

# Iteration & Likelihood

## Review

- ▶ We test hypotheses using  $P(x \leq Data|H)$
- ▶ We can fit models, then test them in this framework
- ▶ We are awfully good at for loops & iteration

## Today

1. Beyond For Loops!
2. Likelihood via iteration
3. Point estimates of probability from data

## For Loops!

```
x<-1:100000
#
for (i in x){
  x[i]<-x[i] +1
}
```

## For Loops: Costs & Benefits

### Benefits:

1. Remove repetitive code
2. Can *map* same operation across a vector, matrix, list, etc.

### Costs:

1. Slow.
2. Lots of Code.

## Better than For Loops!

- ▶ Write code that uses vectors - VECTORIZATION
- ▶ The Apply family

## Speed: Many Operations Faster via *Vectorization*

```
1:10 + 1
```

```
# [1] 2 3 4 5 6 7 8 9 10 11
```

## Many Operations Faster via *Vectorization*

```
system.time(1:100000+1)

# user system elapsed
# 0.001 0.001 0.000

system.time({
  x<-1:100000
  for(i in x) x[i]<-x[i] +1
})

# user system elapsed
# 0.116 0.004 0.119
```

## Vectorization Everywhere

```
dnorm(5, mean = 1:10, sd = 1)

# [1] 1.338e-04 4.432e-03 5.399e-02 2.420e-01 3.989e-01
# [6] 2.420e-01 5.399e-02 4.432e-03 1.338e-04 1.487e-06
```

## Minimizing Code for Mapping Functions: the Apply Family

Take an object type - vector, matrix, list, etc., and map a function to every element, cleanly and quickly.

## sapply for Vectors

```
f <- function(x) x+1
#
sapply(1:5,f)
# [1] 2 3 4 5 6
```

## Anonymous Functions and sapply

```
sapply(1:5, function(x) x+1)
# [1] 2 3 4 5 6
```

## Apply Statements Faster than For Loops

```
system.time({
  x<-1:100000
  for(i in x) x[i]<-x[i] +1
})

#   user  system elapsed
# 0.115   0.003   0.118

system.time(sapply(1:100000, f))

#   user  system elapsed
# 0.180   0.006   0.187
```

## Lots of Possibilities with sapply

```
sampMean <- function (vec, size) mean(sample(vec, size))

sapply(1:5, function(x) sampMean(1:5, x))

# [1] 5.000 1.500 2.667 2.750 3.000
```

## Many Output Types with `sapply`

```
sapply(1:5, function(x) return( c(x+2, x^2) ) )  
  
#      [,1] [,2] [,3] [,4] [,5]  
# [1,]    3    4    5    6    7  
# [2,]    1    4    9   16   25
```

## `apply` for Matrices

```
m <- matrix( c(1, 2,  
              3, 4), ncol=2)
```

```
apply(m, 1, sum)
```

```
# [1] 4 6
```

```
apply(m, 2, sum)
```

```
# [1] 3 7
```



## apply for Matrices

```
apply(m, c(1,2), sum)
```

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

## lapply for Lists - mclapply to use multiple cores

```
x <- list(a = 1:10,  
         beta = exp(-3:3),  
         logic = c(TRUE,FALSE,FALSE,TRUE))  
  
# compute the list mean for each list element  
lapply(x,mean)  
  
# $a  
# [1] 5.5  
#  
# $beta  
# [1] 4.535  
#  
# $logic  
# [1] 0.5
```

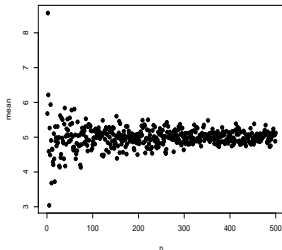
## Exercise: Apply Yourself!

1. Using `sapply`, write a short script to show the relationship between `n` and estimated mean for a normally distributed population with a mean of 5 and SD of 3.
2. Challenge: What is the bootstrapped SE at each sample size? 50 boots!

hint: 2 lines with `bootstrap`, 3-4 with nested `sapply`

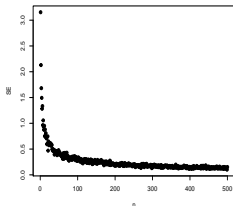
## Exercise: Apply Yourself!

```
set.seed(697)
vec <- sapply(1:500, function(x) mean(rnorm(x, mean=5, sd=3)))
```



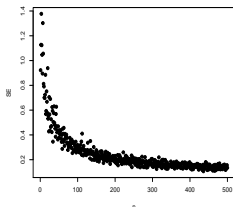
## Exercise: Apply Yourself!

```
seVec <- sapply(1:500, function(x){  
  mvec<- sapply(1:50, function(i) mean(rnorm(x, mean=5, sd=3)))  
  return(sd(mvec))  
})
```



## Exercise: Apply Yourself!

```
library(bootstrap)  
seVec2 <- sapply(1:500, function(x)  
  sd(bootstrap(rnorm(x, mean=5, sd=3), 50, mean)$thetastar))
```



# Application of Iterative Solutions: Likelihood

Likelihood: how well data support a given hypothesis.

Note: Each and every parameter choice IS a hypothesis

## Likelihood Defined

$$L(\theta|D) = p(D|\theta)$$

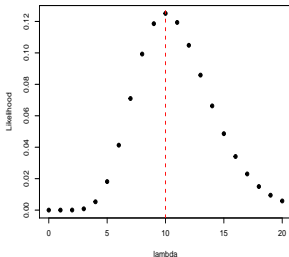
Where  $D$  is the data and  $\theta$  is some choice of parameter values

## Maximum Likelihood

The Maximum Likelihood Estimate is the value at which  $p(D|\theta)$  is highest.

## Example of Maximum Likelihood

Let's say we have counted 10 individuals in a plot. Given that the population is Poisson distributed, what is the value of  $\lambda$ ?



$$p(x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

where  $x=10$

## Enter Iteration

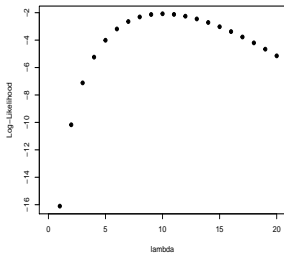
I did this by searching all values of  $\lambda$  using `sapply`

```
count <- 10
#
l <- sapply(0:20, function(x) dpois(count, lambda=x) )
#
plot(0:20, l, ylab="Likelihood", xlab="lambda", pch=19)
abline(v=10, col="red", lwd=2, lty=2)
```

## Maximum Likelihood

We often maximize log-likelihood because of 1) more well behaved ( $\chi^2$ ) properties of Log-Likelihood values and 2) rounding error

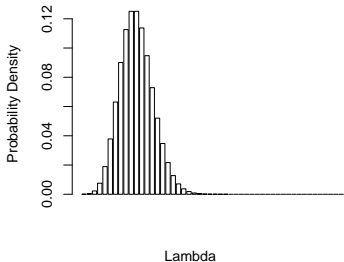
## Log-Likelihood



# What about Many Data Points?

## Start with a Probability Distribution

Density Function at Lambda = 10

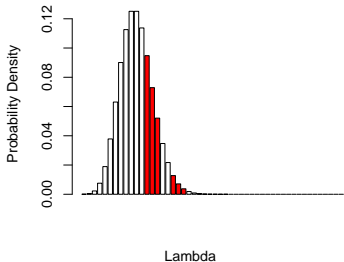


$$p(x) = \frac{\lambda^x e^{-\lambda}}{x!}$$



What is the probability of the data given the parameter?

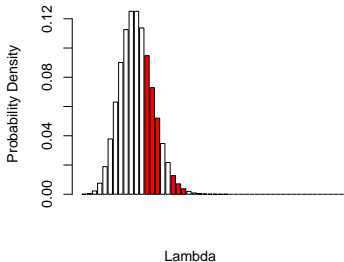
Density Function at Lambda = 10



$$p(x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

What is the probability of the data given the parameter?

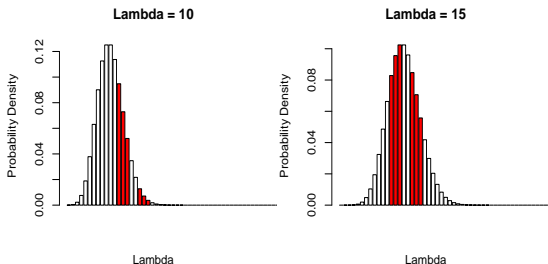
Density Function at Lambda = 10



$$p(a \text{ and } b) = p(a)p(b)$$

$$p(D|\theta) = \prod_{i=1}^n p(d_i|\theta)$$

## Can Compare $p(\text{data} \mid H)$ for alternate $H$



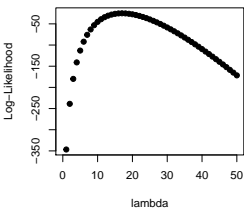
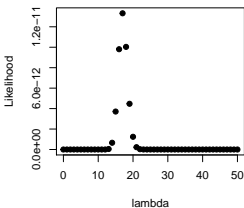
Compare  $p(D|\theta_1)$  versus  $p(D|\theta_2)$

## Calculating Likelihood and Log-Likelihood for a Data Set

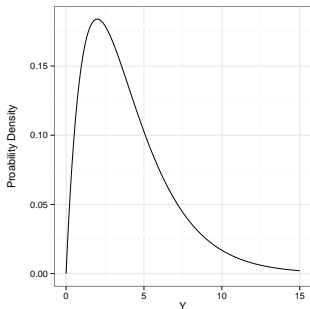
```
lik <- sapply(lambdaVals,  
              function(x) prod( dpois(counts, lambda=x) ) )  
#  
ll <- sapply(lambdaVals,  
             function(x) sum( dpois(counts, lambda=x, log=TRUE) ) )
```

```
max(lik)  
  
# [1] 1.332e-11  
  
lambdaVals[which(lik==max(lik))]  
  
# [1] 17
```

## Likelihood and Log-Likelihood of a Data Set



## The Gamma Distribution



- ▶ Defined by number of events(shape) average time to an event (scale)
- ▶  $Y \sim G(\text{shape}, \text{scale})$
- ▶ Think of time spent waiting for a bus to arrive..or time to death

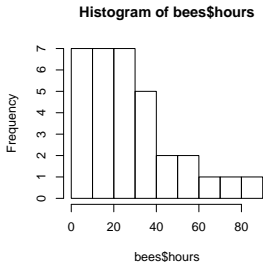
```
dgamma(Y, shape = 2, scale = 2)
```



## Exercise: Likelihood and Bee Death!

- ▶ Load the Bee Lifespan Data
- ▶ Model Bee Lifespans as a Gamma Distribution with shape = 1 (1 bee per death)
- ▶ What is the ML estimate of a Bee's Lifespan in hours?

## Exercise: Likelihood and Bees!



## Exercise: Likelihood and Bees!

```
scaleVals <- seq(0.2, 80, 0.2)
#
beeD <- function(x) sum(dgamma(bees$hours, shape=1,
                              scale=x, log=TRUE))
mll <- sapply(scaleVals, beeD)
#
scaleVals[which(mll==max(mll))]

# [1] 27.8
```

## Exercise: Likelihood and Bees!

