Qualifying Exam 2019

Exam #7

6/1/2019

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

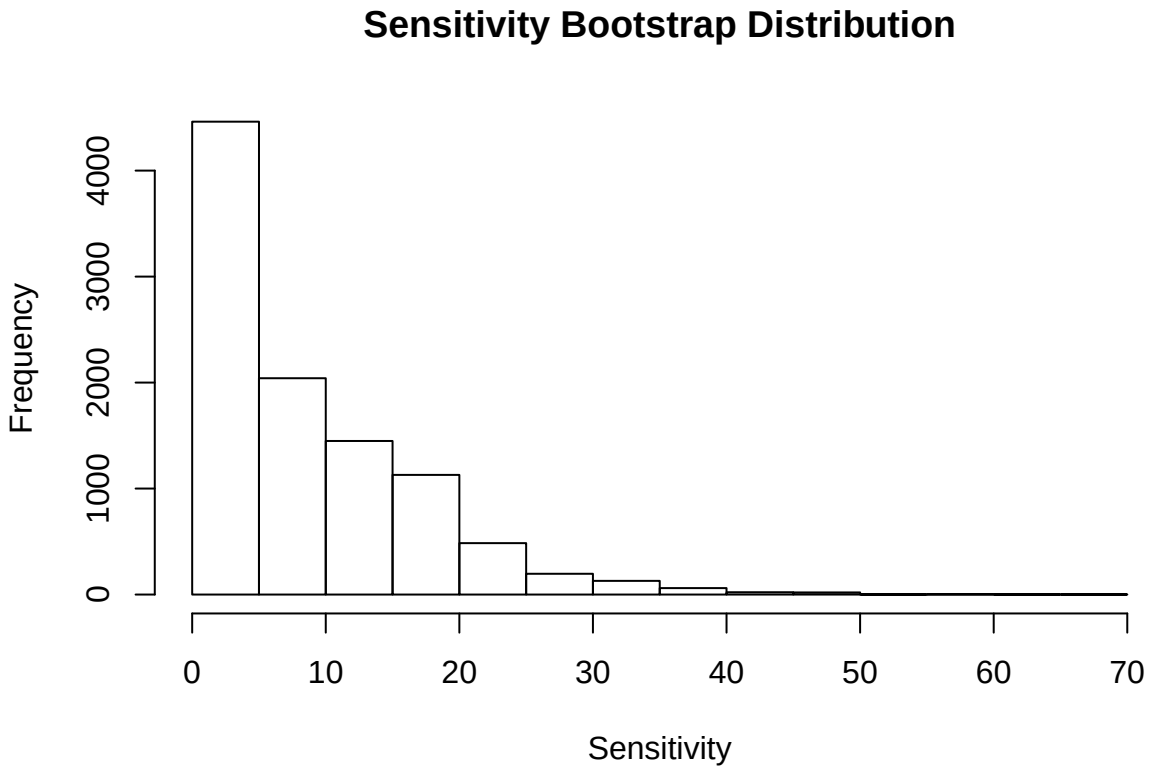
If we want to maximize true positives while minimizing false positives, the optimal threshold is the one with the highest sensitivity and lowest 100 - specificity. A threshold of >= 3 meals per week including fish would provide a 100% true positive rate and a 72% false negative rate.

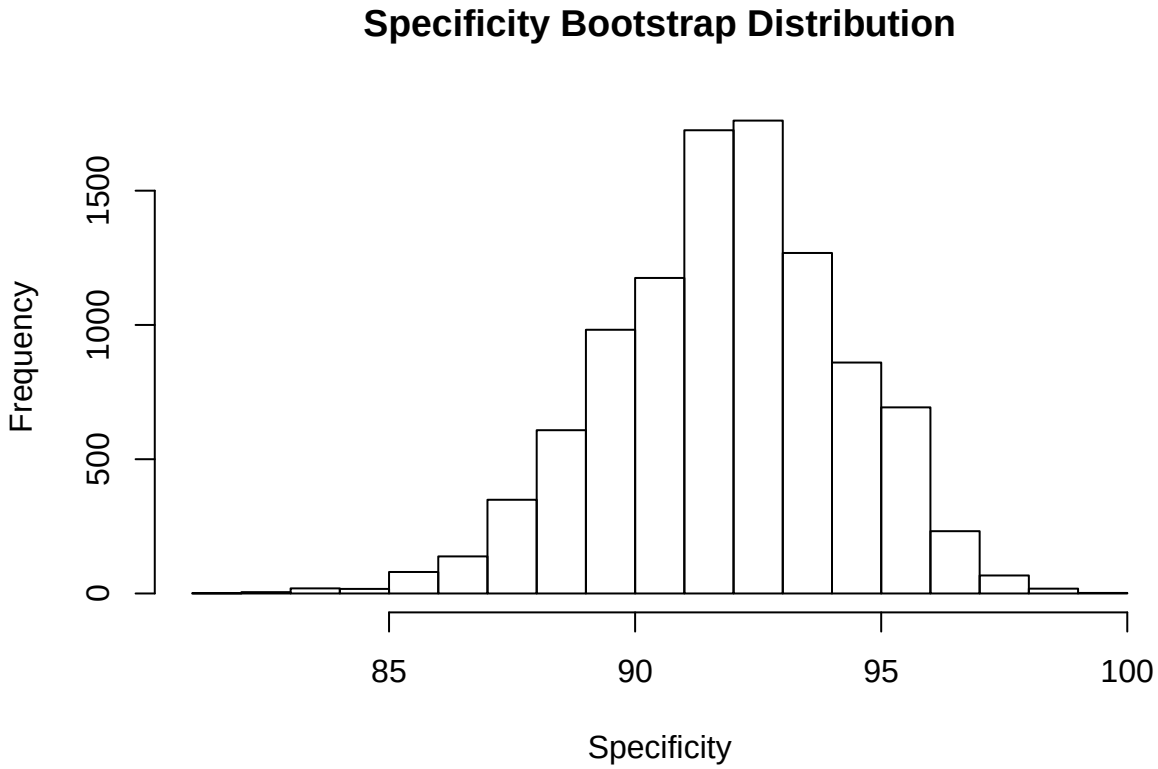
ii. True negatives

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

TALK ABOUT THESE RESULTS

	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. In this model we are treating the number of fish meals per week as continuous. $\hat{\beta}_3$, $\hat{\beta}_4$, 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 do not consume any fish parts.

ii. Results

	Estimate	Std. Error	t value	Pr(>|t|)
Intercept	0.9040000	0.8420135	1.0736170	0.2849986
Fisherman = Yes	0.2464962	0.7417227	0.3323293	0.7401801
Fish Meals per Week	0.0964710	0.0568348	1.6973942	0.0920335
Fish Part = Muscle	3.0608992	1.0782386	2.8387957	0.0052624
Fish Part = Muscle and Whole	1.6757469	1.0186581	1.6450533	0.1023934
Fish Part = Whole	3.0091672	1.3660412	2.2028378	0.0293821

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.

$$log(Y_i^*) \sim N(\beta_0 + \beta_1 X_i, \sigma^2)$$

$$Yi = 1 \text{ when } Y_i^* > l = 0.001$$

$$Yi = 1 \text{ when } log(Y_i^*) > log(l)$$

$$P(log(Y_i^*) > log(l)) = 1 - P(log(Y_i^*) \leq log(l))$$

$$= 1 - P(\frac{log(Y_i^*) - \beta_0 - \beta_1 X_i}{\sigma} \leq \frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma}) = 1 - P(Z \leq \frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma})$$

$$= 1 - \Phi(\frac{log(l) - \beta_0 - \beta_1 X_i}{\sigma})$$

$$\text{Set } P(log(Y_i^*) > log(l)) = \theta_i$$

$$Y_i \ \ \text{Bernoulli}(\theta_i)$$

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 standardized 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 (Hessian):

```r
# 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
```

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

Use glm() to check the coefficient estimates:

```r
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 (Hessian):

```r
# 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, ul=-Inf, ll=Inf) {
  sigma = exp(par[length(par)])
  beta = par[-length(par)]
  if (!is.infinite(ll)) {
    limit = ll
    indicator = y > ll
  } else {
    limit = ul
    indicator = y < ul
  }
  lp = X %*% beta
  ll = sum(indicator * log((1/sigma)*dnorm((y-lp)/sigma)) ) +
    sum((1-indicator) * log(pnorm((lp-limit)/sigma, lower=is.infinite(ll))))
  -ll
}

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$log_Yi, X=X, ll=-7,hessian = T)
mle$par
```

```
##             (Intercept) factor(location)Niagara               log_sigma
##               -4.4733164                2.2163173                0.5030678
```

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

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

```r
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.

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

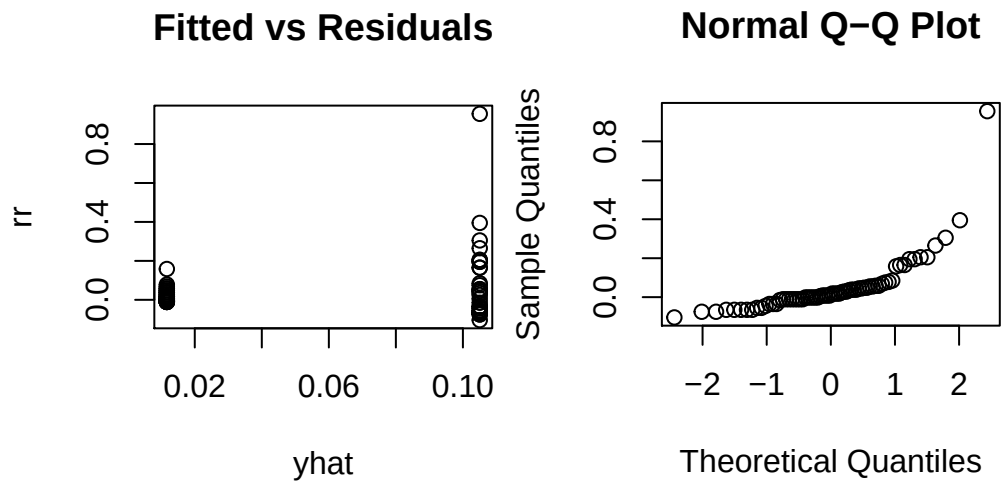
```
##             (Intercept) factor(location)Niagara
##               0.0514566               0.2065025
```

d.

Assuming that log concentration is normally distributed, the estimated mean concentration will be 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, ul=-Inf, ll=Inf) {
  a <- par[3:4]
  sigma = exp(X%*%a)
  beta = par[-length(par)]
  if (!is.infinite(ll)) {
    limit = ll
    indicator = y > ll
  } else {
    limit = ul
    indicator = y < ul
  }
  lp = X %*% beta
  ll = sum(indicator * log((1/sigma)*dnorm((y-lp)/sigma)) ) +
    sum((1-indicator) * log(pnorm((lp-limit)/sigma, lower=is.infinite(ll))))
  -ll
```

```
}
summary(tobit)
```

```
##
## Call:
## AER::tobit(formula = contam.lev ~ factor(location), left = 0.001,
##     dist = "lognormal", data = pcbs)
##
## Observations:
##          Total  Left-censored     Uncensored Right-censored
##             68              9             59              0
##
## Coefficients:
##                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)             -4.45075    0.28489 -15.622  < 2e-16 ***
## factor(location)Niagara  2.19720    0.39867   5.511 3.56e-08 ***
## Log(scale)               0.48580    0.09637   5.041 4.63e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Scale: 1.625
##
## Log Normal distribution
## Number of Newton-Raphson Iterations: 4
## Log-likelihood: 35.54 on 3 Df
## Wald-statistic: 30.38 on 1 Df, p-value: 3.56e-08
```

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

Random intercept for person and random slope for group?

Significant concerns for multicollinearity with week (VIF > 10), so treating time a categorical variable. Justifiable given everyone was measured at the same time and the data is categorical in a sense.

```
time_mod_cat <- lme(logvl ~ trt*factor(week), data=hiv,random = ~1|pid)
kable(anova(time_mod_cat),caption = "Type 3 Tests of Fixed Effects")
```

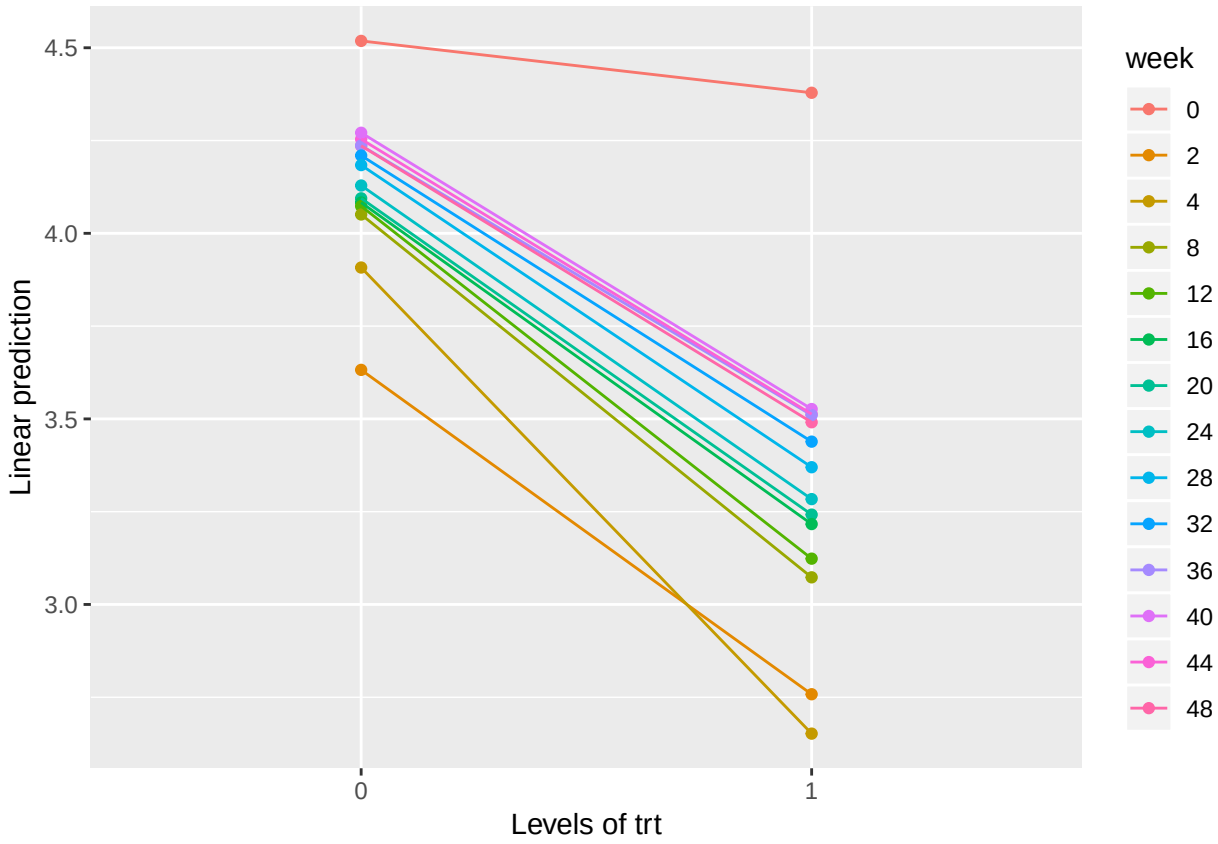
Table 6: Type 3 Tests of Fixed Effects

	numDF	denDF	F-value	p-value
(Intercept)	1	2886	4988.91521	0
trt	1	222	58.50980	0
factor(week)	13	2886	120.30906	0
trt:factor(week)	13	2886	18.70325	0

```
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)
kable(t)
```

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

```
emmip(time_mod_cat, week~trt)
```


4.5
4.0
3.5
3.0
Linear prediction
0
1
Levels of trt
week
0
2
4
8
12
16
20
24
28
32
36
40
44
48

Appendix

Load libraries, import and format HIV data

```
knitr::opts_chunk$set(echo = TRUE)
library(knitr)
library(epiR)
library(ggplot2)
library(lme4)
library(nlme)
library(VGAM)
library(AER)
```

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

Question 1

Create dummy variables for fish data

```
# 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 sensitivity and specificity results

```
# 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

```
# 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 for 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")
# 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
```

Bootstrap and Normal Percentile Confidence Intervals

```
# 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

```
# 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])
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 Model

```
lin_mod <- lm(MeHg ~ factor(fisherman)+fishmlwk+factor(fishpart),data = fish)
results <- as.data.frame(summary(lin_mod)$coefficients)
rownames(results) <- c("Intercept","Fisherman = Yes",
                       "Fish Meals per Week","Fish Part = Muscle",
                       "Fish Part = Muscle and Whole",
                       "Fish Part = Whole")
```

Confidence and Prediction Intervals

Confidence

```
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))
```

Prediction

```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*(1+(t(a)%*%(solve((t(x)%*%x)))%*%a)))
U <- yhat + t*sqrt(mse*(1+(t(a)%*%(solve((t(x)%*%x)))%*%a)))
```

Question 2

Format data

Concentration estimates

```r
tobit_mean <- AER::tobit(contam.lev ~ factor(location)-1, left = 0.001,
                   data = pcbs, dist = "lognormal")
summary(tobit_mean)$coefficients
```

```
##
## Test of coefficients:
##
##                             Estimate Std. Error  z value  Pr(>|z|)
## factor(location)Fort Erie -4.450746   0.284894 -15.6225 < 2.2e-16 ***
## factor(location)Niagara   -2.253542   0.279085  -8.0748 6.761e-16 ***
## Log(scale)                 0.485800   0.096366   5.0412 4.626e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
median(log(pcbs$contam.lev[which(pcbs$location=="Fort Erie")]))
```

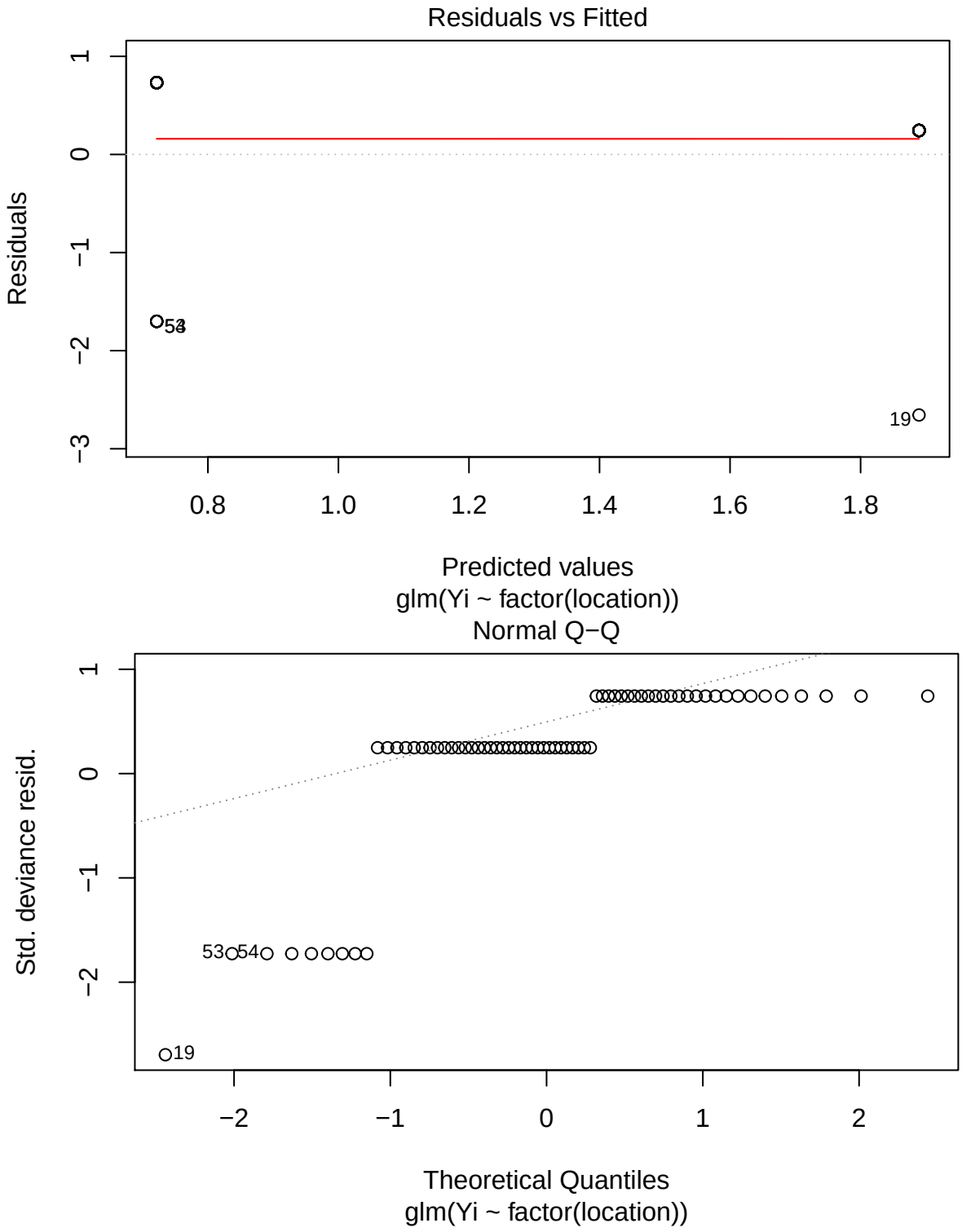
```
## [1] -3.912023
```

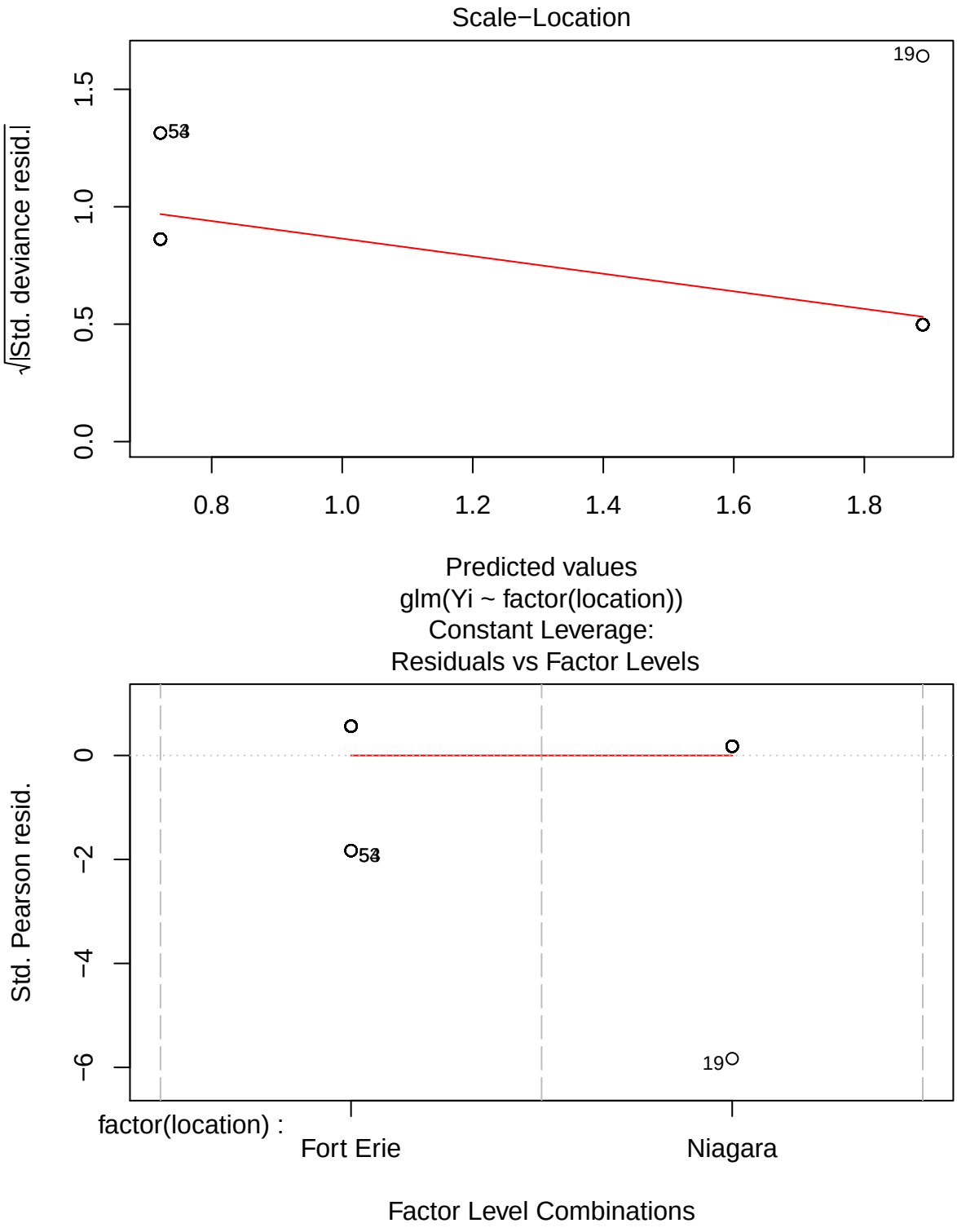
```r
median(log(pcbs$contam.lev[which(pcbs$location=="Niagara")]))
```

```
## [1] -2.003167
```

Probit residuals

```r
plot(probit_mod)
```

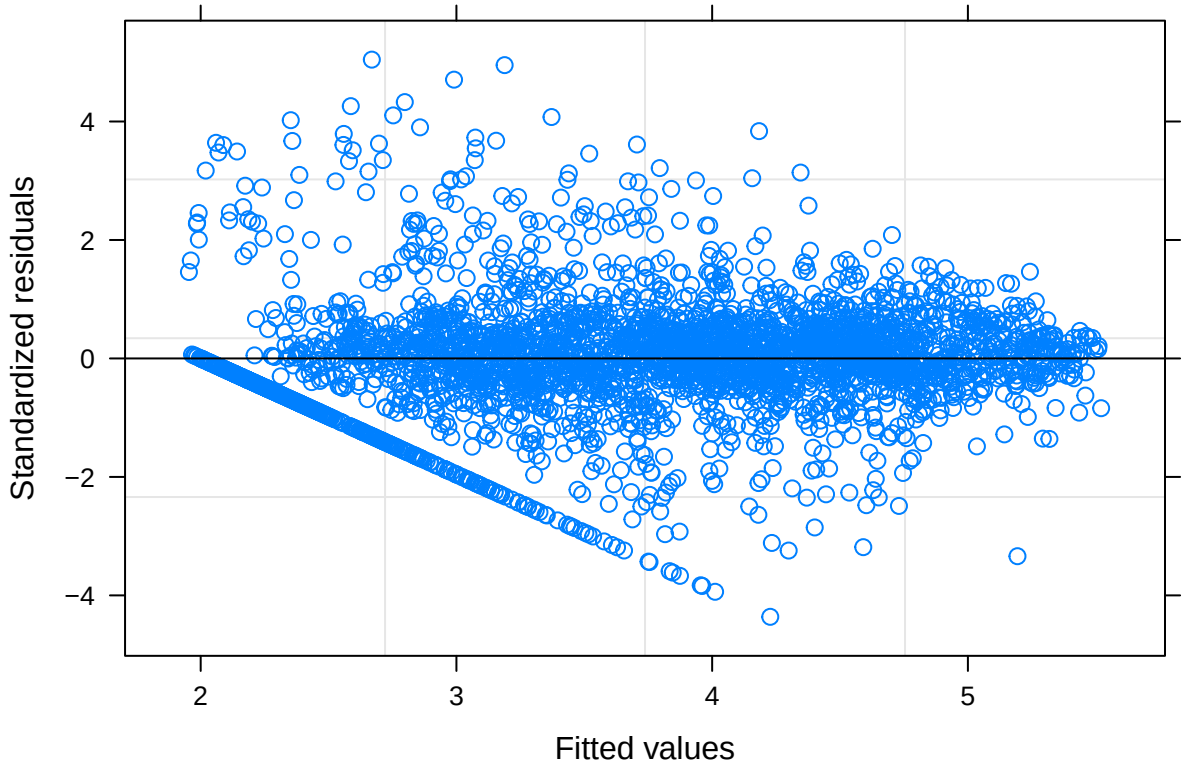

Residuals vs Fitted
Residuals
Predicted values
glm(Yi ~ factor(location))
Normal Q−Q
Std. deviance resid.
Theoretical Quantiles
glm(Yi ~ factor(location))
19
53
54


Scale−Location
19
√|Std. deviance resid.|
1.5
1.0
0.5
0.0
0.8
1.0
1.2
1.4
1.6
1.8
Predicted values
glm(Yi ~ factor(location))
Constant Leverage:
Residuals vs Factor Levels
Std. Pearson resid.
0
−2
−4
−6
19
factor(location) :
Fort Erie
Niagara
Factor Level Combinations

Question 3

Model selection

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




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

	Value	Std.Error	DF	t-value	p-value
(Intercept)	3.6683697	0.0620880	2910	59.083430	0.0000000
week	-0.0059638	0.0021723	2910	-2.745422	0.0060804
I(week^2)	0.0002589	0.0000451	2910	5.744080	0.0000000

```
vif(time_mod)
```

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

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

```
##              Model df      AIC      BIC    logLik   Test L.Ratio p-value
```

```
## time_mod         1  6 5343.823 5380.127 -2665.911
## time_mod_cat     2 16 4475.373 4572.184 -2221.686 1 vs 2  888.45  <.0001
```

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

```
##              Model df      AIC      BIC    logLik   Test  L.Ratio p-value
## time_mod         1  7 5197.977 5240.332 -2591.989
## time_mod_cat     2 16 4475.373 4572.184 -2221.686 1 vs 2 740.6045  <.0001
```

```
# Categorical time
time_mod_cat <- lme(logvl ~ factor(week), data=hiv,random = ~1|pid)
# 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
```

```
# 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
```

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

	Value	Std.Error	DF	t-value	p-value
(Intercept)	4.5184595	0.0837895	2886	53.926292	0.0000000
trt1	-0.1397249	0.1179708	222	-1.184403	0.2375202
factor(week)2	-0.8863874	0.0549521	2886	-16.130169	0.0000000
factor(week)4	-0.6106306	0.0549521	2886	-11.112043	0.0000000
factor(week)8	-0.4676667	0.0549521	2886	-8.510435	0.0000000
factor(week)12	-0.4451982	0.0549521	2886	-8.101562	0.0000000
factor(week)16	-0.4355135	0.0549521	2886	-7.925323	0.0000000
factor(week)20	-0.4241532	0.0549521	2886	-7.718591	0.0000000
factor(week)24	-0.3896306	0.0549521	2886	-7.090362	0.0000000
factor(week)28	-0.3346396	0.0549521	2886	-6.089656	0.0000000
factor(week)32	-0.3088829	0.0549521	2886	-5.620943	0.0000000
factor(week)36	-0.2818739	0.0549521	2886	-5.129442	0.0000003
factor(week)40	-0.2475045	0.0549521	2886	-4.504001	0.0000069
factor(week)44	-0.2648108	0.0549521	2886	-4.818935	0.0000015
factor(week)48	-0.2825045	0.0549521	2886	-5.140918	0.0000003
trt1:factor(week)2	-0.7342409	0.0773694	2886	-9.490064	0.0000000
trt1:factor(week)4	-1.1160773	0.0773694	2886	-14.425300	0.0000000
trt1:factor(week)8	-0.8379617	0.0773694	2886	-10.830654	0.0000000

	Value	Std.Error	DF	t-value	p-value
trt1:factor(week)12	-0.8101292	0.0773694	2886	-10.470920	0.0000000
trt1:factor(week)16	-0.7266104	0.0773694	2886	-9.391439	0.0000000
trt1:factor(week)20	-0.7124398	0.0773694	2886	-9.208284	0.0000000
trt1:factor(week)24	-0.7052278	0.0773694	2886	-9.115069	0.0000000
trt1:factor(week)28	-0.6742453	0.0773694	2886	-8.714621	0.0000000
trt1:factor(week)32	-0.6310906	0.0773694	2886	-8.156846	0.0000000
trt1:factor(week)36	-0.5866483	0.0773694	2886	-7.582429	0.0000000
trt1:factor(week)40	-0.6046725	0.0773694	2886	-7.815392	0.0000000
trt1:factor(week)44	-0.6009060	0.0773694	2886	-7.766710	0.0000000
trt1:factor(week)48	-0.6053008	0.0773694	2886	-7.823513	0.0000000

Cite

https://github.com/m-clark/Miscellaneous-R-Code/blob/master/ModelFitting/tobit.R