Qualifying Exam 2019

Exam #7

6/6/2019

Question 1

a) Number of meals involving fish as a positive test

```
# epiR package for calculating sensitivity and specificity using contingency tables
sensspec0 <- epi.tests(ctable0)
sensspec1 <- epi.tests(ctable1)
sensspec2 <- epi.tests(ctable2)
sensspec3 <- epi.tests(ctable3)
sensspec4 <- epi.tests(ctable4)
sensspec7 <- epi.tests(ctable7)
sensspec14 <- epi.tests(ctable14)
sensspec21 <- epi.tests(ctable21)
```

	Sensitivity	Specificity
>=0	100	0.0
>=1	100	8.0
>=2	100	19.2
>=3	100	28.0
>=4	90	28.8
>=7	70	36.8
>=14	30	89.6
>=21	30	93.6

b) Appropriate thresholds

Sensitivity refers to the true positive rate, or the probability that a test will rule in disease correctly. Specificity indicates the true negative rate, or the probability that a test will correctly rule out disease. Therefore, the probability of a false negative is 100 - sensitivity and the the false positive rate is 100 - specificity.

i. True positives

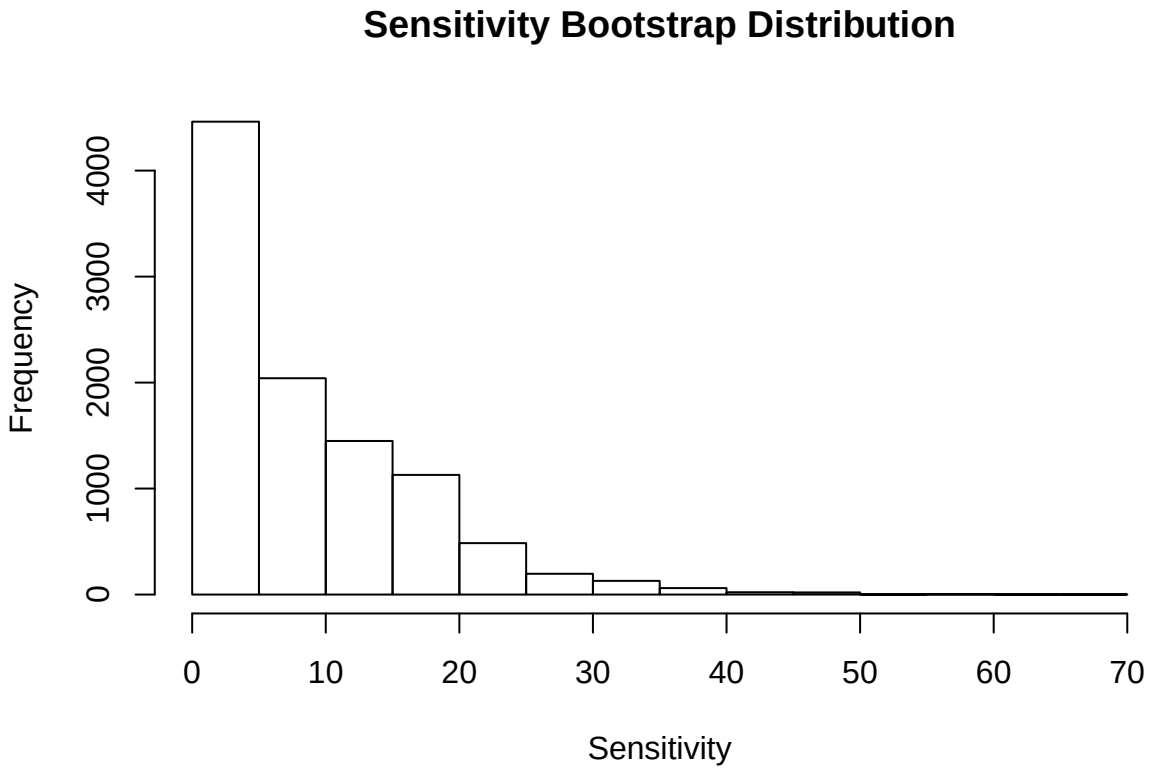
If we want to maximize true positives while minimizing false positives, the optimal threshold is the one with the highest sensitivity and lowest 100 - specificity. A threshold of >= 3 meals per week including fish would provide a 100% true positive rate and a 72% false negative rate.

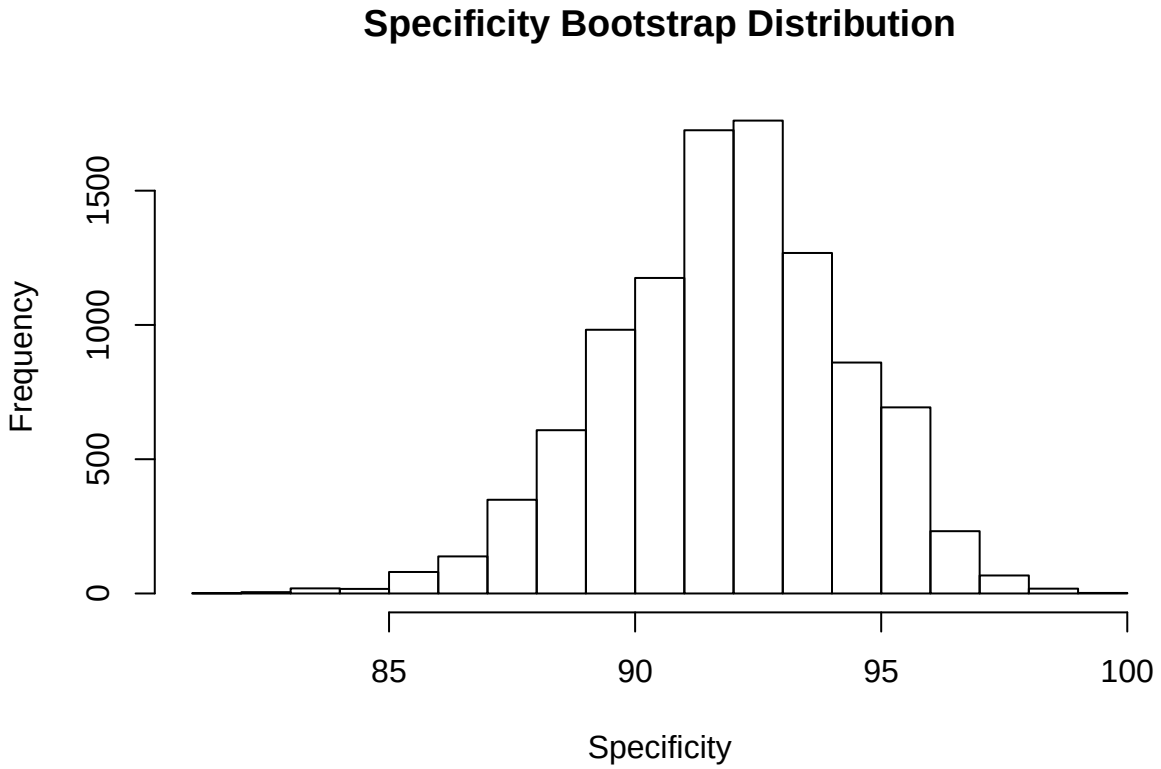
ii. True negatives

Maximizing true negatives first and then true positives requires choosing the test with highest specificity and highest sensitivity. In this case a threshold of >= 21 meals including fish per week would provide a true negative detection rate of 93.6% and a true positive rate of 30%.

c) Bootstrap sampling for >= 21 meals threshold

i. Plots





ii. Mean, SE, and Bias From Bootstrap Distributions

	Mean	Standard Error	Bias
Sensitivity	8.16521	0.0922186	-21.83479
Specificity	91.87575	0.0240551	-1.72425

iii. 90% Bootstrap and Normal Percentile Confidence Intervals

	Normal Percentile	Coverage	Bootstrap CI
Sensitivity	-7%, 23.34%	0%, 6.71%	0%, 25%
Specificity	87.92%, 95.83%	5.58%, 4.9%	87.8%, 95.8%

The bootstrap distribution for sensitivity is not at all normal. The 90% confidence interval for this distribution using normal percentiles is (-7.00%, 23.34%), which does not make sense as sensitivity cannot be negative. Also, none of the bootstrap values were below the lower limit (again, because this is impossible), when we'd expect that 5% would be for a normal distribution. So in this case it would probably be better to use the bootstrap confidence interval (0%, 25%).

The bootstrap distribution for specificity appears to be much closer to normal than for sensitivity. The 90% normal percentile confidence interval is (87.92%, 95.83%), which matches the bootstrap confidence interval very closely (87.80%, 95.80%). Also, approximately 5% percent of the bootstrap values were in each tail, which is what we would expect from a normal distribution.

d. 90% Confidence Intervals Using Exact and Asymptotic Methods

	Clopper-Pearson	Simple Asymptotic
Sensitivity	8.73%, 60.66%	6.16%, 53.84%
Specificity	88.75%, 96.78%	90%, 97.2%

The Clopper-Pearson CI for sensitivity is (8.73%,60.66%). The simple asymptotic CI for sensitivity is (6.16%,53.84%). The Clopper-Pearson CI for specificity is (88.75%, 96.78%). The simple asymptotic CI for specificity is (90.00%, 97.20%).

In general, the normal approximation works best for large sample sizes. Although as a general rule of thumb the Central Limit Theorem applies to sample sizes over 30, the boostrap distribution of sensitivity was not normally distributed so I would use the exact confidence interval in this case.

Confidence intervals are essentially the range for a parameter that is consistent with the data. So based on the exact confidence intervals, if we were to repeat this experiment many times, sensitivity for this test would be between 8.73% and 60.66% in 90% of those experiments.

e. Linear Regression

i. Model Equation

$$\hat{MeHg} = \hat{\beta}_0 + \hat{\beta}_1 X_{\text{fisherman}} + \hat{\beta}_2 X_{\text{fish meals per week}} + \hat{\beta}_3 X_{\text{fish parts=1}} + \hat{\beta}_4 X_{\text{fish parts=2}} + \hat{\beta}_5 X_{\text{fish parts=3}}$$

In the model above, $\hat{\beta}_1$ is the estimate for the effect of being a fisherman on mercury levels. $\hat{\beta}_2$ is the estimated effect of the number of fish meals per week on mercury levels. $\hat{\beta}_3$, $\hat{\beta}_4$, and $\hat{\beta}_5$ are the estimated effect of eating muscle tissue only, muscle tissue and sometimes the whole fish, or the whole fish (respectively). $\hat{\beta}_0$, the intercept, is the average mercury level for someone who is not a fisherman, eats 0 fish meals per week, and does not consume any fish parts.

ii. Results

	Estimate	95% CI	Pr(>|t|)
Intercept	0.904	-0.762, 2.57	0.285
Fisherman = Yes	0.246	-1.221, 1.714	0.740
Fish Meals per Week	0.096	-0.016, 0.209	0.092
Fish Part = Muscle	3.061	0.928, 5.194	0.005
Fish Part = Muscle and Whole	1.676	-0.34, 3.691	0.102
Fish Part = Whole	3.009	0.306, 5.712	0.029

iii. Summary

On average, being a fisherman increases mercury levels by 0.246 (95% CI: -1.221,1.714), but this relationship is not statistically significant (p = 0.740).

f. Fishermen Who Eat 4 Meals of Whole Fish Each Week

i. Average

$$\mathbf{a} = \begin{pmatrix} 1 & 1 & 4 & 0 & 0 & 1 \end{pmatrix}$$

$$\boldsymbol{\beta} = \begin{pmatrix} 0.904 & 0.246 & 0.096 & 3.061 & 1.676 & 3.009 \end{pmatrix}$$

$$\hat{Y} = \mathbf{a}^{\mathrm{T}}\boldsymbol{\beta} = 4.543$$

$$\text{CI} = \hat{Y} \pm t_{\frac{\alpha}{2}} \sqrt{(MSE)\mathbf{a}^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{a}}$$

On average, fishermen who eat 4 meals of whole fish each week will have a mercury level of 4.546 (95% CI: 3.071,6.019).

ii. Individual

$$\hat{Y} = \mathbf{a}^{\mathrm{T}}\boldsymbol{\beta} = 4.543$$

$$\text{CI} = \hat{Y} \pm t_{\frac{\alpha}{2}} \sqrt{(MSE)(1 + \mathbf{a}^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{a})}$$

An individual fisherman who eats 4 meals of whole fish each week will have a mercury level of 4.546 (95% CI: -0.011,9.102).

iii. Prediction Interval vs. Confidence interval

The confidence interval above gives us information about the average mercury level for fishermen who eat 4 meals of whole fish each week in the current sample. However, the prediction interval refers to the mercury level for a theoretical new study participant. So because we are trying to make inference about a broader population, we need to account for some uncertainty in our estimators, which results in a wider interval.

Question 2

a.

$$log(Y_i^*) \sim N(\beta_0 + \beta_1 X_i, \sigma^2)$$
$$Yi = 1 \text{ when } Y_i^* > l = 0.001$$
$$Yi = 1 \text{ when } log(Y_i^*) > log(l)$$
$$P(log(Y_i^*) > log(l)) = 1 - P(log(Y_i^*) \leq log(l))$$
$$= 1 - P(\frac{log(Y_i^*) - \beta_0 - \beta_1 X_i}{\sigma} \leq \frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma}) = 1 - P(Z \leq \frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma})$$
$$= 1 - \Phi(\frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma})$$
$$\text{Set } P(log(Y_i^*) > log(l)) = \theta_i$$
$$Y_i \sim \text{Bernoulli}(\theta_i)$$

In order for this model to work, $\frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma}$ must be between 0 and 1, so $0 \leq log(l) - (\beta_0 + \beta_1 X_i) \leq \sigma$, which means $log(l) \geq (\beta_0 + \beta_1 X_i)$.

Next calculate the log likelihood of Yi:

$$L(Y_i|\theta) = \prod_{i=1}^{n} \theta_i^{Y_i} (1 - \theta_i)^{1 - Y_i}$$
$$logL(Y_i|\theta) = \sum_{i=1}^{n} Y_i log(\theta_i) + (1 - Y_i) log(1 - \theta_i)$$
$$= \sum_{i=1}^{n} Y_i log(1 - \Phi(\frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma})) + (1 - Y_i) log(\Phi(\frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma}))$$

In a Probit model like this, the standard normal can be used in place of a normal with arbitrary mean and SD without loss of generality, so the log-likelihood can be re-written in a more standard matrix form:

$$logL(Y_i|\theta) = \sum_{i=1}^{n} Y_i log(\Phi(X_i^T \beta))) + (1 - Y_i) log(1 - \Phi(X_i^T \beta)))$$

Define the log likelihood function in R and find MLEs using optim(), then calculate asymptotic variance using the Fisher Information Matrix (inverse Hessian):

```
# Contamination as binary variable
pcbs$Yi <- ifelse(pcbs$contam.lev >= 0.001,1,0)
pcbs$Xi <- ifelse(pcbs$location == "Niagara",1,0)
# optim
minus_log_L_fun <- function(params) {
    b0 <- params[1]
    b1 <- params[2]
    log_L <-
      sum(pcbs$Yi*(log(pnorm(b0+b1*pcbs$Xi)))+
            (1-pcbs$Yi)*(log(1-pnorm(b0+b1*pcbs$Xi))))
    return (-log_L)
}
mle <- optim(runif(2), minus_log_L_fun, hessian = TRUE)
mle$par
```

```
## [1] 0.7216058 1.1680902
```

```
I <- mle$hessian
var_Theta_probit <- diag(solve(I))
```

Use glm() to check the coefficient estimates:

```
probit_mod <-
  glm(Yi ~ factor(location), family = binomial(link = "probit"), data = pcbs)
summary(probit_mod)$coefficients
```

```
##                            Estimate Std. Error  z value    Pr(>|z|)
## (Intercept)               0.7215223  0.2365641 3.050007 0.002288358
## factor(location)Niagara 1.1679877  0.4933294 2.367561 0.017905756
```

b.

Because $log(Y_i^*)$ is a standard normal distribution with a location and scale shift, we can rewrite the distribution as $log(Y_i^*) \sim \frac{1}{\sigma}\phi(\frac{log(Y_i^*)-(\beta_0+\beta_1 X_i)}{\sigma})$. Next we define an indicator variable:

$$I(Y_i) = \begin{cases} 0, & \text{if } log(Y_i^*) \leq log(l) \\ 1, & \text{if } log(Y_i^*) > log(l) \end{cases}$$

The resulting likelihood is similar to above, but includes the normal PDF of $log(Y_i^*)$:

$$L(log(Y_i^*)|\theta) = \prod_{i=1}^{n}(\frac{1}{\sigma}\phi(\frac{log(Y_i^*)-(\beta_0+\beta_1 X_i)}{\sigma})^{I(Y_i)}(\Phi(\frac{log(l)-\beta_0-\beta_1 X_i}{\sigma}))^{1-I(Y_i)}$$

$$logL(log(Y_i^*)|\theta) = \sum_{i=1}^{n} I(Y_i)(\frac{1}{\sigma}\phi(\frac{log(Y_i^*)-(\beta_0+\beta_1 X_i)}{\sigma}) + (1-I(Y_i))(\Phi(\frac{log(l)-\beta_0-\beta_1 X_i}{\sigma}))$$

So for observations where $log(Y_i^*) > log(l)$, the contribution to the likelihood function is the PDF of $log(Y_i^*)$. For observations that are below the detection threshold, the contribution to the likelihood is just the probability of being below the threshold as defined above.

Define the log likelihood function in R and find MLEs using optim(), then calculate asymptotic variance using the Fisher Information Matrix (inverse Hessian):

```
# Assign a finite lower censure threshold for those with contam.lev < 0.001
pcbs$log_Yi <- ifelse(log(pcbs$contam.lev) < log(0.001),-7,log(pcbs$contam.lev))
# Tobit function
tobit_fun <- function(par, X, y, limit) {
  sigma = exp(par[length(par)])
  beta = par[1:2]
  indicator = y > limit
  XB = X %*% beta
  log_L = sum(indicator * log((1/sigma)*dnorm((y-XB)/sigma)) ) +
    sum((1-indicator) * log(pnorm((XB-limit)/sigma, lower=F)))
  -log_L
}
initmod = lm(log_Yi ~ factor(location), data=pcbs)
X = model.matrix(initmod)
init = c(coef(initmod), log_sigma=log(summary(initmod)$sigma))
```

```
mle <- optim(par=init, tobit_fun, y=pcbs$log_Yi, X=X, limit=-6.9,hessian = T)
mle$par
```

```
##             (Intercept) factor(location)Niagara               log_sigma
##              -4.4485786               2.1953056               0.4843816
```

```
I <- mle$hessian
var_Theta_tobit <- diag(solve(I))
```

This is a type I Tobit model, which can be checked using the AER package:

```
tobit <- AER::tobit(contam.lev ~ factor(location), left = 0.001,
                    data = pcbs, dist = "lognormal")
summary(tobit)$coefficients
```

```
##
## Test of coefficients:
##
##                            Estimate Std. Error  z value  Pr(>|z|)
## (Intercept)              -4.450746   0.284894 -15.6225 < 2.2e-16 ***
## factor(location)Niagara  2.197204   0.398666   5.5114 3.560e-08 ***
## Log(scale)               0.485800   0.096366   5.0412 4.626e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

c.

Because the Tobit regression is on the log scale, the variance needs to be exponentiated before comparison. Based on this is appears that the Tobit estimators have greater asymptotic variance than the Probit ones.

```
var_Theta_probit[1:2]/exp(var_Theta_tobit)[1:2]
```

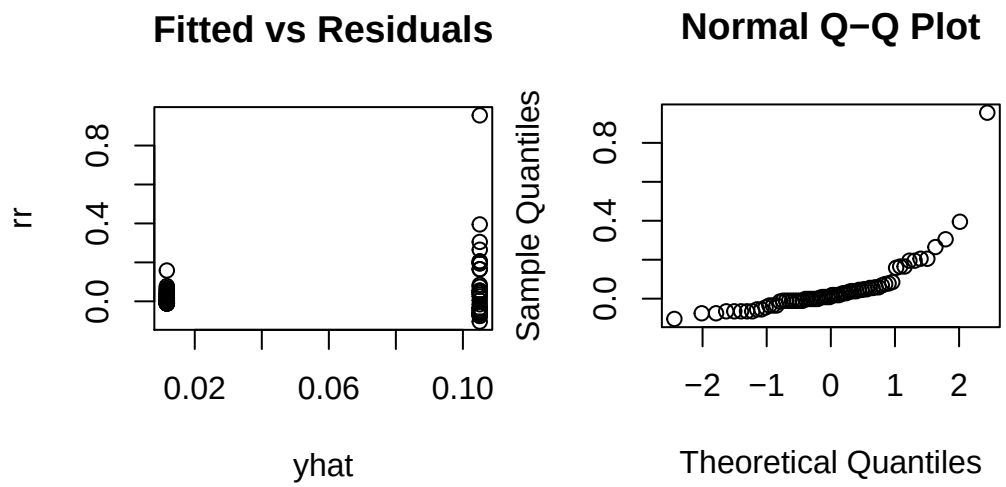
```
##             (Intercept) factor(location)Niagara
##              0.05161321              0.20774276
```

d.

Assuming that log concentration is normally distributed, the estimated mean concentration will be approximately equal to the estimated median. The average log concentration at Fort Erie is -4.451 (95% CI: -5.010, -3.892), and the average log concentration at Niagara is -2.254 (95% CI: -2.801, -1.707). Both of these estimates are slightly lower than the empirical medians from the raw data (-3.912 and -2.003 respectively).

e.

```
plot(fitted(tobit),resid(tobit,type = "response"), main = "Fitted vs Residuals",ylab = "rr",xlab = "yha
qqnorm(resid(tobit,type = "response"))
```



The residuals for this model are clearly not normally distributed, so the fit is not particularly good.

f.

```
# Tobit function with variable sigma
tobit_fun <- function(par, X, y, limit) {
  a = par[3:4]
  AB = X %*% a
  sigma = exp(AB)
  beta = par[1:2]
  indicator = y > limit
  XB = X %*% beta
  log_L = sum(indicator * log((1/sigma)*dnorm((y-XB)/sigma)) ) +
    sum((1-indicator) * log(pnorm((XB-limit)/sigma, lower=F)))
  -log_L
}
initmod = lm(log_Yi ~ factor(location), data=pcbs)
X = model.matrix(initmod)
init = c(coef(initmod), log_sigma=log(summary(initmod)$sigma))

mle <- optim(par=init, tobit_fun, y=pcbs$log_Yi, X=X, limit=-6.9,hessian = T)
mle$par
I <- mle$hessian
var_Theta_tobit_2 <- diag(solve(I))
```

Unfortunately I didn't have time to get this model to work, but the code above provides a rough outline of the approach. Assuming I was able to get reasonable MLEs and asymptotic variances, I would use a Wald test to check whether or not $a_1 = 0$. Under the null hypothesis, the asymptotic distribution of the test statistic is:

$$W = \frac{(\hat{\theta} - \theta_0)^2}{var(\hat{\theta})} = \frac{\hat{a_1}^2}{var(\hat{a_1})} \sim \chi^2_1$$

If this test rejects the null hypothesis that $a_1 = 0$, then it makes sense to use the model where the variance of the outcome also depends on the covariates. Otherwise I would use the simpler model from part b, because $a_1 = 0$ would indicate that the variability of the outcome is independent of the covariates.

g.

Based on the Tobit regression from part b (the model fit wasn't ideal, but it was better than the Probit model and the best I was able to come up with), there is significantly more pollution at Niagara compared to Fort Erie ($Z = 5.511$, $p < 0.0001$).

Question 3

Analysis Plan

Introduction

The data for this analysis are a subset of completers from an early HIV/AIDS clinical trial, in which patients with HIV were treated with monotherapy or combination/dual therapy. Previous studies have shown that lowering HIV viral load can slow progression to AIDS, so it is an important outcome in clinical trials. Unfortunately, the virus develops resistance to mono- and dual-drug therapies.

The aim of this study is to examine the effect over time of mono- and dual-drug therapies on log_{10}(viral load). Participants were treated with only one of the two therapies for 48 weeks, and log_{10}(viral load) was measured at regular intervals during treatment. The primary outcome is the difference between treatment arms at week 48, and a secondary outcome is the difference at week 24.

Hypotheses

Primary null hypothesis: There will be no difference in log_{10}(viral load) between treatment groups at week 48.

Secondary null hypothesis: There will be no difference in log_{10}(viral load) between treatment groups at week 24.

Data Description

Variable descriptions from the data dictionary provided:

- pid: patient identification number; range (9, 6861)
- week: study week (0, 2, 4, 8, 12, 16, 20, 24, 28, 32, 36, 40, 44, 48)
- trt: treatment group (0=mono-therapy arm (AZT or 3TC, N=111), 1=dual arm (AZT+3TC, N=113))
- logvl: log10(viral load); range (2.0, 6.143)

Researchers work with viral load on the log scale so this variable does not need to be tranformed. Participant demographics and CD4 counts unavailable.

Analysis Methods

Because this study is a repeated measures design, log_{10}(viral load) is assumed to be correlated within subjects from week to week. Therefore, all models examined for the purpose of model selection will be linear mixed effect models with a random intercept for participant.

First, models using time ("week") and its higher term polynomials will be examined, because the relationship between log_{10}(viral load) and time is assumed to be non-linear. The highest order term to be included in the final model will be determined based on the Akaike information criterion (AIC) and polynomial orthogonal contrasts. Because higher order polynomial terms increase multicollinearity, the polynomial models will also be assessed by comparison of variance inflation factor (VIF).

In addition to models treating time as a continuous covariate, models treating it as a categorical variable will also be assessed using AIC. Treating time as categorical would significantly reduce the risk of collinearity. This model may also make the most sense in terms of how the data were collected, since the weeks of the study can be considered in terms of categorical "bins" rather than as a truly continuous variable.

Once the most parsimonious model has been selected, differences between the two treatment groups at specific timepoints will be analyzed using linear contrasts. Assumptions of normality and homoscedasticity will be checked with diagnostic plots.

Proposed Tables and Figures

The overall type 3 test of fixed effects and the results of linear contrasts will be reported in tables. Plots of the difference in means at each timepoint (between the groups), and the fitted values of the final model will also be included in the report.

Model selection information, full final model output, and residual plots will be included in the report appendix.

Other Notes

This analysis is potentially biased due to the fact that it is a complete case analysis. For example, it's possible that only participants who experienced good outcomes completed the study. If possible it would be better to include all participants, even those with missing data.

Results

Methods

First, log_{10}(viral load) was modeled with continuous time as the only covariate in order to get a sense of the relationship between time and log_{10}(viral load). Based on polynomial contrasts, it appears that the effect of continuous time is significant up to the 10th degree (i.e. the model would need to include $week$,$week^2$,...,$week^{10}$). Models with high-order terms are very tricky to interpret, and the fact that each term in the model is a function of multiple other terms makes inference difficult (due to multicollinearity, which we try to avoid).

Because of these concerns with polynomial terms, time was modeled as a categorical variable instead. In some ways this makes more sense given the study design, because we don't know the exact length of time between baseline and a given measurement. Each log_{10}(viral load) measure was essentially put in a categorical bin when it was recorded, so trying to model time as continuous doesn't make much sense to begin with (although it's always worth exploring these things). The time as categorical model was also better than all of the continuous time models based on AIC.

Models with and without a random slope for treatment group were also compared, and the model without a random slope had a better AIC. Based on this model selection process, the final model was a linear mixed effect model with a random intercept for participant and time treated as a categorical variable.

Table 1: Type 3 Tests of Fixed Effects

	numDF	denDF	F-value	p-value
Treatment	2	222	2523.71251	<0.001
Week	13	2887	120.30906	<0.001
Treatment:Week	13	2887	18.70325	<0.001

Type 3 tests of fixed effects test whether or not variable is significant overall. The table above indicates that there was a significant effect for treatment and time, and that there was an interaction effect between them.

Table 2: Linear Contrast Results

contrast	week	estimate	SE	df	t.ratio	p.value
0 - 1	0	0.1397249	0.1179708	222	1.184403	0.238
0 - 1	2	0.8739659	0.1179708	222	7.408322	<0.001
0 - 1	4	1.2558023	0.1179708	222	10.645024	<0.001
0 - 1	8	0.9776866	0.1179708	222	8.287529	<0.001
0 - 1	12	0.9498542	0.1179708	222	8.051602	<0.001
0 - 1	16	0.8663353	0.1179708	222	7.343641	<0.001
0 - 1	20	0.8521647	0.1179708	222	7.223521	<0.001
0 - 1	24	0.8449527	0.1179708	222	7.162387	<0.001
0 - 1	28	0.8139703	0.1179708	222	6.899759	<0.001
0 - 1	32	0.7708155	0.1179708	222	6.533950	<0.001
0 - 1	36	0.7263732	0.1179708	222	6.157227	<0.001
0 - 1	40	0.7443974	0.1179708	222	6.310013	<0.001
0 - 1	44	0.7406309	0.1179708	222	6.278086	<0.001
0 - 1	48	0.7450258	0.1179708	222	6.315339	<0.001

Linear contrasts are a method of comparing specific groups based on a model. The table above includes tests of the difference between the mono- and dual-therapy groups at each week. So there was no difference between the groups at baseline, but the difference was significant at each of the following weeks. Here estimate is the difference between the groups, but it is reported as an absolute value and doesn't indicate the direction of the difference. At each timepoint the dual-therapy group had lower log_{10}(viral load) than the mono-therapy group (see figures 1 and 2).

Figure 1: Difference in Means between Treatment Groups

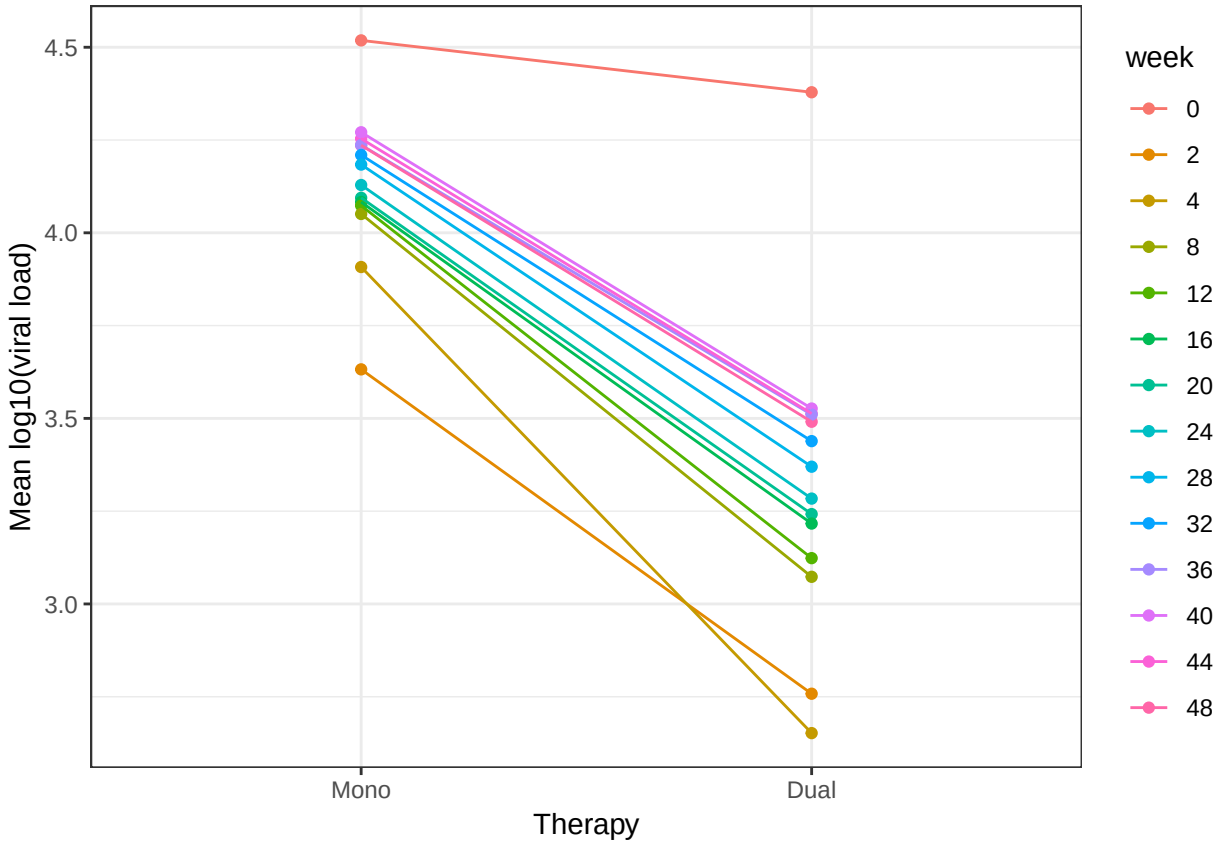


Figure 1 shows the difference in mean log_{10}(viral load) between treatment groups at each week. Clearly the dual-therapy group was always lower, but the difference was smaller at baseline and particarly large at weeks 2 and 4.

Figure 2: Model Results

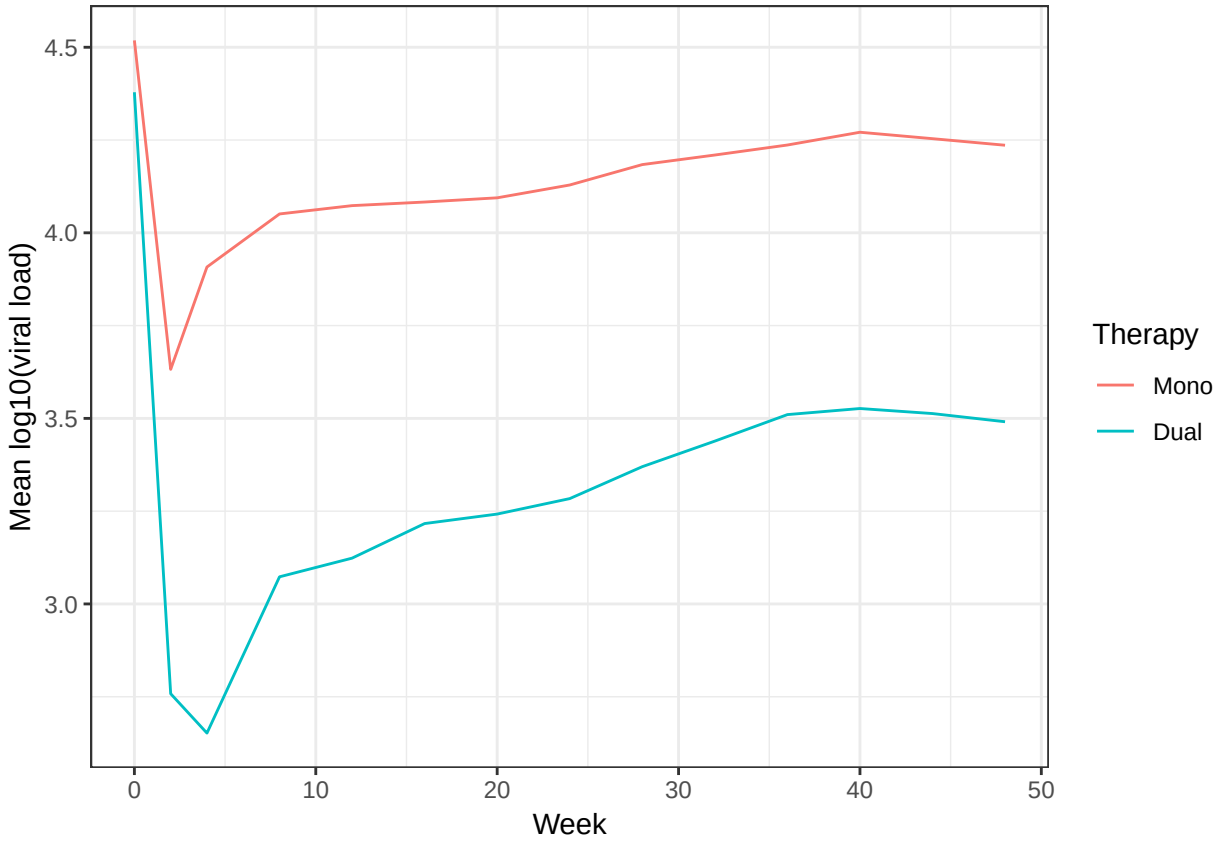


Figure 2 shows the model means at each week, split by treatment group.

Conclusion

At week 24, log_{10}(viral load) was higher in the mono-therapy group compared to dual-therapy by 0.845 (95% CI: 0.6125, 1.077; p $<$ 0.001).

At week 48, log_{10}(viral load) was higher in the mono-therapy group compared to dual-therapy by 0.745 (95% CI: 0.5125, 0.978; p $<$ 0.001).

Appendix

```
# Load libraries
library(tidyverse)
library(knitr)
library(epiR)
library(lme4)
library(nlme)
library(AER)
library(emmeans)
```

```
# Format HIV data
colnames(hiv) <- c("pid","trt","week","logvl")
hiv$trt <- as.factor(hiv$trt)
hiv$pid <- as.factor(hiv$pid)
```

Question 1

Number of meals involving fish as a positive test

```
# Create indicator variables and response
fish$meal0 <- ifelse(fish$fishmlwk >= 0,1,0)
fish$meal1 <- ifelse(fish$fishmlwk >= 1,1,0)
fish$meal2 <- ifelse(fish$fishmlwk >= 2,1,0)
fish$meal3 <- ifelse(fish$fishmlwk >= 3,1,0)
fish$meal4 <- ifelse(fish$fishmlwk >= 4,1,0)
fish$meal7 <- ifelse(fish$fishmlwk >= 7,1,0)
fish$meal14 <- ifelse(fish$fishmlwk >= 14,1,0)
fish$meal21 <- ifelse(fish$fishmlwk >= 21,1,0)
fish$response <- ifelse(fish$MeHg >= 8,1,0)
# Create contingency tables
ctable0 <- table(factor(fish$meal0,levels=1:0),factor(fish$response,levels=1:0))
ctable1 <- table(factor(fish$meal1,levels=1:0),factor(fish$response,levels=1:0))
ctable2 <- table(factor(fish$meal2,levels=1:0),factor(fish$response,levels=1:0))
ctable3 <- table(factor(fish$meal3,levels=1:0),factor(fish$response,levels=1:0))
ctable4 <- table(factor(fish$meal4,levels=1:0),factor(fish$response,levels=1:0))
ctable7 <- table(factor(fish$meal7,levels=1:0),factor(fish$response,levels=1:0))
ctable14 <- table(factor(fish$meal14,levels=1:0),factor(fish$response,levels=1:0))
ctable21 <- table(factor(fish$meal21,levels=1:0),factor(fish$response,levels=1:0))
# Make results table
sens_spec_results <- as.data.frame(matrix(ncol = 2,nrow = 8))
colnames(sens_spec_results) <- c("Sensitivity","Specificity")
rownames(sens_spec_results) <- c(">=0",">=1",">=2",">=3",">=4",">=7",">=14",">=21")
```

```
# Format results
sens_spec_results[">=0","Specificity"] <-
  round(sensspec0$elements$specificity$est*100,1)
sens_spec_results[">=0","Sensitivity"] <-
  round(sensspec0$elements$sensitivity$est*100,1)
sens_spec_results[">=1","Specificity"] <-
  round(sensspec1$elements$specificity$est*100,1)
```

```
sens_spec_results[">=1","Sensitivity"] <-
  round(sensspec1$elements$sensitivity$est*100,1)
sens_spec_results[">=2","Specificity"] <-
  round(sensspec2$elements$specificity$est*100,1)
sens_spec_results[">=2","Sensitivity"] <-
  round(sensspec2$elements$sensitivity$est*100,1)
sens_spec_results[">=3","Specificity"] <-
  round(sensspec3$elements$specificity$est*100,1)
sens_spec_results[">=3","Sensitivity"] <-
  round(sensspec3$elements$sensitivity$est*100,1)
sens_spec_results[">=4","Specificity"] <-
  round(sensspec4$elements$specificity$est*100,1)
sens_spec_results[">=4","Sensitivity"] <-
  round(sensspec4$elements$sensitivity$est*100,1)
sens_spec_results[">=7","Specificity"] <-
  round(sensspec7$elements$specificity$est*100,1)
sens_spec_results[">=7","Sensitivity"] <-
  round(sensspec7$elements$sensitivity$est*100,1)
sens_spec_results[">=14","Specificity"] <-
  round(sensspec14$elements$specificity$est*100,1)
sens_spec_results[">=14","Sensitivity"] <-
  round(sensspec14$elements$sensitivity$est*100,1)
sens_spec_results[">=21","Specificity"] <-
  round(sensspec21$elements$specificity$est*100,1)
sens_spec_results[">=21","Sensitivity"] <-
  round(sensspec21$elements$sensitivity$est*100,1)
```

Bootstrap sampling

```
# Vector for storing results
set.seed(1234)
B <- 10000
sens_results <- numeric(B)
spec_results <- numeric(B)
# Loop
for (i in 1:B) {
  meals <- sample(fish$fishmlwk,replace = T)
  meals <- ifelse(meals >= 21,1,0)
  response <- sample(fish$MeHg,replace = T)
  response <- ifelse(response >= 8,1,0)
  table <- table(factor(meals,levels=1:0),factor(response,levels=1:0))
  sens_results[i] <- (table[1,1]/sum(table[,1])) * 100
  spec_results[i] <- (table[2,2]/sum(table[,2])) * 100
}
```

Mean, SE, and Bias From Bootstrap Distributions

```
boot_results <- as.data.frame(matrix(ncol = 3,nrow = 2))
colnames(boot_results) <- c("Mean","Standard Error","Bias")
rownames(boot_results) <- c("Sensitivity","Specificity")
```

```r
# Sensitivity
boot_results["Sensitivity","Mean"] <- mean(sens_results)
boot_results["Sensitivity","Standard Error"] <-
  sd(sens_results)/sqrt(length(sens_results))
boot_results["Sensitivity","Bias"] <-
  mean(sens_results) - sensspec21$elements$sensitivity$est*100
# Specificity
boot_results["Specificity","Mean"] <- mean(spec_results)
boot_results["Specificity","Standard Error"] <-
  sd(spec_results)/sqrt(length(spec_results))
boot_results["Specificity","Bias"] <-
  mean(spec_results) - sensspec21$elements$specificity$est*100
```

90% Bootstrap and Normal Percentile Confidence Intervals

```r
# Sensitivity
# Normal percentiles
L <- mean(sens_results) - (1.645 * sd(sens_results))
U <- mean(sens_results) + (1.645 * sd(sens_results))
# Specificity
# Normal percentiles
Lc <- mean(spec_results) - (1.645 * sd(spec_results))
Uc <- mean(spec_results) + (1.645 * sd(spec_results))
# Results table
results <- as.data.frame(matrix(ncol = 3,nrow = 2))
rownames(results) <- c("Sensitivity","Specificity")
colnames(results) <- c("Normal Percentile","Coverage","Bootstrap CI")
L <- round(L,2)
U <- round(U,2)
Lc <- round(Lc,2)
Uc <- round(Uc,2)
results["Sensitivity",] <-
  c(paste0(L,"%, ",U,"%"),
    paste0(round(sum(sens_results < L)/B * 100,2),"%, ",
           round(sum(sens_results > U)/B * 100,2),"%"),
    paste0(paste(round(quantile(sens_results,c(0.05,0.95)),2),collapse = "%, "),"%"))
results["Specificity",] <-
  c(paste0(Lc,"%, ",Uc,"%"),
    paste0(round(sum(spec_results < Lc)/B * 100,2),"%, ",
           round(sum(spec_results > Uc)/B * 100,2),"%"),
    paste0(paste(round(quantile(spec_results,c(0.05,0.95)),2),collapse = "%, "),"%"))
```

90% Confidence Intervals Using Exact and Asymptotic Methods

```r
# Sensitivity
# Clopper-Pearson
results <- as.data.frame(matrix(ncol = 2,nrow = 2))
rownames(results) <- c("Sensitivity","Specificity")
colnames(results) <- c("Clopper-Pearson","Simple Asymptotic")
n <- sum(ctable21[,1])
```

```r
x <- ctable21[1,1]
L <- x/(x+((n-x+1)*qf(0.95,(2*(n-x+1)),2*x))) * 100
U <- (x+1)*qf(0.95,(2*(x+1)),2*(n-x))/((n-x)+(x+1)*qf(0.95,(2*(x+1)),2*(n-x))) * 100
results["Sensitivity",1] <- paste0(round(L,2),"%, ",round(U,2),"%")
# Simple asymptotic
n <- sum(ctable21[,1])
phat <- 0.3
L <- (phat - qnorm(0.95)*sqrt((phat*(1-phat))/n))*100
U <- (phat + qnorm(0.95)*sqrt((phat*(1-phat))/n))*100
results["Sensitivity",2] <- paste0(round(L,2),"%, ",round(U,2),"%")
# Specificity
# Clopper-Pearson
n <- sum(ctable21[,2])
x <- ctable21[2,2]
L <- x/(x+((n-x+1)*qf(0.95,(2*(n-x+1)),2*x))) * 100
U <- (x+1)*qf(0.95,(2*(x+1)),2*(n-x))/((n-x)+(x+1)*qf(0.95,(2*(x+1)),2*(n-x))) * 100
results["Specificity",1] <- paste0(round(L,2),"%, ",round(U,2),"%")
# Simple asymptotic
n <- sum(ctable21[,2])
phat <- 0.936
L <- (phat - qnorm(0.95)*sqrt((phat*(1-phat))/n))*100
U <- (phat + qnorm(0.95)*sqrt((phat*(1-phat))/n))*100
results["Specificity",2] <- paste0(round(L,2),"%, ",round(U,2),"%")
```

Linear Regression

```r
lin_mod <- lm(MeHg ~ factor(fisherman)+fishmlwk+factor(fishpart),data = fish)
results <- as.data.frame(round(summary(lin_mod)$coefficients,3))
rownames(results) <- c("Intercept","Fisherman = Yes",
                       "Fish Meals per Week","Fish Part = Muscle",
                       "Fish Part = Muscle and Whole",
                       "Fish Part = Whole")
results <- cbind(results,round(confint(lin_mod),3))
results <- results %>%
  unite("95% CI",`2.5 %`,`97.5 %`,remove = T,sep = ", ") %>%
  select(Estimate,"95% CI","Pr(>|t|)")
```

Confidence/Prediction Intervals

Average

```r
a <- matrix(c(1,1,4,0,0,1))
b <- as.numeric(summary(lin_mod)$coefficients[,1])
yhat <- t(a)%*%b
mse <- mean(lin_mod$residuals^2)
t <- qt(0.1/2,133,lower.tail = F)
x <- model.matrix(lin_mod)
L <- yhat - t*sqrt(mse*(t(a)%*%(solve((t(x)%*%x)))%*%a))
U <- yhat + t*sqrt(mse*(t(a)%*%(solve((t(x)%*%x)))%*%a))
```

Individual

```
a <- matrix(c(1,1,4,0,0,1))
b <- as.numeric(summary(lin_mod)$coefficients[,1])
yhat <- t(a)%*%b
mse <- mean(lin_mod$residuals^2)
t <- qt(0.1/2,133,lower.tail = F)
x <- model.matrix(lin_mod)
L <- yhat - t*sqrt(mse*(1+(t(a)%*%(solve((t(x)%*%x)))%*%a)))
U <- yhat + t*sqrt(mse*(1+(t(a)%*%(solve((t(x)%*%x)))%*%a)))
```

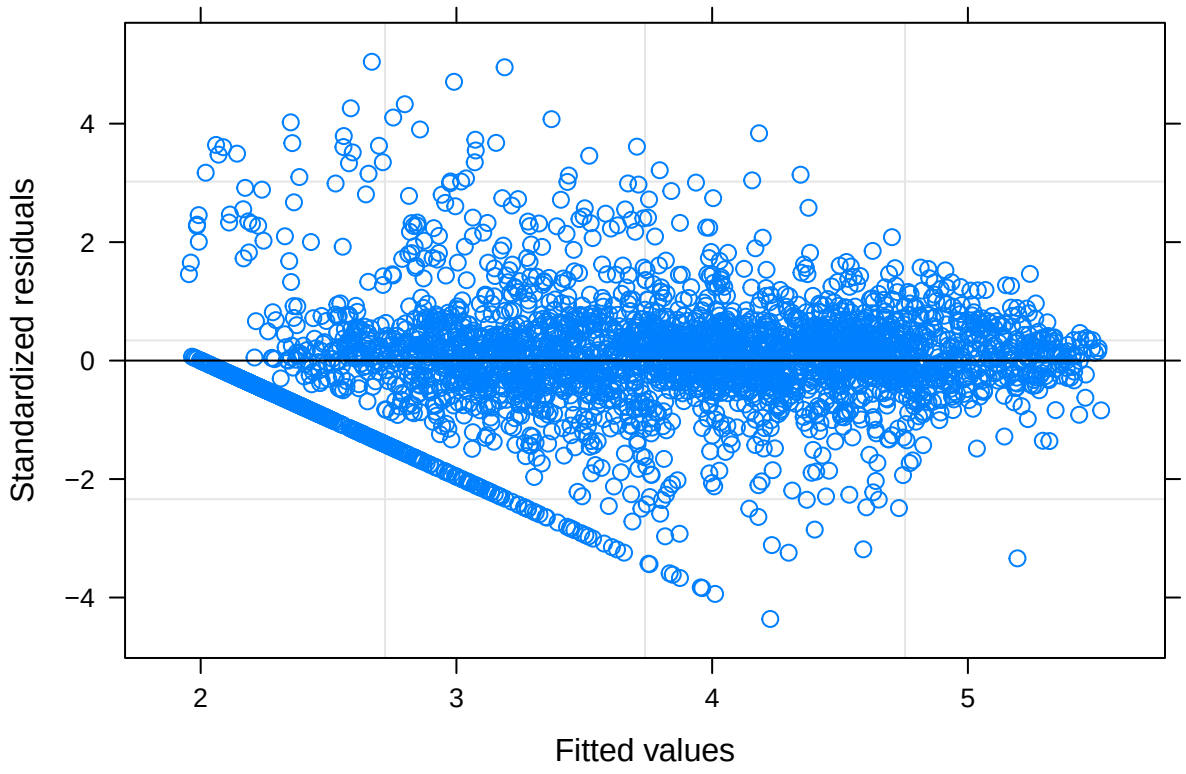
Question 2

All code included in report

Question 3

Model selection

```
# Confirm time is nonlinear
# Linear
time_mod <- lme(logvl ~ week, data=hiv,random = ~1|pid)
plot(time_mod)
```



```r
# Try quadratic
time_mod <- lme(logvl ~ week+I(week^2), data=hiv,random = ~1|pid)
summary(time_mod)$tTable
```

```
##                     Value    Std.Error   DF   t-value      p-value
## (Intercept)  3.6683697247 6.208796e-02 2910 59.083430 0.000000e+00
## week        -0.0059637544 2.172254e-03 2910 -2.745422 6.080448e-03
## I(week^2)    0.0002588596 4.506546e-05 2910  5.744080 1.019784e-08
```

```r
vif(time_mod)
```

```
##      week I(week^2)
##  13.77711  13.77711
```

```r
# Cubic
time_mod <- lme(logvl ~ week+I(week^2)+I(week^3),
                data=hiv,random = ~1|pid,method = "ML")
vif(time_mod)
```

```
##      week I(week^2) I(week^3)
##  77.71362 471.28234 193.02949
```

```r
# Quartic
time_mod <- lme(logvl ~ week+I(week^2)+I(week^3)+I(week^4),
                data=hiv,random = ~1|pid,method = "ML")
vif(time_mod)
```

```
##       week  I(week^2)  I(week^3)  I(week^4)
##   282.8521  5461.6183 11980.7875  2742.8601
```

```r
# Polynomials
time_mod_poly <- lme(logvl ~ poly(week,13),data=hiv,random = ~1|pid,method = "ML")
summary(time_mod_poly)$tTable
```

```
##                        Value Std.Error   DF      t-value      p-value
## (Intercept)       3.72698087 0.0591801 2899  62.97692784 0.000000e+00
## poly(week, 13)1   5.25193280 0.4253238 2899  12.34808083 3.551580e-34
## poly(week, 13)2   2.91701000 0.4253238 2899   6.85832752 8.481891e-12
## poly(week, 13)3  -6.23977611 0.4253238 2899 -14.67064844 4.638758e-47
## poly(week, 13)4   5.93491681 0.4253238 2899  13.95387855 7.040133e-43
## poly(week, 13)5  -7.30229073 0.4253238 2899 -17.16877948 5.345524e-63
## poly(week, 13)6   7.12464569 0.4253238 2899  16.75110938 3.438117e-60
## poly(week, 13)7  -5.21550763 0.4253238 2899 -12.26243979 9.732215e-34
## poly(week, 13)8   3.59486158 0.4253238 2899   8.45205814 4.466692e-17
## poly(week, 13)9  -2.27509549 0.4253238 2899  -5.34909034 9.528060e-08
## poly(week, 13)10  1.47107387 0.4253238 2899   3.45871506 5.505846e-04
## poly(week, 13)11 -0.56196521 0.4253238 2899  -1.32126439 1.865175e-01
## poly(week, 13)12 -0.08836896 0.4253238 2899  -0.20776867 8.354242e-01
## poly(week, 13)13  0.01432468 0.4253238 2899   0.03367947 9.731351e-01
```

```r
# Compare categorical time to continuous time with quadratic
time_mod <- lme(logvl ~ trt*(week+I(week^2)), data=hiv,random = ~1|pid,method = "ML")
time_mod_cat <- lme(logvl ~ trt*factor(week), data=hiv,random = ~1|pid,method = "ML")
anova(time_mod,time_mod_cat) # AIC much better for categorical time, and makes sense with design
```

```
##              Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## time_mod         1  8 5433.230 5481.636 -2708.615
## time_mod_cat     2 30 4215.428 4396.949 -2077.714 1 vs 2 1261.802  <.0001
```

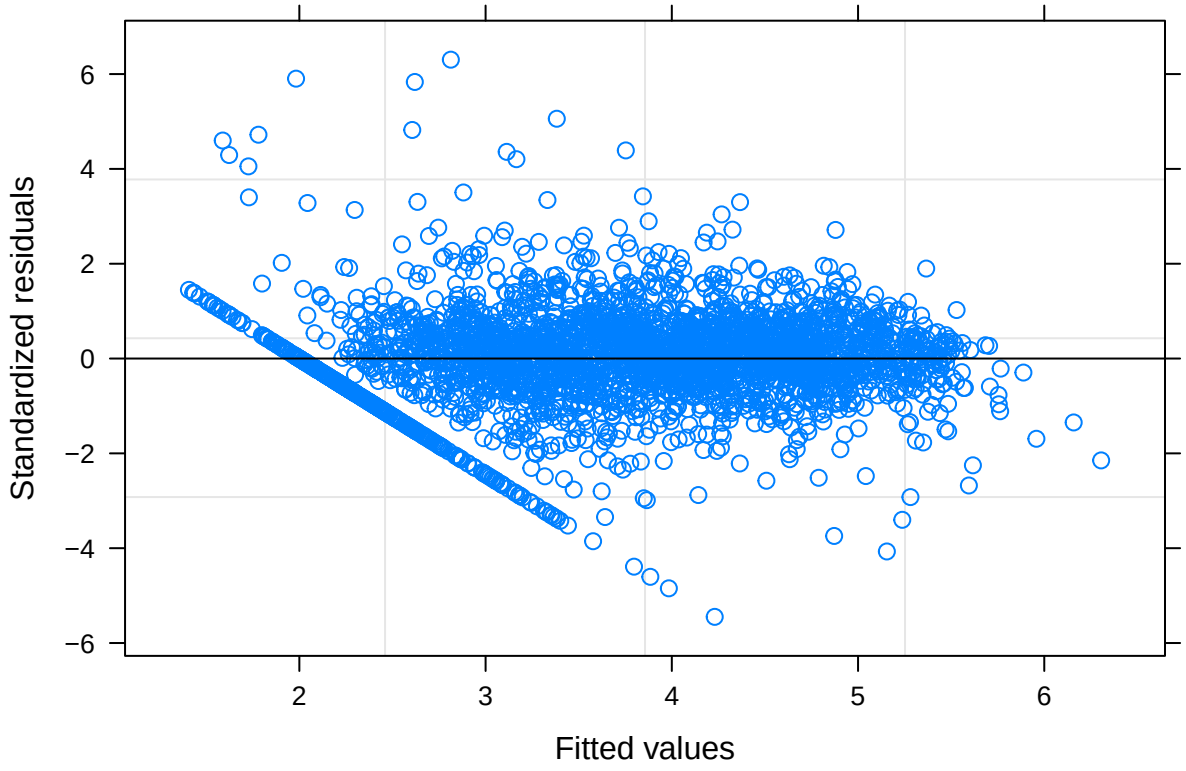
```r
# Compare with random slope for treatment
time_mod_cat_slope <- lme(logvl ~ trt*factor(week), data=hiv,random = ~1+trt|pid,method = "ML")
anova(time_mod_cat,time_mod_cat_slope) # slightly better without random slope
```

```
##                    Model df      AIC      BIC    logLik   Test   L.Ratio
## time_mod_cat           1 30 4215.428 4396.949 -2077.714
## time_mod_cat_slope     2 32 4219.212 4412.834 -2077.606 1 vs 2 0.2160245
##                    p-value
## time_mod_cat
## time_mod_cat_slope  0.8976
```

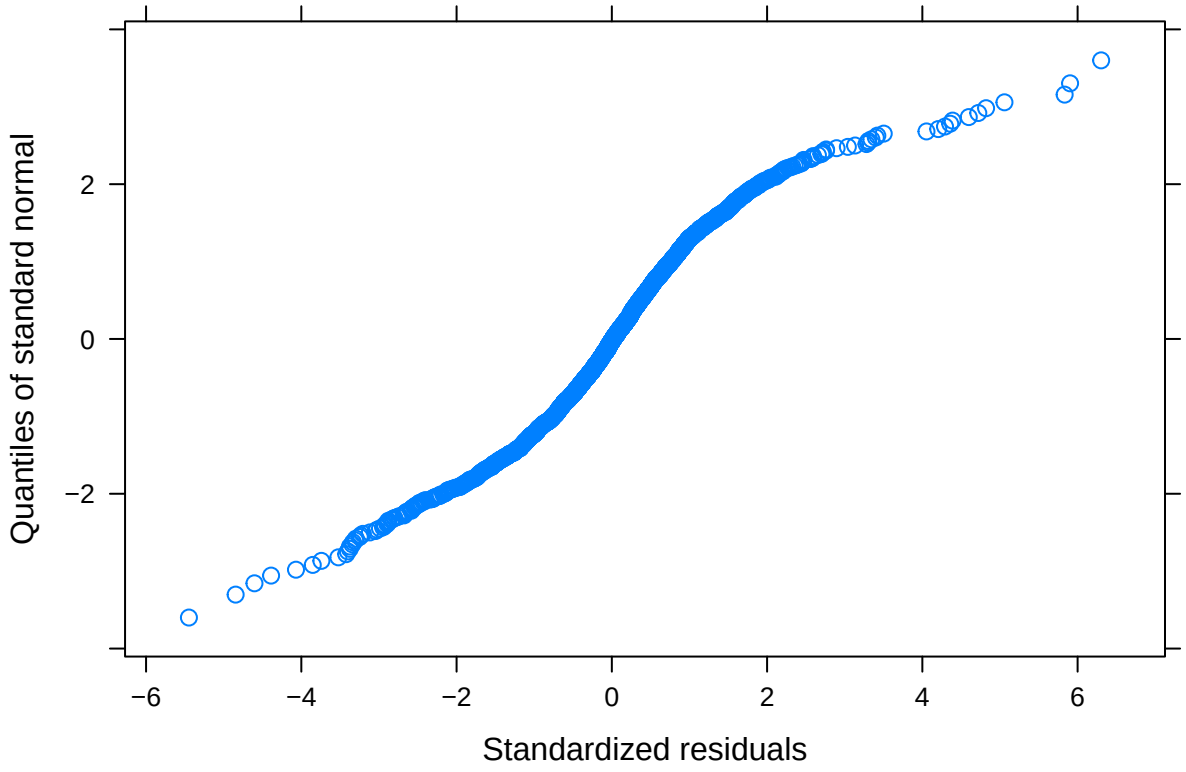
```r
# Full output for final model
time_mod_cat <- lme(logvl ~ trt*factor(week), data=hiv,random = ~1|pid)
summary(time_mod_cat)$tTable
```

```
##                          Value  Std.Error   DF    t-value      p-value
## (Intercept)          4.5184595 0.08378954 2886  53.926292 0.000000e+00
## trt1                -0.1397249 0.11797082  222  -1.184403 2.375202e-01
## factor(week)2       -0.8863874 0.05495215 2886 -16.130169 4.138115e-56
## factor(week)4       -0.6106306 0.05495215 2886 -11.112043 4.042958e-28
## factor(week)8       -0.4676667 0.05495215 2886  -8.510435 2.742902e-17
## factor(week)12      -0.4451982 0.05495215 2886  -8.101562 7.924443e-16
## factor(week)16      -0.4355135 0.05495215 2886  -7.925323 3.221280e-15
## factor(week)20      -0.4241532 0.05495215 2886  -7.718591 1.609218e-14
## factor(week)24      -0.3896306 0.05495215 2886  -7.090362 1.675100e-12
## factor(week)28      -0.3346396 0.05495215 2886  -6.089655 1.281471e-09
## factor(week)32      -0.3088829 0.05495215 2886  -5.620943 2.080574e-08
## factor(week)36      -0.2818739 0.05495215 2886  -5.129442 3.098323e-07
## factor(week)40      -0.2475045 0.05495215 2886  -4.504001 6.933276e-06
## factor(week)44      -0.2648108 0.05495215 2886  -4.818935 1.517884e-06
## factor(week)48      -0.2825045 0.05495215 2886  -5.140918 2.916525e-07
## trt1:factor(week)2  -0.7342409 0.07736944 2886  -9.490064 4.666467e-21
## trt1:factor(week)4  -1.1160773 0.07736944 2886 -14.425300 1.332131e-45
## trt1:factor(week)8  -0.8379617 0.07736944 2886 -10.830654 8.022056e-27
## trt1:factor(week)12 -0.8101292 0.07736944 2886 -10.470920 3.305082e-25
## trt1:factor(week)16 -0.7266104 0.07736944 2886  -9.391439 1.162913e-20
## trt1:factor(week)20 -0.7124398 0.07736944 2886  -9.208284 6.192708e-20
## trt1:factor(week)24 -0.7052278 0.07736944 2886  -9.115069 1.433780e-19
## trt1:factor(week)28 -0.6742453 0.07736944 2886  -8.714621 4.828819e-18
## trt1:factor(week)32 -0.6310906 0.07736944 2886  -8.156846 5.074057e-16
## trt1:factor(week)36 -0.5866483 0.07736944 2886  -7.582429 4.543219e-14
## trt1:factor(week)40 -0.6046725 0.07736944 2886  -7.815392 7.614484e-15
## trt1:factor(week)44 -0.6009060 0.07736944 2886  -7.766710 1.110561e-14
## trt1:factor(week)48 -0.6053008 0.07736944 2886  -7.823513 7.148355e-15
```

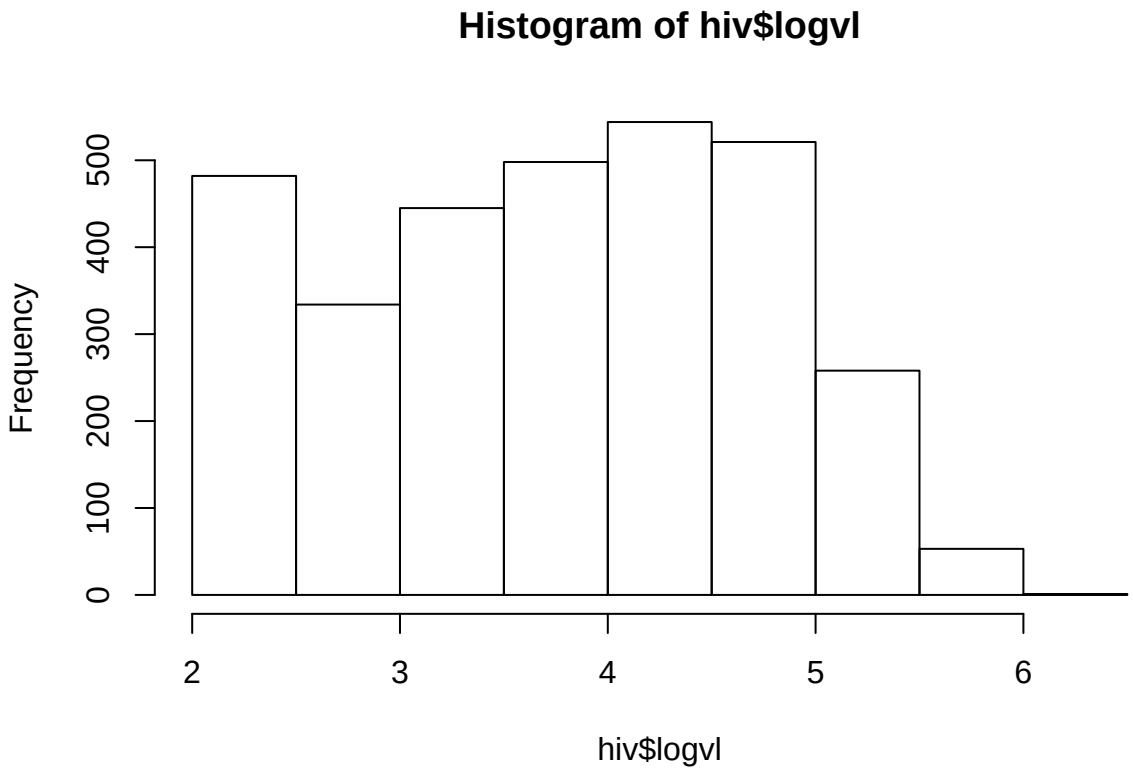
```r
# Residuals
plot(time_mod_cat)
```



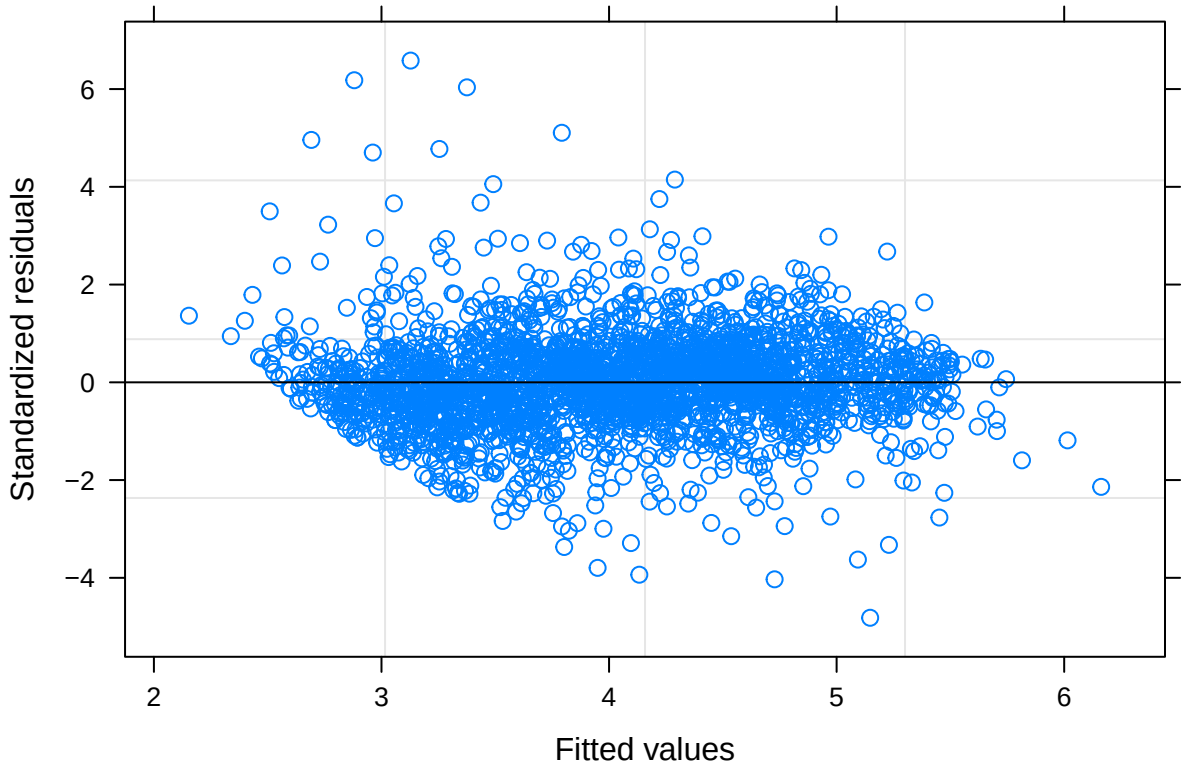

```
qqnorm(time_mod_cat)
```



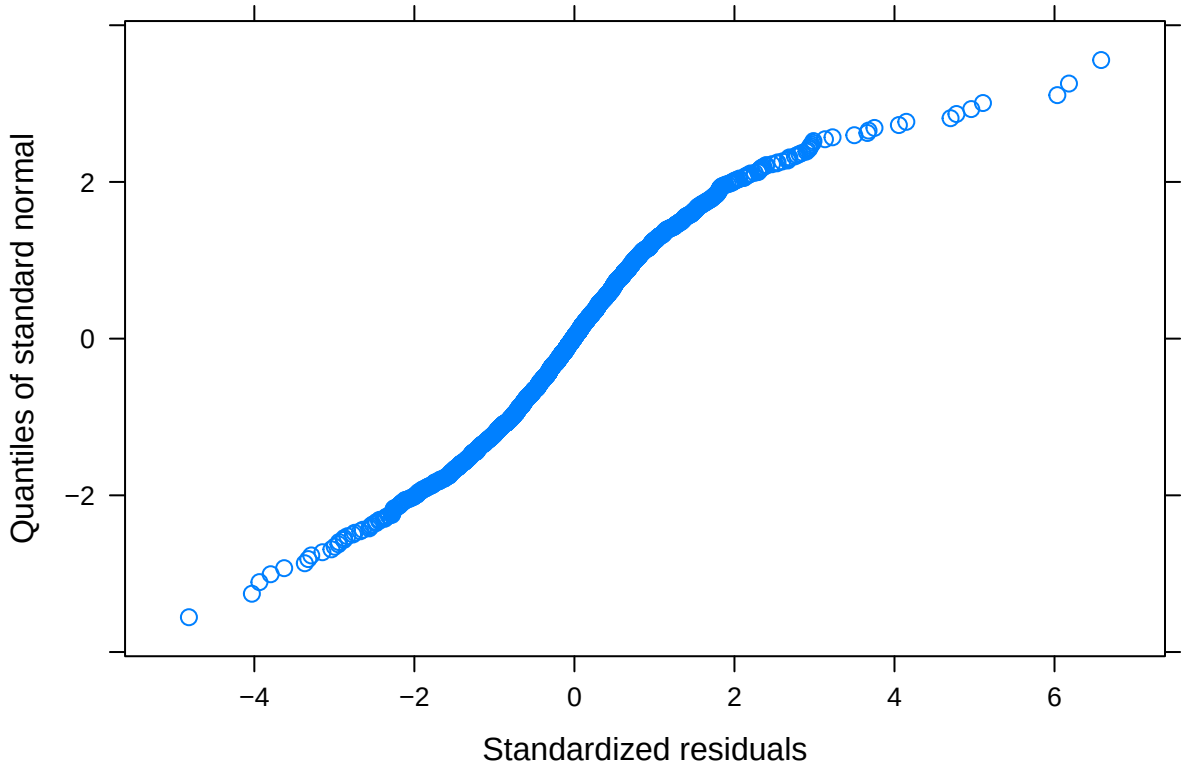

```
# Odd residuals most likely due to higher frequency of values between 2 and 2.5
hist(hiv$logvl)
```




```
# Check without values < 2.5
time_mod_cat_check <- lme(logvl ~ trt*factor(week), data=hiv[hiv$logvl>2.5,],random = ~1|pid)
plot(time_mod_cat_check) # Still not ideal, but a little better
```



```
qqnorm(time_mod_cat_check)
```




```
# Although we're assuming that logvl is normally distributed at each week,
# this doesn't appear to be the case.
```

Final model and test of overall effect

```
time_mod_cat <- lme(logvl ~ trt*factor(week), data=hiv,random = ~1|pid)
time_mod_cat_noint <- lme(logvl ~ trt*factor(week)-1, data=hiv,random = ~1|pid)
type_3_results <- as.data.frame(anova(time_mod_cat_noint))
rownames(type_3_results) <- c("Treatment","Week","Treatment:Week")
type_3_results$`p-value` <- format.pval(type_3_results$`p-value`,
                                        digits = 3,eps=0.001)
```

Linear Contrasts

```
t <- emmeans(time_mod_cat,~ trt*factor(week))
t <- as.data.frame(pairs(t,by = "week"))
t$p.value <- format.pval(t$p.value,digits = 3,eps=0.001)
```

Figure 1

```
em_plot <- emmip(time_mod_cat, week~trt) +
  xlab("Therapy") +
```

```
  scale_x_discrete(labels=c("Mono","Dual")) +
  ylab("Mean log10(viral load)") +
  theme_bw()
```

Figure 2

```
plot_data <- as.data.frame(emmeans(time_mod_cat,~ trt*factor(week)))
mod_plot <- ggplot(plot_data,aes(x=week,y=emmean, group=trt)) +
  geom_line(aes(color=trt)) +
  xlab("Week") +
  ylab("Mean log10(viral load)") +
  scale_color_discrete(name = "Therapy", labels = c("Mono","Dual"))+
  theme_bw()
```

Confidence intervals

```
confint(pairs(emmeans(time_mod_cat,~ trt*factor(week)),by="week"))
```

```
## week =  0:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.140 0.118 222  -0.0928    0.372
##
## week =  2:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.874 0.118 222   0.6415    1.106
##
## week =  4:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       1.256 0.118 222   1.0233    1.488
##
## week =  8:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.978 0.118 222   0.7452    1.210
##
## week = 12:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.950 0.118 222   0.7174    1.182
##
## week = 16:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.866 0.118 222   0.6338    1.099
##
## week = 20:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.852 0.118 222   0.6197    1.085
##
## week = 24:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.845 0.118 222   0.6125    1.077
##
## week = 28:
```

```
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.814 0.118 222   0.5815    1.046
##
## week = 32:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.771 0.118 222   0.5383    1.003
##
## week = 36:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.726 0.118 222   0.4939    0.959
##
## week = 40:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.744 0.118 222   0.5119    0.977
##
## week = 44:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.741 0.118 222   0.5081    0.973
##
## week = 48:
##  contrast estimate    SE  df lower.CL upper.CL
##  0 - 1       0.745 0.118 222   0.5125    0.978
##
## Confidence level used: 0.95
```

Session Info

```
sessionInfo()
```

```
## R version 3.6.0 (2019-04-26)
## Platform: x86_64-apple-darwin15.6.0 (64-bit)
## Running under: macOS Mojave 10.14.5
##
## Matrix products: default
## BLAS:   /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRblas.0.dylib
## LAPACK: /Library/Frameworks/R.framework/Versions/3.6/Resources/lib/libRlapack.dylib
##
## locale:
## [1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8
##
## attached base packages:
## [1] stats     graphics  grDevices utils     datasets  methods   base
##
## other attached packages:
##  [1] emmeans_1.3.4     AER_1.2-6         sandwich_2.5-1
##  [4] lmtest_0.9-37     zoo_1.8-6         car_3.0-3
##  [7] carData_3.0-2     nlme_3.1-140      lme4_1.1-21
## [10] Matrix_1.2-17     epiR_1.0-2        survival_2.44-1.1
## [13] knitr_1.23        forcats_0.4.0     stringr_1.4.0
## [16] dplyr_0.8.1       purrr_0.3.2       readr_1.3.1
## [19] tidyr_0.8.3       tibble_2.1.2      ggplot2_3.1.1
## [22] tidyverse_1.2.1
##
```

```
## loaded via a namespace (and not attached):
##  [1] httr_1.4.0        jsonlite_1.6      splines_3.6.0
##  [4] modelr_0.1.4      Formula_1.2-3     assertthat_0.2.1
##  [7] highr_0.8         cellranger_1.1.0  yaml_2.2.0
## [10] pillar_1.4.1      backports_1.1.4   lattice_0.20-38
## [13] glue_1.3.1        digest_0.6.19     rvest_0.3.4
## [16] minqa_1.2.4       colorspace_1.4-1  htmltools_0.3.6
## [19] plyr_1.8.4        pkgconfig_2.0.2   broom_0.5.2
## [22] haven_2.1.0       xtable_1.8-4      mvtnorm_1.0-10
## [25] scales_1.0.0      openxlsx_4.1.0.1  rio_0.5.16
## [28] generics_0.0.2    withr_2.1.2       lazyeval_0.2.2
## [31] cli_1.1.0         magrittr_1.5      crayon_1.3.4
## [34] readxl_1.3.1      estimability_1.3  evaluate_0.14
## [37] MASS_7.3-51.4     xml2_1.2.0        foreign_0.8-71
## [40] tools_3.6.0       data.table_1.12.2 hms_0.4.2
## [43] munsell_0.5.0     zip_2.0.2         compiler_3.6.0
## [46] rlang_0.3.4       grid_3.6.0        nloptr_1.2.1
## [49] rstudioapi_0.10   labeling_0.3      rmarkdown_1.13
## [52] boot_1.3-22       gtable_0.3.0      abind_1.4-5
## [55] curl_3.3          R6_2.4.0          lubridate_1.7.4
## [58] stringi_1.4.3     BiasedUrn_1.07    Rcpp_1.0.1
## [61] tidyselect_0.2.5  xfun_0.7
```