MTHM506-Statistical Data Modelling - Individual Project

James R Lewis

24-02-2025

Contents

Declaration of AI Assistance: I have used OpenAI's ChatGPT tool in creating this report.

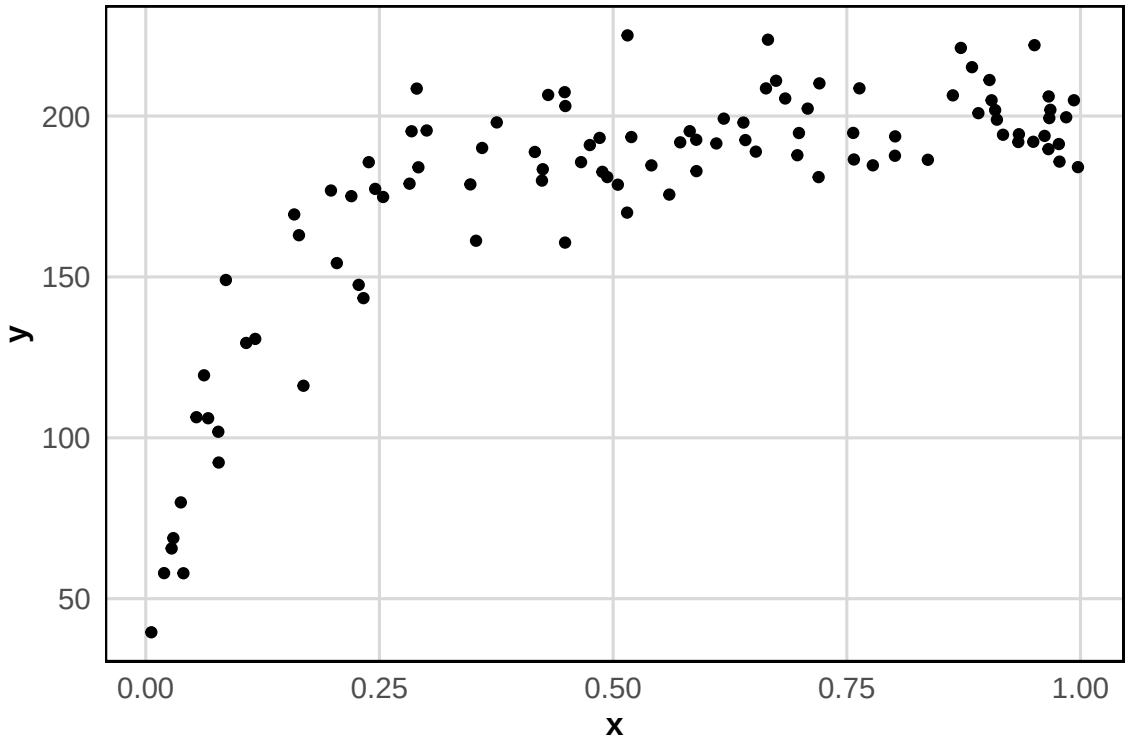
AI-supported/AI-integrated use is permitted in this assessment. I acknowledge the following uses of GenAI tools in this assessment:

1. I have used GenAI tools to check and correct my code.
2. I have used GenAI tools to proofread and correct grammar or spelling errors.
3. I have used GenAI tools to give me feedback on a draft.

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

Question 1

The data frame 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 this data we wish to consider this model:

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

i = 1, 2,..., 100, Y_i independent

a. Exploratory Analysis

From looking at the model and the plot above we can tell that x and y have a non-linear relationship. The mean of this model exhibits a non-linear relationship involving the unknown parameters θ_1 and θ_2. Specifically, the presence of $\theta_2 + x_i$ in the denominator introduces a dependency between x_i and the parameters that cannot be expressed as a linear combination. Consequently, this violates the linearity assumption required for a standard linear regression model, as the relationship between x_i and y_i cannot be transformed into a linear form, as indicated by the scatter plot.

b. Likelihood and Log-Likelihood Functions

The likelihood function L(θ_1,θ_2, σ^2;y,x) is the product of the probability density functions of the normal distribution for each observation y_i: This is central to estimating parameters which best explain the observed data.

Likelihood Function:

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

The log-likelihood $(\theta_1,\theta_2, \sigma^2;y,x)$ Since the likelihood function involves a product over all observations, working with the log-likelihood function, which converts the product into a sum, makes computations more manageable. By maximizing the log-likelihood, we obtain the maximum likelihood estimates, which ensure the best possible fit to the data.

Log-Likelihood Function:

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

c. Log-Likelihood Function in R:

```
mylike <- function(params, y, x) {
  # Defining the Parameters
  theta1 <- params[1]
  theta2 <- params[2]
  sigma <- params[3]
  # Defining the Mean
  mu <- (theta1 * x) / (theta2 +x)
  # Defining n
  n <- length(y)
  result <- -n/2 * log(2 * pi) - n/2 * log(sigma^2) -
    sum((y - mu)^2) / (2 * sigma^2)
  # Here I multiply by a minus, so I work with the negative log-likelihood
  return(-result)
}
```

d. Maximum Likelihood Estimates

Using the mylike() function above, we can now use the non-linear minimisation nlm() function to estimate the parameters. However, nlm() will minimise by default, so we computed the **negative log-likelihood** above to maximise the log-likelihood.

Setting up the nlm function:

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

Good starting values are needed to ensure the model optimisation and efficient convergence, as poor values lead to the slow convergence or local minima. The code below shows how the starting values are computed, and then added into the model.

```r
# max(y) = 225.0743
theta1_init <- max(y)
# median(x) = 0.5302
theta2_init <- median(x)
# sd(y) = 39.45
sigma_init <- sd(y)
# Combine into a vector
inits <- c(theta1_init, theta2_init, sigma_init)

result <- nlm(mylike, p = inits, hessian = T, x = x, y = y,
              iterlim = 10000, steptol = 1e-10)
```

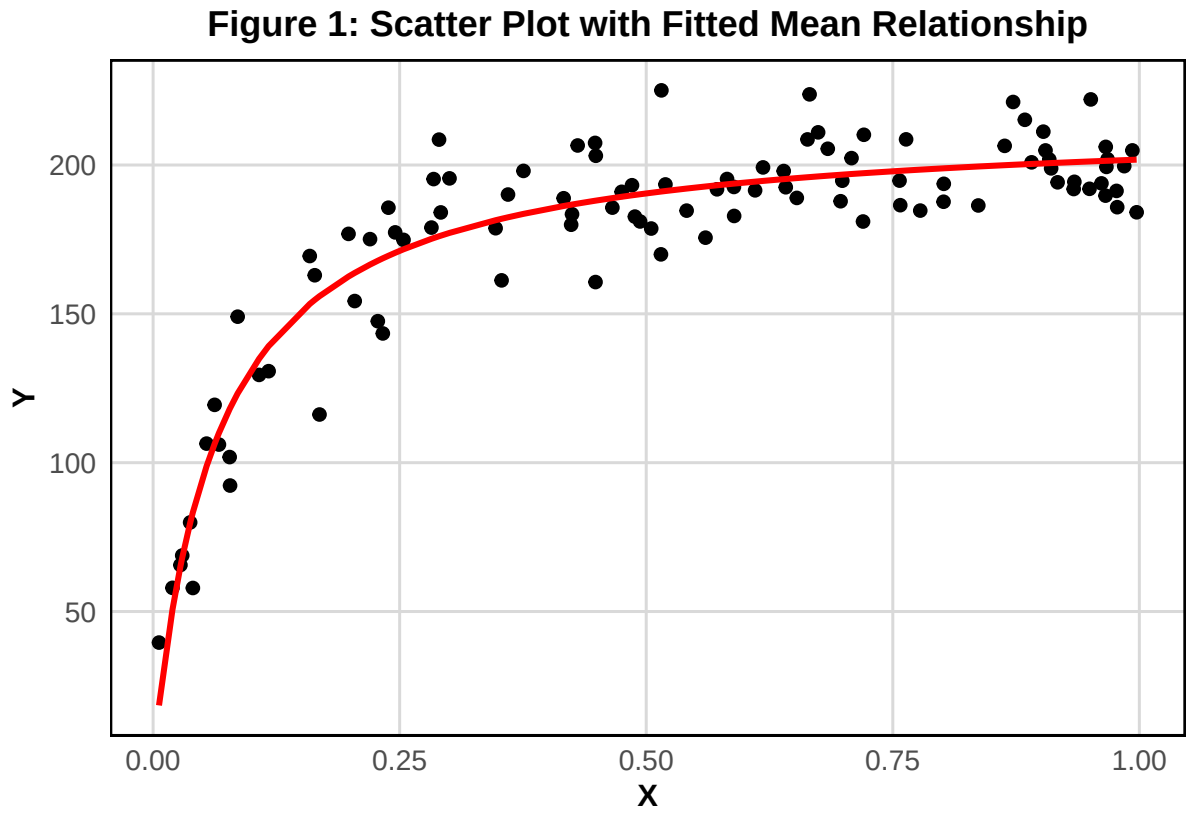
```
##       theta1       theta2        sigma
## 214.65009556   0.06353448  13.61564751
```

Starting values:

- **Theta 1** A reasonable initial estimate for θ_1, is the maximum observed y value, because as seen in the model's mean function, as x increases, the function asymptotically approaches θ_1. Indicating that θ_1 represents the upper limit of y for large x.
- **Theta 2** appears in the denominator and affects how quickly the function increases as x increases. When $x = \theta_2$, the function reaches half of θ_2. Thus a reasonable estimate for θ_2 is the median of x. This way, it represents the center of the distribution whilst, not taking into account extreme values (unlike the mean).
- **Sigma** represents the variance of residuals around the mean function. This assumes that the spread of y around the mean is roughly similar to its overall variance. Thus a logical first approximation is simply the variance of y.

```r
# First extracting MLEs
theta1_mle <- mle_params[1]
theta2_mle <- mle_params[2]
# Calculate the fitted mean
fitted_mu <- (theta1_mle * x)/(theta2_mle + x)
plot_data <- data.frame(x = x, y = y, fitted_mu = fitted_mu)
```

The figure below shows the fitted mean obtained from estimated parameters (in the code above). By plotting this against the observed data, we can see whether it is a good fit.



The fitted mean relationship shows a diminishing returns effect, where y increases rapidly at lower x values before plateauing. The mean curve effectively captures the trend, indicating a good fit for modeling the data. However, variability in the data points increases at higher x values, suggesting potential heteroscedasticity.

e. Standard Errors and 95% Confidence Intervals

Having obtained the maximum likelihood estimates (MLEs) for θ_1 and θ_2, the next step is to assess the uncertainty associated with these estimates. This is achieved by computing standard errors (SEs) and using them to construct 95% confidence intervals (CIs).

The standard errors are derived from the observed information matrix (OIM), which is the inverse of the Hessian matrix. The hessian matrix provides information on how sensitive the likelihood function is to changes in parameter values.

```
result$hessian
```

```
##                [,1]           [,2]           [,3]
## [1,]  3.819026e-01 -1.582326e+02 -0.0006019984
## [2,] -1.582326e+02  1.034061e+05 -0.7616117334
## [3,] -6.019984e-04 -7.616117e-01  1.0782917506
```

```
OIM <- solve(result$hessian) # Observed information matrix.
# Extract standard errors by squareroot of diagonal elements
std_errors <- sqrt(diag(OIM))
names(std_errors) <- c("theta1", "theta2", "sigma")
```

Standard Errors for θ_1, θ_2 and σ^2

```
##      theta1      theta2       sigma
## 2.674798908 0.005140381 0.963024839
```

The standard errors quantify the variability in our estimates — a smaller standard error relative to the size of the parameter indicates higher precision, where as a larger SE indicated greater uncertainty in the estimates.

The estimates all have relatively low standard errors, so we can have a higher level of confidence in the accuracy of the model.

```r
# Constructing 95% CIs
# Extract standard errors for theta1 and theta2
theta1_se <- std_errors["theta1"]
theta2_se <- std_errors["theta2"]
# Compute 95% CIs
theta1_ci <- c(theta1_mle - 1.96 * theta1_se, theta1_mle + 1.96 * theta1_se)
theta2_ci <- c(theta2_mle - 1.96 * theta2_se, theta2_mle + 1.96 * theta2_se)
```

95% Confidence Intervals for θ_1 and θ_2

```
## 95% CI for $ heta_1$: [ 209.4075 , 219.8927 ]
## 95% CI for $ heta_2$: [ 0.05345933 , 0.07360962 ]
```

We can have confidence in this model the parameters sit within a well defined and narrow range, adding reliability to the model.

f. Hypothesis Testing

We test the hypothesis that $\theta_2 = 0.08$ at the 5% significance level. This allows us to assess whether θ_2 is significantly different from a hypothesised value, providing insight into the reliability of our model estimates.

The null and alternative hypotheses are defined as follows:

- **Null Hypothesis (H_0):** $\theta_2 = 0.08$
- **Alternative Hypothesis (H_1):** $\theta_2 \neq 0.08$

Z-statistic Test: measures how many standard deviations our estimate is from the hypothesised value.

The Z-statistic is computed as:

$$Z = \frac{\hat{\theta_2} - 0.08}{\text{SE}(\hat{\theta_2})}$$

Where:

- $\hat{\theta_2}$ is the MLE for θ_2,
- $\text{SE}(\hat{\theta_2})$ is the standard error of $\hat{\theta_2}$.

A two-tailed test is appropriate here because we are testing whether θ_2 is either greater than or less than 0.08, not just in one direction.

The p-value for the two-tailed test is given by:

$$\text{p-value} = 2 \cdot (1 - \Phi(|Z|))$$

Where:

- Φ is the cumulative distribution function (CDF) of the standard normal distribution

```
# Compute the z-statistic
z_stat <- (theta2_mle - 0.08) / theta2_se
# Print the z-statistic
z_stat
```

```
##    theta2
## -3.203172
```

A Z-Statistic test value of -3.20 suggests that θ_2 is far from 0.08, providing evidence against the null hypothesis.

Computing the P-Value

```
# z-stat is negative so use pnorm() which computes P(Z < z)
# multiply by 2 to account for deviations in both directions
p_value <- 2*(pnorm(z_stat, 0, 1))
#Print p-value
p_value
```

```
##     theta2
## 0.00135923
```

The **P-Value** is smaller than 0.05, so we reject the null hypothesis at the 5% significance level. This indicates that θ_2 is significantly different from 0.08, moreover, 0.08 doesn't sit within the 95% confidence intervals computed earlier, which further supports the rejection of the null hypothesis.

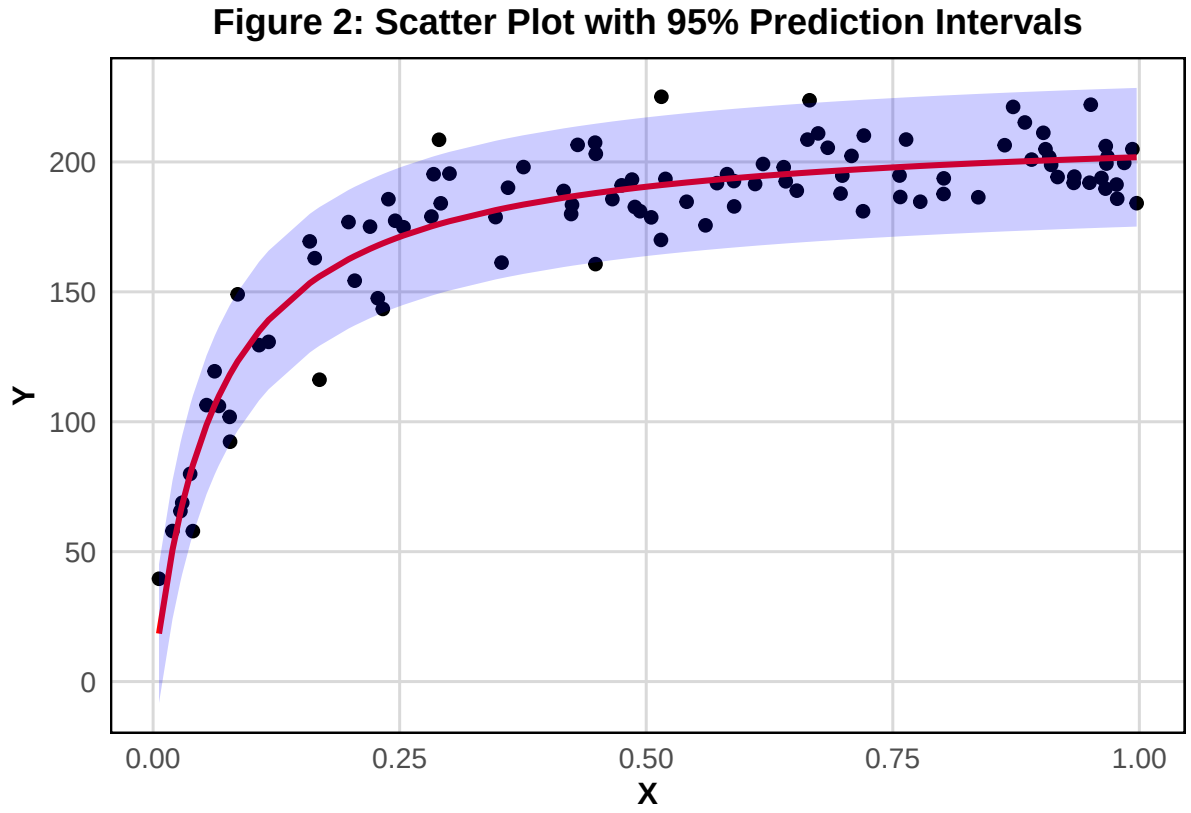
g. Predictive Modelling

In this section, we construct 95% prediction intervals using the plug-in method and visualise them alongside the observed data and fitted values.

Prediction intervals account for both the uncertainty in the mean estimate and the inherent variability in new observations. This makes them wider and more useful when predicting new responses.

```
# Compute 95% prediction intervals
lower_bound <- fitted_mu - 1.96 * mle_params["sigma"]
upper_bound <- fitted_mu + 1.96 * mle_params["sigma"]
```

Plotting the Observed Data, Fitted Mean, and Prediction Intervals



The prediction intervals appropriately capture most data points, with some deviations in the middle-to-upper range of x, indicating possible variance underestimation.

While the model follows the data trend well, slight asymmetry in residual spread suggests potential heteroscedasticity.

Question 2

The data frame aids data 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 (Poisson) is:

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

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

Model 2 (Normal) is:

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

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

a. Exploratory Analysis

To understand the trend in the number of AIDS cases over time, below is a plot of y_i(cases) against x_i (date). This allows us to assess the distribution of cases and determine whether the two proposed models are reasonable.

Note: As you can see in the data below, the date variable isn't in a usable format, so it is transformed.

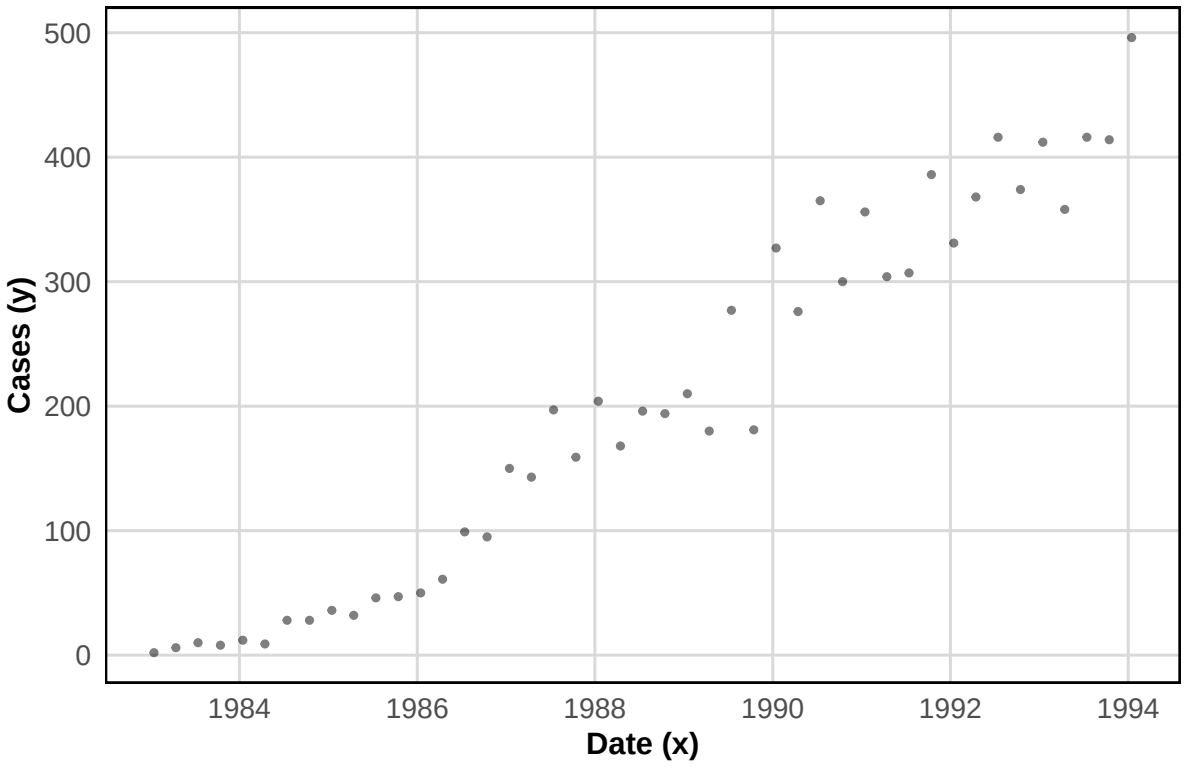
```
head(aids)
```

```
##   cases quarter  date
## 1     2       1 83.00
## 2     6       2 83.25
## 3    10       3 83.50
## 4     8       4 83.75
## 5    12       1 84.00
## 6     9       2 84.25
```

```
# Transform the time variable, from 1983 to 1994
aids$year <- floor(aids$date) + 1900
aids$quarter <- as.numeric(as.character(aids$quarter))
aids$month <- (aids$quarter - 1) * 3 + 1
aids$date_proper <- as.Date(sprintf("%d-%02d-15", aids$year, aids$month))
aids$date_numeric <- scale(as.numeric(aids$date_proper - min(aids$date_proper)))
# Checking the updated data frame
head(aids)
```

```
##   cases quarter  date year month date_proper date_numeric
## 1     2       1 83.00 1983     1  1983-01-15    -1.674450
## 2     6       2 83.25 1983     4  1983-04-15    -1.599409
## 3    10       3 83.50 1983     7  1983-07-15    -1.523535
## 4     8       4 83.75 1983    10  1983-10-15    -1.446826
## 5    12       1 84.00 1984     1  1984-01-15    -1.370118
## 6     9       2 84.25 1984     4  1984-04-15    -1.294243
```

Figure 3: Aids cases over time



The data suggests a non-linear relationship between time and the number of AIDS cases. Both proposed models assume a log-linear relationship between the mean number of cases and time. However, the variance increases over time, as seen in the greater spread of data points after 1988. This suggests potential overdispersion, where variance exceeds the mean.

The **Poisson model** is generally well-suited for count data, as it assumes a log-link function and non-negative integer values. However, a key assumption of the Poisson distribution is that variance equals the mean. Since we observe heteroscedasticity, the Poisson model may underestimate uncertainty in the data. A possible improvement could involve using a Negative Binomial Model, which introduces an extra dispersion parameter to account for overdispersion.

The **Gaussian (Normal) model** assumes normally distributed errors, which is problematic for count data since counts are non-negative and often right-skewed. While the log transformation may help stabilize variance, it does not fully address the discrete nature of the data. This makes the Gaussian model less appropriate for modeling count-based data.

b. Fitting the Models

To assess the fit of each model, we construct the Poisson and Normal models using the generalised linear model (GLM) framework in R. Both models assume a log-link function, modeling the number of cases over time:

- Pmodel1: Poisson Model
- Nmodel1: Normal Model

```
# Model 1:
Pmodel1 <- glm(cases ~ date_proper, data = aids, family = poisson(link = "log"))

# Model 2:
Nmodel1 <- glm(cases ~ date_proper, data = aids,
               family = gaussian(link = "log"))
```

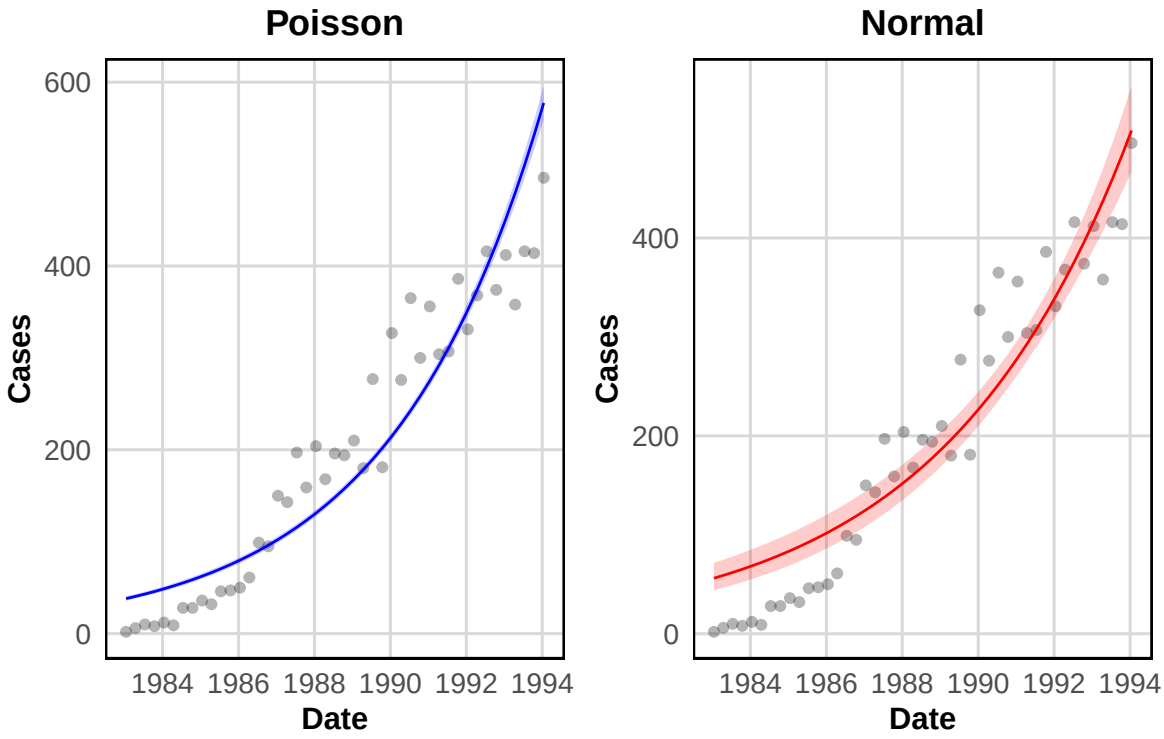
Next, we plot the predicted means against the observed data, including 95% confidence intervals to visualise the range of expected values. Since both models use a log transformation, we apply a back-transformation to obtain meaningful predictions.

```
# Create a sequence of dates for prediction
prediction_data <- data.frame(date_proper = seq(min(aids$date_proper), max(aids$date_proper)

# Model 1 Predictions (Poisson)
pred1 <- predict(Pmodel1, newdata = prediction_data, type = "link",
                 se.fit = TRUE)
prediction_data$fit_model1 <- exp(pred1$fit)  # Back-transform from log
prediction_data$lower_model1 <- exp(pred1$fit - 1.96 * pred1$se.fit)  # 95% CI
prediction_data$upper_model1 <- exp(pred1$fit + 1.96 * pred1$se.fit)

# Model 2 Predictions (Normal)
pred2 <- predict(Nmodel1, newdata = prediction_data, type = "link",
                 se.fit = TRUE)
prediction_data$fit_model2 <- exp(pred2$fit)
prediction_data$lower_model2 <- exp(pred2$fit - 1.96 * pred2$se.fit)
prediction_data$upper_model2 <- exp(pred2$fit + 1.96 * pred2$se.fit)
```

Figure 4: AIDS Cases with Model Predictions




```
## AIC for Poisson Model: 1154.09
## AIC for Normal Model: 482.8203
```

Model Fit Analysis

The **Poisson Model** exhibits a poor fit, with observed data points widely scattered around the predicted line, indicating overdispersion (variance exceeding the mean).

The **Normal Model** shows a marginally better fit, with data more symmetrically distributed around the predicted line. This improved alignment is likely due to the log transformation stabilizing variance.

Model Selection Using AIC

To formally compare model suitability, we use the **Akaike Information Criterion (AIC)**:

- **Poisson Model AIC = 1154.09**
- **Normal Model AIC = 482.82**

A lower AIC suggests a better balance between model fit and complexity. The Normal Model has a substantially lower AIC, indicating it is statistically preferred—provided its assumptions (normality of residuals) hold.

However, if the Poisson model's poor fit is due to overdispersion, a **Negative Binomial Model** could be a more appropriate alternative.

c. Model Diagnostic Tests

We do this to check the deviance residuals against the fitted values for each model. First we extract the residuals and fitted values from the models, using the code below. This informs us on each model's fit and accuracy.

```
# Compute deviance residuals and fitted values for both models
aids$residuals_model1 <- residuals(Pmodel1, type = "deviance")
aids$fitted_model1 <- fitted(Pmodel1)

aids$residuals_model2 <- residuals(Nmodel1, type = "deviance")
aids$fitted_model2 <- fitted(Nmodel1)
```

Figure 5: Deviance Residuals vs Fitted Values for Both Models

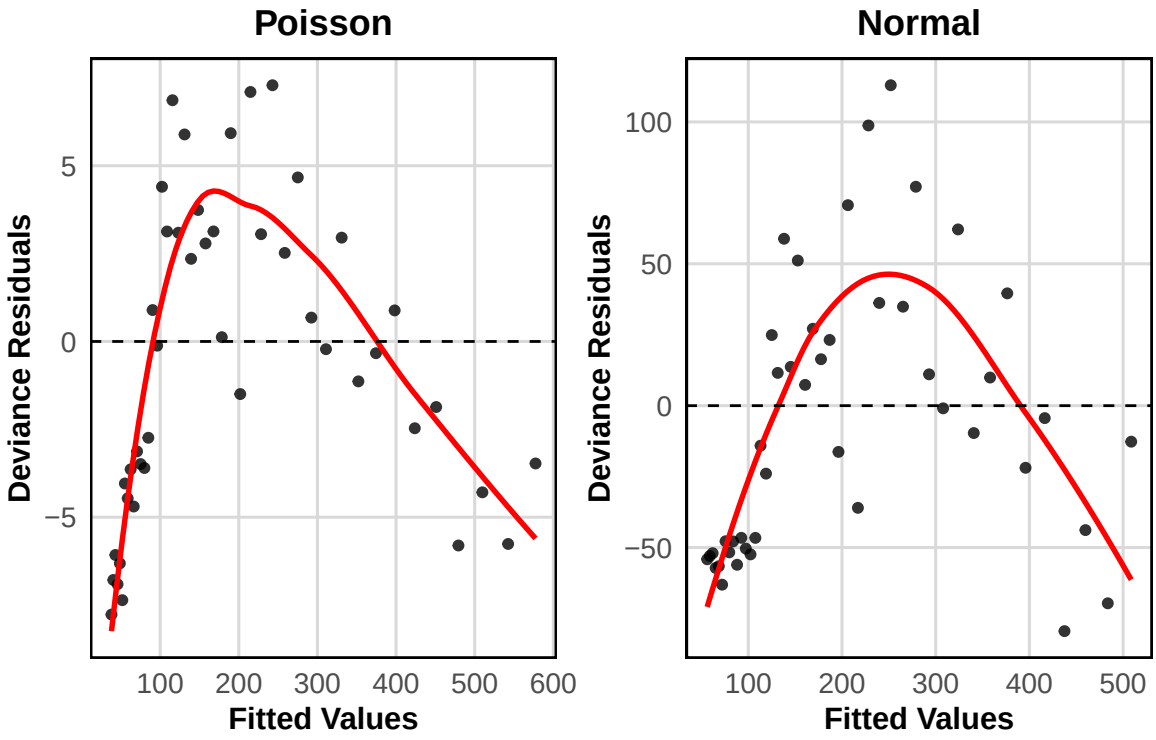


Figure 5 shows deviance residuals against fitted values for both the Poisson and Normal models. Both models shows a lack of fit.

The Poisson Model shows a curved trend and a right skew, moreover, indicates overdispersion given that the residual spread increases with the number of fitted values. To address the heteroscedasticity, adding **polynomial terms** may help capture the relationship more accurately.

The Normal Model's residuals also resemble a normal distribution curve, but the values of the residuals are much larger. As we can see, the log-transformation didn't handle the variance very well. To improve the Normal Model, adding a **polynomial terms** could better capture the non-linear relationships in the data, improving the model's fit.

d. Hypothesis Testing for Model Extensions

To address non-linearity and overdispersion identified earlier, we extend the Poisson and Normal models with polynomial terms. Hypothesis tests determine which extensions significantly improve fit, leading to final model selections.

We extend the models and test improvements:

The extended Poisson models are:

$$\log(\lambda_i) = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 \quad \text{(Quadratic)}$$

$$\log(\lambda_i) = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 \quad \text{(Cubic)}$$

$$\log(\lambda_i) = \beta_0 + \beta_1 x_i + \beta_2 x_i^2 + \beta_3 x_i^3 + \beta_3 4 x_i^4 \quad \text{(Quartic)}$$

The extended Normal models are:

$$log(\mu_i) = \beta_0 + \gamma_1 x_i + \gamma_2 x_i^2 \quad \text{(Quadratic)}$$

$$log(\mu_i) = \gamma_0 + \gamma_1 x_i + \gamma_2 x_i^2 + \gamma_3 x_i^3 \quad \text{(Cubic)}$$

$$log(\mu_i) = \gamma_0 + \gamma_1 x_i + \gamma_2 x_i^2 + \gamma_3 x_i^3 + \gamma_4 x_i^4 \quad \text{(Quartic)}$$

Hypothesis Testing

In order to find the optimal model, we will test each extended model against each other. At the end, we will know how many polynomial terms are needed to fit the data well.

Poisson Hypothesis Test

We use the chi-squared test for Poisson model hypothesis testing because it uses the deviance statistic, which approximates a chi-squared distribution for nested GLMs, providing a good way to assess whether additional terms significantly improve fit.

1. Model with Quadratic Terms vs. Base Poisson Model

To determine if a quadratic degrees significantly improves model fit, we compare the base model to the quadratic model.

Quadratic Term Hypothesis Test:

$$H_0 : \beta_2 = 0 \quad \text{(The quadratic term is not significant)}$$

$$H_1 : \beta_2 \neq 0 \quad \text{(The quadratic term is significant)}$$

```
Pmodel_quad <- glm(cases ~ date_numeric + I(date_numeric^2),
                   data = aids, family = poisson(link = "log"))
```

```r
# Comparing the linear model with the quadratic model
anova(Pmodel1, Pmodel_quad, test = "Chisq")
```

The quadratic model significantly reduced residual deviance from **854.24** (base) to **250.30** (quadratic), with a highly significant p-value ($<$ **2.2e-16**). This indicates a nonlinear relationship between date and cases, which the quadratic term effectively captures. Therefore, we can **reject** the null hypothesis and determine that the quadratic term is significant.

2. Cubic term vs. Quadratic Poisson Model

Similarly, we test whether the cubic term adds significant explanatory power over the quadratic model.

Cubic Term Hypothesis Test:

$$H_0 : \beta_3 = 0 \quad \text{(The cubic term is not significant)}$$

$$H_1 : \beta_3 \neq 0 \quad \text{(The cubic term is significant)}$$

```r
Pmodel_cubic <- glm(cases ~ date_numeric + I(date_numeric^2) + I(date_numeric^3),
                    data = aids, family = poisson(link = "log"))

# Compare the Cubic model with the quadratic model
anova(Pmodel_quad, Pmodel_cubic, test = "Chisq")
```

The output shows a residual deviance drop from **250.30** (quadratic) to **166.78** (cubic). The p-value ($<$ **2.2e-16**) is highly significant, **rejecting** the null hypothesis. This indicates the cubic term significantly enhances the model, capturing a more complex non-linear relationship. Thus, the cubic model is preferred over the quadratic model based on this test.

3. Quartic term vs. Cubic Poisson Model

Next, we test whether the Quartic term adds significant explanatory power over the Cubic model.

Quartic Term Hypothesis Test:

$$H_0 : \beta_4 = 0 \quad \text{(The fourth-degree term is not significant)}$$

$$H_1 : \beta_4 \neq 0 \quad \text{(The fourth-degree term is significant)}$$

```r
Pmodel_4th <- glm(cases ~ date_numeric + I(date_numeric^2) +
                  I(date_numeric^3) + I(date_numeric^4), data = aids,
                  family = poisson(link = "log"))

# Compare the linear model with the quadratic model
anova(Pmodel_cubic, Pmodel_4th, test = "Chisq")
```

The output shows a residual deviance decrease from **166.78** (cubic) to **166.27** (quartic), a small deviance difference of **0.51**. The p-value (**0.4737**) is not significant ($p > 0.05$), **failing to reject** the null hypothesis. Thus, the fourth-degree term **does not add** significant explanatory power over the cubic model, suggesting the cubic model is sufficient.

4. Testing for Overdispersion in the Poisson Model

To check for overdispersion, we calculate the **dispersion statistic**. For a well-fitting Poisson model, the dispersion (Pearson chi-squared statistic divided by the degrees of freedom) should be close to 1.

```
QPmodel <- glm(cases ~ date_numeric, data = aids, family = quasipoisson(link = "log"))
dispersiontest(Pmodel1)
```

```
##
##  Overdispersion test
##
## data:  Pmodel1
## z = 6.744, p-value = 7.705e-12
## alternative hypothesis: true dispersion is greater than 1
## sample estimates:
## dispersion
##   17.30182
```

The overdispersion test for Pmodel1 (base Poisson model) yields a dispersion estimate of **17.302**, with a z-statistic of **6.744** and p-value of **7.705e-12** (< 0.001), strongly rejecting the null hypothesis: dispersion $= 1$. This confirms significant overdispersion, suggesting the Poisson model is inappropriate, and a Quasi-Poisson or negative binomial should be considered.

Normal Hypothesis tests

We use the F-test for Normal model hypothesis testing because it compares the reduction in residual sum of squares between nested models, assuming constant variance and normal errors. This is a good way to assess whether additional terms, like a quadratic term, significantly improve fit.

1. Quadratic vs. Linear Normal Model

To determine if a quadratic term significantly improves the model, we compare the Normal model to the quadratic Normal model.

Quadratic Term Hypothesis Test:

$$H_0 : \beta_2 = 0 \quad \text{(The quadratic term is not significant)}$$

$$H_1 : \beta_2 \neq 0 \quad \text{(The quadratic term is significant)}$$

```
Nmodel_quad <- glm(cases ~ date_numeric + I(date_numeric^2), data = aids,
                   family = gaussian(link = "log"))

# Compare the linear and quadratic Normal models using an F-test
anova(Nmodel1, Nmodel_quad, test = "F")
```

The output shows a significant reduction in residual deviance from **105,323** (base) to **46,331** (quadratic), with an F-statistic of **53.478** and p-value < 0.001. Thus we **reject** the null hypothesis, which indicates the quadratic term significantly improves the model fit.

2. Cubic vs. Quadratic Normal Model

Next, we test whether the cubic term adds significant explanatory power over the quadratic model.

Cubic Term Hypothesis Test (Normal Model):

$$H_0 : \beta_3 = 0 \quad \text{(The cubic term is not significant)}$$

$$H_1 : \beta_3 \neq 0 \quad \text{(The cubic term is significant)}$$

```
# Fit the cubic Normal model (Nmodel_cubic)
Nmodel_cubic <- glm(cases ~ date_numeric + I(date_numeric^2) + I(date_numeric^3),
                    data = aids, family = gaussian(link = "log"))

# Compare the quadratic model with the cubic model
anova(Nmodel_quad, Nmodel_cubic, test = "F")
```

The output shows a residual deviance decrease from **46,311** (quadratic) to **37,948** (cubic), an decrease in the F-statistic to **0.51334** and p-value of **0.0044**. Thus, we can **reject** the null hypothesis. This indicates that the cubic term significantly enhances the model, and is preferred over the quadratic model based on this F-Test.

3. Fourth-Degree vs. Cubic Normal Model

Finally, we test whether the addition of a fourth-degree term further improves the model over the cubic model.

Fourth-Degree Term Hypothesis Test (Normal Model):

$$H_0 : \beta_4 = 0 \quad \text{(The fourth-degree term is not significant)}$$

$$H_1 : \beta_4 \neq 0 \quad \text{(The fourth-degree term is significant)}$$

```
# Fit the fourth-degree Normal model (Nmodel_4th)
Nmodel_4th <- glm(cases ~ date_numeric + I(date_numeric^2) + I(date_numeric^3)
                  + I(date_numeric^4), data = aids,
                  family = gaussian(link = "log"))

# Compare the cubic model with the fourth-degree model
anova(Nmodel_cubic, Nmodel_4th, test = "F")
```

The output shows a residual deviance decrease from **37,948** (cubic) to **37,945** (quartic), an F-statistic of **0.0034** and p-value of **0.954** (not significant). Thus, **failing to reject** the null

hypothesis. This tells us that the fourth-degree term does not add significant explanatory power over the cubic model, suggesting the cubic model is sufficient.

4. Testing for Heteroscedasticity in the Normal Model

While the above tests evaluate the contribution of higher-order terms, it is also important to check the assumption of constant variance (homoscedasticity) in the Normal model. For this purpose, we can conduct a Breusch-Pagan test.

$$H_0 : \text{(There is homoscedasticity)}$$

$$H_1 : \quad \text{(There is hetroscedasticity)}$$

```
bp_test <- bptest(Nmodel_cubic)  # Using the cubic model as it provides the best fit
print(bp_test)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  Nmodel_cubic
## BP = 7.4198, df = 3, p-value = 0.05965
```

The P-value of **0.059** indicates that there is homoscedasticity, and we **fail to reject** the null hypothesis. This finding is surprising as the model looks inaccurate as a result of hetroscedasticity.

e. Hypothesis Test: Results

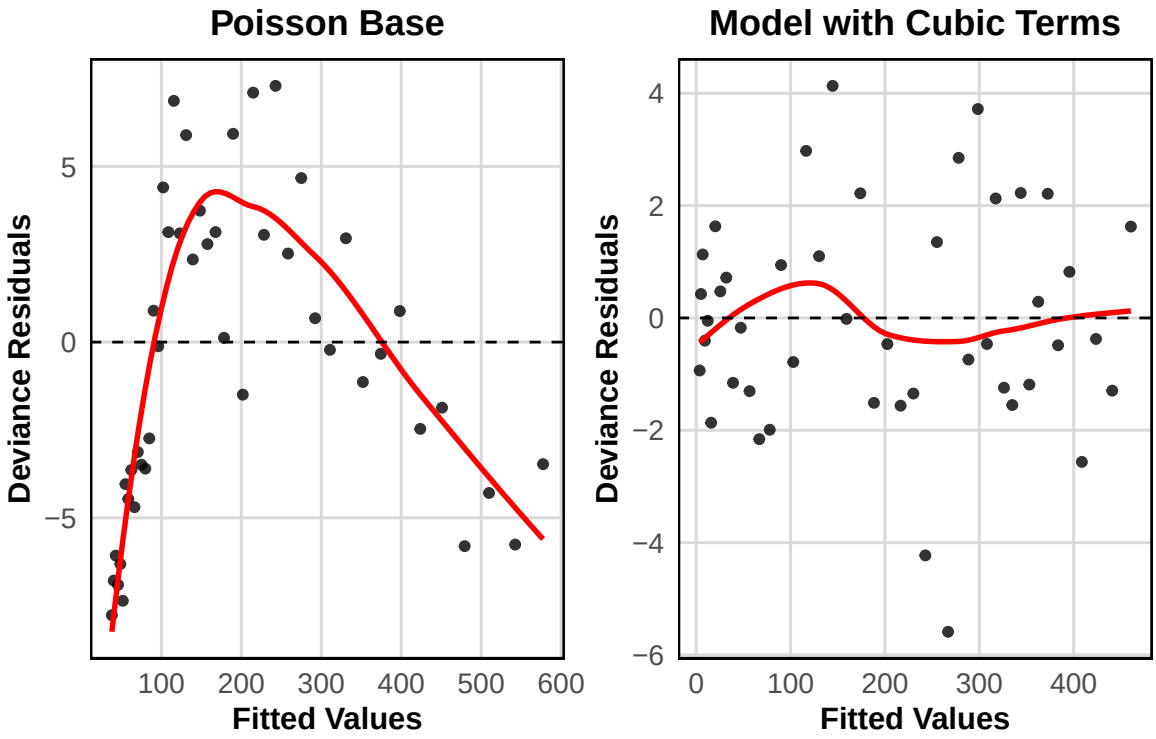
Poisson Extended Models

```
aic_table <- c(AIC(Pmodel1), AIC(Pmodel_quad), AIC(Pmodel_cubic), AIC(Pmodel_4th))
names(aic_table) <- c("Base Model", "Quadratic", "Cubic", "Quartic")
print(aic_table)
```

```
## Base Model  Quadratic      Cubic    Quartic
##  1154.0897   552.1500   470.6322   472.1188
```

The **cubic** Poisson model (Pmodel_cubic) is preferred, with the lowest AIC (**470**) versus Base model (1154), Quadratic (552), and Quartic (472), balancing fit and simplicity.

Figure 6: Deviance Residuals vs Fitted Values: Poisson



The Cubic Model shows residuals scattered more randomly around zero with a slight curvilinear trend, a much better fit than the base model, though some structure remains, hinting at residual overdispersion. **This supports preferring the Cubic Poisson Model** but indicates a Quasi-Poisson or negative binomial approach may still be necessary to address overdispersion fully.

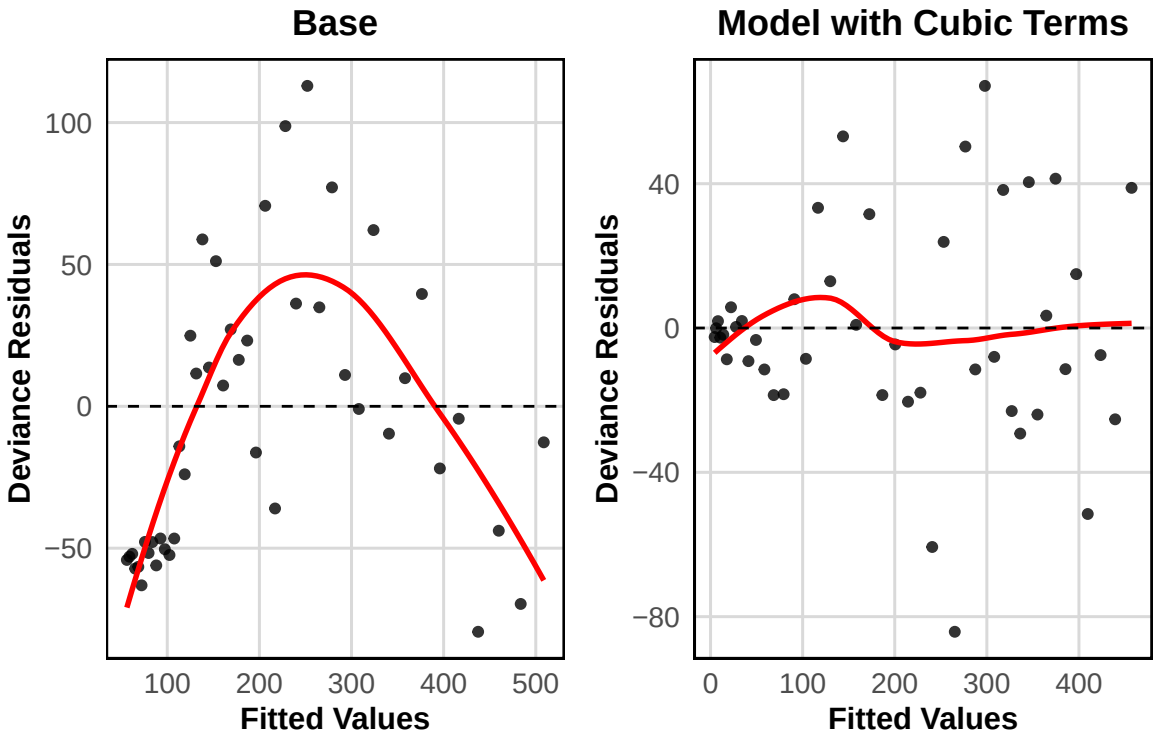
Normal Extended Models

```
aic_table <- c(AIC(Nmodel1), AIC(Nmodel_quad), AIC(Nmodel_cubic), AIC(Nmodel_4th))
names(aic_table) <- c("Base", "Quadratic Terms", "Cubic Terms", "Quartic Terms")
print(aic_table)
```

```
##           Base Quadratic Terms     Cubic Terms   Quartic Terms
##       482.8203        447.8647        440.8830        442.8793
```

The **cubic** Normal model (Nmodel_cubic) is preferred, with the lowest AIC (**440**) versus Base (482), Quadratic (447), and Quartic (442), balancing fit and simplicity.

Figure 7: Deviance Residuals vs Fitted Values: Normal



The Cubic Normal Model exhibits residuals scattered more randomly around zero with a slight curvilinear trend, but shows a significantly better fit. **We can confirm that the Normal model with cubic terms is the best extended normal model**.

e.2. Model Evaluation

Comparing them in figures 6 and 7, Cubic Normal looks slightly better than the Cubic Poisson, as the line is relatively closer to zero, however, the Cubic Normal shows a more equal spread of residuals. That being said, both models appear to still fit fairly poorly.

Normal Cubic has the lower AIC score of 440, compared to the Cubic Poisson of 470, indicating that the normal model is a better fitting the data.

In terms of deviance, both models seems quite weak here: Poisson: residual deviance of 166.78, but strong indications of overdispersion. Normal: residual deviance of 37,948, and we know the assumptions of normality and constant varience might not hold.

The Poisson Cubic, while theoretically suited for counts, is less preferable due to overdispersion, making it not ideal for accurate inference without adjustment. **Thus, the Normal Cubic model is the best overall**, due to its lower AIC (440), better residual distribution, and ability to capture nonlinearity, despite potential violations of normality or variance assumptions.

However, both these models are far from idea. The **Normal Cubic** assumes normality and constant variance, which may not hold given the count nature of cases and potential residual non-normality or heteroscedasticity as seen in fugure 7 and overdispersion in the dataset. The **Poisson Cubic** suffers from significant overdispersion (dispersion = 5.608), violating the

Poisson assumption of equal mean and variance, leading to unreliable standard errors and p-values for inference unless adjusted to Quasi-Poisson, which wasn't directly assessed here for AIC comparison.

f. Final Model: Negative Binomial Cubic Model

To address the significant overdispersion identified in the Poisson Cubic model (dispersion = 5.608, p = 9.634e-05), we extend it to a **Negative Binomial Model**, which incorporates a dispersion parameter (θ) to model variance as $\mathbf{Var}(Y_i) = \mu_i + (1/\theta)\mu_i^2$, effectively capturing overdispersion in the count data (`cases`).

```
NBmodel_cubic <- glm.nb(cases ~ date_numeric + I(date_numeric^2) +
                          I(date_numeric^3), data = aids, link = "log")
```

Model Testing

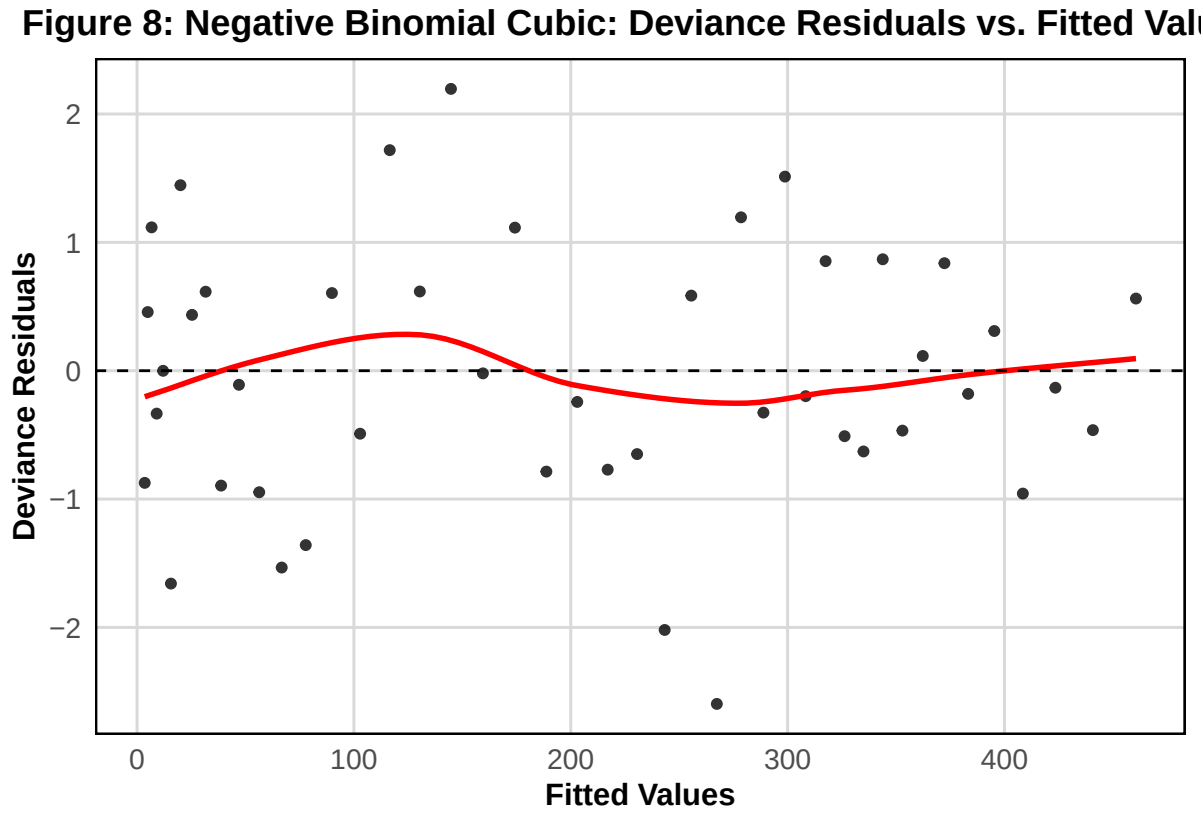


Figure 8 shows that the NB model has a more random scatter around zero with a slight curvilinear trend, indicating a good fit that effectively addresses the overdispersion. Compared to the Poisson Cubic model, which has a broader spread and bigger curvilinear trend as well as the possibility to exhibit non-normaility and hetroscedasticity, this plot supports that the Negative Binomial with Cubic terms' provides significantly better fit.

Model Fit

```
deviance(NBmodel_cubic)
```

```
## [1] 45.03474
```

```
summary(NBmodel_cubic)$theta
```

```
## [1] 64.01135
```

```
1 - pchisq(NBmodel_cubic$deviance,NBmodel_cubic$df.residual)
```

```
## [1] 0.3068304
```

The Negative Binomial's residual deviance is significantly lower than the Poisson, reflecting improved fit by accounting for overdispersion. Moreover, the NB model estimates a dispersion parameter ($= 64$), indicating mild overdispersion compared to the Poisson Cubic model's severe overdispersion.

This is further supported by a chi-squared goodness-of-fit test yields a p-value of 0.306, indicating no significant lack of fit ($p > 0.05$)

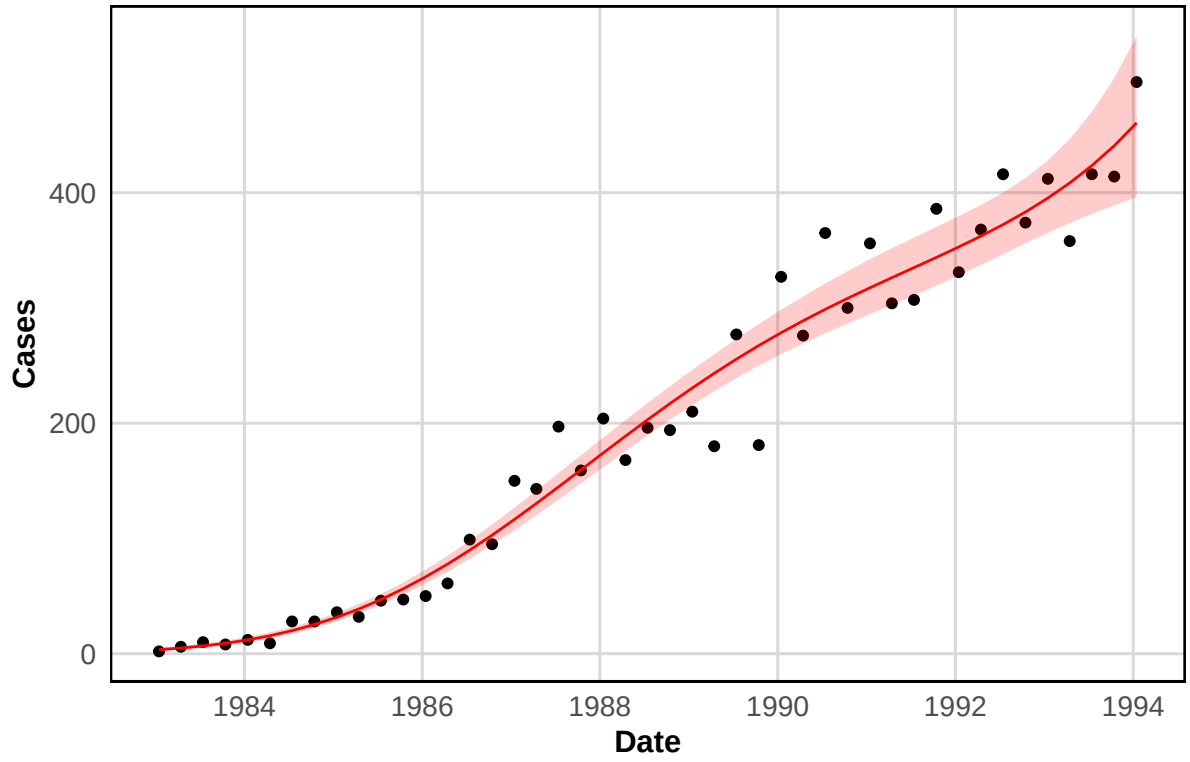
Below, is a plot displaying the model's predictions with 95% confidence intervals.

```
pred3 <- predict(NBmodel_cubic, type = "link",
                 se.fit = TRUE)
prediction_data$fit_model3 <- exp(pred3$fit)
prediction_data$lower_model3 <- exp(pred3$fit - 1.96 * pred3$se.fit)
prediction_data$upper_model3 <- exp(pred3$fit + 1.96 * pred3$se.fit)

aids$fit_model3 <- prediction_data$fit_model3
aids$lower_model3 <- prediction_data$lower_model3
aids$upper_model3 <- prediction_data$upper_model3


ggplot() +
  geom_point(data = aids, aes(x = date_proper, y = cases), alpha = 1,
             color = "black") +
  geom_line(data = aids, aes(x = date_proper, y = fit_model3),
            color = "red") +
  geom_ribbon(data = aids, aes(x = date_proper, ymin = lower_model3,
              ymax = upper_model3), fill = "red", alpha = 0.2) +
  labs(x = "Date", y = "Cases", title = "Figure 9: Negative Binomial with Cubic Terms") +
  custom_theme
```

Figure 9: Negative Binomial with Cubic Terms



In Figure 9, you can see that the **Negative Binomial Cubic model** fits the data well, fairly accurately capturing the non-linear trend and heteroscedasticity, indicated by the 95% confidence interval getting larger over time.

AIC

```
AIC(NBmodel_cubic)
```

```
## [1] 405.9744
```

The **Negative Binomial Cubic model** is preferable to both the Poisson Cubic and Normal Cubic models, as it addresses overdispersion, offers the lowest AIC (405), and exhibits random residuals (Figure 8), making it ideal for both inference and prediction on AIDS cases. It outperforms the Poisson Cubic (AIC = 470.6322, overdispersion issues) and the Normal Cubic (AIC = 440, less suited for counts due to normality assumptions and potential heteroscedasticity).

Limitations

It is important to note, that the AIDS dataset has a very small sample size (45 observations), thus, the fit of these models should be interpreted carefully, given the unreliability a limited amount of data can provide. Following this, higher order polynomial terms increase the risk of overfitting.