Stat 405/705
Class 9
Statistical computing with R

Richard P. Waterman

Wharton

Table of contents I

Today's module

Topics to be covered in this module:

- Last time
- Simulation modeling
- Use cases
  - Data generation
  - Methodology: performance and comparison (bias, variance, coverage for confidence intervals)
  - Monte Carlo: obtain the cumulative distribution function of a feature of interest
- Functions used in today's class
- Next time

Last time

- Case study

Probability distributions

- There are many probability distributions available to model outcomes.
- The trick is to match the distribution to the problem at hand.
- Knowing the *characterization* of the probability distribution is helpful to do this matching.

Core discrete distributions

Distribution	Parameters	R command	Characterization
Bernoulli	π	`rbinom`	A dichotomous outcome
Binomial	n, π	`rbinom`	The number of success in n independent trials

Core discrete distributions

Distribution	Parameters	R command	Characterization
Poisson	λ	`rpois`	The number of events in a fixed time interval
Hypergeometric	m, n, k	`rhyper`	The number of white balls drawn in k trials without replacement, from an urn with m white and n black balls in the population
Geometric	π	`rgeom`	The number of failures before the first success, where the success probability is π.

Binomial does sampling with replacement and the hypergeometric without replacement.

Core discrete distributions

Example R usage:

```
#### Ten realizations:
set.seed(20160411)
rbinom(n = 10, size = 1,prob = 0.5) #10 Bernoullis with probability of succ

## [1] 1 0 0 1 1 1 0 0 0 1

rbinom(n = 10, size = 5, prob =  0.5) #10 Binomials, size = 5, with probabi

## [1] 4 4 4 4 4 3 4 3 3 3

rpois(n = 10,lambda = 4) # 10 Poisson with mean  = 4

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

Core discrete distributions

Example R usage:

```
#### Ten realizations:
                 # 10 hypergeometric rvs. Draw 3 from an urn with
rhyper(nn=10,m=5,n=5,k=3) # 5 whites and 5 blacks, and count the whites.

##  [1] 2 2 2 1 1 1 3 1 1 2

rgeom(n = 10,prob = 0.2)    # 10 geometrics, with probability of success 0.


##  [1]  0  4 10  0  1  7  4  5  1  0
```

Core continuous distributions

Distribution	Parameters	R command	Characterization
Normal	μ, σ	`rnorm`	A normal outcome
Uniform	a, b	`runif`	Every outcome in the range (a,b) is equally likely
Beta	α, β	`rbeta`	Generalization of the uniform, values in 0 to 1 (good for proportions and percentages)

Core continuous distributions

Be careful. Most of these distributions have multiple parameterizations. For example, some use λ and others $1/\lambda$. Check carefully to make sure that the R parameterization matches what you think it should be.

Distribution	Parameters	R command	Characterization
Exponential	λ	`rexp`	The waiting time until an event happens. Lack of memory between events
Gamma	λ, r	`rgamma`	Generalization of the exponential (sum of exponentials)
Weibull	λ, k	`rweibull`	Another generalization of the exponential, used as a failure time distribution. Allows for changes in failure rates.

Core continuous distributions

Example R usage:

```r
#### Ten realizations:
rnorm(n=10,mean=5,sd=2) # 10 normals with mean 5, sd 2.

##  [1] 2.095082 5.161096 1.147631 3.913278 1.261207 7.781277
##  [7] 2.912268 2.681030 2.037521 5.148620

runif(n=10,min=2,max=4) # 10 uniforms between 2 and 4.

##  [1] 2.031147 3.738294 3.716006 3.244911 3.772819 3.977030
##  [7] 2.238181 3.154317 2.361134 2.038861

rbeta(n=10,shape1=2,shape2=3) # 10 betas with shape parameters 2 and 3.

##  [1] 0.6165977 0.2508416 0.4112645 0.2762560 0.3917938
##  [6] 0.6152192 0.2944554 0.7082962 0.2838051 0.1163324
```

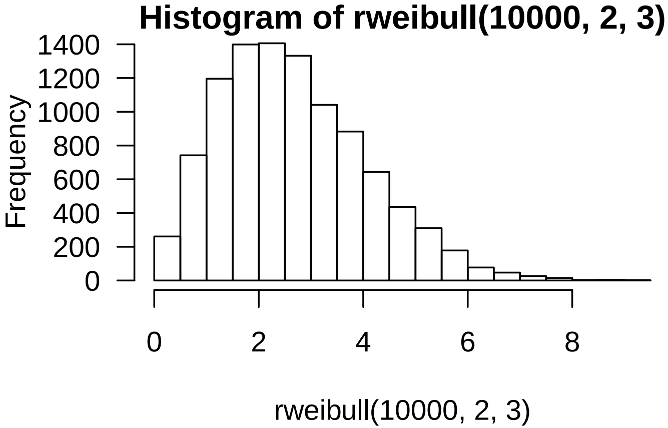
Core continuous distributions

Example R usage:

```r
rexp(n=10,rate=3) # 10 exponentials with rate 3.

##  [1] 0.63671125 0.17456099 0.23605418 0.65223466 0.03652439
##  [6] 0.17062203 0.04119400 0.15128546 1.25490879 0.66403117

rgamma(n=10,shape=2,rate=3) # 10 gammas with shape 2 and rate 3

##  [1] 0.4270519 0.5453671 1.3480656 0.6929860 0.3232398
##  [6] 0.4357640 0.7607179 0.1740898 0.2198849 1.4816516

rweibull(n=10,shape=2,scale=3) # 10 Weibulls with shape 2 and scale 3

##  [1] 0.2792830 6.0183830 2.0825576 3.7892457 2.5225584
##  [6] 0.9200342 3.1325152 2.2694199 0.8441547 2.8635047
```

Plotting lets you know what you have generated

```r
hist(rweibull(10000,2,3))    # Have a look at the distribution
```



Mixtures of distributions

- We sometimes let the parameter of one distribution be the realized value of a random variable from another distribution.
- There are some very special combinations of distributions called conjugate pairs, that are mathematically very elegant.
- But with Monte Carlo, realism can be allowed trump elegance.

Mixtures of distributions

- This is a particularly common set-up where we have repeat observations on the same individual/experimental unit.
- We give each subject their own subject specific parameter (λ_i), but let λ_i itself come from a mixing distribution.
- This typically [1]induces dependence between the repeat subject level observations.
- Common in marketing analyses where we may want to model customer level behaviour.

[1]see Canvas document, corrleation.pdf in the Misc folder

Commonly used mixtures of distributions

Subject distn.	Mixing distn.	Outcome distn.
Normal (μ_i, σ)	$\mu_i \sim N(\tau, \sigma_0)$	Normal: `rnorm`
Binomial (π_i)	$\pi_i \sim \mathit{Beta}$	Beta-binomial: `rbetabinom`
Poisson (λ_i)	$\lambda_i \sim \mathit{Gamma}$	Negative Binomial: `rnbinom`

Insurance example

- How many accidents does a driver have a year?
- For a given individual it is natural to think of the outcome being distributed as a Poisson random variable.
- But some individuals have more or less of a personal predisposition for getting into an accident. Think of comparing an individual who drinks and drives compared to someone who never drives above 40mph.
- So it makes sense to allow heterogeneity of the yearly accident rate across individuals.
- We will use a Gamma distribution for the personal accident rate.
- If the mean personal accident rate rate is 1, with a standard deviation of 1.5, this relates to gamma parameters (in R's parameterization) of `shape = 4/9` and `rate = 4/9`.
- You work this out by using the fact that the mean of a Gamma is `rate/shape` and the variance is $rate/shape^2$, so if you have the mean and the variance, you can back out the rate and the shape.

Insurance example

Plan:

- Simulate the number of accidents in a single year of 1000 drivers.
- First draw 1000 personal accident rates from a gamma distribution.
- This gives each person their own individual accident rate.
- For each person, generate a Poisson number of accidents using their personal accident rate.

Insurance example

- We will exploit the fact that we can use vectors for the argument values in the `rpois` command.
- If λ is a single value (scalar) then all realization get the same mean.
- But if λ is a vector, then each realization gets a different mean based on the λ vector.

```
rpois(n=10,lambda = 1) # Generate 10 Poissons, all with mean 1.

##  [1] 3 3 1 2 2 0 1 1 1 0

# generate 10 Poissons each with their own mean.
rpois(n=10,lambda=c(1,2,3,4,5,6,7,8,9,10))

##  [1]  0  0  7  6  3  7  3 12  6  9
```

Insurance example

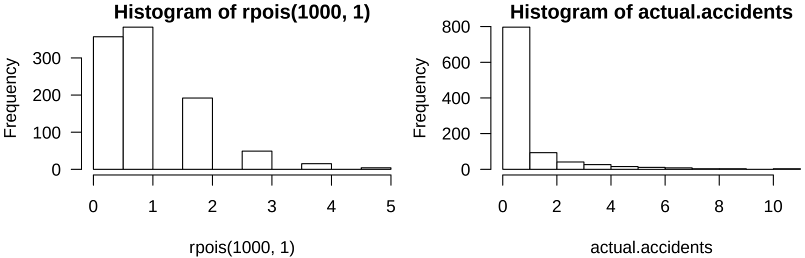
Implementation in R for a year of accidents for these 1000 people:

```r
n.people  <- 1000 # The number of people

pers.acc.rate <- rgamma(n = n.people,
                        shape = 4/9, rate = 4/9) #Personal accident rates
actual.accidents <- rpois(n = n.people,
                          lambda = pers.acc.rate) #Actual accidents
```

Insurance example

Comparing a pure Poisson to a Poisson mixture:

```r
hist(rpois(1000,1))
hist(actual.accidents)
```



By incorporating heterogeneity, we generate the long tail, commonly seen in practice.

Multivariate distributions

A multivariate distribution produces a **vector** of outcomes for each realization, for example, purchase activity on a basket of goods, or returns on a correlated portfolio of stocks.

Distribution	Parameters	Comment	R command
Multivariate normal	Mean vector: μ, var/covariance $\mathbf{\Sigma}$	Allows correlation between individual normals	`mvrnorm`
Multinomial	Probability vector π, trials n	Extends the binomial to more than just success/failure	`rmultinom`

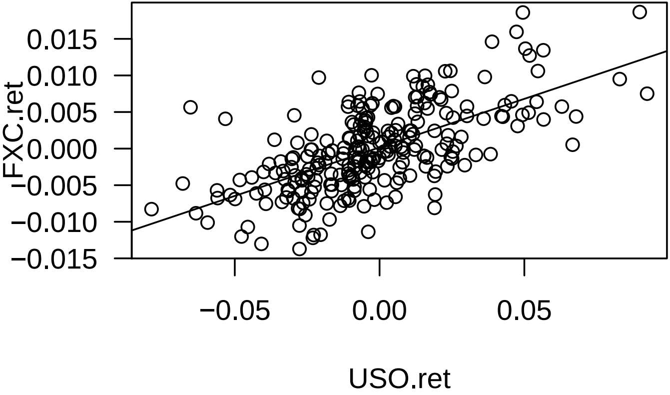
Multivariate distributions

Generating correlated returns.
We are interested in oil and the CAD exchange rate against a basket of currencies. USO v. FXC.

```
#Read in some data
my.fx <- read.csv(file=
         'C:/Users/richardw/Dropbox (Penn)/Teaching/705s2019/Data/FX_OIL_CA
#Create returns
USO.ret <- diff(my.fx$USO)/my.fx$USO[-length(my.fx$USO)]
FXC.ret <- diff(my.fx$FXC)/my.fx$FXC[-length(my.fx$FXC)]
cor(USO.ret,FXC.ret) # The correlation between returns

## [1] 0.6369186
```

- Canada is a natural resource country in general, and oil in particular. When oil does well, its currency does well.
- If we want to simulate returns from this *pair* then we need to include the correlation feature.

```r
plot(USO.ret,FXC.ret)
lm.out <- lm(FXC.ret ~ USO.ret) # Find the regression line
abline(lm.out) # add it to the plot
```



Multivariate distributions

To sample from such a distribution of returns, assuming normality, we need the mean vector of returns and the variance covariance matrix:

```
mean.uso <-  mean(USO.ret) # Mean of USO returns
mean.fxc <-  mean(FXC.ret) # Mean of FXC returns
covar.uso.fxc <- var(cbind(USO.ret,FXC.ret)) # The var/covar matrix
mean.uso

## [1] -0.001951381

mean.fxc

## [1] -9.312244e-05

covar.uso.fxc

##              USO.ret      FXC.ret
## USO.ret 0.0007807878 1.035696e-04
## FXC.ret 0.0001035696 3.386596e-05
```

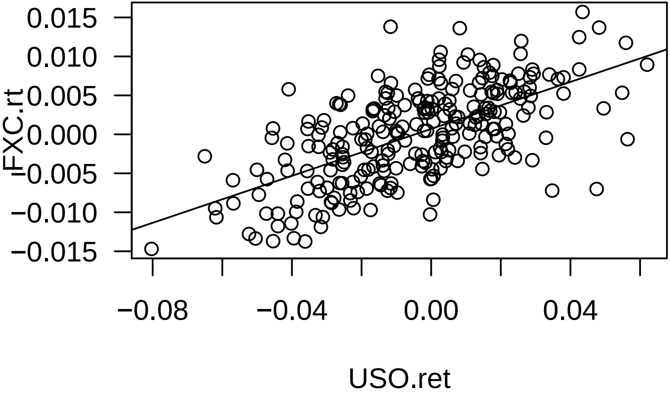
Multivariate distributions

To generate a multivariate sample we use the `mvrnorm` command:

```
library(MASS) #More on libraries later
n.days <- 250
rets <- mvrnorm(n.days, mu = c(mean.uso, mean.fxc), Sigma = covar.uso.fxc)
apply(rets,2,mean)

##      USO.ret      FXC.ret
## -0.004527970  0.000020013

 #We get the right mean
var(rets)  # We get the right var/covar structure

##              USO.ret      FXC.ret
## USO.ret 6.498146e-04 9.786227e-05
## FXC.ret 9.786227e-05 3.704597e-05
```

Multivariate distributions

```
plot(rets[,1],rets[,2],xlab="USO.ret",ylab="FXC.rt") # Plot the simulated r

lm.out.two <- lm(rets[,2] ~ rets[,1]) # Find the regression line
abline(lm.out.two) # add it to the plot
```



The sample command

- Sampling from an arbitrary distribution. Draw samples from a population.
- Call the population x.
- The number of samples n.
- Decide to sample with or without replacement.
- Set probability weights to apply to the population elements. By default, all elements have equal probability.

The sample command

Examples:

```
sample(x=10) # A random permutation of the numbers 1:10

##  [1] 10  3  6  7  1  2  4  8  5  9

# sample of size 20 from the numbers 1 though 10
sample(x=10,size = 20, replace=TRUE)

##  [1]  6  9  8  6  8 10  2  6  1  9 10  8 10 10  9  8 10 10
## [19]  5  5

sample(x=10,size = 20, replace=FALSE) # Bombs

## Error in sample.int(x, size, replace, prob):  cannot take a sample
larger than the population when 'replace = FALSE'
```

The sample command

```r
#Sample 20 observations, from 3 categories with specific probabilities
sample(x = c("YES","NO","MAYBE"),
       size= 20,
           replace=TRUE,
           prob = c(0.1,0.2,0.7)
)

##  [1] "MAYBE" "MAYBE" "YES"   "MAYBE" "MAYBE" "MAYBE" "NO"
##  [8] "MAYBE" "YES"   "MAYBE" "NO"    "YES"   "NO"    "MAYBE"
## [15] "NO"    "NO"    "MAYBE" "MAYBE" "MAYBE" "MAYBE"
```

Summarizing the results of a simulation

- The most complete summary of a probability distribution is called the Cumulative Distribution Function (CDF).
- It is defined as:

$$F_X(x) = P(X \leq x).$$

- From this you can calculate everything.
- The cumulative probabilities are calculated with the prefix `p` to the random number generation commands, instead of `r`.

```
# What's the probability that a Z is less than -1?
pnorm(q = -1, mean = 0, sd = 1)

## [1] 0.1586553
```

The CDF

- Quantiles work in the other direction. What value of x has a given probability that X is less than or equal to x?
- The quantiles are calculated with the prefix `q` to the random number generation commands, instead of `r`.

```
# What value of x has 0.025 probability to the left of it?
qnorm(p = 0.025, mean = 0, sd = 1)

## [1] -1.959964
```

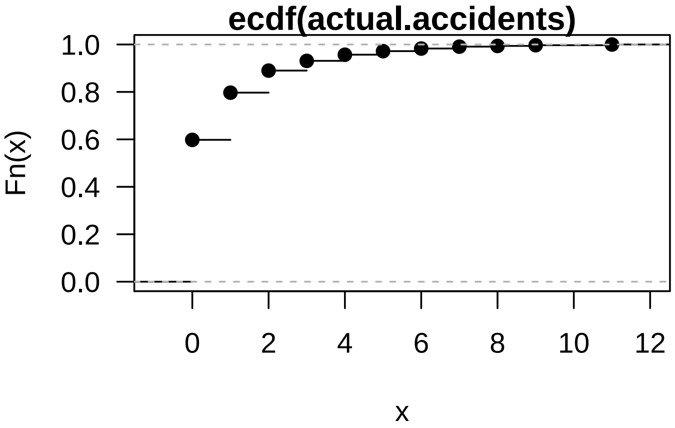
The Empirical Cumulative Distribution Function

In practice we rarely have the true CDF, so we estimate it with data from the simulation. The command to do this is `ecdf`.
Using the results from the insurance example we have:

```
summary(ecdf(actual.accidents))

## Empirical CDF:   11 unique values with summary
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   0.000   2.500   5.000   5.091   7.500  11.000
```

The Empirical Cumulative Distribution Function

```
plot(ecdf(actual.accidents))
```



Quantiles

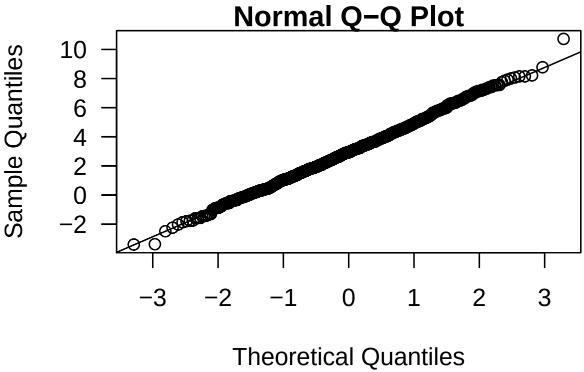
Quantiles (otherwise known as percentiles) of the ECDF are found using the `quantile` command (this is how we will make confidence intervals):

```
#Find the 50th and 90th and 99th percentiles
quantile(x = actual.accidents, probs= c(0.5,0.9,0.99))

## 50% 90% 99%
##   0   3   7
```

Normal quantile plots

You may be familiar with a Normal Quantile Plot, to see how close a distribution is to normality.

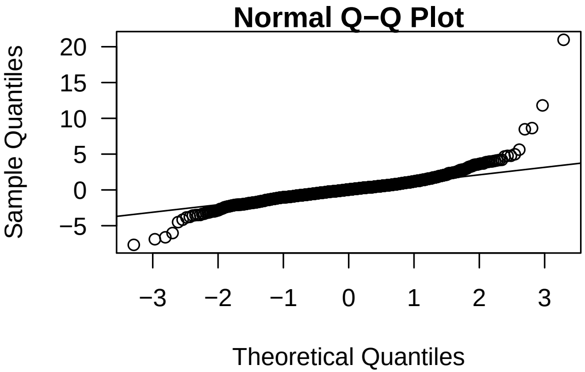
```
x <- rnorm(n=1000,mean = 3,sd = 2) # Create 1000 normals and plot them
qqnorm(x) # The normal quantile plot
qqline(x) # Add the reference line
```



Normal quantile plots

Now create a heavy tailed distribution for comparison. We will use a *t-distribution* on 3 degrees of freedom.

```
x <- rt(n=1000,df = 3) # Create 1000 t-3s
qqnorm(x) # The normal quantile plot
qqline(x) # Add the reference line
```



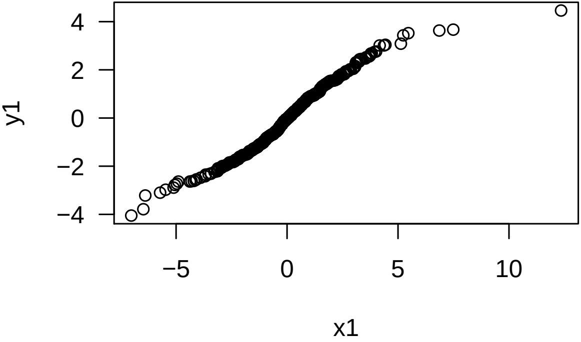
QQ-plots

- Normal quantile plots are a special case of the more general QQ-plot, the Quantile-Quantile plot.
- You can compare any two distributions to see if their quantiles match up.
- If the two distributions are the same, then the points in the QQ-plot should follow a straight line.

Quantile plots

These can be useful in randomized experiments, for checking that the cases and controls have the same distribution for all background variables. Example:

```
x1 <- rt(n=1000,df = 3) # Create 1000 t-3s
y1 <- rt(n=1000,df = 10) # Create 1000 t-10s
qqplot(x1,y1) # The qq plot
```



Module summary

Topics covered today include:

- Random number generation
- Mixtures of distributions
- CDFs
- Quantiles

Next time

- Run Monte Carlo simulations.

Today's function list

Do you know what each of these functions does?

```
ecdf
pnorm
qnorm
qqline
qqnorm
qqplot
rnorm
sample
```