```
knitr::opts_knit$set(root.dir = '~/Desktop/bio_modelling_course/mod3/proj3/')
```

Code for question 2

```
library(magrittr)
library(tidyverse)
```

```
## ── Attaching packages ─────────────────────────────────────────────────── tidyverse 1.2.1 ──
```

```
## ✔ ggplot2 2.2.1     ✔ purrr   0.2.4
## ✔ tibble  1.4.2     ✔ dplyr   0.7.4
## ✔ tidyr   0.8.0     ✔ stringr 1.3.0
## ✔ readr   1.1.1     ✔ forcats 0.3.0
```

```
## ── Conflicts ────────────────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ tidyr::extract()   masks magrittr::extract()
## ✖ dplyr::filter()    masks stats::filter()
## ✖ dplyr::lag()       masks stats::lag()
## ✖ purrr::set_names() masks magrittr::set_names()
```

```
library(gridExtra)
```

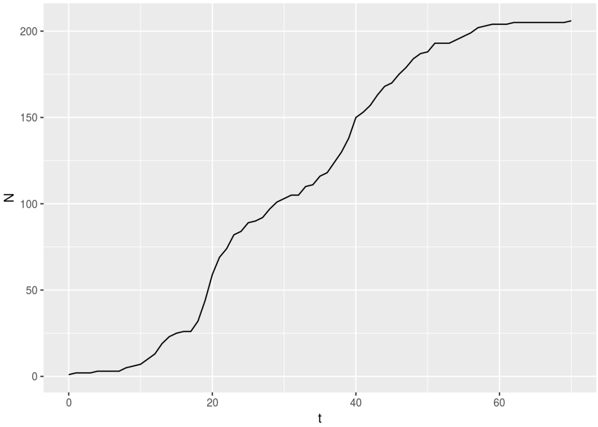
```
##
## Attaching package: 'gridExtra'
```

```
## The following object is masked from 'package:dplyr':
##
##     combine
```

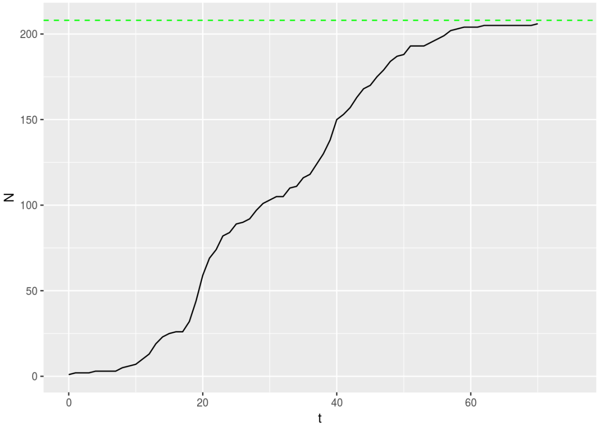
```
sars<-read_table('SARS.txt') %>%
  mutate_all(funs(as.integer))
```

```
## Parsed with column specification:
## cols(
##   Dag = col_integer(),
##   `Smittede  ` = col_integer()
## )
```

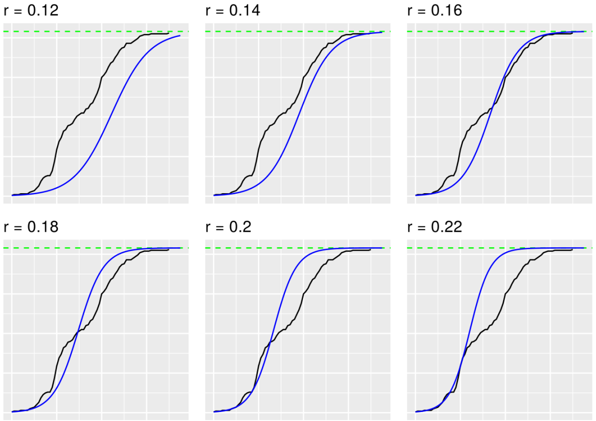
```
colnames(sars) <- c('t', 'N')

plt <- sars %>%
  ggplot(aes(t, N)) +
  geom_line()
plt
```

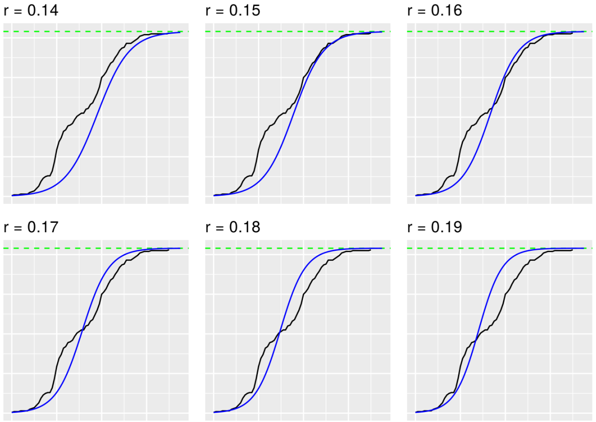


```
plt2 <- plt +
  geom_hline(yintercept = 208,
             color='green',
             linetype='dashed') +
  xlim(0, 75)
plt2
```



```
make_logit <- function(r) {
  list(Vectorize(function(t) 208/(1 + 207*exp(-r*t))), r)
}

find_r_plots <- function(model_builder, search_spaces) {
  map(search_spaces,
      function (space) {
        grid.arrange(grobs=map(map(space, model_builder),
                           function(l) {
                             plt2 +
                               stat_function(fun=l[[1]], color='blue') +
                               ggtitle(str_c('r = ', l[[2]])) +
                               theme(axis.text = element_blank(),
                                     axis.ticks = element_blank(),
                                     axis.title = element_blank())
                           }),
                 nrow=2)
      }
  )
}

find_r_plots(make_logit, list(seq(0.12,0.22,by=0.02), seq(0.14,0.19,by=0.01)))
```



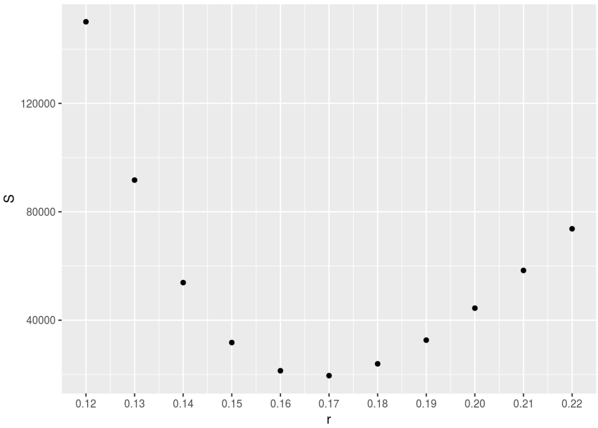


```
## [[1]]
## TableGrob (2 x 3) "arrange": 6 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
## 3 3 (1-1,3-3) arrange gtable[layout]
## 4 4 (2-2,1-1) arrange gtable[layout]
## 5 5 (2-2,2-2) arrange gtable[layout]
## 6 6 (2-2,3-3) arrange gtable[layout]
## 
## [[2]]
## TableGrob (2 x 3) "arrange": 6 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
## 3 3 (1-1,3-3) arrange gtable[layout]
## 4 4 (2-2,1-1) arrange gtable[layout]
## 5 5 (2-2,2-2) arrange gtable[layout]
## 6 6 (2-2,3-3) arrange gtable[layout]
```

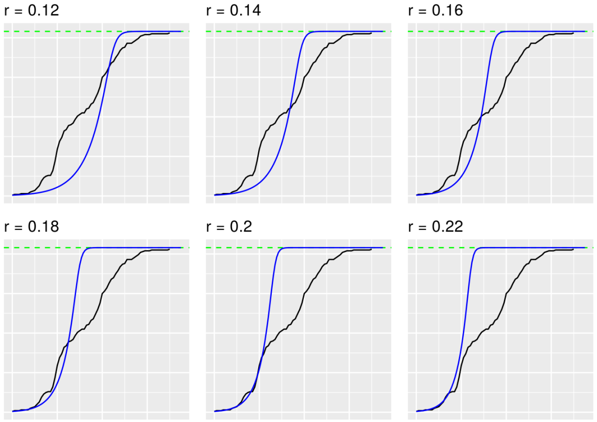
```
S <- function(model) {
  sars %>%
    mutate(pred = model(t),
           res_sqrd = (N-pred)**2) %>%
    summarise(S = sum(res_sqrd)) %>%
    .$S
}

S(make_logit(0.17)[[1]])
```

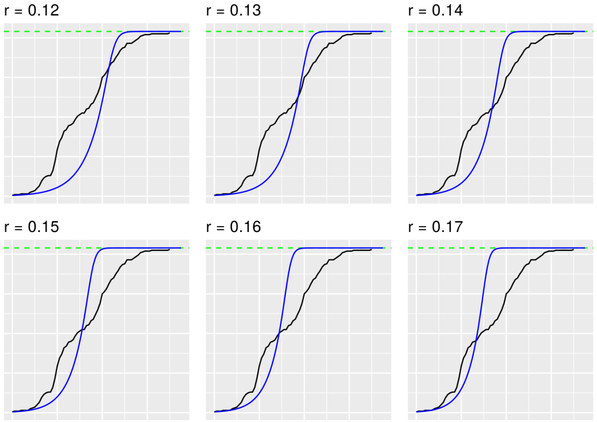
```
## [1] 19549.88
```

```
eval_mods <- function(rs, model_builder) {
  data_frame(r=rs) %>%
    mutate(S = map_dbl(map(rs, function(r) model_builder(r)[[1]]), S)) %>%
    ggplot(aes(r,S)) +
    geom_point() +
    scale_x_continuous(breaks=rs)
}

eval_mods(seq(.12,.22,by=.01), make_logit)
```



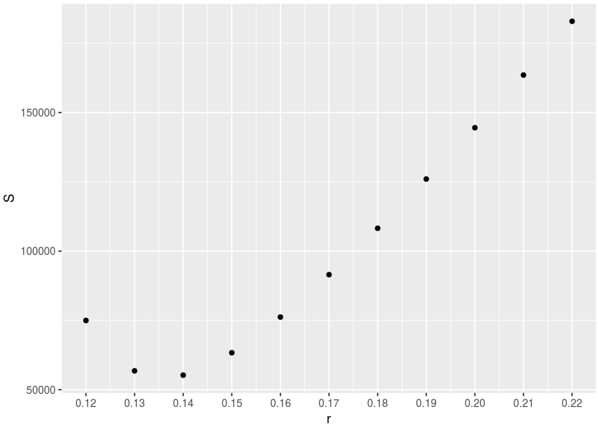
```
make_mod_logit <- function(r) {
  list(Vectorize(function(t) 208/((1 + (208**5 - 1)*exp(-5*r*t)))**(1/5)), r)
}

find_r_plots(make_mod_logit, list(seq(0.12,0.22,by=0.02), seq(0.12,0.17,by=0.01)))
```





```
## [[1]]
## TableGrob (2 x 3) "arrange": 6 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
## 3 3 (1-1,3-3) arrange gtable[layout]
## 4 4 (2-2,1-1) arrange gtable[layout]
## 5 5 (2-2,2-2) arrange gtable[layout]
## 6 6 (2-2,3-3) arrange gtable[layout]
##
## [[2]]
## TableGrob (2 x 3) "arrange": 6 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
## 3 3 (1-1,3-3) arrange gtable[layout]
## 4 4 (2-2,1-1) arrange gtable[layout]
## 5 5 (2-2,2-2) arrange gtable[layout]
## 6 6 (2-2,3-3) arrange gtable[layout]
```

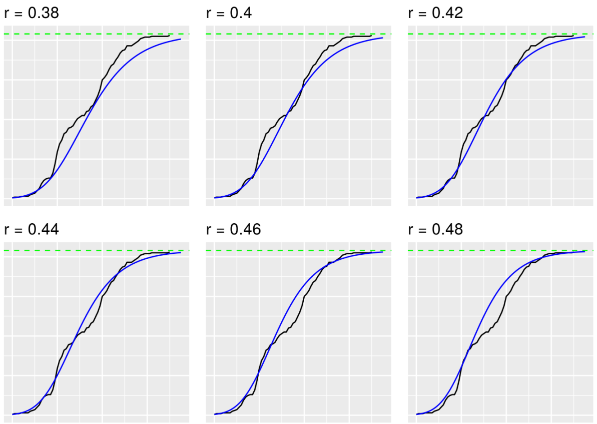
```
eval_mods(seq(.12,.22,by=.01), make_mod_logit)
```

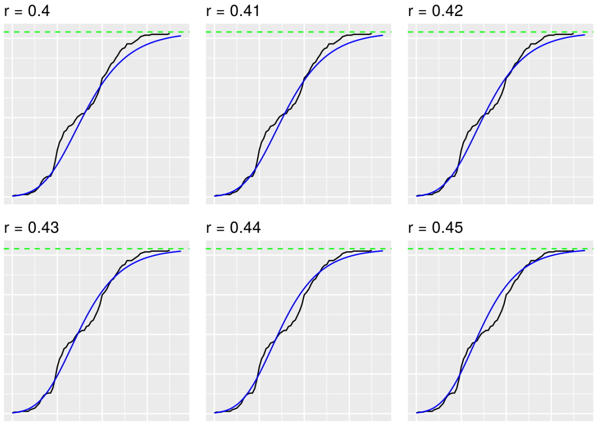


```
S(make_mod_logit(0.14)[[1]])
```

```
## [1] 55249.06
```

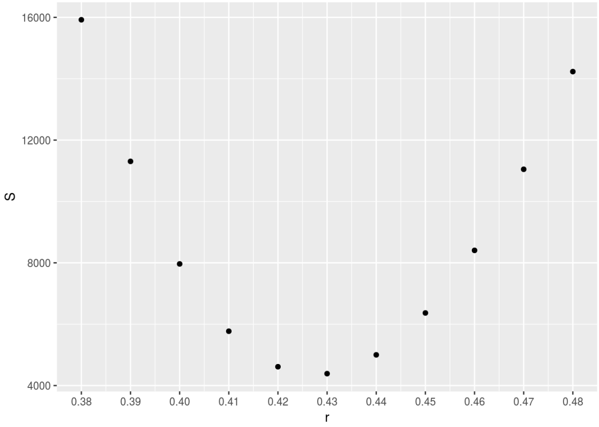
```
make_mod_logit_2 <- function(r) {
  list(Vectorize(function(t) 208/((1 + 1.908*exp(-0.2*r*t)))**5), r)
}

find_r_plots(make_mod_logit_2, list(seq(0.38,0.48,by=0.02), seq(0.40,0.45,by=0.01)))
```





```
## [[1]]
## TableGrob (2 x 3) "arrange": 6 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
## 3 3 (1-1,3-3) arrange gtable[layout]
## 4 4 (2-2,1-1) arrange gtable[layout]
## 5 5 (2-2,2-2) arrange gtable[layout]
## 6 6 (2-2,3-3) arrange gtable[layout]
## 
## [[2]]
## TableGrob (2 x 3) "arrange": 6 grobs
##   z     cells    name           grob
## 1 1 (1-1,1-1) arrange gtable[layout]
## 2 2 (1-1,2-2) arrange gtable[layout]
## 3 3 (1-1,3-3) arrange gtable[layout]
## 4 4 (2-2,1-1) arrange gtable[layout]
## 5 5 (2-2,2-2) arrange gtable[layout]
## 6 6 (2-2,3-3) arrange gtable[layout]
```

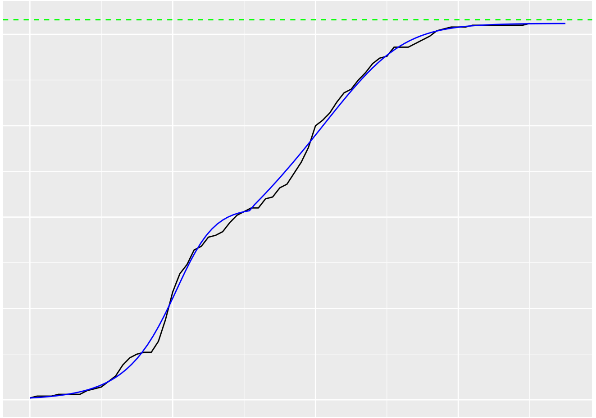
```
eval_mods(seq(0.38,0.48,by=0.01), make_mod_logit_2)
```



```
S(make_mod_logit_2(0.43)[[1]])
```

```
## [1] 4388.52
```

```
N1 <- function(t) 105/((1 + 10182*exp(-0.415*t)))**(0.504)
N2 <- function(t) 206/((1 + 146757*exp(-0.243*t)))**(0.154)

last_mod <- Vectorize(function(t) {
  if (t <= 31) N1(t)
  else N2(t)
})

plt2 +
  stat_function(fun=last_mod, color='blue') +
  theme(axis.text = element_blank(),
        axis.ticks = element_blank(),
        axis.title = element_blank())
```



```
c(N1(31), N2(31))
```

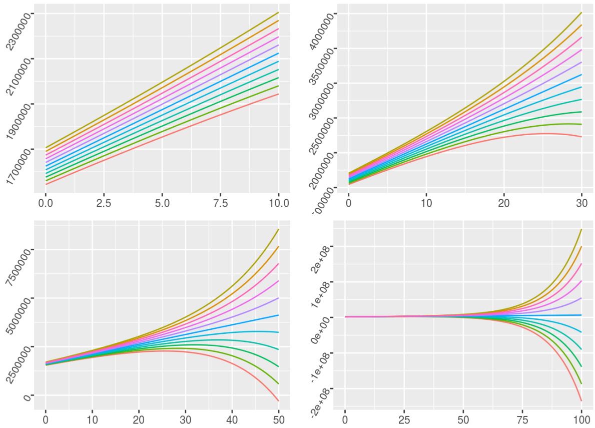
```
## [1] 103.6330 104.9982
```

```
S(last_mod)
```

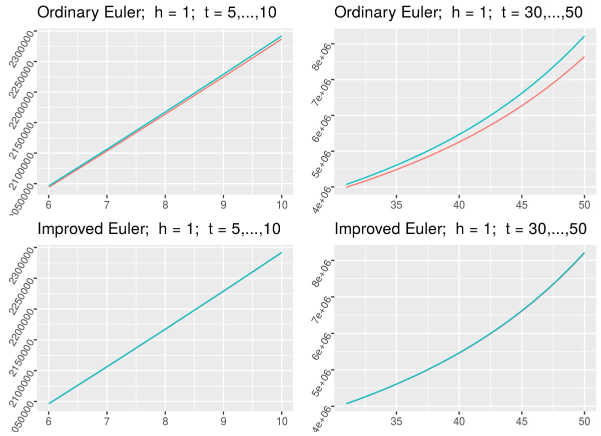
```
## [1] 667.2926
```

Code for question 3

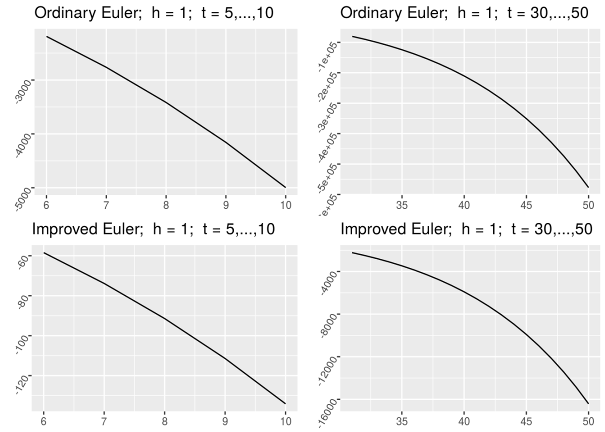
```r
library(magrittr)
library(tidyverse)
library(gridExtra)

r0 <- 0.08
u0 <- 80000
alpha <- 4000

xmin <- (r0*u0 + alpha)/r0**2

h <- Vectorize(function(t) {
  xmin + (alpha/r0)*t
})

x <- function(t, x0) {
  h(t) + (x0 - h(0))*exp(r0*t)
}
ts <- seq(0,100,by=0.2)
x0s <- xmin*seq(0.95,1.05,by=0.01)
xs <- map(map(x0s, function(x0) {function(t) x(t, x0)}),
    function(f) {f(ts)})
xs <- data.frame(xs)
colnames(xs) <- str_c('x', as.character(1:11))
plot_lines <- function(max_t) {
  xs %>%
    mutate(t = ts) %>%
    filter(t <= max_t) %>%
    gather(key='key', value='value', -t) %>%
    ggplot(aes(t, value, color=key)) +
    geom_line() +
    theme(legend.position = 'none',
          axis.text.y = element_text(angle = 60),
          axis.title = element_blank())
}
grid.arrange(grobs=map(c(10,30,50,100), plot_lines), nrow=2)
```



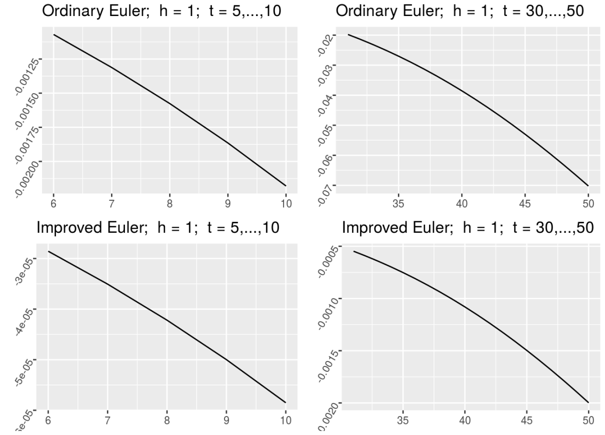
```
###############################################################################

euler <- function(diff,
                  x0, y0, x_end,
                  h, improv = F) {
  xs <- seq(x0, x_end + x_end%%h, by = h)
  ys <- rep(NA, length(xs))
  ys[1] <- y0
  for (i in 2:length(xs)) {
    ystar <- ys[i-1] + diff(xs[i-1], ys[i-1])*h
    if (improv) {
      ys[i] <- ys[i-1] + (diff(xs[i-1], ys[i-1]) + diff(xs[i], ystar))*h/2
    }
    else {
      ys[i] <- ystar
    }
  }
  data_frame(t=xs, x_euler=ys)
}

approx <- function(h, t_end = 10, improv = F) {
  euler(function(t, x) r0*x - u0 - alpha*t,
        0, 1700000, t_end,
        h, improv) %>%
    mutate(x_true = map_dbl(t, function(t) x(t,1700000)),
           error = x_euler - x_true,
           error_percent = error / x_true)
}

approx_plot <- function(h, t_end, improv, min_t) {
  if (improv) {
    name <- 'Improved Euler'
  }
  else {
    name <- 'Ordinary Euler'
  }
  approx(h, t_end, improv) %>%
    filter(min_t<t) %>%
    select(-error, -error_percent) %>%
    gather(key='key', value='value', -t) %>%
    ggplot(aes(t, value, color=key)) +
    geom_line() +
    theme(legend.position = 'none',
          axis.text.y = element_text(angle=60),
          axis.title = element_blank()) +
    ggtitle(str_c(name, ';  h = ', h, ';  t = ', min_t, ',...,', t_end))
}

error_plot <- function(h, t_end, improv, min_t, error_type) {
  if (improv) {
    name <- 'Improved Euler'
  }
  else {
    name <- 'Ordinary Euler'
  }
  approx(h, t_end, improv) %>%
    filter(min_t<t) %>%
    ggplot(aes_string('t', error_type)) +
    geom_line() +
    theme(axis.text.y = element_text(angle=60),
          axis.title = element_blank()) +
    ggtitle(str_c(name, ';  h = ', h, ';  t = ', min_t, ',...,', t_end))
}

grid.arrange(grobs = list(approx_plot(1, 10, F, 5),
                          approx_plot(1, 50, F, 30),
                          approx_plot(1, 10, T, 5),
                          approx_plot(1, 50, T, 30)),
             nrow=2)
```



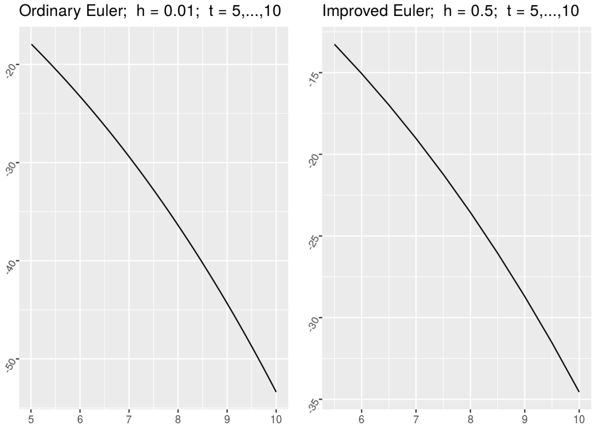
```
grid.arrange(grobs = list(error_plot(1, 10, F, 5, 'error'),
                          error_plot(1, 50, F, 30, 'error'),
                          error_plot(1, 10, T, 5, 'error'),
                          error_plot(1, 50, T, 30, 'error')),
             nrow=2)
```



```
grid.arrange(grobs = list(error_plot(1, 10, F, 5, 'error_percent'),
                          error_plot(1, 50, F, 30, 'error_percent'),
                          error_plot(1, 10, T, 5, 'error_percent'),
                          error_plot(1, 50, T, 30, 'error_percent')),
             nrow=2)
```



```
grid.arrange(grobs = list(error_plot(0.01, 10, F, 5, 'error'),
                          error_plot(0.5, 10, T, 5, 'error')),
             nrow=1)
```



```
h_threshold_ord <- uniroot(function(h) approx(h) %>% .$error %>% tail(1) + 100, c(0.01, 1))$root
h_threshold_improv <- uniroot(function(h) approx(h,improv=T) %>% .$error %>% tail(1) + 100, c(0.5, 1))$root
h_threshold_ord
```

```
## [1] 0.01875349
```

```
h_threshold_improv
```

```
## [1] 0.8469925
```

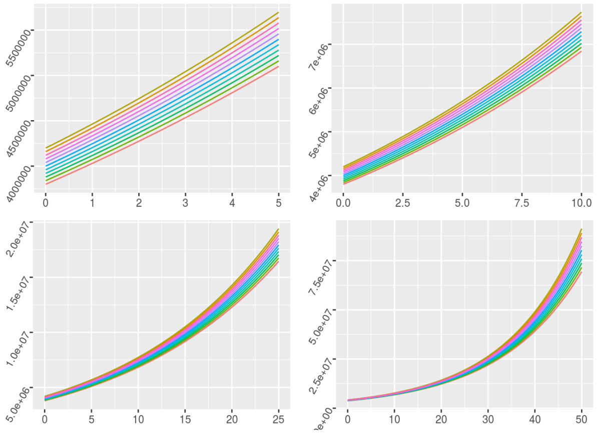
```
approx(h_threshold_ord, improv=F) %>% tail(1)
```

```
## # A tibble: 1 x 5
##       t  x_euler   x_true  error error_percent
##   <dbl>    <dbl>    <dbl>  <dbl>         <dbl>
## 1 10.00 2291538. 2291638. -100.0    -0.0000436
```

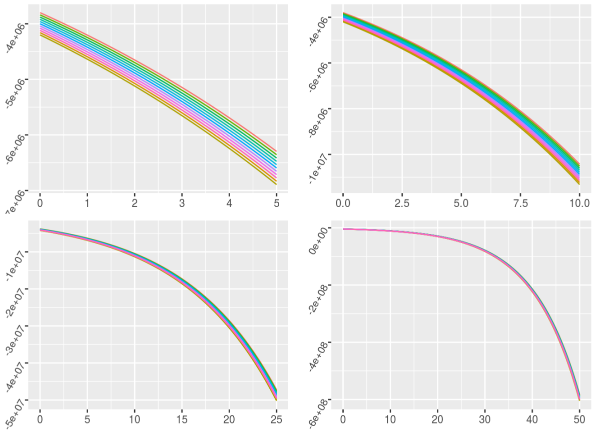
```
approx(h_threshold_improv, improv=T) %>% tail(1)
```

```
## # A tibble: 1 x 5
##       t  x_euler   x_true  error error_percent
##   <dbl>    <dbl>    <dbl>  <dbl>         <dbl>
## 1  10.2 2302214. 2302314. -100.0    -0.0000434
```

```
#############################################################################

r0 <- 0.08
u0 <- 80000
beta <- 0.06
xmin <- u0/(r0-beta)

h <- Vectorize(function(t) {
  xmin*exp(beta*t)
})

x <- function(t, x0) {
  h(t) + (x0 - h(0))*exp(r0*t)
}
ts <- seq(0,500,by=0.2)
x0s <- xmin*seq(0.95,1.05,by=0.01)
xs <- map(map(x0s, function(x0) {function(t) x(t, x0)}),
          function(f) {f(ts)})
xs <- data.frame(xs)
colnames(xs) <- str_c('x', as.character(1:11))
plot_lines <- function(tlims) {
  xs %>%
    mutate(t = ts) %>%
    filter(tlims[1] <= t , t <= tlims[2]) %>%
    gather(key='key', value='value', -t) %>%
    ggplot(aes(t, value, color=key)) +
    geom_line() +
    theme(legend.position = 'none',
          axis.text.y = element_text(angle = 60),
          axis.title = element_blank())
}
grid.arrange(grobs=map(list(c(0,5),c(0,10),c(0,25), c(0,50)), plot_lines), nrow=2)
```



```
r0 <- 0.08
u0 <- 80000
beta <- 0.1
xmin <- u0/(r0-beta)

h <- Vectorize(function(t) {
  xmin*exp(beta*t)
})

x <- function(t, x0) {
  h(t) + (x0 - h(0))*exp(r0*t)
}
ts <- seq(0,500,by=0.2)
x0s <- xmin*seq(0.95,1.05,by=0.01)
xs <- map(map(x0s, function(x0) {function(t) x(t, x0)}),
          function(f) {f(ts)})
xs <- data.frame(xs)
colnames(xs) <- str_c('x', as.character(1:11))
plot_lines <- function(tlims) {
  xs %>%
    mutate(t = ts) %>%
    filter(tlims[1] <= t , t <= tlims[2]) %>%
    gather(key='key', value='value', -t) %>%
    ggplot(aes(t, value, color=key)) +
    geom_line() +
    theme(legend.position = 'none',
          axis.text.y = element_text(angle = 60),
          axis.title = element_blank())
}
grid.arrange(grobs=map(list(c(0,5),c(0,10),c(0,25), c(0,50)), plot_lines), nrow=2)
```



-4e+06
-5e+06
-6e+06
-7e+06
0
1
2
3
4
5
-4e+06
-6e+06
-8e+06
-1e+07
0.0
2.5
5.0
7.5
10.0
-1e+07
-2e+07
-3e+07
-4e+07
-5e+07
0
5
10
15
20
25
0e+00
-2e+08
-4e+08
-6e+08
0
10
20
30
40
50