Bias, Efficiency, and the Gauss Markov Theorem

Ben Gilbert

9/1/2021

Bias, Efficiency, and the Gauss Markov Theorem

- This is a program to illustrate bias and inefficiency when Gauss Markov assumptions fail
  - Bias: deviation of expected value of sample parameter estimate from "true" population parameter
  - Efficiency: the variance of the sample parameter estimate should be as small as possible
  - Consistency: distribution of sample parameter estimate should converge to population value as sample size grows
- Gauss-Markov: OLS (Ordinary Least Squares) is Best (lowest variance) Linear Unbiased Estimator (BLUE) IF:
  1. true model linear in parameters and residuals:
  $$y_t = \beta_0 + \beta_1 * x_{1t} + \beta_2 * x_{2t} + e_t$$
  2. X variables (right hand side) are not constants or perfectly correlated with each other
  3. Residuals "e" have constant variance
  - (not more noisy for some X's than others, homoskedasticity vs. heteroskedasticy)
  4. Residuals "e" are uncorrelated with each other
  - (no peer effects, no serial correlation)
  5. All X variables are uncorrelated with the residual e
  - (observed X is not picking up some unobserved or uncontrolled factor)
- In each case we will run the linear regression
  $$y_t = \beta_0 + \beta_1 * x_{1t} + \beta_2 * x_{2t} + e_t$$
  on the data, but the "true" model or "data generating process" is different

---

Define directory and load packages

Load (and install if necessary) any packages that we want to use. If we want to define a working directory for this session, we can.

```
setwd("C:/Users/bgilbert_a/Dropbox/Econometrics/TimeSeriesCourse/Fall2021")
# install.packages("MASS")
require(MASS)
# install.packages("car")
require(car)
```

Violate Assumption 1: True model is not linear in parameters

- Set the seed for replicability.
- Generate the true model

```
set.seed(826)
# Covariance matrix of x1, x2, and e
Sig <- matrix(c(4,1,0,1,2,0,0,0,1),3,3)
Sig       # Notice e does not covary with x1 or x2 (assumption 5)
```

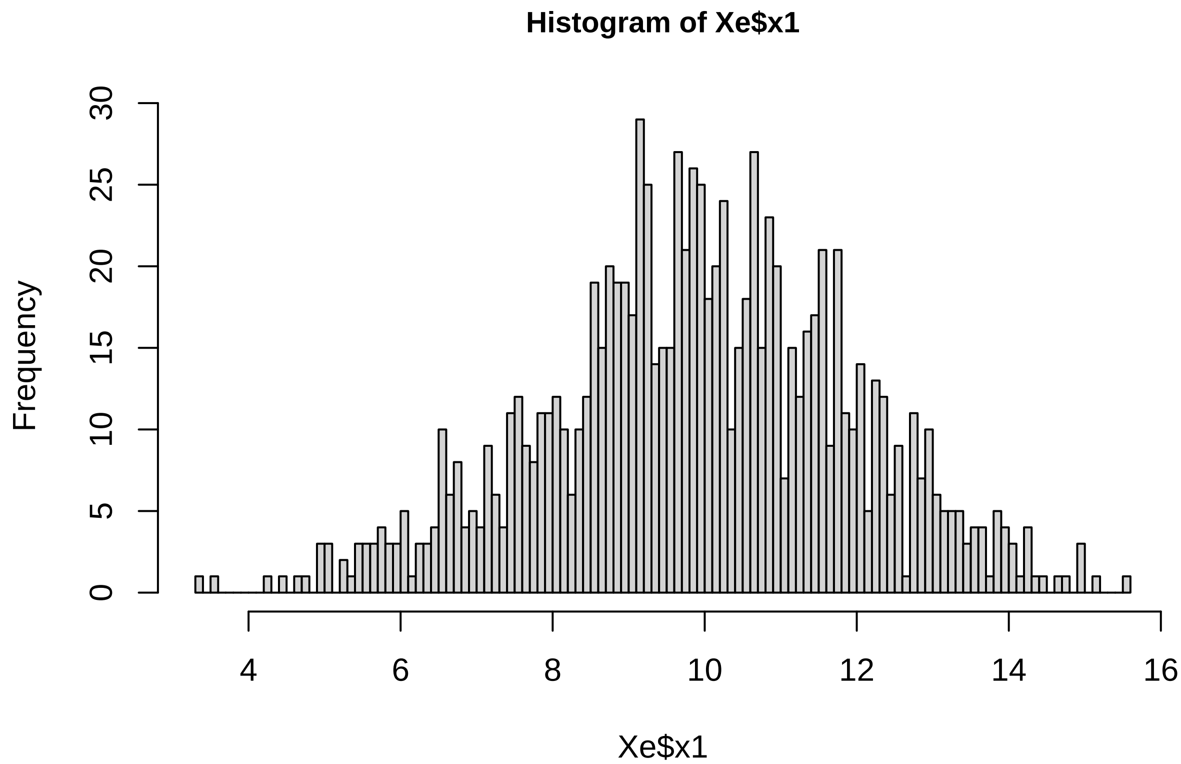
```
##      [,1] [,2] [,3]
## [1,]    4    1    0
## [2,]    1    2    0
## [3,]    0    0    1
```

```
          # also x1 and x2 can covary, but not perfectly (assumption 2)

# Mean of x1, x2, and e
moo <- c(10,3,0)

# generate data
Xe <- mvrnorm(n=1000,mu=moo,Sigma=Sig)

# give the variables names
colnames(Xe)<-c("x1","x2","e")

# store as a data frame
Xe <- as.data.frame(Xe)
head(Xe)
```

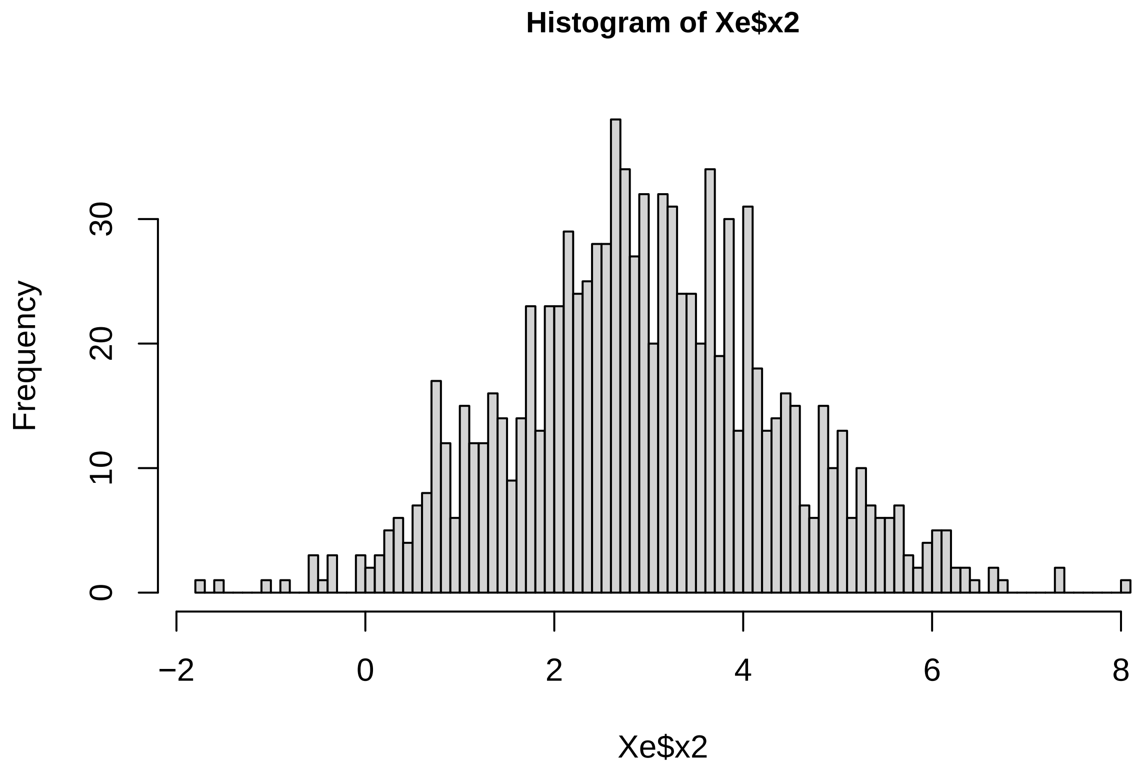
```
##          x1        x2          e
## 1  9.815559 2.8349325  1.0499626
## 2  8.443130 2.1208185  0.7884180
## 3  9.618013 3.3454521 -0.8777746
## 4  6.371419 1.7631024  0.6244005
## 5 10.091449 0.6709607  0.2481989
## 6  5.828882 1.1333412  0.4228365
```

- Investigate the data.
  - plot empirical distribution of each variable

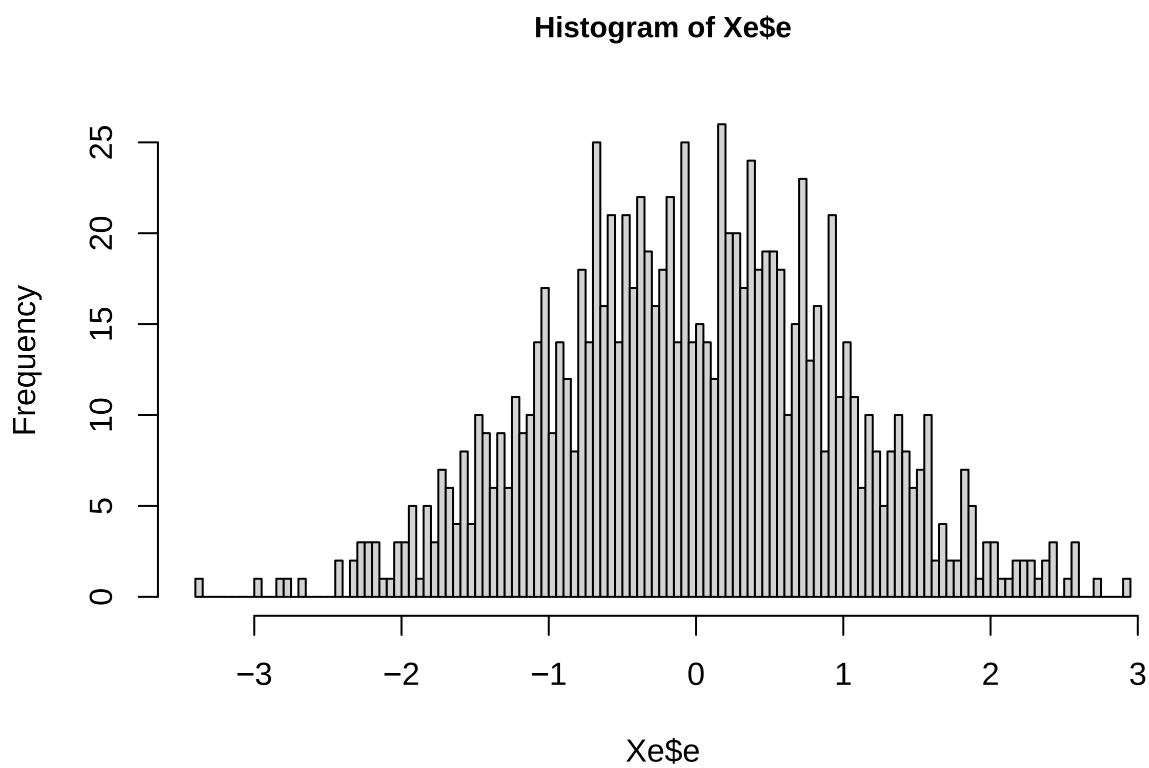
```
hist(Xe$x1, breaks = 100, cex.main = 0.9)
```




```
hist(Xe$x2, breaks = 100, cex.main = 0.9)
```




```
hist(Xe$e, breaks = 100, cex.main = 0.9)
```



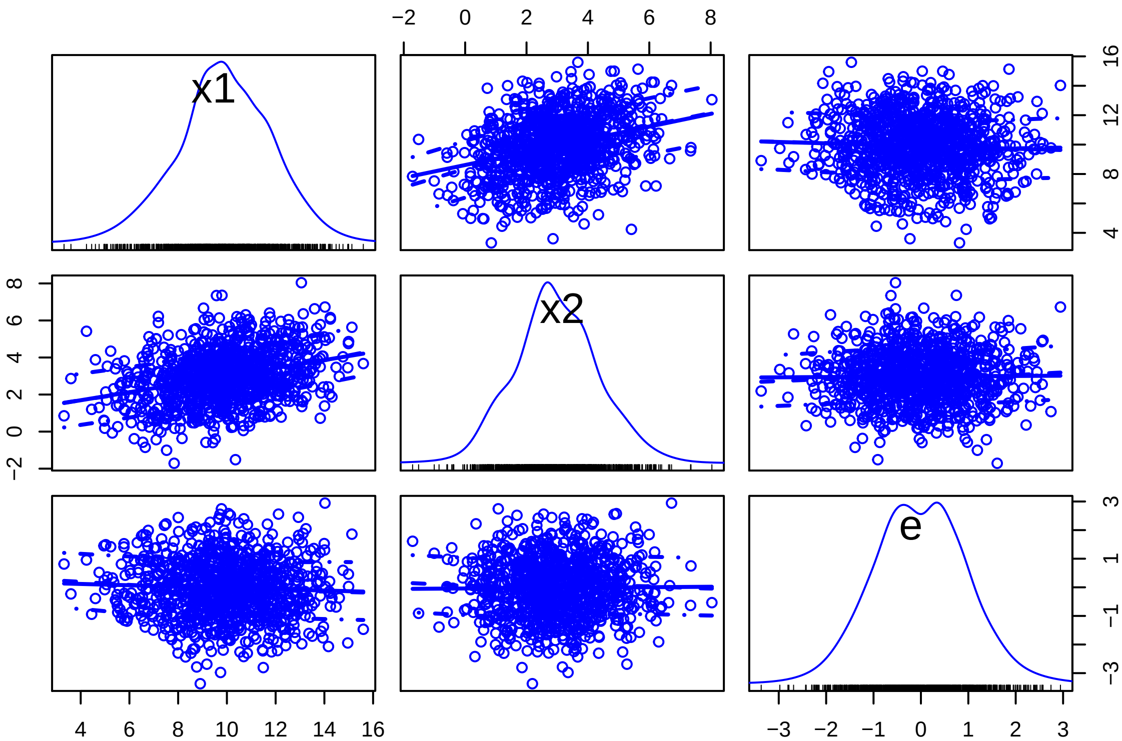

```r
# Sample correlations and covariances (notice difference from "truth")
cov(Xe)
```

```
##              x1         x2           e
## x1  4.02081193 0.87289399 -0.09254668
## x2  0.87289399 2.00633796  0.01490771
## e  -0.09254668 0.01490771  1.00873015
```

```r
cor(Xe)
```

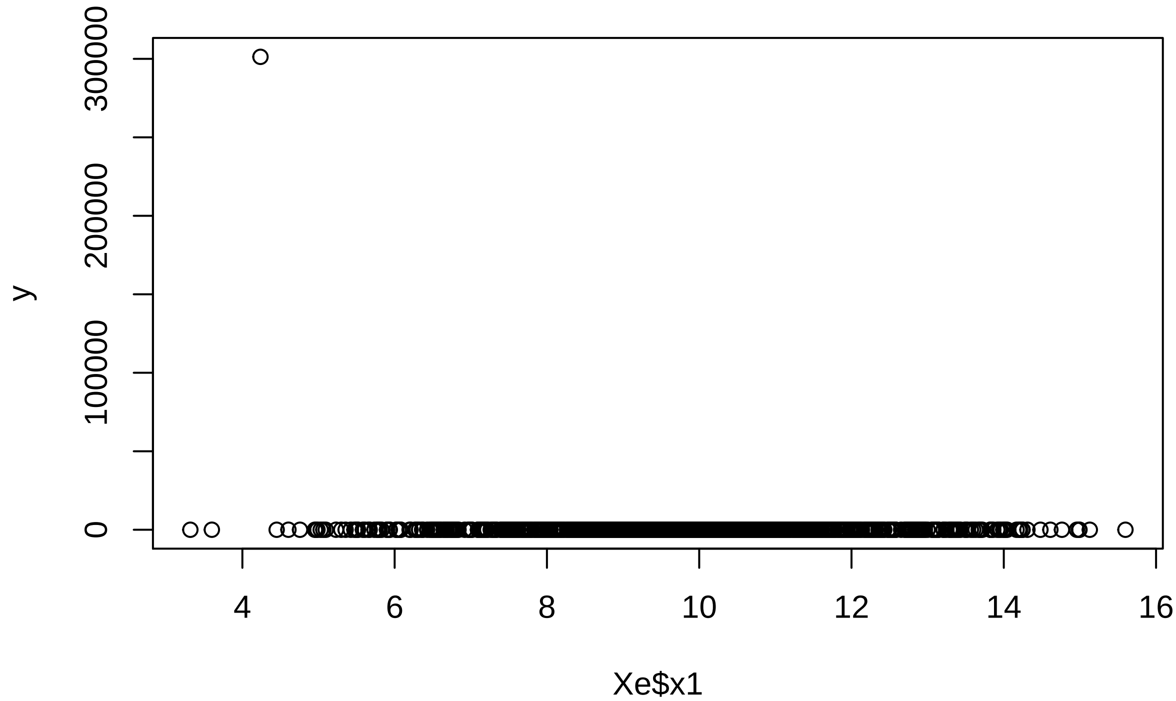
```
##              x1         x2           e
## x1  1.00000000 0.30732832 -0.04595327
## x2  0.30732832 1.00000000  0.01047904
## e  -0.04595327 0.01047904  1.00000000
```

```r
scatterplotMatrix(Xe)
```

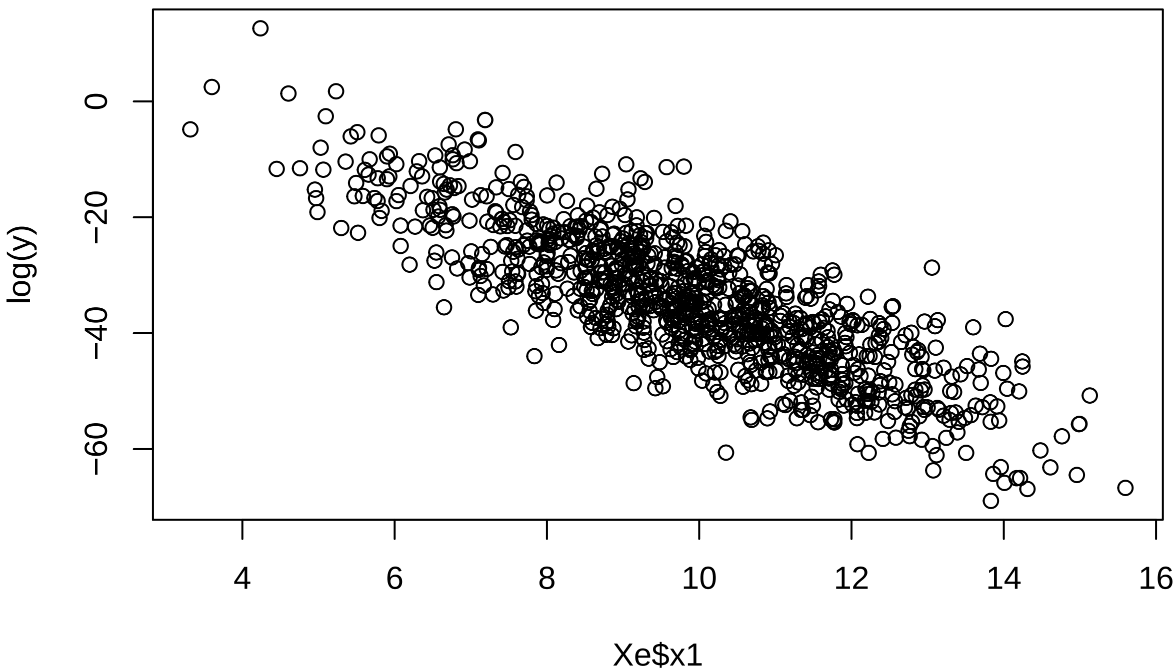


- Generate the true model for outcome y

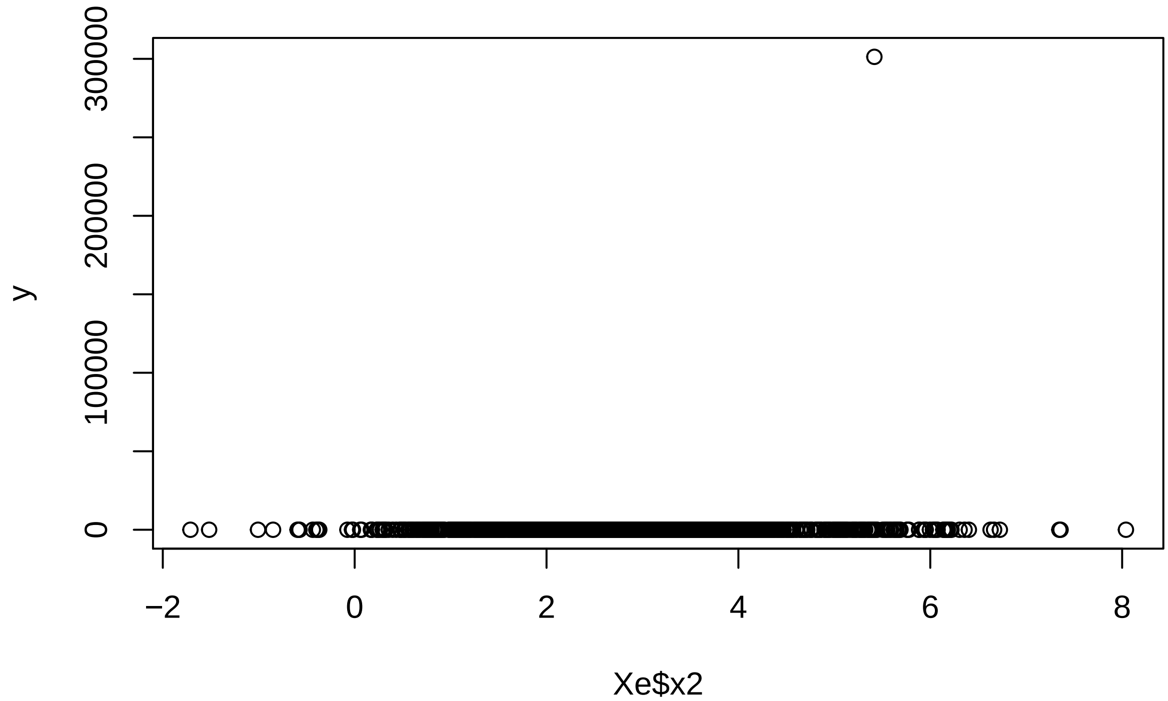
```
y <- exp(10 - 6*Xe$x1 + 5*Xe$x2 + Xe$e) # log(y) is linear in parameters and
                                        # residual, but y is not.
plot(Xe$x1,y)
```



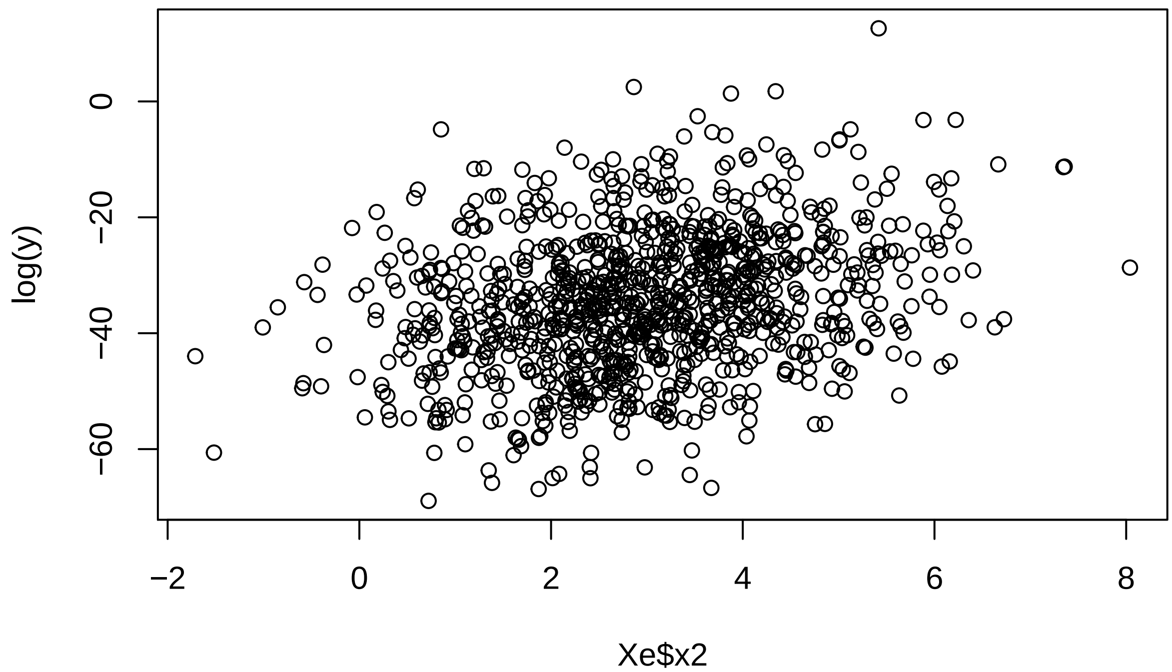

```
plot(Xe$x1,log(y))
```




```
plot(Xe$x2,y)
```




```
plot(Xe$x2,log(y))
```



- Run the linear regression when the true model is linear vs not linear.

```
summary(lm(y~x1+x2,data=Xe))
```

```
##
## Call:
## lm(formula = y ~ x1 + x2, data = Xe)
##
## Residuals:
##    Min     1Q Median     3Q    Max
##  -3802  -1104   -264    495 296312
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   4009.1     1519.8   2.638 0.008470 **
## x1            -557.4      156.9  -3.553 0.000399 ***
## x2             609.8      222.1   2.746 0.006148 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 9462 on 997 degrees of freedom
## Multiple R-squared:  0.01545,    Adjusted R-squared:  0.01347
## F-statistic: 7.822 on 2 and 997 DF,  p-value: 0.000426
```

```
summary(lm(log(y)~x1+x2,data=Xe))
```

```
##
## Call:
```

```
## lm(formula = log(y) ~ x1 + x2, data = Xe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3647 -0.6814 -0.0060  0.6925  3.0024
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.19399    0.16125   63.22   <2e-16 ***
## x1          -6.02720    0.01665 -362.07   <2e-16 ***
## x2           5.01926    0.02357  212.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.004 on 997 degrees of freedom
## Multiple R-squared:  0.9931, Adjusted R-squared:  0.993
## F-statistic: 7.126e+04 on 2 and 997 DF,  p-value: < 2.2e-16
```

Violate Assumption 2: Right hand side variables are constants or are perfectly correlated

- NOTE: The intercept is a constant, but no other variables can be constant
  - otherwise perfectly correlated with the intercept
- Set the seed for replicability.
- Generate the true model

```
set.seed(826)
# Covariance matrix of x1 and e
Sig <- matrix(c(4,0,0,1),2,2)
Sig       # Notice e does not covary with x1 (assumption 5)
```

```
##      [,1] [,2]
## [1,]    4    0
## [2,]    0    1
```

```
# We will make x2 an exact multiple of x1 (violate assumption 2)
# Mean of x1 and e
moo <- c(10,0)
# generate data
Xe <- mvrnorm(n=1000,mu=moo,Sigma=Sig)
# generate x2 as arbitrary multiple of x1
x2 <- 7*Xe[,1]
Xe <- as.data.frame(cbind(Xe[,1],x2,Xe[,2]))
# give the variables names
colnames(Xe)<-c("x1","x2","e")
head(Xe)
```

```
##         x1        x2           e
## 1 10.22234  71.55639 -0.06505297
## 2 11.68949  81.82641 -0.17189937
## 3 10.21010  71.47071  0.36952559
## 4 13.64180  95.49257  0.19523319
## 5 10.76801  75.37608 -1.73650622
## 6 14.34835 100.43846 -0.10192330
```

- Investigate sample correlations and covariances
  - notice difference from "truth"

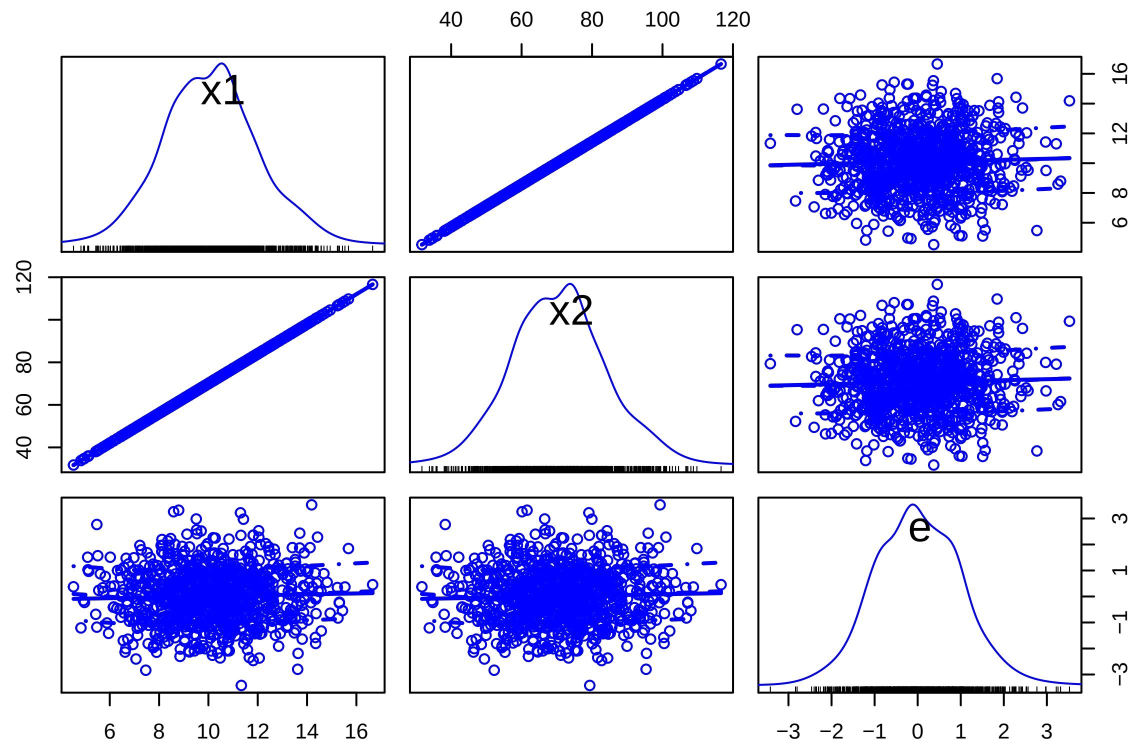
```
cov(Xe)
```

```
##              x1          x2          e
## x1  3.93549450  27.5484615 0.07180935
## x2 27.54846153 192.8392307 0.50266547
## e   0.07180935   0.5026655 1.06201033
```

```
cor(Xe)
```

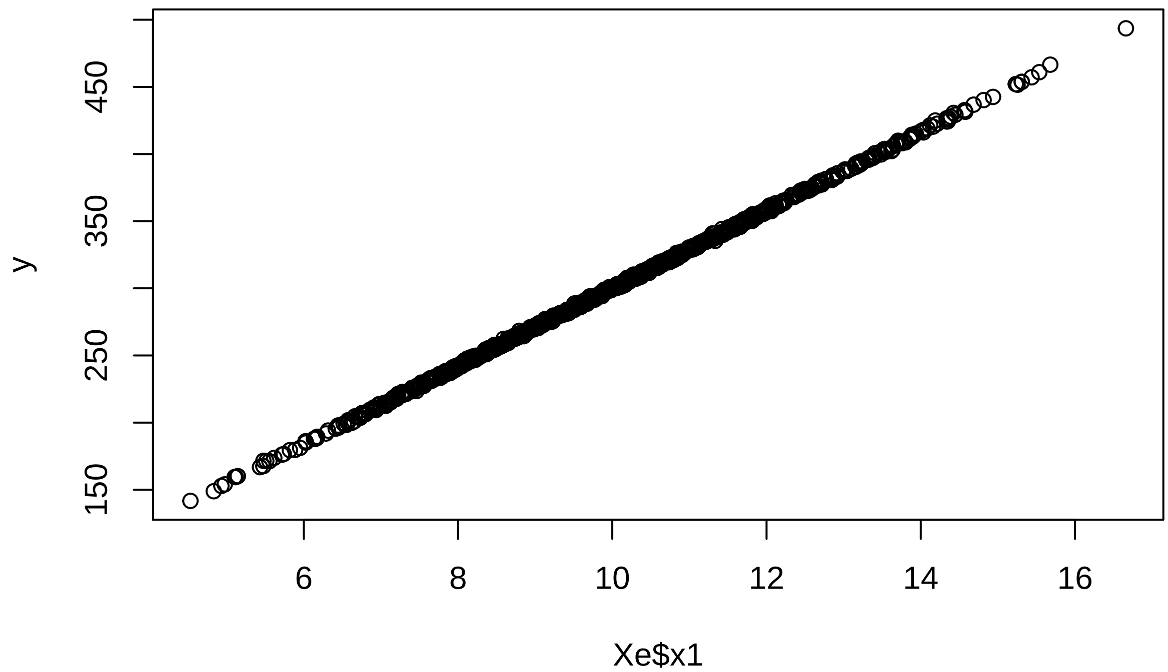
```
##            x1         x2          e
## x1 1.00000000 1.00000000 0.03512505
## x2 1.00000000 1.00000000 0.03512505
## e  0.03512505 0.03512505 1.00000000
```

```
scatterplotMatrix(Xe)
```

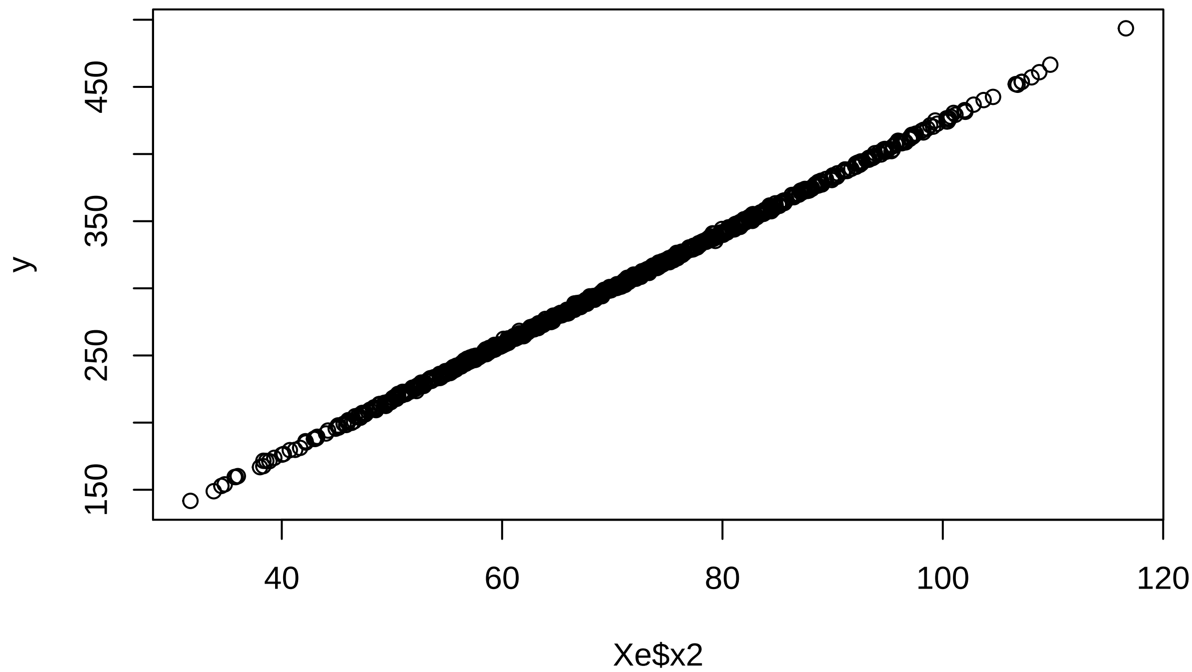


- Generate the true y (outcome)

```
y <- 10 - 6*Xe$x1 + 5*Xe$x2 + Xe$e # y is linear in parameters (assumption 1)
plot(Xe$x1,y)
```




```
plot(Xe$x2,y) # Should be a tight fit!
```



- Run the linear regression

```
summary(lm(y~x1+x2,data=Xe)) # Why an intercept of 10 and coefficient of 29?
```

```
##
## Call:
## lm(formula = y ~ x1 + x2, data = Xe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4532 -0.7331 -0.0450  0.7074  3.4402
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.82550    0.16904   58.12   <2e-16 ***
## x1          29.01825    0.01643 1765.79   <2e-16 ***
## x2                NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.03 on 998 degrees of freedom
## Multiple R-squared:  0.9997, Adjusted R-squared:  0.9997
## F-statistic: 3.118e+06 on 1 and 998 DF,  p-value: < 2.2e-16
```

```
                              # -6 + 5*7 = 29
                              # some stats packages will not produce output
```

```r
# NOTE: regressing on an intercept ONLY estimates the mean of y - intercept can
# be constant.
mean(y)
```

```
## [1] 302.725
```

```r
summary(lm(y~1))
```

```
##
## Call:
## lm(formula = y ~ 1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -160.99  -39.38   -0.12   35.42  190.88
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  302.725      1.821   166.3   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 57.58 on 999 degrees of freedom
```

- But what if another regressor is also constant?

```r
set.seed(826)
# Covariance matrix of x1, x2, and e
Sig <- matrix(c(4,0,0,0,0,0,0,0,1),3,3)
Sig       # Notice e does not covary with x1 or x2 (assumption 5)
```

```
##      [,1] [,2] [,3]
## [1,]    4    0    0
## [2,]    0    0    0
## [3,]    0    0    1
```

```r
          # But x2 also has no variance (violate assumption 2)
# Mean of x1, x2, and e
moo <- c(10,3,0)
# generate data
Xe <- mvrnorm(n=1000,mu=moo,Sigma=Sig)
# give the variables names
colnames(Xe)<-c("x1","x2","e")
Xe <- as.data.frame(Xe)
head(Xe)
```

```
##         x1 x2           e
## 1 9.777659  3  0.06505297
## 2 8.310513  3  0.17189937
## 3 9.789899  3 -0.36952559
## 4 6.358205  3 -0.19523319
## 5 9.231989  3  1.73650622
## 6 5.651648  3  0.10192330
```

```r
# Sample correlations and covariances (notice difference from "truth")
cov(Xe)
```

```
##              x1 x2          e
## x1 3.93549450  0 0.07180935
## x2 0.00000000  0 0.00000000
## e  0.07180935  0 1.06201033
```

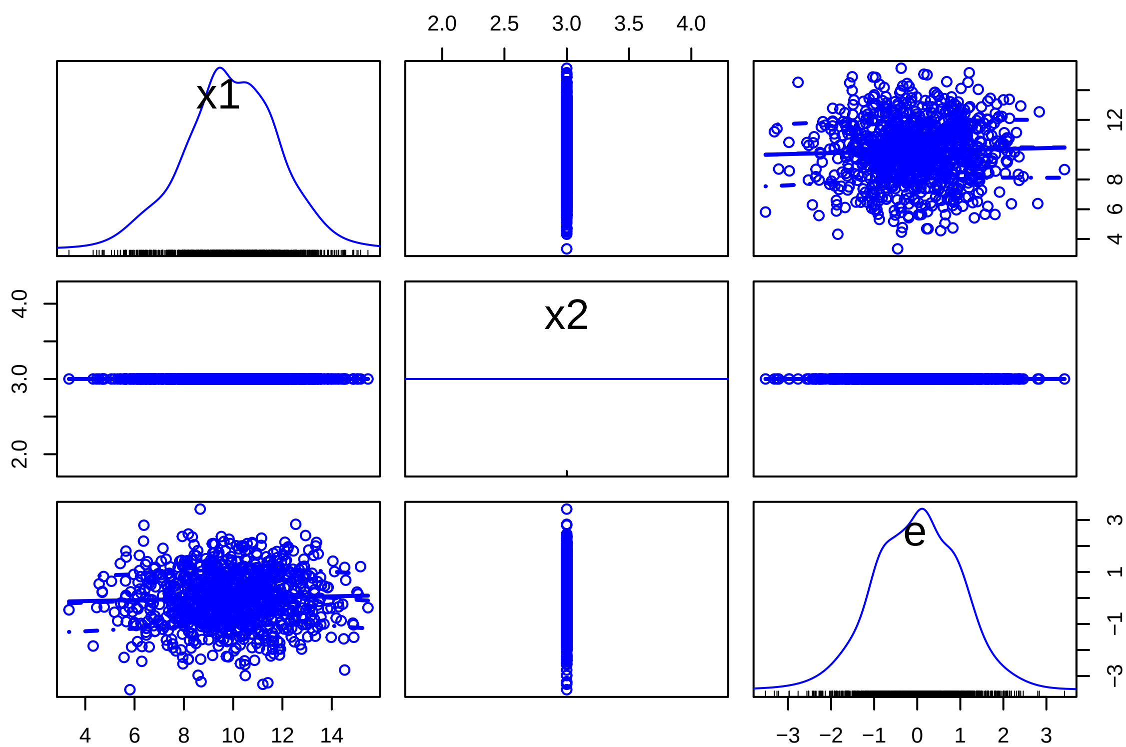
```
cor(Xe)
```

```
## Warning in cor(Xe): the standard deviation is zero
```

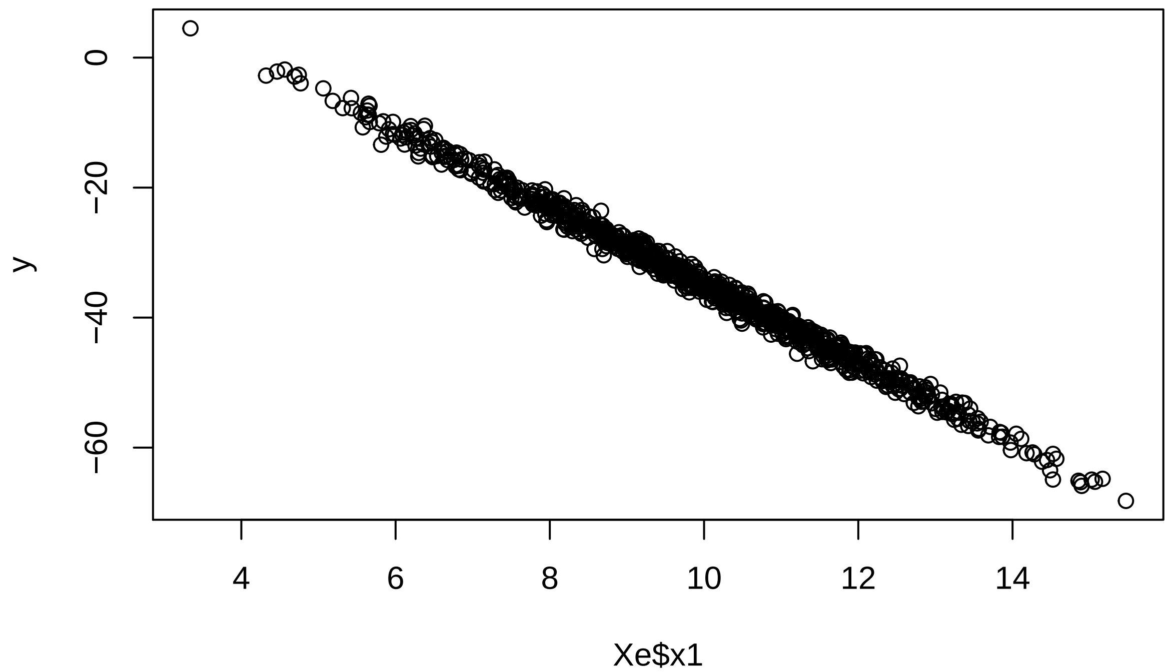
```
##              x1 x2          e
## x1 1.00000000 NA 0.03512505
## x2         NA  1         NA
## e  0.03512505 NA 1.00000000
```

```
scatterplotMatrix(Xe)
```

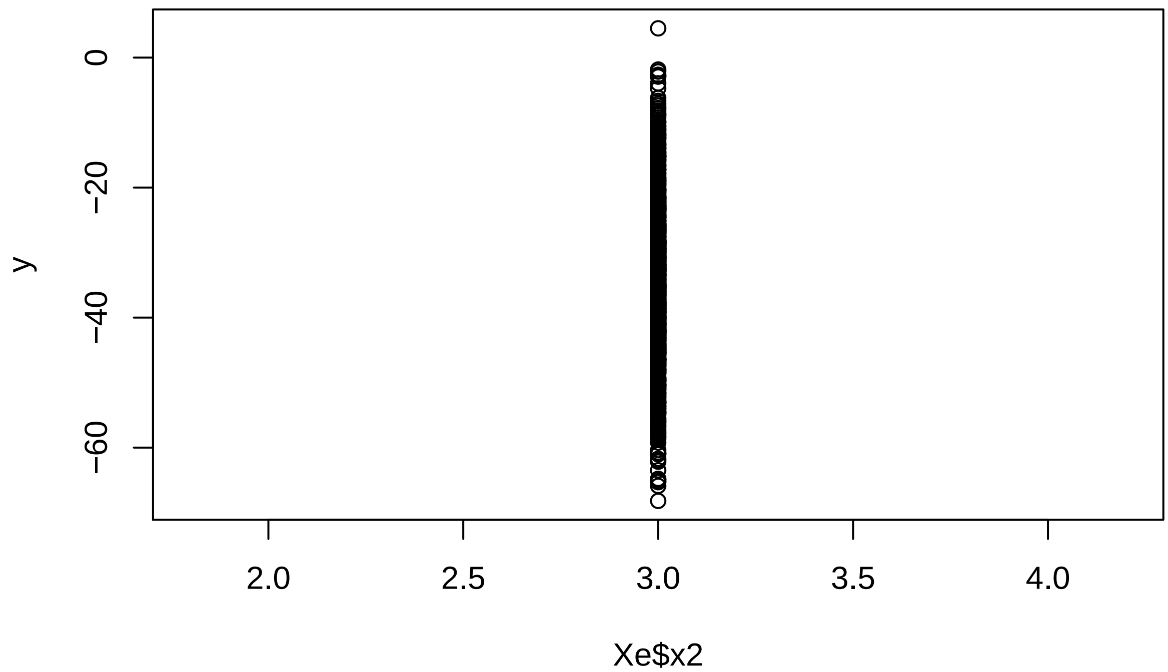
```
## Warning in smoother(x[subs], y[subs], col = smoother.args$col[i], log.x =
## FALSE, : could not fit negative part of the spread
```




```
# generate the true y (outcome)
y <- 10 - 6*Xe$x1 + 5*Xe$x2 + Xe$e # y is linear in parameters (assumption 1)
plot(Xe$x1,y)
```




```
plot(Xe$x2,y) # Should be a tight fit!
```




```
# run the linear regression
summary(lm(y~x1+x2,data=Xe)) # Why an intercept of 25 and coefficient of -6?
```

```
##
## Call:
## lm(formula = y ~ x1 + x2, data = Xe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.4402 -0.7074  0.0450  0.7331  3.4532
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 24.80957    0.16603   149.4   <2e-16 ***
## x1          -5.98175    0.01643  -364.0   <2e-16 ***
## x2                NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.03 on 998 degrees of freedom
## Multiple R-squared:  0.9925, Adjusted R-squared:  0.9925
## F-statistic: 1.325e+05 on 1 and 998 DF,  p-value: < 2.2e-16
```

```
                             # 10 + 5*3
                             # some stats packages will not even produce output
```

Violate Assumption 3: Residuals "e" do not have constant variance

- Residuals are "heteroskedastic" (different variance)
  - Variance changes at different places in the population
  - For different values of X or different time periods t
- Set the seed and generate the true model

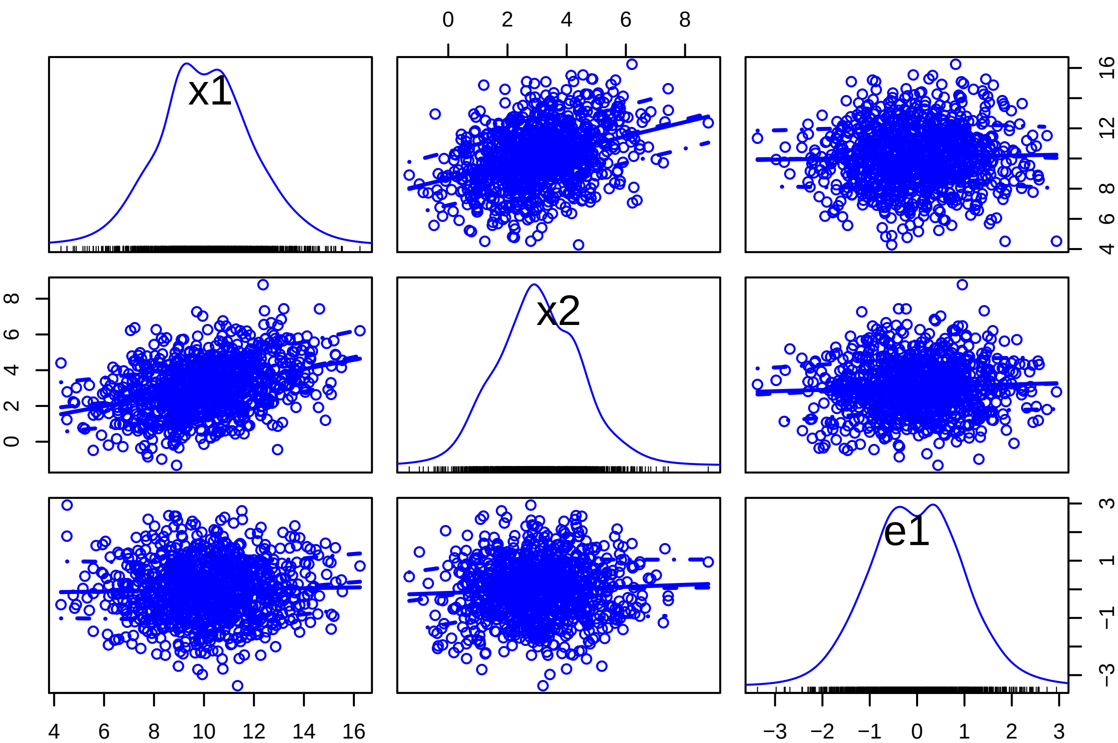
```
set.seed(826)
# Covariance matrix of x1, x2
Sig <- matrix(c(4,1,1,2),2,2)
Sig
```

```
##      [,1] [,2]
## [1,]    4    1
## [2,]    1    2
```

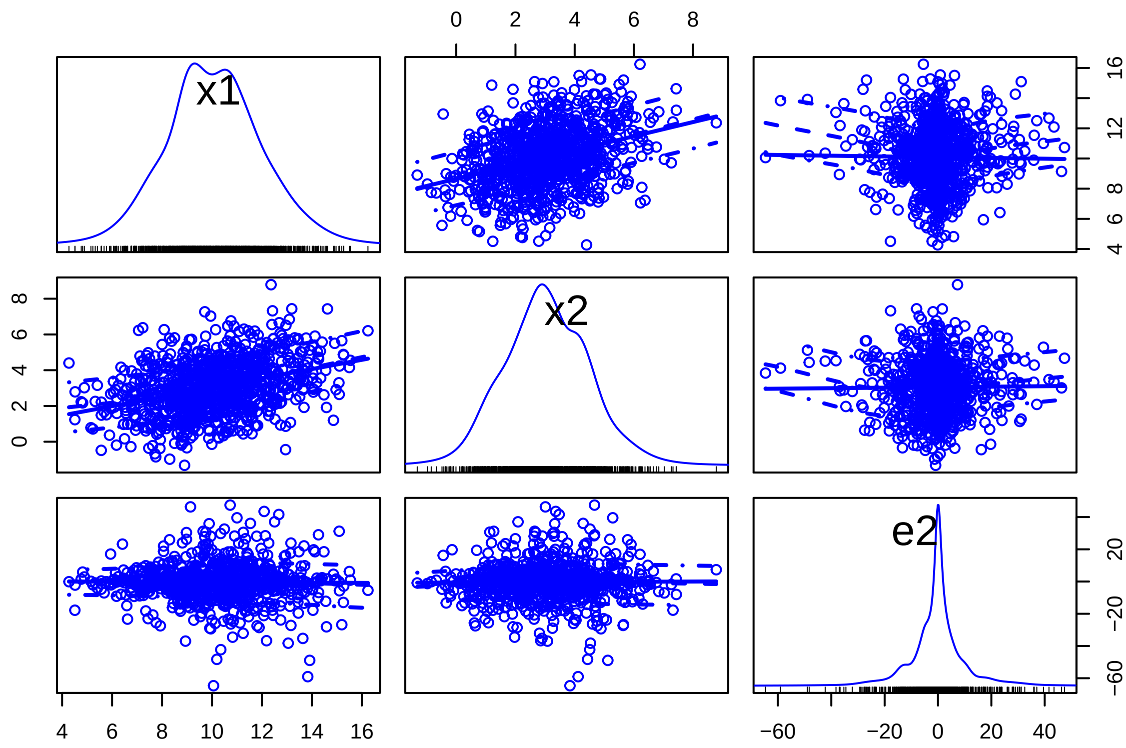
```
# Mean of x1, x2
moo <- c(10,3)
# generate data
Xe <- mvrnorm(n=1000,mu=moo,Sigma=Sig)

# generate homoskedastic residuals
eps = rnorm(n=1000,mean=0,sd=sqrt(1))

# generate heteroskedastic residuals
sigma2 = (eps^2)*(Xe[,1]^2+Xe[,2]^2)
eps2 = rnorm(n=1000,mean=0,sd=sqrt(sigma2))

Xe1 = as.data.frame(cbind(Xe,eps))
colnames(Xe1)<-c("x1","x2","e1")
Xe2 = as.data.frame(cbind(Xe,eps2))
colnames(Xe2)<-c("x1","x2","e2")
```

- Investigate sample correlations and covariances
  - notice difference from "truth"

```
scatterplotMatrix(Xe1)
```




```
scatterplotMatrix(Xe2)
```

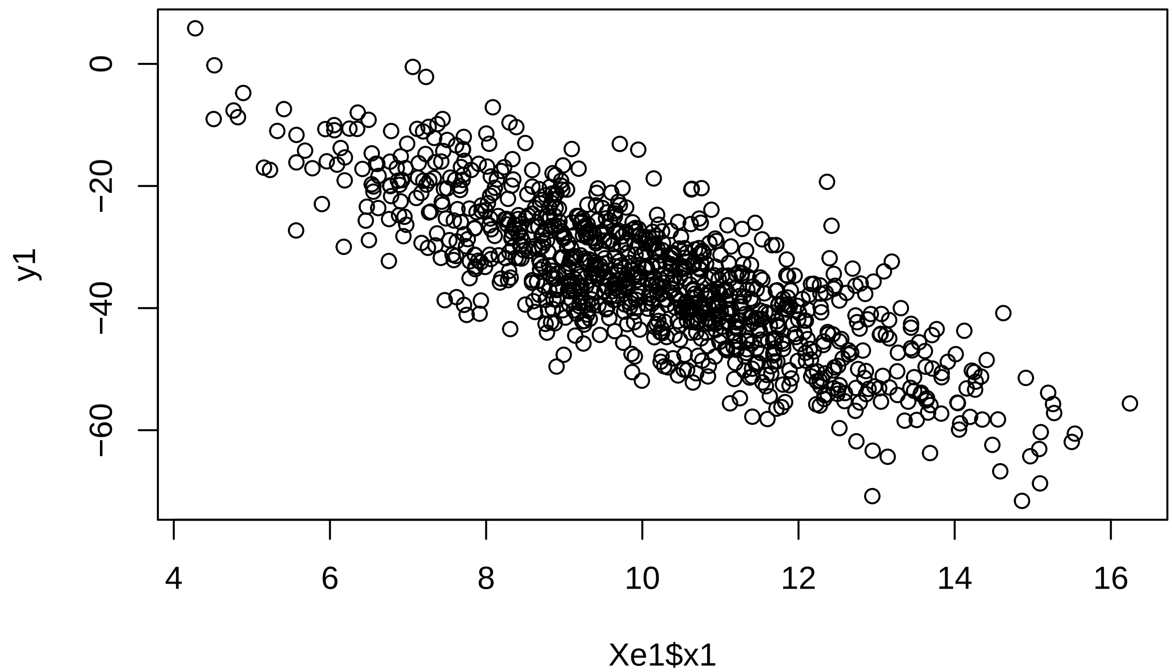


- Generate the true y (outcome)

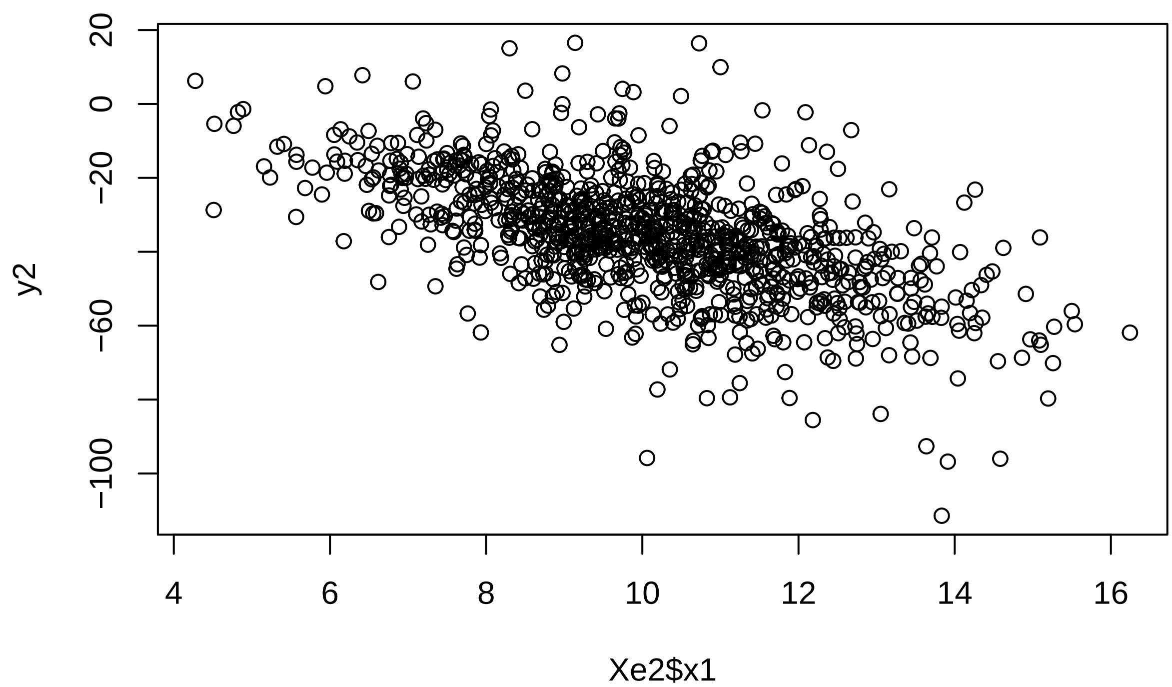
```
y1 <- 10 - 6*Xe1$x1 + 5*Xe1$x2 + Xe1$e1
y2 <- 10 - 6*Xe2$x1 + 5*Xe2$x2 + Xe2$e2
```

- Notice there are not necessarily obvious differences in the plot

```
plot(Xe1$x1,y1)
```




```
plot(Xe2$x1,y2)
```



- Run the linear regression
  - notice the difference in residual standard error, coefficient std. error, Rsquared.

```
summary(lm(y1~x1+x2,data=Xe1))
```

```
##
## Call:
## lm(formula = y1 ~ x1 + x2, data = Xe1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3647 -0.6814 -0.0060  0.6925  3.0024
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.82512    0.16589   59.23   <2e-16 ***
## x1          -5.99439    0.01719 -348.62   <2e-16 ***
## x2           5.03285    0.02317  217.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.004 on 997 degrees of freedom
## Multiple R-squared:  0.9925, Adjusted R-squared:  0.9925
## F-statistic: 6.596e+04 on 2 and 997 DF,  p-value: < 2.2e-16
```

```
summary(lm(y2~x1+x2,data=Xe2))
```

```
##
## Call:
## lm(formula = y2 ~ x1 + x2, data = Xe2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -64.361  -4.232   0.309   3.882  47.679
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  10.3175     1.8010   5.729 1.34e-08 ***
## x1           -6.1093     0.1867 -32.726  < 2e-16 ***
## x2            5.1291     0.2515  20.391  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 10.9 on 997 degrees of freedom
## Multiple R-squared:  0.5383, Adjusted R-squared:  0.5374
## F-statistic: 581.2 on 2 and 997 DF,  p-value: < 2.2e-16
```

Violate Assumption 4: Residuals "e" are serially correlated (autocorrelated)

- Set the seed and generate the true model

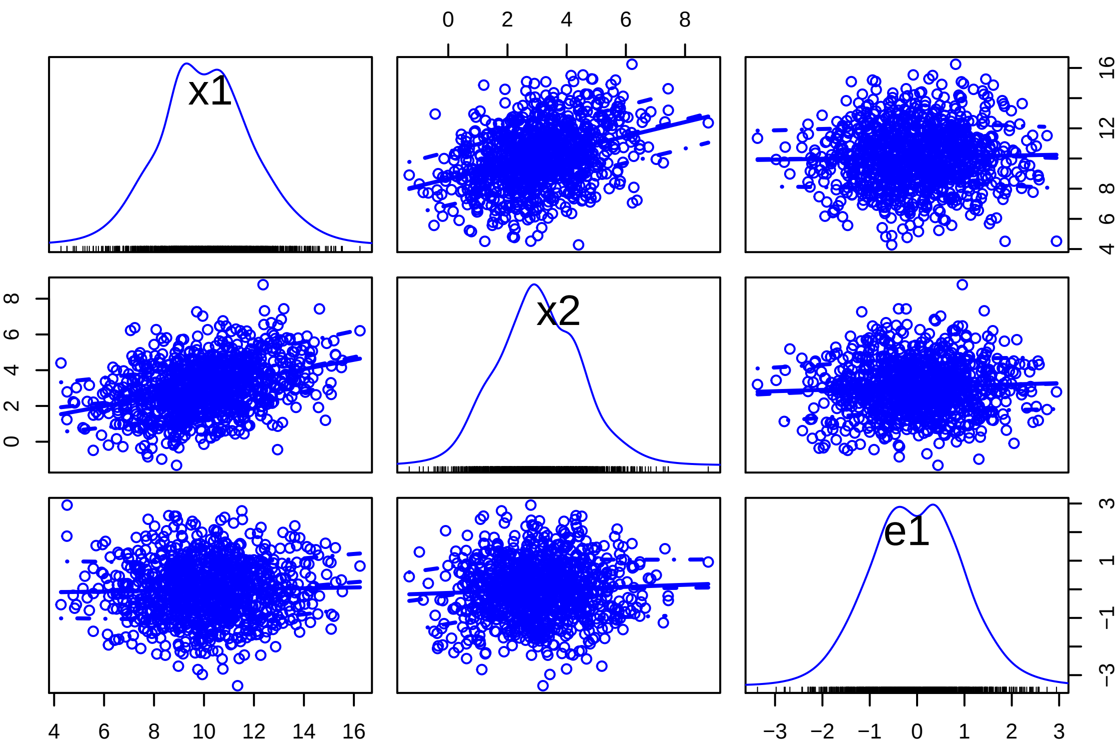
```
set.seed(826)
# Covariance matrix of x1, x2
Sig <- matrix(c(4,1,1,2),2,2)
Sig
```

```
##      [,1] [,2]
## [1,]    4    1
## [2,]    1    2
```

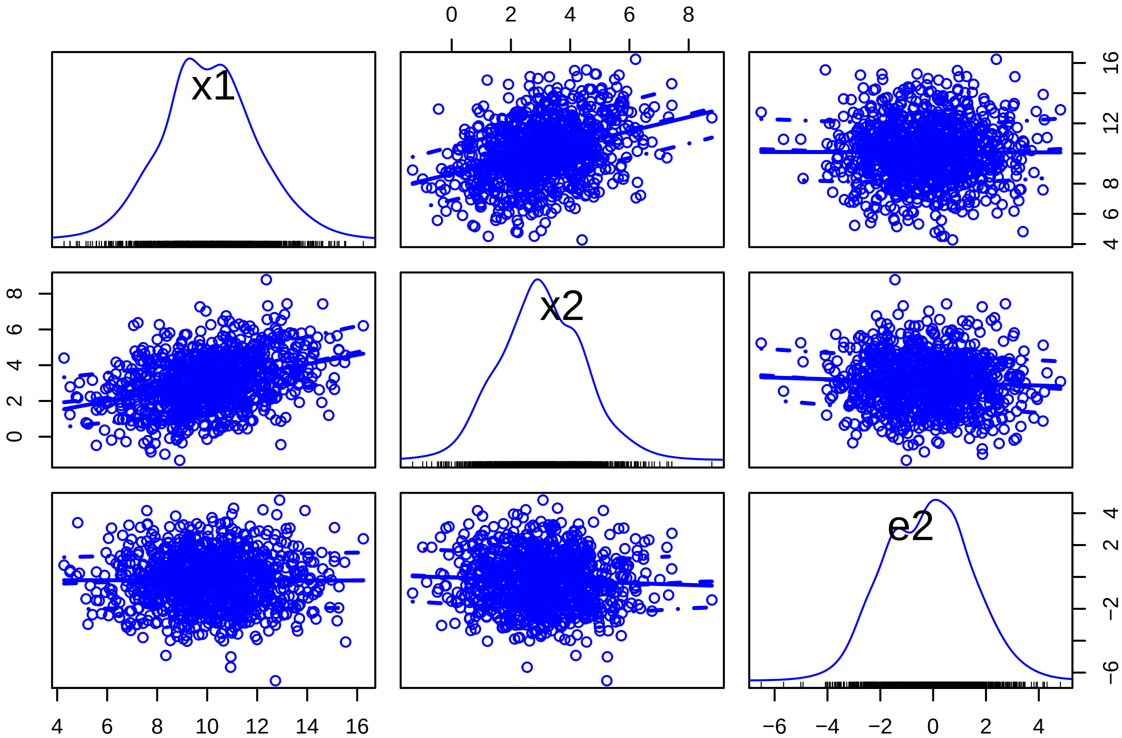
```
# Mean of x1, x2
moo <- c(10,3)
# generate data
Xe <- mvrnorm(n=1000,mu=moo,Sigma=Sig)

# generate independent residuals
eps = rnorm(n=1000,mean=0,sd=sqrt(1))

# generate serially correlated residuals
eps2 <- arima.sim(model=list(ar=c(0.8)),n=1000,sd=1)

Xe1 = as.data.frame(cbind(Xe,eps))
colnames(Xe1)<-c("x1","x2","e1")
Xe2 = as.data.frame(cbind(Xe,eps2))
colnames(Xe2)<-c("x1","x2","e2")
```

- Investigate sample correlations and covariances
  - notice difference from "truth"

```
scatterplotMatrix(Xe1)
```




```
scatterplotMatrix(Xe2)
```

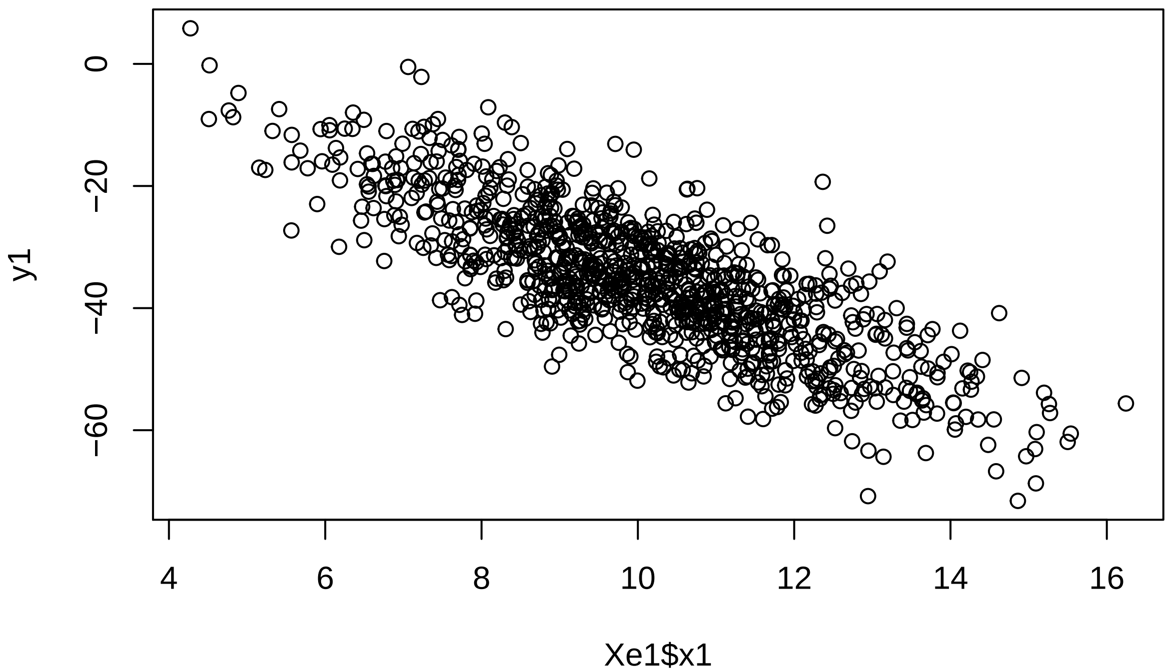


- Generate the true y (outcome)

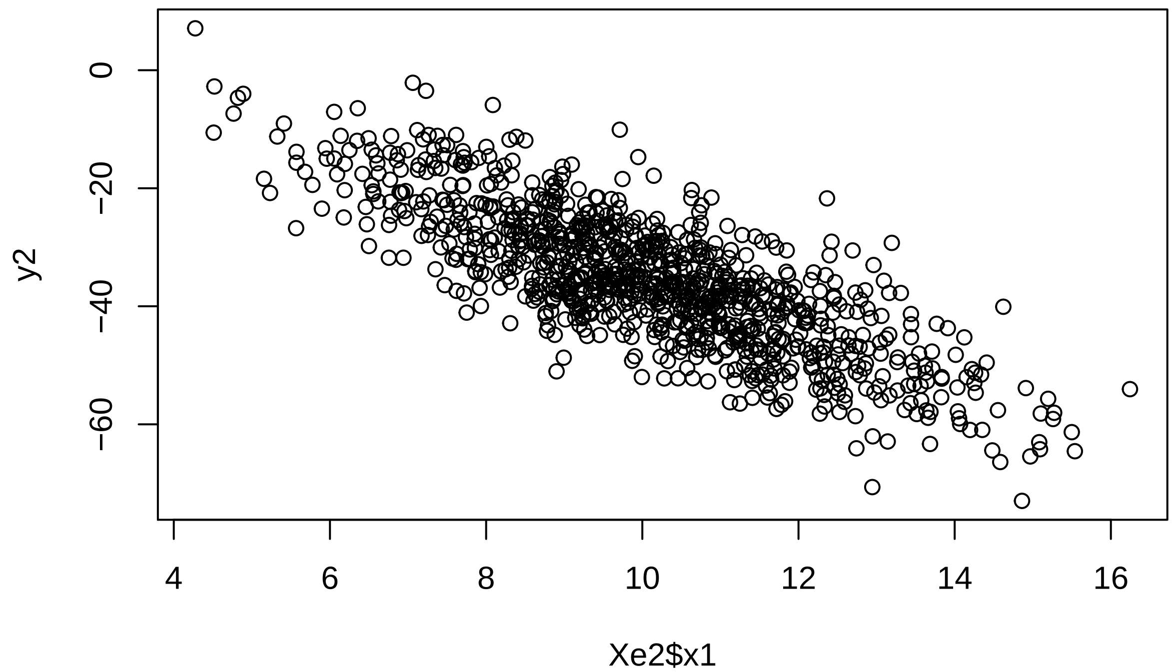
```
y1 <- 10 - 6*Xe1$x1 + 5*Xe1$x2 + Xe1$e1
y2 <- 10 - 6*Xe2$x1 + 5*Xe2$x2 + Xe2$e2
```

- Note that there are not necessarily obvious differences in the plot

```
plot(Xe1$x1,y1)
```




```
plot(Xe2$x1,y2)
```



- Run the linear regression
  - notice the difference in residual standard error, coefficient std. error, Rsquared.

```
summary(lm(y1~x1+x2,data=Xe1))
```

```
##
## Call:
## lm(formula = y1 ~ x1 + x2, data = Xe1)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3647 -0.6814 -0.0060  0.6925  3.0024
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.82512    0.16589   59.23   <2e-16 ***
## x1          -5.99439    0.01719 -348.62   <2e-16 ***
## x2           5.03285    0.02317  217.23   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.004 on 997 degrees of freedom
## Multiple R-squared:  0.9925, Adjusted R-squared:  0.9925
## F-statistic: 6.596e+04 on 2 and 997 DF,  p-value: < 2.2e-16
```

```
summary(lm(y2~x1+x2,data=Xe2))
```

```
##
## Call:
## lm(formula = y2 ~ x1 + x2, data = Xe2)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -6.1938 -1.2369  0.0212  1.1345  4.9921
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  9.83236    0.27727   35.46   <2e-16 ***
## x1          -5.98476    0.02874 -208.24   <2e-16 ***
## x2           4.93465    0.03872  127.43   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.678 on 997 degrees of freedom
## Multiple R-squared:  0.9791, Adjusted R-squared:  0.9791
## F-statistic: 2.339e+04 on 2 and 997 DF,  p-value: < 2.2e-16
```

Violate Assumption 5: Residuals "e" are correlated with X variables

- Some X variables are **endogenous**
- Many flavors of this

```r
set.seed(826)
# Covariance matrix of x1, x2, and e
n <- 3
A <- matrix(runif(n^2)*2-1, ncol=n)
Sig <- t(A) %*% A
Sig       # Notice e covaries with x1 and x2
```

5a. Residuals (unobservable) covary with X's (observable)

```
##            [,1]       [,2]      [,3]
## [1,]  1.1012487 -0.4083661 0.5785058
## [2,] -0.4083661  1.4481911 0.6653081
## [3,]  0.5785058  0.6653081 1.0935305
```

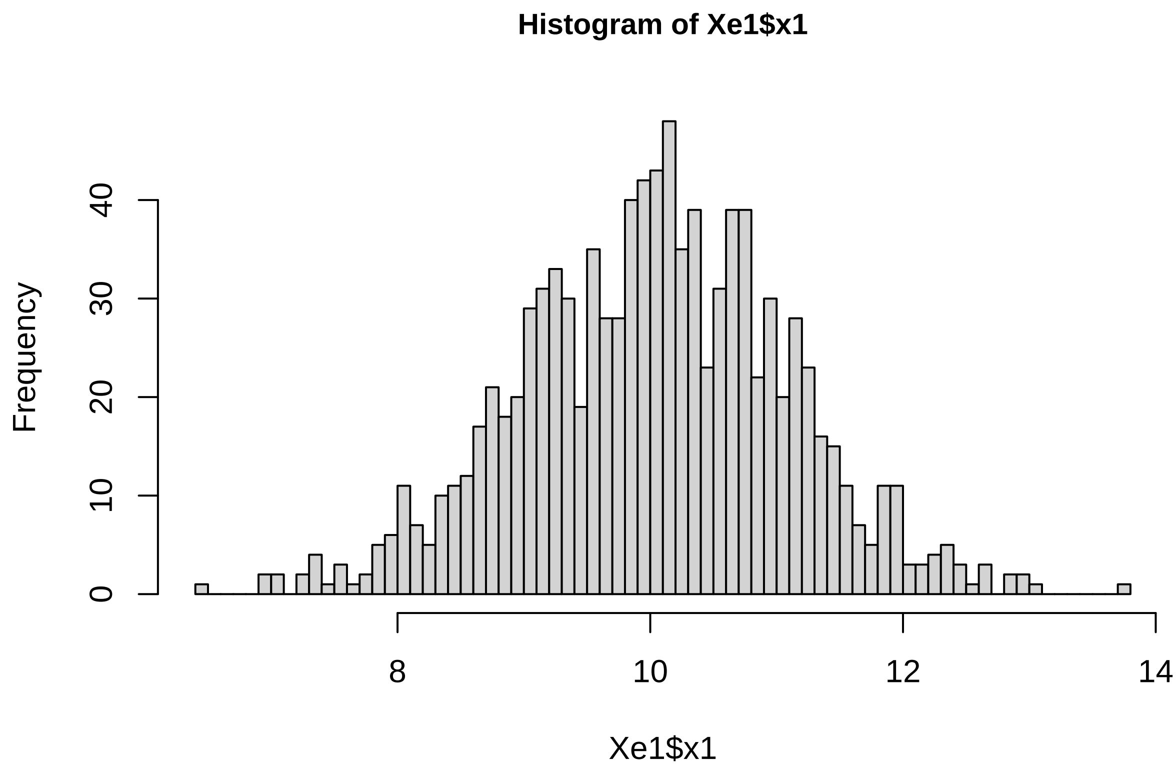
```r
# also x1 and x2 can covary, but not perfectly (assumption 2)
# Mean of x1, x2, and e
moo <- c(10,3,0)
# generate data
Xe1 <- mvrnorm(n=1000,mu=moo,Sigma=Sig)

# give the variables names
colnames(Xe1)<-c("x1","x2","e1")
# store as a data frame
Xe1 <- as.data.frame(Xe1)
head(Xe1)
```

```
##         x1       x2         e1
## 1  7.733325 3.255037 -1.6600797
```

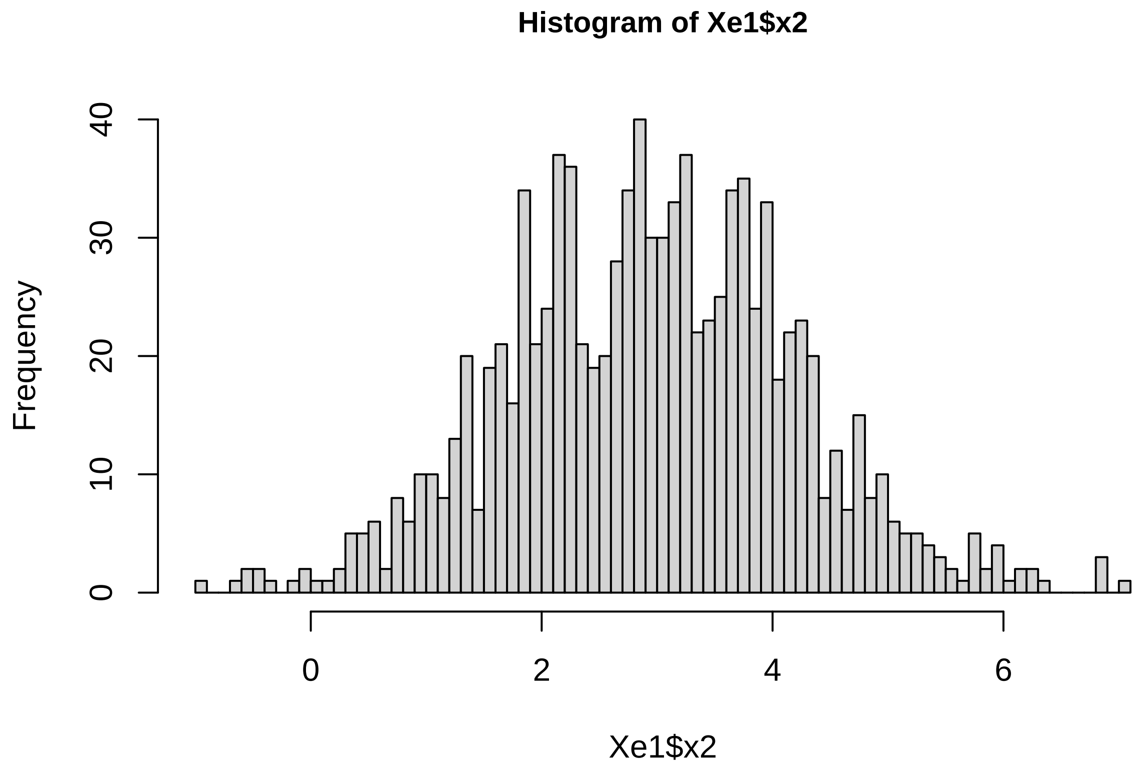
```
## 2 10.031196 4.345290  0.6720729
## 3 10.853379 3.439955  0.6454713
## 4 10.285622 3.138029  0.7302525
## 5 11.973566 3.084988  1.9555648
## 6  9.263035 5.055792  0.2941055
```

- Investigate patterns in the data

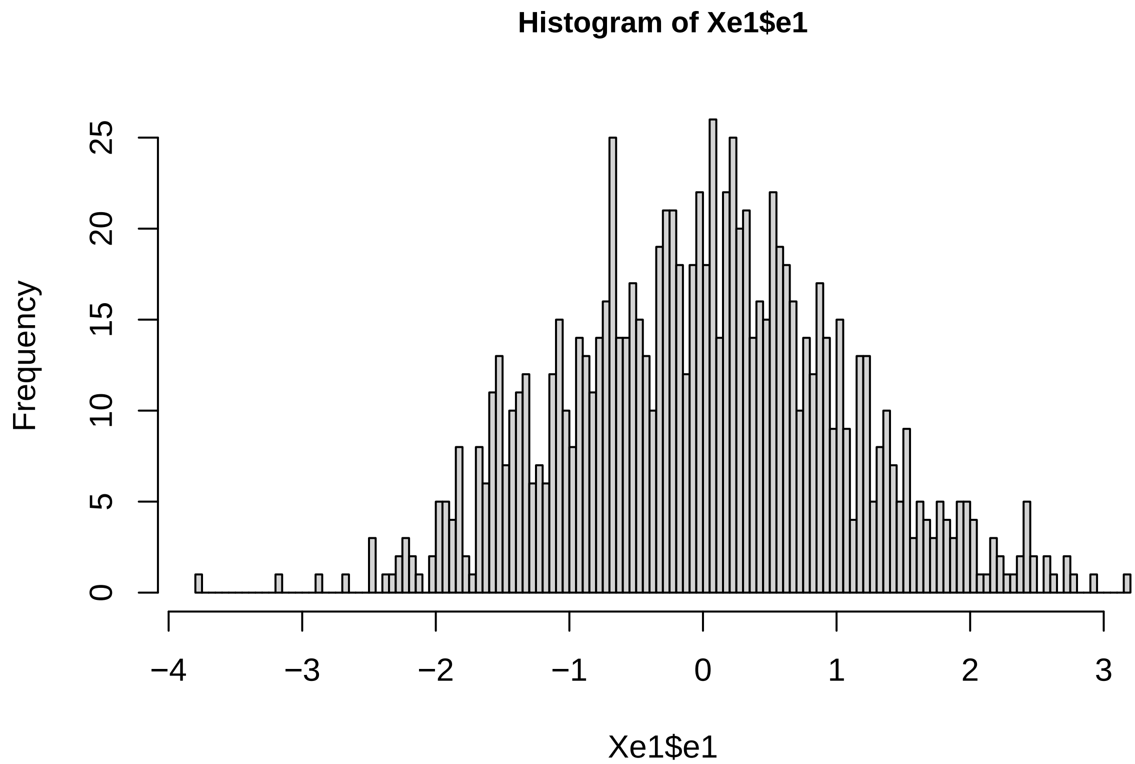
```
# plot empirical distribution of each:
hist(Xe1$x1, breaks = 100, cex.main = 0.9)
```




```
hist(Xe1$x2, breaks = 100, cex.main = 0.9)
```




```
hist(Xe1$e1, breaks = 100, cex.main = 0.9)
```



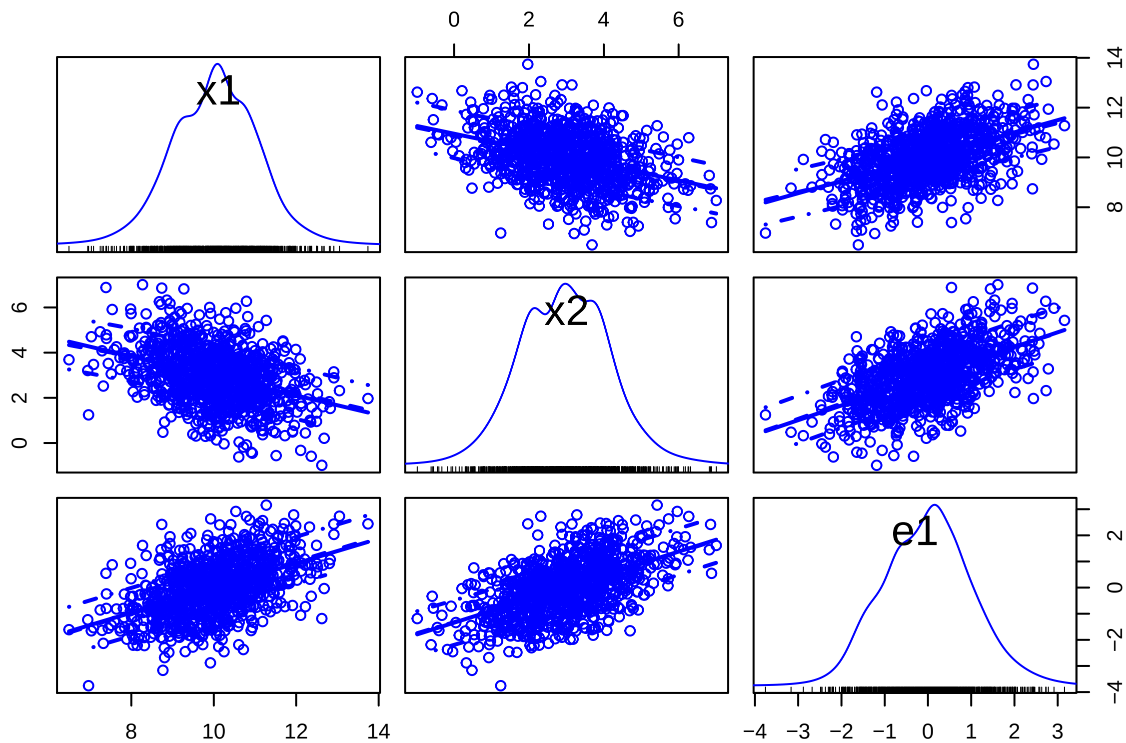

```
# Sample correlations and covariances (notice difference from "truth")
cov(Xe1)
```

```
##             x1         x2        e1
## x1  1.1261792 -0.4863156 0.5318302
## x2 -0.4863156  1.5560936 0.7052121
## e1  0.5318302  0.7052121 1.0883354
```

```
cor(Xe1)
```

```
##             x1         x2        e1
## x1  1.0000000 -0.3673640 0.4803833
## x2 -0.3673640  1.0000000 0.5419017
## e1  0.4803833  0.5419017 1.0000000
```

```
scatterplotMatrix(Xe1)
```

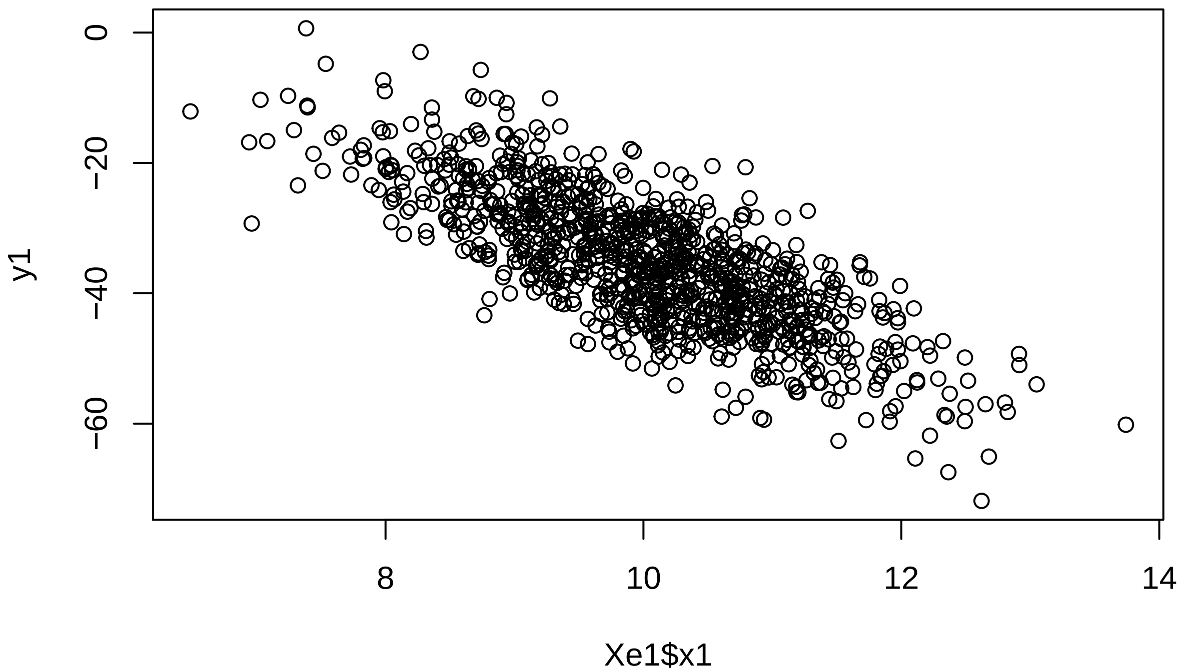


- Generate the true y (outcome)

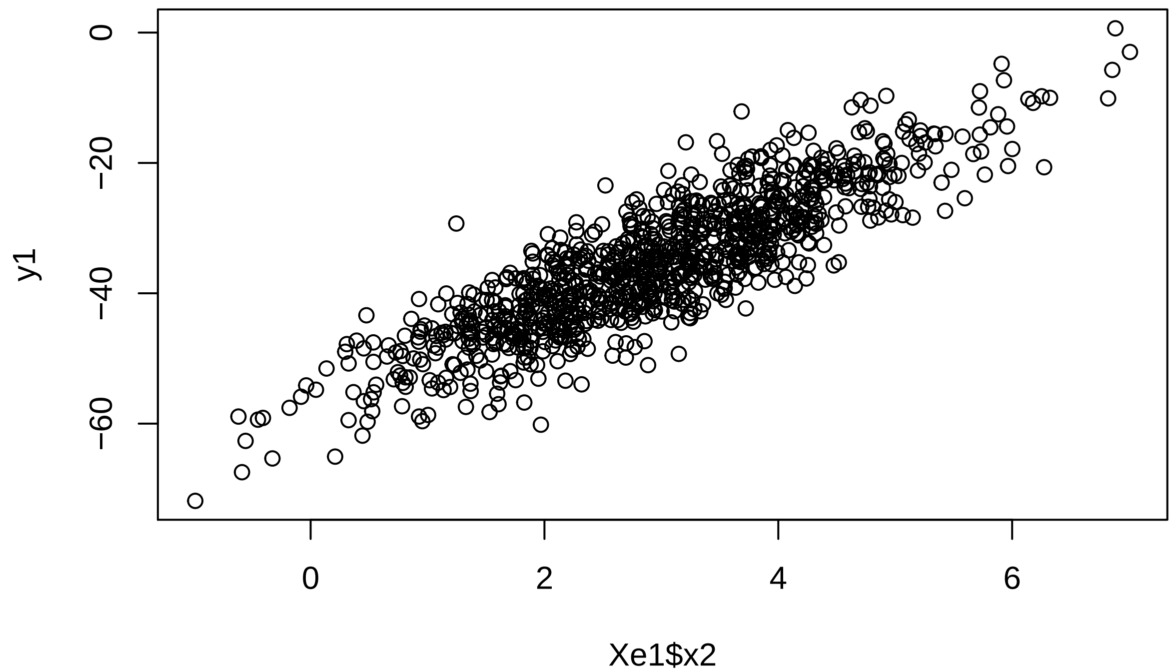
```
y1 <- 10 - 6*Xe1$x1 + 5*Xe1$x2 + Xe1$e1
```

- No obvious problems in the plot of X's against y

```
plot(Xe1$x1,y1)
```




```
plot(Xe1$x2,y1)
```



- Run the linear regression
  - notice the difference in sample coefficients from true values.

```
summary(lm(y1~x1+x2,data=Xe1))
```

```
##
## Call:
## lm(formula = y1 ~ x1 + x2, data = Xe1)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.54018 -0.29630 -0.01474  0.28478  1.25077
##
## Coefficients:
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  0.20152    0.15636    1.289    0.198
## x1          -5.22785    0.01391 -375.858   <2e-16 ***
## x2           5.69451    0.01183  481.250   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4339 on 997 degrees of freedom
## Multiple R-squared:  0.9983, Adjusted R-squared:  0.9983
## F-statistic: 2.923e+05 on 2 and 997 DF,  p-value: < 2.2e-16
```

```
set.seed(826)
# Covariance matrix of x1, x2, and e
Sig <- matrix(c(4,1,0,1,2,0,0,0,1),3,3)
Sig       # Notice e does not covary with x1 or x2 (assumption 5)
```

5b. An observable variable (that is correlated with included variables) was omitted

```
##      [,1] [,2] [,3]
## [1,]    4    1    0
## [2,]    1    2    0
## [3,]    0    0    1
```

```
# also x1 and x2 can covary, but not perfectly (assumption 2)
# Mean of x1, x2, and e
moo <- c(10,3,0)

# generate data
Xe <- mvrnorm(n=1000,mu=moo,Sigma=Sig)
# give the variables names
colnames(Xe)<-c("x1","x2","e")
# store as a data frame
Xe <- as.data.frame(Xe)
```

- Generate the true y (outcome)

```
y2 <- 10 - 6*Xe$x1 + 5*Xe$x2 + Xe$e
```

- Run the linear regression with an omitted variable

```
summary(lm(y2~x1+x2,data=Xe))
```

```
##
## Call:
## lm(formula = y2 ~ x1 + x2, data = Xe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.3647 -0.6814 -0.0060  0.6925  3.0024
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 10.19399    0.16125   63.22   <2e-16 ***
## x1          -6.02720    0.01665 -362.07   <2e-16 ***
## x2           5.01926    0.02357  212.99   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.004 on 997 degrees of freedom
## Multiple R-squared:  0.9931, Adjusted R-squared:  0.993
## F-statistic: 7.126e+04 on 2 and 997 DF,  p-value: < 2.2e-16
```

```
summary(lm(y2~x1,data=Xe))
```

```
##
## Call:
## lm(formula = y2 ~ x1, data = Xe)
```

```
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -23.8197  -4.4152  -0.2228   4.5777  22.8015
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   14.327      1.091   13.13   <2e-16 ***
## x1            -4.938      0.108  -45.73   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.843 on 998 degrees of freedom
## Multiple R-squared:  0.677,  Adjusted R-squared:  0.6766
## F-statistic:  2091 on 1 and 998 DF,  p-value: < 2.2e-16
```

```
summary(lm(y2~x2,data=Xe))
```

```
##
## Call:
## lm(formula = y2 ~ x2, data = Xe)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -33.793  -7.934  -0.057   7.530  41.335
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) -41.7047     0.8497 -49.083   <2e-16 ***
## x2            2.3970     0.2580   9.291   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 11.55 on 998 degrees of freedom
## Multiple R-squared:  0.07961,    Adjusted R-squared:  0.07869
## F-statistic: 86.32 on 1 and 998 DF,  p-value: < 2.2e-16
```

```
set.seed(826)
# generate an autocorrelated residual
eps <- arima.sim(model=list(ar=c(0.8)),n=999,sd=1)

# generate an autocorrelated outcome that has "eps" as its residual
y1 <- list()
y10 <- rnorm(n=1,mean=10,sd=1)
y1[[1]] <- y10
for(i in 2:1000) {
  y1[[i]] <- 10 + 0.4*y1[[i-1]] + eps[i]
}
y1 <- unlist(y1)

arima(y1,order=c(1,0,0))
```

5c. y **is autocorrelated, may or may not have serial correlation/autocorrelation in the residual.**

```
##
## Call:
## arima(x = y1, order = c(1, 0, 0))
##
## Coefficients:
##          ar1  intercept
##       0.9146    16.3248
## s.e.  0.0132     0.3819
##
## sigma^2 estimated as 1.086:  log likelihood = -1459.74,  aic = 2925.48
```