Modelling the effects of public health interventions on COVID-19 transmission using R

ANKUSH JIGYAS TANMAY

Installing Libraries

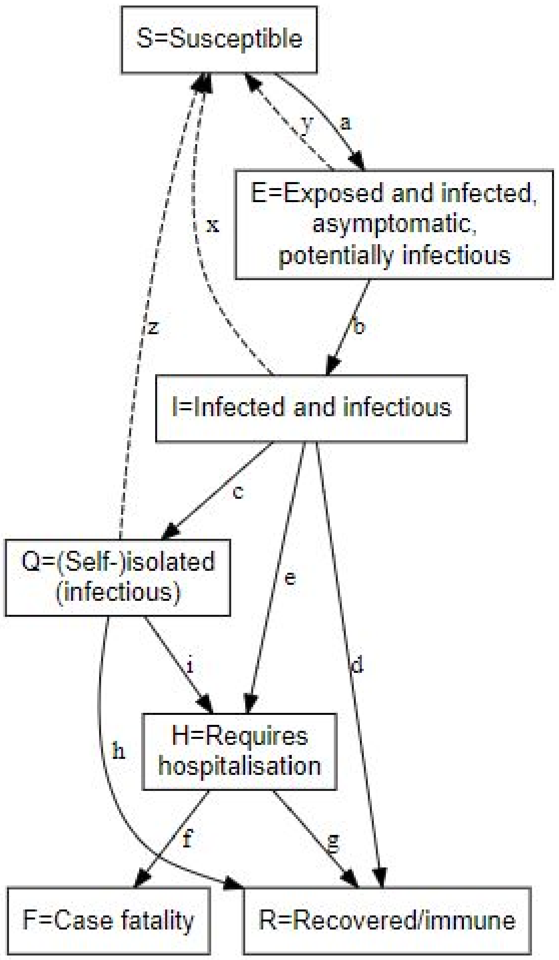
```
library(tidyverse)
library(magrittr)
library(lubridate)
library(stringr)
library(tibble)
library(broom)
library(ggplot2)
library(gt)
library(knitr)
library(devtools)
library(DiagrammeR)
library(parallel)
library(foreach)
library(tictoc)
suppressMessages(library(EpiModel))
library(incidence)
library(earlyR)
```

Giving source paths

```
source_files <- c("_icm.mod.init.seiqhrf.R",
"_icm.mod.status.seiqhrf.R",
    "_icm.mod.vital.seiqhrf.R", "_icm.control.seiqhrf.R",
"_icm.utils.seiqhrf.R",
    "_icm.saveout.seiqhrf.R", "_icm.icm.seiqhrf.R")
```

```
src_path <- paste0("Project-by-ankush-jigyas-tanmay")

gist_url <-
"https://gist.github.com/jigyas1810/92073d0ea75cfbd387f91f7c6e624bd7"

local_source <- FALSE

for (source_file in source_files) {
    if (local_source) {
        source(paste(src_path, source_file, sep = ""))
    } else {
        source_gist(gist_url, filename = source_file)
    }
}
```

Transition diagram


S=Susceptible
y
a
E=Exposed and infected, asymptomatic, potentially infectious
x
z
b
I=Infected and infectious
c
Q=(Self-)isolated (infectious)
e
d
i
h
H=Requires hospitalisation
f
g
F=Case fatality
R=Recovered/immune

Simulation parameters

DiagramRef	Parameter	Default	Explanation
	type	SEIQHRF	Type of model: SI, SIR, SIS, SEIR, SEIQHR and SEIQHRF available, but only SEIQHRF is likely to work in the current version of the code.
	nsteps	366	Number of days for simulation. Note that day 1 is for initialisation, day 2 is the first day of the simulation, hence default of 366 for 1 year.
	nsims	10	Number of simulations to run and then average.
	ncores	10	Number of CPU cores to use for parallel execution.
b	prog.rand	FALSE	Method for progression from E compartment to I. If TRUE, random binomial draws at prog.rate, if FALSE, random draws from a Weibull distribution (yes, I know it should be a discrete Weibull distribution but it makes little difference and speed of computation matters), with parameters prog.dist.scale and prog.dist.shape
d,g,h	rec.rand	FALSE	Method for recovery transition from I, Q or H to R. If TRUE, random binomial draws at prog.rate, if FALSE, random draws from a random draws from a Weibull distribution, with parameters rec.dist.scale and rec.dist.shape

f	fat.rand	FALSE	Method for case fatality transition from H to F. If TRUE, random binomial draws at fat.rate.base, if FALSE, random sample with a sample fraction also given by fat.rate.base. However, if the current number of patients in the H (needs hospitalisation) compartment is above a hospital capacity level specified by hosp.cap, then the fatality rate is the mean of the base fatality rate weighted by the hospital capacity, plus a higher rate, specified by fat.rate.overcap, weighted by the balance of those requiring hospitalisation (but who can't be accommodated). By setting fat.rate.overcap higher, the effect of exceeding the capacity of the health care system can be simulated. There is also a coefficient fat.tcoeff for the fatality rates that increases them as a linear function of the number of days the individual has been in the H compartment. Use of the co-efficient better approximates the trapezoid survival time distribution typical of ICU patients.
c	quar.rand	FALSE	Method for self-isolation transition from I to Q. If TRUE, random binomial draws at quar.rate, if FALSE, random sample with a sample fraction also given by `quar.rate.
e,i	hosp.rand	FALSE	Method for transition from I or Q to H -- that is, from infectious or from self-isolated to requiring hospitalisation. If TRUE, random binomial draws at hosp.rate, if FALSE, random sample with a sample fraction also given by `hosp.rate.
e,i	disch.rand	FALSE	Method for transition from H to R -- that is, from requiring hospitalisation to recovered. If TRUE, random binomial draws at disch.rate, if FALSE, random sample with a sample fraction also given by disch.rate. Note that the only way out of the H compartment is recovery or death.
	infection.FUN	infection.seiqhrf.icm	No, being infected with SARS-CoV2 is not fun. Rather this is the the name of the function to implement infection processes. Use the default.

	departures.FUN	departures.seiqhrf.icm	Handles background demographics, specifically departures (deaths not due to the virus, and emigration). Use the default.
	arrivals.FUN	arrivals.icm	Handles background demographics, specifically arrivals (births and immigration). Uses the original EpiModel code currently. A replacement that implements modelling the arrival of infected individuals is under development -- but for now, all arrivals go into the S compartment.
	get_prev.FUN	get_prev.seiqhrf.icm	Utility function that collects prevalence and transition time data from each run and stores it away in the simulation result object. Use the default.
	s.num	9997	Initial number of *S compartment individuals in the simulated population. An overall population of 10,000 is a good compromise. A set of models will still take several minutes or more to run, in parallel.
	e.num	0	Initial number of E compartment individuals in the simulated population.
	i.num	3	Initial number of I compartment individuals in the simulated population.
	q.num	0	Initial number of Q compartment individuals in the simulated population.
	h.num	0	Initial number of H compartment individuals in the simulated population.
	r.num	0	Initial number of R compartment individuals in the simulated population.
	f.num	0	Initial number of F compartment individuals in the simulated population.

x	act.rate.i	10	The number of exposure events (*acts*) between infectious individuals in the I compartment and susceptible individuals in the S compartment, per day. It's stochastic, so the rate is an average, some individuals may have more or less. Note that not every exposure event results in infection - that is governed by the inf.prob.i parameters (see below). Reducing act.rate.i is equivalent to increasing social distancing by people in the I compartment.
x	inf.prob.i	0.05	Probability of passing on infection at each exposure event for interactions between infectious people in the I compartment and susceptibles in S. Reducing inf.prob.i is equivalent to increasing hygiene measures, such as not putting hands in eyes, nose or moth, use of hand sanitisers, wearing masks by the infected, and so on.
y	act.rate.e	10	The number of exposure events (*acts*) between infectious individuals in the E compartment and susceptible individuals in the S compartment, per day. Otherwise as for act.rate.i.
y	inf.prob.e	0.02	Probability of passing on infection at each exposure event for interactions between infectious people in the E compartment and susceptibles in S. Note the default is lower than for inf.prob.i reflecting the reduced infectivity of infected but asymptomatic people (the E compartment). Otherwise as for inf.exp.i.
z	act.rate.q	2.5	The number of exposure events (*acts*) between infectious individuals in the Q compartment (isolated, self or otherwise) and susceptible individuals in the S compartment, per day. Note the much lower rate than for the I and E compartments, reflecting the much greater degree of social isolation for someone in (self-)isolation. The exposure event rate is not zero for this group, just much less. Otherwise as for act.rate.i.

z	inf.prob.q	0.02	Probability of passing on infection at each exposure event for interactions between infectious people in the Q compartment and susceptibles in S. Note the default is lower than for inf.prob.i reflecting the greater care that self-isolated individuals will, on average, take regarding hygiene measures, such as wearing masks, to limit spread to others. Otherwise as for inf.exp.i.
c	quar.rate	1/30	Rate per day at which symptomatic (or tested positive), infected I compartment people enter self-isolation (Q compartment). Asymptomatic E compartment people can't enter self-isolation because they don't yet know they are infected. Default is a low rate reflecting low community awareness or compliance with self-isolation requirements or practices, but this can be tweaked when exploring scenarios.
e,i	hosp.rate	1/100	Rate per day at which symptomatic (or tested positive), infected I compartment people or self-isolated Q compartment people enter the state of requiring hospital care -- that is, become serious cases. A default rate of 1% per day with an average illness duration of about 10 days means a bit less than 10% of cases will require hospitalisation, which seems about right (but can be tweaked, of course).
g	disch.rate	1/15	Rate per day at which people needing hospitalisation recover.
b	prog.rate	1/10	Rate per day at which people who are infected but asymptomatic (E compartment) progress to becoming symptomatic (or test-positive), the I compartment. See prog.rand above for more details.
b	prog.dist.scale	5	Scale parameter for Weibull distribution for progression, see prog.rand for details.
b	prog.dist.shape	1.5	Shape parameter for Weibull distribution for progression, see prog.rand for details. Read up on

			the Weibull distribution before changing the default.
d	rec.rate	1/20	Rate per day at which people who are infected and symptomatic (I compartment) recover, thus entering the R compartment. See rec.rand above for more details.
d	rec.dist.scale	35	Scale parameter for Weibull distribution for recovery, see rec.rand for details.
d	rec.dist.shape	1.5	Shape parameter for Weibull distribution for recovery, see rec.rand for details. Read up on the Weibull distribution before changing the default.
f	fat.rate.base	1/50	Baseline mortality rate per day for people needing hospitalisation (deaths due to the virus). See fat.rand for more details.
f	hosp.cap	40	Number of available hospital beds for the modelled population. See fat.rand for more details.
f	fat.rate.overcap	1/25	Mortality rate per day for people needing hospitalisation but who can't get into hospital due to the hospitals being full (see hosp.cap and fat.rand). The default rate is twice that for those who do get into hospital.
f	fat.tcoeff	0.5	Time co-efficient for increasing mortality rate as time in the H compartment increases for each individual in it. See fat.rand for details.
	vital	TRUE	Enables demographics, that is, arrivals and departures, to and from the simulated population.
	a.rate	(10.5/365)/1000	Background demographic arrival rate. Currently all arrivals go into the S compartment, the default is approximately the daily birth rate for Australia. Will be extended to cover immigration in future versions.
	ds.rate, de.rate, de.rate,	various rates	Background demographic departure (death not due to virus) rates. Defaults based on Australian

dq.rate, dh.rate, dr.rate		crude death rates. Can be used to model emigration as well as deaths.
out	mean	Summary function for the simulation runs. median is also available, or percentiles, see the EpiModel documentation.

Time-variant parameters

```
nsteps=366
quar.rate = c(rep(1/10,30), rep(1/3, 335))
```

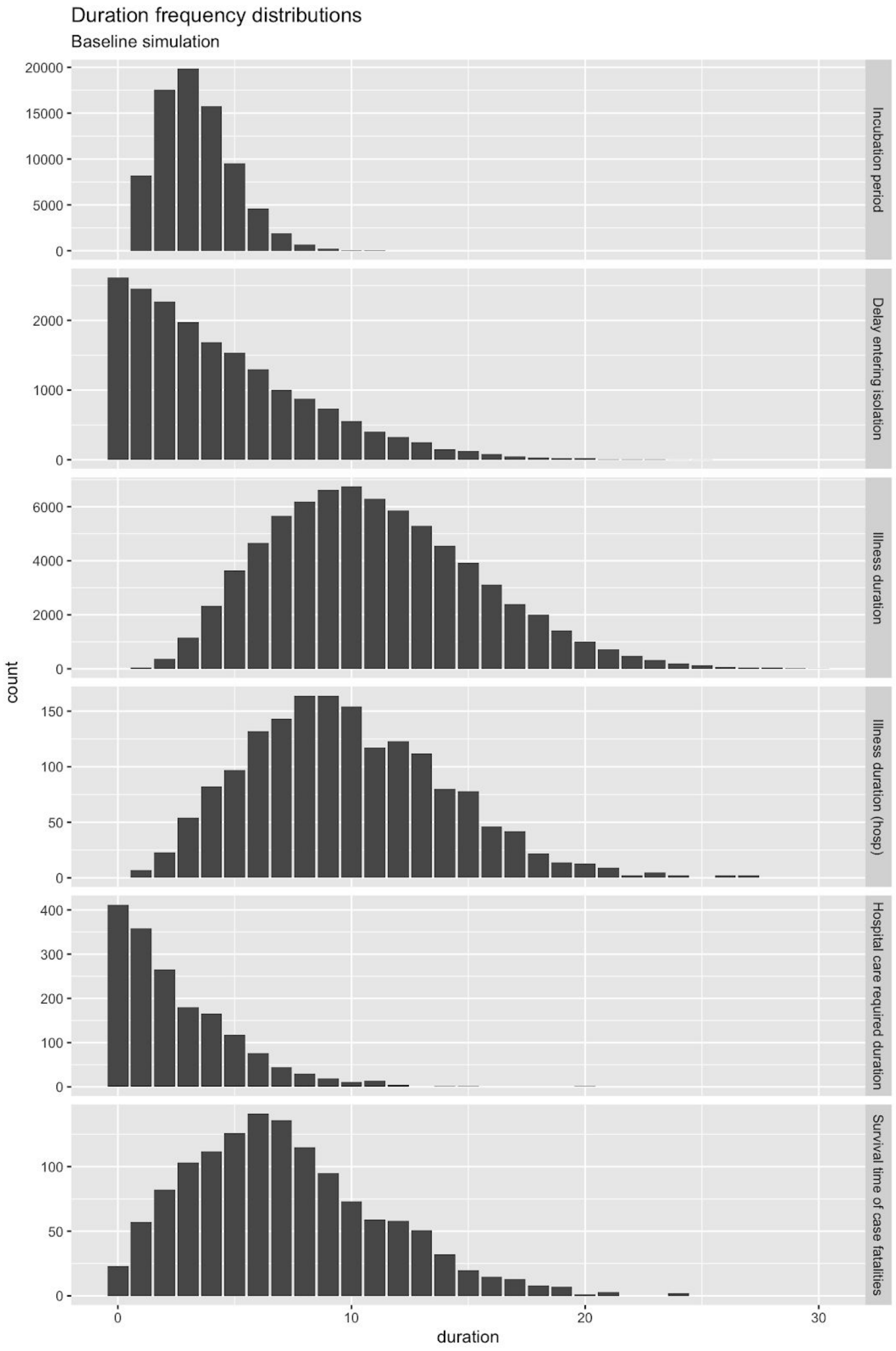
simulate() wrapper function

```
# function to set-up and run the baseline simulations
simulate <- function(# control.icm params
                     type = "SEIQHRF",
                     nsteps = 366,
                     nsims = 8,
                     ncores = 4,
                     prog.rand = FALSE,
                     rec.rand = FALSE,
                     fat.rand = TRUE,
                     quar.rand = FALSE,
                     hosp.rand = FALSE,
                     disch.rand = TRUE,
                     infection.FUN = infection.seiqhrf.icm,
                     recovery.FUN = progress.seiqhrf.icm,
                     departures.FUN = departures.seiqhrf.icm,
                     arrivals.FUN = arrivals.icm,
                     get_prev.FUN = get_prev.seiqhrf.icm,
                     # init.icm params
                     s.num = 9997,
                     e.num=0,
                     i.num = 3,
                     q.num=0,
                   h.num=0,
                   r.num = 0,
                   f.num = 0,
                   # param.icm params
                   inf.prob.e = 0.02,
                   act.rate.e = 10,
                   inf.prob.i = 0.05,
                   act.rate.i = 10,
                   inf.prob.q = 0.02,
                   act.rate.q = 2.5,
                   quar.rate = 1/30,
                   hosp.rate = 1/100,
                   disch.rate = 1/15,
                   prog.rate = 1/10,
                   prog.dist.scale = 5,
                   prog.dist.shape = 1.5,
                   rec.rate = 1/20,
                   rec.dist.scale = 35,
                   rec.dist.shape = 1.5,
                   fat.rate.base = 1/50,
                   hosp.cap = 40,
                   fat.rate.overcap = 1/25,
                   fat.tcoeff = 0.5,
                   vital = TRUE,
                   a.rate = (10.5/365)/1000,
                   a.prop.e = 0.01,
                   a.prop.i = 0.001,
                   a.prop.q = 0.01,
                   ds.rate = (7/365)/1000,
                   de.rate = (7/365)/1000,
                   di.rate = (7/365)/1000,
                   dq.rate = (7/365)/1000,
                   dh.rate = (20/365)/1000,
                   dr.rate = (7/365)/1000,
                   out="mean"
                  ) {

control <- control.icm(type = type,
                       nsteps = nsteps,
                       nsims = nsims,
                       ncores = ncores,
                       prog.rand = prog.rand,
                       rec.rand = rec.rand,
                       infection.FUN = infection.FUN,
                       recovery.FUN = recovery.FUN,
                       arrivals.FUN = arrivals.FUN,
                       departures.FUN = departures.FUN,
                       get_prev.FUN = get_prev.FUN)

init <- init.icm(s.num = s.num,
                 e.num = e.num,
                   i.num = i.num,
                   q.num = q.num,
                   h.num = h.num,
                   r.num = r.num,
                   f.num = f.num)

  param <-  param.icm(inf.prob.e = inf.prob.e,
                      act.rate.e = act.rate.e,
                      inf.prob.i = inf.prob.i,
                      act.rate.i = act.rate.i,
                      inf.prob.q = inf.prob.q,
                      act.rate.q = act.rate.q,
                      quar.rate = quar.rate,
                      hosp.rate = hosp.rate,
                      disch.rate = disch.rate,
                      prog.rate = prog.rate,
                      prog.dist.scale = prog.dist.scale,
                      prog.dist.shape = prog.dist.shape,
                      rec.rate = rec.rate,
                      rec.dist.scale = rec.dist.scale,
                      rec.dist.shape = rec.dist.shape,
                      fat.rate.base = fat.rate.base,
                      hosp.cap = hosp.cap,
                      fat.rate.overcap = fat.rate.overcap,
                      fat.tcoeff = fat.tcoeff,
                      vital = vital,
                      a.rate = a.rate,
                      a.prop.e = a.prop.e,
                      a.prop.i = a.prop.i,
                      a.prop.q = a.prop.q,
                      ds.rate = ds.rate,
                      de.rate = de.rate,
                      di.rate = di.rate,
                      dq.rate = dq.rate,
                      dh.rate = dh.rate,
                      dr.rate = dr.rate)

  sim <- icm.seiqhrf(param, init, control)
  sim_df <- as.data.frame(sim, out=out)

  return(list(sim=sim, df=sim_df))
}
```

Running simulations

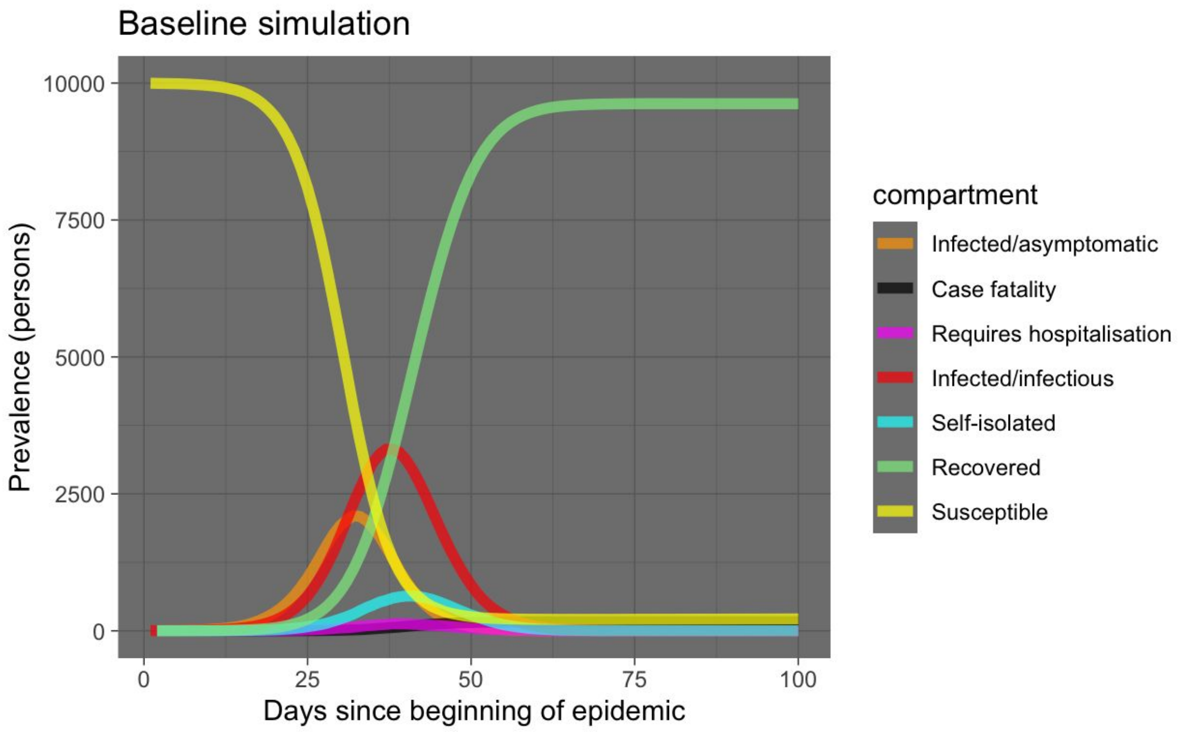
```
baseline_sim <- simulate(ncores = 4)
# define a function to extract timings and assemble a data
# frame
```

```
get_times <- function(simulate_results) {

    sim <- simulate_results$sim

    for (s in 1:sim$control$nsims) {
        if (s == 1) {
            times <- sim$times[[paste("sim", s, sep = "")]]
            times <- times %>% mutate(s = s)
        } else {
            times <- times %>% bind_rows(sim$times[[paste("sim",
                s, sep = "")]] %>% mutate(s = s))
        }
    }

    times <- times %>% mutate(infTime = ifelse(infTime < 0, -5,
        infTime), expTime = ifelse(expTime < 0, -5, expTime)) %>%
        mutate(incubation_period = infTime - expTime, illness_duration
= recovTime -
            expTime, illness_duration_hosp = dischTime - expTime,
            hosp_los = dischTime - hospTime, quarantine_delay =
quarTime -
                infTime, survival_time = fatTime - infTime) %>%
        select(s, incubation_period, quarantine_delay,
illness_duration,
            illness_duration_hosp, hosp_los, survival_time) %>%
        pivot_longer(-s, names_to = "period_type", values_to =
"duration") %>%
        mutate(period_type = factor(period_type, levels =
c("incubation_period",
            "quarantine_delay", "illness_duration",
"illness_duration_hosp",
            "hosp_los", "survival_time"), labels = c("Incubation
period",
            "Delay entering isolation", "Illness duration", "Illness
duration (hosp)",
            "Hospital care required duration", "Survival time of case
fatalities"),
            ordered = TRUE))
    return(times)
}

times <- get_times(baseline_sim)

times %>% filter(duration <= 30) %>% ggplot(aes(x = duration)) +
    geom_bar() + facet_grid(period_type ~ ., scales = "free_y") +
    labs(title = "Duration frequency distributions", subtitle =
"Baseline simulation")
```

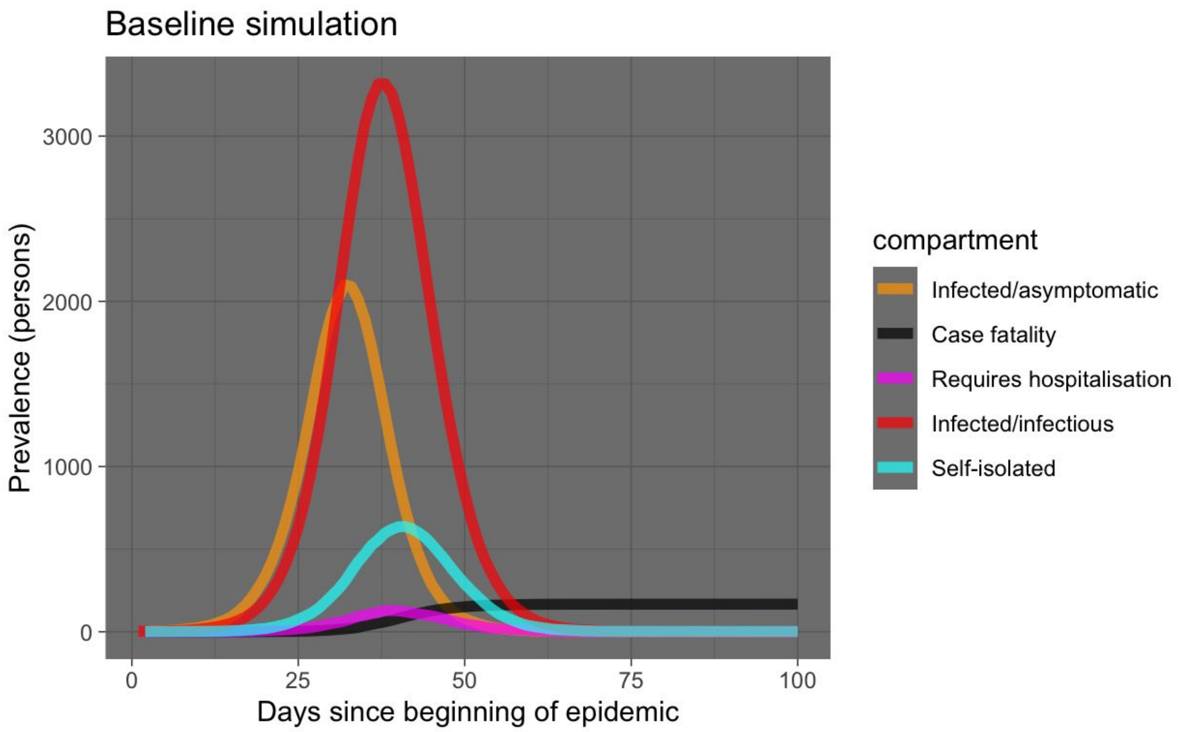

Duration frequency distributions
Baseline simulation
count
duration
0
10
20
30
20000
15000
10000
5000
0
Incubation period
2000
1000
0
Delay entering isolation
6000
4000
2000
0
Illness duration
150
100
50
0
Illness duration (hosp)
400
300
200
100
0
Hospital care required duration
100
50
0
Survival time of case fatalities

Visualise prevalence

```
baseline_plot_df <- baseline_sim$df %>% # use only the prevalence
columns
select(time, s.num, e.num, i.num, q.num, h.num, r.num, f.num) %>%
    # examine only the first 100 days since it is all over by
# then using the default parameters
filter(time <= 100) %>% pivot_longer(-c(time), names_to =
"compartment",
    values_to = "count")

# define a standard set of colours to represent compartments
compcols <- c(s.num = "yellow", e.num = "orange", i.num = "red",
    q.num = "cyan", h.num = "magenta", r.num = "lightgreen",
    f.num = "black")
complabels <- c(s.num = "Susceptible", e.num = "Infected/asymptomatic",
    i.num = "Infected/infectious", q.num = "Self-isolated", h.num =
"Requires hospitalisation",
    r.num = "Recovered", f.num = "Case fatality")

baseline_plot_df %>% ggplot(aes(x = time, y = count, colour =
compartment)) +
    geom_line(size = 2, alpha = 0.7) + scale_colour_manual(values =
compcols,
    labels = complabels) + theme_dark() + labs(title = "Baseline
simulation",
    x = "Days since beginning of epidemic", y = "Prevalence (persons)")
```

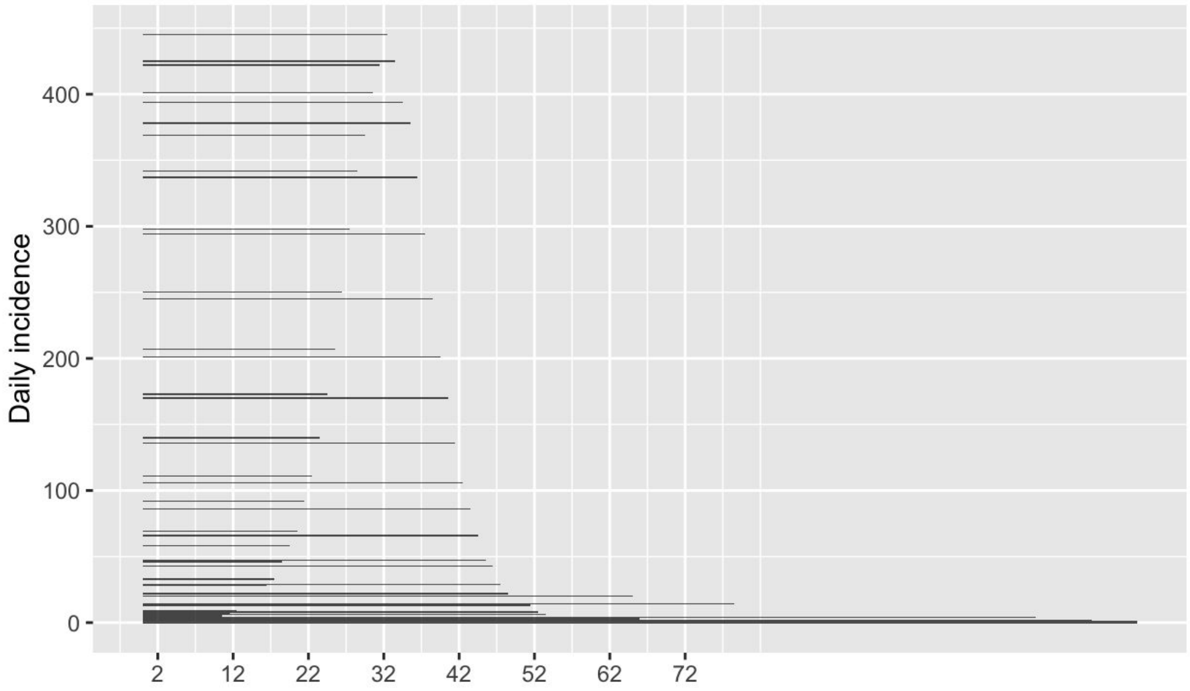


```
baseline_plot_df %>% filter(compartment %in% c("e.num", "i.num",
    "q.num", "h.num", "f.num")) %>% ggplot(aes(x = time, y = count,
    colour = compartment)) + geom_line(size = 2, alpha = 0.7) +
    scale_colour_manual(values = compcols, labels = complabels) +
    theme_dark() + labs(title = "Baseline simulation", x = "Days since
beginning of epidemic",
    y = "Prevalence (persons)")
```

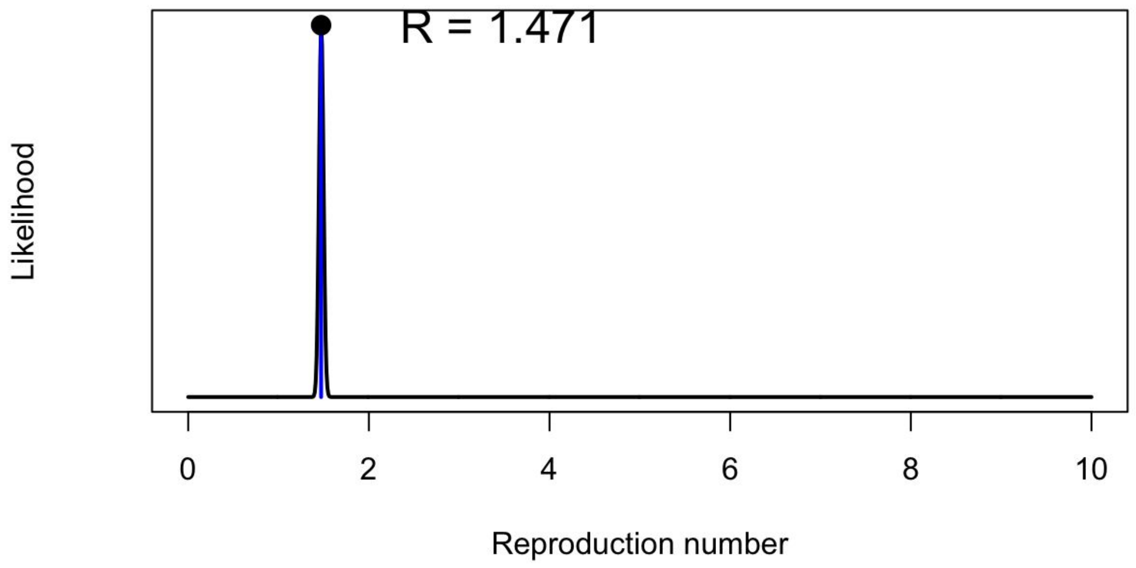


Checking the basic reproduction number R0

```
# get the S-> E compartment flow, which is our daily
# incidence rate
incidence_counts <- baseline_sim$df %>% select(time, se.flow)
# uncount them
incident_case_dates <- incidence_counts %>% uncount(se.flow) %>%
    pull(time)
# convert to an incidence object
incidence_all <- incident_case_dates %>% incidence(.)

# plot the incidence curve
plot(incidence_all)
```

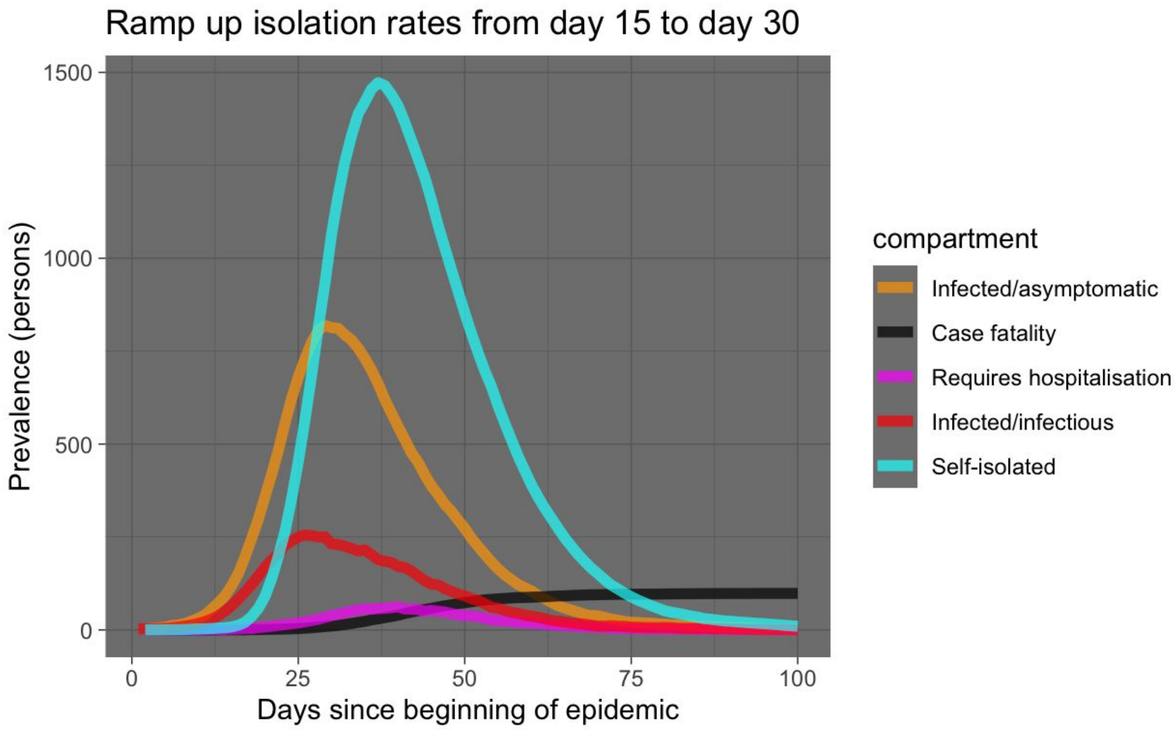


```
# find the peak of the epidemic curve
peak_of_epidemic_curve <- find_peak(incidence_all)

# repeat with just the growth part of the epidemic curve
incident_case_dates_growth_phase <- incidence_counts %>% filter(time <=
    peak_of_epidemic_curve) %>% select(time, se.flow) %>%
uncount(se.flow) %>%
    pull(time)

incidence_growth_phase <- incident_case_dates_growth_phase %>%
    incidence(., last_date = peak_of_epidemic_curve)
# specify serial interval mean and SD since the last blog
# post new studies have appeared suggesting 4.5 is a better
# mean for the SI
si_mean <- 4.5
si_sd <- 3.4

# get a set of MLE estimates for R0 and plot
res <- get_R(incidence_growth_phase, si_mean = si_mean, si_sd = si_sd)
plot(res, "R")
```

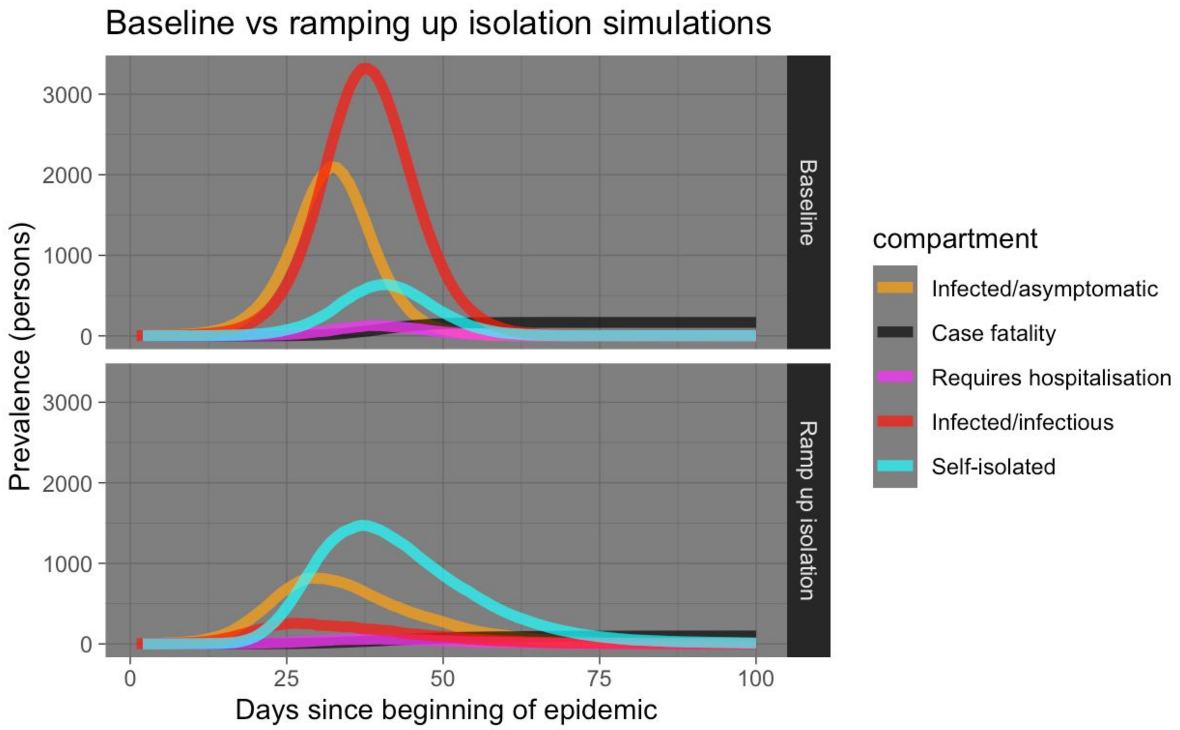


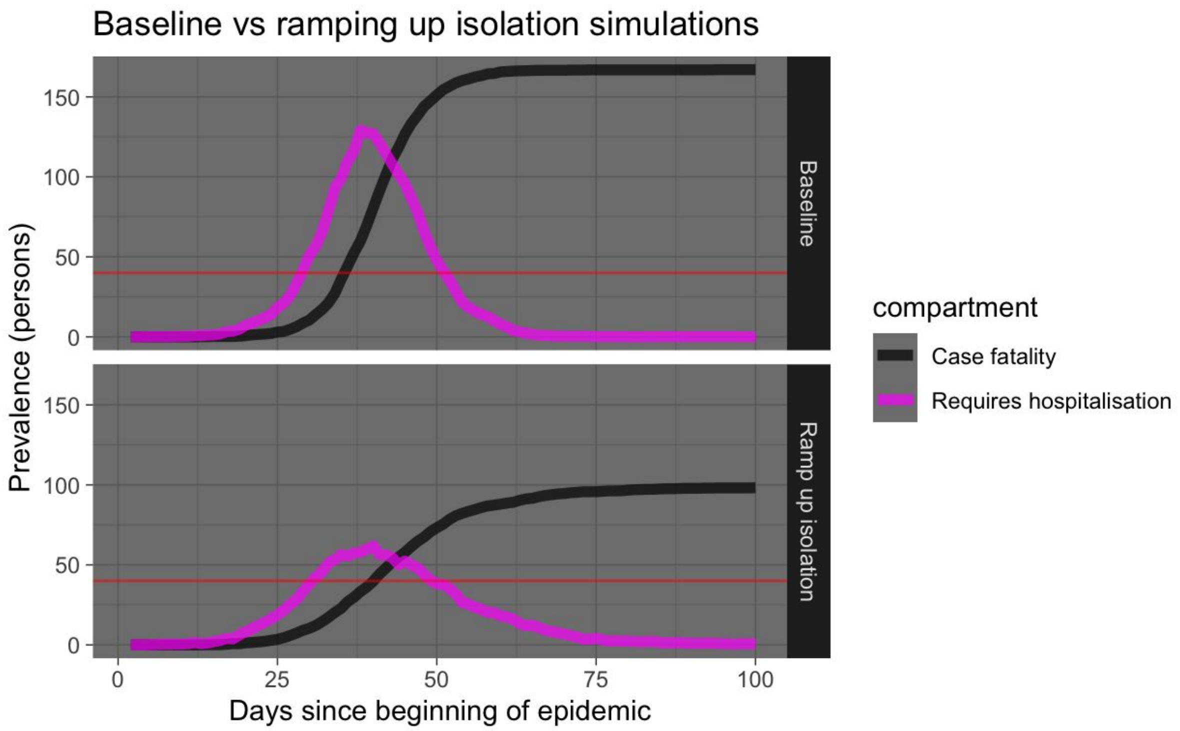
Running an intervention experiment

```
quar_rate_ramp <- function(t) {
    ifelse(t < 15, 0.0333, ifelse(t <= 30, 0.0333 + (t - 15) *
        (0.5 - 0.0333)/15, 0.5))
}

ramp_quar_rate_sim <- simulate(quar.rate = quar_rate_ramp(1:366))
```



```
baseline_plot_df %>% mutate(experiment = "Baseline") %>%
bind_rows(ramp_quar_rate_sim_plot_df %>%
    mutate(experiment = "Ramp up isolation")) %>% filter(compartment
%in%
    c("e.num", "i.num", "q.num", "h.num", "f.num")) %>% ggplot(aes(x =
time,
    y = count, colour = compartment)) + geom_line(size = 2, alpha =
0.7) +
    facet_grid(experiment ~ .) + scale_colour_manual(values = compcols,
    labels = complabels) + theme_dark() + labs(title = "Baseline vs
ramping up isolation simulations",
    x = "Days since beginning of epidemic", y = "Prevalence (persons)")
```


Baseline vs ramping up isolation simulations
Prevalence (persons)
3000
2000
1000
0
Baseline
Ramp up isolation
Days since beginning of epidemic
0
25
50
75
100
compartment
Infected/asymptomatic
Case fatality
Requires hospitalisation
Infected/infectious
Self-isolated


Baseline vs ramping up isolation simulations
Prevalence (persons)
150
100
50
0
Baseline
Ramp up isolation
Days since beginning of epidemic
0
25
50
75
100
compartment
Case fatality
Requires hospitalisation

Running lots of intervention experiments

Experiment 2 - more hospital beds

Over a four week period, let’s ramp up hospital capacity to triple the baseline level, starting at day 15. Hey, China built a 1000+ bed COVID-19 hospital in Wuhan in just 10 days…

```
hosp_cap_ramp <- function(t) {
    ifelse(t < 15, 40, ifelse(t <= 45, 40 + (t - 15) * (120 -
        40)/30, 120))
}

raise_hosp_cap_sim <- simulate(hosp.cap = hosp_cap_ramp(1:366))
```

Experiment 3 - more social distancing starting at day 15

Let’s step up social distancing (decrease exposure opportunities), starting at day 15, in everyone except the self-isolated, who are already practising it. But we’ll leave the self-isolation rate at the baseline desultory rate. The increase in social distancing will, when full ramped up by day 30, halve the number of exposure events between the infected and the susceptible each day.

```
social_distancing_day15_ramp <- function(t) {
    ifelse(t < 15, 10, ifelse(t <= 30, 10 - (t - 15) * (10 -
        5)/15, 5))
}

t15_social_distancing_sim <- simulate(act.rate.i =
social_distancing_day15_ramp(1:366),
    act.rate.e = social_distancing_day15_ramp(1:366))
```

Experiment 4 - more social distancing but starting at day 30

Let's repeat that, but we'll delay starting the social distancing ramp-up until day 30.

```
social_distancing_day30_ramp <- function(t) {
    ifelse(t < 30, 10, ifelse(t <= 45, 10 - (t - 30) * (10 -
        5)/15, 5))
}

t30_social_distancing_sim <- simulate(act.rate.i =
social_distancing_day30_ramp(1:366),
    act.rate.e = social_distancing_day30_ramp(1:366))
```

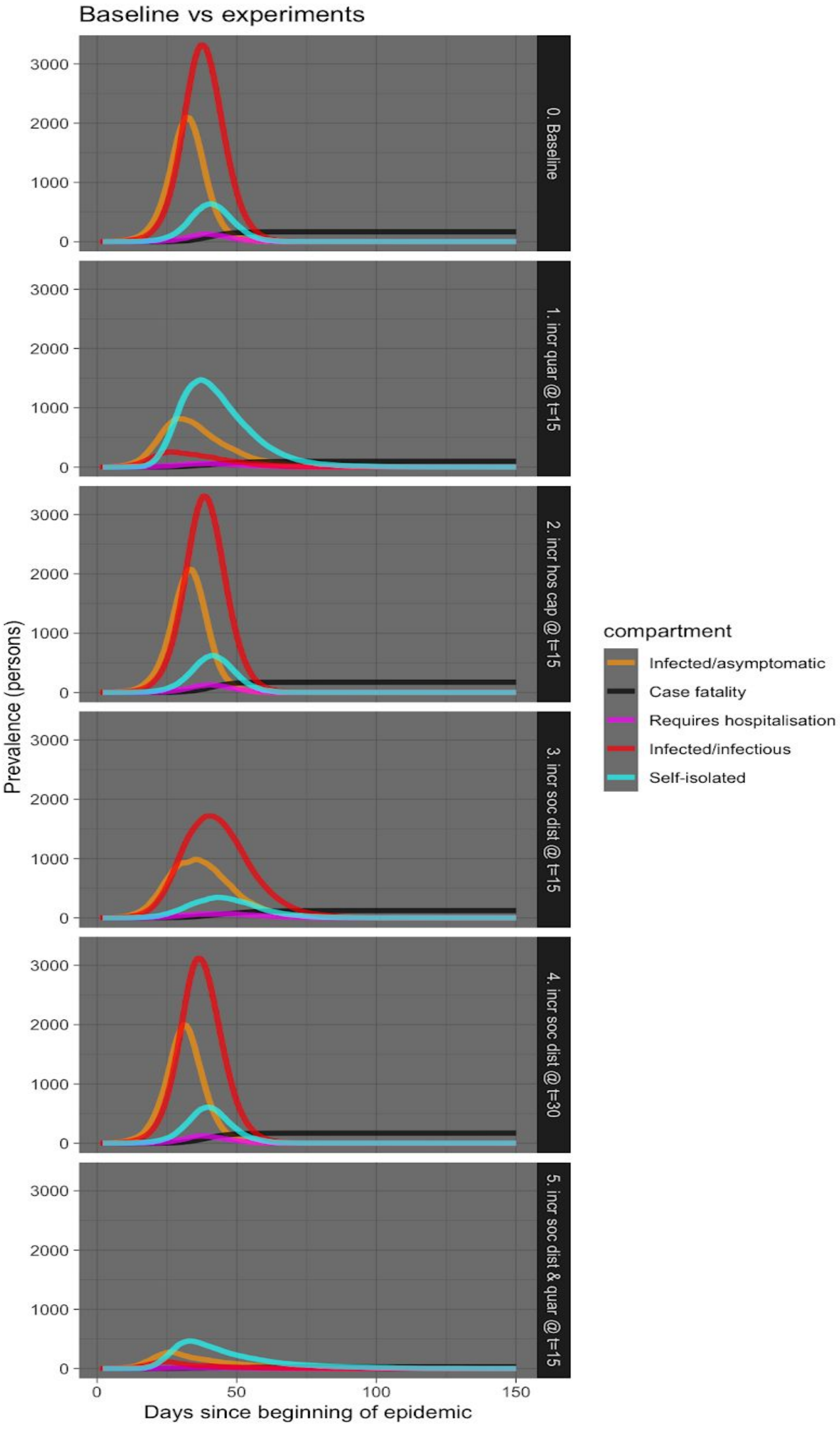
Experiment 5 - increase both social distancing and increased self-isolation rates starting day 15

```
quar_rate_ramp <- function(t) {
    ifelse(t < 15, 0.0333, ifelse(t <= 30, 0.0333 + (t - 15) *
        (0.5 - 0.0333)/15, 0.5))
}

ramp_quar_rate_sim <- simulate(quar.rate = quar_rate_ramp(1:366))
t15_soc_dist_quar_sim <- simulate(act.rate.i =
social_distancing_day15_ramp(1:366),
    act.rate.e = social_distancing_day15_ramp(1:366), quar.rate =
quar_rate_ramp(1:366))
```

Comparing experiments

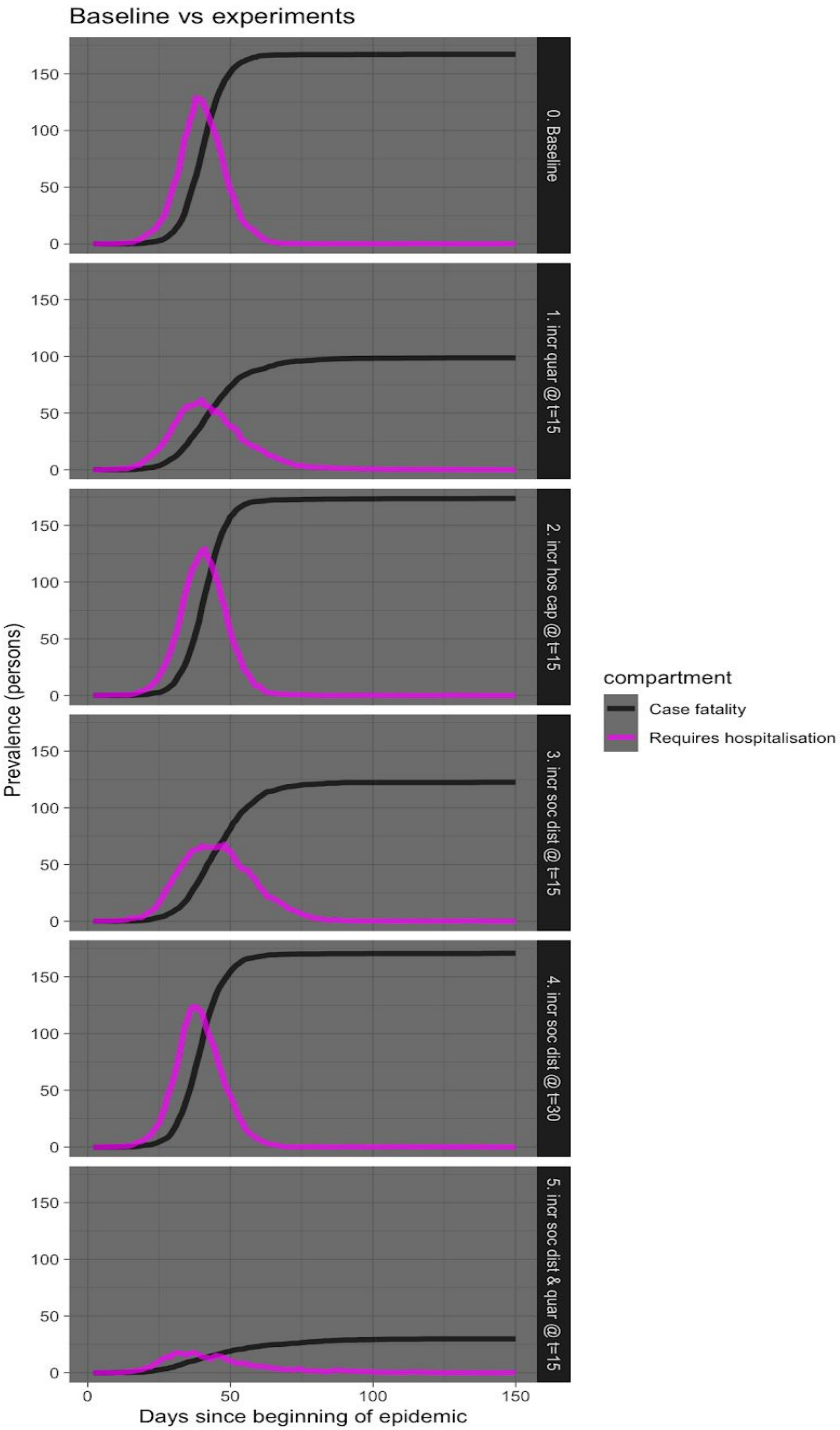
Let's visualise all those experiments in one plot.

```
plot_df <- baseline_sim$df %>% select(time, s.num, e.num, i.num,
    q.num, h.num, r.num, f.num) %>% mutate(experiment = "0. Baseline")
%>%
    bind_rows(ramp_quar_rate_sim$df %>% select(time, s.num, e.num,
        i.num, q.num, h.num, r.num, f.num) %>% mutate(experiment = "1.
incr quar @ t=15")) %>%
    bind_rows(raise_hosp_cap_sim$df %>% select(time, s.num, e.num,
        i.num, q.num, h.num, r.num, f.num) %>% mutate(experiment = "2.
incr hos cap @ t=15")) %>%
    bind_rows(t15_social_distancing_sim$df %>% select(time, s.num,
        e.num, i.num, q.num, h.num, r.num, f.num) %>% mutate(experiment
= "3. incr soc dist @ t=15")) %>%
    bind_rows(t30_social_distancing_sim$df %>% select(time, s.num,
        e.num, i.num, q.num, h.num, r.num, f.num) %>% mutate(experiment
= "4. incr soc dist @ t=30")) %>%
    bind_rows(t15_soc_dist_quar_sim$df %>% select(time, s.num,
        e.num, i.num, q.num, h.num, r.num, f.num) %>% mutate(experiment
= "5. incr soc dist & quar @ t=15")) %>%
    filter(time <= 150) %>% pivot_longer(-c(time, experiment),
    names_to = "compartment", values_to = "count") %>%
filter(compartment %in%
    c("e.num", "i.num", "q.num", "h.num", "f.num"))

plot_df %>% ggplot(aes(x = time, y = count, colour = compartment)) +
    geom_line(size = 1.5, alpha = 0.7) + facet_grid(experiment ~
    .) + scale_colour_manual(values = compcols, labels = complabels) +
    theme_dark() + labs(title = "Baseline vs experiments", x = "Days
since beginning of epidemic",
    y = "Prevalence (persons)")
```


Baseline vs experiments
Prevalence (persons)
3000
2000
1000
0
0. Baseline
1. incr quar @ t=15
2. incr hos cap @ t=15
3. incr soc dist @ t=15
4. incr soc dist @ t=30
5. incr soc dist & quar @ t=15
0
50
100
150
Days since beginning of epidemic
compartment
Infected/asymptomatic
Case fatality
Requires hospitalisation
Infected/infectious
Self-isolated

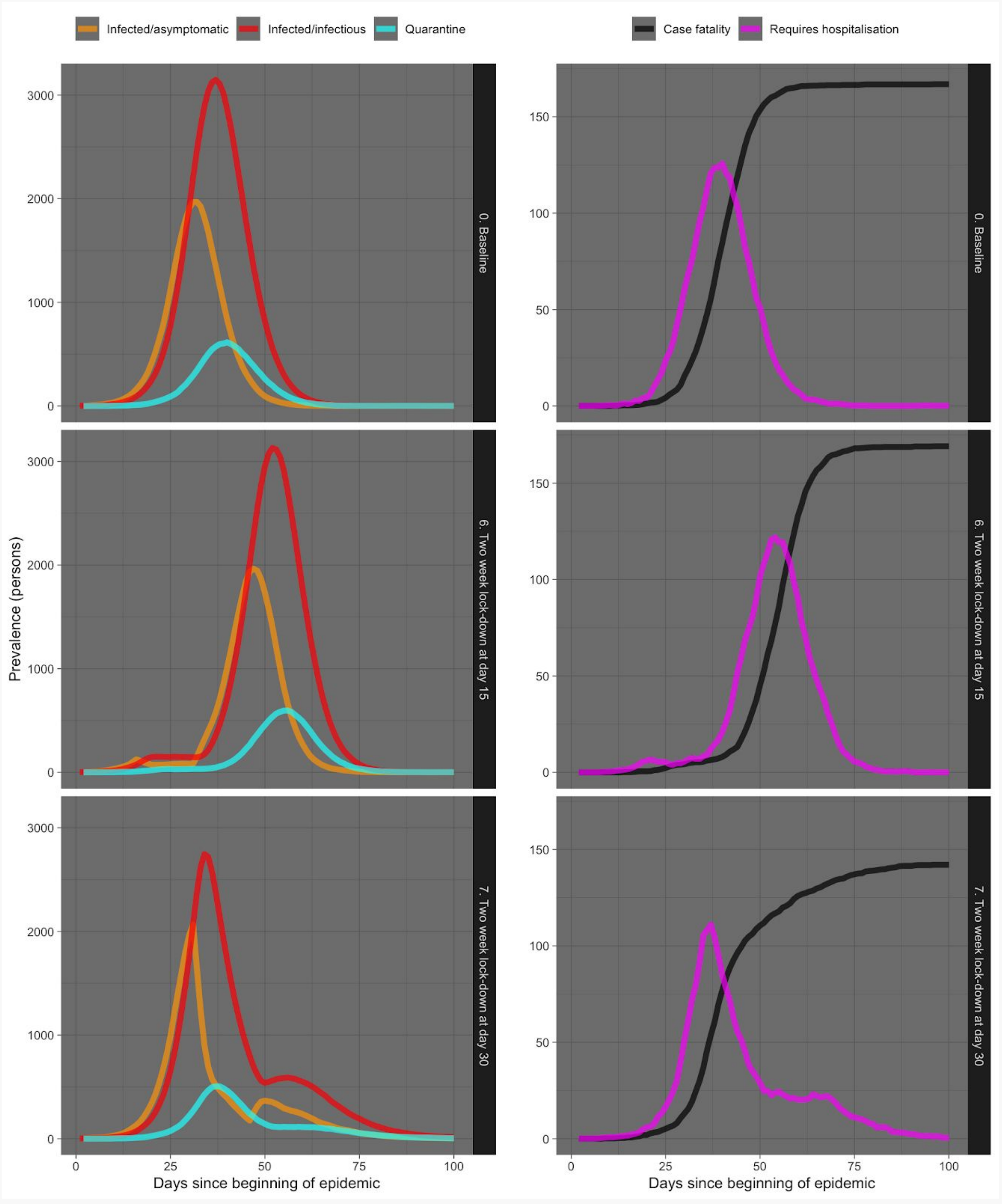
Let’s see that again showing just the *requiring hospitalisation* and *case fatality* prevalence numbers.

```
plot_df %>% filter(compartment %in% c("h.num", "f.num")) %>%
    ggplot(aes(x = time, y = count, colour = compartment)) +
    geom_line(size = 1.5, alpha = 0.7) + facet_grid(experiment ~
    .) + scale_colour_manual(values = compcols, labels = complabels) +
    theme_dark() + labs(title = "Baseline vs experiments", x = "Days
since beginning of epidemic",
    y = "Prevalence (persons)")
```


Baseline vs experiments
Prevalence (persons)
0
50
100
150
0. Baseline
1. incr quar @ t=15
2. incr hos cap @ t=15
3. incr soc dist @ t=15
4. incr soc dist @ t=30
5. incr soc dist & quar @ t=15
0
50
100
150
Days since beginning of epidemic
compartment
Case fatality
Requires hospitalisation

Experiment 6: 2 week lockdown

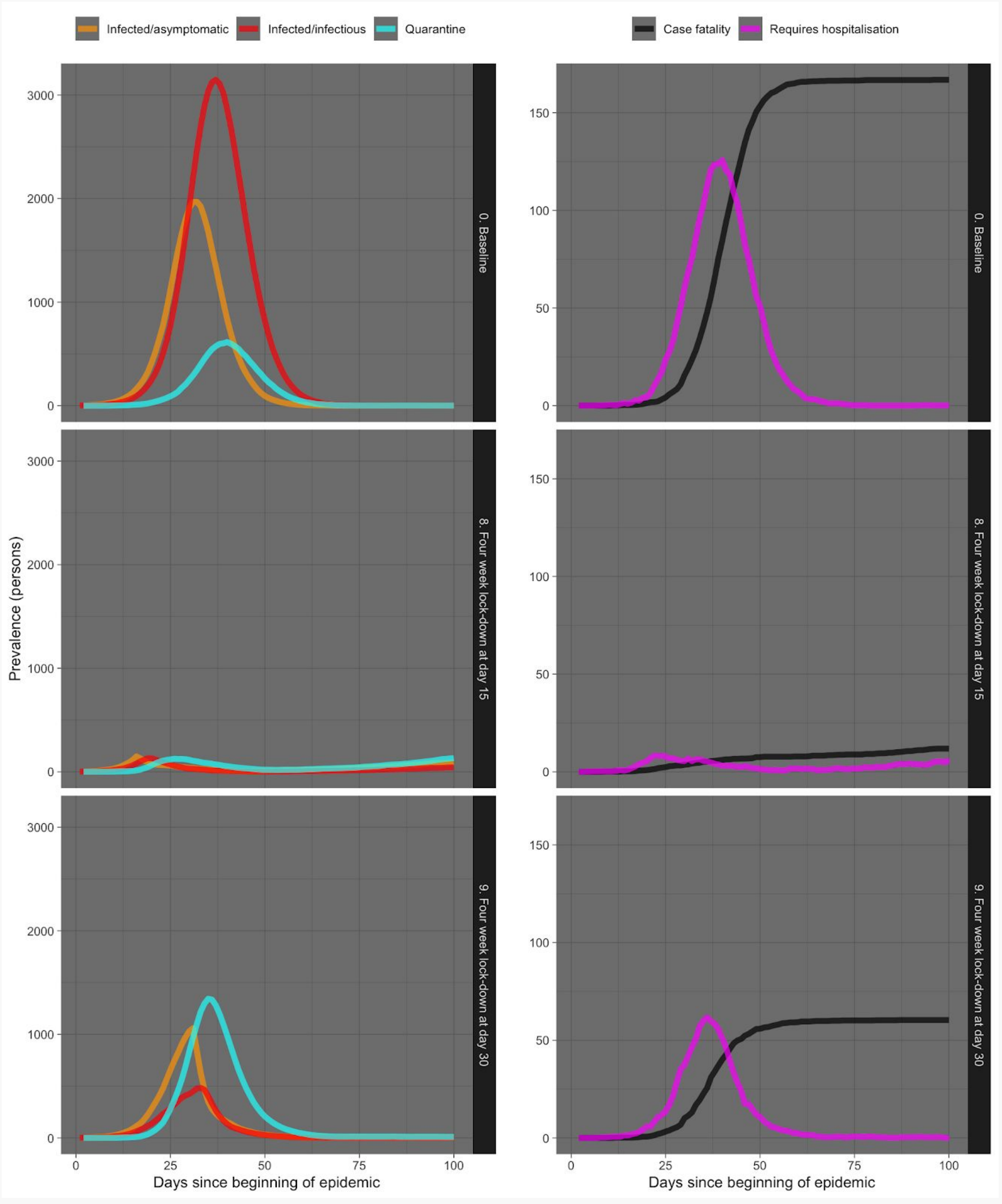
```
twoweek_lockdown_day15_vector <- c(rep(10, 15), rep(2.5, 15),
  rep(10, 336))
twoweek_lockdown_day30_vector <- c(rep(10, 30), rep(2.5, 15),
  rep(10, 321))

twoweek_lockdown_day15_sim <- simulate(act.rate.i =
twoweek_lockdown_day15_vector,
  act.rate.e = twoweek_lockdown_day15_vector)

twoweek_lockdown_day30_sim <- simulate(act.rate.i =
twoweek_lockdown_day30_vector,
  act.rate.e = twoweek_lockdown_day30_vector)
```



Infected/asymptomatic
Infected/infectious
Quarantine
Case fatality
Requires hospitalisation
Prevalence (persons)
0
1000
2000
3000
0
50
100
150
0. Baseline
6. Two week lock-down at day 15
7. Two week lock-down at day 30
0
25
50
75
100
Days since beginning of epidemic
Days since beginning of epidemic

Experiment 7: 4 weeks lockdown

```
fourweek_lockdown_day15_vector <- c(rep(10, 15), rep(2.5, 30),
  rep(7.5, 321))
fourweek_lockdown_day30_vector <- c(rep(10, 30), rep(2.5, 30),
  rep(7.5, 306))

fourweek_lockdown_day15_sim <- simulate(act.rate.i =
fourweek_lockdown_day15_vector,
  act.rate.e = fourweek_lockdown_day15_vector, quar.rate =
quarantine_ramp(1:366),
  inf.prob.q = 0.01)

fourweek_lockdown_day30_sim <- simulate(act.rate.i =
fourweek_lockdown_day30_vector,
  act.rate.e = fourweek_lockdown_day30_vector, quar.rate =
quarantine_ramp(1:366),
  inf.prob.q = 0.01)
```



Conclusions

Increasing hospital capacity is probably sensible, but may have little effect on fatalities if the numbers swamp available capacity by an order of magnitude. And increasing hospital capacity is not easy, nor can it be done swiftly, except perhaps in China.

Rigorous and swift self-isolation in those who are symptomatic is very effective, especially if done early, even in the presence of infectivity in the asymptomatic infected.

Increasing social distancing also works, but works much better if done early, possibly before there is an obvious need for it based on numbers infected or deaths. Implement early, implement hard!

A combination of prompt self-isolation plus moderate social distancing is also very effective, without necessarily killing the entire economy. This combination warrants further simulations.

The two week lock-down starting at day 15 isn't effective at all - it just stops the spread in its tracks for two weeks, and then it just resumes again. But a two-week lock-down starting at day 30 is somewhat more effective, presumably because there are more infected people being taken out of circulation from day 30 onwards. But the epidemic still partially bounces back after the two weeks are over. What this tells us is that single lock-downs for only two weeks aren't effective.

Whereas in Experiment 7: 4 week lockdown a shrinking in the curve is observed telling us that it can be effective at a very large level.