Alzheimer's Data Analysis

Babak Shahbaba and Sam Behseta
UC Irvine and California State University Fullerton

July and August, 2022

Data Science in
Biomedical Sciences
UCI COSMOS

Let's Begin!

- Today we will discuss the role of distributions in the context of the analysis of Alzheimer's data.

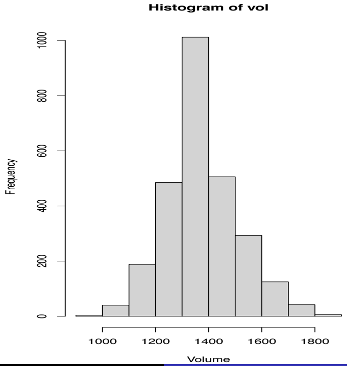
Let's Begin!

- Today we will discuss the role of distributions in the context of the analysis of Alzheimer's data.
- I would like to start, somewhat unusually, with the *Normal* or *Gaussian* distribution.

Let's revisit *vol*!

- Recall that *vol* represents the intercarnial volume. Below we calculate its mean and standard deviation and recreate its histogram.

```
vol.m=mean(vol)
vol.sd=sd(vol)
vol.m
vol.sd
hist(vol,xlab="Volume",breaks=10)
```



Calculating Some Proportions With the Original Data

- Proportion of patients whose *vol* measurement is above or below 1000:

```
length(vol[vol>1000])
length(vol[vol<=1000])

l=length(vol)
length(vol[vol<=1000])/l
```

Calculating Some Proportions With the Original Data

- Proportion of patients whose *vol* measurement is above or below 1000:

  ```
  length(vol[vol>1000])
  length(vol[vol<=1000])

  l=length(vol)
  length(vol[vol<=1000])/l
  ```

- Proportion of *vol* between 1300 and 1600:

  ```
  length(vol[vol>=1300 & vol <=1600])/l
  ```

Calculating Areas Under a Theoretical Gaussian Curve

- We pretend there is an underlying Normal model with the same mean and standard deviation that might have generated this data.

Calculating Areas Under a Theoretical Gaussian Curve

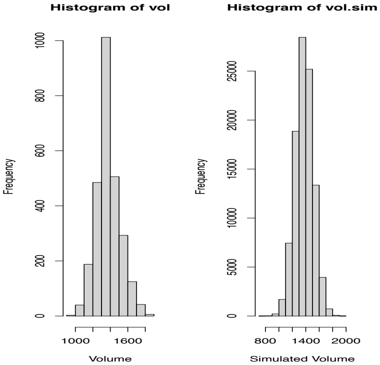
- We pretend there is an underlying Normal model with the same mean and standard deviation that might have generated this data.
- Below see my attempt in repeating the calculations of the previous slide, this time through a theoretical normal distribution.

```
pnorm(1000,vol.m,vol.sd)
pnorm(1600,vol.m,vol.sd)-pnorm(1300,vol.m,vol.sd)
```

Simulating Data From the Theoretical Distribution

- Now let's contrast the original data versus the simulated data!

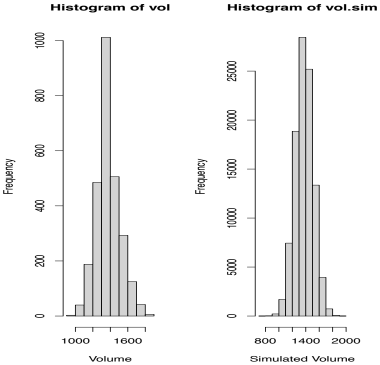
```
vol.sim=rnorm(100000,vol.m,vol.sd)
par(mfrow=c(1,2))
hist(vol,xlab="Volume",breaks=10)
hist(vol.sim,xlab="Simulated Volume",breaks=10)
```



Simulating Data From the Theoretical Distribution

- Now let's contrast the original data versus the simulated data!

```
vol.sim=rnorm(100000,vol.m,vol.sd)
par(mfrow=c(1,2))
hist(vol,xlab="Volume",breaks=10)
hist(vol.sim,xlab="Simulated Volume",breaks=10)
```



- How did we do? Hard to tell!

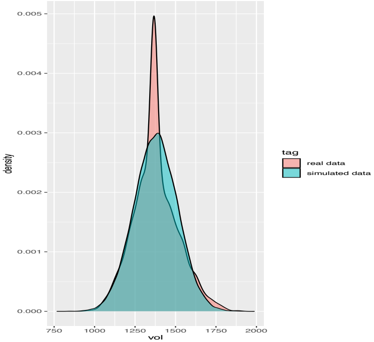
Simulating Data From the Theoretical Distribution

- How did we do?

```
par(mfrow=c(1,1))

vol.frame <- data.frame(vol)
vol.sim.frame <- data.frame(vol.sim)
colnames(vol.sim.frame)[colnames(vol.sim.frame)=="vol.sim"]<-"vol"

vol.frame$tag<-"real data"
vol.sim.frame$tag<-"simulated data"

volTot<-rbind(vol.frame,vol.sim.frame)
head(volTot)

ggplot(volTot, aes(vol, fill = tag))+
 geom_density(alpha = 0.5)
```

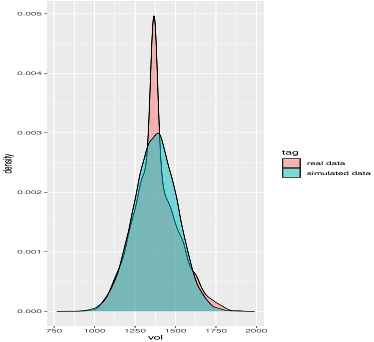
Simulating Data From the Theoretical Distribution

- How did we do?

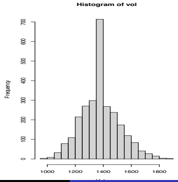


Simulating Data From the Theoretical Distribution

- How did we do?



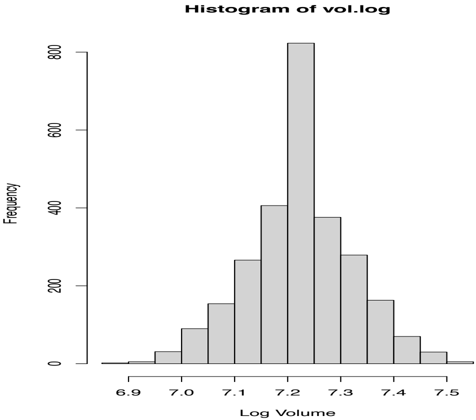
- Why? Let's look at the original data, one more time!



Simulating Data From the Theoretical Distribution

- No fears though! Data scientist always has a trick or two up her sleeves!

  ```
  vol.log=log(vol)
  hist(vol.log,xlab="Log Volume",breaks=15)
  ```



Verifying the 68-95-99.7 Rule for Normal Distributions

- Check out the code below:

```
m.sim=mean(vol.sim)
sd.sim=sd(vol.sim)
l.sim=100000

length(vol.sim[vol.sim>=m.sim-sd.sim & vol.sim<=m.sim+sd.sim])/l.sim
length(vol.sim[vol.sim>=m.sim-2*sd.sim & vol.sim<=m.sim+2*sd.sim])/l.sim
length(vol.sim[vol.sim>=m.sim-3*sd.sim & vol.sim<=m.sim+3*sd.sim])/l.sim
```

Class Activity 1 – Group Work

- Repeat the steps resulting in the juxtaposed histograms of the log volume and its simulated version from a hypothetical normal.
- Exchange your work with the nearest team around you. Discuss and compare their work and grade their write ups (on a scale of 0 to 100)!

Simulating From the Uniform Distribution

- **Continuous Uniform**

  ```
  a=runif(10000,0,10)
  hist(a,breaks=10)
  ```

Simulating From the Uniform Distribution

- **Continuous Uniform**

  ```
  a=runif(10000,0,10)
  hist(a,breaks=10)
  ```

- **Discrete Uniform**

  ```
  b=seq(1,10,1)
  c<-sample(b,1000000,replace=TRUE)
  d=table(c)
  d
  barplot(d)
  ```

Simulating From the Uniform Distribution

- **Continuous Uniform**

```
a=runif(10000,0,10)
hist(a,breaks=10)
```

- **Discrete Uniform**

```
b=seq(1,10,1)
c<-sample(b,1000000,replace=TRUE)
d=table(c)
d
barplot(d)
```

- How can you simulate discrete uniform with *runif*?

Bernoulli and Binomial Distributions

- Simulating a sequence of 0's and 1's with the probability of success p=0.5.

```
bern<-rbinom(1000,1,0.5)
bern.table<-table(bern)
barplot(bern.table)
```

Bernoulli and Binomial Distributions

- Simulating a sequence of 0's and 1's with the probability of success p=0.5.

  ```
  bern<-rbinom(1000,1,0.5)
  bern.table<-table(bern)
  barplot(bern.table)
  ```

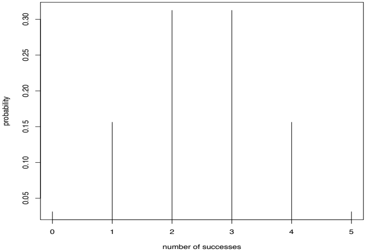
- Simulating a sequence of binary outcomes with $n = 5$ the probability of success $p = 0.5$.

  ```
  binom<-rbinom(1000,5,0.5)
  binom.table<-table(binom)
  barplot(binom.table)
  ```

A Bit More on Binomial Distribution

- Let's understand the difference between *dbinom*, *pbinom*, and *qbinom*. Suppose we are interested in a binomial with $n = 5$, and $p = 0.5$

```
dbinom(0,5,.5)
dbinom(0,5,0.5)+dbinom(1,5,0.5)
pbinom(1,5,0.5)
qbinom(0.1875,5,0.5)
pbinom(-1:5,5,0.5)
plot(0:5,dbinom(0:5,5,0.5),type="h",ylab="probability",
xlab="number of successes")
```


probability
number of successes