a2

```
data = read.csv("aquifer.csv")
type <- factor(data$type)
levels(type)
```

```
## [1] "Axeann"     "Bellefonte" "Nittany"    "Stonehenge"
```

```
data$type = type
regtype = lm(porosity~type, data=data)
regdensity = lm(porosity~density, data=data)
regresidue = lm(porosity~residue, data=data)
regglength = lm(porosity~glength, data=data)

summary(regtype)
```

```
##
## Call:
## lm(formula = porosity ~ type, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.3787 -0.7544 -0.2380  0.3234  5.4112
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)      1.5833     0.5513   2.872  0.00785 **
## typeBellefonte  -0.1189     0.7797  -0.152  0.87994
## typeNittany      1.6054     0.8037   1.998  0.05593 .
## typeStonehenge  -0.8453     0.9225  -0.916  0.36762
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.654 on 27 degrees of freedom
## Multiple R-squared:  0.2334, Adjusted R-squared:  0.1482
## F-statistic: 2.739 on 3 and 27 DF,  p-value: 0.06286
```

```
summary(regdensity)
```

```
##
## Call:
## lm(formula = porosity ~ density, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7517 -1.0377 -0.6670  0.8708  6.7735
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)    6.374     14.947   0.426    0.673
```

```
## density       -1.683      5.531  -0.304    0.763
##
## Residual standard error: 1.82 on 29 degrees of freedom
## Multiple R-squared:  0.003182,   Adjusted R-squared:  -0.03119
## F-statistic: 0.09258 on 1 and 29 DF,  p-value: 0.7631
```

```
summary(regresidue)
```

```
##
## Call:
## lm(formula = porosity ~ residue, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.4271 -0.9281 -0.3276  0.3381  5.7335
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.76015    0.51382   1.479   0.1498
## residue      0.10942    0.04308   2.540   0.0167 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.649 on 29 degrees of freedom
## Multiple R-squared:  0.182,  Adjusted R-squared:  0.1538
## F-statistic: 6.451 on 1 and 29 DF,  p-value: 0.0167
```

```
summary(regglength)
```

```
##
## Call:
## lm(formula = porosity ~ glength, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.2089 -0.9742 -0.5304  0.7724  6.1904
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   8.9269     3.6047   2.476   0.0193 *
## glength      -0.6517     0.3297  -1.977   0.0576 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.711 on 29 degrees of freedom
## Multiple R-squared:  0.1188, Adjusted R-squared:  0.08837
## F-statistic: 3.908 on 1 and 29 DF,  p-value: 0.05763
```

We have when limestone type is Axeann, porosity is expected to be 1.5833 (Other features stay the same)
Other features stay the same, When limestone type is Bellefone, its porosity is expected to be 0.1189 less than a Axeann type limestone
Other features stay the same, When limestone type is Nittany , its porosity is expected to be 1.6054 more than a Axeann type limestone
Other features stay the same, When limestone type is Stonehenge, its porosity is expected to be 0.8453 less than a Axeann type limestone
Other features stay the same, when stone density increase by 1, porosity is expected to decrease by 1.683

Other features stay the same, when stone residue increase by 1, porosity is expected to increase by 0.1094
Other features stay the same, when stone glength increase by 1, porosity is expected to decrease by 0.6517

we observe density has smallest R-square 0.003182, meaning smallest relationship with porosity
glength has 2nd smallest R-square 0.1188, meaning 2nd smallest relationship with porosity
residue has second largest R-square 0.182, meaning 2nd largest relationship with porosity
type has largest R-square 0.2334, meaning largest relationship with porosity
pvalue of H0: "density has no relationship with porosity" is 0.7631, we do not reject null hypothesis
pvalue of H0: "glength has no relationship with porosity" is 0.05763, we do not reject null hypothesis, but there is no reason we should trust the null hypotheses
pvalue of H0: "residue has no relationship with porosity" is 0.0167, we reject null hypothesis
pvalue of H0: "type has no relationship with porosity" is 0.06286, we do not reject null hypothesis, but there is no reason we should trust the null hypotheses

we observe although type has largest R-square, it does not have lowest pval, which is as expected as degree of freedom of type is smallest in F test as there are more than 1 parameters.

(b)

```
model = lm(porosity~type+residue+glength+density, data = data)
summary(model)
```

```
##
## Call:
## lm(formula = porosity ~ type + residue + glength + density, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -1.7453 -0.8990 -0.1925  0.7051  4.5626
##
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)
## (Intercept)    27.54944   18.16052   1.517   0.1423
## typeBellefonte  0.60229    0.96957   0.621   0.5403
## typeNittany     0.86258    0.95245   0.906   0.3741
## typeStonehenge -0.67589    0.88180  -0.766   0.4509
## residue         0.10680    0.04741   2.253   0.0337 *
## glength        -0.69642    0.35651  -1.953   0.0625 .
## density        -7.20444    6.85068  -1.052   0.3034
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.508 on 24 degrees of freedom
## Multiple R-squared:  0.4335, Adjusted R-squared:  0.2919
## F-statistic: 3.061 on 6 and 24 DF,  p-value: 0.02279
```

We observe in our new model, the estimate of beta_density (expected increase of porosity as density increase by 1) and intercept (expected porosity when stone type is Axeann) are significantly different in the two models , beta_density is smaller in our multiple linear regression model, and intercept is larger in multiple linear regression. Other features'estimates does not vary by a lot.

Overall, in this case, betahat of feature with small p-values, which are "good" explanatory variables on their own in simple linear regression, change less in multiple linear regression than betahat of "bad" explanatory variables. This is fishy as it seems increase in in glength/residue seems to imply same increase in porosity in both model(whether other features stays the same or not), and they are strong explanatory variable in both

model.

For t-statistics, the categorical variable type has significantly different pvals in our new multiple linear regressoin model than in our old simple linear regression model,

Bellfonte is smaller, Nittany is significantly larger, density is significantly smaller in multiple linear regression model. other feature has more or less similarly p-value in both models. glength/residue remains strongest in both model, density remains the weakest explanatory variable.

Overall, large p-values in our first model has smaller p-values in our second model. This is because those variable alone does not explain the model, but together with other feature, they could explain the model at least better.

```
beta.residue.se = summary(model)$coefficients['residue','Std. Error']
beta.residue.hat = summary(model)$coefficients['residue','Estimate']
beta.residue.ci = beta.residue.hat + c(1,-1)*beta.residue.se*qt(0.975,24)
# 95% confidence interval
beta.residue.ci
```

```
## [1] 0.204654669 0.008947842
```

we are 95% certain that true value of beta_residue falls with 0.00895 and 0.205 given a multiple linear regression model.

(d)

```
md = lm(porosity~type+glength+density+residue*glength, data=data)
summary(md)
```

```
##
## Call:
## lm(formula = porosity ~ type + glength + density + residue *
##     glength, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5524 -0.7639 -0.1415  0.5531  3.1471
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)     16.69866   17.23191   0.969   0.3426
## typeBellefonte   0.70616    0.88827   0.795   0.4348
## typeNittany     -0.03249    0.94923  -0.034   0.9730
## typeStonehenge  -0.67260    0.80689  -0.834   0.4131
## glength          0.05829    0.45497   0.128   0.8992
## density         -6.32676    6.27953  -1.008   0.3242
## residue          1.65296    0.65116   2.538   0.0184 *
## glength:residue -0.13580    0.05707  -2.380   0.0260 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.38 on 23 degrees of freedom
## Multiple R-squared:  0.5454, Adjusted R-squared:  0.4071
## F-statistic: 3.942 on 7 and 23 DF,  p-value: 0.005751
```

with increase in one unit of residue (while other feature stays the same), porosity is expected to increase by (1.653-0.136*density)

(e)

```r
me = lm(porosity~residue*glength, data=data)
summary(me)
```

```
##
## Call:
## lm(formula = porosity ~ residue * glength, data = data)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -2.5058 -0.6858 -0.0531  0.5966  3.4260
##
## Coefficients:
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.03791    4.76811   0.008  0.99371
## residue          1.52034    0.53607   2.836  0.00855 **
## glength          0.02515    0.43469   0.058  0.95429
## residue:glength -0.12364    0.04776  -2.589  0.01532 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.326 on 27 degrees of freedom
## Multiple R-squared:  0.5069, Adjusted R-squared:  0.4521
## F-statistic: 9.252 on 3 and 27 DF,  p-value: 0.0002241
```

```r
anova(md)
```

```
## Analysis of Variance Table
##
## Response: porosity
##                 Df Sum Sq Mean Sq F value  Pr(>F)
## type             3 22.482  7.4940  3.9356 0.02105 *
## glength          1  3.778  3.7777  1.9840 0.17235
## density          1  3.965  3.9654  2.0825 0.16248
## residue          1 11.540 11.5396  6.0603 0.02175 *
## glength:residue  1 10.784 10.7836  5.6632 0.02600 *
## Residuals       23 43.795  1.9041
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

```r
anova(me)
```

```
## Analysis of Variance Table
##
## Response: porosity
##                 Df Sum Sq Mean Sq F value   Pr(>F)
## residue          1 17.532 17.5324  9.9644 0.003900 **
## glength          1 19.512 19.5121 11.0895 0.002521 **
## residue:glength  1 11.792 11.7924  6.7021 0.015325 *
## Residuals       27 47.507  1.7595
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

this is sensible because residue standard error is lower for the reduced model, Also its F test has significantly smaller pval than the full model, meaning we are more certain features in the reduced model explains porosity well.

Also if we look at the ANOVA, each parameter smaller p-val in the reduced model than in the full model, and the mean sqaure residuals is also less.

H_0: beta_residue = 0, we reject H_0 at $\alpha = 0.05$ level

H_0: beta_glength = 0, we reject H_0 at $\alpha = 0.05$ level

H_0: beta_residue*glength = 0, we reject H_0 at $\alpha = 0.05$ level

conclusion: we reject null hyphothesis that beta_residue, beta_glength, and beta_residue*glength are zero individually, so they are all important explanatory feature in our multiple linear regression model (with correlated features)

4

```
original.data = read.csv("satisfaction.csv")
data = original.data[1:20,]
set.seed(2070480)
simuY = -2+0*data$Age+0.2*data$Severity+1*data$Stress+rnorm(20,mean=0,sd=2)
data$Satisfaction2 = simuY

model = lm(Satisfaction2~Age+Severity+Stress, data=data)
s=summary(model)
df = model$df.residual
age_t =s$coefficients['Age', 't value']
pval_age = 1-pt(age_t, df)
# note this is Pr(T>t)
pval_age
```

```
## [1] 0.8741149
```

```
severity_t =s$coefficients['Severity', 't value']
pval_severity= 1-pt(severity_t, df)
pval_severity
```

```
## [1] 0.489097
```

we do not reject null hypothesis beta_age == 0

we do not reject null hypothesis beta_severity == 0

(v)

```
age_reject_count = 0
severity_reject_count = 0

for (i in 1:4000){
  simuY = -2+0*data$Age+0.2*data$Severity+1*data$Stress+rnorm(20,mean=0,sd=2)
  data$Satisfaction2 = simuY

  model = lm(Satisfaction2~Age+Severity+Stress, data=data)
  s=summary(model)
  age_t =s$coefficients['Age', 't value']
  age_pval = 1-pt(age_t, df)

  severity_t =s$coefficients['Severity', 't value']
  severity_pval = 1-pt(severity_t, df)

  if (age_pval <0.05){
    age_reject_count = age_reject_count + 1
  }
  if (severity_pval <0.05){
    severity_reject_count = severity_reject_count + 1
  }
}

A1=age_reject_count/4000
A2=severity_reject_count/4000
A1
```

```
## [1] 0.05775
```

```
A2
```

```
## [1] 0.3045
```

B

```
age_reject_count = 0
severity_reject_count = 0
data=original.data

for (i in 1:4000){
  simuY = -2+0*data$Age+0.2*data$Severity+1*data$Stress+rnorm(46,mean=0,sd=2)
  data$Satisfaction2 = simuY

  model = lm(Satisfaction2~Age+Severity+Stress, data=data)
  s=summary(model)

  age_t =s$coefficients['Age', 't value']
  age_pval = 1-pt(age_t, df)

  severity_t =s$coefficients['Severity', 't value']
  severity_pval = 1-pt(severity_t, df)


  if (age_pval <0.05){
    age_reject_count = age_reject_count + 1
  }
  if (severity_pval <0.05){
    severity_reject_count = severity_reject_count + 1
  }
}

B1=age_reject_count/4000
B2=severity_reject_count/4000
B1
```

```
## [1] 0.0425
```

```
B2
```

```
## [1] 0.6115
```

```
age_reject_count = 0
severity_reject_count = 0
data=original.data[1:20,]

for (i in 1:4000){
  simuY = -2+0*data$Age+0.2*data$Severity+1*data$Stress+rnorm(20,mean=0,sd=1)
  data$Satisfaction2 = simuY

  model = lm(Satisfaction2~Age+Severity+Stress, data=data)
  s=summary(model)


  age_t =s$coefficients['Age', 't value']
  age_pval = 1-pt(age_t, df)
```

```
  severity_t =s$coefficients['Severity', 't value']
  severity_pval = 1-pt(severity_t, df)


  if (age_pval <0.05){
    age_reject_count = age_reject_count + 1
  }
  if (severity_pval <0.05){
    severity_reject_count = severity_reject_count + 1
  }
}

C1=age_reject_count/4000
C2=severity_reject_count/4000

C1
```

```
## [1] 0.05275
```

```
C2
```

```
## [1] 0.75925
```

```
age_reject_count = 0
severity_reject_count = 0
data=original.data

for (i in 1:4000){
  simuY = -2+0*data$Age+0.2*data$Severity+1*data$Stress+rnorm(46,mean=0,sd=1)
  data$Satisfaction2 = simuY

  model = lm(Satisfaction2~Age+Severity+Stress, data=data)
  s=summary(model)

  age_t =s$coefficients['Age', 't value']
  age_pval = 1-pt(age_t, df)

  severity_t =s$coefficients['Severity', 't value']
  severity_pval = 1-pt(severity_t, df)

  if (age_pval <0.05){
    age_reject_count = age_reject_count + 1
  }
  if (severity_pval <0.05){
    severity_reject_count = severity_reject_count + 1
  }
}

D1=age_reject_count/4000
D2=severity_reject_count/4000

D1
```

```
## [1] 0.04375
```

```
D2
```

```
## [1] 0.98925
```

```
matr = matrix(c(C1, D1,A1 ,B1),nrow = 2, ncol = 2)
dimnames(matr)= list(c("numsim=20","numsim=46"), c("sigma=1","sigma=2"))
# for beta 1, the percentage of null hypothesis that got rejected is:
matr
```

```
##           sigma=1 sigma=2
## numsim=20 0.05275 0.05775
## numsim=46 0.04375 0.04250
```

```
library(knitr)
kable(matr, caption = "beta 1, fraction of time H0 rejected")
```

Table 1: beta 1, fraction of time H0 rejected

	sigma=1	sigma=2
numsim=20	0.05275	0.05775
numsim=46	0.04375	0.04250

```
matr = matrix(c(C2, D2, A2,B2),nrow = 2, ncol = 2)
dimnames(matr)= list(c("numsim=20","numsim=46"), c("sigma=1","sigma=2"))
# for beta 2, the percentage of null hypothesis that got rejected is:
kable(matr, caption = "beta 2, fraction of time H0 rejected")
```

Table 2: beta 2, fraction of time H0 rejected

	sigma=1	sigma=2
numsim=20	0.75925	0.3045
numsim=46	0.98925	0.6115

we calculate standard deviation for beta1 and beta2 to get a clearer understanding

```
m=lm(Satisfaction~Age+Severity+Stress, data=original.data)
X=model.matrix(m)
#variance of beta_1 when n=46 and sigma=1,2
c(1,2)*solve(t(X)%*%X)[2,2]
```

```
## [1] 0.0004560816 0.0009121633
```

```
#variance of beta_2 when n=46 and sigma=1,2
c(1,2)*solve(t(X)%*%X)[3,3]
```

```
## [1] 0.002392481 0.004784963
```

```
m=lm(Satisfaction~Age+Severity+Stress, data=original.data[1:20,])
X=model.matrix(m)
#variance of beta_1 when n=20 and sigma=1,2
c(1,2)*solve(t(X)%*%X)[2,2]
```

```
## [1] 0.001001342 0.002002684
```

```r
#variance of beta_2 when n=20 and sigma=1,2
c(1,2)*solve(t(X)%*%X)[3,3]
```

```
## [1] 0.006562683 0.013125365
```

we observe it's easier to reject $\beta_2 \leq 0$ than to reject $\beta_1 \leq 0$, this is because beta2 actually > 0, and its variance, especially when n is large, forbids it from going under 0.

we also observe pval for $\beta_1 \leq 0$ is some constant around 0.1, whereas pval for $\beta_2 \leq 0$ differs as n and σ change.

This is because true CI for $\hat{\beta}_1$ always covers 0, while true CI for $\hat{\beta}_2$ only covers 0 when its confidence interval is large enough (aka, when sigma is large). the CI for β_2 also gets shorter as n decrease, making it harder to cover 0.

we also observe as n increase and alpha decrease, we get more accurate result on $\hat{\beta}_2$. this is reasonable as the confidence interval becomes smaller, it's easier to reject H_0