STAT40850 - Assignment 3 - Intro to Bayesian Analysis

Conor Ryan - 16340116

2023-03-14

Introduction

In this assignment we wish to develop two Bayesian models for the trajectory of the number of COVID-19 cases in Ireland during the initial three months of the pandemic. We will then compare both models and consider ways to improve model fit.

We suppose that the observed number of COVID-19 cases on day t is denoted by y_t where t ranges from 1 to T. We then assume that y_t can be modelled using a Poisson distribution to describe the uncertainty in the number of COVID-19 cases in day t.

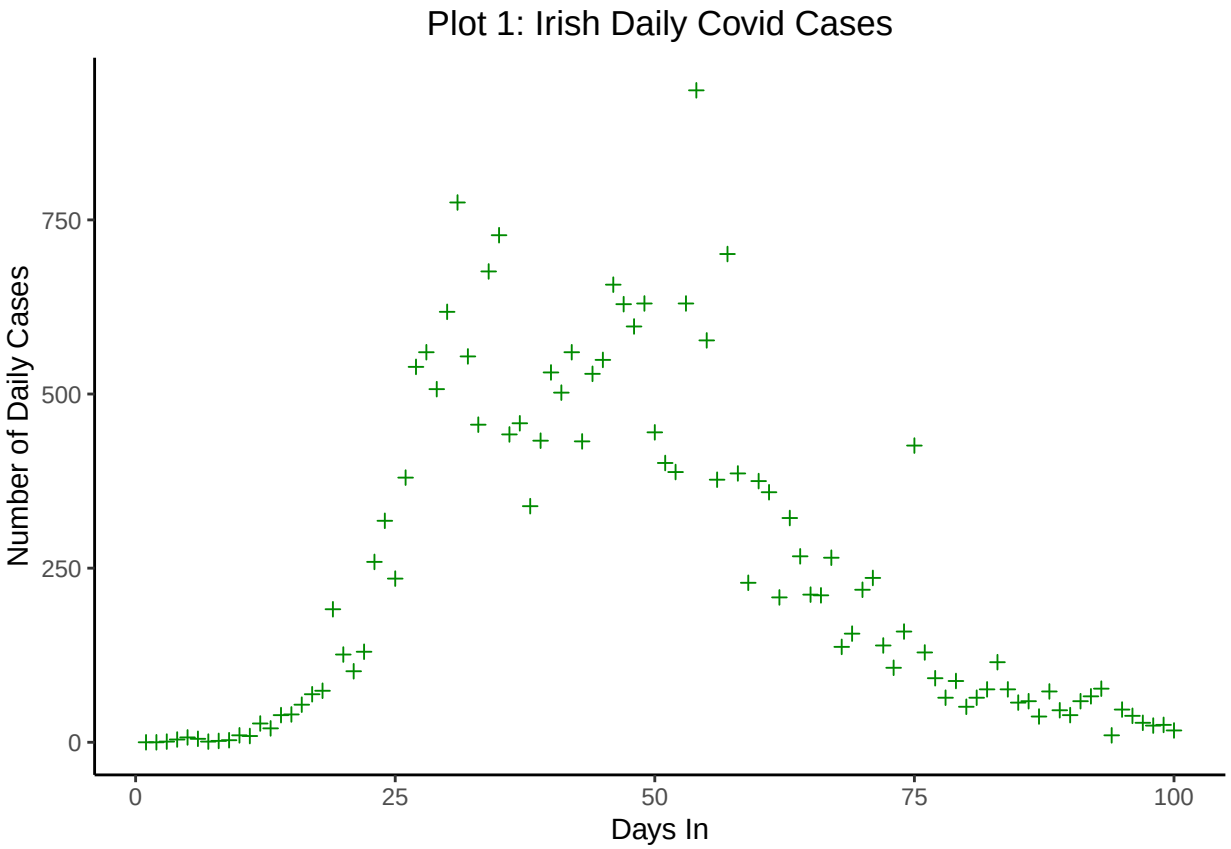
$$y_t \sim Po(\lambda(t)), t = 1, ..., T.$$

This is a regression model, where the Poisson rate parameter is $\lambda(t)$. We will consider two potential functions, the *logistic function* and the *Gompertz function*, as possible choices for the rate parameter $\{\lambda(t) : t = 1, ..., T\}$.

Below, I read in the Irish data set which contains the number of daily recorded cases of COIVD-19 in Ireland for the first three months of the pandemic. I also produce a plot of the the daily number of cases to get a sense of how the case numbers look over this time period.

```
# Reading in cases data from text file
data <- read.csv("ireland1.txt")
# Setting column name to be "Cases"
colnames(data) <- "Cases"
# Creating a new "Days" column
data$Days <- 1:100
# re-ordering columns (Days column first)
data <- data[,c(2,1)]
```

```
# Plotting Irish COVID data
ggplot() +
  geom_point(data = data,
             aes(Days,Cases), shape = 3, color = 'green4') +
  theme_classic() +
  ggtitle("Plot 1: Irish Daily Covid Cases") +
  labs(x = "Days In", y = "Number of Daily Cases") +
  theme(plot.title = element_text(hjust = 0.5))
```



Question 1

In question 1, we will develop a Bayesian model for the number of daily COVID-19 cases in Ireland using the logistic function. This model is saved as q1.stan and is represented mathematically below. The derivation of $\lambda_l(t)$ from the derivative of the logistic function $l(t)$ can be found in the appendix.

$$y_t \sim Po(\lambda_l(t))$$
$$\lambda_l(t) = \frac{\theta_1\theta_3 exp(-\theta_3(t-\theta_2))}{(1+exp^{-\theta_3(t-\theta_2)})^2}$$

All three parameters are positive and can be interpreted as follows:
θ_1 is the maximum value that $l(t)$ takes,
θ_2 is the value of t corresponding to the mid-point of the $l(t)$ function and,
θ_3 is the growth rate of the function.

$$\theta_1 \sim N(1000, 100000)$$

Where θ_1 is truncated at 0 to ensure positivity.

$$\theta_2 \sim N(50, 100)$$

Where θ_2 is truncated below at 0 and above at 100, so that $\theta_2 \in [0, 100]$.

$$\theta_3 \sim U(0, 1)$$

I will now briefly discuss the choices of prior distributions for each of the three parameters θ_1, θ_2 and θ_3 and their appropriateness. θ_1 represents the maximum value that $l(t)$ can take and therefore needs to be sufficiently large to cover the maximum COVID-19 cases in a day. I believe a normal distribution centered on 1,000 with a variance of 100,000 is therefore reasonable for this parameter. θ_1 is also strictly positive and therefore we use a truncated prior normal distribution to ensure all values of $\theta_1 > 0$.

We use a normal prior distribution for θ_2 with mean 50 and variance 100. As θ_2 is the mid-point of the $l(t)$ function I believe this is a reasonable choice of prior. We have 100 days of data and therefore the centering of this prior on 50 (mid-point between 1 and 100) is reasonable and there is also sufficient variance to cover an array of possible mid-point values. We again use a truncated prior here with a lower bound of 0 and an upper bound of 100 to ensure $\theta_2 \in [0, 100]$.

Finally, a uniform $(0, 1)$ prior is used for the distribution of θ_3 which represents the growth rate of the function. θ_3 must be > 0 and should not take values > 1, so I believe this is a reasonable choice of prior for this parameter.

I will now implement the above model using stan, see q1.stan file for reference. I fit the model using 5,000 iterations and the default 4 chains. I set a seed in order to ensure reproducible results.

```
# organising data into a list
dat1 <- list(T = nrow(data),
             days = data$Days,
             cases = data$Cases)

# fitting the model using stan
fit_q1 <- stan(file = 'q1.stan',data = dat1,seed = 1759,iter = 5000)
```

```
# print output of the fitted model
print(fit_q1, probs = c(0.05, 0.5, 0.95), pars = c("theta_1","theta_2","theta_3"))
```

```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=5000; warmup=2500; thin=1;
## post-warmup draws per chain=2500, total post-warmup draws=10000.
##
##             mean se_mean     sd       5%      50%      95% n_eff Rhat
## theta_1 25534.93    1.61 161.99 25271.16 25533.95 25803.79 10132    1
## theta_2    45.94    0.00   0.11    45.76    45.94    46.13 10058    1
## theta_3     0.10    0.00   0.00     0.10     0.10     0.10 10003    1
##
## Samples were drawn using NUTS(diag_e) at Tue Mar 14 00:56:54 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

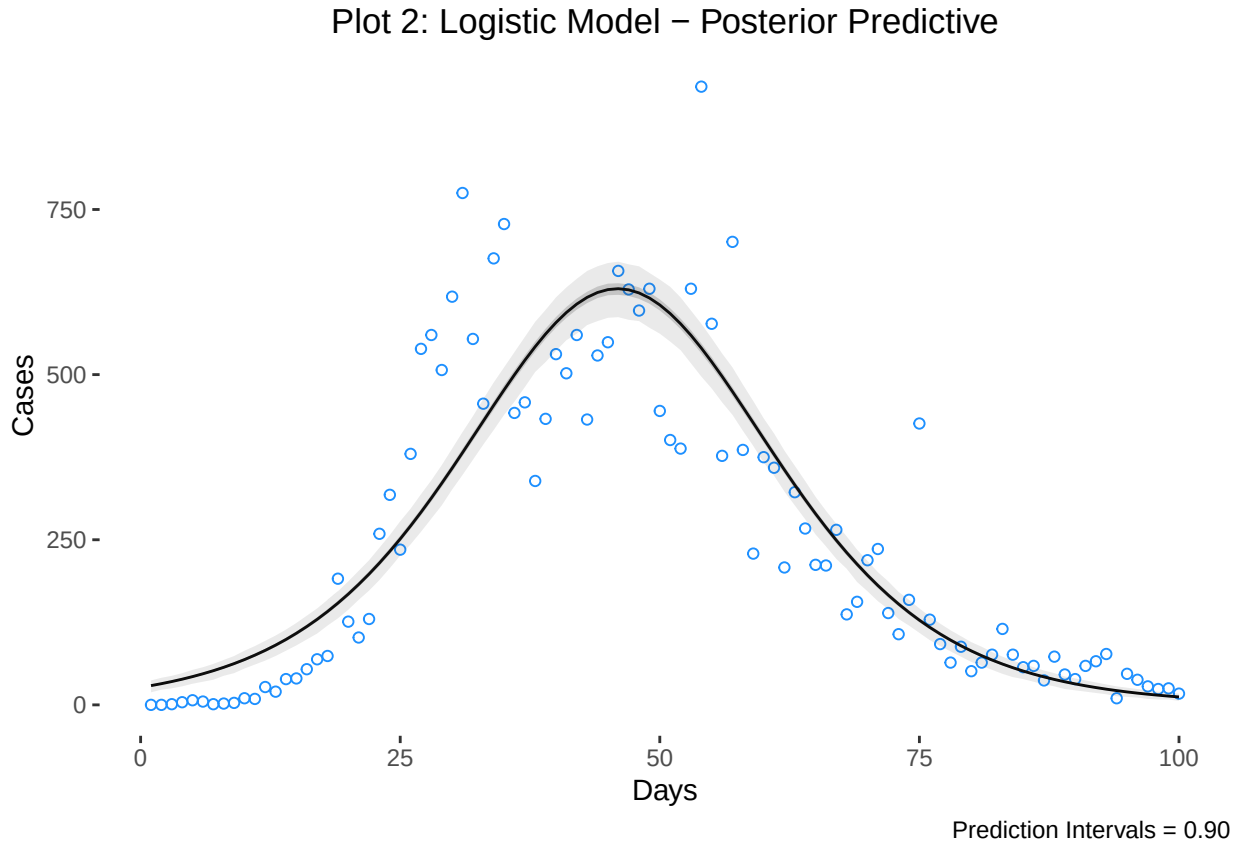
We can see the output of our fitted Bayesian model above. The average values for each of the three parameters is 25534.93, 45.94 and 0.1 for θ_1, θ_2 and θ_3 respectively. We also observe the 90% credible intervals for each of these parameters.

Next, I extract the lambda values from the output of the stan model in order to plot the posterior mean line, 90% credible interval for the mean and also the 90% credible interval for the predicted number of COVID-19 cases each day.

```
# Extracting fitted lambda values from STAN model
lambda1 <- extract(fit_q1)$lambda
```

```r
# Calculating posterior mean line (lambda) across each day
avg_lambda1 <- colMeans(lambda1)

# Calculating 90% posterior credible interval for the mean across each day
y_hdi1 <- HDInterval::hdi(lambda1,credMass = 0.90)
hpdi_l1 <- y_hdi1[1,]
hpdi_u1 <- y_hdi1[2,]

# Extracting predicted cases from model to generate prediction value for each day
y_pred1 <- extract(fit_q1)$cases_rep
yphdi1 <- HDInterval::hdi(y_pred1,credMass = 0.90)
pi_l1 <- yphdi1[1,]
pi_u1 <- yphdi1[2,]

# Extracting log-likelihood (predicted) values (for use in  loo package)
log_lik1 <- extract(fit_q1)$log_lik
```

```r
# plotting 90% prediction interval for the mean and
# 90% credible interval for predicted cases
p <- ggplot()

p2 <- p +
  geom_point(data = data,
      aes(Days, Cases), shape = 1, color = 'dodgerblue') +
  geom_line(data = data,
            aes(Days,avg_lambda1)) +
  geom_ribbon(data = data,
            mapping = aes(Days, ymin = hpdi_l1, ymax = hpdi_u1), alpha = .2) +
  geom_ribbon(data = data,
      mapping = aes(Days, ymin=pi_l1, ymax=pi_u1), alpha = .1) +
  ggtitle("Plot 2: Logistic Model - Posterior Predictive") +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(caption = "Prediction Intervals = 0.90")

p2
```



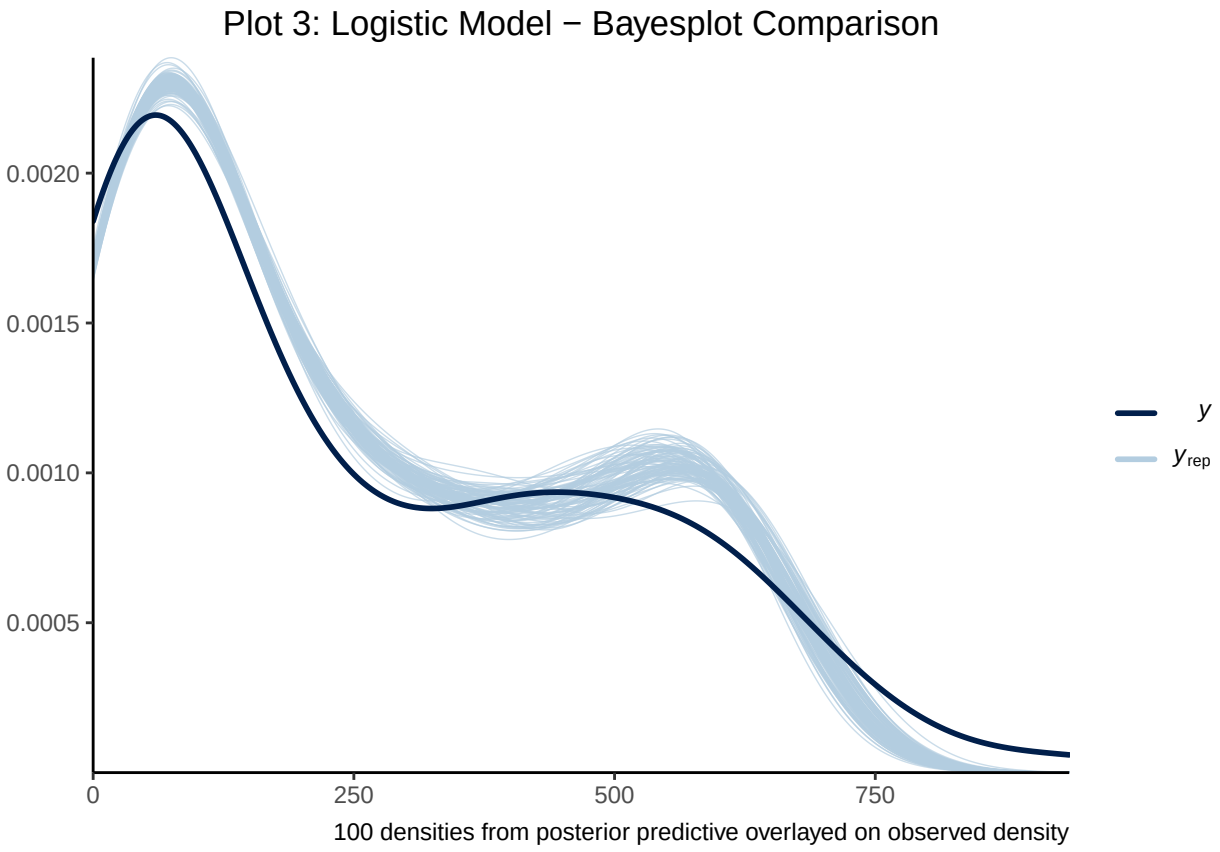
As we can see from the above posterior predictive distribution plot against the observed data points, the model does not fit the data particularly well. In the early stage of the epidemic (Days 0-20), all observed case numbers are outside of the 90% credible interval for predicted cases (all data points are below the lower bound of the 90% credible interval).

In the middle portion of the above plot (Days 21-70), we can see the cases data is quite noisy and the majority of observed data points lie well outside of our 90% credible interval for predicted cases.

In the final portion of the plot (Days 71-100) we can see the observed data points are closer to the 90% credible interval however, there is still a majority of data points lying outside of the 90% confidence interval. All of the aforementioned points indicate a poor model fit.

I will now use the Bayesplot package to overlay a collection of densities of replicated data sets produced from the posterior predictive distribution and compare these to the observed density.

```
# overlay 100 densities from posterior predictive over the observed dataset
p <- ggplot()

p3 <- ppc_dens_overlay(data$Cases, y_pred1[1:100,]) +
  theme_classic() +
  ggtitle("Plot 3: Logistic Model - Bayesplot Comparison") +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(caption = "100 densities from posterior predictive overlayed on observed density")

p3
```



In the above plot we can see the observed density (dark line) and 100 overlaid simulated densities (lighter blue lines) from the posterior predictive distribution. We can see that our simulated data does not correspond too well with our observed data. We can see at very low observed cases our models track relatively well with the observed data but for cases up to c.270, our models have consistently higher densities (plots shifted upwards based on observed density). We can see between c.550 and 650 cases our models have higher densities than the observed density and for case numbers >750 our models have lower densities than the observed data.

As in the analysis of the posterior predictive distribution, the Bayesplot comparison of simulated densities from the posterior predictive distribution and the observed density of cases indicates a poor model fit.

I will now predict COVID-19 case counts for 5 days into the future (i.e. days 101 - 105 inclusive), using the fitted stan model. I have used a generated quantities block in stan to compute lambda values for each of the 5 days and use these values to extract simulated values from a poisson distribution, thus producing case number estimates for each of the 5 days.

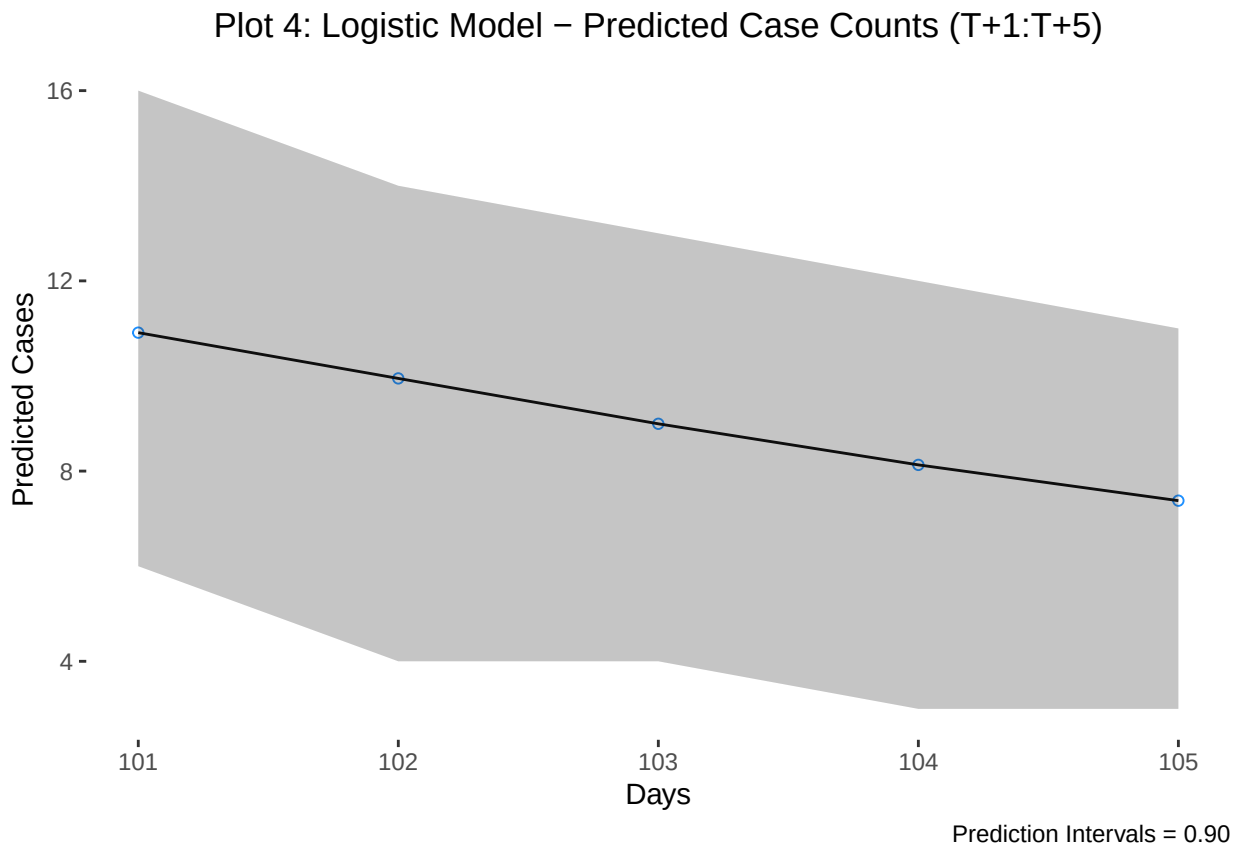
```
# Extracting predicted case counts for 5 days into the future
# i.e. days 101 - 105 inclusive
pred_days1 <- extract(fit_q1)$cases_pred
days_predicted <- seq(101,105,1)
avg_pred_days1 <- colMeans(pred_days1)

# print output of the fitted model
print(fit_q1, probs = c(0.05, 0.5, 0.95), pars = c("theta_1","theta_2","theta_3",
                                                   "cases_pred[1]","cases_pred[2]",
                                                   "cases_pred[3]","cases_pred[4]",
                                                   "cases_pred[5]"))
```

```
## Inference for Stan model: anon_model.
```

```
## 4 chains, each with iter=5000; warmup=2500; thin=1;
## post-warmup draws per chain=2500, total post-warmup draws=10000.
##
##                    mean se_mean     sd       5%      50%      95% n_eff Rhat
## theta_1       25534.93    1.61 161.99 25271.16 25533.95 25803.79 10132    1
## theta_2          45.94    0.00   0.11    45.76    45.94    46.13 10058    1
## theta_3           0.10    0.00   0.00     0.10     0.10     0.10 10003    1
## cases_pred[1]    10.91    0.03   3.26     6.00    11.00    16.00  9979    1
## cases_pred[2]     9.95    0.03   3.15     5.00    10.00    15.00 10207    1
## cases_pred[3]     8.99    0.03   3.01     4.00     9.00    14.00  9951    1
## cases_pred[4]     8.13    0.03   2.88     4.00     8.00    13.00 10145    1
## cases_pred[5]     7.38    0.03   2.71     3.00     7.00    12.00 10176    1
##
## Samples were drawn using NUTS(diag_e) at Tue Mar 14 00:56:54 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

As we can see from the stan model output above, the expected COVID-19 cases for each of days 101-105 are, 11, 10, 9, 8 and 7 respectively. We also observe the 90% credible interval for the predicted case numbers for each of the 5 days. In general, the 90% credible interval is between +- 5 and +- 4 days of the mean predicted values.

```
# Calculating 90% posterior credible interval for the mean (lambda) across each day
y_hdi_pred1 <- HDInterval::hdi(pred_days1,credMass = 0.90)
hpdi_pred_l1 <- y_hdi_pred1[1,]
hpdi_pred_u1 <- y_hdi_pred1[2,]

# Extracting predicted cases from above model to generate prediction value for each day
yphdi_pred1 <- HDInterval::hdi(pred_days1,credMass = 0.90)
pi_pred_l1 <- yphdi_pred1[1,]
pi_pred_u1 <- yphdi_pred1[2,]

# plotting 90% prediction interval for the mean and 90% credible interval for predicted cases 5 days in
p <- ggplot()

p4 <- p +
  geom_point(aes(days_predicted, avg_pred_days1), shape = 1, color = 'dodgerblue') +
  geom_line(aes(days_predicted,avg_pred_days1)) +
  geom_ribbon(mapping = aes(days_predicted, ymin = hpdi_pred_l1, ymax = hpdi_pred_u1), alpha = .2) +
  geom_ribbon(mapping = aes(days_predicted, ymin = pi_pred_l1, ymax = pi_pred_u1), alpha = .1) +
  ggtitle("Plot 4: Logistic Model - Predicted Case Counts (T+1:T+5)") +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(caption = "Prediction Intervals = 0.90",
       x = "Days",
       y = "Predicted Cases")

p4
```



To represent the predicted case numbers graphically, I have produced a plot above showing the predicted case numbers for each day (blue circles) and 90% credible interval for each predicted value (grey shaded area).

Question 2

In question 2, we develop another Bayesian model for the number of daily COVID-19 cases in Ireland, this time using a Gompertz function for λ_t. This model is saved as q2.stan and is represented mathematically below. The derivation of $\lambda_g(t)$ from the derivative of the Gompertz function $g(t)$ can be found in the appendix.

$$y_t \sim Po(\lambda_g(t))$$

$$g'(t) := \lambda_g(t) = -\theta_1\theta_2\theta_3^t ln(\theta_3)e^{-\theta_2\theta_3^t}$$

All three parameters are positive and can be interpreted as follows:
θ_1 is the maximum value that $g(t)$ takes and so can take potentially large values depending on the final size of the epidemic.
θ_2 controls the displacement of the curve $g(t)$ along the x-axis. The larger it's value the more the curve is shifted to the right. It is a positive parameter.
θ_3 is the growth rate of the function and therefore takes positive values between 0 and 1.

The same prior distributions for each of the above parameters are used as was used in the Logistic model. As such, I will not discuss the appropriateness of the choice of prior distribution for each of the 3 parameters as I have already discussed this in question 1 above.

$$\theta_1 \sim N(1000, 100000)$$

Where θ_1 is truncated at 0 to ensure positivity.

$$\theta_2 \sim N(50, 100)$$

where θ_2 is truncated below at 0 and above at 100, so that $\theta_2 \in [0, 100]$.

$$\theta_3 \sim U(0, 1)$$

I will now implement the above model using stan, see q2.stan file for reference. I fit the model using 5,000 iterations and the default 4 chains. I set the seed in order to ensure reproducible results.

```
# organising data into a list
dat2 <- list(T = nrow(data),
             days = data$Days,
             cases = data$Cases)

# fitting the model using stan
fit_q2 <- stan(file = 'q2.stan',data = dat2,seed = 1759,iter = 5000)
```

```
# print output of the fitted model
print(fit_q2, probs = c(0.05, 0.5, 0.95), pars = c("theta_1","theta_2","theta_3"))
```

```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=5000; warmup=2500; thin=1;
## post-warmup draws per chain=2500, total post-warmup draws=10000.
##
##             mean se_mean     sd       5%      50%      95% n_eff Rhat
## theta_1 25564.99    2.22 160.53 25305.06 25564.60 25825.99  5222    1
## theta_2    14.20    0.00   0.19    13.89    14.20    14.52  3632    1
## theta_3     0.94    0.00   0.00     0.93     0.94     0.94  3641    1
##
## Samples were drawn using NUTS(diag_e) at Tue Mar 14 00:57:26 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

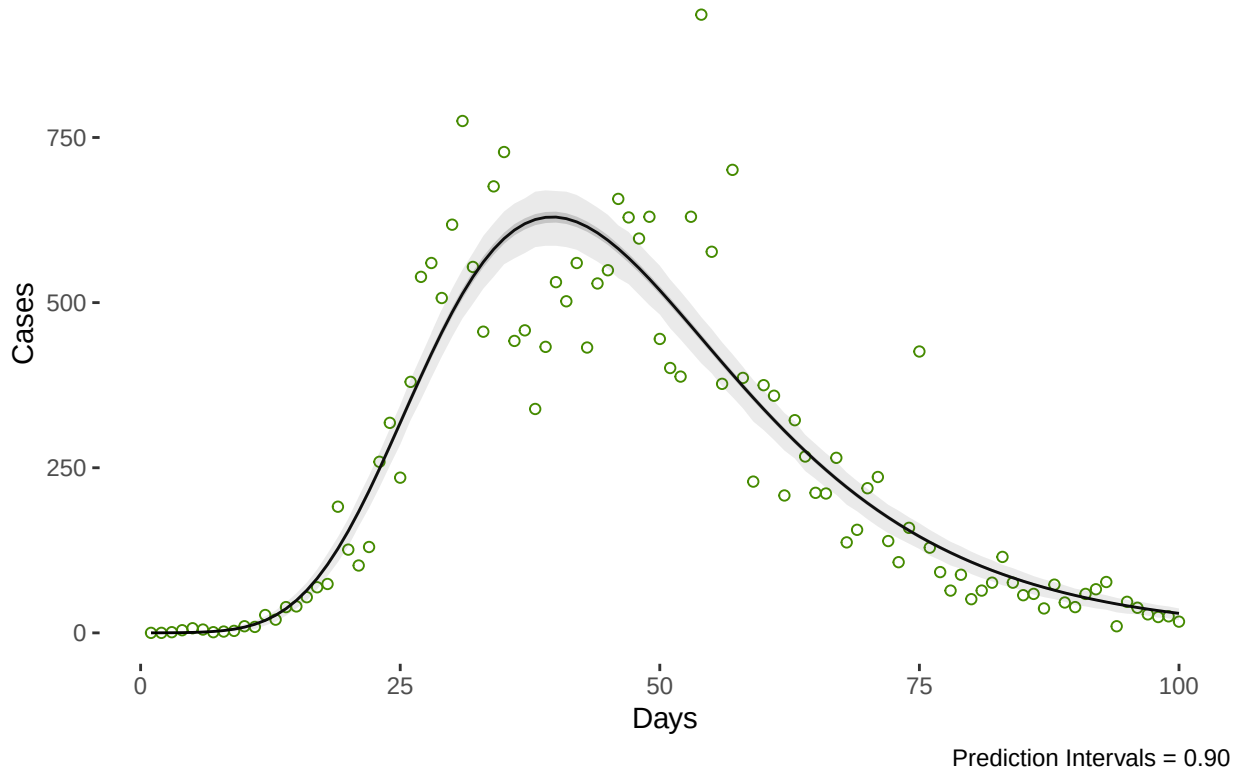
We can see the output of our fitted Bayesian model above. The average values for each of the three parameters is 25564.99, 14.20 and 0.94 for θ_1, θ_2 and θ_3 respectively. We also observe the 90% credible intervals for each of these parameters.

Next, I extract the lambda values from the output of the stan model in order to plot the posterior mean line, 90% credible interval for the mean and also the 90% credible interval for the predicted number of cases each day.

```
# Extracting fitted lambda values from STAN model
lambda2 <- extract(fit_q2)$lambda

# Calculating posterior mean line (lambda) across each day
avg_lambda2 <- colMeans(lambda2)

# Calculating 90% posterior credible interval for the mean (lambda) across each day
```

```
y_hdi2 <- HDInterval::hdi(lambda2,credMass = 0.90)
hpdi_l2 <- y_hdi2[1,]
hpdi_u2 <- y_hdi2[2,]

# Extracting predicted cases from above model to generate prediction value for each day
y_pred2 <- extract(fit_q2)$cases_rep
yphdi2 <- HDInterval::hdi(y_pred2,credMass = 0.90)
pi_l2 <- yphdi2[1,]
pi_u2 <- yphdi2[2,]

# Extracting log-likelihood (predicted) values (for use in model comparison and LOO package)
log_lik2 <- extract(fit_q2)$log_lik
```

```
# plotting 90% prediction interval for the mean and 90% credible interval for predicted cases
p <- ggplot()

p5 <- p +
  geom_point(data = data,
      aes(Days, Cases), shape = 1, color = 'chartreuse4') +
  geom_line(data = data,
            aes(Days,avg_lambda2)) +
  geom_ribbon(data = data,
            mapping = aes(Days, ymin = hpdi_l2, ymax = hpdi_u2), alpha = .2) +
  geom_ribbon(data = data,
      mapping = aes(Days, ymin=pi_l2, ymax=pi_u2), alpha = .1) +
  ggtitle("Plot 5: Gompertz Model - Posterior Predictive") +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(caption = "Prediction Intervals = 0.90")

p5
```

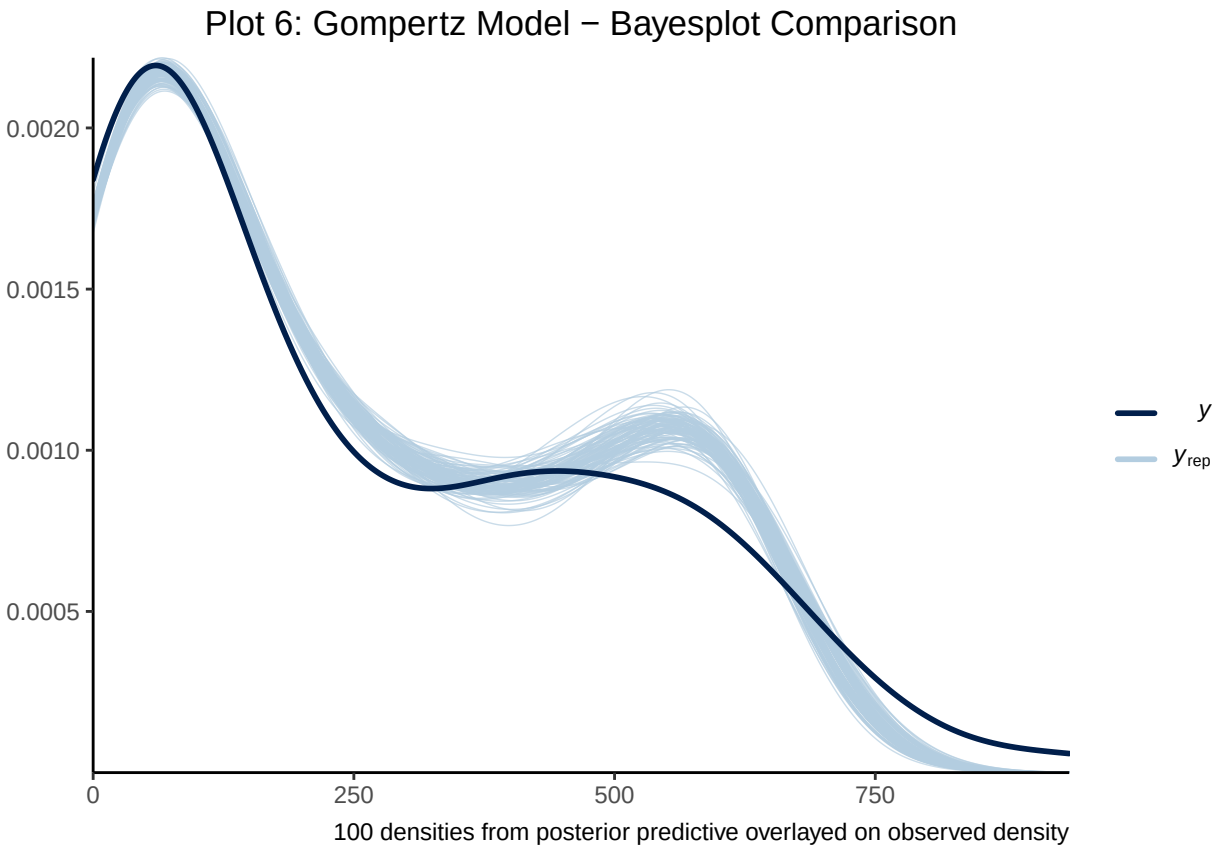


As we can see from the above posterior predictive distribution plot against the observed data points, the Gompertz model fits the data better than the logistic model however, the model still fails to adequately explain the variance in COVID-19 daily case data. In contrast to the logistic model, the Gompertz model fits the data well In the early stage of the epidemic (Days 0-20). Most of the observed data points are within the 90% credible interval for predicted cases.

However, in the remaining stages (days 20+), we can see that the majority of the data lies outside of the 90% credible interval. This is similar to the logistic model from question 1. Again, these aspects of the plot all indicate poor model fit.

I will now use the Bayesplot package to overlay a collection of densities of replicated data sets produced from the posterior predictive distribution and compare these to the observed data.

```
# overlay 100 densities from posterior predictive over the observed dataset
p <- ggplot()

p6 <- ppc_dens_overlay(data$Cases, y_pred2[1:100,]) +
  theme_classic() +
  ggtitle("Plot 6: Gompertz Model - Bayesplot Comparison") +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(caption = "100 densities from posterior predictive overlayed on observed density")

p6
```



From the above Bayesplot, we can see that the Gompertz model is a better fit to the data compared to that of the logistic model as the simulated densities from the predictive posterior distribution align more closely to the observed density. There are still areas where the Gompertz posterior predictive densities are higher that the observed density (notably cases between c.500 - c.650). We also observe areas where the simulated densities are lower than the observed (notably cases 750+).

As in the analysis of the posterior predictive distribution, the Bayesplot comparison of simulated densities from the posterior predictive distribution and the observed density of cases indicates a better fit to the data than that of the logistic model, but overall still a relatively poor model fit.

I will now predict COVID-19 case counts for 5 days into the future using the fitted stan model as before.

```
# Extracting predicted case counts for 5 days into the future
# i.e. days 101 - 105 inclusive
pred_days2 <- extract(fit_q2)$cases_pred
days_predicted <- seq(101,105,1)
avg_pred_days2 <- colMeans(pred_days2)

# print output of the fitted model
print(fit_q2, probs = c(0.05, 0.5, 0.95), pars = c("theta_1","theta_2","theta_3",
                                                   "cases_pred[1]","cases_pred[2]",
                                                   "cases_pred[3]","cases_pred[4]",
                                                   "cases_pred[5]"))
```

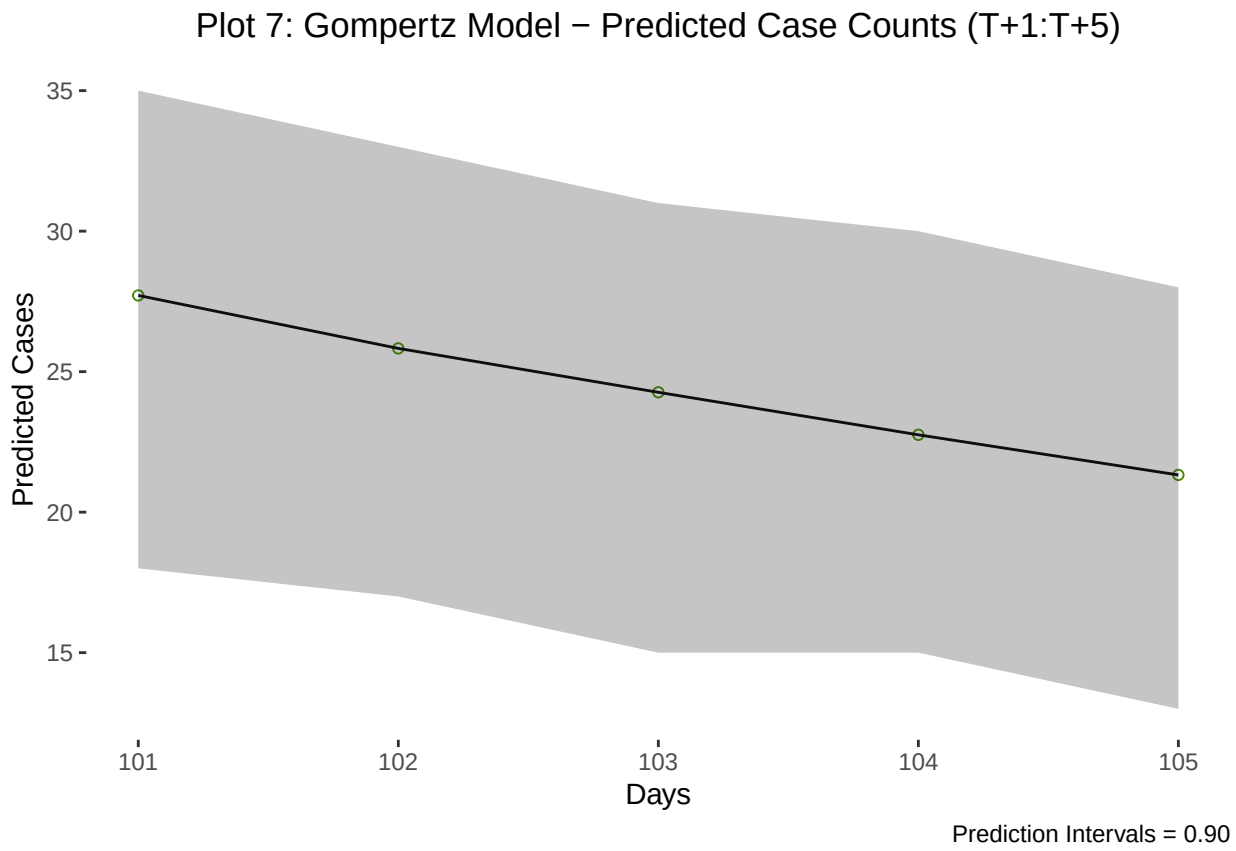
```
## Inference for Stan model: anon_model.
## 4 chains, each with iter=5000; warmup=2500; thin=1;
## post-warmup draws per chain=2500, total post-warmup draws=10000.
```

```
##
##                    mean se_mean     sd       5%      50%      95% n_eff Rhat
## theta_1       25564.99    2.22 160.53 25305.06 25564.60 25825.99  5222    1
## theta_2          14.20    0.00   0.19    13.89    14.20    14.52  3632    1
## theta_3           0.94    0.00   0.00     0.93     0.94     0.94  3641    1
## cases_pred[1]    27.71    0.05   5.31    19.00    28.00    37.00  9702    1
## cases_pred[2]    25.83    0.05   5.07    18.00    26.00    34.00  9680    1
## cases_pred[3]    24.27    0.05   4.95    16.00    24.00    33.00  9432    1
## cases_pred[4]    22.75    0.05   4.75    15.00    23.00    31.00  9906    1
## cases_pred[5]    21.32    0.05   4.65    14.00    21.00    29.00  9875    1
##
## Samples were drawn using NUTS(diag_e) at Tue Mar 14 00:57:26 2023.
## For each parameter, n_eff is a crude measure of effective sample size,
## and Rhat is the potential scale reduction factor on split chains (at
## convergence, Rhat=1).
```

As we can see from the stan model output above, the expected cases for each of days 101-105 are, 28, 26, 24, 23 and 21 respectively. We also observe the 90% credible interval for the predicted case numbers for each of the 5 days.

These predicted case values are noticeably larger than the predicted case values from the logistic model. We also observe a larger 90% credible interval than that of the logistic model with 90% of the data lying between +- 9 and +- 7 days of the predicted values.

```
# Calculating 90% posterior credible interval for the mean (lambda) across each day
y_hdi_pred2 <- HDInterval::hdi(pred_days2,credMass = 0.90)
hpdi_pred_l2 <- y_hdi_pred2[1,]
hpdi_pred_u2 <- y_hdi_pred2[2,]

# Extracting predicted cases from above model to generate prediction value for each day
yphdi_pred2 <- HDInterval::hdi(pred_days2,credMass = 0.90)
pi_pred_l2 <- yphdi_pred2[1,]
pi_pred_u2 <- yphdi_pred2[2,]

# plotting 90% prediction interval for the mean and 90% credible interval for predicted cases 5 days in
p <- ggplot()

p7 <- p +
  geom_point(aes(days_predicted, avg_pred_days2), shape = 1, color = 'chartreuse4') +
  geom_line(aes(days_predicted,avg_pred_days2)) +
  geom_ribbon(mapping = aes(days_predicted, ymin = hpdi_pred_l2, ymax = hpdi_pred_u2), alpha = .2) +
  geom_ribbon(mapping = aes(days_predicted, ymin = pi_pred_l2, ymax = pi_pred_u2), alpha = .1) +
  ggtitle("Plot 7: Gompertz Model - Predicted Case Counts (T+1:T+5)") +
  theme(plot.title = element_text(hjust = 0.5)) +
  labs(caption = "Prediction Intervals = 0.90",
       x = "Days",
       y = "Predicted Cases")

p7
```



To represent the predicted case numbers graphically, I have produced a plot above showing the predicted case numbers for each day.

Question 3

In question 3, we will compare the two previously developed Bayesian models using the "loo" package in R.

```
# comparing the two models using the loo package
loo1 <- loo(fit_q1)
loo2 <- loo(fit_q2)
loo_compare(loo1,loo2)
```

```
##        elpd_diff se_diff
## model2    0.0       0.0
## model1 -924.0     247.9
```

Based on the above loo_compare function results we can see that model2 is clearly preferred over model1 in terms of out-of-sample performance. This is in line with our previous observations that the Gompertz model fits the observed data better. This also indicates that the Gompertz function is not over-fitted in comparison to the Logistic function as if this were the case we would see model1 performing better in the elpd test.

I also compare the models using the "waic" function below.

```
# comparing models using WAIC
waic1 <- loo::waic(log_lik1)
```

```
waic2 <- loo::waic(log_lik2)
loo_compare(waic1,waic2)
```

```
##        elpd_diff se_diff
## model2    0.0       0.0
## model1 -927.8     248.3
```

We can see above that we again clearly prefer the Gompertz model over the logistic model based on the WAIC which compliments our previous analyses.

Conclusion

In conclusion, while the Gompertz function fits the data better than the Logistic function, neither model is a great fit to the data. I believe this is due to the overly simplistic one-parameter poisson model for the number of daily COVID-19 cases. Neither model currently takes into account any external explanatory variables such as behavioural factors, mobility of people, government restrictions etc. I believe if a more-complex model was implemented to explain the variation in daily cases then a better model fit could be achieved and both in and out-of-sample performance could be improved.

I also believe that the COVID-19 daily case data for Ireland is particularly noisy and thus makes it quite difficult to accurately model. There was a known anomaly in daily case data which began in April-2020 as a result of outsourced COVID-19 tests being sent back from a German laboratory in bulk and therefore artificially inflating the case numbers on the days when the results of these outsourced tests were recorded (and subsequently deflating the days where the patients actually tested positive for COVID-19). This anomaly can be observed quite clearly in some of the data points and would correspond to the period between days 35-55 where we can see some large outliers (occurring on the days when the outsourced tests were recorded in bulk). See https://www.sciencedirect.com/science/article/pii/S2211883720300952 for details.

I believe if these anomalies were corrected for in the data, a better model fit could be achieved.

Appendix

Question 1

Below I show the derivation of $\lambda_l(t)$ from the derivative of the logistic function $l(t)$. As we are interested in modelling the daily number of COVID-19 cases we need to differentiate the logistic function (which models the cumulative number of cases after t days) in order to derive an appropriate model for $\lambda_l(t)$.

The logistic function can be described by the below function.

$$l(t) = \frac{\theta_1}{1 + e^{-\theta_3(t-\theta_2)}}, for t = 1, ...T$$

All three parameters are positive and can be interpreted as follows:
θ_1 is the maximum value that $l(t)$ takes,
θ_2 is the value of t corresponding to the mid-point of the $l(t)$ function and,
θ_3 is the growth rate of the function.

The derivative of the logistic function can be derived using the quotient rule as shown below.

$$\frac{df(t)}{d(t)} = \frac{v(t)u'(t) - u(t)v'(t)}{v(t)^2}$$

$$v(t) = 1 + e^{-\theta_3(t-\theta_2)}, v'(t) = -\theta_3 e^{-\theta_3(t-\theta_2)}$$

$$u(t) = \theta_1, u'(t) = 0$$

therefore by the Quotient rule,

$$l'(t) := \lambda_l(t) = \frac{(1 + e^{-\theta_3(t-\theta_2)}.0) - (\theta_1. - \theta_3 e^{-\theta_3(t-\theta_2)})}{(1 + e^{-\theta_3(t-\theta_2)})^2}$$

which simplifies to,

$$\lambda_l(t) = \frac{\theta_1\theta_3 e^{(-\theta_3(t-\theta_2))}}{(1 + e^{-\theta_3(t-\theta_2)})^2}$$

Question 2

Below I show the derivation of $\lambda_g(t)$ from the derivative of the Gompertz function $g(t)$. As we are interested in modelling the daily number of COVID-19 cases we need to differentiate the Gompertz function (which models the cumulative number of cases after t days) in order to derive an appropriate model for $\lambda_g(t)$.

The Gompertz function can be described by the below function

$$g(t) = \theta_1 e^{-\theta_2\theta_3^t}, for t = 1, ...T$$

All three parameters are positive and can be interpreted as follows:
θ_1 is the maximum value that $l(t)$ takes,
θ_2 controls the displacement of the curve along the x-axis,
θ_3 is the growth rate of the function.

The derivative of the Gompertz function can be derived using the chain rule as shown below.

$$g'(t) := \lambda_g(t) = \frac{d}{dt}(\theta_1 e^{-\theta_2\theta_3^t})$$

$$\lambda_g(t) = \theta_1 \frac{d}{dt}(e^{-\theta_2\theta_3^t})$$

By the chain rule this is equal to,

$$\lambda_g(t) = \theta_1(e^{-\theta_2\theta_3^t} \frac{d}{dt}(-\theta_2\theta_3^t))$$

$$\lambda_g(t) = \theta_1(e^{-\theta_2\theta_3^t}(-\theta_2\theta_3^t ln(\theta_3)))$$

$$\lambda_g(t) = \theta_1(-\theta_2\theta_3^t ln(\theta_3)e^{-\theta_2\theta_3^t})$$

which simplifies to,

$$\lambda_g(t) = -\theta_1\theta_2\theta_3^t ln(\theta_3)e^{-\theta_2\theta_3^t}$$