EpiEstim: Practical

Rob Challen

2024-03-12

We are going to use some UK COVID-19 data to estimate the reproduction number using infectivity profiles from the literature. We'll use a couple of data sets and compare.

Get some data

First of all lets get some retrospective COVID data from the UK coronavirus site. The API we used during COVID has been decommissioned now so we have bundled a cleaned copy of the data set here:

`https://github.com/bristol-vaccine-centre/edam-epiestim/raw/main/input/covid-timeseries.csv`

We are going to focus only on the first wave up to 1st Sept 2020:

- Read the link to the CSV file in using `readr::read_csv`.
- Filter the data set to $<$ 2020-09-01.
- Quickly explore the structure using `glimpse`.

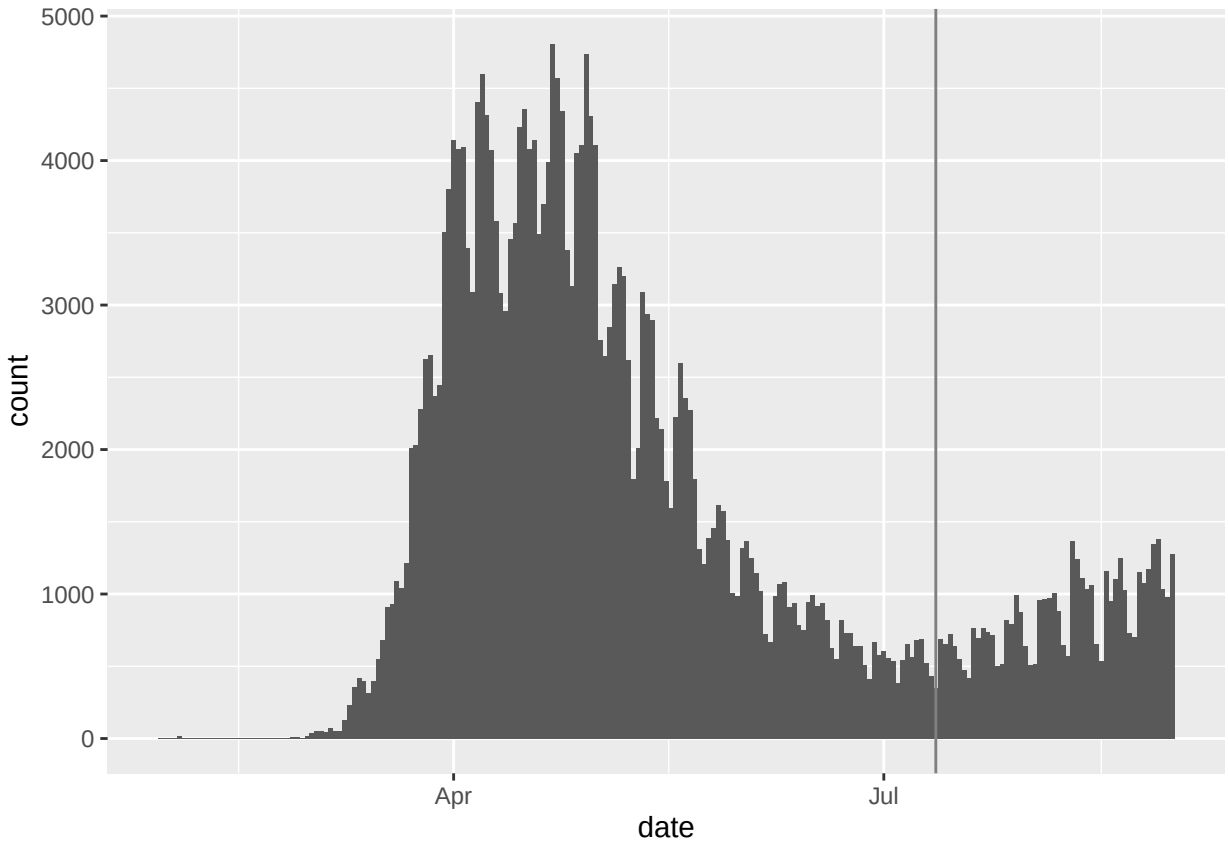
```
england_cases = readr::read_csv(
  "https://github.com/bristol-vaccine-centre/edam-epiestim/raw/main/input/covid-timeseries.csv"
  ) %>%
  filter(date < "2020-09-01")

england_cases %>% glimpse()
```

```
## Rows: 215
## Columns: 2
## $ date  <date> 2020-01-30, 2020-01-31, 2020-02-01, 2020-02-02, 2020-02-03, 202~
## $ count <dbl> 1, 0, 0, 1, 18, 0, 1, 0, 0, 3, 1, 1, 3, 1, 1, 0, 0, 0, 1, 0, 0, ~
```

We will use cases as a proxy for infections.

- Plot a timeline of the cases with `ggplot` (e.g. `geom_bar` with `stat=identity`)

```
ggplot(england_cases,aes(x=date,y=count))+
  geom_bar(stat="identity")+
  geom_vline(xintercept = as.Date("2020-07-12"), colour="grey50")
```



The epidemic clearly reached a low in late June, early July. Lets see if we can get `EpiEstim` to determine when the epidemic started growing again.

First we need:

- ✓ Incident cases
- ✗ Prior for R
- ✗ Window
- ✗ Infectivity profile (e.g. Generation time distribution)

Infectivity profile

We are going to use an estimate from the literature. We will use the central estimate of the generation interval from the singapore data from Ganyani et al 2020 (`https://pmc.ncbi.nlm.nih.gov/articles/PMC7201952/`). They model the generation interval as a gamma distributed quantity.

- Find the central estimates of mean and SD of the generation interval distribution from the paper.

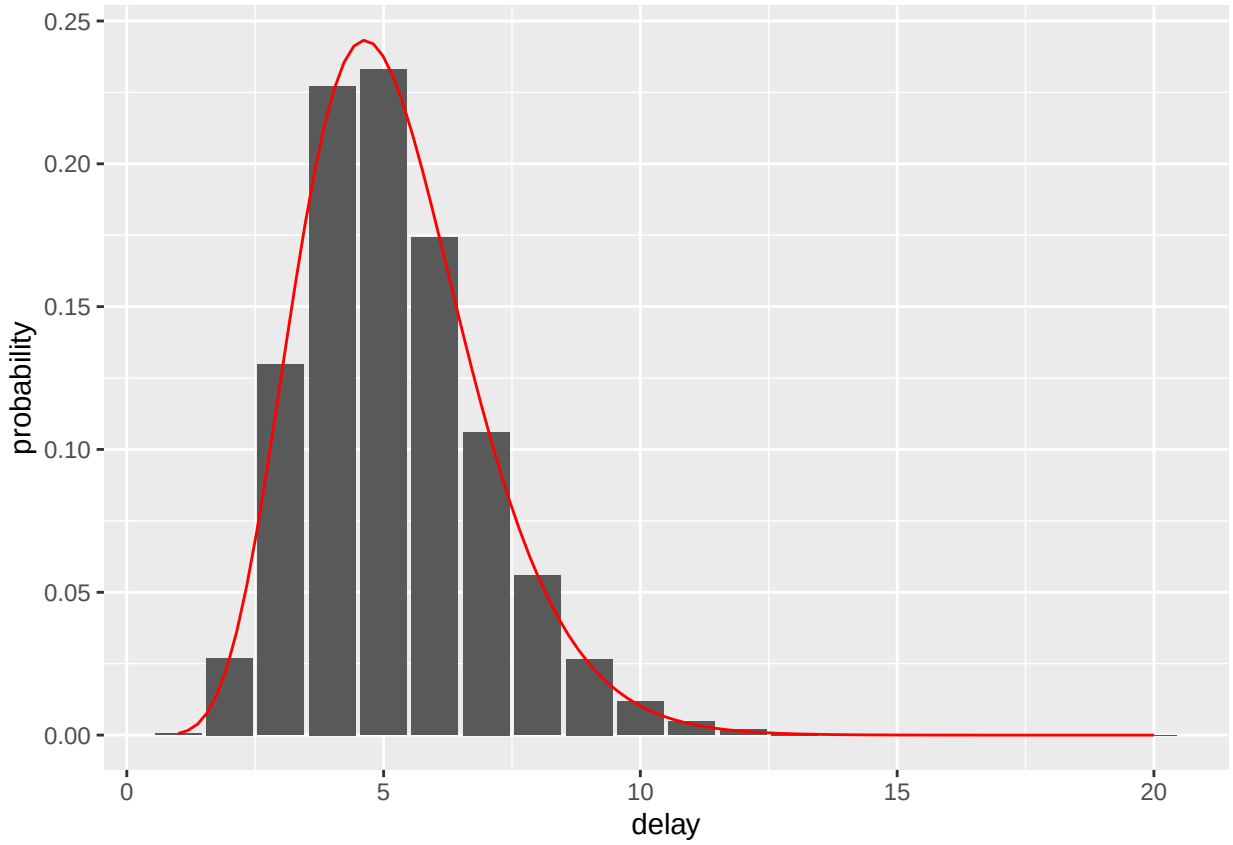
```
ganyani_mean_gi = 5.2 # credible intervals were 3.78 - 6.78
ganyani_sd_gi = 1.72    # credible intervals were 0.91 - 3.93
```

We can get EpiEstim to calculate a discrete infectivity profile distribution using the `EpiEstim::discr_si` function. N.B. EpiEstim calls the infectivity profile the "serial interval" throughout. We are using an estimate of the generation interval instead of a serial interval.

- Get a discrete infectivity profile distribution (for days 1 to 20).
- Plot it (plus the continuous version using `geom_function` if you are keen).
- Does the mean of the infectivity profile distribution equal that of the original Ganyani estimate?

$$mean = \sum x \times P(x)$$
$$std = \sqrt{\sum (x - mean)^2 \times P(x)}$$
$$shape = \frac{mean^2}{sd^2}$$
$$rate = \frac{mean}{sd^2}$$

```
si_probability = EpiEstim::discr_si(1:20, ganyani_mean_gi, ganyani_sd_gi)

shape = ganyani_mean_gi^2/ganyani_sd_gi^2
rate = ganyani_mean_gi/ganyani_sd_gi^2

ggplot(tibble(
    delay = 1:20,
    probability = si_probability
  ), aes(x=delay, y=probability))+
  geom_bar(stat="identity")+
  geom_function(fun = \(x) dgamma(x, shape, rate), colour = "red")
```




```
# Mean and SD of the discretised probability distribution:
mean_si = sum(1:20*si_probability)
sd_si = sqrt(sum((1:20-mean_si)^2*si_probability))

print(mean_si)
print(sd_si)
```

```
## [1] 5.199997
## [1] 1.767775
```

This is pretty close by N.B. we don't actually need the discrete distribution as EpiEstim does this internally when we use the method `"parametric_si"`.

EpiEstim uses a configuration object which you have to create with `EpiEstim::make_config`.

- Look at the help page (`?EpiEstim::make_config`), focussing on the `parametric_si` option.

We use a minimally informative prior for R (i.e. `mean_prior = 1` and `std_prior = 1`) and we will use a window of duration 7 days. `EpiEstim::make_config` needs this window in terms of a start and end index (beginning at 2… obvs):

```
window = 7
t_start = 2:(nrow(england_cases)-window)
t_end = t_start+window
```

We now have:

- ✓ Incident cases
- ✓ Prior for R
- ✓ Window
- ✓ Infectivity profile (e.g. Generation time distribution)

Assemble this into a configuration object using `EpiEstim::make_config` and the method `parametric_si`:

```
simple_config = EpiEstim::make_config(
  method = "parametric_si",
  mean_si = ganyani_mean_gi,
  std_si = ganyani_sd_gi,
  mean_prior = 1,
  std_prior = 1,
  t_start = t_start,
  t_end = t_end
)
```

Our incident cases are not quite in the right format. It is described in the help file `?EpiEstim::estimate_R`. Basically you need to rename the columns.

Use the configuration object with the renamed incident case data to get an estimate of Rt.

```
simple_ts = england_cases %>%
  select(dates = date, I = count) %>%
  arrange(dates)

epiestim_out = EpiEstim::estimate_R(simple_ts,method = "parametric_si", config=simple_config)

epiestim_out$R %>% glimpse()
```

```
## Rows: 207
## Columns: 11
## $ t_start             <dbl> 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16~
## $ t_end               <dbl> 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21,~
## $ `Mean(R)`           <dbl> 2.2048534, 1.7084275, 1.4283350, 1.2717674, 0.4748~
## $ `Std(R)`            <dbl> 0.4811385, 0.3487313, 0.2856670, 0.2543535, 0.1501~
## $ `Quantile.0.025(R)` <dbl> 1.36483902, 1.09462174, 0.92434312, 0.82302083, 0.~
## $ `Quantile.0.05(R)`  <dbl> 1.47746438, 1.17803471, 0.99309997, 0.88424087, 0.~
## $ `Quantile.0.25(R)`  <dbl> 1.8641464, 1.4621088, 1.2267137, 1.0922469, 0.3668~
## $ `Median(R)`         <dbl> 2.1699562, 1.6847588, 1.4093364, 1.2548513, 0.4591~
```

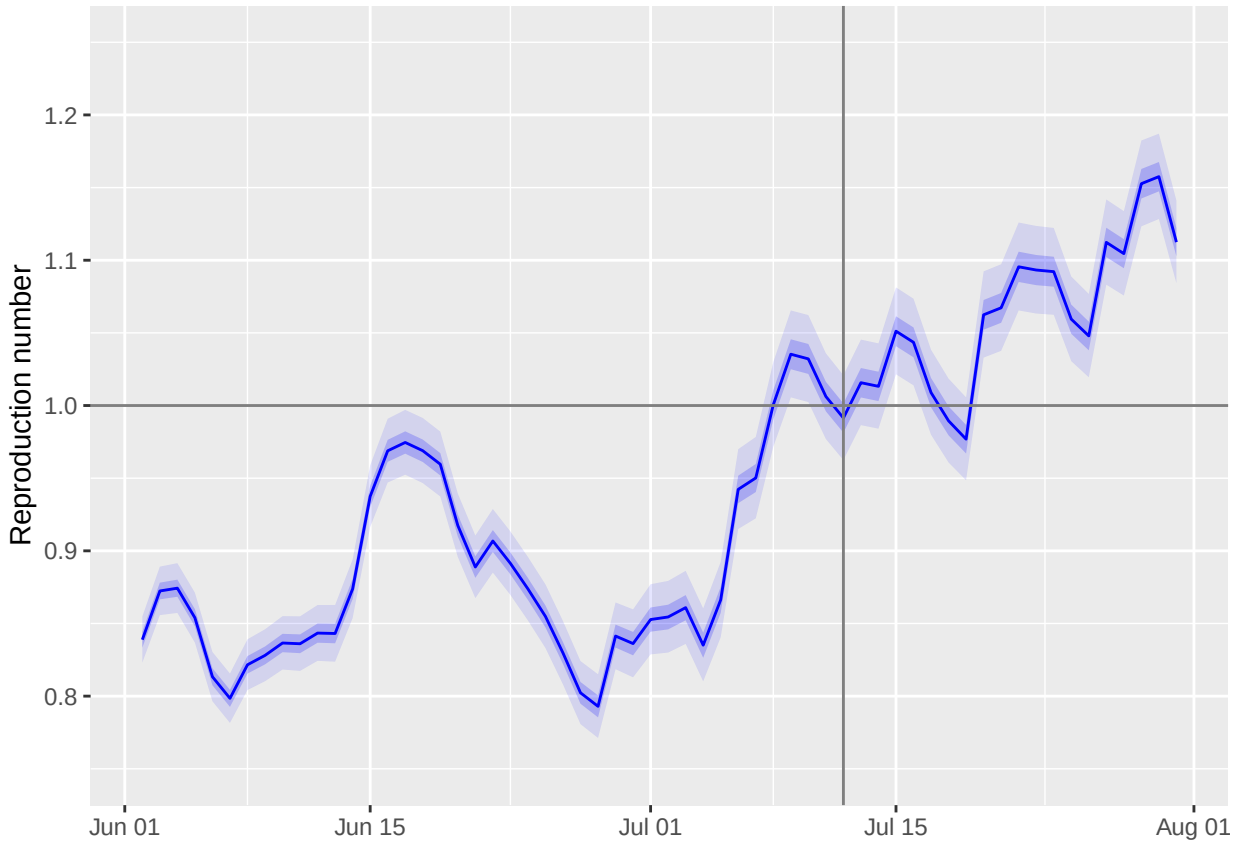
```
## $ `Quantile.0.75(R)`  <dbl> 2.5075615, 1.9289700, 1.6092652, 1.4328649, 0.5657~
## $ `Quantile.0.95(R)`  <dbl> 3.0513091, 2.3195736, 1.9283896, 1.7170083, 0.7458~
## $ `Quantile.0.975(R)` <dbl> 3.2430641, 2.4566684, 2.0402393, 1.8165976, 0.8113~
```

The output is a list with `R` and `dates` items. You almost always want to combine the two. By convention the estimate is assigned to the end of the window, but this is a matter of interpretation, as a key assumption of the Cori method is that Rt is fixed over the window.

```
plot_data = epiestim_out$R %>%
  mutate(date = epiestim_out$dates[t_end]) %>%
  filter(date > "2020-06-01" & date < "2020-08-01")
```

- Plot the time series of reproduction number estimates (`geom_ribbon` is useful for the confidence intervals):

```
type = "blue"
ggplot(plot_data)+
    geom_line(mapping=aes(x=date, y=`Mean(R)`, colour=type))+
    geom_ribbon(mapping=aes(x=date, ymin=`Quantile.0.25(R)`,ymax=`Quantile.0.75(R)`, fill=type), colour=
    geom_ribbon(mapping=aes(x=date, ymin=`Quantile.0.025(R)`,ymax=`Quantile.0.975(R)`, fill=type), colou
    geom_hline(yintercept = 1, colour="grey50")+
    geom_vline(xintercept = as.Date("2020-07-12"), colour="grey50")+
    scale_color_identity(aesthetics=c("fill","colour"))+
    xlab(NULL)+
    ylab("Reproduction number")+
    coord_cartesian(ylim=c(0.75,1.25))
```




```
# N.B. you can get EpiEstim to plot this for you:
# look at ?EpiEstim::plot.estimate_R
```

```
plot(epiestim_out,
  what="R",
  options_R = list(
    xlim=as.Date(c("2020-06-01", "2020-08-01")),
    ylim=c(0.75,1.25)
  )
)
```

- On the 12th July, if they had asked, what would you advise a senior government official who is thinking about stimulating the economy with a scheme that involves increasing contacts?
- What happens if you change the window? Rt prior?
- When would you have been confident that the epidemic was growing?

"Real time" cases

During the early pandemic there was no government dashboard. Public data was extremely limited, and published haphazardly. A community effort to collate the data emerged and you can still see the effort made (and frustrations encountered) scraping the data today:

`https://github.com/tomwhite/covid-19-uk-data/tree/master`

The last update to this data was Aug 2020. The data for England is in the `data` subdirectory. We can explore what this data looked like at any point in time by looking at the commit history:

`https://github.com/tomwhite/covid-19-uk-data/commits/master/data/covid-19-totals-england.csv`

Lets look at early July. I'll show you how to get there but the commit for the 12th July (a Sunday) can be found here:

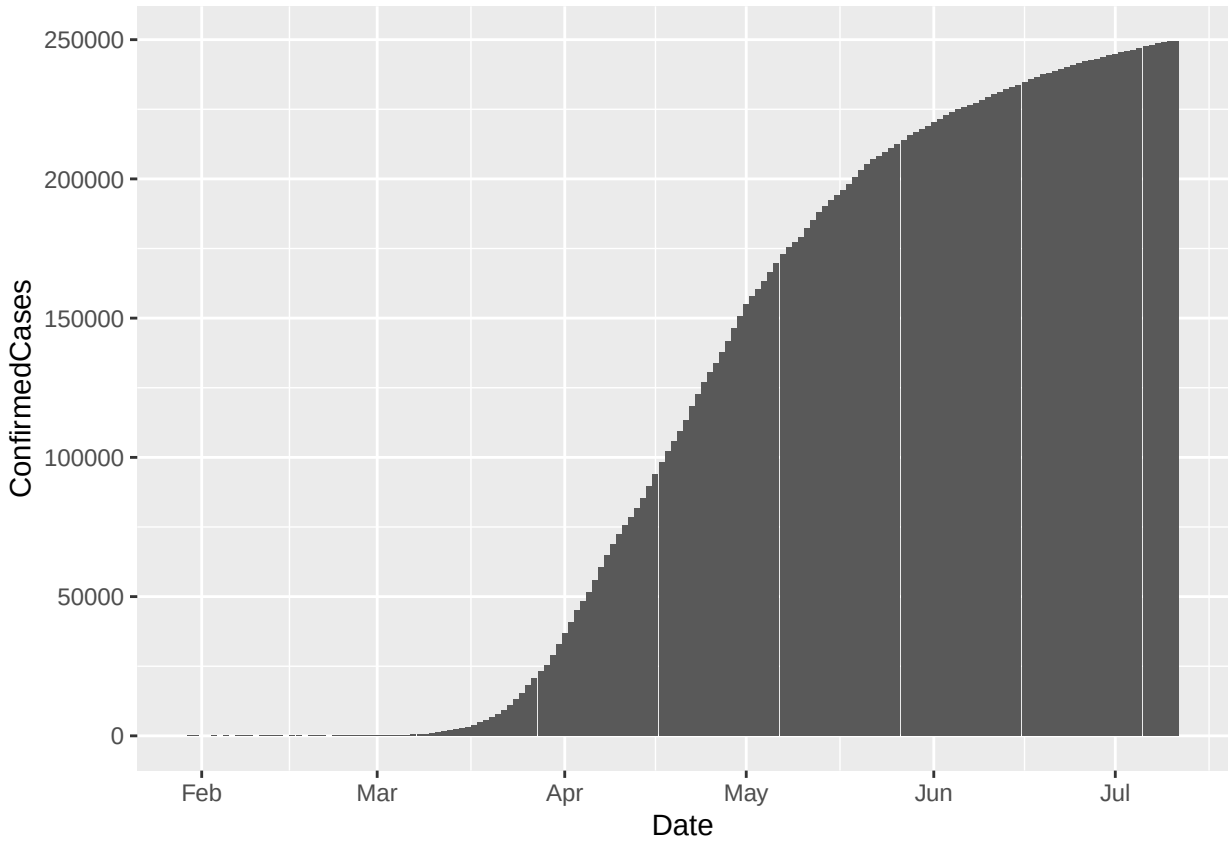
`https://github.com/tomwhite/covid-19-uk-data/tree/b5e3384a4a6cb6bbf3aa292395ccd90a36771af1`

```
commit_id = "b5e3384a4a6cb6bbf3aa292395ccd90a36771af1"
tom_white_cases = readr::read_csv(paste0(
  "https://github.com/tomwhite/covid-19-uk-data/raw/",commit_id,"/data/covid-19-totals-england.csv"
))
tom_white_cases %>% glimpse()
```

```
## Rows: 157
## Columns: 4
## $ Date           <date> 2020-01-30, 2020-01-31, 2020-02-03, 2020-02-05, 2020-0~
## $ Tests          <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
## $ ConfirmedCases <dbl> 2, 3, 8, 3, 6, 7, 8, 9, 18, 19, 20, 18, 21, 22, 11, 12,~
## $ Deaths         <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA,~
```

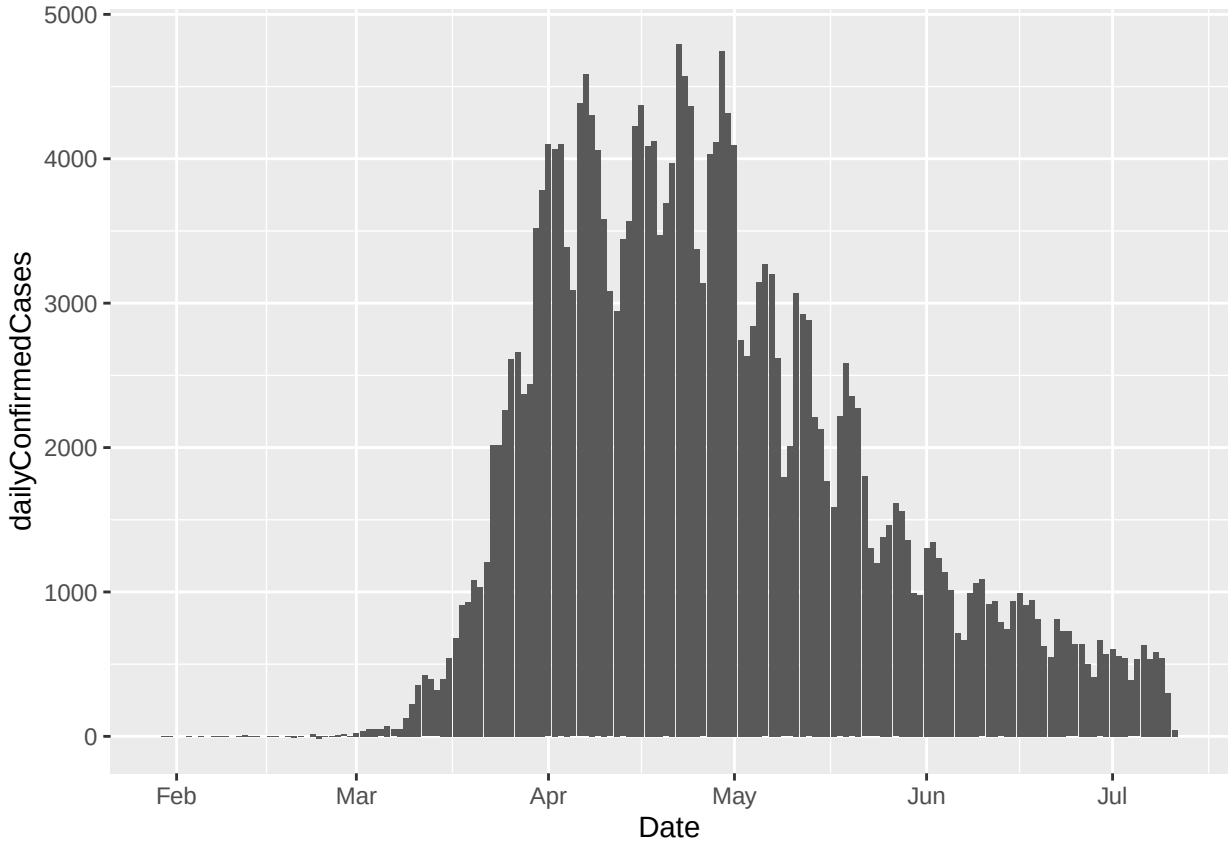
As we can see here this was presented as a cumulative case count:

```
ggplot(tom_white_cases,aes(x=Date,y=ConfirmedCases))+
  geom_bar(stat="identity")
```



To get a daily count we can do a diff:

```
tom_white_cases = tom_white_cases %>%
  mutate(dailyConfirmedCases = ConfirmedCases - lag(ConfirmedCases, default = 0))

ggplot(tom_white_cases,aes(x=Date,y=dailyConfirmedCases))+
  geom_bar(stat="identity")
```



There are several issues with this data set that need fixing before we can use `EpiEstim` which needs

- A complete time series (i.e. no missing days).
- Ordered in time.
- Includes the start of the outbreak.
- Count data is representative of incident infections.
- No missing and no negative values.

Dealing with more "real" data:

- Can you check for completeness and missing values?
- Can you fix the issues? (Hint: `tidyr::complete` and `tidyr::fill` are pretty useful here.)
- Any issues with your solution for missing values and what else could you do?

```
# The data is not conformant
if (any(is.na(tom_white_cases$dailyConfirmedCases)))
  print("missing values found")
if (any(tom_white_cases$dailyConfirmedCases < 0))
  print("negatives found")
if (any(na.omit(lead(tom_white_cases$Date)-tom_white_cases$Date != 1)))
  print("missing dates")

# To get EpiEstim to work we will have to fill in missing dates and values, and
# ensure the case count is strictly positive.

tom_white_cases_fixed = tom_white_cases %>%
  tidyr::complete(
    Date = tidyr::full_seq(Date,1), fill = list(dailyConfirmedCases = NA)
  ) %>%
```

```
  mutate(
    dailyConfirmedCases = ifelse(dailyConfirmedCases<0,NA,dailyConfirmedCases)
  ) %>%
  tidyr::fill(dailyConfirmedCases)

tom_white_cases_fixed %>% glimpse()

if (!any(is.na(tom_white_cases_fixed$dailyConfirmedCases)))
  print("missing values fixed")
if (!any(tom_white_cases_fixed$dailyConfirmedCases < 0))
  print("negatives fixed")
if (!any(na.omit(lead(tom_white_cases_fixed$Date)-tom_white_cases_fixed$Date != 1)))
  print("missing dates fixed")
```

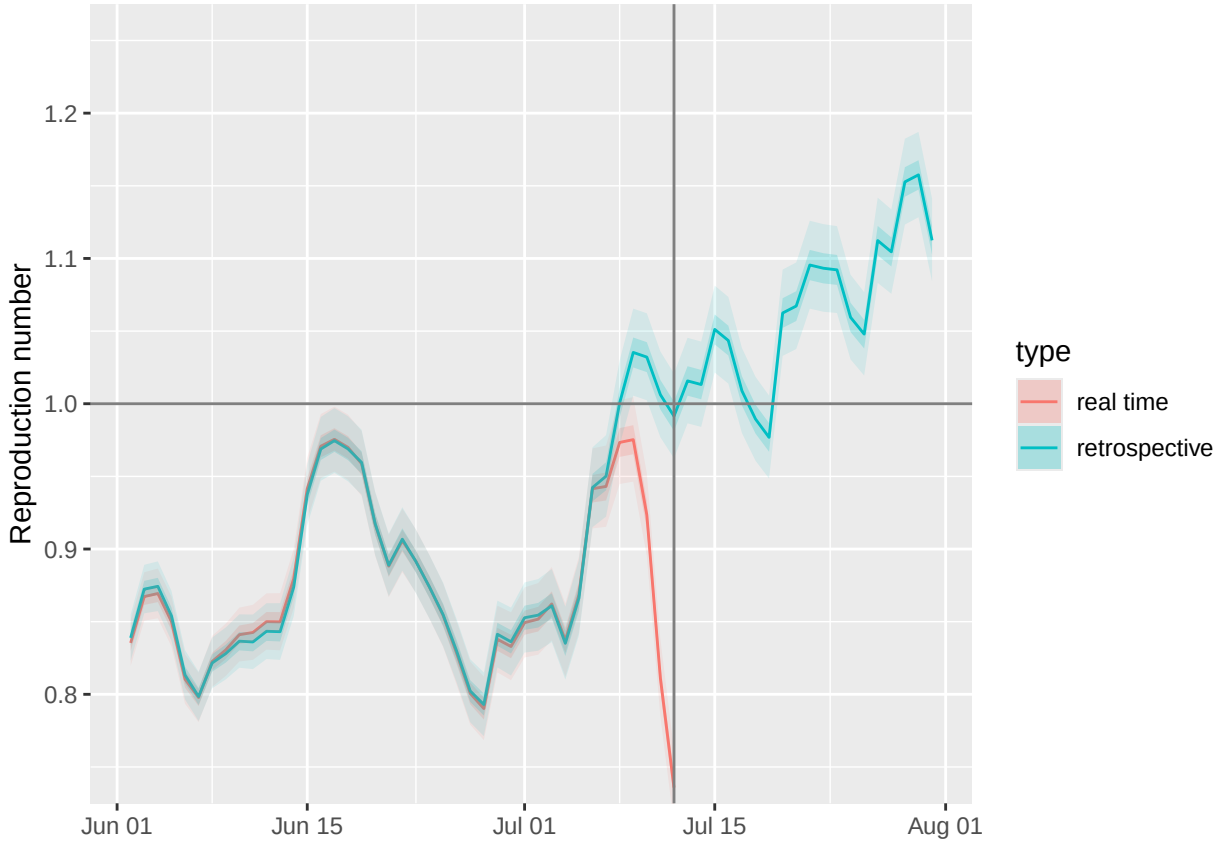
```
## [1] "missing values found"
## [1] "negatives found"
## [1] "missing dates"
## Rows: 165
## Columns: 5
## $ Date                <date> 2020-01-30, 2020-01-31, 2020-02-01, 2020-02-02, 2~
## $ Tests               <lgl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ ConfirmedCases      <dbl> 2, 3, NA, NA, 8, NA, 3, NA, 6, 7, 8, NA, 9, 18, 19~
## $ Deaths              <dbl> NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA, NA~
## $ dailyConfirmedCases <dbl> 2, 1, 1, 1, 5, 5, 5, 5, 3, 1, 1, 1, 1, 9, 1, 1, 1,~
## [1] "missing values fixed"
## [1] "negatives fixed"
## [1] "missing dates fixed"
```

We now have a fixed data set as if it was the 12th July 2020 (a Sunday).

- Can you modify the code we used above to get a reproduction number estimate for this data and plot it?

```
window = 7
t_start = 2:(nrow(tom_white_cases_fixed)-window)
t_end = t_start + window

simple_config_2 = EpiEstim::make_config(
  method = "parametric_si",
  mean_si = ganyani_mean_gi,
  std_si = ganyani_sd_gi,
  mean_prior = 1,
  std_prior = 1,
  t_start = t_start,
  t_end = t_end
)

simple_ts_2 = tom_white_cases_fixed %>%
  select(dates = Date, I = dailyConfirmedCases) %>%
  arrange(dates)

epiestim_out_2 = estimate_R(
    simple_ts_2,
    method = "parametric_si",
    config=simple_config_2
```

```
  )

plot_data_2 = epiestim_out_2$R %>%
  mutate(date = epiestim_out_2$dates[t_end]) %>%
  filter(date > "2020-06-01" & date < "2020-08-01")

combined_plot_data = bind_rows(
  plot_data %>% mutate(type = "retrospective"),
  plot_data_2 %>% mutate(type = "real time")
)

ggplot(combined_plot_data)+
    geom_line(mapping=aes(x=date, y=`Mean(R)`, colour=type))+
    geom_ribbon(mapping=
        aes(x=date, ymin=`Quantile.0.25(R)`,ymax=`Quantile.0.75(R)`, fill=type),
        colour=NA, alpha=0.2)+
    geom_ribbon(mapping=
        aes(x=date, ymin=`Quantile.0.025(R)`,ymax=`Quantile.0.975(R)`, fill=type),
        colour=NA, alpha=0.1)+
    geom_hline(yintercept = 1, colour="grey50")+
    geom_vline(xintercept = as.Date("2020-07-12"), colour="grey50")+
    xlab(NULL)+
    ylab("Reproduction number")+
    coord_cartesian(ylim=c(0.75,1.25))
```



- Can you explain the differences? Why does it drop off at the end?
- What would you have advised if this was the data you saw?

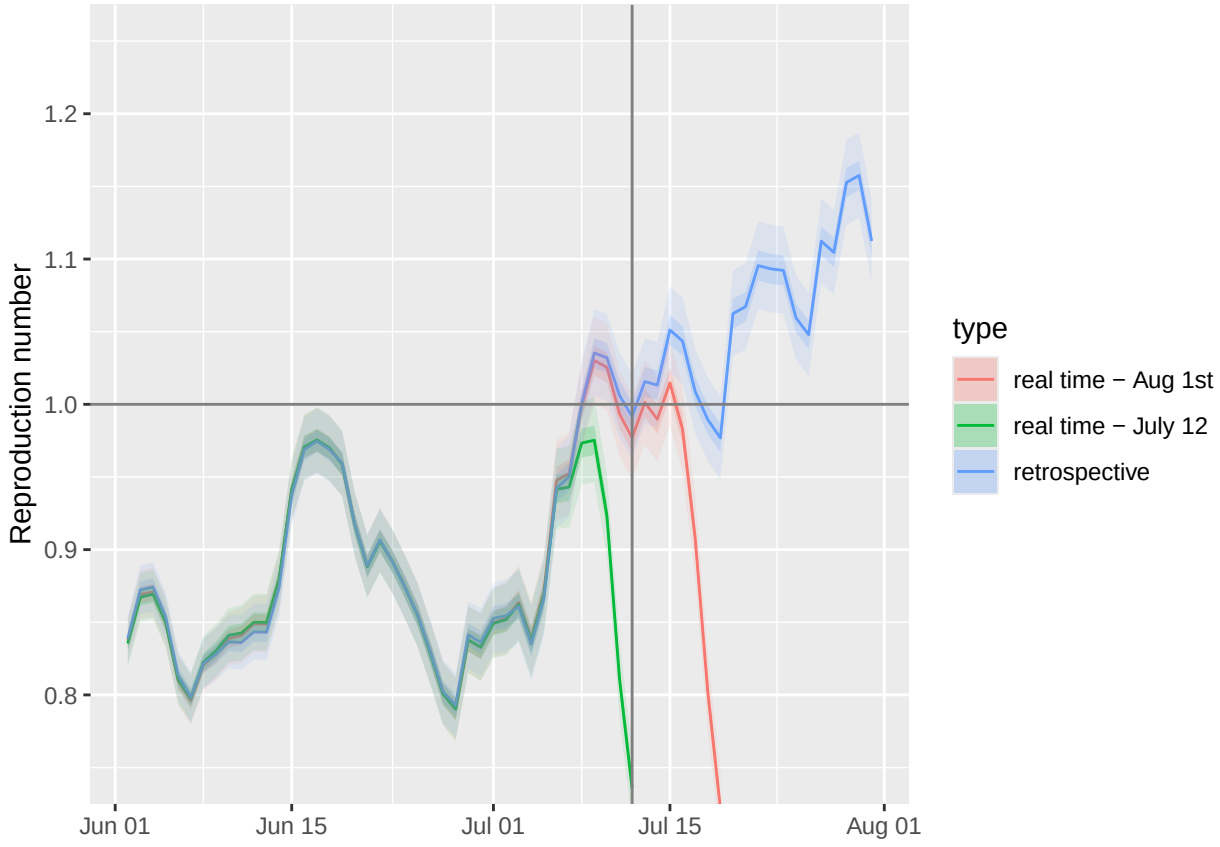
- What do you think about the confidence intervals? Is there a source of uncertainty we are not accounting for?
- The tom white data for 19th July 2020 is here:

https://github.com/tomwhite/covid-19-uk-data/raw/e890a26b7a2bc2e36e11241d9f584e074d5a2888/data/covid-19-totals-england.csv

- What do you see with the additional 1 weeks data?

```
commit_id_2 = "e890a26b7a2bc2e36e11241d9f584e074d5a2888"

tom_white_cases_2 = readr::read_csv(paste0(
  "https://github.com/tomwhite/covid-19-uk-data/raw/",commit_id_2,"/data/covid-19-totals-england.csv"
))

tom_white_cases_2 = tom_white_cases_2 %>%
  mutate(dailyConfirmedCases = ConfirmedCases - lag(ConfirmedCases, default = 0))

tom_white_cases_fixed_2 = tom_white_cases_2 %>%
  tidyr::complete(
    Date = tidyr::full_seq(Date,1), fill = list(dailyConfirmedCases = NA)
  ) %>%
  mutate(
    dailyConfirmedCases = ifelse(dailyConfirmedCases<0,NA,dailyConfirmedCases)
  ) %>%
  tidyr::fill(dailyConfirmedCases)

window = 7
t_start = 2:(nrow(tom_white_cases_fixed_2)-window)
t_end = t_start + window

simple_config_3 = EpiEstim::make_config(
  method = "parametric_si",
  mean_si = ganyani_mean_gi,
  std_si = ganyani_sd_gi,
  mean_prior = 1,
  std_prior = 1,
  t_start = t_start,
  t_end = t_end
)

simple_ts_3 = tom_white_cases_fixed_2 %>%
  select(dates = Date, I = dailyConfirmedCases) %>%
  arrange(dates)

epiestim_out_3 = estimate_R(
    simple_ts_3,
    method = "parametric_si",
    config=simple_config_3
  )

plot_data_3 = epiestim_out_3$R %>%
  mutate(date = epiestim_out_3$dates[t_end]) %>%
  filter(date > "2020-06-01" & date < "2020-08-01")
```

```r
combined_plot_data = bind_rows(
  plot_data %>% mutate(type = "retrospective"),
  plot_data_2 %>% mutate(type = "real time - July 12"),
  plot_data_3 %>% mutate(type = "real time - Aug 1st")
)

ggplot(combined_plot_data)+
    geom_line(mapping=aes(x=date, y=`Mean(R)`, colour=type))+
    geom_ribbon(mapping=
        aes(x=date, ymin=`Quantile.0.25(R)`,ymax=`Quantile.0.75(R)`, fill=type),
        colour=NA, alpha=0.2)+
    geom_ribbon(mapping=
        aes(x=date, ymin=`Quantile.0.025(R)`,ymax=`Quantile.0.975(R)`, fill=type),
        colour=NA, alpha=0.1)+
    geom_hline(yintercept = 1, colour="grey50")+
    geom_vline(xintercept = as.Date("2020-07-12"), colour="grey50")+
    xlab(NULL)+
    ylab("Reproduction number")+
    coord_cartesian(ylim=c(0.75,1.25))
```



Further questions

Official estimates

I've bundled the official SPI-M estimates here:

`https://github.com/bristol-vaccine-centre/edam-epiestim/raw/main/input/covid-consensus-rt.csv`

- How do the EpiEsim estimates compare?
- Which is more timely?

Uncertainty

We've talked about uncertainty in infectivity profile and the generation interval vs. serial interval debacle.

In the Ganyani paper we saw a central estimate for the mean and sd of the generation time distribution, with 95% credible intervals. In their paper they say: "Posterior point estimates are given by the 50% percentiles of the converged MCMC chain. CrIs are given by the 2.5% and 97.5% percentiles of the converged MCMC chain."

EpiEstim can account for uncertainty in the infectivity profile. From the help file: "In method"uncertain_si" the mean and sd of the serial interval are each drawn from truncated normal distributions, with parameters specified by the user"

Take a look at the help file for `EpiEstim::make_config` again, looking at the parameters `mean_si`,`std_si`,`std_mean_si`,`min_m`
`std_std_si`, `min_std_si`, `max_std_si` and the method option `uncertain_si`.

- What does Ganyani actually report?
- What does EpiEstim actually need?
- What is missing?
- If we assumed the posteriors for mean and SD of the generation interval were log-normally distributed could you infer what you need?