Homework 08 Solutions

11/10/2021

```
library(tidyverse)
library(pander)
library(nlme)
library(janitor)
```

Problem 1

```
data_mat <- matrix(c(
  1, 300, 0, 0, 4, 6,
  2, 300, 1, 0, 4, 6,
  3, 500, 0, 1, 4, 6,
  4, 500, 1, 1, 4, 6,
  5, 200, 0, 0, 10, 12,
  6, 200, 1, 0, 10, 12,
  7, 200, 0, 1, 10, 12,
  8, 200, 1, 1, 10, 12), byrow = TRUE, nrow = 8)

colnames(data_mat) <- c("Category", "num", "x",
                        "T", "y_0", "y_1")
df <- data.frame(data_mat)
```

(a)

```
N <- sum(df$num)
ace <- 1/N*sum(df$num * (df$y_1 - df$y_0))
ace
```

```
## [1] 2
```

We can actually directly see that for each individual, the difference between y^1 and y^0 is 2 (which means the average is also this value), but the above equation also shows us the the ACE is 2.

(b)

If we only view x as the only pre-treatment predictor (as suggested in the beginning of the problem), then we can observe that there T and x appear to be independent - aka $P(T = t, X = x) = P(T = t)P(X = x)$, this is explored specifically in part (d), but the below tables show this through showing that $P(T = t) = P(T = t|x = x)$ for $x \in \{0, 1\}$.

```
tab_info <- df %>%
  mutate(T = paste0("T",T)) %>%
  group_by(x,T) %>% summarise(count = sum(num)) %>%
  tidyr::pivot_wider(id_cols = x, names_from = T, values_from = count) %>%
  janitor::adorn_totals(where = c("row", "col"))
```

```
## `summarise()` has grouped output by 'x'. You can override using the `.groups` argument.
```

```
tab_info
```

```
##       x   T0   T1 Total
##       0  500  700  1200
##       1  500  700  1200
##   Total 1000 1400  2400
```

```
tab_info %>%
  janitor::adorn_percentages()
```

```
##       x        T0        T1 Total
##       0 0.4166667 0.5833333     1
##       1 0.4166667 0.5833333     1
##   Total 0.4166667 0.5833333     1
```

With our full fiew of the data we can see that (y^0, y^1) pairing actually differ between the first four categories and the last four categories. Given that x doesn't capture this different it wouldn't be too crazy to suggest that there are actually more pre-treatment predictors of the outcome. If one just defines another grouping variable `x2` to be $(1, 1, 1, 1, 0, 0, 0, 0)$ to capture this we can see that `x2` and `T` are not independent of each other. None-the-less this goes beyond the problem description.

(c)

```
N_T <- sum(df$num[df$T == 1])
N_not_T <- sum(df$num[df$T == 0])

1/N_T * sum((df$num * df$y_1)[df$T == 1]) -
  1/N_not_T * sum((df$num * df$y_0)[df$T == 0])
```

```
## [1] 1.314286
```

If there was no other confounders we'd expect this value to approximate the ACE. Interestingly it does not, but that seems to be related to the second commentary we had in part (b), which showed that the proportion those in the treatment group where `x2 = 1` was larger than the proportion of those in the treatment group where `x2 = 0`, and this relateds to the smaller value we see for the treatment effect as those in the `x2 = 1` group had smaller outcomes under both treatments than the `x2 = 0` group.

(d)

```
prob_T <-sum(df$num * (df$T == 1)) / sum(df$num)

prob_T
```

```
## [1] 0.5833333
```

```
prob_T_given_x1 <- sum(df$num * (df$x == 1) * (df$T == 1)) /
  sum(df$num * (df$x == 1))

prob_T_given_x1
```

```
## [1] 0.5833333
```

We find that, in the above population, $P(T = 1) = 0.5833$, which is the same as $P(T = 1|x = 1)$.

(e)

Below are 2 approaches to estimate the desired coefficient and se. Note that the coefficient values for the intercept and T directly related to parts of the calculation in part (c). The intercept being the `mean(y[T=0])` and the coefficient of T being the difference between `mean(y[T=1]) - mean(y[T=0])`.

Weighted regression approach

A weighted regression solution is faster that expanding out the full dataset, but requires a correct ion for the model's degrees of freedom. See this blogpost for more comments.

```
df <- df %>%
  mutate(y = ifelse(T, y_1, y_0))

wls <- lm(y ~ T + x, data = df, weights = num)
wls$df.residual <- sum(df$num) - length(wls$coefficients)
#^ correction for number of observations / degrees of freedom

summ_w <- summary(wls)
```

```
## Warning in summary.lm(wls): residual degrees of freedom in object suggest this
## is not an "lm" fit
```

```
summ_w
```

```
##
## Call:
## lm(formula = y ~ T + x, data = df, weights = num)
##
## Weighted Residuals:
##    Min     1Q Median     3Q    Max
## -41.57 -39.14   6.29  53.34  60.61
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.4000     0.1058   60.51   <2e-16 ***
## T             1.3143     0.1163   11.30   <2e-16 ***
## x             0.0000     0.1147    0.00        1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.81 on 2397 degrees of freedom
## Multiple R-squared:  0.05055,    Adjusted R-squared:  0.9972
## F-statistic: 63.81 on 2 and 2397 DF,  p-value: < 2.2e-16
```

```
coef(summ_w)[2,]
```

```
##     Estimate   Std. Error      t value     Pr(>|t|)
## 1.314286e+00 1.163413e-01 1.129682e+01 7.202413e-29
```

Individualized approach

Another way would just be to create an "individual" data frame (where each row represents one individual). Below is a messy way to create the desired data.

```
df2 <- data.frame(x = unlist(sapply(1:nrow(df),
                                    function(idx){rep(df$x[idx],
                                                      each = df$num[idx])})),
```

```
                  T = unlist(sapply(1:nrow(df),
                                    function(idx){rep(df$T[idx],
                                                      each = df$num[idx])})),
                  y = unlist(sapply(1:nrow(df),
                                    function(idx){rep(df$y[idx],
                                                      each = df$num[idx])})))

ls_all_obs <- lm(y ~ T + x, data = df2)
summ <- summary(ls_all_obs)
summ
```

```
##
## Call:
## lm(formula = y ~ T + x, data = df2)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -2.400 -1.886 -1.714  3.600  4.286
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.400e+00  1.058e-01   60.51   <2e-16 ***
## T           1.314e+00  1.163e-01   11.30   <2e-16 ***
## x           2.901e-15  1.147e-01    0.00        1
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.81 on 2397 degrees of freedom
## Multiple R-squared:  0.05055,    Adjusted R-squared:  0.04976
## F-statistic: 63.81 on 2 and 2397 DF,  p-value: < 2.2e-16
```

```
coef(summ)[2,]
```

```
##     Estimate   Std. Error      t value     Pr(>|t|)
## 1.314286e+00 1.163413e-01 1.129682e+01 7.202413e-29
```

Problem 2.

(a)

The treatment variable is no longer randomized in the way intended by the researcher, because individuals have selected their own level of treatment. This selection process is very likely confounded with the outcome variable, meaning that there is some unobserved variable that is correlated with both the treatment variable and the outcome.

For example, imagine that the therapy actually has no effect, but there is an "optimism" variable that determines both how many sessions a person attends and their emotional state at the end of the study. In other words, the people who engage the most with the treatment are those in the best state at the end of the study, but not because of the therapy. Then the number of sessions attended will strongly predict the outcome variable, but not because of an effect of the therapy.

An unobserved confounder could also obscure a real treatment effect. Suppose that instead of an optimism variable, there is a variable representing emotional wellbeing that is inversely correlated with the number of sessions attended. That is, those who start out the most depressed seek the most treatment, perhaps because they feel that they have the most to gain. Suppose the treatment *does* have a beneficial effect that increases

with the number of sessions attended, and suppose that everyone seeks out just enough treatment to get them to roughly the same level of emotional wellbeing at the end of the study. Then a coefficient for number of treatment sessions could end up being 0 even though there is a treatment effect!

In sum: whenever there is noncompliance in an experiment, there is potential for confounding due to selection effects. Including a variable for the amount of treatment received *does not* address this and can lead to highly misleading results.

(b)

This is perfectly set up for an Instrumental Variable analysis. The assignment variable (assignment to treatment or control) is the instrument, and the number of sessions attended is the treatment variable. The assignment variable almost certainly affects the treatment variable, but it should have no effect on the outcome through other pathways. (That is, the mere fact of having been assigned to the treatment or control group shouldn't affect participants' emotional wellbeing at the end of the study.) An experimental design with noncompliance is almost always a good candidate for an IV analysis.

An analysis based on propensity score matching won't work here unless we have variables with which to construct the propensity score. The propensity score is the probability of receiving treatment as a function of some set of covariates X. That is, the propensity score is a function $\pi(x) := P(A = 1|X = x)$, where $A = 1$ means that someone receives treatment. We can define it more broadly as the probability of receiving a particular amount of treatment a: $\pi_a(x) = P(A = a|X = x)$. We want as rich a set of covariates X as possible in order to perform matching. If the investigator collected these prior to the start of the study, then this analysis may be possible.

Problem 3

(a)

A summary of the model is below. Diagnostic plots are in Figure 1, and the estimated autocorrelation between the residuals is plotted in Figure 2.

The residuals are very roughly normally distributed, but they appear to show an increasing trend relative to the fitted values. The scale-location plot also appears to show that the variance is increasing with fitted values, and the autocorrelation plot shows large autocorrelations, all of which suggest that the linear model with assumed independence between the residuals is not appropriate.

```
carlsen <- read.table("CarlsenQ.txt", header=TRUE)
m1 <- lm(Sales ~ . -Case - Time, data=carlsen)
pander(summary(m1), caption = "Problem 3a: Summary of ordinary linear model.")
```

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	4421	1235	3.58	0.00336
Temp	78.18	33.89	2.307	0.03817
Sun	2.585	1.556	1.661	0.1206
Q2	-879.7	1121	-0.7845	0.4468
Q3	-1552	1541	-1.007	0.3323
Q4	715.4	474.3	1.508	0.1554

Table 2: Problem 3a: Summary of ordinary linear model.

Observations	Residual Std. Error	R^2	Adjusted R^2
19	287.2	0.9537	0.9359

```
# summary(m1)
```

```
par(mfrow = c(2, 2),  mar = c(4, 5, 2, 1))
plot(m1)
```

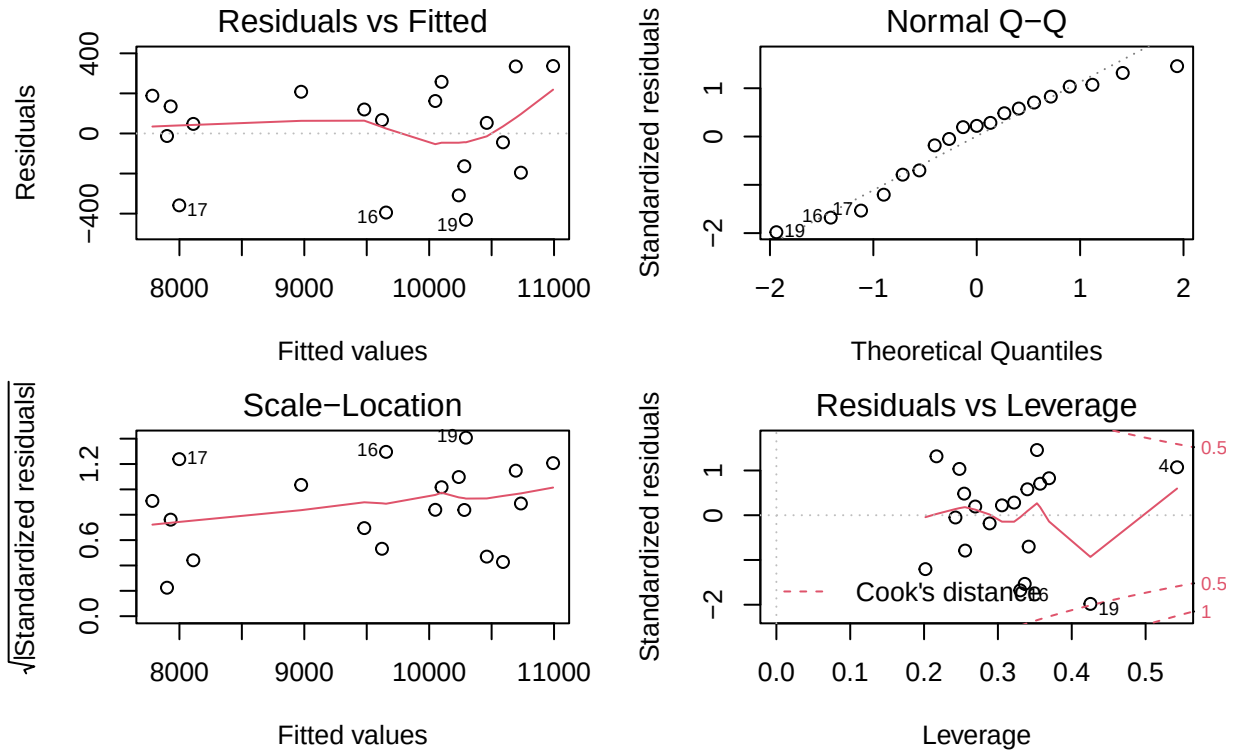

Figure 1: Problem 3a: Diagnostic plots for ordinary linear model.

```
acf(resid(m1), main = "")
```

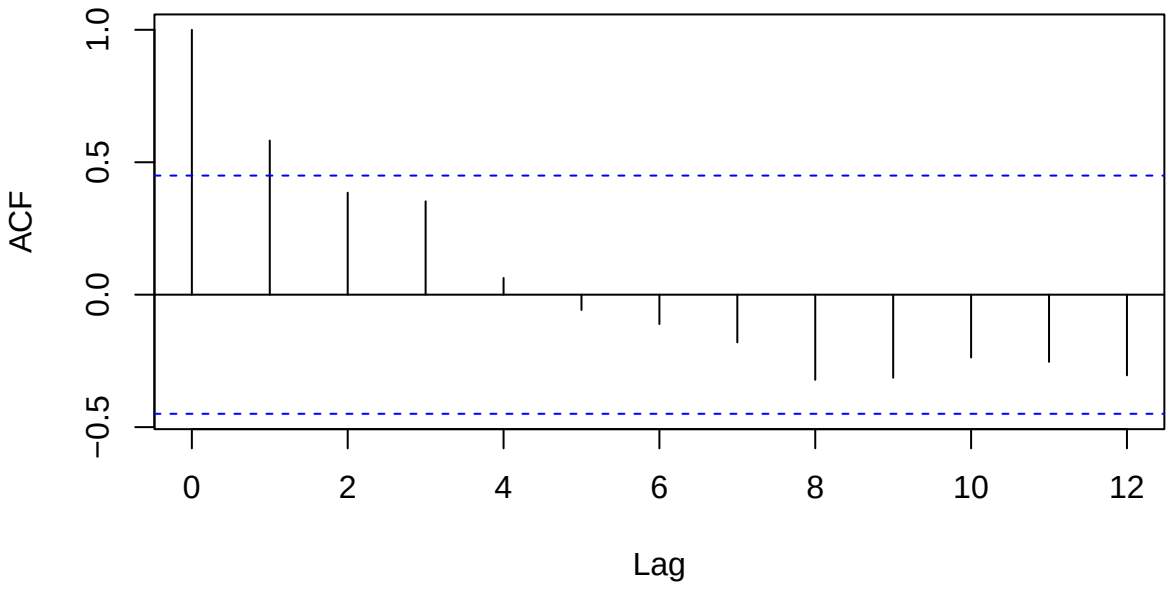

Figure 2: Problem 3a: acf plot for ordinary linear model residuals.

(b)

Note that the model can be estimated with either Maximum Likelihood (ML) or Restricted Maximum Likelihood (REML). I have used the default, which is REML, but either one is fine. They will produce different estimates. The code should look like one of the following:

```
model <- gls(Sales ~ . -Case - Time, data=carlsen, correlation=corAR1())
model <- gls(Sales ~ . -Case - Time, data=carlsen, correlation=corAR1(),
             method = "ML")
```

i. A summary of this model is below. From this summary, the estimated value of ρ, the lag-1 autocorrelation parameter, is 0.87. (If you use ML for estimation, the estimate is 0.79).

ii. The coefficient values have changed in magnitude, though not in sign. The estimated standard errors in the AR1 model are much smaller than in the ordinary linear model, which is to be expected if the model is properly accounting for correlation among the residuals. As a result, both `Sun` and `Q4` are significant in the AR1 model but not in the ordinary linear model at conventional thresholds.

```
m2 <- gls(Sales ~ . -Case - Time, data=carlsen, correlation=corAR1())
cat("Problem 3b: Summary of AR1 model:"); summary(m2)
```

```
## Problem 3b: Summary of AR1 model:
```

```
## Generalized least squares fit by REML
##   Model: Sales ~ . - Case - Time
##   Data: carlsen
##        AIC      BIC    logLik
##   212.9427 217.4623 -98.47134
##
```

```
## Correlation Structure: AR(1)
##  Formula: ~1
##  Parameter estimate(s):
##       Phi
## 0.8688414
##
## Coefficients:
##                Value Std.Error   t-value p-value
## (Intercept) 5215.269  747.4869  6.977071  0.0000
## Temp          55.545   22.7958  2.436611  0.0300
## Sun            2.192    0.7802  2.809438  0.0148
## Q2           -36.255  662.9242 -0.054690  0.9572
## Q3          -371.191  941.7222 -0.394162  0.6999
## Q4           946.075  295.3927  3.202772  0.0069
##
##  Correlation:
##      (Intr) Temp   Sun    Q2     Q3
## Temp -0.870
## Sun   0.005 -0.357
## Q2    0.914 -0.938  0.033
## Q3    0.912 -0.960  0.100  0.992
## Q4    0.747 -0.937  0.485  0.840  0.876
##
## Standardized residuals:
##        Min         Q1        Med         Q3        Max
## -1.1843635 -0.2908069  0.4983337  0.5709198  1.1275844
##
## Residual standard error: 409.3627
## Degrees of freedom: 19 total; 13 residual
```

(c)

```
T <- -2*(as.numeric(logLik(m1)) - as.numeric(logLik(m2)))
pval <- pchisq(T, df = 1, lower.tail = FALSE)
```

The log likelihoods are -131 for the ordinary linear model and -98 for the AR1 mode. The difference in degrees of freedom is 1, so we compare the statistic $-2(-131-(-98)) = 66$ to a $\chi^2_{(1)}$ distribution. The pvalue is $8.0841612 \times 10^{-16}$, so we reject the null hypothesis and conclude that the autocorrelation is not 0.

(d)

A summary of the model is below. Diagnostic plots are in Figure 3, and the autocorrelation plot of the residuals is in Figure 4.

The diagnostic plots suggest problems with the fit. The residuals are not very close to normally distributed, and the scale-location plot appears to show a relationship between the fitted values and the variance. The autocorrelation plot, however, shows much reduced autocorrelation among the residuals, as hoped for.

The likelihood of the model is `-130`, as compared to `-98` for the model in 3c. It appears to be more effective to estimate the autocorrelation simultaneously with estimating the model, as in 2c, rather than following a two-stage estimation procedure.

```
r <- resid(m1)
rho <- cor(r[-1], r[-length(r)])
Sigma <- diag(nrow(carlsen))
```

```
Sigma <- rho^abs(row(Sigma) - col(Sigma))
S <- chol(Sigma)
S_inv <- solve(t(S))

Xstar <- S_inv %*% model.matrix(m1)
ystar <- S_inv %*% carlsen$Sales

m3 <- lm(ystar ~ Xstar - 1)
pander(summary(m3),
       caption = paste("Problem 2d: Summary of ordinary linear model",
                       "with transformed variables."))
```

	Estimate	Std. Error	t value	Pr(>|t|)
Xstar(Intercept)	5090	793.5	6.415	2.29e-05
XstarTemp	61.01	24.64	2.476	0.02781
XstarSun	2.196	0.8827	2.488	0.02722
XstarQ2	-214.5	725.8	-0.2956	0.7722
XstarQ3	-631.1	1026	-0.6148	0.5493
XstarQ4	879.1	322.7	2.724	0.01738

Table 4: Problem 2d: Summary of ordinary linear model with transformed variables.

Observations	Residual Std. Error	R^2	Adjusted R^2
19	277.7	0.9979	0.9969

```
par(mfrow = c(2, 2),  mar = c(4, 4.5, 2, 1))
plot(m3)
```

```
acf(resid(m3), main = "")
```

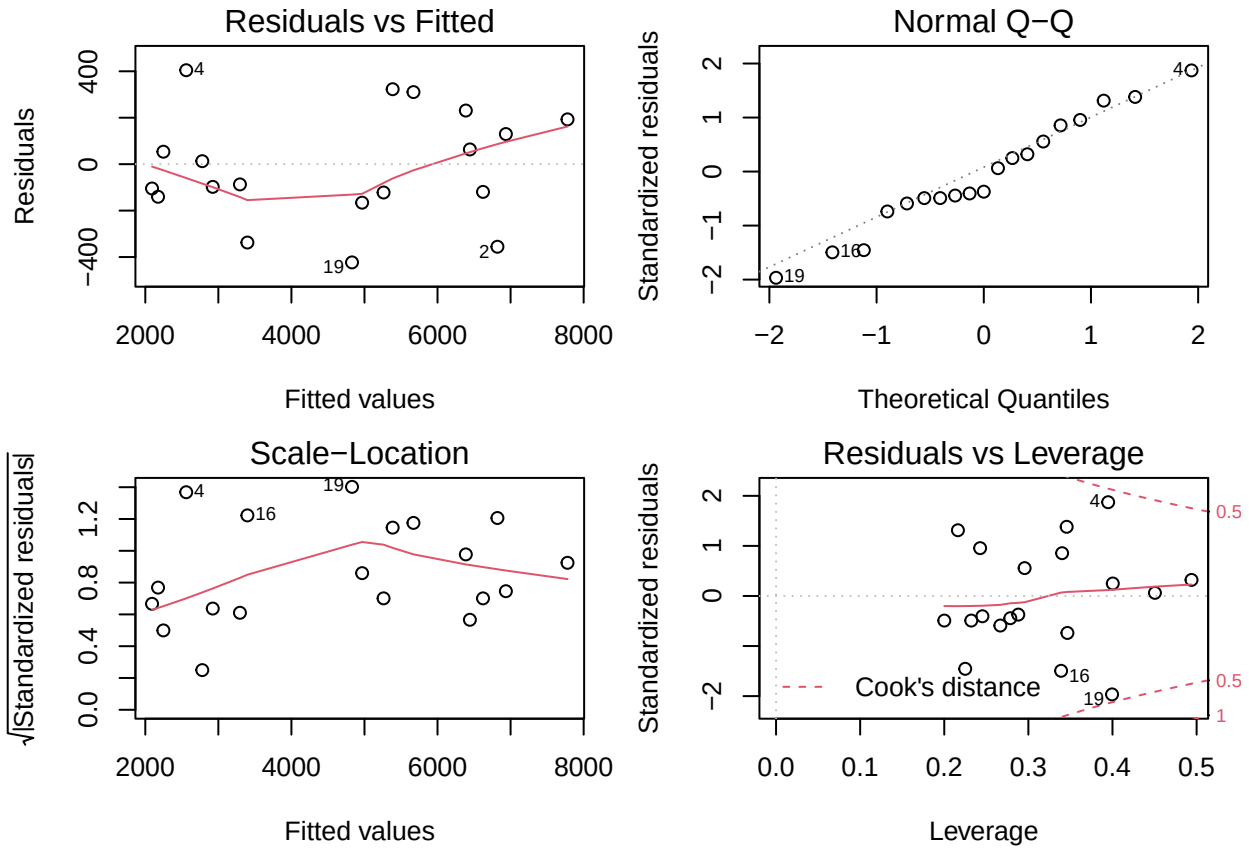

Figure 3: Problem 3d: diagnostic plots.

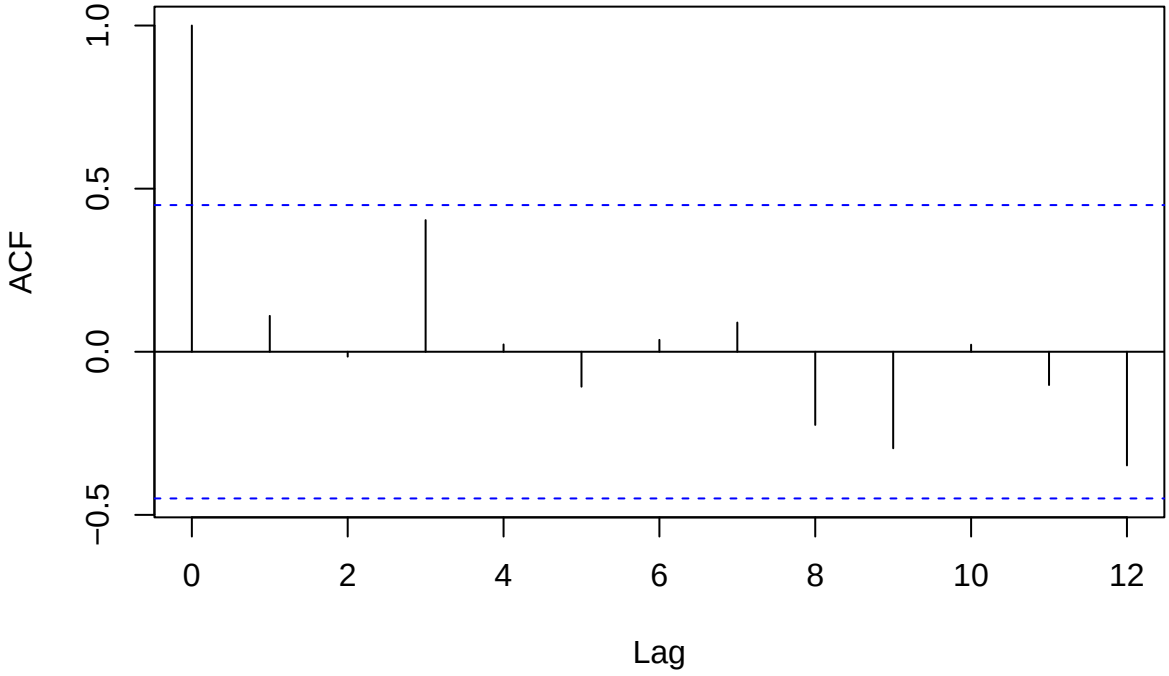

Figure 4: Problem 3d: acf plot for ordinary linear model residuals.