .3), y = c(S = 0, E = 0, I = 0, R = 0))
  ) %>% tidy_dagitty() %>% mutate(label_edge = c(" beta", " gamma", " lambda", NA))
diag_seir %>% ggplot(aes(x = x, y = y, xend = xend, yend = yend)) +
  geom_dag_node(size = 30) +
  geom_dag_edges(edge_width = 2, arrow_directed = grid::arrow(length = grid::unit(.1, " npc"), type = " closed")) +
  geom_dag_text(aes(label = label)) +
  geom_label(aes(x = (x + xend)/2, y = (y + yend)/2, label = label_edge), parse = T, size = 10) +
  theme_dag_blank()
```

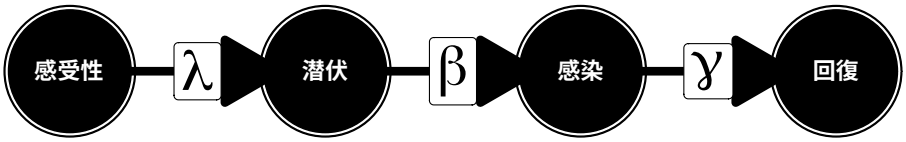


図9　SEIR モデルの構造

一方で, 早くも **?**, **?** が SEIR モデルの応用で中国国内の COVID-19 の流行をシミュレーションした研究が arXiv に投稿されている[*11]. これら研究では, SEIR モデルを拡張して現在の中国政府の政策を反映している. つまり, 検疫措置の効果を考慮できるようにしている[*12].

? は, YJ-SEIR モデルなるものを提案している. 彼らは湖北省の内外でこれまでに感染が確認された人数の推移を検証し, COVID-19 が新型であると確認された 2020/2/12 に大きく増加しているが, それ以降の新規患者数は安定していることから, 中国政府の対策が効果をあげているとして, これを表現するためにモデルを考案した. 現在わかっていることを確認すると, 以下の通り.

1. 現在武漢市では交通を遮断し, 住民の外出も制限することで市全体が隔離されている
2. 全国から医療従事者が招集され臨時の救急医療体制が構築されている
3. 一方で, 交通を遮断する以前には多くの人間の流入出があった
4. COVID-19 は間接接触 (感染者が触れた物に触れること) でも感染する可能性がある

よって, 隔離を行った後でも 2 通りの感染拡大の可能性が残されている. 実際, **ここまでの感染者数の推移は標準的な SEIR モデルの結果とは異なる**と彼らは指摘している.

そこで, YJ-SEIR モデルは図 @ref(fig:diagram-yjseir) のように, まだ感染が確認されていないため隔離されていない人間と, 間接接触で感染した人間という 2 つのグループを追加していることで, 残存した COVID-19 による汚染の影響を考慮したモデルとなっている.

[*11] 知っている人は知っている話だが一応注意書き. arXivはあくまで研究内容を共有するためのサイトであり, 研究内容の信頼性を担保するものではないことに注意.

[*12] 冒頭で「シミュレーションが現実世界の実情を反映しない思考実験でも良い」とした理由の 1 つは, 現実世界では人間が対策を立てることでいくらでも将来の結果を変えられる可能性があるからである. シミュレーションで想定していなかった行動を取ることで結果は良い方にも悪い方にも変えられる. ここで重要なのは将来を予言することではなく, もしこの行動を取ったら

```
diag_yjseir <- dagify(
  Y ~ E,
  J ~ E,
  S ~ Y,
  E ~ S,
  I ~ E + Y,
  R ~ J + I,
  labels = c("Y" = "Suspected", "J" = "間接接触", "S" = "感受性",
             "E" = "潜伏", "I" = "感染", "R" = "回復")
  ) %>% tidy_dagitty() %>% mutate(label_edge = c(
    "beta", NA, NA, "gamma", NA, "lambda",  NA, NA, NA))
diag_yjseir %>% ggplot(aes(x = x, y = y, xend = xend, yend = yend)) +
  geom_dag_node(size = 30) +
  geom_dag_edges(edge_width = 2, arrow_directed = grid::arrow(
    length = grid::unit(.1, "npc"), type = "closed")) +
  geom_dag_text(aes(label = label)) +
  geom_label(aes(x = (x + xend)/2, y = (y + yend)/2, label = label_edge), parse = T, size = 10) +
  theme_dag_blank()
```

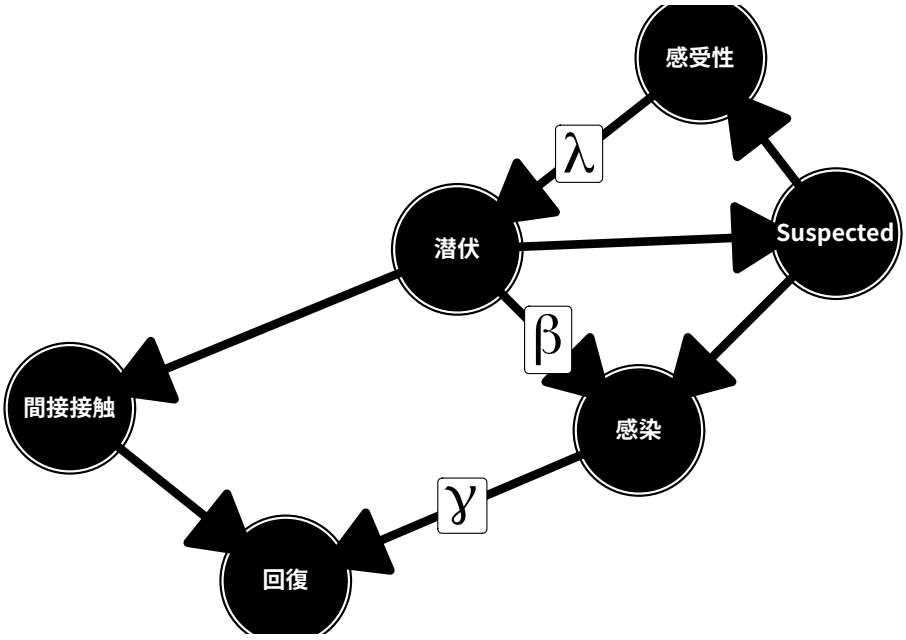


図10　YJ-SEIR モデルの構造

一方で？は提案するモデルを一般化 SEIR モデルと呼んでいる. こちらもやはり, 政府による隔離政策の影響を考慮したモデルである. 一般化 SEIR は YJ-SEIR よりもいくらかシンプルである. SEIR から追加された状態は感受性から移行する, 感染の恐れのなくなった非感受状態 P と感染状態から移行する隔離状態 Q, そして死亡状態

D を明示的に回復人口から分離している (図 @ref(fig:diagram-gSEIR)).

```
diag_g_seir <- dagify(
  P ~ S,
  E ~ S,
  I ~ E,
  Q ~ I,
  R ~ Q,
  D ~ Q,
  labels = c("Q" = "隔離", "P" = "非感受性", "D" = "死亡",
             "S" = "感受性", "E" = "潜伏", "I" = "感染", "R" = "回復"),
  coords = list(x = c(S = 0, P = 1, E = 1, I = 2, Q = 3, R = 4, D = 4),
                y = c(S = 0, P = -1, E = 1, I = 1, Q = 1, R = 2, D = 0)
                )
  ) %>% tidy_dagitty() %>% mutate(label_edge = c(
    "beta", "delta", "mu", "gamma", "lambda", "alpha", NA, NA, NA))
diag_g_seir %>% ggplot(aes(x = x, y = y, xend = xend, yend = yend)) +
  geom_dag_node(size = 30) +
  geom_dag_edges(edge_width = 2, arrow_directed = grid::arrow(
    length = grid::unit(.1, "npc"), type = "closed")) +
  geom_dag_text(aes(label = label)) +
  geom_label(aes(x = (x + xend)/2, y = (y + yend)/2,
                 label = label_edge), parse = T, size = 10) +
  theme_dag_blank() +
  labs(caption="注: パラメータ名は元論文から変更している")
```

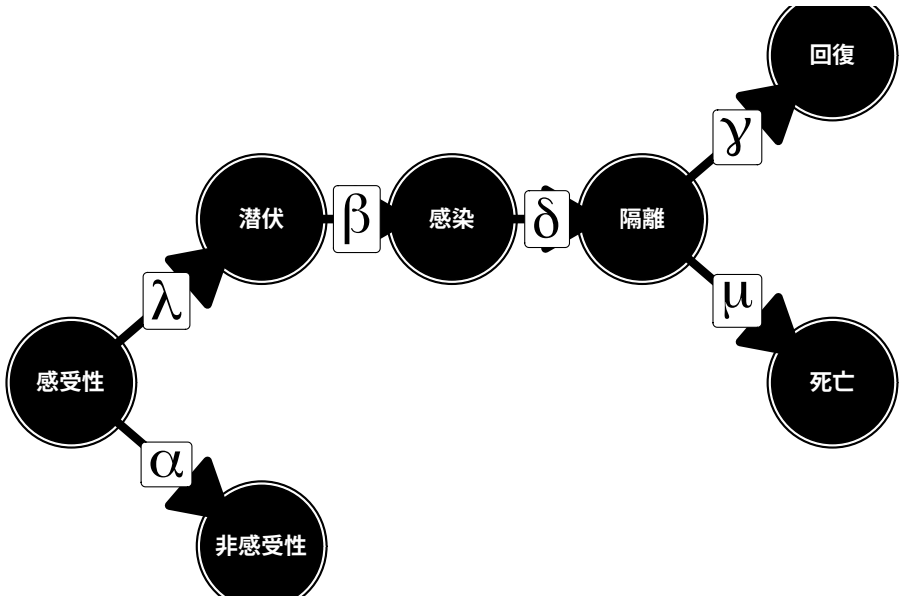
彼らの主目的は, COVID-19 が発生した時期と, この「アウトブレイク」がいつ頃終息するか, の推定である. そのためにはまずモデルのパラメータを正確に推定する必要があり, これにはかなり気を遣っている用に見える.

Generalized GSEIR モデルでのシミュレーション

これらのモデルのうち, ? の一般化 GSEIR モデルは推定したパラメータの数値を掲載しているので, 試しに武漢 (Wuhan) 市を除く湖北 (Hubei) 省のケースでの計算を再現できるように R でやってみる. ただし, 彼らのモデルでは死亡率・回復率パラメータは時間によって変化するものとしている. 時間変化パラメータは正確な数値が記載されていなかったのでグラフから目分量で,

$$\mu(t) = \exp(-0.9(t+3)) + 0.001,$$
$$\gamma(t) = \mathit{logistic}((t-60)/10)$$

と決めた. よって, これらのパラメータも状態変数として微分方程式を計算する. ロジスティック関数の導関数はロジスティック分布の密度関数と同じなので, 計算には `stats::plogis()`, `stats::dlogis()` を使える. シミュレーション結果はグループごとにスケールが違いすぎるため感染者だけを図 @ref(fig:plot-gseir) に掲載する.



注: パラメータ名は元論文から変更している

図11　一般化 SEIR モデルの構造

```
gseir <- function(t, y, params){
  with(
    as.list(c(params, y)),
    list(c(
      S = -lambda * S * I - alpha * S,
      P = alpha * S,
      E = lambda * S * I - beta * E,
      I = beta * E - delta * I,
      Q = delta * I - (gamma + mu) * Q,
      R = gamma * Q,
      D = mu * Q,
      gamma = dlogis(t, 60, 10),
      mu = -0.9 * mu
      )))
}
# Hubei(without Wuhan)
span <- 60
E0 <- 592/45e6
I0 <- 515/45e6
out_gseir <- ode(
  y = c(S = 1 - E0 - I0, P = 0, E = E0, I = I0, Q = 0, R = 0, D = 0,
        gamma = plogis(1, 60, 10), mu = exp(-.9 * (1 + 3)) + .001),
```

```
  times = 1:span, func =  gseir,
  parms = c(alpha = .133, lambda = 1.0, beta = 2^-1, delta = 7.2^-1))
df_gseir <-  as_tibble(out_gseir) %>% mutate_all(as.numeric) %>%
  mutate(date = seq(as.Date(" 2020/1/20 " ), as.Date(" 2020/1/20 " ) + span - 1,
                    by = " 1 day " )) %>%
  pivot_longer(cols = S:D) %>%
  mutate(name = factor(name, levels = c(" S " , " P " , " E " , " I " , " Q " , " R " , " D " ),
                       labels = c(" 感受性 " , " 非感受性 " , " 潜伏期間 " ,
                                  " 感染 " , " 隔離 " , " 回復 " , " 死亡 " )))
```

```
df_gseir %>% filter(name %in% c(" 潜伏期間 " , " 感染 " , " 隔離 " , " 回復 " , " 死亡 " )) %>%
  mutate(value = if_else(value < 0, 0, value),
         category = if_else(name %in% c(" 潜伏期間 " , " 感染 " ), " 未隔離 " , " 隔離 " )
         ) %>%
  ggplot() + aes(x = date, y = value, group = name, fill = name) + geom_area() +
  scale_fill_tableau() + thm + theme(axis.title.x = element_blank(),
                                     axis.text.x = element_text(angle = -45)) +
  facet_wrap(~category, scales = " free_y " , ncol = 1, strip.position = " left " ) +
  labs(y = " 人口割合 " , caption = " ※ 数値は実際の感染者数を予想するものではありません " ) +
  scale_x_date(date_breaks = " 2 weeks " ) + scale_y_continuous(labels = scales::percent)
```

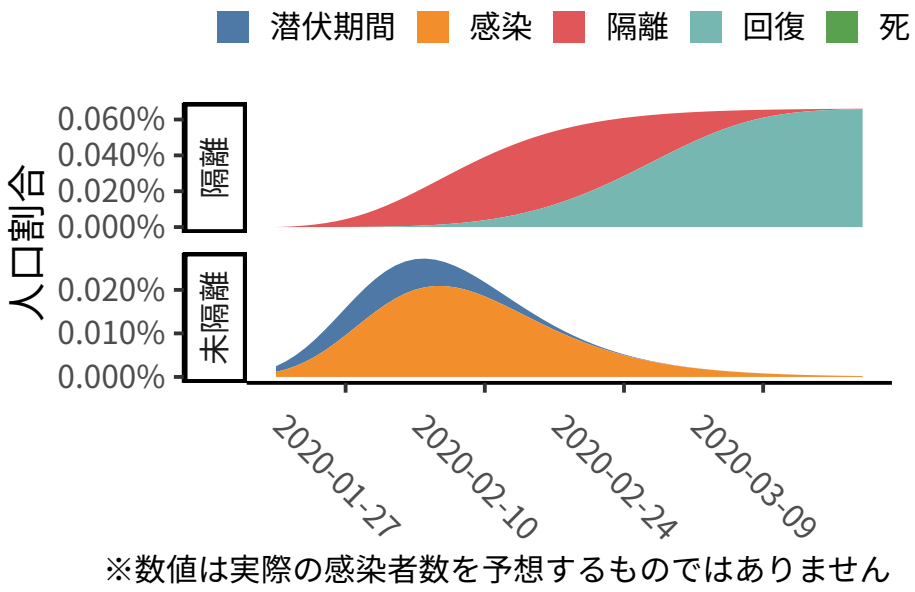


図12　一般化 GSEIR モデルによる感染者の推移: 隔離済み感染者

このシミュレーションでは, 3 月上中旬あたりでで隔離人口が減少して終息することになる. なお死亡者数は今回のやり方だと**計算誤差によりところどころマイナス**になってしまう, というかそれくらい少数の例なので, どちらにせよ**死亡者数に関して信頼性のある予想はできない**ということだ.

しかし繰り返しになるが, このパラメータを**条件の違う日本での流行予測にそのまま適用することはできない**ことに注意してほしい. そもそも, 中国国内の場合であっても, パラメータがどの程度正確かで結果は変わってくる. 詳しくは **?** の内容を読んでほしいが, パラメータが少し違うだけでも期間や感染者数はだいぶ変わってくる.

その他のモデル

(これ以降は時間がなかったので名前を挙げるだけである) 一般に子どもや高齢者ほど病気にかかりやすいため, 感染力は年齢別人口比がどうなっているかでも変化すると予想できる. これを考慮したモデルは**感受性変動モデル**と呼ばれる. もともと SIR モデルも変動を考慮していた. あるいは, 地域ごとに条件の異なるモデルというものも考えられる. ここまで消化したモデルはどれも世界や国家全体の人口に対するものである. よりミクロな感染メカニズムに興味があるのなら, シミュレーションもまたよりミクロな現象を考慮する必要がある. つまり, **エージェントベースド**なシミュレーションである. 例えば, 複数の人間が一定のルールに則って移動し他人と接触する様子を計算で表現することで, 実際の感染の広がり方を想像することができる. 本来はこれもやりたかったのだが, 今週は風邪で寝込んでいたり仕事が忙しかったりしたので間に合わなかった. (マルチエージェントシミュレーションの応用は例えば **?** がインフルエンザでの例を紹介している).

今回紹介したモデルは MAB を除いて全て確率的な分岐のない**確定的**なモデルであった. 今回定性的な情報を得たいのでこれで十分だが, データに基づいて具体的な数値を推定したい場合は**確率的**なノイズを考慮する必要がある. 確率的なモデルとはつまり統計学的なモデルとも言い換えられ, 平均的な変化だけでなく, どれくらいの誤差があるかも予測できる.

わかったことのまとめ

- パンデミックの数理モデルで最もシンプルなのは SIR モデル
- SIR モデルでは感染者の増加が感染からの回復を上回るとパンデミックが始まる.
  - 感染増加を抑える取り組みはパンデミックの開始を遅らせたり, ピーク時の感染者数を軽減できる.
- 感染力が変化するモデル, 再感染があるモデル, マルチエージェントシミュレーションなど, いろいろなモデルが存在する
- 今回の COVID-19 についても, すでに隔離措置を考慮したモデルで感染規模をシミュレーションした研究がいくつかある
- 実際に感染がどれだけ広がるのか, それはまだ混沌の中…
- **それが……, パンデミック…!**

(APPENDIX) 補遺

連立常微分方程式の解き方

ある関数 $y(t)$ があって, その導関数 $dy(t)/dt$ を含んだ方程式を常微分方程式 (もし偏導関数が含まれているなら「偏微分方程式」) といい, この方程式から元の関数 $y(t)$ を再現することを「微分方程式を解く」という. 例えば,

$$dy(t)/dt = \alpha,$$
$$\alpha \neq 0$$

という微分方程式は, $(\alpha t)' = \alpha$ から, $y(t) = \alpha t + c$ と解ける (詳しい話は後述の参考文献を参照). この c は定数項だが, 微分した時点で情報が失われているため値を特定できない. これを**一般解**という. しかし, $y(1) = 0$ という情報があれば, $y(t) = \alpha t - \alpha$ であることが分かる. このように, 定数を特定した答えを**特殊解**という. 微分方程式の実際の問題は多くの場合 $t = 0$ のときの $y(0)$ の値を元に特殊解を求めることが多い. このタイプの問題は**初期値問題**と呼ばれる. 今回の SIR モデルも, 感染人口の初期値を与えてその後の推移を見るので, 数学的には初期値問題を解いていることになる.

関数は $y(t)$ であっても $f(x)$ であっても何でもいいのだが, 微分方程式は多くの場合時間的な変化を表すために使うので, $y(t)$ のように t を変数として書くことが多い. このとき, $y(t)$ は, 時間によって変化する変数とも見なせるので, 「**状態変数**」とも呼ばれる.

微分方程式を解いて $y(t)$ の式を明示的に表す方法 (これを**代数解**とか**解析解**とか**厳密解**とか呼ぶ) にはいくつかのパターンがあるが, 全ての常微分方程式をこのように解けるとは限らない. その場合は数値を近似的に表すことで解くしかない. これを**数値解**という.

SIR モデルを数値的に計算するには, 連立常微分方程式を計算する必要がある. R では `deSolve::ode()` 関数で計算できる. SIR モデルを計算するという例は既に以下のブログで紹介されている.

http://statmodeling.hatenablog.com/entry/sir-model-ode-1

まずは `ode()` の使い方について簡単に書いておく. と言ってもヘルプに記載されている情報以上のことではないが. まず, 微分方程式のうち, この関数に最低限必要な引数は順に, 状態変数の初期値 $y(0)$, 計算する t の期間, 微分方程式を表現した関数オブジェクト, である. 初期値と時間はベクトルで与えるだけなので, 関数について解説する. この関数に最低限与えるべき引数の数は 3 つと決まっている. 順に, 時間変数 t, その時点での状態変数 $y(t)$, そしてそれ以外のパラメータである. 一般に, これらの情報があれば, その時点での $dy(t)/dy$ の値を計算することができる. SIR モデルならば, 既に $dS(t)/dt = ...$, $dI(t)/dt = ...$, の形で表しているため, これをそのまま計算して返すような関数を実装すればよい. あるいは紹介した一般化 SEIR モデルでは一部のパラメータが時間に応じて変化するため, パラメータも状態変数とみなして計算するように書く必要がある (この辺は状態空間モデルの時変パラメータと似たような要領になる). 実装に関しては, ヘルプでの用例や上記のブログでやっ

ているように, with() と名前付きベクトル/リストを利用するとコードの表現が数式に近くなり, 見やすくなると思う.

```
sir <- function(t, y, params){
  with(
    as.list(c(params, y)),
    list(c(
      S = -beta * S * I,
      I = beta * S * I - gamma * I,
      R = gamma * I)
    ))
}
```

このように, as.list(c(params, y)) で名前付きリストにまとめた状態変数とパラメータを with() に与えることで, $ や [[]] を使わずに直接パラメータ・状態変数の名前で参照できる.

では微分方程式の数値計算とは具体的に何をやっているのかというと, 常微分方程式の計算方法はいろいろある. おそらく大学の授業でやることが多いのは比較的単純な**オイラー法**や**ルンゲ＝クッタ法**だろう. 実用上はもっと複雑なアルゴリズムが使われることもあるが, 基本的にはどれも t の微小な変化に対する $y(t)$ の差分を計算することで近似するものである. 私も網羅的に勉強したわけではないが, 例えば **?** の講義ノートの lekce-2 にも常微分方程式をオイラー法で解く方法の解説がある. 今回は簡単な常微分方程式しか扱っていないため, こちらで解説されているオイラー法のアイディアが理解できれば十分だろう. 計算精度でみた実用性について気になる場合は **?** の 7 章などを見れば良い. しかし, 実際にはオイラー法やルンゲクッタ法での計算は, 例えば硬い (stiff) 方程式の問題ではあまりうまくいかないことが多い. deSolve パッケージのマニュアルによれば ode() は ODEPACK などの外部ライブラリを利用して計算しており, デフォルトでは lsoda というアルゴリズムを利用している. これは計算中の解の挙動から stiff でないと判定すれば Adams 法で, stiff と判定すれば後退差分法 (BDF) で計算, と適応的に使い分けることで stiff かどうかにかかわらず解けるようにしたプログラムとのことである[*13]. この辺は **?** もアルゴリズムの性質について紹介している.

上記の記事でも言及されているように, 微分方程式がパラメータによって解がどう変わってくるか ($I(0)$ なら一切感染は発生しないし, SIR より複雑なモデルでは解がなくなる場合もある) という分析を安定性解析という. 数理モデルではこのような定性的な分析も重要になる.

補足: SEIR モデルの計算例

アリゾナ州立大学の Hans Nesse の個人ページに SIER モデルを計算するスクリプト (とモデルの定義) が置いてあるので, それを参考にしてほしい. http://www.public.asu.edu/~hnesse/classes/seir.html いちおう作っておいた. この設定から分かるように, ここでいう潜在期間とは「**感染待ち期間** (latency period)」ともいい, 感染はしているものの発症せず他人に伝染することもない期間を表す. 一般には SEIR モデルは単に SEIR の 4 つの状態がある

[*13] https://people.sc.fsu.edu/~jburkardt/f77_src/odepack/odepack.html

だけで, 式には差異がある.

```
seir <- function(t, y, params){
  with(
    as.list(c(params, y)),
    list(c(
      S = -lambda * S * I,
      E = lambda * S * I - beta * E,
      I = beta * E - gamma * I,
      R = gamma * I)
    ))
}
out_seir <- ode(y = c(S = .9999999, E = .0000001, I = 0, R = 0), times = 1:100, func = seir, parms = c(lambda = 1.0, beta = .5,
as_tibble(out_seir) %>% mutate_all(as.numeric) %>% pivot_longer(cols = S:R) %>%
  mutate(name = factor(name, levels = c(" S" , " R" , " E" , " I" ),
                       labels = c(" 未感染人口" , " 回復人口" , " 潜伏期間人口" , " 感染人口" ))) %>%
  ggplot() + aes(x = time, y = value, group = name, fill = name) + geom_area() +
  scale_fill_tableau() + thm +
  labs(x = " 経過時間" , y = " 人口割合" ,
       caption = " ※数値は実際の感染者数を予想するものではありません" ) +
  scale_y_continuous(labels = scales::percent)
```

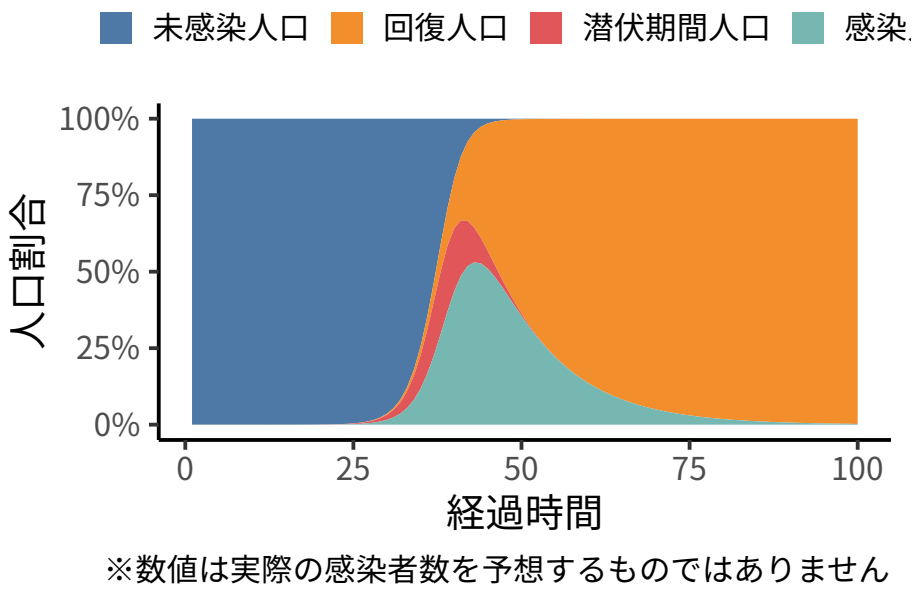


図13　SEIR モデルによるパンデミックが起こった場合のシミュレーション結果

疫学・感染症の数理モデルに関する教科書

これは私の専門分野ではないので,「入門」教科書として今回は **?** を参考にした. これは基本用語の定義から書いてあったり, 最初は差分方程式で説明するなど簡単にする工夫がなされているが, 一方で各モデルの定性的な性質の違いについてあまり詳しく書いていない. それ以外にも英語の教科書はいくつも存在するようだが, 日本語の教科書はほとんど見かけない. 稲葉寿による『感染症の数理モデル』(培風館, 2008 年) というのがあるらしいが高かったので買ってない. それ以外に参照できるものは稲葉寿氏が一般公開している論文くらいだろうか. 関連分野の教科書もいくつかあるようだが, 古すぎてあまり出回っていないものが多いようだ. **?** の邦訳『微分方程式で数学モデルを作ろう』(日本評論社, 1990 年) では SIR モデルにも言及されているが, 全体的にいろいろな分野での微分方程式の応用例を紹介する内容で, 一般的な解法などの説明にも乏しく今回の話の参考文献には挙げづらい. 単純に微分方程式の解き方とかを知りたいなら日本語でもいくつも教科書があると思うので大学の授業でよく使われるものを選ぶのが無難だと思う. 例えば補遺 @ref(ODE) で言及した **?** の講義ノートは一般公開されている上にかなり充実した内容だが, タイトルの通り偏微分方程式がメインなので微分方程式を全く知らない人が最初に読むと難しいかもしれない. というか今回の内容の範囲だと最初のほうを読むだけで十分である.