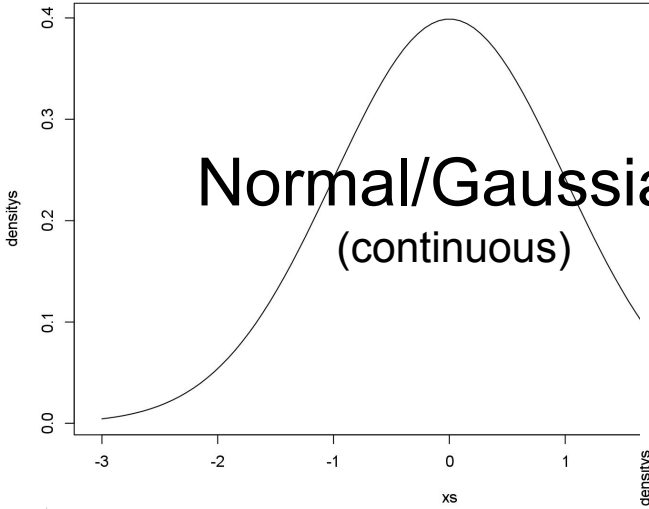
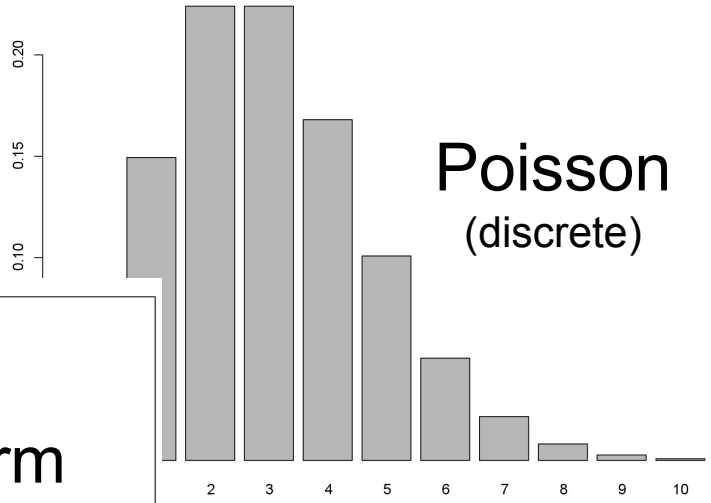
Introduction to Data Science

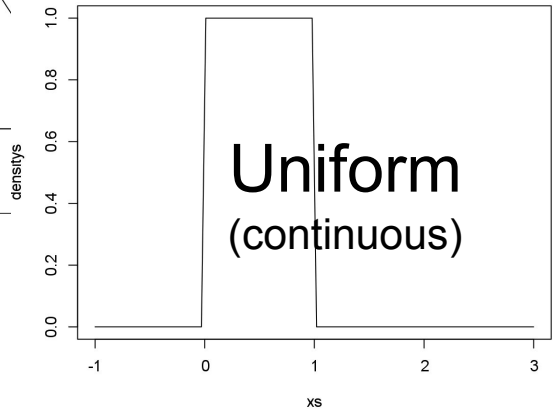
Statistic Modeling

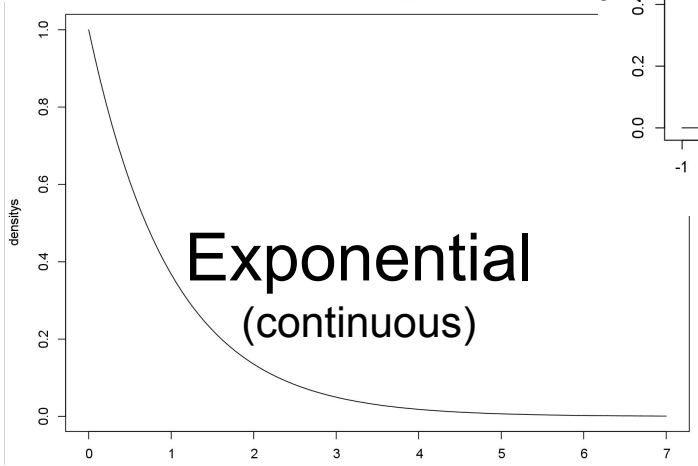
Prof. Dr. Ralf Lämmel & M.Sc. **Johannes Härtel**
(johanneshaertel@uni-koblenz.de)

Recap Distributions


Normal/Gaussian
(continuous)
densitys
xs

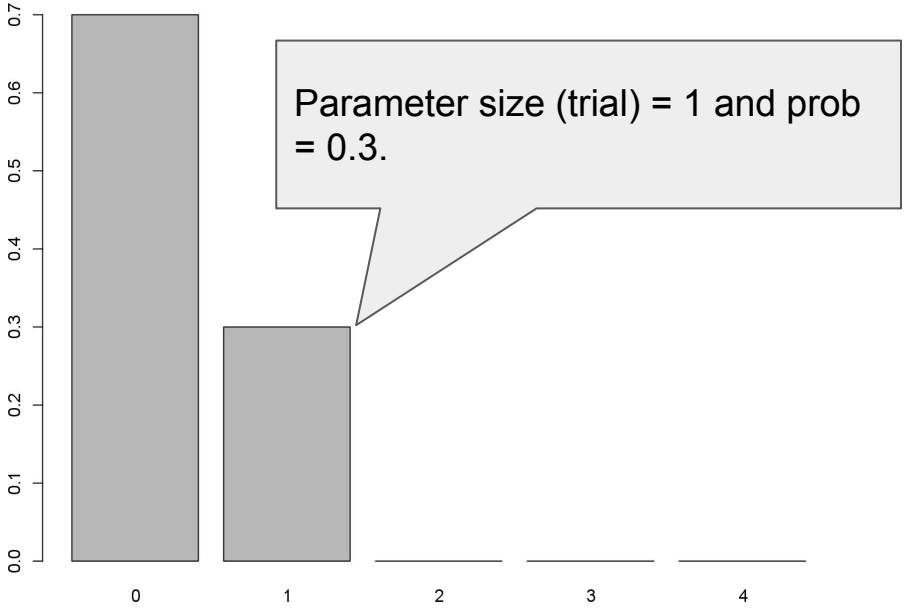

Poisson
(discrete)

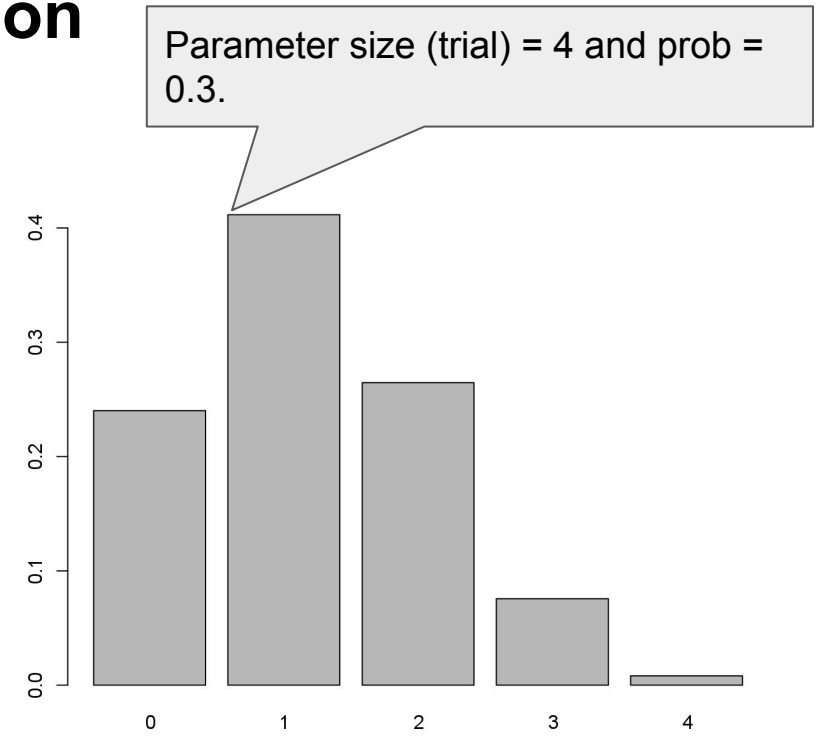

Uniform
(continuous)
densitys
xs


Exponential
(continuous)
densitys


Binomial
(discrete)

Recap: The **binomial distribution**





Recap: The **parameters** of the binomial distribution

We have two important parameters, the size (trials) and the prob parameter.

Binomial {stats} R Documentation

The Binomial Distribution

Description

Density, distribution function, quantile function and random generation for the binomial distribution with parameters `size` and `prob`.

This is conventionally interpreted as the number of 'successes' in `size` trials.

Usage

```
dbinom(x, size, prob, log = FALSE)
pbinom(q, size, prob, lower.tail = TRUE, log.p = FALSE)
qbinom(p, size, prob, lower.tail = TRUE, log.p = FALSE)
rbinom(n, size, prob)
```

Arguments

`x, q`	vector of quantiles.
`p`	vector of probabilities.
`n`	number of observations. If `length(n) > 1`, the length is taken to be the number required.
`size`	number of trials (zero or more).
`prob`	probability of success on each trial.

Recap: **Generating random numbers / sampling from** a binomial distribution

```
set.seed(1)
n <- 18

D <- rbinom(n, size = 1, prob = 0.2)
D
```

0 0 0 1 0 1 1 0 0 0 0 0 0 0 0 0 1

Recap: The **probability density / mass function**

$$Pr(X = k) = \binom{size}{k} prob^k (1 - prob)^{size-k}$$

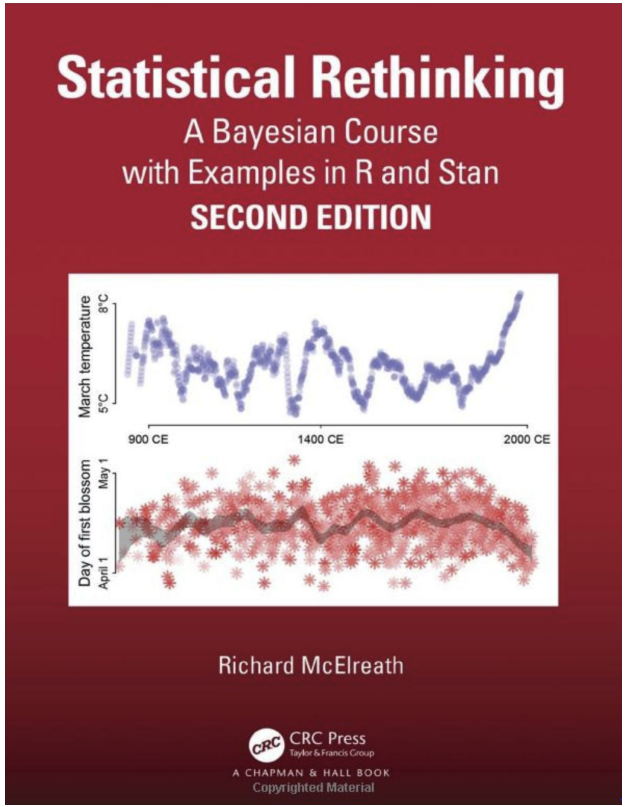
Build in as ***dbinom*** in R

$$k = 0..n$$

A Bayesian Course

(to statistic modeling)

Data Science @ Softlang — Johannes Härtel (johanneshaertel@uni-koblenz.de)



The book “Statistical Rethinking” motivates the content of this lecture. We will refer to the book as [McElreath20].

Why a Bayesian course?

- We will be able to express statistic questions in a **uniform language**.
- The language can be **executed**.
- We will **not meet new challenges** whenever a new type of model appears.
- We will **not struggle with confidence intervals and p-values** that much.
- We will be able to explicitly **specify every assumption of a model** to better understand it.
- The Bayesian approach helps to **understand (dominant) frequentists approach**.

Building a Bayesian statistic model

The abstract data story

We have data and try to come up with a story how the data came to be.

This involves:

- **Variables**
  - Observed Variables (i.e., data and likelihood)
  - Unobserved Variable (i.e., parameters)
- **Definitions** that relate variables to each other.

What we (currently) miss is a formal way to express this.

Some reasons why we might be interested in unobserved variables (the parameters):

- What is the **average difference** between treatment groups?
- How strong is the **association** between a treatment and an outcome?
- Does the **effect** of the treatment depend upon a co-variate?
- How much **variation** is there among groups?

*(potential parameters highlighted in **bold**)*

Questions copied directly from [McElreath20]

A concrete data story

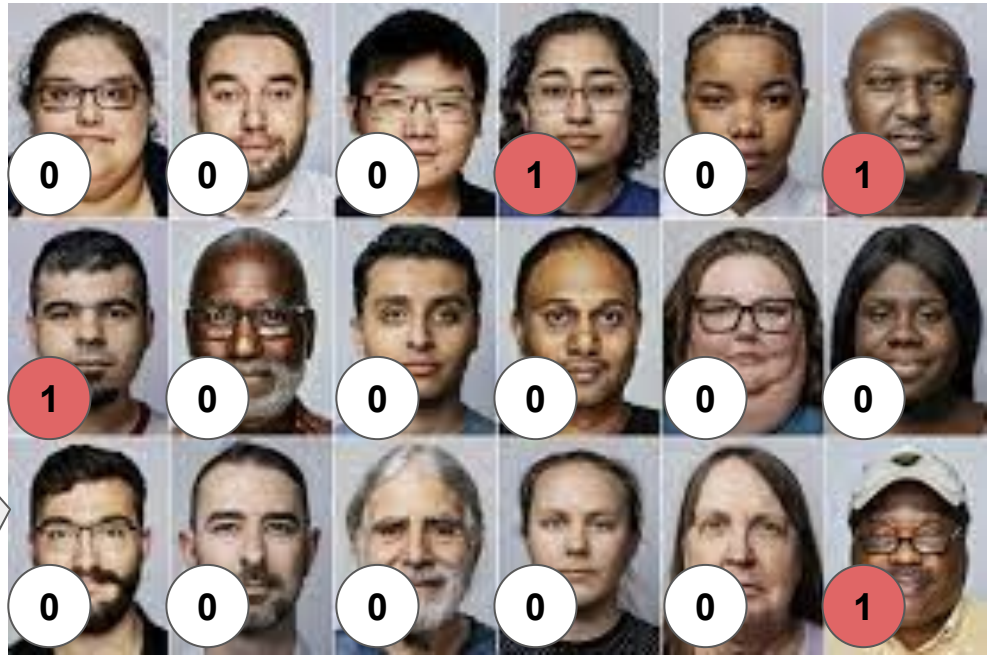
(a binomial story)

Example (Question)

We have 18 random persons (from Germany) and want to know the probability of an undetected infection with COVID-19.

Example (Question)

We have 18 random persons (from Germany) and want to know the probability of an undetected infection with COVID-19.



Again, we used **simulated data** (the data from the recap). This is better for **illustration,** but does **not answer the real question.**

The concrete data story

- Having a binomial distribution with a single trial (size) seems to be clear in our story.
- We have the data D for the 18 persons (observed variable).
- **We do not know the *prob* parameter of the distribution (unobserved variable).**

A way of writing this data story (‘math’ notation).

$$D \sim \text{Binomial}(1, \text{prob})$$

Demo

The intuitive way how a programmer would infer prob

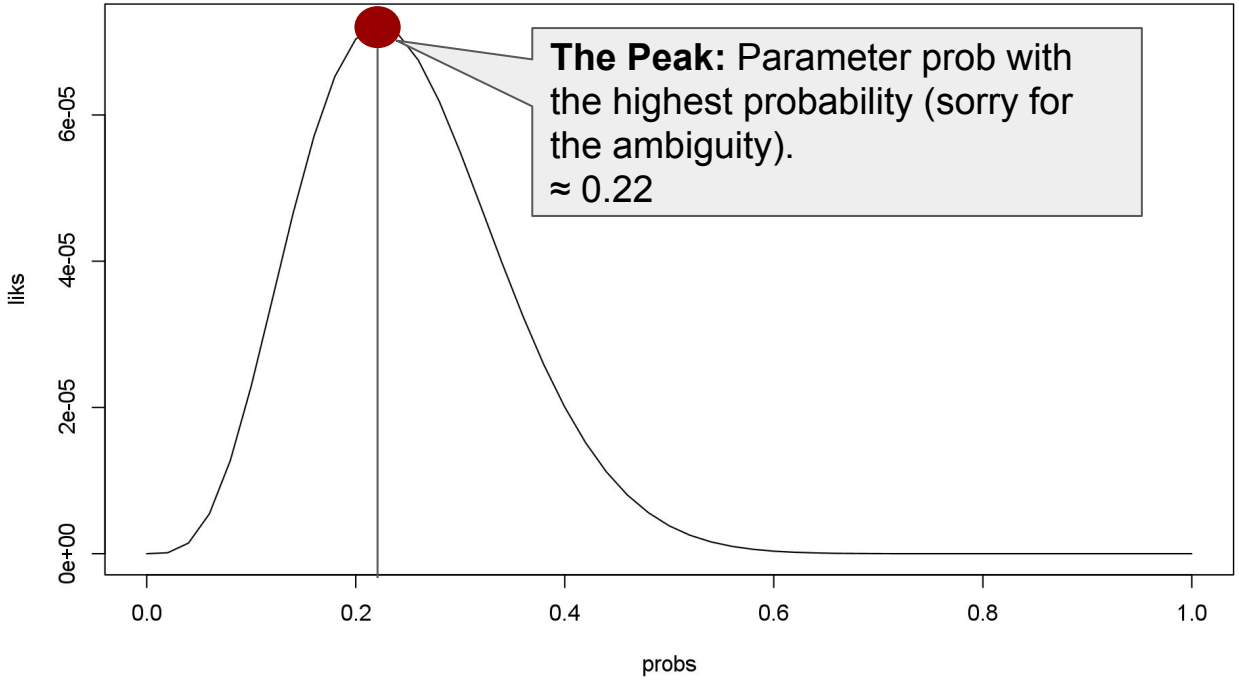
(grid approximation)

Backup grid approximation

```
# ‘D’ is our data (0,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,0,1).
# Different probs that might be possible (the grid).
probs <- seq(0, 1, length.out = 51)

# The likelihood, i.e., that prob may have been producing the
data.
likelihood <- sapply(probs, function(prob) {
 return(prod(dbinom(D, size= 1, prob)))
})

# Plot both.
plot(probs, likelihood, type = "l")
```

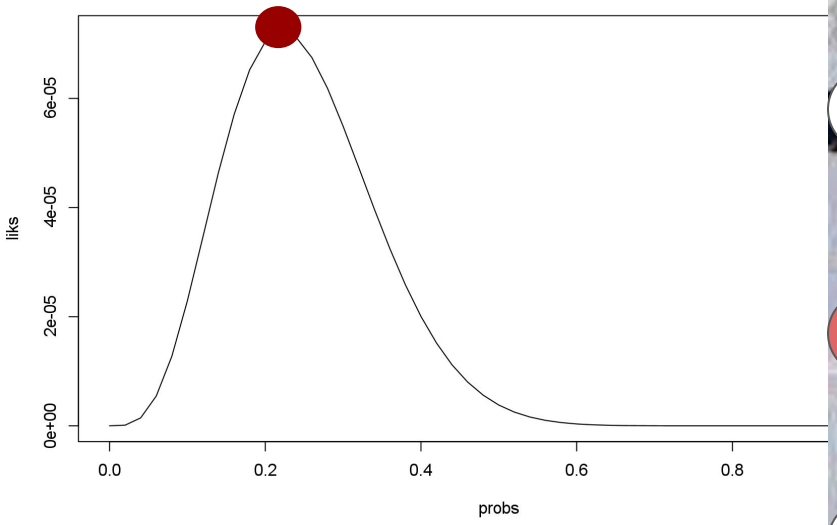
We use the probability density/mass function (PDF).

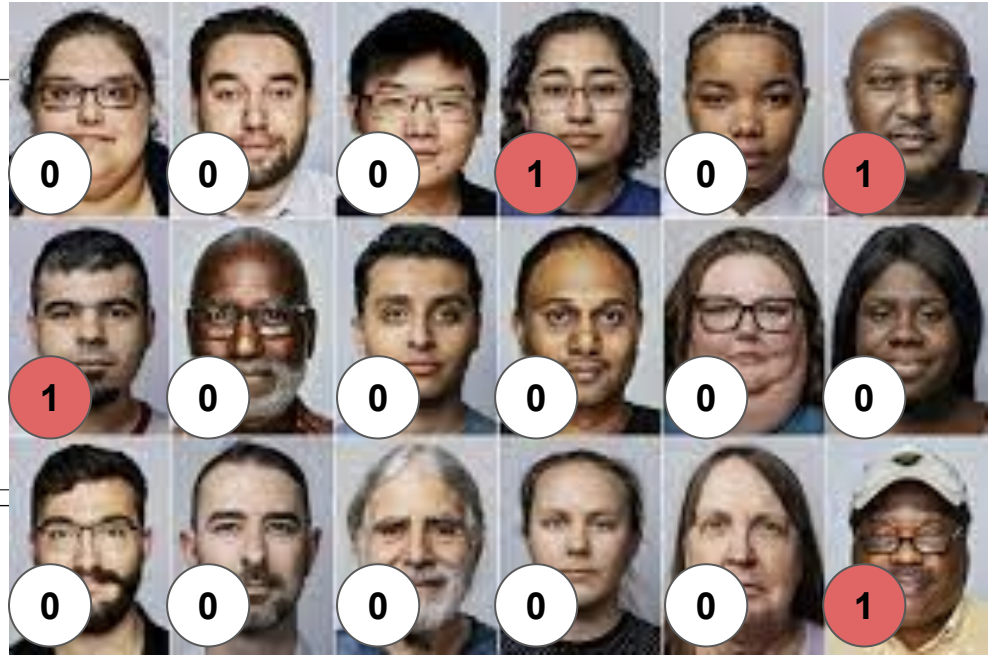
The shape of the **likelihood** over the parameter prob



Example (Answer)

The unknown variable (parameter) prob seems to be 0.22.





Likelihood and the **probability density/mass** function (PDF)

Parameter θ (greek theta, taking about Poisson)

Data

It is called “joint” because it is about the full data.

The **likelihood function** $L(\boldsymbol{\theta}; \mathbf{y})$ is algebraically the same as the joint probability density function $f(\mathbf{y}; \boldsymbol{\theta})$ but the change in notation reflects a shift of emphasis from the random variables $\mathbf{y}$, with $\boldsymbol{\theta}$ fixed, to the parameters $\boldsymbol{\theta}$, with $\mathbf{y}$ fixed. Since L is defined in terms of the random vector $\mathbf{y}$, it is itself

Copy from [DobsonB18]

Bayesian updating

(prior, likelihood, and posterior)

Priors

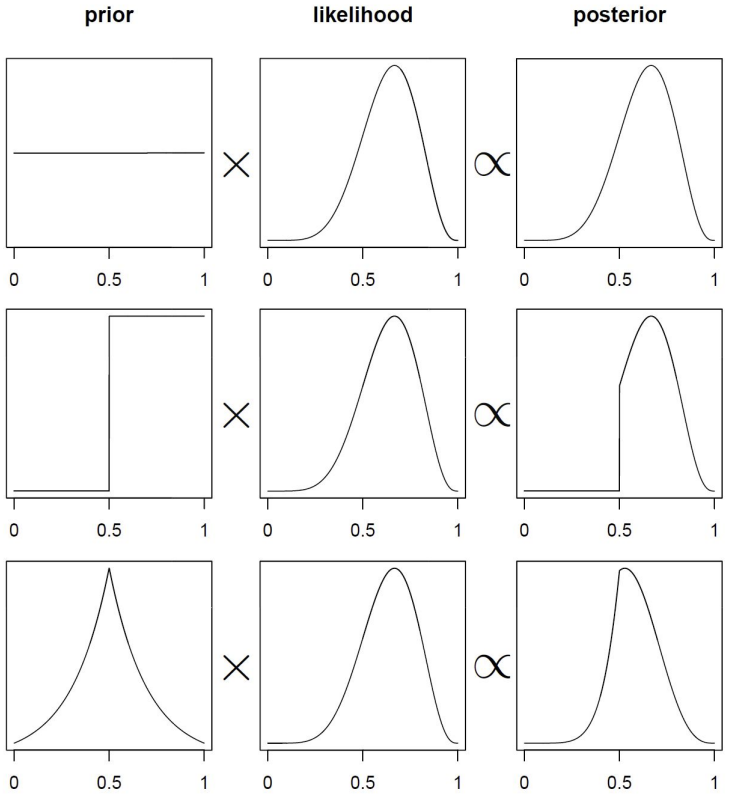
Bayesian models include **distributions of prior plausibility** for parameters. This is useful, as there might be knowledge on reasonable parameters values before seeing the data D.

$$D \sim \text{Binomial}(1, \text{prob})$$

$$\text{prob} \sim \text{Uniform}(0, 1)$$

We add a prior for prob to our model.

How **prior**, **likelihood** and **posterior** relate to each other



When using Bayesian models, we are interested in the **posterior**.

Copied directly from [McElreath20]

Demo

The intuitive way how a programmer would infer prob

(grid approximation + prior)

Backup grid approximation + prior

```r
# ‘D’ is our data (0,0,0,1,0,1,1,0,0,0,0,0,0,0,0,0,0,1).
# Different probs that might be possible (the grid).
probs <- seq(0, 1, length.out = 51)

# Producing the likelihood, i.e., that prob has may have produced the data.
likelihood <- sapply(probs, function(prob) {
 return(prod(dbinom(D, size = 1, prob)))
})

prior <- rep(1, length(probs)) # Every prob has the same prior probability.
posterior <- prior * likelihood # Multiplication.

# Plot both.
plot(probs, posterior, type = "l")
```

We compose the posterior out of prior and likelihood.

Motors

(running the model)

How to run a model

The following motors are the most relevant to compute the posterior (condition the prior on the data):

- **Grid approximation** (we have done this, good for illustration, does not scale),
- **Quadratic approximation** (limited, assumes posterior follows a normal distribution, see [McElreath20] for details),
- **Markov chain Monte Carlo (MCMC)** (more complicated, but it scales, we cover that later).

For now, we do not dive into details; however, we want something better than manually writing a grid approximation.

https://mc-stan.org/

INSTALLATION DOCUMENTATION COMMUNITY ABOUT US YOUR SUPPORT SEARCH

Stan

About Stan

Stan is a state-of-the-art platform for statistical modeling and high-performance statistical computation. Thousands of users rely on Stan for statistical modeling, data analysis, and prediction in the social, biological, and physical sciences, engineering, and business.

We want to end up using Stan (MCMC), but we start modeling using ‘training wheels’.

Math, **ULAM** and its relation to **STAN**

We use a domain-specific programming language (DSL) that comes close to math notation.

$$D \sim \text{Binomial}(1, \text{prob})$$

$$\text{prob} \sim \text{Uniform}(0, 1)$$

Math

correspondsTo

```
ulam(list(
 D ~ dbinom(1, prob),
 prob ~ dunif(0, 1)
), data = list(D = D))
```

ULAM

compilesTo

```
data{
        int D[18];
}
parameters{
        real<lower=0,upper=1> prob;
}
model{
        prob ~ uniform( 0 , 1 );
        D ~ binomial( 1 , prob );
}
```

Stan

Math, **ULAM** and its relation to **STAN**

We use a domain-specific programming language (DSL) that comes close to math notation.

$$D \sim \text{Binomial}(1, \text{prob})$$

$$\text{prob} \sim \text{Uniform}(0, 1)$$

Math

The **math** notation or our model

correspondsTo

Wrapper **ULAM** that hides some **technical complexities**.

```
ulam(list(
 D ~ dbinom(1, prob),
 prob ~ dunif(0, 1)
), data = list(D = D))
```

ULAM

compilesTo

Executable **Stan** code with technical complexities.

```
data{
        int D[18];
}
parameters{
        real<lower=0,upper=1> prob;
}
model{
        prob ~ uniform( 0 , 1 );
        D ~ binomial( 1 , prob );
}
```

Stan

Demo

Using a probabilistic programming language

(ULAM + Stan)

Backup ulam + stan

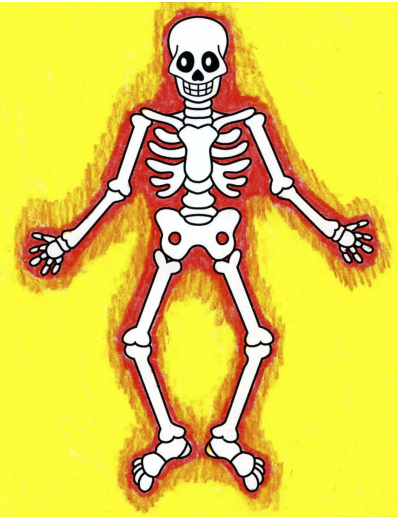
```
library(rethinking) # This is importing the ULAM wrapper (install
from: https://github.com/rmcelreath/rethinking).

# The definition of the model
model <- ulam(list(
 D ~ dbinom(1, prob),
 prob ~ dunif(0, 1)
), data = list(D = D))

# To get the Stan code it compiles to.
stancode(model)
```

Interpreting the results of our model

(interpreting the posterior)

Recap: Interpreting results of a statistical test in the first lecture

Confusing statistical software + confusing output = confused users



Experiment: Supporting our intuition

Welch Two Sample t-test

data: time by group

t = -0.94034, df = 10.424, p-value = 0.3683

alternative hypothesis: true difference in means is not equa

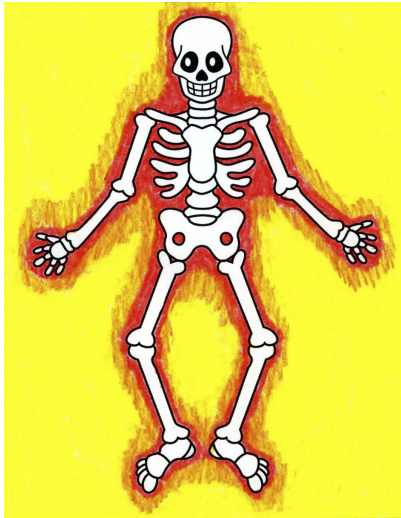
95 percent confidence interval:

-8.666981 3.502671

sample estimates:

mean in group MegaL mean in group None

39.61606 42.19822



More…

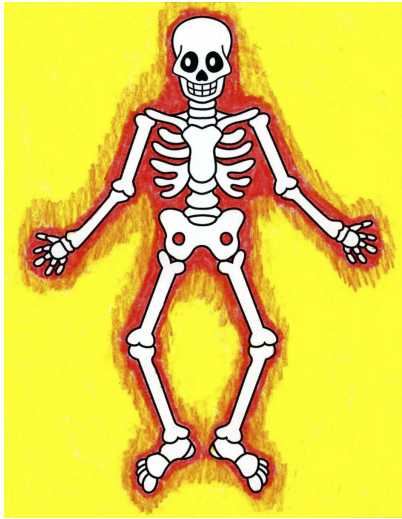
Stata Results

```
. reg price mpg foreign

      Source |       SS       df       MS              Number of obs =      74
-------------+------------------------------           F(  2,    71) =   14.07
       Model |  180261702     2  90130850.8           Prob > F      =  0.0000
    Residual |  454803695    71  6405685.84           R-squared     =  0.2838
-------------+------------------------------           Adj R-squared =  0.2637
       Total |  635065396    73  8699525.97           Root MSE      =  2530.9
```

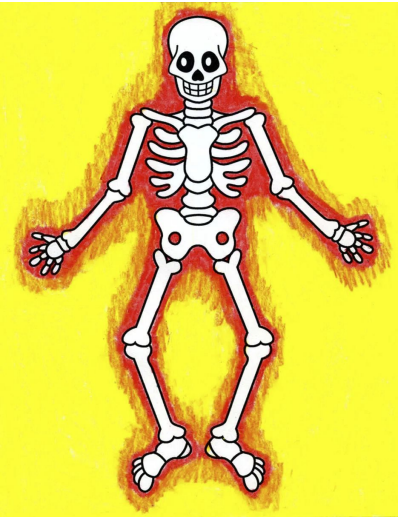
the coefficients (betas)

price	Coef.	Std. Err.	t	P>|t|	[95% Conf.	Interval]
mpg	-294.1955	55.69172	-5.28	0.000	-405.2417	-183.1494
foreign	1767.292	700.158	2.52	0.014	371.2169	3163.368
_cons	11905.42	1158.634	10.28	0.000	9595.164	14215.67

the constant (alpha)



MORE…



Model Summary[b]

Model	R	R Square	Adjusted R Square	Std. Error of the Estimate
1	.886[a]	.785	.785	4.525

a. Predictors: (Constant), Age 11 standard marks

b. Dependent Variable: Age 14 standard marks

ANOVA[b]

Model		Sum of Squares	df	Mean Square	F	Sig.
1	Regression	1059510.757	1	1059510.757	51750.500	.000[a]
	Residual	289412.550	14136	20.473		
	Total	1348923.307	14137			

a. Predictors: (Constant), Age 11 standard marks

b. Dependent Variable: Age 14 standard marks

Coefficients[a]

Model		Unstandardized Coefficients		Standardized Coefficients	t	Sig.
		B	Std. Error	Beta		
1	(Constant)	.261	.038		6.848	.000
	Age 11 standard marks	.873	.004	.886	227.487	.000

a. Dependent Variable: Age 14 standard marks

MORE!

The fact that every picture on standard statistic practice you find using google image search **needs red clarification** is suspicious.

T-Test

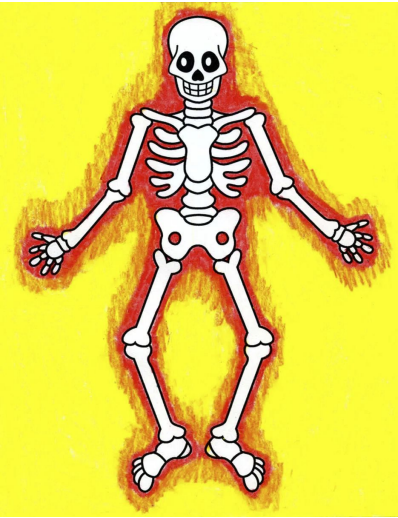
Statistik bei einer Stichprobe

	N	Mittelwert	Standardabweichung	Standardfehler des Mittelwertes
Nettoeinkommen pro Jahr	1365	35714,65	25570,576	692,109

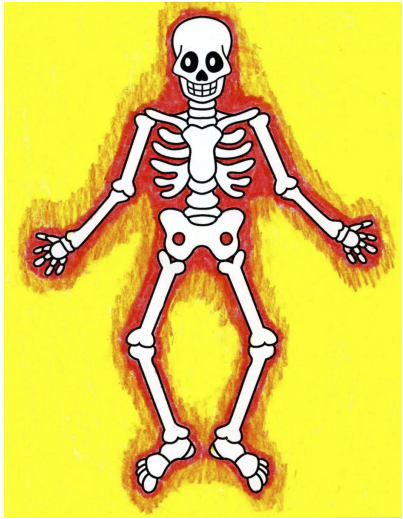
Test bei einer Sichprobe

	Testwert = 50000					
					95% Konfidenzintervall der Differenz	
	T	df	Sig. (2-seitig)	Mittlere Differenz	Untere	Obere
Nettoeinkommen pro Jahr	-20,640	1364	,000	-14285,348	-15643,06	-12927,64

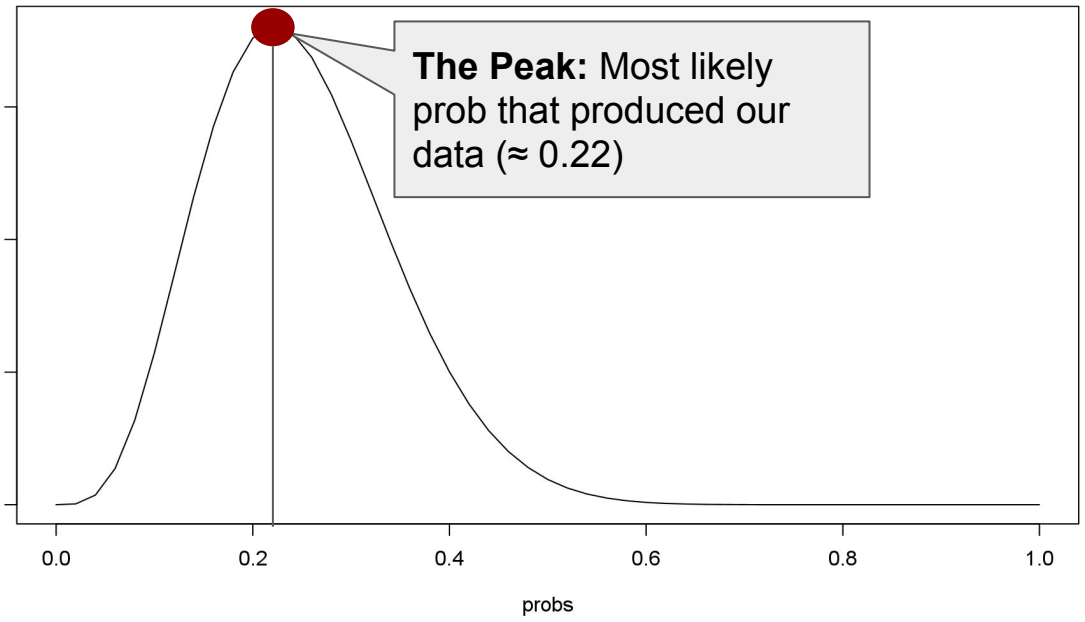
Interpreting the result or our Bayesian model



What we (currently) miss is a standardized method to interpret the results

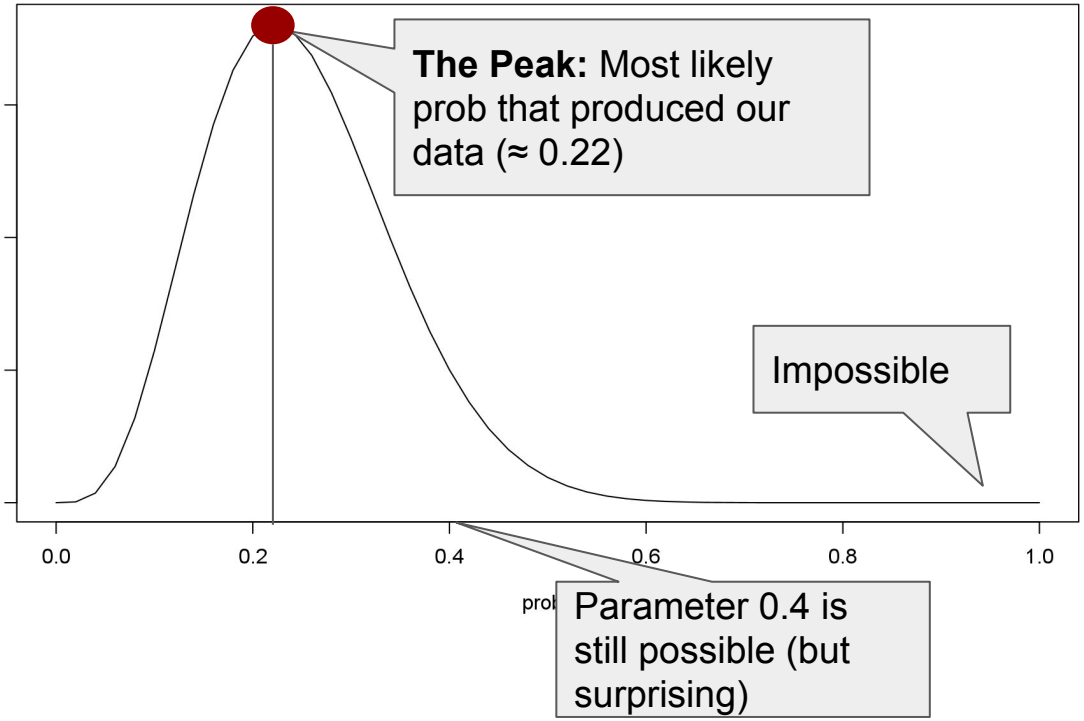


Until now, we did **not use much** of the posterior for the parameter …



… but there is **more** than that.

We are interested in the full posterior.



Summarizing the posterior

Typical question that we might have in the context of our research:

- How much posterior probability **lies below** some parameter value?
- How much posterior probability **lies between** two parameter values?
- Which parameter value marks the **lower 5%** of the posterior probability?
- Which range of parameter values **contains 90%** of the posterior probability?
- Which parameter value has the **highest** posterior probability?

(and there is not always space to show the full posterior).

Questions copied directly from [McElreath20]

Summarizing the posterior (cont)

In essence, these are three types or questions:

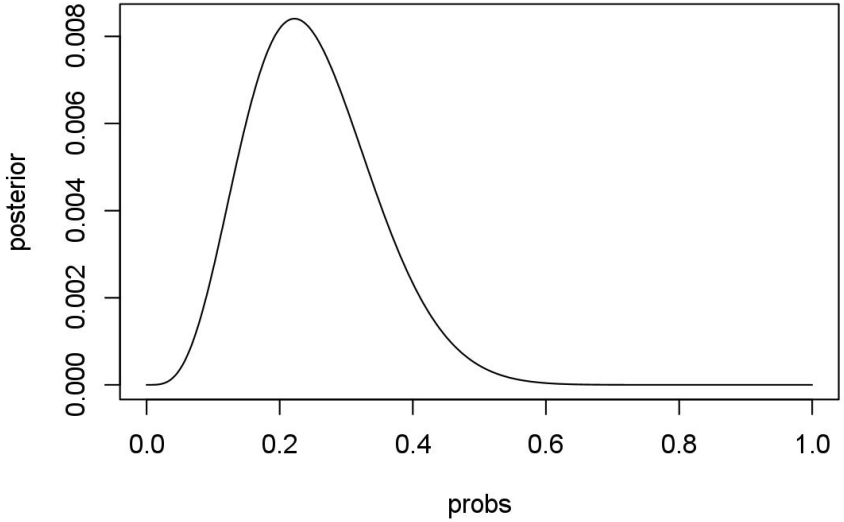
- intervals of defined boundaries,
- defined probability mass and
- point estimates.

You might remember: the **cumulative distribution function (CDF) …**

or the **quantile function (inverse CDF)** from the last lecture.

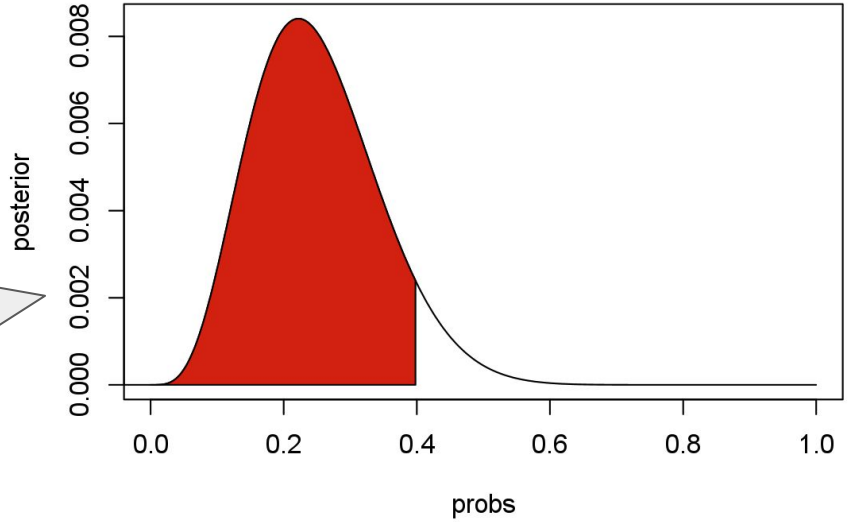
Questions copied directly from [McElreath20]

Example: How much posterior probability lies below the parameter value 0.4 (intervals of defined boundaries)?



Example: How much posterior probability lies below the parameter value 0.4 (intervals of defined boundaries)?

Remember the **cumulative distribution function (CDF)**.



Problems with interpreting the posterior

- You may remember problems (last lecture) computing the CDF for the normal distribution, since there was **no closed form solution**.
- We prefer not to be **limited** to a normal distribution.

Hence, we need **another mechanism** to work with our posterior (it is the same as in the previous lecture).

Sampling the Imaginary*

We are pulling out samples from the posterior and **process the samples**.

```
# Code for getting samples from our grid approximation.

samples <- sample(probs, prob = posterior, size = 1e4, replace = T)
```

```
samples
  [1] 0.272 0.272 0.424 0.136 0.250 0.328 0.252 0.176
 [14] 0.226 0.086 0.264 0.146 0.342 0.246 0.174 0.350
 [27] 0.458 0.248 0.134 0.198 0.186 0.134 0.272 0.190
 [40] 0.324 0.344 0.138 0.354 0.296 0.132 0.272 0.264
 [53] 0.134 0.100 0.298 0.256 0.212 0.154 0.302 0.166
 [66] 0.262 0.084 0.056 0.230 0.302 0.336 0.120 0.248
```

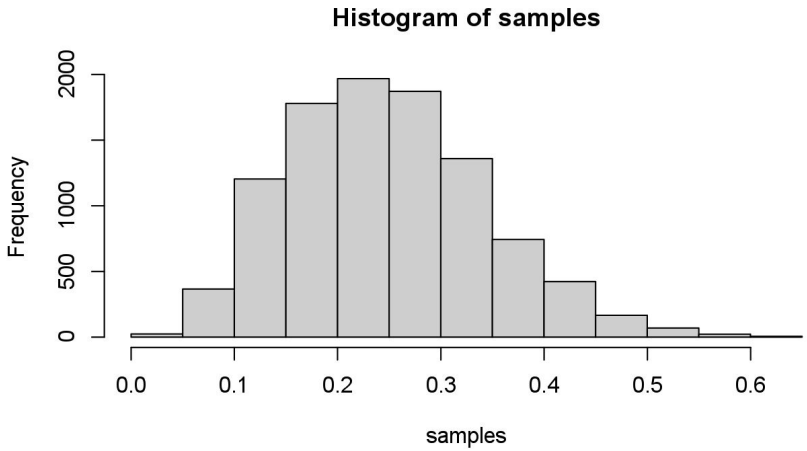
Our results are just a vector/array that represents the posterior.

(*)The title is borrowed from [McElreath20]

Sampling the Imaginary*

We are pulling out samples from the posterior and **process the samples**.

```
# Code plotting the samples that describe the posterior.

hist(samples)
```



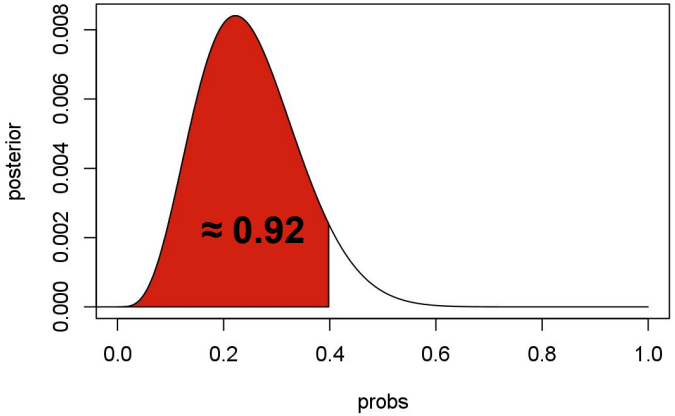
(*)The title is borrowed from [McElreath20]

Sampling the Imaginary*

We are pulling out samples from the posterior and **process the samples**.

```
# Code for answering the question on
'intervals of defined boundaries' by simple
counting:

sum(samples < 0.4)/ length(samples)
```

≈ 0.92



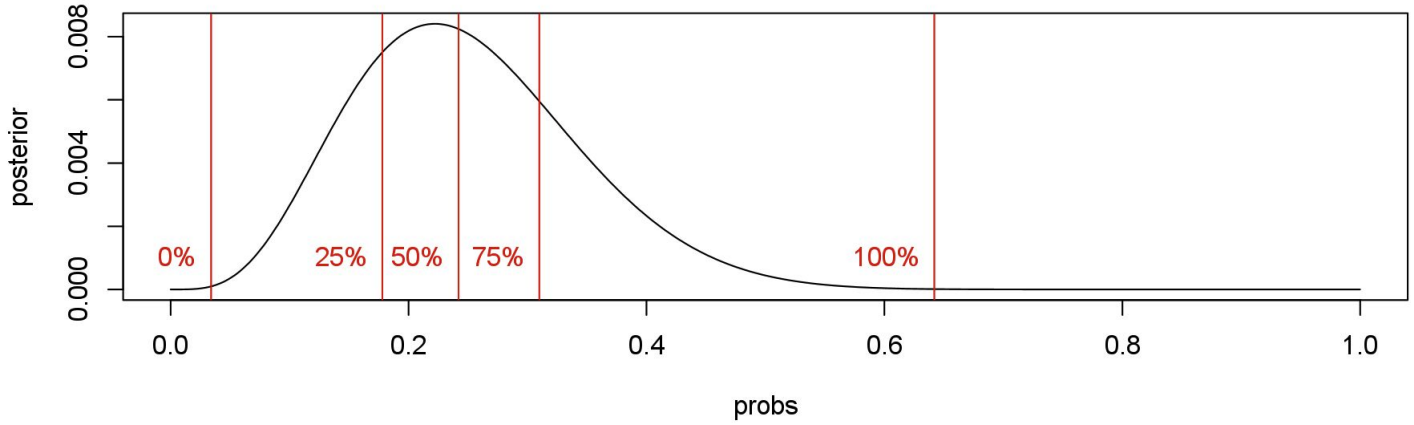
(*)The title is borrowed from [McElreath20]

Sampling the Imaginary*

We are pulling out samples from the posterior and **process the samples**.

```
# Code for answering the question on 'defined probability mass'.

quantile(samples)
```

0%	25%	50%	75%	100%
0.034	0.178	0.242	0.310	0.642



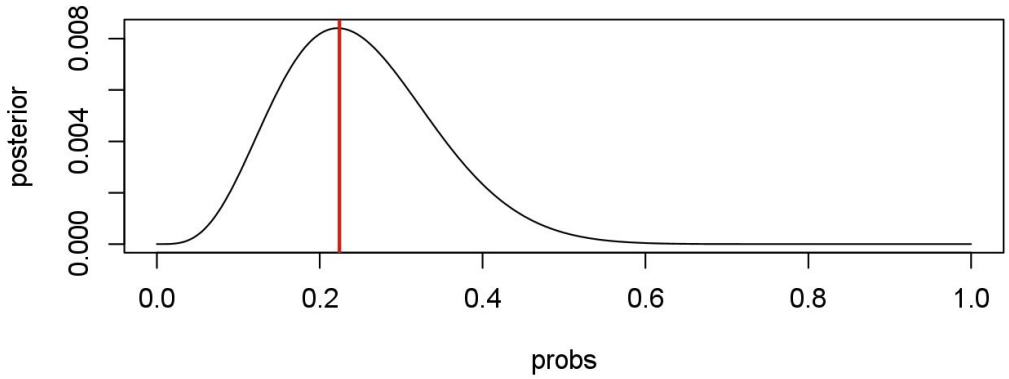
(*)The title is borrowed from [McElreath20]

Sampling the Imaginary*

We are pulling out samples from the posterior and **process the samples**.

```
# Code for answering questions on 'point estimates'.

chainmode(samples, adj=0.01)
```

≈ 0.22



The mode is the value that appears most often (hence, it is called chainmode).

(*)The title is borrowed from [McElreath20]

Why do we do this: **MCMC provides us with samples**

```
# Code defining and running the model.
model <- ulam(list(
 D ~ dbinom(1, prob),
 prob ~ dunif(0, 1)
), data = list(D = D))

# Code for getting samples out of ULAM (backed by Stan using
MCMC).
samples <- extract.samples(model)
```

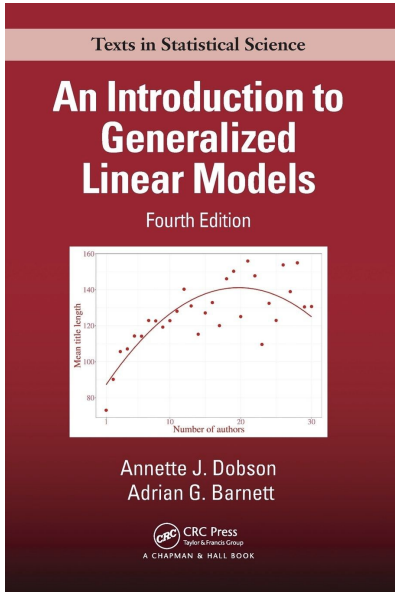
Summary

Summary

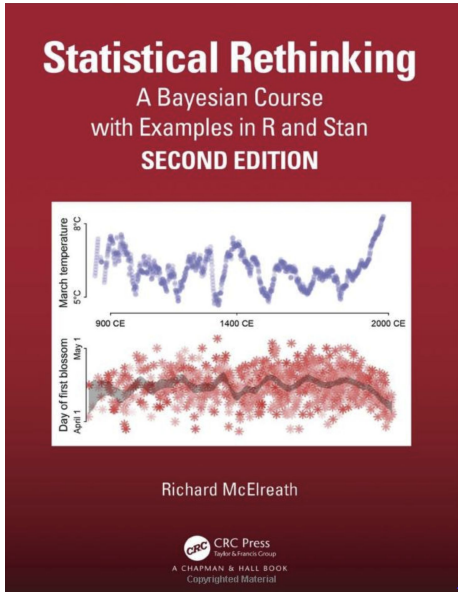
- We have introduced the **first statistic model** asking for an unobserved variable (a parameter).
- We used a **uniform language** for writing such model.
- We know how to **run** such models inferring the posterior of a parameter (doing grid approximation or MCMC).
- We have seen how to **interpret the posterior** of a parameter.

Next lecture we will develop some advanced statistic model.

References:



[DobsonB18]



[McElreath20]