Homework 3

DATA604 Simulation and Modeling

Daniel Dittenhafer

March 6, 2016

```
##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

```
library(knitcitations)
library(RefManageR)

cleanbib()

cite_options(style="markdown")

bibPkgTseries <- bibentry(bibtype="Misc",
                 author=personList(person(family="Trapletti", given="Adrian"),
                                   person(family="Hornik", given="Kurt")),
                 journal="Annual Review of Sociology",
                 title="tseries: Time Series Analysis and Computational Finance",
                 year=2015,
                 note="R package version 0.10-34",
                 url="http://CRAN.R-project.org/package=tseries")
```

1

Starting with $X_0 = 1$, write down the entire cycle for $X_i = 11X_{i-1}$ $mod(16)$

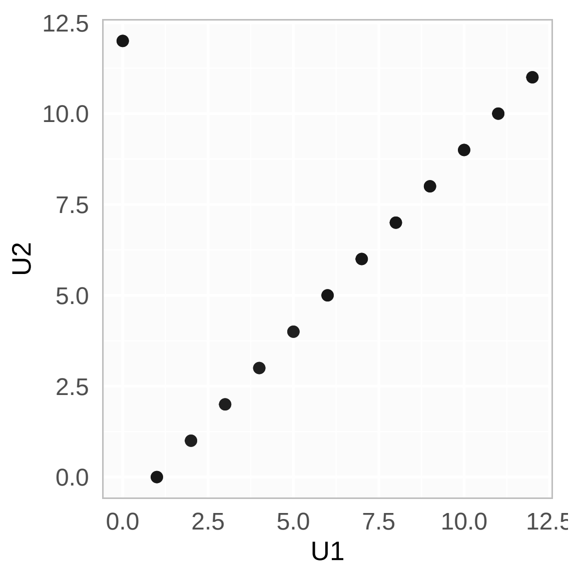
```
fn1 <- function(x0)
{
  df <- data.frame(X=c(), R=c())
  x <- x0
  continue <- TRUE

  while(continue)
  {
    xi <- (11 * x) %% 16
    df <- rbind(df, data.frame(X=x, R=xi))
    x <- xi

    if(xi == x0)
    {
      break
    }
  }

  return(df)
}

res <- fn1(1)
```

X	R
1	11
11	9
9	3
3	1

2

*Using the LCG provided below: $X_i = (X_{i-1} + 12) mod(13)$, plot the pairs $(U_1.U_2), (U_2, U_3), ...$ and observe the lattice structure obtained. Discuss what you observe.

```r
fn2 <- function(x0, max=100)
{
  df <- data.frame(U1=c(), U2=c())
  x <- x0

  for(i in 1:max)
  {
    xi <- (x + 12) %% 13
    df <- rbind(df, data.frame(U1=x, U2=xi))
    x <- xi
  }

  return(df)
}
# Call the function starting at x0=1
res <- fn2(1)
```



The chart above suggests there are only 13 points, but actually the LCG cycle period is 13 and numbers are repeating.

U1	U2
1	0
0	12
12	11
11	10
10	9
9	8
8	7
7	6

U1	U2
6	5
5	4
4	3
3	2
2	1
1	0

3

Implement the pseudo-random number generator:

$$X_i = 16807X_{i-1} \bmod(2^{31} - 1)$$

Using the seed 1234567, run the generator for 100,000 observations. Perform a chi-square goodness-of-fit test on the resulting PRN's. Use 20 equal-probability intervals and level $\alpha = 0.05$. Now perform a runs up-and-down test with $\alpha = 0.05$ on the observvations to see if they are independent.

```
fnLCG3 <- function(seed = 1, n = 1)
{
  rands <- rep(NA, n)
  x <- seed
  modVal <- (2^31 - 1)

  for(i in 1:n)
  {
    xi <- (16807 * x) %% (modVal)
    rands[i] <- xi
    x <- xi
  }

  return(rands)
}

n=100000
rn <- fnLCG3(1234567, n)
```

1422014746
456328559
849987676
681650688
1825340118
1687465831

Chi-Square Test

```
intervals <- 20
maxRn <- max(rn)
minRn <- min(rn)
intWidth <- (maxRn - minRn) / intervals
lwr <- minRn
dfCounts <- data.frame(intID=c(), count=c())
# Bin the data ourselves, I'd guess there
# is an easier way, but this will do.
```

```
for(i in 1:intervals)
{
  upr <- lwr + intWidth
  inRange <- rn[lwr <= rn & rn < upr]
  dfCounts <- rbind(dfCounts, data.frame(intID=i, count=length(inRange)))
  # setup for next interval range
  lwr <- upr
}
# Do our own Chi-Squared test
Expected <- (100000 / intervals)
chi2 <- sum((dfCounts$count - Expected)^2 / Expected)
chi2
```

```
## [1] 14.7762
```

```
# Use built-in function to compare
chiTest <- chisq.test(dfCounts$count)
chiTest
```

```
##
##  Chi-squared test for given probabilities
##
## data:  dfCounts$count
## X-squared = 14.776, df = 19, p-value = 0.7367
```

The p-value = 0.7367029 is not less than $\alpha = 0.05$, therefore we doesn't reject the null hypothesis that the distrubtion is uniform.

intID	count
1	5069
2	5028
3	5044
4	5087
5	4948
6	4953
7	4937
8	4933
9	4900
10	4957
11	5088
12	4994
13	5076
14	5019
15	5002
16	5067
17	4981
18	4914
19	5062
20	4940

Runs Up-and-Down Test

Using the `tseries` package, we execute the Runs test (Trapletti and Hornik, 2015). First we have to construct the +/- vector. Here we simply convert to boolean.

```r
s <- rep(NA, n - 1)
for(i in 1:n - 1)
{
  s[i] <- rn[i] < rn[i + 1]
}

runsTest <- runs.test(as.factor(s))
runsTest
```

```
##
##  Runs Test
##
## data:  as.factor(s)
## Standard Normal = 105.84, p-value < 2.2e-16
## alternative hypothesis: two.sided
```

Based on the p-value < 0.05, we reject the null hypothesis and conclude there is not evidence to support independence in the psuedo-random numbers.

4.

Give inverse-transforms, composition, and acceptance-rejection algorithms for generating from the following density:

$$f(x) = \begin{cases} \frac{3x^2}{2} & -1 \le x \le 1 \\ 0 & otherwise \end{cases}$$

Inverse-Transforms

First find the definite integral of the probability density function:

$$F(x) = \int \frac{3x^2}{2} dx = \frac{x^3}{2}$$

Next set $F(x) = R$ and solve for x in terms of R:

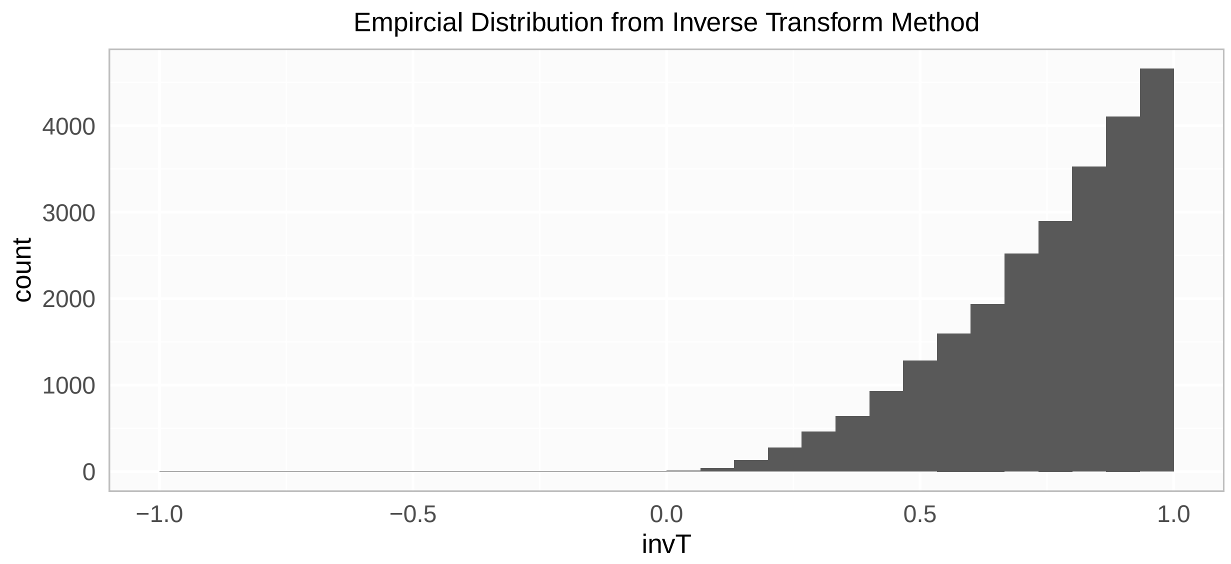
$$\frac{x^3}{2} = R$$

$$x^3 = 2R$$

$$x = \sqrt[3]{2R}$$

```r
# Define a function of the F^-1(X)
invTfn4 <- function(r)
{
  vals <- (2 * r)^(1/3)
  return (vals)
}

# Generate the uniform psuedo-random vars
rVals <- runif(n, -1, 1)
# Convert to the desired distribution using the inverse transform method.
invTVals <- invTfn4(rVals)
```

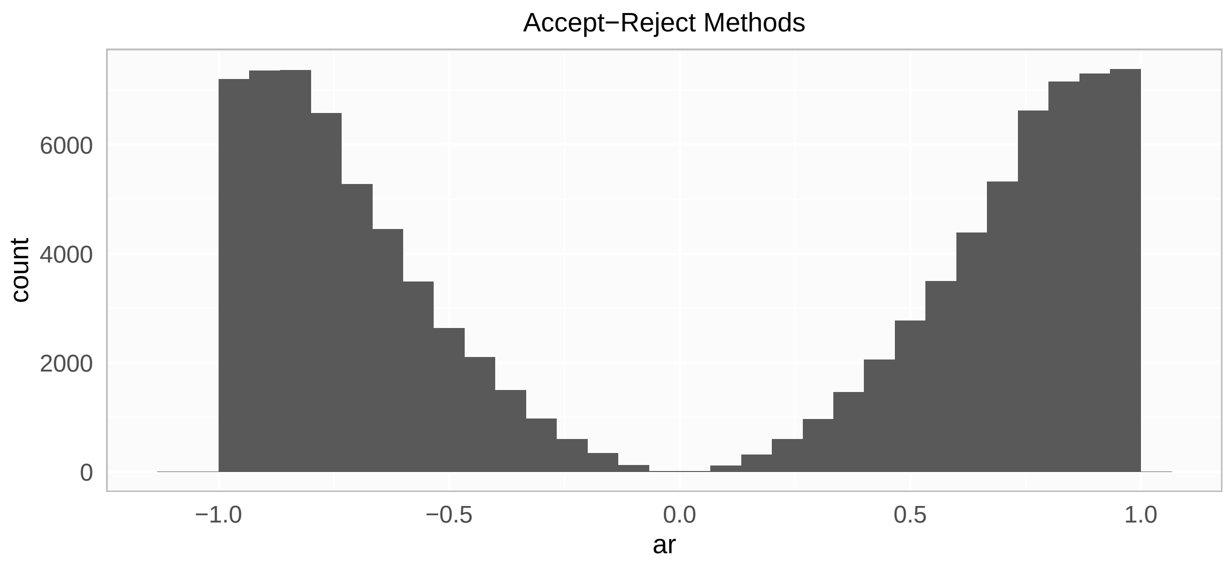
```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.

## Warning: Removed 74997 rows containing non-finite values (stat_bin).

## Warning: Removed 2 rows containing missing values (geom_bar).
```



Acceptance-Rejection Method

```r
# Helper function for the Accept/Reject approach
myRARmethod <- function(fun, min, max)
{
  M <- 2
  accepted <- FALSE
  while(!accepted)
  {
    # Get a random value from uniform distrubtion (g(x) for us)
    r <- runif(1, min, max)

    # Sample x from g(x) and u from U(0,1) (the uniform distribution over the unit interval)
    u <- runif(1, 0, 1)
    gx <- dunif(r, min, max)

    # Check whether or not u<f(x)/Mg(x).
    if(u < fun(r) / (M * gx))
    {
      accepted = TRUE
    }
  }

  return(r)
}

# Define a function for the PDF
Arfn4 <- function(x)
{
  if(-1 <=x && x <= 1)
  {
    val <- (3 * x^2) / 2
  }
  else
  {
    val = 0
  }
  return (val)
}

# Loop to generate the values
rarVals <- rep(NA, n)
for(i in 1:n)
{
  rarVals[i] <- myRARmethod(Arfn4, -1, 1)
}
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



References

Trapletti, A. and K. Hornik. tseries: Time Series Analysis and Computational Finance. R package version 0.10-34. 2015. URL: http://CRAN.R-project.org/package=tseries.