

STAT-S631

Assignment 4

John Koo

```
dp <- loadNamespace('dplyr')
import::from(magrittr, `%>%`, `%<>%`)
library(ggplot2)
theme_set(theme_bw())
```

Question 1

[From ALR 2.16]

```
un.df <- alr4::UN11 %>%
  dp$mutate(country = rownames(.))
```

Part 1

```
fertility.mod <- lm(log(fertility) ~ log(ppgdp), data = un.df)

summary(fertility.mod)
```

Call:

```
lm(formula = log(fertility) ~ log(ppgdp), data = un.df)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-0.79828	-0.21639	0.02669	0.23424	0.95596

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	2.66551	0.12057	22.11	<2e-16 ***
log(ppgdp)	-0.20715	0.01401	-14.79	<2e-16 ***

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3071 on 197 degrees of freedom

Multiple R-squared: 0.526, Adjusted R-squared: 0.5236

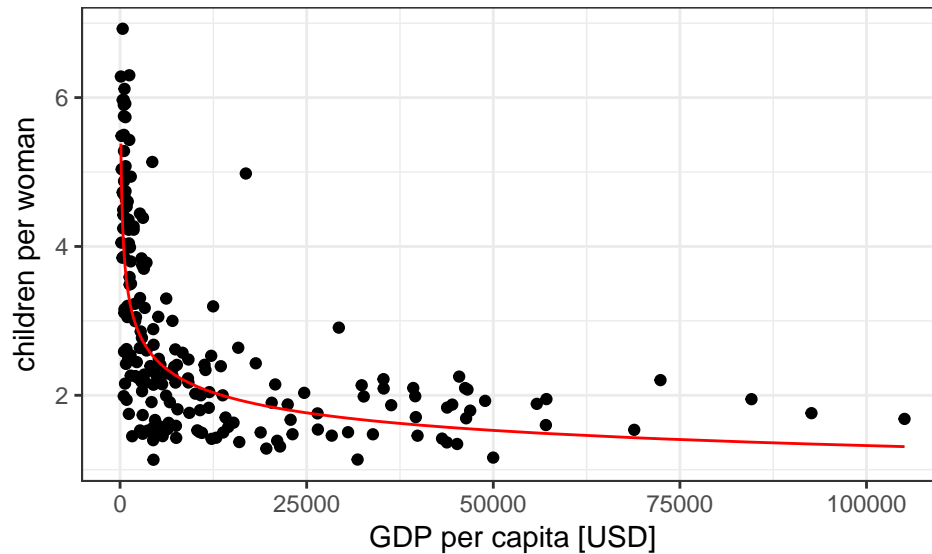
F-statistic: 218.6 on 1 and 197 DF, p-value: < 2.2e-16

Part 2

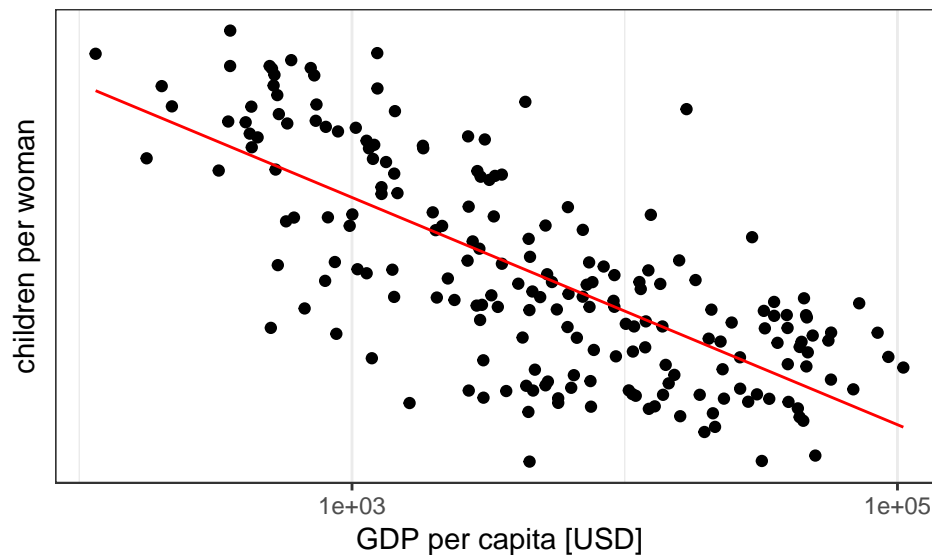
```
fert.vs.gdp.plot <- un.df %>%
  dp$mutate(fert.hat =
    exp(fertility.mod$coefficients['(Intercept)']) *
    ppgdp ** fertility.mod$coefficients['log(ppgdp)']) %>%
```

```
ggplot() +
  geom_point(aes(x = ppgdp, y = fertility)) +
  geom_line(aes(x = ppgdp, y = fert.hat), colour = 'red') +
  labs(x = 'GDP per capita [USD]', y = 'children per woman')
```

fert.vs.gdp.plot



fert.vs.gdp.plot + `scale_x_log10()` + `scale_y_log10()`



Part 3

Using $\alpha = 0.05$:

alternatively, we can just look at summary(fertility.mod)

```
b1.mean <- fertility.mod$coefficients['log(ppgdp)']
```

```
b1.sterr <- summary(fertility.mod)$coefficients['log(ppgdp)', 'Std. Error']
```

```
b1.tval <- unname(-b1.mean / b1.sterr)
pt(-abs(b1.tval), df = fertility.mod$df.residual)
```

```
[1] 4.531178e-34
```

Here our p -value is much smaller than α .

Part 4

```
summary(fertility.mod)$r.squared
```

```
[1] 0.525985
```

The R^2 value can be interpreted as the proportion of variance that has been explained by the model. In this case, about half of the variance of Y is accounted for by the linear fit with X .

Part 5

```
cross.prod.sum <- function(x, y = NULL) {
  # sum of cross product (e.g., SXX, SYX, SXY)

  if (is.null(y)) y <- x
  sum((x - mean(x)) * (y - mean(y)))
}

pred.y <- function(x, model, alpha = .05) {
  # predicts new y given a simple linear model and input
  # also gives confidence interval for a certain alpha value

  # compute t for the alpha C.I. and deg of freedom
  t.alpha <- qt(1 - alpha / 2, model$df.residual)

  # compute expected
  y.mean <- unname(model$coefficients[1] + model$coefficients[2] * x)

  # compute standard error
  x.mean <- mean(model$model[, 2])
  sxx <- cross.prod.sum(model$model[, 2])
  y.se <-
    sigma(model) *
    sqrt(1 + 1 / nrow(model$model) + (x - x.mean) ** 2 / sxx)

  return(list(y = y.mean,
             ci = c(y.mean - y.se * t.alpha, y.mean + y.se * t.alpha)))
}

new.y <- pred.y(log(1000), fertility.mod)
new.y
```

```
$y
[1] 1.234567
```

```
$ci  
[1] 0.6258791 1.8432555
```

```
lapply(new.y, exp)
```

```
$y  
[1] 3.436891
```

```
$ci  
[1] 1.869889 6.317070
```

So the 95% prediction interval for Y given $\log X = 1000$ is [1.87, 6.317]

Part 6

```
# lowest fertility  
dp$filter(un.df, fertility == min(fertility))$country
```

```
[1] "Bosnia and Herzegovina"
```

```
# highest fertility  
dp$filter(un.df, fertility == max(fertility))$country
```

```
[1] "Niger"
```

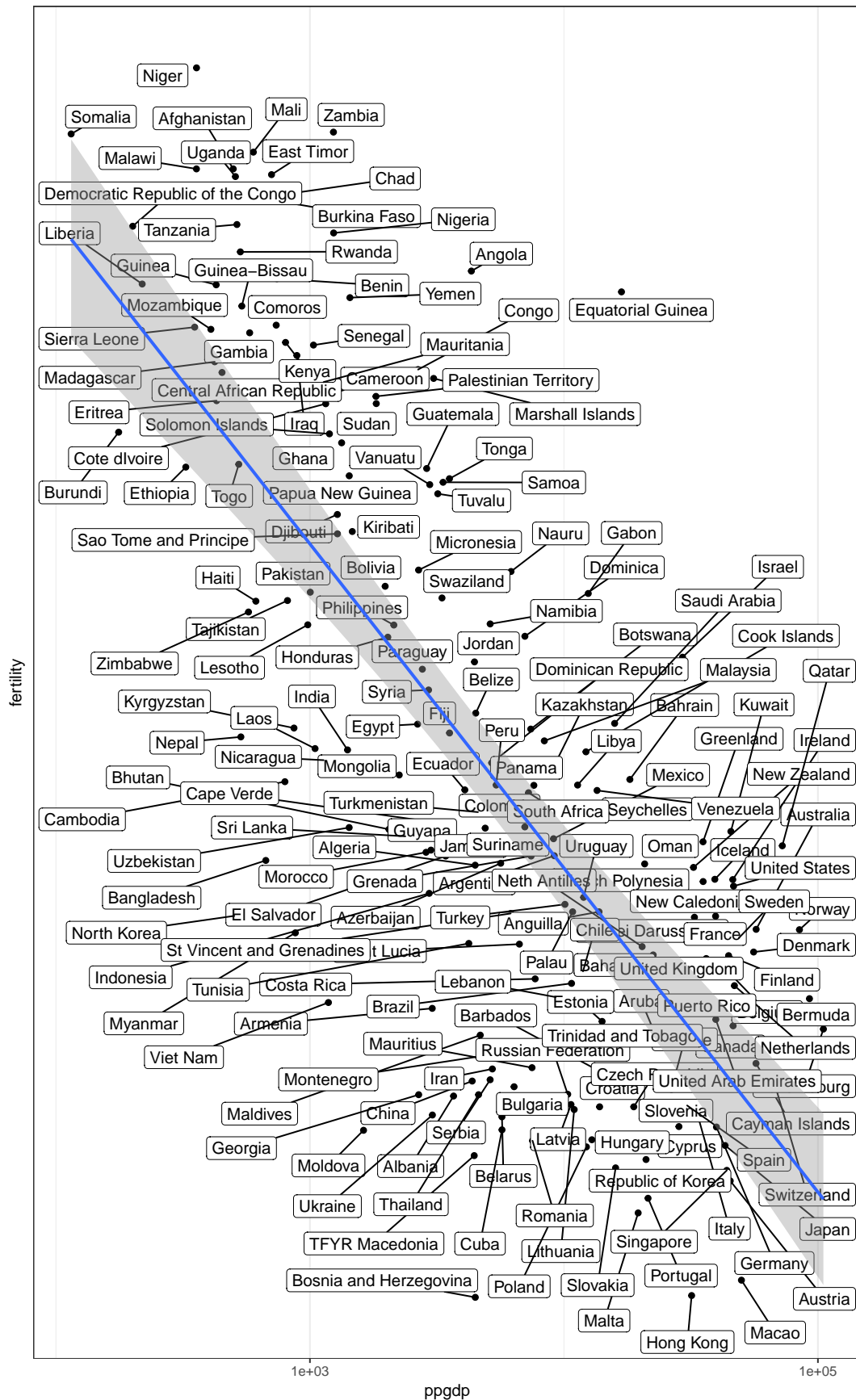
```
# most positive residuals  
un.df$country[order(-fertility.mod$residuals)[1:2]]
```

```
[1] "Equatorial Guinea" "Angola"
```

```
# most negative residuals  
un.df$country[order(fertility.mod$residuals)[1:2]]
```

```
[1] "Bosnia and Herzegovina" "Moldova"
```

```
ggplot(un.df) +  
  geom_point(aes(x = ppgdp, y = fertility)) +  
  ggrepel::geom_label_repel(aes(x = ppgdp, y = fertility, label = country)) +  
  scale_x_log10() + scale_y_log10() +  
  stat_smooth(aes(x = ppgdp, y = fertility), method = 'lm', level = .99)
```

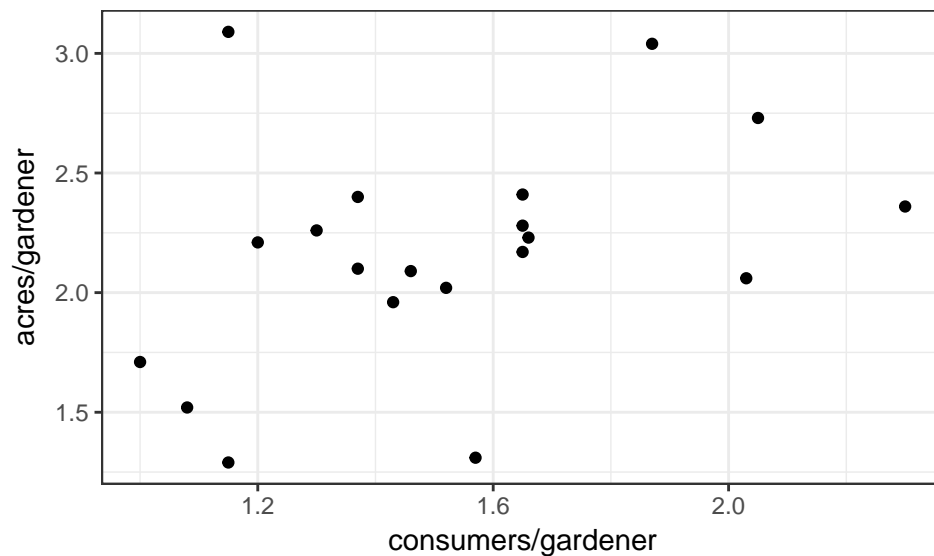


Question 2

```
sahlins.df <- read.delim('~/.dev/stats-hw/stat-s631/Sahlins.txt', sep = ' ')
```

Part a

```
ggplot(sahlins.df) +  
  geom_point(aes(x = consumers, y = acres)) +  
  labs(x = 'consumers/gardener', y = 'acres/gardener')
```



From the scatterplot, the data do not appear to be particularly linear, although there appears to be a very slight positive correlation. We can compute this:

```
cor(sahlins.df$consumers, sahlins.df$acres)
```

```
[1] 0.3756561
```

Most of the data appear to be clustered in the center with a ring of points surrounding it. One household has an unusually high value for acres per gardener—it's almost 3 times its value for consumers per gardener (~3 vs ~1).

Part b

```
sahlins.mod <- lm(acres ~ consumers, data = sahlins.df)  
summary(sahlins.mod)
```

Call:

```
lm(formula = acres ~ consumers, data = sahlins.df)
```

Residuals:

Min	1Q	Median	3Q	Max
-0.8763	-0.1873	-0.0211	0.2135	1.1206

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.3756	0.4684	2.937	0.00881 **
consumers	0.5163	0.3002	1.720	0.10263

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.4543 on 18 degrees of freedom

Multiple R-squared: 0.1411, Adjusted R-squared: 0.0934

F-statistic: 2.957 on 1 and 18 DF, p-value: 0.1026

The results indicate that, if we set the conventional value of $\alpha = .05$, we would fail to reject the null hypothesis that $\beta_1 \neq 0$, implying that there is no significant relationship between acres per gardener and consumers per gardener. However, using the same value of α , we reject the null hypothesis that $\beta_0 = 0$, indicating that each household receives some amount regardless of productivity.

The residual standard error $\hat{\sigma}$ is 0.4543

If we remove the 4th data point:

```
sahlins.mod.2 <- lm(acres ~ consumers, data = sahlins.df[-4, ])  
summary(sahlins.mod.2)
```

Call:

```
lm(formula = acres ~ consumers, data = sahlins.df[-4, ])
```

Residuals:

Min	1Q	Median	3Q	Max
-0.82291	-0.16808	0.03215	0.23505	0.69061

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.0000	0.3969	2.519	0.0221 *
consumers	0.7216	0.2514	2.870	0.0106 *

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.3681 on 17 degrees of freedom

Multiple R-squared: 0.3264, Adjusted R-squared: 0.2868

F-statistic: 8.238 on 1 and 17 DF, p-value: 0.01061

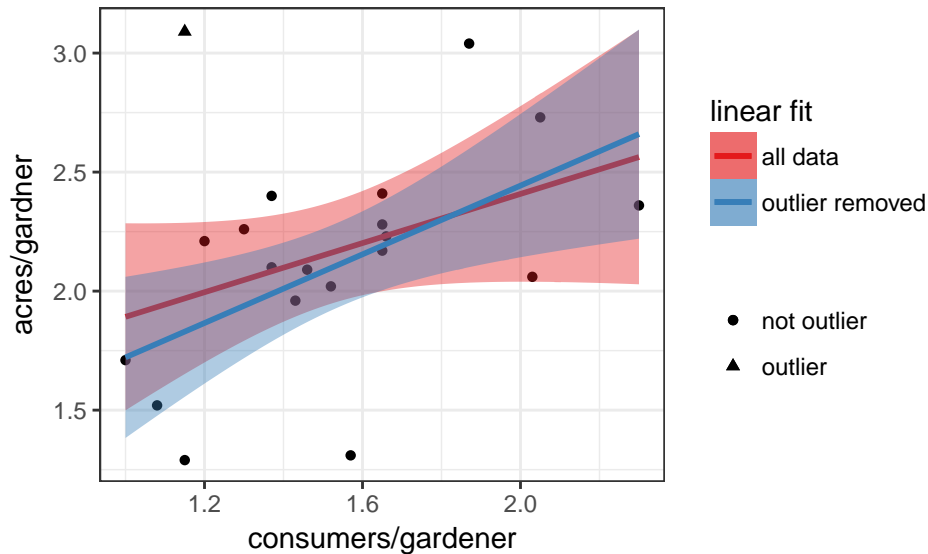
We would reject the null hypothesis for both β_0 and β_1 . This would imply that each household receives some base amount but also can work for additional resources. This is a different conclusion than before, when we used all the data.

```
ggplot() +  
  geom_point(data = sahlins.df[-4, ],  
            aes(x = consumers, y = acres, shape = 'not outlier')) +  
  labs(x = 'consumers/gardener',  
       y = 'acres/gardner',  
       colour = 'linear fit', fill = 'linear fit',  
       shape = NULL) +  
  geom_point(data = sahlins.df[4, ],  
            aes(x = consumers, y = acres, shape = 'outlier')) +  
  stat_smooth(data = sahlins.df,  
            aes(x = consumers, y = acres,
```

```

    colour = 'all data', fill = 'all data'),
  method = 'lm') +
stat_smooth(data = sahlins.df[-4, ],
  aes(x = consumers, y = acres,
    colour = 'outlier removed', fill = 'outlier removed'),
  method = 'lm') +
scale_colour_brewer(palette = 'Set1') +
scale_fill_brewer(palette = 'Set1')

```



We can say that the first model is not a good fit since we failed to reject the null hypothesis that $\beta_1 \neq 0$.

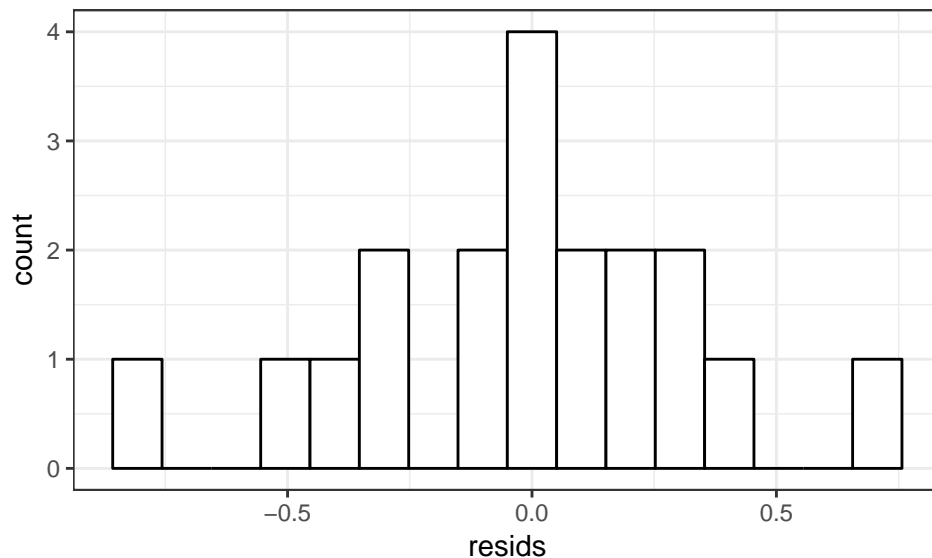
For the second model (with the 4th data point removed), we can check the residuals:

```

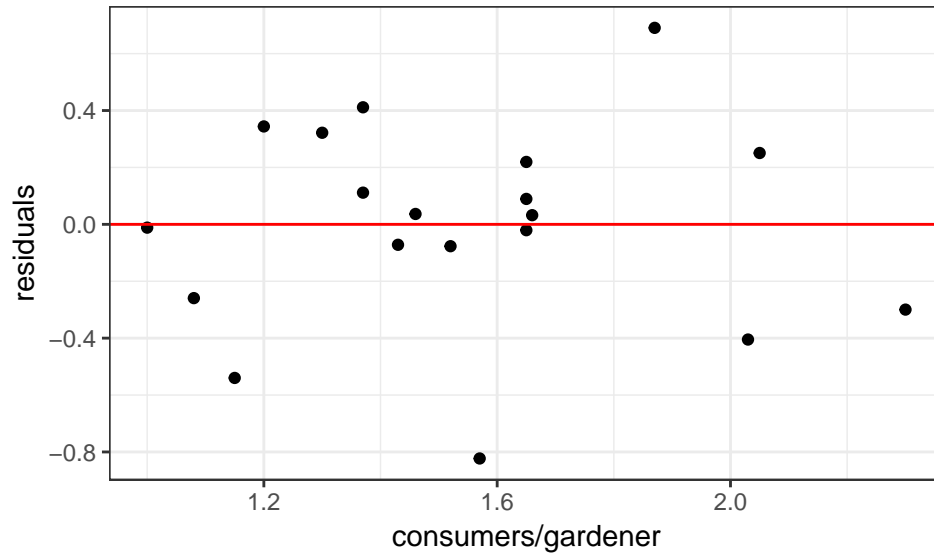
sahlins.outlier.removed.df <- sahlins.df[-4, ] %>%
  dp$mutate(resids = sahlins.mod.2$residuals)

ggplot(sahlins.outlier.removed.df) +
  geom_histogram(aes(x = resids),
    fill = 'white', colour = 'black', bins = 16)

```

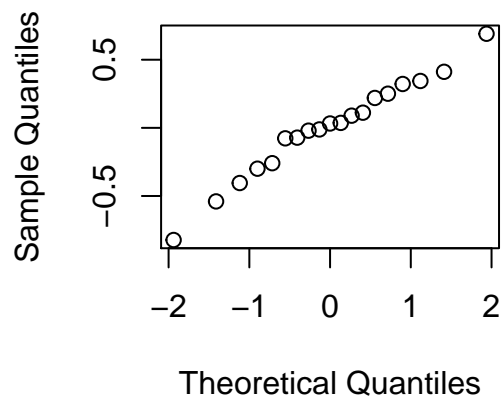



```
ggplot(sahlins.outlier.removed.df) +
  geom_point(aes(x = consumers,
                 y = resids)) +
  geom_hline(yintercept = 0, colour = 'red') +
  labs(x = 'consumers/gardener', y = 'residuals')
```



```
qqnorm(sahlins.outlier.removed.df$resids)
```

Normal Q-Q Plot



Here we can see that the residuals do not depend on the input variable and appear approximately normally distributed around 0, so the assumptions of the linear model hold. As for the strength of the predictions, $\hat{\sigma} \approx 0.3681$, so on average the model is ~ 0.4 acres/gardener off the true value.

Part c

For the first model

```
summary(sahlins.mod)$coefficients
```

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	1.3756445	0.4684047	2.936872	0.008813794

```
consumers    0.5163201  0.3002335  1.719728  0.102629261
```

If we set $\alpha = .05$ and the null hypotheses as $\beta_0 = 0$ and $\beta_1 = 0$, then we would reject the $\beta_0 = 0$ and fail to reject $\beta_1 = 0$ (these are from the `summary` command from part (b)).

Using the same α , we can compute the confidence intervals:

```
alpha <- .05
t.alpha <- qt(1 - alpha / 2, sahlins.mod$df.residual)

mod1.b0.ci <-
  c(-t.alpha, t.alpha) *
  summary(sahlins.mod)$coefficients['(Intercept)', 'Std. Error'] +
  sahlins.mod$coefficients['(Intercept)']

mod1.b1.ci <-
  c(-t.alpha, t.alpha) *
  summary(sahlins.mod)$coefficients['consumers', 'Std. Error'] +
  sahlins.mod$coefficients['consumers']

print(mod1.b0.ci)

[1] 0.3915628 2.3597263
print(mod1.b1.ci)
```

```
[1] -0.1144471 1.1470872
```

The C.I. for β_0 does not contain 0 while the C.I. for β_1 does, which is consistent with our hypothesis tests.

For the second model

Using the same α , we reject the null hypothesis for both $\beta_0 = 0$ and $\beta_1 = 0$ (from the `summary` command from part (b)).

For the confidence intervals:

```
alpha <- .05
t.alpha <- qt(1 - alpha / 2, sahlins.mod.2$df.residual)

mod2.b0.ci <-
  c(-t.alpha, t.alpha) *
  summary(sahlins.mod.2)$coefficients['(Intercept)', 'Std. Error'] +
  sahlins.mod.2$coefficients['(Intercept)']

mod2.b1.ci <-
  c(-t.alpha, t.alpha) *
  summary(sahlins.mod.2)$coefficients['consumers', 'Std. Error'] +
  sahlins.mod.2$coefficients['consumers']

print(mod2.b0.ci)

[1] 0.1625647 1.8374433
print(mod2.b1.ci)
```

```
[1] 0.191157 1.252031
```

The C.I.s for neither β_0 nor β_1 contain 0, which is consistent with the results of our hypothesis tests.

Part d

For prediction (the first question):

```
pred.y(1.5, sahlins.mod, alpha = .02)
```

```
$y  
[1] 2.150125
```

```
$ci  
[1] 0.961766 3.338483
```

```
# alternatively  
predict(sahlins.mod,  
        newdata = data.frame(consumers = 1.5),  
        interval = 'predict',  
        level = .98)
```

```
      fit      lwr      upr  
1 2.150125 0.961766 3.338483
```

For our estimate of the mean (the second question):

```
mean.y <- function(x, model, alpha = .05) {  
  # predicts mean y given a simple linear model and input  
  # also gives confidence interval for a certain alpha value  
  
  # compute t for the alpha C.I. and deg of freedom  
  t.alpha <- qt(1 - alpha / 2, model$df.residual)  
  
  # compute expected  
  y.mean <- unname(model$coefficients[1] + model$coefficients[2] * x)  
  
  # compute standard error  
  x.mean <- mean(model$model[, 2])  
  sxx <- cross.prod.sum(model$model[, 2])  
  y.se <-  
    sigma(model) *  
    sqrt(1 / nrow(model$model) + (x - x.mean) ** 2 / sxx)  
  
  return(list(y = y.mean,  
             ci = c(y.mean - y.se * t.alpha, y.mean + y.se * t.alpha)))  
}  
  
mean.y(1.5, sahlins.mod, alpha = .02)
```

```
$y  
[1] 2.150125
```

```
$ci  
[1] 1.890234 2.410016
```

```
# alternatively  
predict(sahlins.mod,  
        newdata = data.frame(consumers = 1.5),  
        interval = 'confidence',  
        level = .98)
```

	fit	lwr	upr
1	2.150125	1.890234	2.410016

The prediction interval (first value) is what we expect a new value to have given that $x = 1.5$. On the other hand, the confidence interval (second value) is what we expect the mean of all y would be given that $x = 1.5$. The standard error for each case is different.