HW09

107070008

Question 1)

```
library(data.table)
ac_bundles_dt <- fread("piccollage_accounts_bundles.csv")
ac_bundles_matrix <- as.matrix(ac_bundles_dt[, -1, with=FALSE])
```

a. Let's explore to see if any sticker bundles seem intuitively similar:

i) (recommended) Download PicCollage onto your mobile from the App Store and take a look at the style and content of various bundles in their Sticker Store: how many recommendations does each bundle have? (NOTE: the Android app might not have recommendations)

6 recommendations

ii)Find a single sticker bundle that is both in our limited data set and also in the app's Sticker Store (e.g., "sweetmothersday"). Then, use your intuition to recommend (guess!) five other bundles in our dataset that might have similar usage patterns as this bundle.

Pellington Image intuition top5: Random, Monsterhigh, alien, KLL, newyearsparty

b. Let's find similar bundles using geometric models of similarity:

i) Let's create cosine similarity based recommendations for all bundles:

1. Create a matrix or data.frame of the top 5 recommendations for all bundles

```
matrix <- data.frame(
  top5 <- apply(ac_bundles_matrix, 2, function(x){sort(x, decreasing =
T)[1:5]})
)
#matrix
```

2. Create a new function that automates the above functionality: it should take an accounts-bundles matrix as a parameter, and return a data object with the top 5 recommendations for each bundle in our data set, using cosine similarity.

```
library(lsa)

## Loading required package: SnowballC

cos_m <- function(x){
  cos_m <- x
  for (i in c(1:165)){cos_m[i,i] <- -10000}
  return(cos_m)
```

```
}
automated_cossin <- function(x) {
  cossin <- cosine(x)
  nameLists <- colnames(cossin)
  recommand_matrix <- data.frame(
    cos_top5 <- apply(cos_m(cossin), 2, function(x){
      nameLists[order(x, decreasing = T)[1:5]]
    })
  )
  return (recommand_matrix)
}
#automated_cossin(ac_bundles_matrix)
```

3. What are the top 5 recommendations for the bundle you chose to explore earlier?

top5: springrose, 8bit2, mmlm, julyfourth, tropicalparadise

ii) Let's create correlation based recommendations.

1. Reuse the function you created above (don't change it; don't use the cor() function)

2. But this time give the function an accounts-bundles matrix where each bundle (column) has already been mean-centered in advance.

```
library(lsa)
cos_m <- function(x){
  cos_m <- x
  for (i in c(1:165)){cos_m[i,i] <- -10000}
  return(cos_m)
}
automated_cossin <- function(x) {
  bundle_mean <- apply(ac_bundles_matrix, 2, mean)
  bundel_mean_matrix <- t(replicate(nrow(ac_bundles_matrix), bundle_mea
n))
  ac_bundles_mc_b <- ac_bundles_matrix - bundel_mean_matrix
  cor_sim <- cosine(ac_bundles_mc_b)
  nameLists <- colnames(cor_sim)
  recommand_matrix <- data.frame(
    cos_top5 <- apply(cos_m(cor_sim), 2, function(x){
      nameLists[order(x, decreasing = T)[1:5]]
    })
  )
  return (recommand_matrix)
}
#automated_cossin(ac_bundles_matrix)
```

3. Now what are the top 5 recommendations for the bundle you chose to explore earlier?

top5: springrose, 8bit2, tropicalparadise, mmlm, julyfourth

iii) Let's create adjusted-cosine based recommendations.

1. Reuse the function you created above (you should not have to change it)

2. But this time give the function an accounts-bundles matrix where each account (row) has already been mean-centered in advance.

```
library(lsa)
cos_m <- function(x){
  cos_m <- x
  for (i in c(1:165)){cos_m[i,i] <- -10000}
  return(cos_m)
}
automated_cossin <- function(x) {
  bundle_mean <- apply(ac_bundles_matrix, 1, mean)
  bundle_mean_matrix <- t(replicate(ncol(ac_bundles_matrix), bundle_mea
n))
  tmp <- as.data.frame(t(bundle_mean_matrix), row.names=F)
  for (i in c(1:165)){
    colnames(tmp)[i] =  nameLists[i]
  }
  bundle_mean_matrix_rev <- tmp
  ac_bundles_mc_b <- ac_bundles_matrix - bundle_mean_matrix_rev
  cossin <- cosine(x)
  nameLists <- colnames(cossin)
  recommand_matrix <- data.frame(
    cos_top5 <- apply(cos_m(cossin), 1, function(x){
      nameLists[order(x, decreasing = T)[1:5]]
    })
  )
  return (recommand_matrix)
}
#automated_cossin(ac_bundles_matrix)
```

3. What are the top 5 recommendations for the bundle you chose to explore earlier?

top5: springrose, 8bit2, mmlm, julyfourth, tropicalparadise

c. (not graded) Are the three sets of geometric recommendations similar in nature (theme/keywords) to the recommendations you picked earlier using your intuition alone? What reasons might explain why your computational geometric recommendation models produce different results from your intuition?

My intuition is totally different from the result of geometric recommendations. Because my intuition is depend on the name of the bundles, the computational result is depend on the data similarity and the number of recommendation. Therefore, I think it will be totally different.

d. (not graded) What do you think is the conceptual difference in cosine similarity, correlation, and adjusted-cosine?

My computational result is different among cosine similarity, correlation, and adjusted-cosine, because those quantities represent different physical entities. The cosine similarity computes the similarity between two samples, whereas the Pearson correlation coefficient computes the correlation between two jointly distributed random variables.

Question 2) Correlation is at the heart of many data analytic methods so let's explore it further.

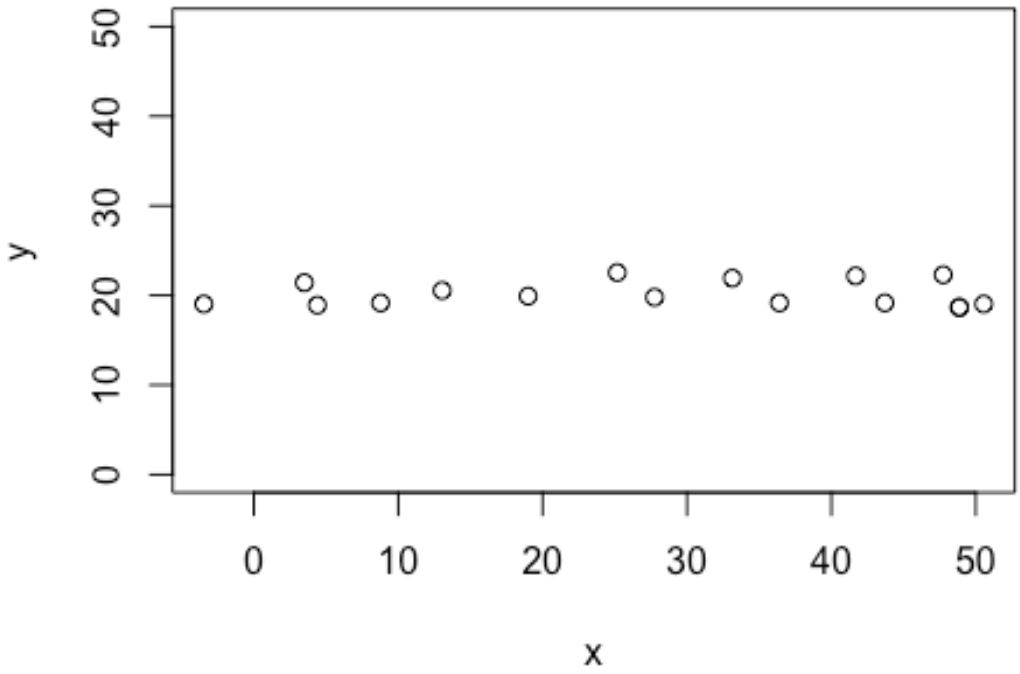
```
source('demo_simple_regression.R')
```

a. Create a horizontal set of random points, with a relatively narrow but flat distribution.

i. What raw slope of x and y would you generally expect?

ii. What is the correlation of x and y that you would generally expect?

```
#qa <- interactive_regression()
qa <- read.table("./qa.txt")
plot(qa, ylim = c(0,50))
```



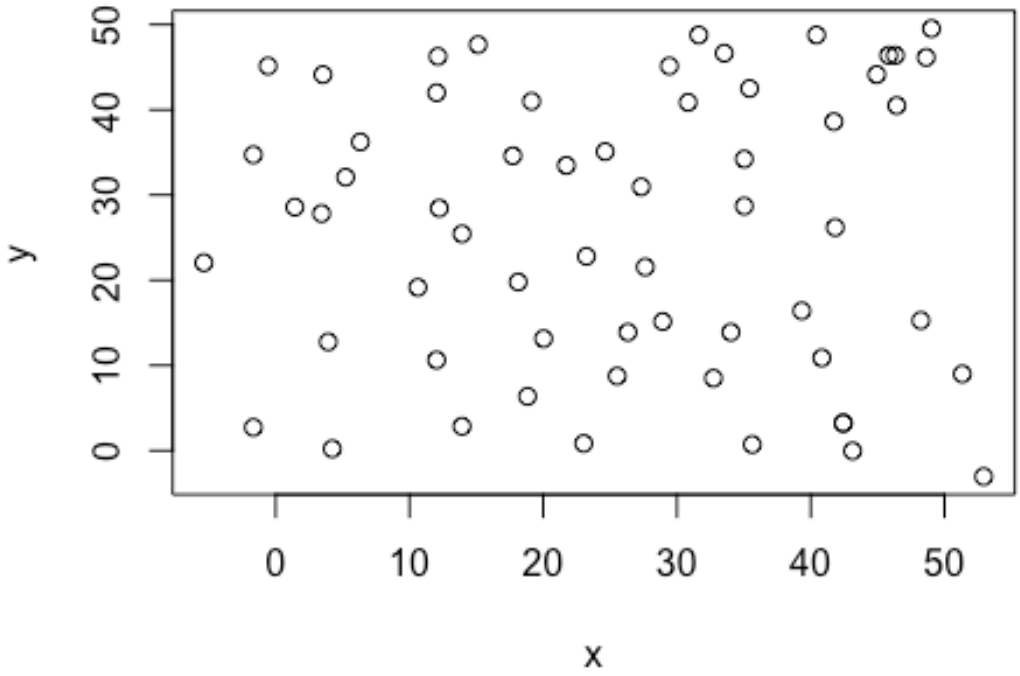

```
lm(qa$y ~ qa$x)$coeff[[2]]

## [1] 0.00191375

cor(qa)

##             x          y
## x 1.00000000 0.02463556
## y 0.02463556 1.00000000
```

b. Create a completely random set of points to fill the entire plotting area, along both x-axis and y-axis

i. What raw slope of the x and y would you generally expect?

ii. What is the correlation of x and y that you would generally expect?

```
#qb <- interactive_regression()
qb <- read.table("./qb.txt")
plot(qb)
```



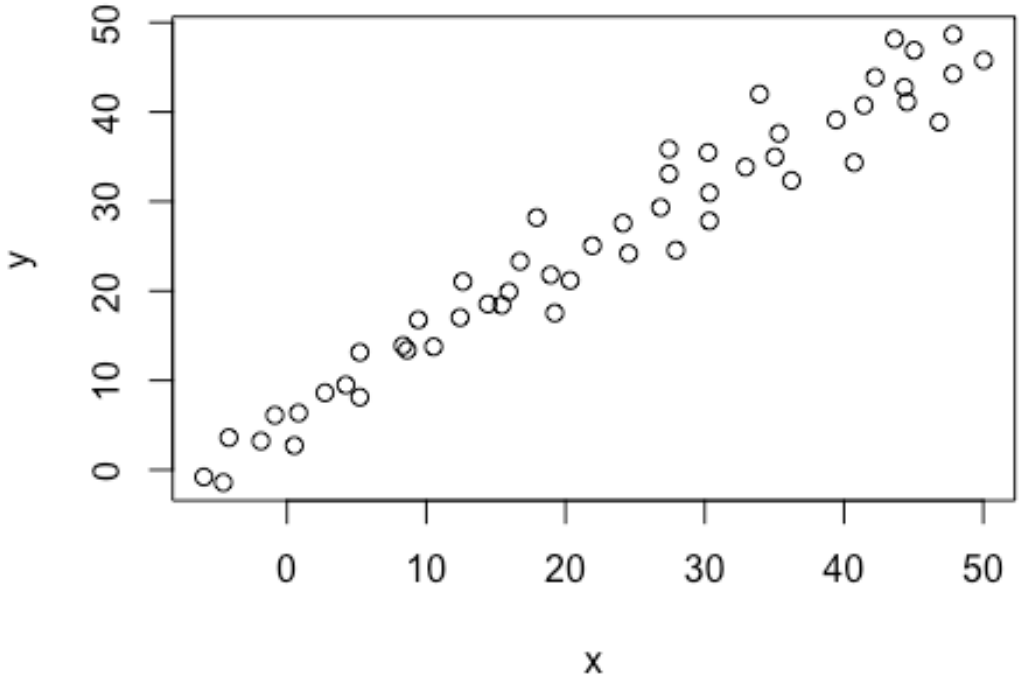

```
lm(qb$y ~ qb$x)$coeff[[2]]

## [1] 0.01122265

cor(qb)

##             x          y
## x 1.00000000 0.01105383
## y 0.01105383 1.00000000
```

c. Create a diagonal set of random points trending upwards at 45 degrees

i. What raw slope of the x and y would you generally expect? (note that x, y have the same scale)

ii. What is the correlation of x and y that you would generally expect?

```
qc <- read.table("./qc.txt")
plot(qc)
```



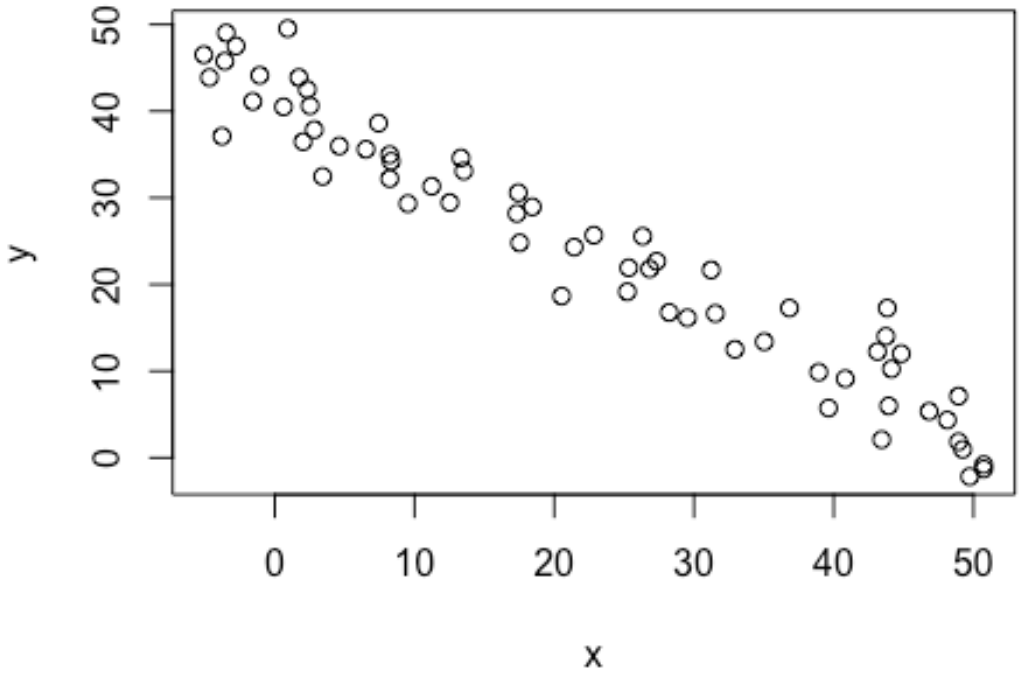

```
lm(qc$y ~ qc$x)$coeff[[2]]

## [1] 0.8408485

cor(qc)

##           x         y
## x 1.0000000 0.9749436
## y 0.9749436 1.0000000
```

d. Create a diagonal set of random trending downwards at 45 degrees

i. What raw slope of the x and y would you generally expect? (note that x, y have the same scale)

ii. What is the correlation of x and y that you would generally expect?

```
qd <- read.table("./qd.txt")
plot(qd)
```

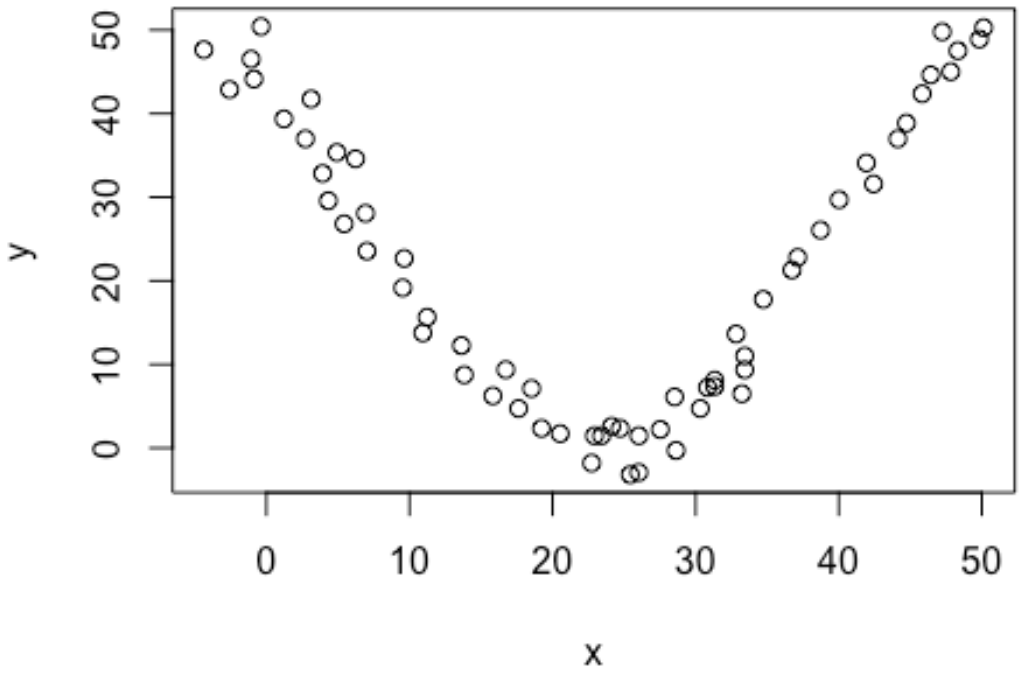



```
lm(qd$y ~ qd$x)$coeff[[2]]

## [1] -0.7756349

cor(qd)

##             x          y
## x  1.0000000 -0.9695215
## y -0.9695215  1.0000000
```

e. Apart from any of the above scenarios, find another pattern of data points with no correlation (r ≈ 0).(can create a pattern that visually suggests a strong relationship but produces r ≈ 0?)

Yes!

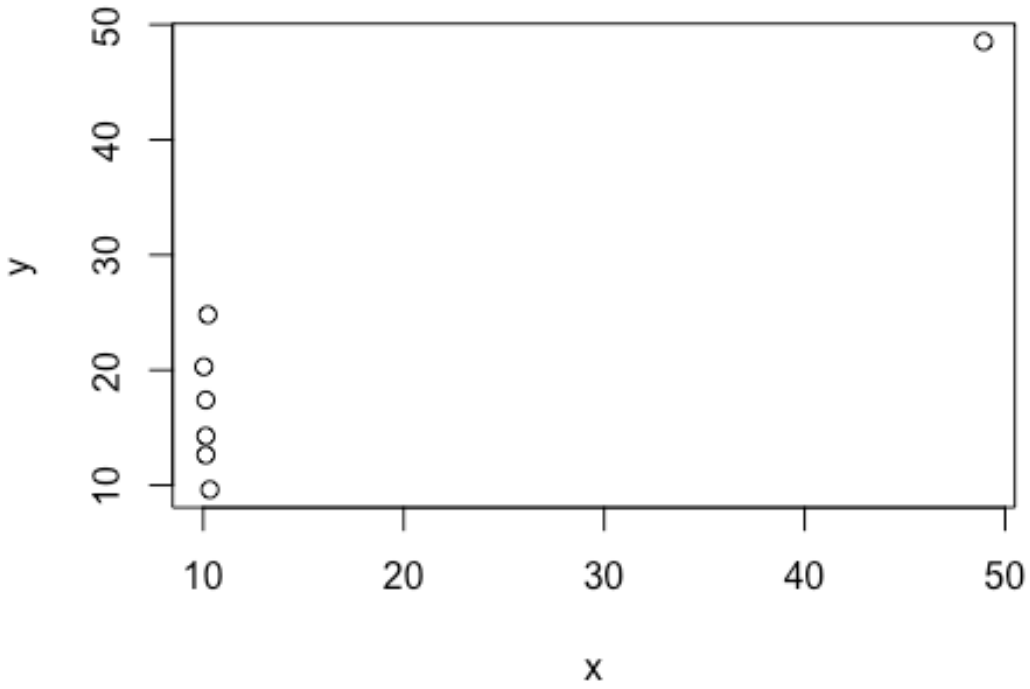
```
Qe <- read.table("./Qe.txt")
plot(Qe)
```



f. Apart from any of the above scenarios, find another pattern of data points with perfect correlation (r ≈ 1).(can you find a scenario where the pattern visually suggests a different relationship?)

Yes!

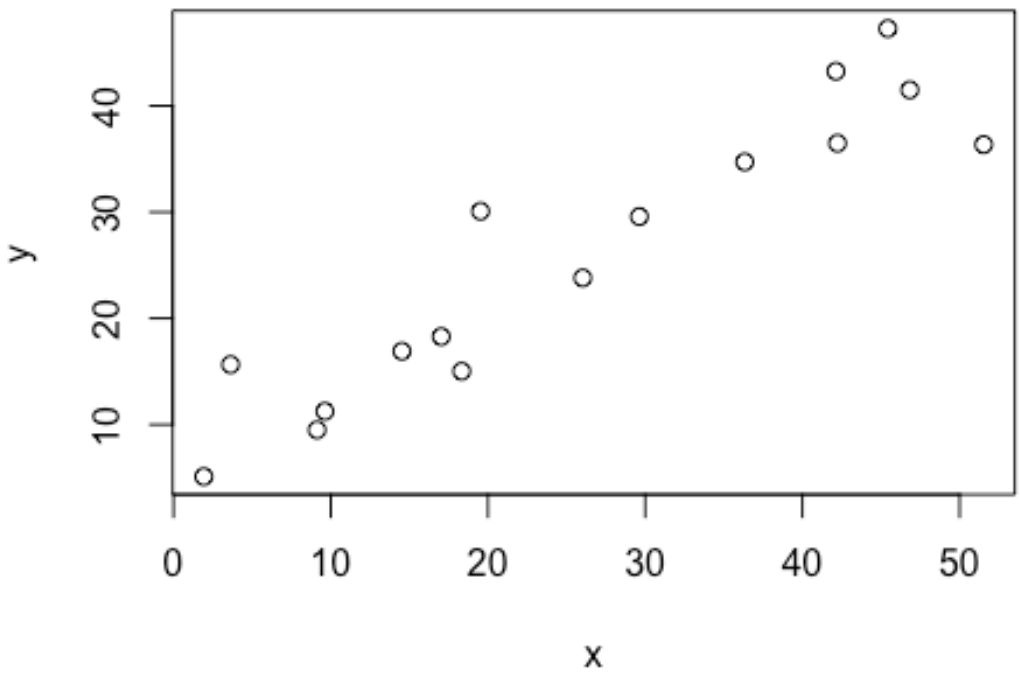
```
Qf <- read.table("./Qf.txt")
plot(Qf)
```



g. Let's see how correlation relates to simple regression, by simulating any linear relationship you wish:

i. Run the simulation and record the points you create: pts <- interactive_regression() (simulate either a positive or negative relationship)

```
pts <- read.table("./pts.txt")
plot(pts)
```



ii. Use the lm() function to estimate the regression intercept and slope of pts to ensure they are the same as the values reported in the simulation plot: summary(lm())

```
summary(lm(pts$y ~ pts$x))

##
## Call:
## lm(formula = pts$y ~ pts$x)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.8073 -2.5875 -0.7669  2.0025  8.9077
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  6.53175    2.34029   2.791   0.0144 *
## pts$x        0.74962    0.07706   9.728 1.31e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 4.905 on 14 degrees of freedom
## Multiple R-squared:  0.8711, Adjusted R-squared:  0.8619
## F-statistic: 94.64 on 1 and 14 DF,  p-value: 1.313e-07
```

iii. Estimate the correlation of x and y to see it is the same as reported in the plot: cor(pts)

```
cor(pts)

##           x         y
## x 1.0000000 0.9333429
## y 0.9333429 1.0000000
```

iv. Now, standardize the values of both x and y from pts and re-estimate the regression slope

```
pts_mean <- apply(pts, 2, mean)
pts_mean_matix <- t(replicate(nrow(pts), pts_mean))
pts_sd <- apply(pts, 2, sd)
pts_sd_matix <- t(replicate(nrow(pts), pts_sd))
pts_std <- (pts - pts_mean_matix)/pts_sd_matix
lm(pts_std$y ~ pts_std$x)$coeff[[2]]

## [1] 0.9333429

cor(pts_std)

##           x         y
## x 1.0000000 0.9333429
## y 0.9333429 1.0000000

summary(lm(pts_std$y ~ pts_std$x))

##
## Call:
## lm(formula = pts_std$y ~ pts_std$x)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.66720 -0.19602 -0.05809  0.15170  0.67481
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.813e-16  9.290e-02   0.000        1
## pts_std$x    9.333e-01  9.594e-02   9.728 1.31e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.3716 on 14 degrees of freedom
## Multiple R-squared:  0.8711, Adjusted R-squared:  0.8619
## F-statistic: 94.64 on 1 and 14 DF,  p-value: 1.313e-07
```

v. What is the relationship between correlation and the standardized simple-regression estimates? The intercept in a standardized simple-regression is 0. Since the mean value of x is not extrapolated, standardized regression also makes the intercept interpretable if unstandardized metric is less meaningful.