MTHM506 - Statistical Data Modelling

Joshua Harrison

2025-02-20

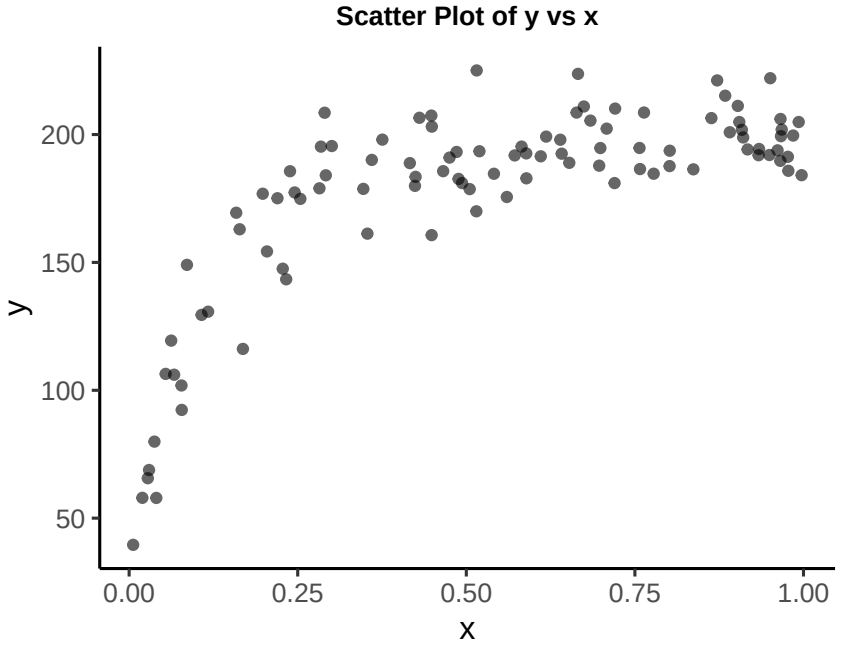
AI-supported/AI-integrated use is permitted in this assessment. I acknowledge the following uses of GenAI tools in this assessment:

- I have used GenAI tools to proofread and correct grammar or spelling errors
- I have used GenAI to help cross-check coding/plotting

I declare that I have referenced use of GenAI outputs within my assessment in line with the University referencing guidelines.

Question 1: Non-Linear Model Estimation

The dataframe `nlmodel` contains data on a response variable y and a single explanatory variable x. A scatter plot of y versus x suggests a strong non-linear relationship:



Suppose for these data we wish to consider the model:

$$Y_i \sim N\left(\frac{\theta_1 x_i}{\theta_2 + x_i}, \sigma^2\right)$$

$$i = 1, 2, \ldots, 100, \quad Y_i \text{ independent}$$

(a) Why this model can't be fit using a linear regression model

The model shown above specifies that the mean response is:

$$\mu(x) = \frac{\theta_1 x_i}{\theta_2 + x_i}$$

This expression is not linear with respect to the parameters θ_1 and θ_2. A linear regression model assumes that the relationship between the response variable and predictor is of the form:

$$Y = \beta_0 + \beta_1 x + \epsilon$$

This is where β_0 and β_1 are both unknown parameters with ϵ representing the random error. However, in our model, θ_1 is in the numerator with θ_2 being in the denominator, creating fractional non-linearity. The mean response function of the model is non-linear in parameters and cannot be written as a linear combination of θ_1 and θ_2. Therefore, due to the non-linear relationship between the response and parameters, a linear model is not feasible.

(b) Writing down the likelihood $L(\theta_1, \theta_2, \sigma^2; y, x)$ and the log-likelihood $\ell(\theta_1, \theta_2, \sigma^2; y, x)$:

The likelihood function is given by:

$$L(\theta_1, \theta_2, \sigma^2; y, x) = \prod_{i=1}^{n} \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(y_i - \frac{\theta_1 x_i}{\theta_2 + x_i})^2}{2\sigma^2}\right)$$

The log-likelihood function $\ell(\theta_1, \theta_2, \sigma^2; y, x)$ is the natural logarithm of the likelihood function shown above. Therefore simplifying it we get:

$$\ell(\theta_1, \theta_2, \sigma^2; y, x) = -\frac{n}{2}\log(2\pi\sigma^2) - \frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - \frac{\theta_1 x_i}{\theta_2 + x_i})^2$$

(c) Defining the Negative Log-Likelihood Function

The goal for Maximum Likelihood Estimation is to maximise the likelihood function, but most numerical optimisation functions such as nlm() in R are designed to minimise a function. Therefore, we take the negative log-likelihood, so that minimising the negative log likelihood is equivalent to maximising the likelihood.

```
# Define the negative log-likelihood function

mylike <- function(params, y, x) {
  theta1 <- params[1]
  theta2 <- params[2]
  sigma <- abs(params[3])

# Number of Observations

  n <- length(y)

# Mean function

  mu <- (theta1 * x) / (theta2 + x)

# Computing log-likelihood

  logL <- - (n /2) * log(2 * pi * sigma^2) - sum((y - mu)^2) / (2 * sigma^2)
```

```
# Return negative log-likelihood

  return(-logL)

}
```

(d) Finding Maximum Likelihood Estimates (MLE)

Before estimating the parameters, we need to provide **sensible starting values** for the optimisation algorithm. This is particularly important because functions such as `nlm()` are sensistive to intial conditions and can converge to local optima.

```
x <- nlmodel$x
y <- nlmodel$y

# Choosing starting values:

theta1_start <- (max(y) - min(y)) / (max(x) - min(x))

theta2_start <- median(x)

sigma_start <- sd(y)

#Combining into a vector

start_values <- c(theta1_start, theta2_start, sigma_start)
```

For θ_1: I have chosen to use the range of y divided by the range of x. This is because it provides an estimate to the rate of change in y with respect to x.

For θ_2: I have chosen to use the median of x instead of the mean, in order to avoid extreme values in the denominator.

For σ: I have chosen to use the standard deviation of y as an estimate for the error variance.

After I chose my start values, I combined them into a vector because the `nlm()` function requires the parameters to be passed as a single numeric vector and it ensures compatibility with the `mylike()` function.

```
# Minimising the negative log-likelihood using nlm()

fit <- nlm(mylike, p = start_values, y = y, x = x, hessian=TRUE)
```

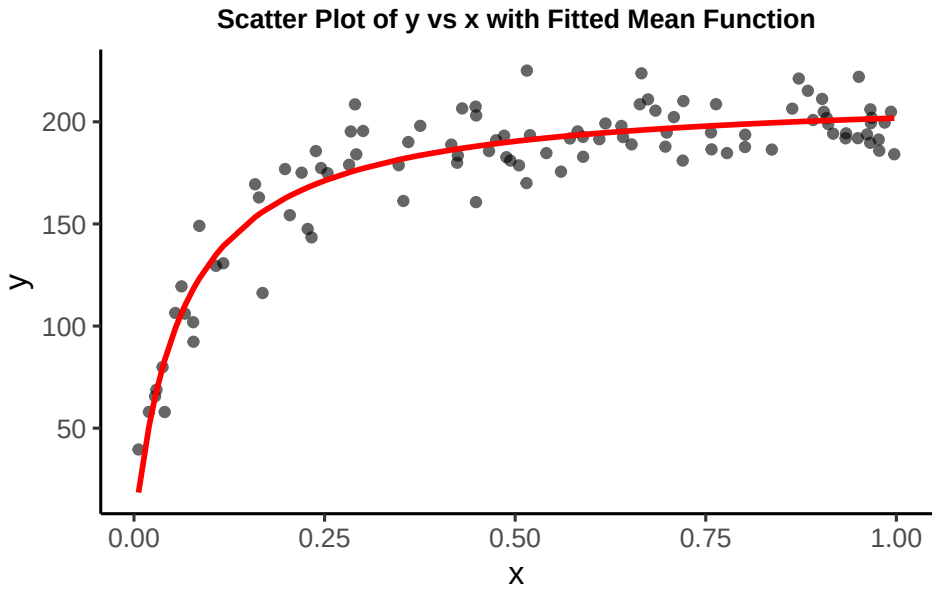
Maximum Likelihood Estimates (MLEs):

Theta1: 214.65

Theta2: 0.06353429

Sigma: 13.61564

```
# Plotting data and fitted mean function

fitted_mu <- (theta1_hat * x) / (theta2_hat + x)
```



The scatter plot shown above suggests a saturation effect, where increases in x lead to diminishing increases in y. The fitted mean function shown by the red line follows the general shape of the data, capturing the asymptotic trend.

(e) Computing standard errors and Confidence Intervals.

```r
# Computing Hessian and variance-covariance matrix

hessian_matrix <- fit$hessian
OIM <- solve(hessian_matrix) # Observed Information Matrix

# Extracting standard errors (square roots of diagonal elements)

se_theta1 <- sqrt(OIM[1, 1])
se_theta2 <- sqrt(OIM[2, 2])
se_sigma <- sqrt(OIM[3, 3])
```

The Hessian Matrix is the matrix of the second-order partial derivatives of the log-likelihood function, providing information about the curvature of the likelihood function. The Observed Information Matrix (OIM) is the inverse of the Hessian matrix and gives the estimated variance-covariance matrix of the parameter estimates. The diagonal elements of the OIM represent the variances of the parameter estimates. Standard errors can be calculated by taking the square roots of these diagonal elements as shown. The standard errors measure the uncertainty of the estimated parameters.

Standard Errors for θ_1 and θ_2:

Parameter	Standard.Error
Theta1	2.674793
Theta2	0.005140
Sigma	0.962764

```r
# Computing at a 95% confidence interval

ci_theta1 <- c(theta1_hat - 1.96 * se_theta1, theta1_hat + 1.96 * se_theta1)
ci_theta2 <- c(theta2_hat - 1.96 * se_theta2, theta2_hat + 1.96 * se_theta2)
```

Table 2: Standard Errors and 95% Confidence Intervals

Parameter	Std..Error	X95..CI.Lower	X95..CI.Upper
Theta1	2.675	209.407	219.893
Theta2	0.005140	0.053459	0.073609

The standard errors represent the variability of the parameter estimates. A smaller standard error suggests a more precise estimate. Theta1 has a standard error of 2.675 implying it has some variation, but its confidence interval is wide shown by the 95% CI of (209.407, 219.893). Theta2 has a much smaller standard error of 0.00514 which is very small, indicating a precise estimate for Theta2. Theta2 has a narrow confidence interval between 0.053459 and 0.073609, indicating that this parameter is well-estimated.

(f) Hypothesis Testing

Defining Hypotheses

We are testing whether $\theta_2 = 0.08$ at the 5% significance level:

- **Null Hypothesis (H_0)**:

$$\theta_2 = 0.08$$

- **Alternative Hypothesis (H_A)**:

$$\theta_2 \neq 0.08$$

  This is a **two tailed test**.

Computing the Test Statistic

The test statistic follows a *standard nomral distribution*:

$$Z = \frac{\hat{\theta}_2 - 0.08}{SE(\hat{\theta}_2)}$$

where:

- $\hat{\theta}_2$ is the **MLE estimate of** θ_2,
- $SE(\hat{\theta}_2)$ is the **standard error of** θ_2.

We calculate this statistic using our estimated values.

```
theta2_0 <- 0.08 #H0: theta2 = 0.08

# Computing z-test statistic

Z_score <- (theta2_hat - theta2_0) / se_theta2
Z_score
```

[1] -3.203221

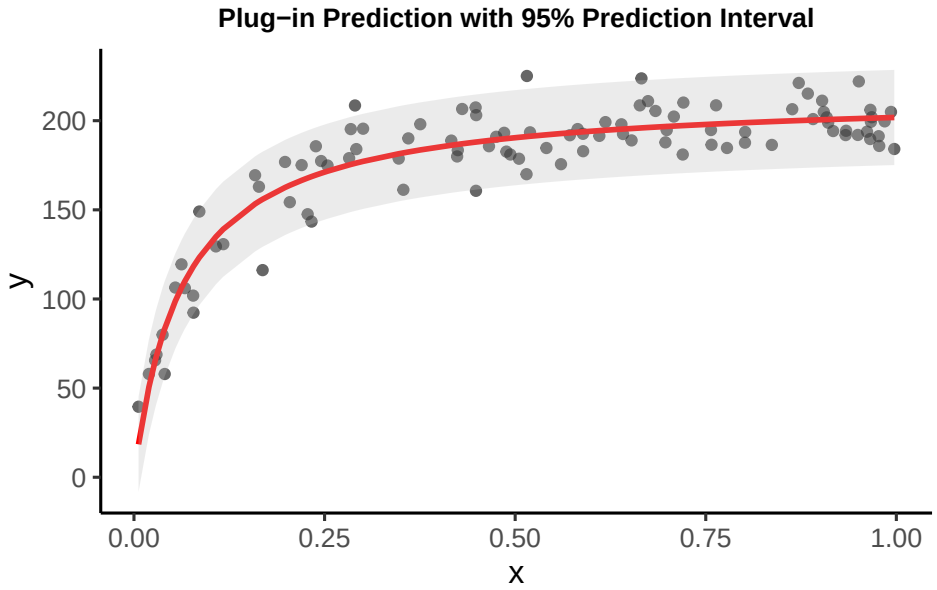
```
# Computing two-tailed p-value

p_value <- 2 * (1 - pnorm(abs(Z_score)))
p_value
```

[1] 0.001358996

Therefore the p-value is less than 0.05, so we reject H_0 and accept (H_A), concluding that θ_2 is significantly different from 0.08.

(g) Constructing 95% Prediction Intervals

```
predicted_y <- (theta1_hat * x) / (theta2_hat + x)

# Compute 95% prediction interval bounds

lower_bound <- predicted_y - 1.96 * sigma_hat
upper_bound <- predicted_y + 1.96 * sigma_hat
```



The shaded area in the graph above represents the 95% prediction interval, meaning that future observations are expected to fall within this range 95% of the time. The prediction interval captures most data points, meaning that the model has reasonable predictive accuracy. Some data points fall outside the prediction interval suggesting possible heteroskedasticity where the variance is not constant.

Question 2: Analysing Quartely AIDS Cases in the UK

The dataframe `aids` relates to the number of quarterly AIDS cases in the UK, y_i, from January 1983 to March 1994. The variable `cases` is y_i, and `date` is time, symbolised here as x_i. In this question we consider two competing models to describe the trend in the number of cases.

Model 1 follows a Poisson distribution:

$$Y_i \sim \text{Pois}(\lambda_i)$$

$$\log(\lambda_i) = \beta_0 + \beta_1 x_i$$

Model 2 follows a Normal distribution:

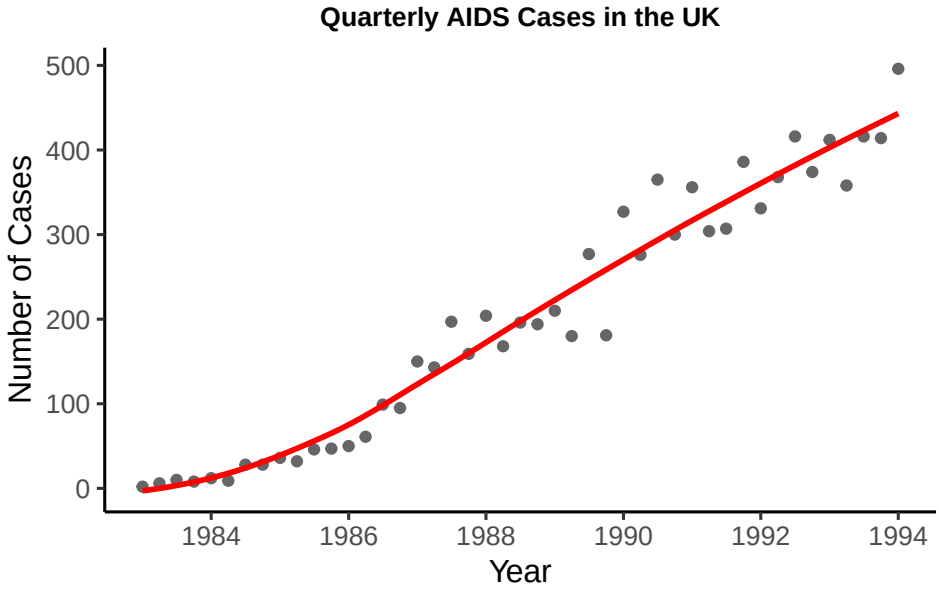
$$Y_i \sim N(\mu_i, \sigma^2)$$

$$\log(\mu_i) = \gamma_0 + \gamma_1 x_i$$

(a) Exploratory Data Analysis (EDA)

Before fitting models to describe the trend in the number of AIDS cases, we will understand the relationship, structure and trends within the data. First, the dates are converted to date format to facilitate plotting. We have calculated a mean number of cases, which can provide as a reference point for the trend. Using ggplot, we have then created a scatter plot below displaying AIDS cases over time with a LOESS (Locally Estimated Scatterplot Smoothing) curve to capture trends in the data.

```
aids$date <- as.Date(as.yearqtr(aids$date, format="%YQ%q"))

year_corrected <- year(aids$date) + 1900

aids$date <- make_date(year_corrected, month(aids$date), day(aids$date))

mean_cases <- mean(aids$cases, na.rm = TRUE)
```



As shown by the scatter plot above which maps y_i (number of cases) against x_i (time), we can see that the mean number of cases is increasing exponentially over time. This supports a log-linear relationship displayed by the red trend line.

The Poisson distribution model (Model 1) may be a suitable choice since it assumes the mean number of cases follows this exponential trend. From the graph, the observed variance increases as the mean increases. However, the Poisson model may not be the best pick due to over dispersion where the observed variance in the data is significantly greater than the mean. This violates the Poisson distribution assumption that the mean and variance should be equal, suggesting that a Negative Binomial model could be more appropriate.

The Normal distribution model (Model 2) could also work given the data is following an exponential-like trend, therefore a log-transformed Normal model could be suitable. However, this depends on if the data is symmetrically distributed after transformation. If the transformed data remains skewed then the Normal model may not be the best fit.

(b) Model Fitting and Comparison

We will now fit both models using Poisson regression and Gaussian regression with a log-link.

Fitting Model 1:

$$\log(\lambda_i) = \beta_0 + \beta_1 x_i$$

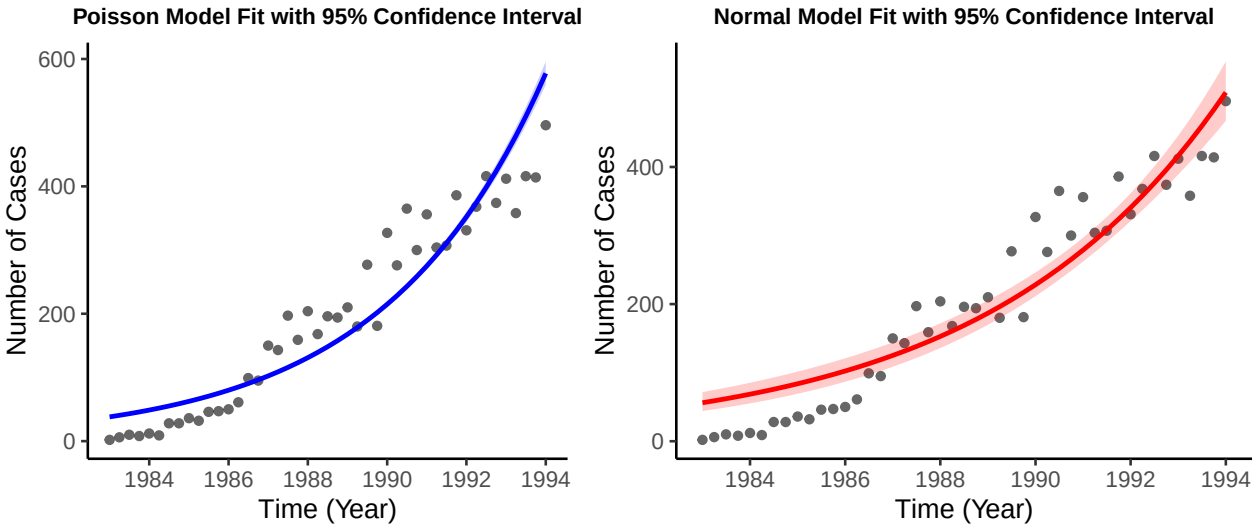
```
#Fitting the Poisson Model
model_1 <- glm(cases ~ date, data = aids, family = poisson(link = "log"))
```

Fitting Model 2:

$$\log(\mu_i) = \gamma_0 + \gamma_1 x_i$$

```
#Fitting the Normal Model
model_2 <- glm(cases ~ date, data = aids, family = gaussian(link = "log"))
```

```
# Compute Predictions and Confidence Intervals for Model 1 (Poisson)
poisson_pred <- predict(model_1, type = "link", se.fit = TRUE)
aids$poisson_fit <- exp(poisson_pred$fit)
aids$poisson_lwr <- exp(poisson_pred$fit - 1.96 * poisson_pred$se.fit)  # Lower CI
aids$poisson_upr <- exp(poisson_pred$fit + 1.96 * poisson_pred$se.fit)  # Upper CI


# Compute Predictions and Confidence Intervals for Model 2 (Normal)
normal_pred <- predict(model_2, interval = "confidence", se.fit = TRUE)
aids$normal_fit <- exp(normal_pred$fit)
aids$normal_lwr <- exp(normal_pred$fit - 1.96 * normal_pred$se.fit)  # Lower CI
aids$normal_upr <- exp(normal_pred$fit + 1.96 * normal_pred$se.fit)  # Upper CI
```



Validity of the Poisson Model (Model 1)

The poisson model assumes the number of cases follows a Poisson distribution, making it suitable for count data. The Poisson Model captures the overall increasing trend of the data well. The log-linear assumption is reasonable. However, there is potential over dispersion with variance in the data appearing larger than the model assumes. There are many underestimates between 1987 to 1992, with some points being consistently above the predicted mean. The confidence intervals also widen significantly over time, which suggests that there is increasing uncertainty in predictions.

Validity of the Normal Model (Model 2)

The Normal Model curve captures the general trend of the data reasonably well, much alike the Poisson Model. The confidence intervals are much wider in this model but are more stable, helping predict more

values between 1987 to 1992 compared to the Poisson Model. The log-transformation has linearised the relationship well. However, the assumption of normally distributed residuals may not hold. The model is not able to account for discrete nature of case counts like the Poisson Model can. The widening confidence intervals towards later years shows there is greater prediction uncertainty which is expected as case counts grow.

```
aic_results <- data.frame(
  Model = c("Poisson Model (Model 1)", "Normal Model (Model 2)"),
  AIC_Value = c(AIC(model_1), AIC(model_2))
)

kable(aic_results, caption = "AIC Comparison Models")
```

Table 3: AIC Comparison Models

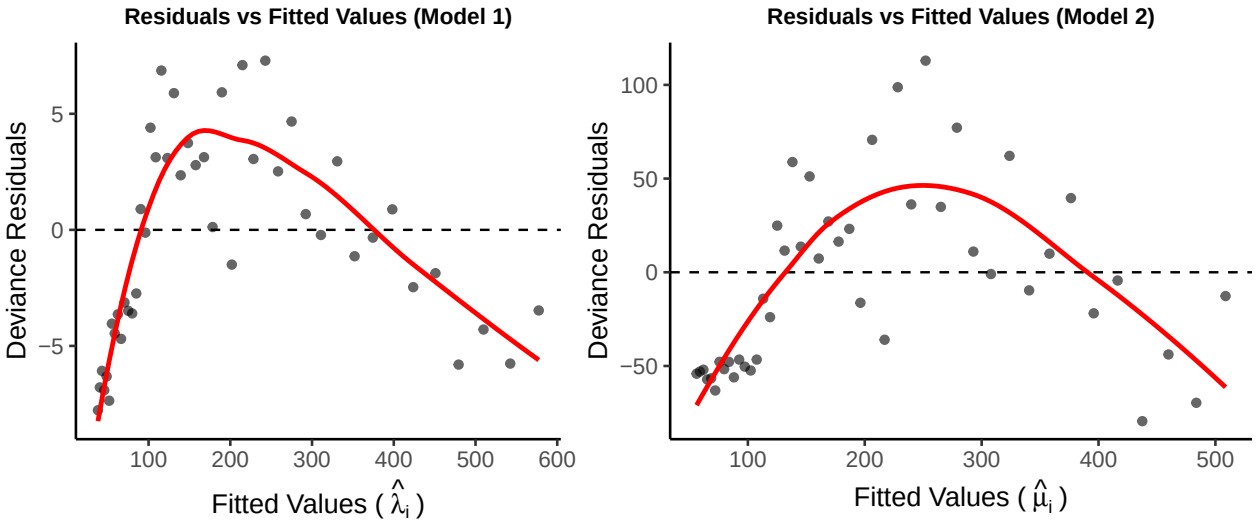
Model	AIC_Value
Poisson Model (Model 1)	1154.0897
Normal Model (Model 2)	482.8203

AIC (Akaike Information Criterion) is a measure used to compare models by assessing their relative quality. Models with lower AIC's are preferred because they explain the data well with fewer parameters. Looking at the AIC values above, the Normal Model (Model 2) has a much lower AIC value of 482.82 compared to the Poisson Model (Model 1) of 1154.09. This implies that the Normal Model (Model 2) fits the data better and is preferable compared to the Poisson Model (Model 1).

(c) Model Diagnostics - Deviance Residuals

In order to assess the models, the deviance residuals are plotted against the fitted values. These can be used to check whether the residuals exhibit any particular patterns or deviance from 0 that suggest a poor model fit.

```
# Computing residuals and fitted values for Poisson model
aids$poisson_resid <- residuals(model_1, type = "deviance")
aids$poisson_fitted <- fitted(model_1)

# Computing residuals and fitted values for Normal model
aids$normal_resid <- residuals(model_2, type = "deviance")
aids$normal_fitted <- fitted(model_2)
```



Poisson Model (Model 1) Residuals vs Fitted Values

The Poisson Model (Model 1) has a LOESS curve that is curved rather than flat around 0. The residuals are showing a trend rather than being randomly scattered. The residuals increase and then decrease which suggests non-linearity that the Poisson model does not capture. The spread of the residuals increases with more fitted values, indicating that there is possible overdispersion with the variance being greater than the mean.

In order to address the non-linearity, we could add higher-order terms such as quadratic or cubic terms to allow for a better fit for non-linear trends. This would make the residuals become more randomly scattered. A negative binomial model may be better suited to address the overdispersion shown from the Poisson model (Model 1) because they allow variance to be greater than the mean.

Normal Model (Model 2) Residuals vs Fitted Values

The Normal Model (Model 2) has a similar curved trend to the Poisson Model. The deviance residuals scale is much larger, particularly for larger fitted values. The range is shown to be much wider than the Poisson Model ranging from -50 to 100, suggesting heteroscedasticity. This means that residual variance is increasing with fitted values. The normality assumption may not hold well for this model.

To potentially improve the fit of the model, We could also add a quadratic term to allow for a better fit for non-linear trends much like with the poisson model. If the LOESS curve in the deviance residual graph flattens around 0 more so than the deviance residual graph shown above, then adding the quadratic term would have improved the model fit.

(d) Hypothesis Testing for Model Extensions

Defining Hypotheses

We perform hypothesis tests to determine whether adding a quadratic term ($date^2$) significantly improves each model.

Poisson Model:

- **Null Hypothesis (H_0)**: The relationship is strictly linear, meaning the quadratic term does not improve the model ($\beta_2 = 0$).
- **Alternative Hypothesis (H_A)**: The quadratic term significantly improves the model ($\beta_2 \neq 0$).

Normal Model:

- **Null Hypothesis (H_0)**: The relationship is strictly linear, meaning the quadratic term does not improve the model ($\beta_2 = 0$).
- **Alternative Hypothesis (H_A)**: The quadratic term significantly improves the model ($\beta_2 \neq 0$).

```
aids$numeric_date <- as.numeric(aids$date)

# Fit Poisson model with quadratic term
model_poisson_quad <- glm(cases ~ numeric_date + I(numeric_date^2), data = aids, family = poisson(link =

# Fit Normal model with quadratic term
model_normal_quad <- glm(cases ~ numeric_date + I(numeric_date^2), data = aids, family = gaussian(link =
```

```
# Likelihood Ratio Test for Poisson Model
poisson_lrt <- lrtest(model_1, model_poisson_quad)
poisson_lrt
```

```
## Likelihood ratio test
##
## Model 1: cases ~ date
## Model 2: cases ~ numeric_date + I(numeric_date^2)
##   #Df  LogLik Df  Chisq Pr(>Chisq)
## 1   2 -575.04
## 2   3 -273.08  1 603.94  < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Having fit the new Poisson Model with a quadratic term, the likelihood ratio test performed on the new model above compares the Simpler Model (model 1/poisson Model) with the Extended Model (model_poisson_quad). The new term added is the date^2 term. As shown above, the p-value is 2.2e-16, which is highly significant. Since p<0.05, we reject the null hypothesis and accept the alternate hypothesis: that the quadratic term significantly improves the model.

```
# ANOVA Test for Normal Model
normal_anova <- anova(model_2, model_normal_quad)
normal_anova
```

```
## Analysis of Deviance Table
##
## Model 1: cases ~ date
## Model 2: cases ~ numeric_date + I(numeric_date^2)
##   Resid. Df Resid. Dev Df Deviance      F    Pr(>F)
## 1        43     105323
## 2        42      46331  1    58993 53.478 5.205e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

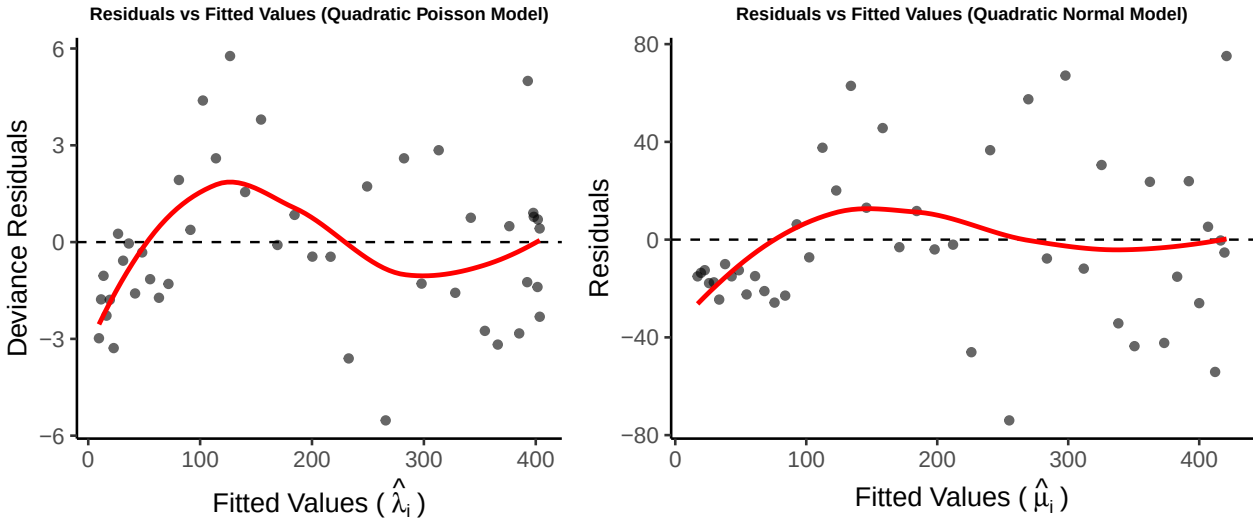
After fitting the extended normal model (model_normal_quad) with a quadratic term, the anova test performed above compares the simpler normal model (model_2) with the extended normal model. As shown above, the residual deviance for the simpler model was 105323 and the extended model's residual deviance is 46331, suggesting that the quadratic model explains more variation in the data. The F-statistic is high with a value of 53.478, showing that the model has an improved fit for the data. The p-value is 5.205e-09 which is much smaller than 0.05, so we reject the null hypothesis and accept the alternate hypothesis, concluding that the quadratic term significantly improves the model.

```
# Compare AIC for all models
AIC(model_1, model_poisson_quad, model_2, model_normal_quad)
```

```
##                      df       AIC
## model_1               2 1154.0897
## model_poisson_quad    3  552.1500
## model_2               3  482.8203
## model_normal_quad     4  447.8647
```

By adding the quadratic term to both models, the AIC values decrease showing better model fit and explanation of the data. The poisson model's AIC value was significantly decreased by over half, while the normal model's AIC value only dropped partially after the addition of the quadratic term.

Deviance Residuals for Extended Models



As we can see from the deviance residuals of the extended models, the quadratic poisson model has a flatter LOESS curve suggesting the quadratic term is better than the linear model but still not perfect. There is some non-random curvature, meaning that some structure remains unexplained. At higher fitted values, the residuals are more dispersed.

The quadratic normal model has a flatter LOESS curve from its simpler model but the model still displays heteroscedasticity (unequal spread of residuals). The residuals are spreading out as fitted values increase. This suggests that variance is increasing which violates the assumption of constant variance.

Updating hypothesis statement to improve models further

To consider adding higher-order polynomial terms to find the best fitting model, our hypothesis statements are updated to reflect the iterative model refinement process:

Poisson Model:

- **Null Hypothesis (H_0)**: The relationship between time and the number of cases is adequately captured by the current polynomial degree k (e.g., quadratic). Adding higher-order terms ($\beta_{k+1}, \beta_{k+2}, \ldots$) does not significantly improve the model.

$$H_0 : \beta_{k+1} = \beta_{k+2} = \cdots = 0$$

- **Alternative Hypothesis (H_A)**: Higher-order terms significantly improve model fit, meaning the true relationship is more complex than the current model.

$$H_A : \exists\, \beta_{k+1} \neq 0, \beta_{k+2} \neq 0, \ldots$$

Normal Model:

- **Null Hypothesis (H_0)**: The relationship between time and the log-transformed mean response is adequately captured by the current polynomial degree k (e.g., quadratic). Adding additional higher-order terms does not significantly improve model fit.

$$H_0 : \gamma_{k+1} = \gamma_{k+2} = \cdots = 0$$

- **Alternative Hypothesis (H_A)**: The relationship is better captured by a higher-degree polynomial model.

$$H_A : \exists\, \gamma_{k+1} \neq 0, \gamma_{k+2} \neq 0, \ldots$$

```
# Fitting cubic models
model_poisson_cubic <- glm(cases ~ numeric_date + I(numeric_date^2) + I(numeric_date^3),
                           data = aids, family = poisson(link = "log"))

model_normal_cubic <- glm(cases ~ numeric_date + I(numeric_date^2) + I(numeric_date^3),
                          data = aids, family = gaussian(link = "log"))

# Fitting the quartic models
model_poisson_quartic <- glm(cases ~ numeric_date + I(numeric_date^2) + I(numeric_date^3)
                             + I(numeric_date^4),
                             data = aids, family = poisson(link = "log"))

model_normal_quartic <- glm(cases ~ numeric_date + I(numeric_date^2) + I(numeric_date^3)
                            + I(numeric_date^4),
                            data = aids, family = gaussian(link = "log"))
```

```
# Perform Likelihood Ratio Tests
poisson_lrt_cubic <- lrtest(model_poisson_quad, model_poisson_cubic)
poisson_lrt_quartic <- lrtest(model_poisson_cubic, model_poisson_quartic)

# Extract p-values
p_cubic <- poisson_lrt_cubic$`Pr(>Chisq)`[2]  # Extract p-value for cubic test
p_quartic <- poisson_lrt_quartic$`Pr(>Chisq)`[2]  # Extract p-value for quartic test
```

Table 4: Likelihood Ratio Test Results for Poisson Model

Model_Comparison	P_Value
Quadratic vs. Cubic	0.0000000
Cubic vs. Quartic	0.4736985

Having fitted the cubic and quartic model extensions to the poisson model. We once again undergo the likelihood ratio test. The p value for cubic test is 2.2e-16 which is too small to show on the table above. This value is much smaller than p<0.05 and therefore we reject the null hypothesis for this model and accept the alternate hypothesis: Higher-order terms significantly improve model fit, meaning the true relationship is more complex than the current model.

For the quartic model however, the p-value is 0.4736 which is greater than 0.05 and therefore we accept the null hypothesis at this stage. Therefore the best fitting poisson model is the cubic model. This could be due to the fact that adding too many polynomial terms can lead to over fitting with the model capturing noise rather than the trend in the data.

```r
# Perform ANOVA Tests
normal_anova_cubic <- anova(model_normal_quad, model_normal_cubic)
normal_anova_quartic <- anova(model_normal_cubic, model_normal_quartic)

# Extract p-values
p_cubic <- normal_anova_cubic$`Pr(>F)`[2]  # Extract p-value for cubic test
p_quartic <- normal_anova_quartic$`Pr(>F)`[2]  # Extract p-value for quartic test
```

Table 5: ANOVA Test Results for Normal Model

Model_Comparison	P_Value
Quadratic vs. Cubic	0.0044614
Cubic vs. Quartic	0.9540059

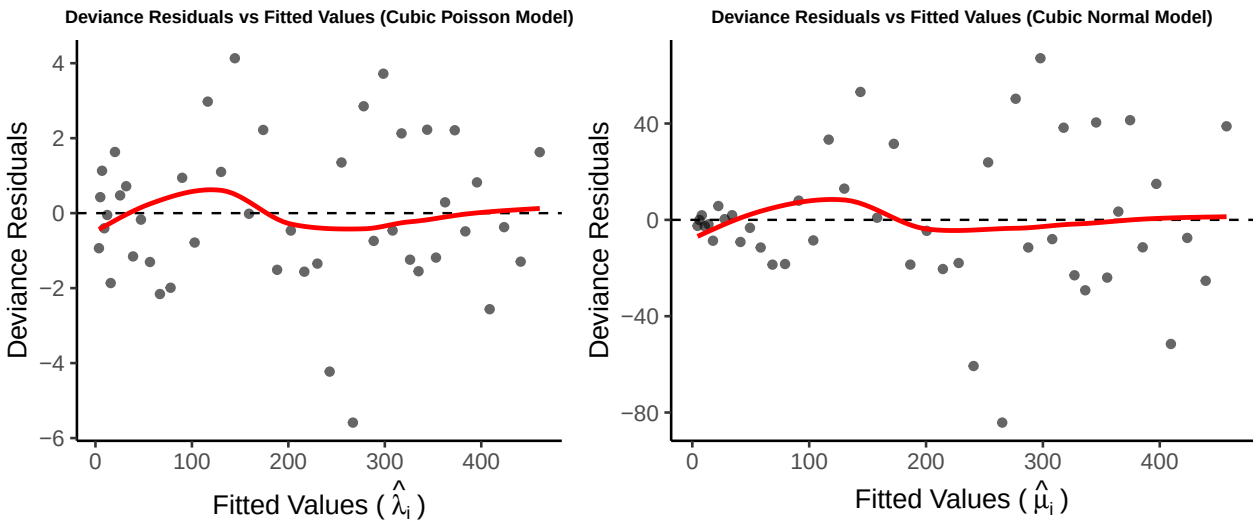
After adding the cubic and quartic terms to the normal quadratic model, the anova test performed shows that the cubic model significantly improves the models fit as the p value is 0.0044 which is less than 0.05. In this case we reject the null hypothesis and accept the alternate hypothesis that: The relationship is better captured by a higher-degree polynomial model.

However, when adding the quartic term, the p value is 0.954 which suggests that this term does not improve the model fit. Therefore, the final version for the normal model is the cubic normal model and we reject the use of the quartic term.

(e) Model Selection and Evaluation

In this section, we compare the final cubic models created above to determine which provides the best fit. Originally we fit two models (model 1 and 2). The first was a poisson model and the second was a normal model, both with log link functions. These models attempted to describe the trend between the number of AIDS cases in the UK between January 1983 to March 1994. We then extended these models to include higher order powers until we arrived at a final version for both models which was the cubic poisson model and cubic normal model.

```r
aids$poisson_cubic_resid <- residuals(model_poisson_cubic, type = "deviance")
aids$poisson_cubic_fitted <- fitted(model_poisson_cubic)

aids$normal_cubic_resid <- residuals(model_normal_cubic, type = "deviance")
aids$normal_cubic_fitted <- fitted(model_normal_cubic)
```



The deviance residuals vs fitted values for the cubic poisson model shown above is mostly scattered randomly, suggesting the model captures the trend well. This shows improvement over the lower-order models. There is still possible overdispersion with the variance of residuals being relatively large in areas.

The deviance residuals vs fitted values for the cubic normal model shown displays mostly random residuals around zero with the red LOESS curve remaining much closer to zero compared to the lower-order normal models. The spread of residuals still increases as fitted values increase, suggesting that the issue of heteroscedasticity still remains.

Comparing AIC's

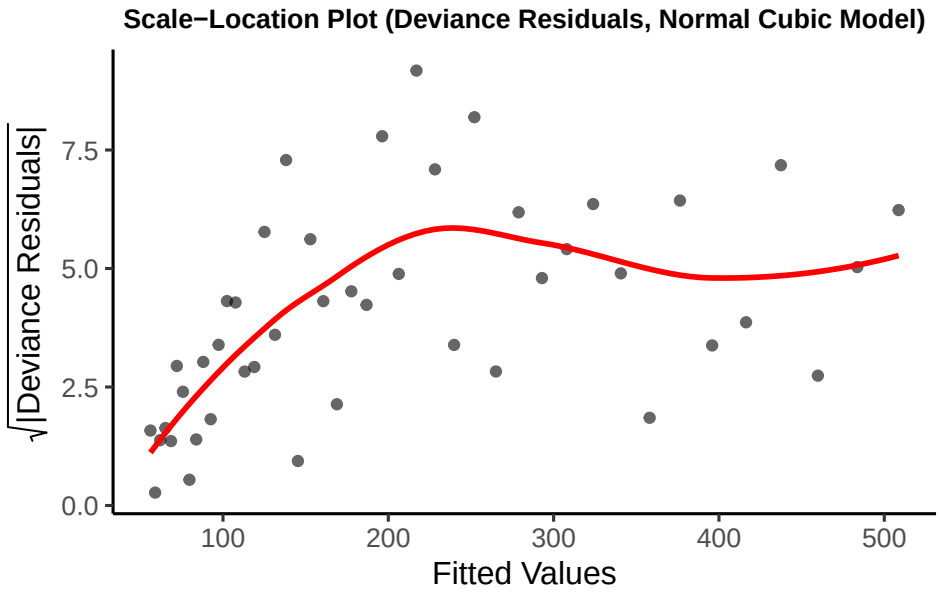
```
model_comparison <- data.frame(
  Model = c("Poisson Quadratic", "Poisson Cubic", "Normal Quadratic", "Normal Cubic"),
  Deviance = c(deviance(model_poisson_quad), deviance(model_poisson_cubic),
               deviance(model_normal_quad), deviance(model_normal_cubic)),
  AIC = c(AIC(model_poisson_quad), AIC(model_poisson_cubic),
          AIC(model_normal_quad), AIC(model_normal_cubic))
)


kable(model_comparison, caption = "Comparison of Deviance and AIC for Quadratic and Cubic Models")
```

Table 6: Comparison of Deviance and AIC for Quadratic and Cubic Models

Model	Deviance	AIC
Poisson Quadratic	250.2998	552.1500
Poisson Cubic	166.7820	470.6322
Normal Quadratic	46330.5545	447.8647
Normal Cubic	37947.6966	440.8830

To understand which model is the best out the cubic models, we can compare the AIC values in the table provided above. The AIC and deviance values of both poisson models again confirm that the cubic model is favorable (470.6322 vs 552.1500), showing that the cubic model fits the data better and improves the fit without excessive complexity. The AIC value for the cubic normal is also lower with a value of 440.88 which is lower than the normal quadratic AIC value of 447.86. This shows that the cubic model has a stronger fit and is preferred.

Going on AIC alone, the best model out of the two would be the normal cubic model with a value of 440.88, suggesting this model balances fit and complexity better. However, AIC does not check model validity and we know from the deviance residual graphs shown above that the normal cubic model violates homoscedasticity. The deviance value for the normal cubic model is also extremely high with a value of 37947.70, indicating a worse fit than the poisson cubic model which has a deviance score of 166.78. This suggests the poisson cubic model has a better fit to observed counts.



Heteroscedasticity for the normal cubic model can be checked using this scale-location plot shown above. The red trend line displays an increasing pattern, confirming that heteroscedasticity is present. The higher fitted values have a greater residual spread and therefore the variance of the residuals is not constant. This therefore violates the normality assumption of the GLM, meaning the standard errors and confidence intervals can be deemed unreliable.

```
dispersiontest(model_poisson_cubic)
```

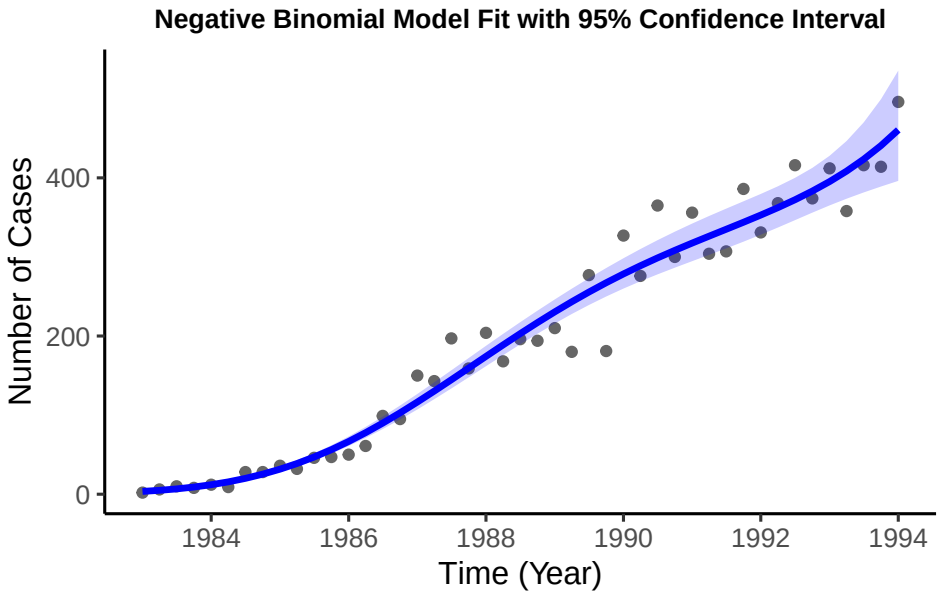
```
##
##  Overdispersion test
##
## data:  model_poisson_cubic
## z = 3.166, p-value = 0.0007727
## alternative hypothesis: true dispersion is greater than 1
## sample estimates:
## dispersion
##   3.671739
```

Overdispersion of the cubic poisson model can be checked using the dispersion test shown above. The p-value for the model is 0.0007 which is $p<0.05$. Therefore, the model is overdispersed and not valid.

Having considered both models, while they both attempt to explain the trend in the number of AIDS cases, they both fail on validity aspects when model testing. While neither model is fully valid, the normal cubic model is preferred, due to its lower AIC and better trend representation. However, an extension of the poisson model to a negative binomial model should be considered to improve count data modelling and provide a better model fit.

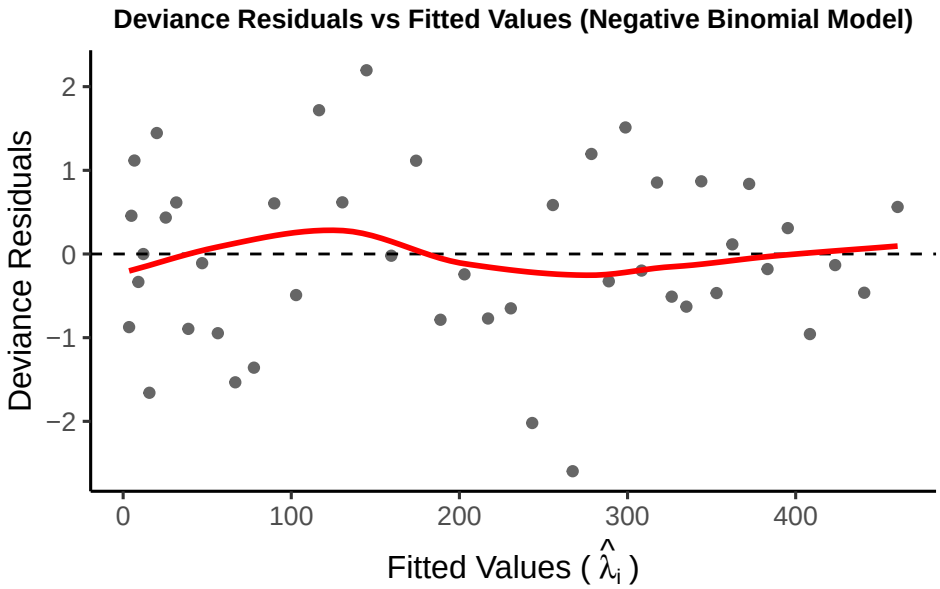
(f) Fitting a Negative Binomial Model

```
model_nb_cubic <- glm.nb(cases ~ numeric_date + I(numeric_date^2) + I(numeric_date^3), data = aids)
```



The Negative Binomial model effectively captures the non-linear increase in cases over time. The model accounts for overdispersion, allowing for variance greater than the mean. This makes the model more flexible than the previous poisson model. The data points mostly fall within the confidence interval, showing that the model predicts the observed cases well and is not overfitting.

```
aids$nb_cubic_resid <- residuals(model_nb_cubic, type = "deviance")
aids$nb_cubic_fitted <- fitted(model_nb_cubic)
```



The deviance residuals vs fitted values plot for the negative binomial model is shows random scatter around zero, demonstrating a good model fit. There is no clear pattern, suggesting the negative binomial model captures the trend well. The red LOESS line stays close to zero across all the fitted values. There is no curved pattern or systematic trend shown. There is also a consistent spread of residuals which indicates homoscedasticity (constant variance), this also reinforces that the model is a good fit.

Comparing AIC's

```
model_comparison <- data.frame(
  Model = c("Poisson Cubic", "Normal Cubic", "Negative Binomial Cubic"),
  Deviance = c(deviance(model_poisson_cubic), deviance(model_normal_cubic), deviance(model_nb_cubic)),
  AIC = c(AIC(model_poisson_cubic), AIC(model_normal_cubic), AIC(model_nb_cubic))
)

kable(model_comparison, caption = "Comparison of Deviance and AIC Across Models")
```

Table 7: Comparison of Deviance and AIC Across Models

Model	Deviance	AIC
Poisson Cubic	166.78195	470.6322
Normal Cubic	37947.69656	440.8830
Negative Binomial Cubic	45.03474	405.9744

Finally, comparing the AIC and deviance scores of the negative binomial model compared to the two extended poisson cubic and normal cubic models, the negative binomial model is preferred. The negative binomial model has both lower deviance and AIC scores. The lower deviance score of 45.03 suggests this model best fits the data. The lower AIC score of 405.97 indicates that this model best balances fit and complexity of the data compared to the other models. The negative binomial model is the best model overall because it meets all GLM assumptions, ensuring a reliable inference while handling overdispersion and providing the best fit for the data with the lowest AIC.