STAT 37810 HW1

Boxin

10/11/2018

section 4.1.1, exercises 2

(a)

```
# Construct a Fibonacci sequence first
Fibonacci <- numeric(30)
Fibonacci[1] <- Fibonacci[2] <- 1
for (i in 3:30){
  Fibonacci[i] <- Fibonacci[i - 2] + Fibonacci[i - 1]
}


# Use the Fibonacci sequence to constructe the new sequence
new.sequence <- numeric(30)
new.sequence[1] <- 1
for (i in 2:30) {
  new.sequence[i] <- Fibonacci[i] / Fibonacci[i - 1]
}
new.sequence
```

```
##  [1] 1.000000 1.000000 2.000000 1.500000 1.666667 1.600000 1.625000
##  [8] 1.615385 1.619048 1.617647 1.618182 1.617978 1.618056 1.618026
## [15] 1.618037 1.618033 1.618034 1.618034 1.618034 1.618034 1.618034
## [22] 1.618034 1.618034 1.618034 1.618034 1.618034 1.618034 1.618034
## [29] 1.618034 1.618034
```

The sequence apperas to be converging

(b)

```
(1 + sqrt(5)) / 2
```

```
## [1] 1.618034
```

It seems like the sequence is converging to golden ratio

Simple Proof: Let

$$a_n = f_n / f_{n-1}$$

The we can show that an satifies

$$a_{n+1} = 1 + 1/a_n$$

Assume that the limit of an is x, then x must satisfy

$$x = 1 + 1/x$$

Solve this equation and abandon the negative solution, we have

$$x = (1 + \sqrt{5})/2$$

section 4.1.1, exercises 3

(a)

```
answer <- 0
for (j in 1:5) answer <- answer + j
answer
```

```
## [1] 15
```

(b)

```
answer <- NULL
for (j in 1:5) answer <- c(answer, j)
answer
```

```
## [1] 1 2 3 4 5
```

(c)

```
answer <- 0
for (j in 1:5) answer <- c(answer, j)
answer
```

```
## [1] 0 1 2 3 4 5
```

(d)

```
answer <- 1
for (j in 1:5) answer <- answer * j
answer
```

```
## [1] 120
```

(e)

```
answer <- 3
for (j in 1:15) answer <- c(answer, (7 * answer[j]) %% 31)
answer
```

```
## [1]  3 21 23  6 11 15 12 22 30 24 13 29 17 26 27  3
```

section 4.1.2, exercises 4

```
InterestEarnedByGIC <- function(P, n){
  if (n <= 3){
    total <- P * ((1 + 0.04) ^ n)
  }
  else{
    total <- P * ((1 + 0.05) ^ n)
  }
  return (total - P)
}
```

section 4.1.2, exercises 5

```
MonthlyMortgagePayment <- function(P, n, open){

  if (open == TRUE){
    i <- 0.005
  }
  else {
    i <- 0.004
  }

  R <- (P * i) / (1 - (1 + i) ^ (-n))

  return (R)
}
```

section 4.1.3, exercises 2

```
Fibonacci <- numeric(3)
Fibonacci[1] <- 1
Fibonacci[2] <- 1
Fibonacci[3] <- Fibonacci[1] + Fibonacci[2]
while (Fibonacci[length(Fibonacci)] < 300){
  Fibonacci[(length(Fibonacci) + 1)] <- Fibonacci[length(Fibonacci)] + Fibonacci[(length(Fibonacci) -
1)]
}
Fibonacci <- Fibonacci[1:(length(Fibonacci) - 1)]
Fibonacci
```

```
## [1]   1   1   2   3   5   8  13  21  34  55  89 144 233
```

section 4.1.3, exercises 4

```
i1 <- 0.006
i2 <- (1 - (1 + i1) ^ (-20)) / 19
while (abs(i1 - i2) >= 0.000001){
  itemp <- (1 - (1 + i2) ^ (-20)) / 19
  i1 <- i2
  i2 <- itemp
}
i2
```

```
## [1] 0.004954139
```

```
i1 <- 1
i2 <- (1 - (1 + i1) ^ (-20)) / 19
while (abs(i1 - i2) >= 0.000001){
  itemp <- (1 - (1 + i2) ^ (-20)) / 19
  i1 <- i2
  i2 <- itemp
}
i2
```

```
## [1] 0.004953779
```

```
i1 <- 10
i2 <- (1 - (1 + i1) ^ (-20)) / 19
while (abs(i1 - i2) >= 0.000001){
  itemp <- (1 - (1 + i2) ^ (-20)) / 19
  i1 <- i2
  i2 <- itemp
}
i2
```

```
## [1] 0.004953779
```

When you try different starting guess, the answer has little change

section 4.1.3, exercises 5

```
i1 <- 0.006
i2 <- (1 - (1 + i1) ^ (-20)) / 19
k <- 1
while (abs(i1 - i2) >= 0.000001){
  itemp <- (1 - (1 + i2) ^ (-20)) / 19
  i1 <- i2
  i2 <- itemp
  k <- k + 1
}
i2
```

```
## [1] 0.004954139
```

```
k
```

```
## [1] 74
```

```
i1 <- 0.1
i2 <- (1 - (1 + i1) ^ (-20)) / 19
k <- 1
while (abs(i1 - i2) >= 0.000001){
  itemp <- (1 - (1 + i2) ^ (-20)) / 19
  i1 <- i2
  i2 <- itemp
  k <- k + 1
}
i2
```

```
## [1] 0.004953499
```

```
k
```

```
## [1] 106
```

```
i1 <- 10
i2 <- (1 - (1 + i1) ^ (-20)) / 19
k <- 1
while (abs(i1 - i2) >= 0.000001){
  itemp <- (1 - (1 + i2) ^ (-20)) / 19
  i1 <- i2
  i2 <- itemp
  k <- k + 1
}
i2
```

```
## [1] 0.004953779
```

```
k
```

```
## [1] 106
```

section 4.1.5, exercise 2

(b)

Suppose that there is a m, which is not prime. By the algorithm, m must can be represented as:

$$m = k_1 \times k_2 \quad k_1, k_2 > p$$

Thus

$$m > p^2 \geq n$$

Which is a contradiction

(c)

```
Eratosthenes <- function(n) {
  # Print prime numbers up to n (based on the sieve of Eratosthenes)
  if (n >= 2) {
    sieve <- seq(2, n)
    primes <- c()
    while (length(sieve) > 0) {
      p <- sieve[1]
      if (p >= sqrt(n)) {
        primes <- c(primes, sieve)
        break
      }
      primes <- c(primes, p)
      sieve <- sieve[(sieve %% p) != 0]
    }
    return(primes)
  } else {
    stop("Input value of n should be at least 2.")
  }
}
```

section 4.2.1, exercises 2

(a)

```
compound.interest <- function(P, i.r, n){
  return (P * ((1 + i.r) ^ n))
}
```

(b)

```
compound.interest(1000, 0.01, 30)
```

```
## [1] 1347.849
```

section 4.2.1, exercises 3

```
TestFunction <- function(x){
  return (x + 1)
}

threshold <- 0.01

CalculateZeroPoint <- function(f){
  for (i in c(-20:20)){
    if (f(i) < 0){
      low.bound <- i
      break
    }
  }
  for (i in c(-20:20)){
    if (f(i) > 0){
      up.bound <- i
      break
    }
  }
  while (abs(low.bound - up.bound) > threshold){
    mid <- (low.bound + up.bound) / 2
    if (f(mid) < 0){
      low.bound <- mid
    } else if (f(mid) > 0){
      up.bound <- mid
    } else {
      return (mid)
    }
  }
  return (mid)
}

CalculateZeroPoint(TestFunction)
```

```
## [1] -1.005859
```

section 4.4.3, exercises 1

```
factorial(10)
```

```
## [1] 3628800
```

```
factorial(50)
```

```
## [1] 3.041409e+64
```

```
factorial(100)
```

```
## [1] 9.332622e+157
```

```
factorial(1000)
```

```
## Warning in factorial(1000): 'gammafn'里的值在范围外
```

```
## [1] Inf
```

section 4.4.3, exercises 2

(a)

```
BinoCoefficient <- function(n, m){
  return (factorial(n) / (factorial(n - m) * factorial(m)))
}
```

(b)

```
BinoCoefficient(4, 2)
```

```
## [1] 6
```

```
BinoCoefficient(50, 20)
```

```
## [1] 4.712921e+13
```

```
BinoCoefficient(5000, 2000)
```

```
## Warning in factorial(n): 'gammafn'里的值在范围外
```

```
## Warning in factorial(n - m): 'gammafn'里的值在范围外
```

```
## Warning in factorial(m): 'gammafn'里的值在范围外
```

```
## [1] NaN
```

(c)

```
ImprovedBinoCoefficient <- function(n, m){
  if(n > 10 && (n - m > 5)){
    # By Stirling's approximation, we can improve the calculation
    return (((n / (2 * pi * (n - m) * m)) ^ (1/2)) * exp(-(n - m) * log(1 - (m / n)) - m * log(m /
 n)))
  } else {
    return (factorial(n) / (factorial(n - m) * factorial(m)))
  }
}
```

(d)

```
ImprovedBinoCoefficient(4, 2)
```

```
## [1] 6
```

```
ImprovedBinoCoefficient(50, 20)
```

```
## [1] 4.737859e+13
```

```
ImprovedBinoCoefficient(5000, 2000)
```

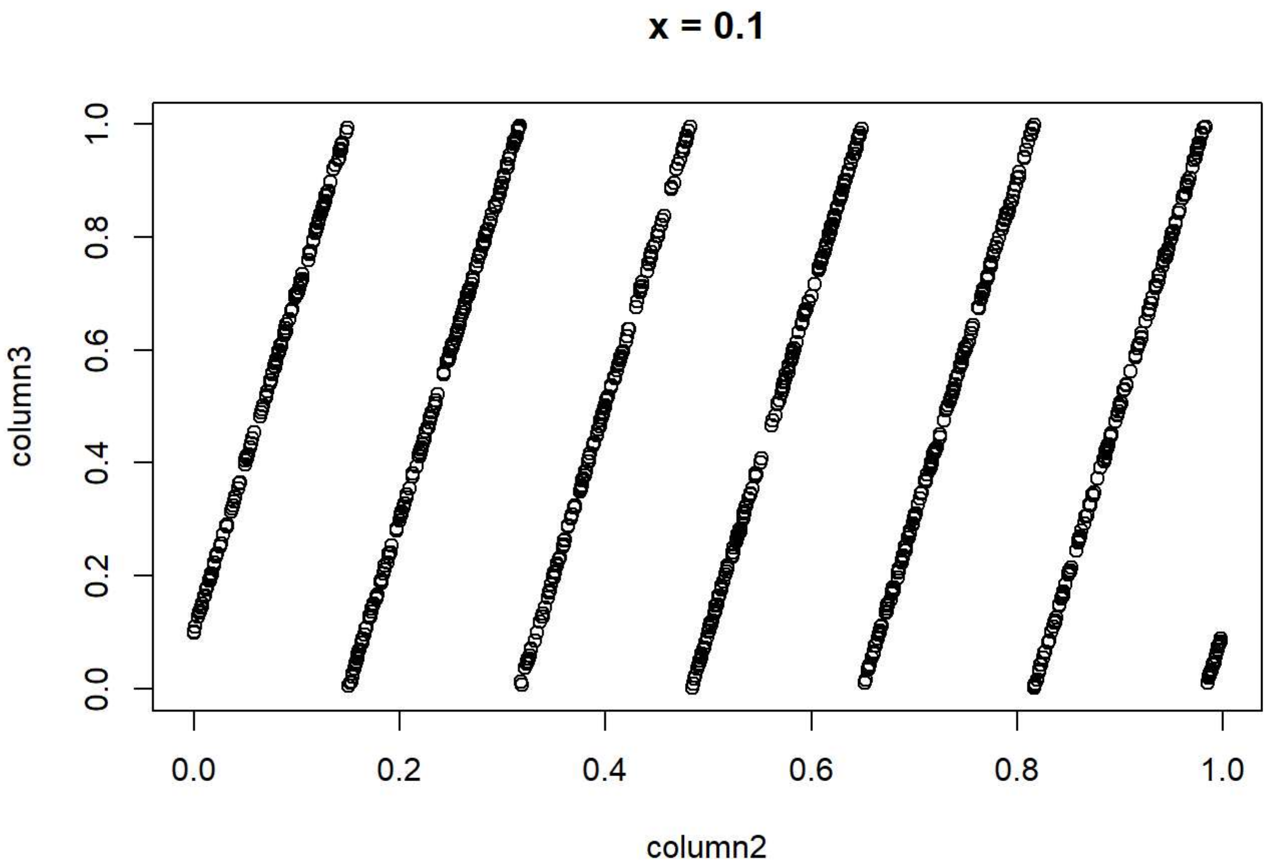
```
## [1] Inf
```

Chapter 4 exercises 1

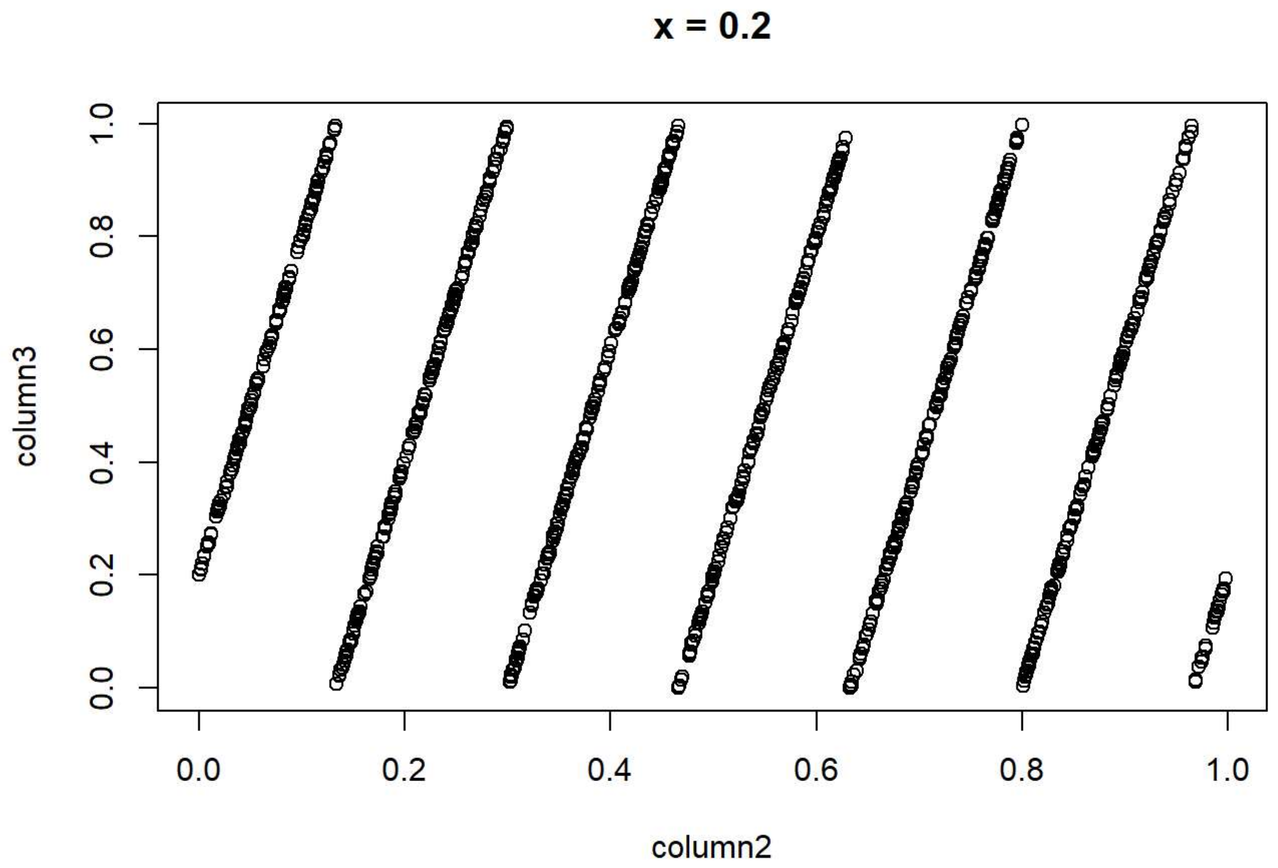
```
results <- numeric(3000000)
x <- 123
for (i in 1:3000000) {
  x <- (65539 * x) %% (2 ^ 31)
  results[i] <- x / (2 ^ 31)
}
results <- round(results, 3)

m <- matrix(results, nrow=1000000, ncol=3, byrow=TRUE)
```

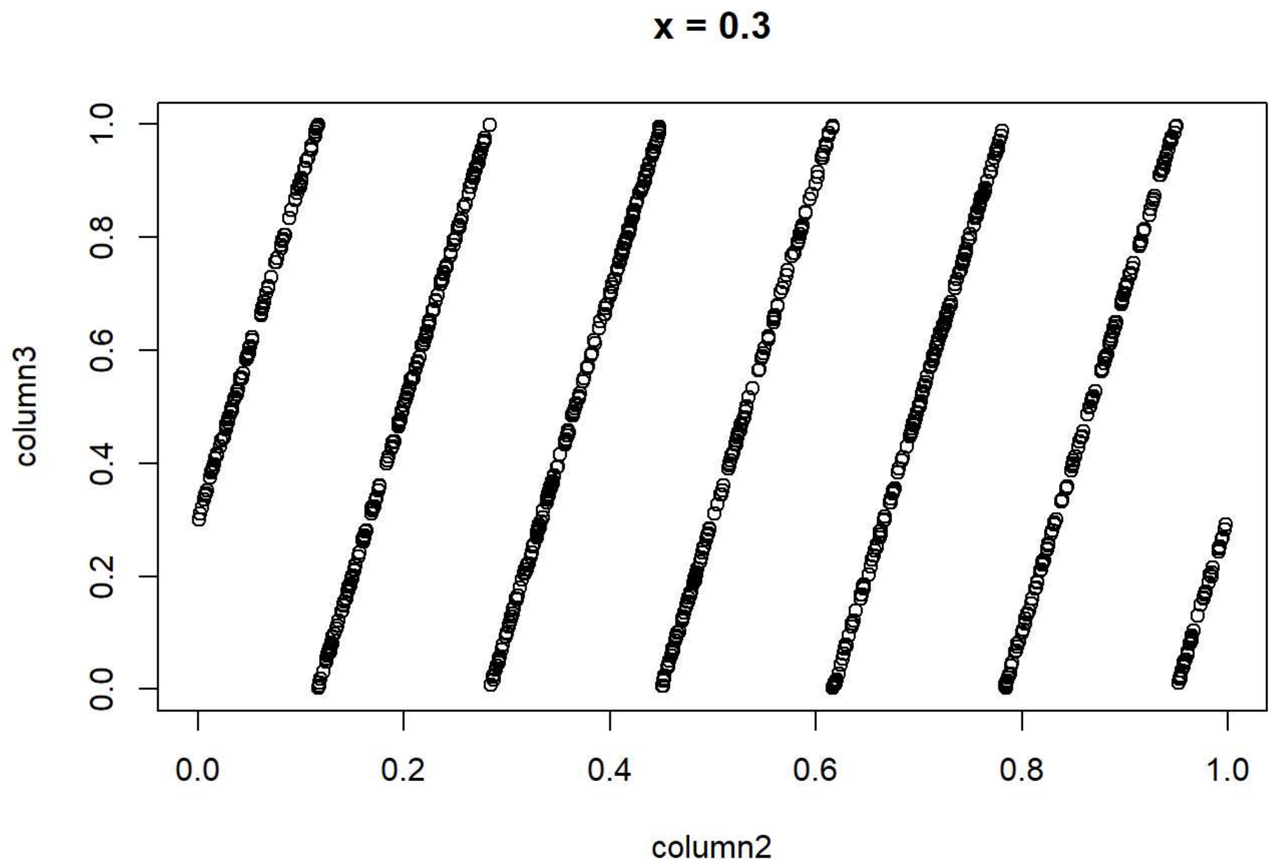
```
x <- 0.1
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.1", xlab="column2", ylab="column3")
```



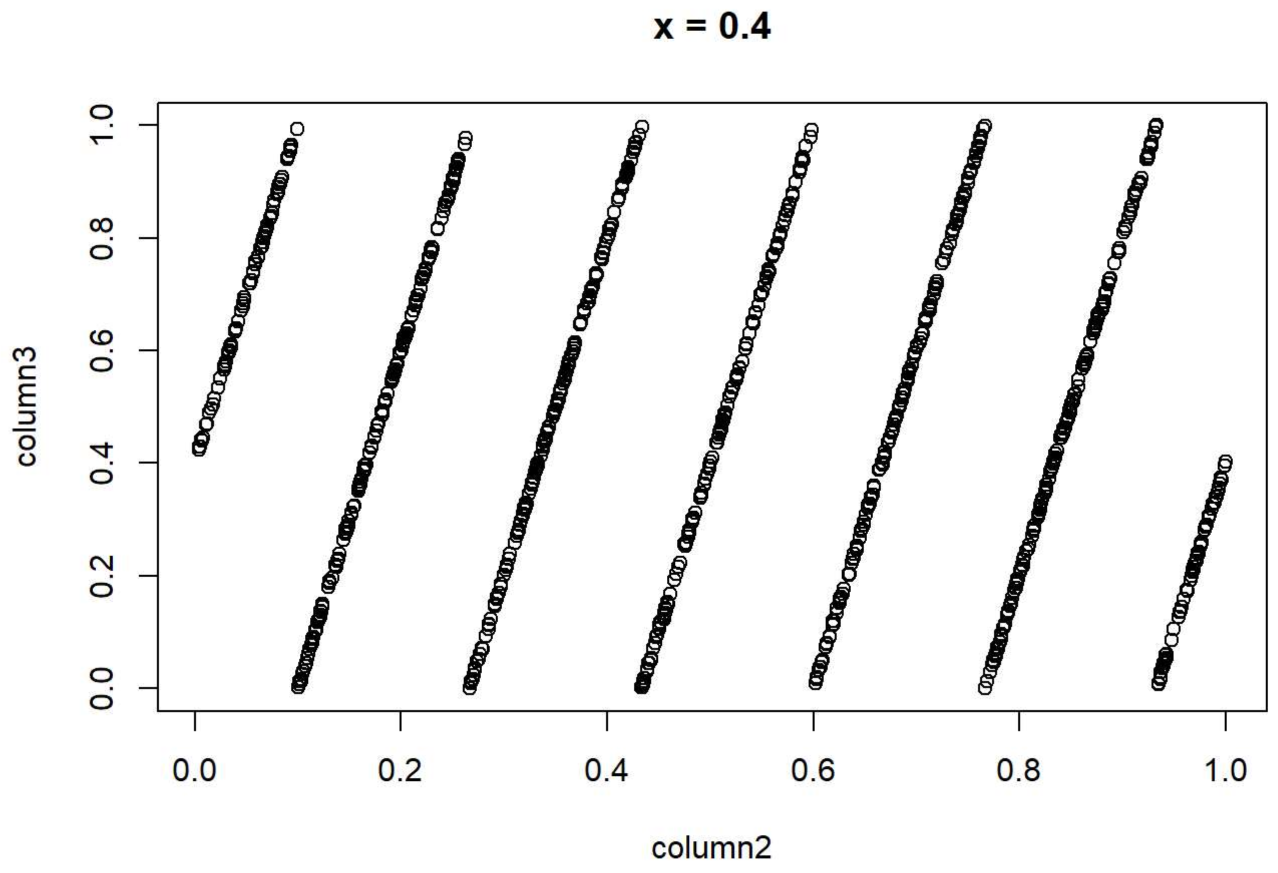

```
x <- 0.2
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.2", xlab="column2", ylab="column3")
```



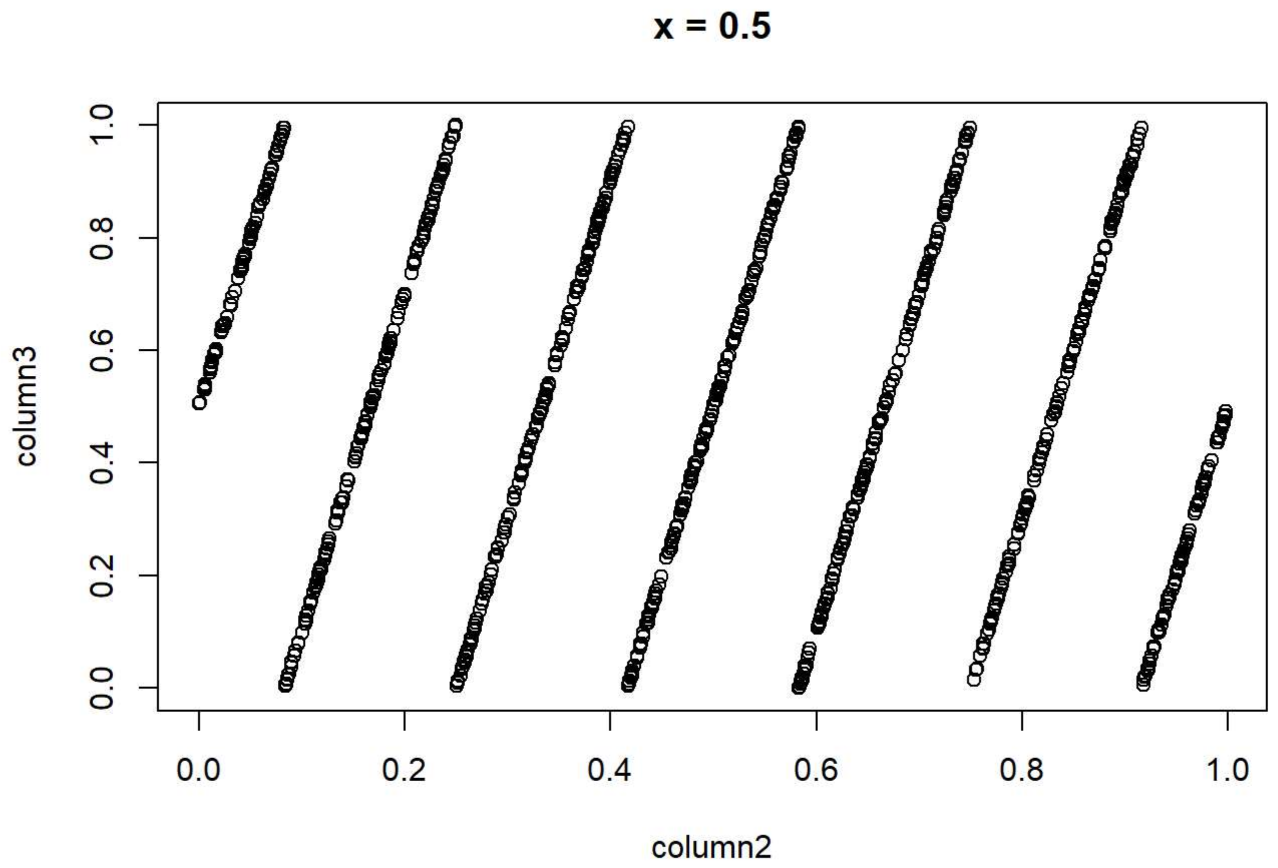

```
x <- 0.3
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.3", xlab="column2", ylab="column3")
```



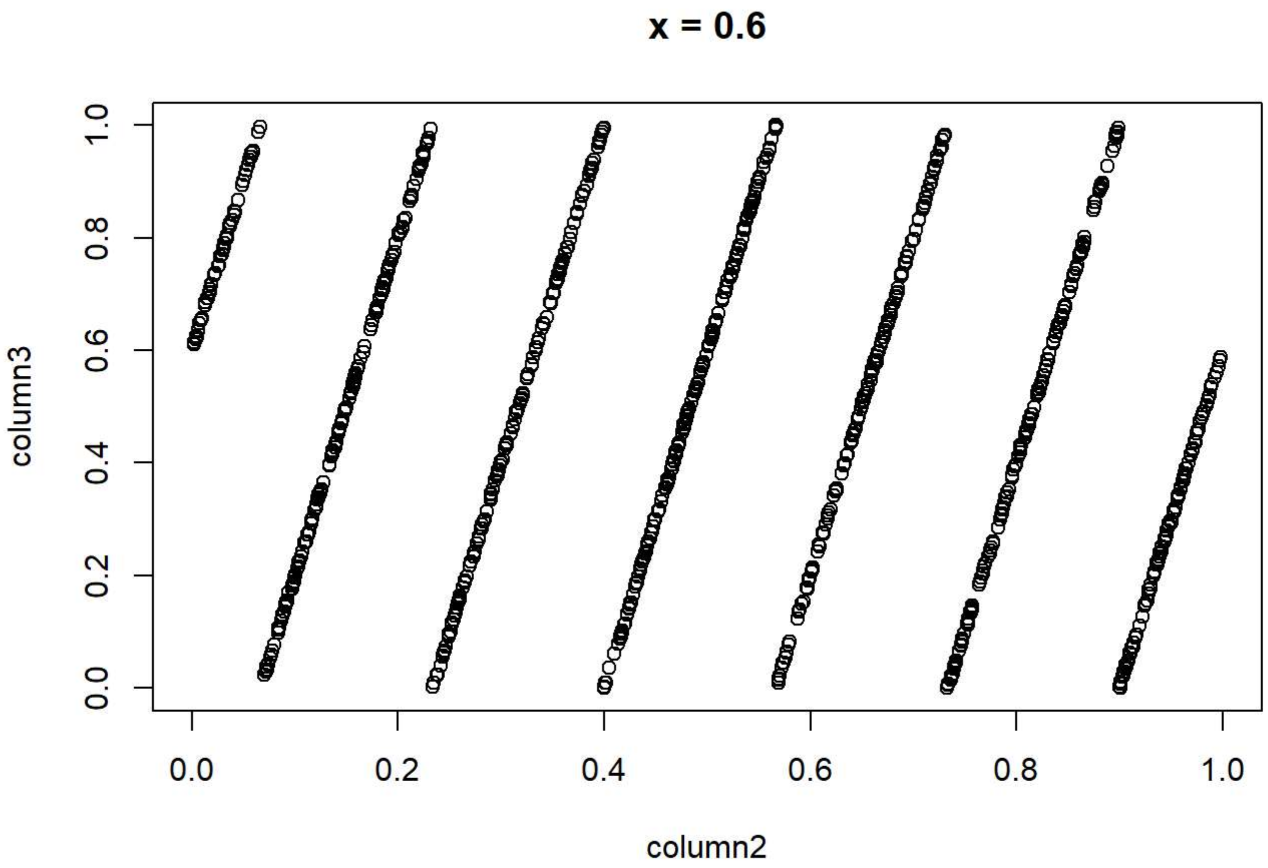

```
x <- 0.4
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.4", xlab="column2", ylab="column3")
```



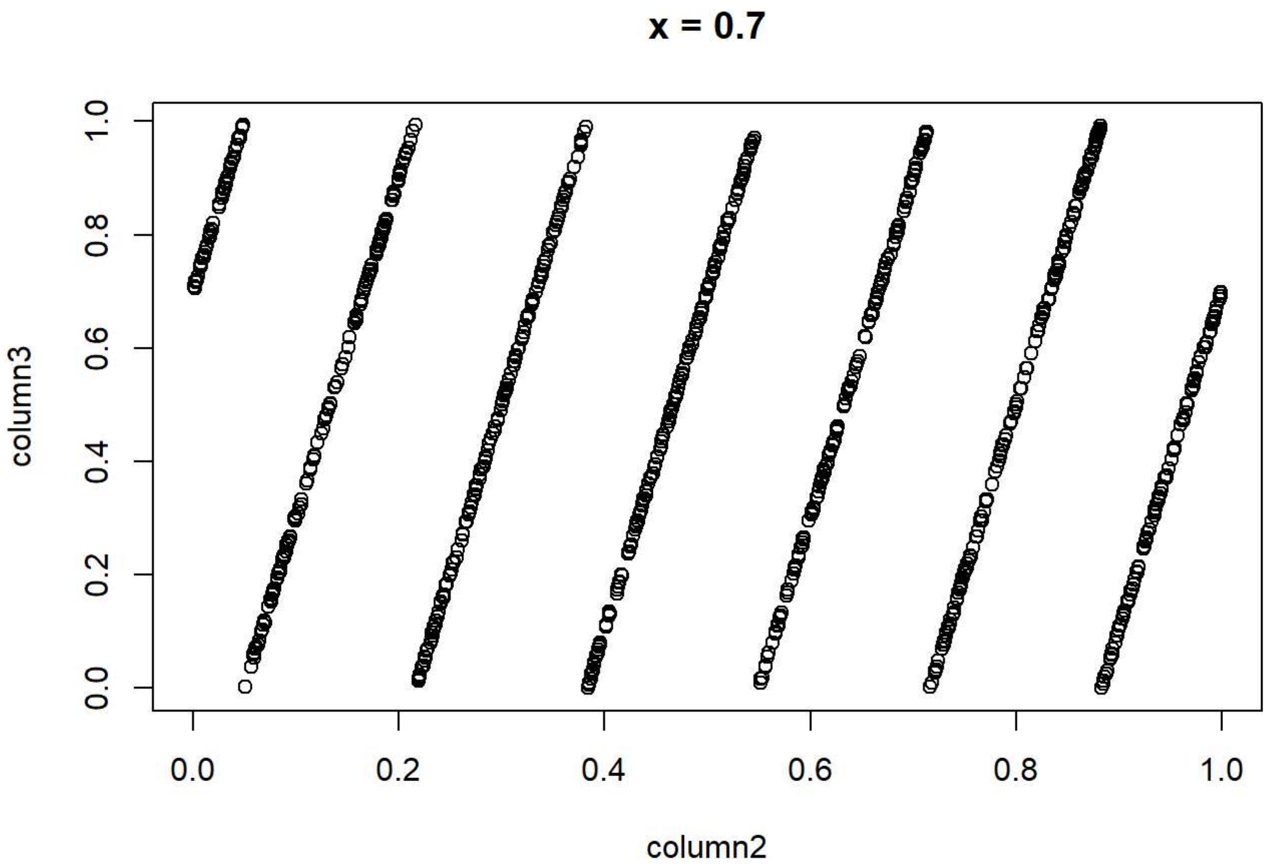

```
x <- 0.5
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.5", xlab="column2", ylab="column3")
```



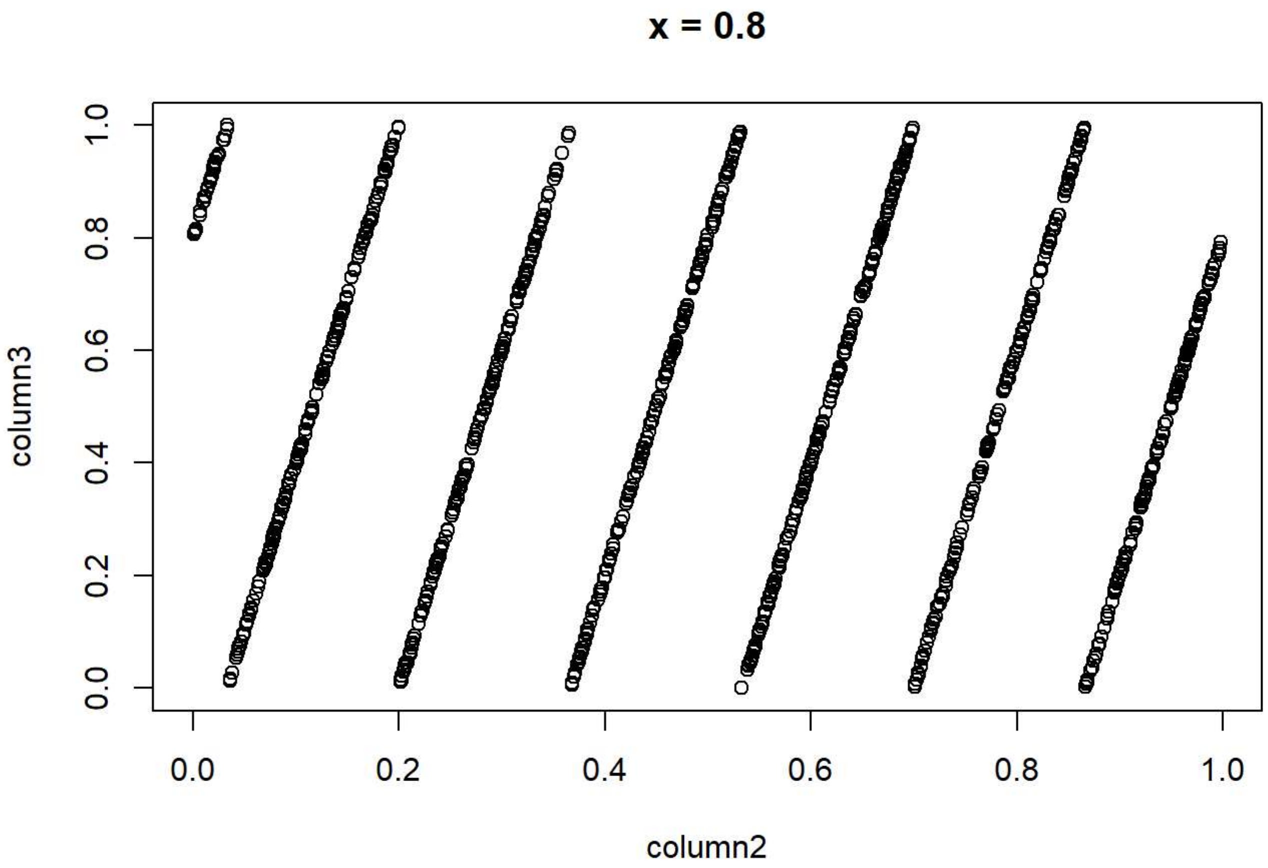

```
x <- 0.6
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.6", xlab="column2", ylab="column3")
```



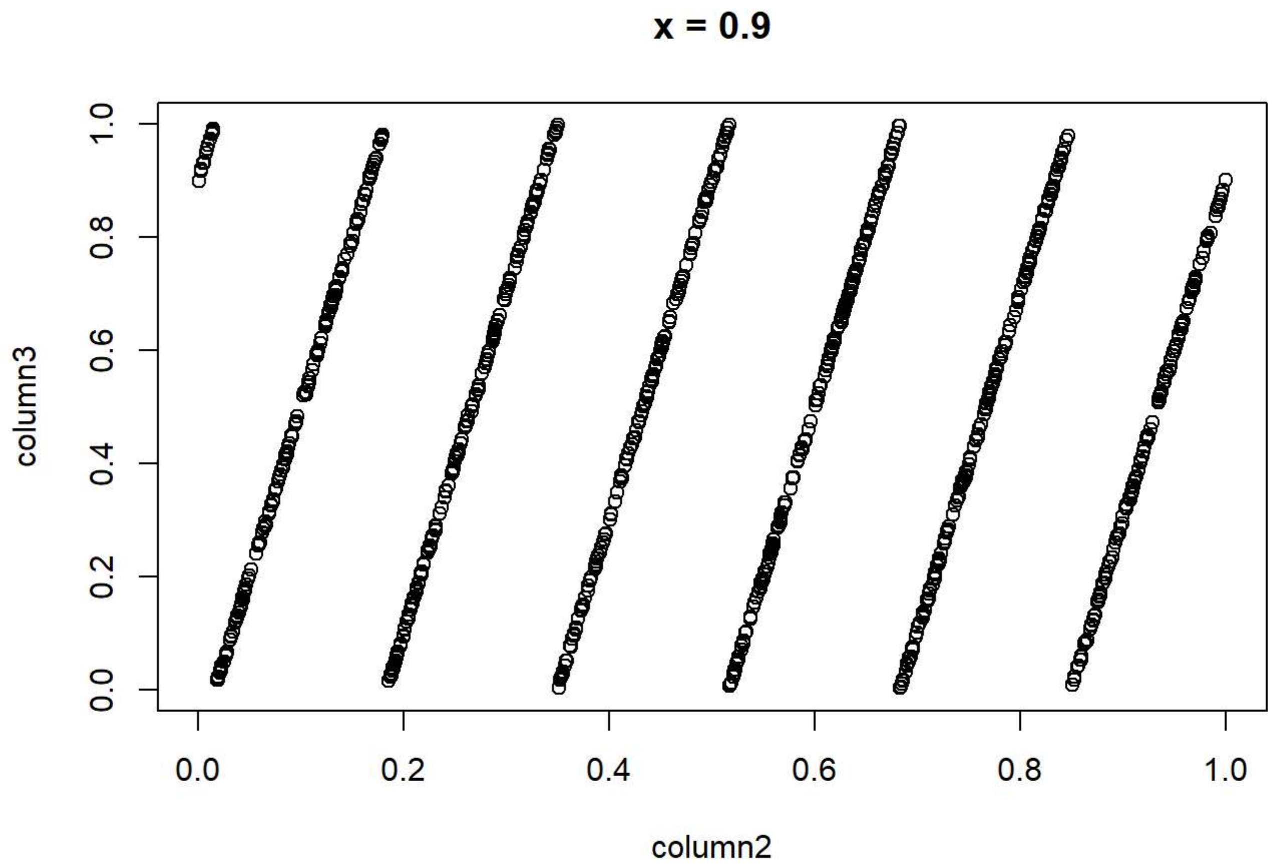

```
x <- 0.7
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.7", xlab="column2", ylab="column3")
```




```
x <- 0.8
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.8", xlab="column2", ylab="column3")
```




```
x <- 0.9
m.sub <- m[(m[, 1] == x), ]
plot(m.sub[, 2], m.sub[, 3], main="x = 0.9", xlab="column2", ylab="column3")
```



Chapter 4 exercises 2

```
directpoly <- function(x, v){
  n <- length(v)
  result <- 0
  for (i in c(1:n)){
    result <- result + x ^ (i - 1) * v[i]
  }
  return (result)
}

directpoly(2, c(1, 2, 3))
```

```
## [1] 17
```

Chapter 4 exercises 3

```r
hornerpoly <- function(y, c){
  result <- numeric(length(y))
  k <- 0
  for (x in y){
    n <- length(c)
    a <- numeric(n)
    a[n] <- c[n]
    for (i in c((n - 1):1)){
      a[i] <- a[i+1] * x + c[i]
    }
    k <- k + 1
    result[k] <- a[1]
  }
  return (result)
}

hornerpoly(2, c(1, 2, 3))
```

```
## [1] 17
```

```r
hornerpoly(c(2, 3), c(1, 2, 3))
```

```
## [1] 17 34
```