Introduction to Computational BioStatistics with R: Distributions

Erik Spence

SciNet HPC Consortium

14 October 2021

SciNet compute • calcul CANADA

Today's slides

Today's slides can be found here. Go to the "Introduction to Computational BioStatistics with R" page, under Lectures, "Distributions".

`https://scinet.courses/1182`

SciNet compute • calcul CANADA

Today's class

Today we are going to begin our adventure in the world of statistics.

- Distributions.
- R functions for calculating distributions.
- The *apply family of functions.

As with all classes, please stop me if you have a question.

SciNet compute • calcul CANADA

Statistics

To begin at the beginning: what is statistics?

- Statistics is a collection of techniques for empirically describing populations, collections of populations, and the relationships between them.
- Usually we do not have the complete population at our disposal, we only have a sample of the population.
- We use this sample to draw conclusions about the true population from which the sample was drawn, assuming that the sample is representative of the whole population.
- We also use the sample to perform tests to determine the relationship between different populations.

Right. No problem.

SciNet compute • calcul CANADA

Data come from distributions

We usually use a probability distribution to model our data. What is a probability distribution?

- A probability distribution indicates the probability of a given event (or measurement, or observation) happening.
- There are two types of probability distributions:
  - discrete: data come in individual steps, there are no data points between those steps (flips of a coin, rolls of dice).
  - continuous: data are real numbers, with decimal places.
- All probability distributions have the following properties:
  - The sum of all probabilities equals one.
    - Discrete: $\sum_i \boldsymbol{P}(\boldsymbol{x_i}) = \mathbf{1}$
    - Continuous: $\int \boldsymbol{\rho}(\boldsymbol{x})\boldsymbol{dx} = \mathbf{1}$
  - One minus the probability of something is the probability of not something. $\boldsymbol{P'}(\boldsymbol{x}) = \mathbf{1} - \boldsymbol{P}(\boldsymbol{x})$

SciNet compute • calcul CANADA

Discrete distributions

Discrete distributions apply when the data are not continuous, the data come in discrete steps.

Binomial distribution:

$$P(x = k) = \binom{n}{k} p^k (1-p)^{n-k}$$

where p is the probability of success, k is the number of successes and n is the number of attempts.

An example of this would be picking n marbles out of a bag of red and black marbles, and picking k red marbles.

Coin toss:

$$P(x) = \begin{cases} 0.5, & x = \text{heads}, \\ 0.5, & x = \text{tails} \end{cases}$$

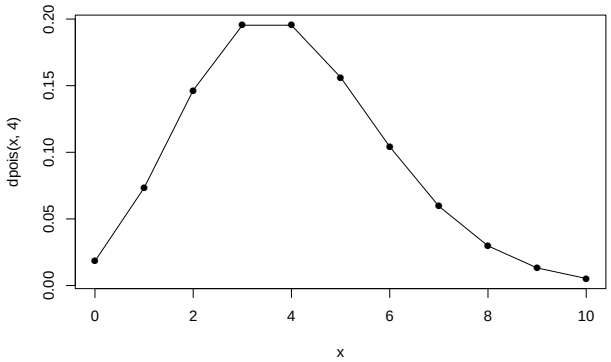
Roll of a die:

$$P(x) = \begin{cases} 1/6, & x = 1, \\ 1/6, & x = 2, \\ 1/6, & x = 3, \\ 1/6, & x = 4, \\ 1/6, & x = 5, \\ 1/6, & x = 6 \end{cases}$$

Poisson distribution

The Poisson distribution is used for discrete events that happen infrequently. The probability of the event happening must increase with time.

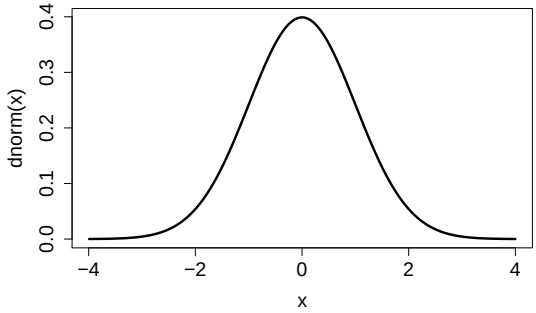
Please do not use continuous distributions for discrete variables!



Poisson distribution ($\lambda = 4$):

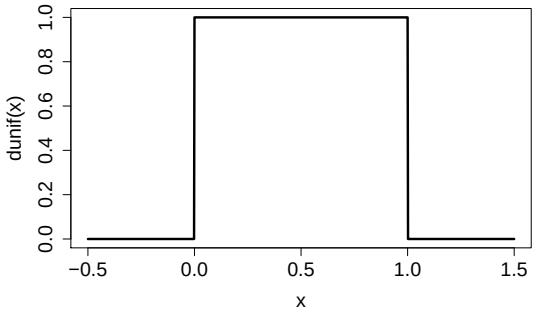
$$P(x) = \frac{\lambda^x e^{-\lambda}}{x!}$$

Continuous distributions



$$\rho(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

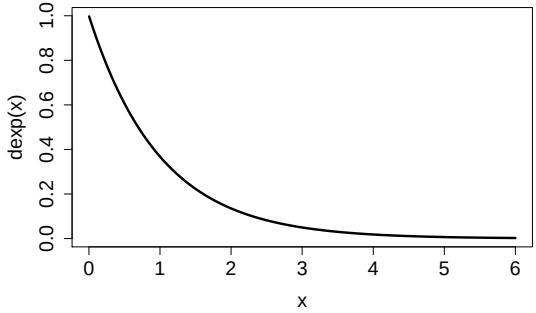
Gaussian



$$\rho(x) = \begin{cases} 1, & 0 \leq x \leq 1 \\ 0, & \text{otherwise} \end{cases}$$

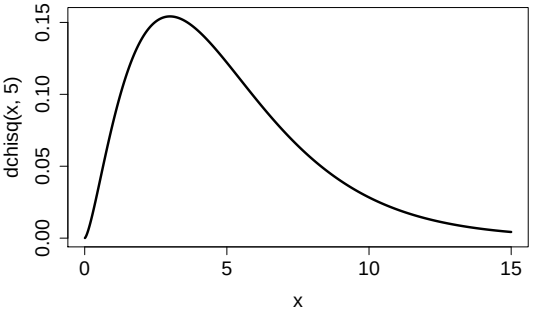
Uniform

Continuous distributions, continued



$$\rho(x) = \lambda e^{-\lambda x}$$

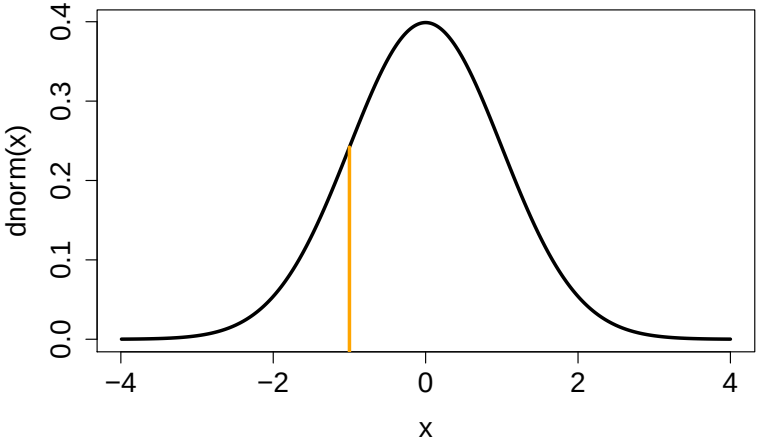
Exponential



$$\rho(x) = \frac{1}{2^{\frac{n}{2}}\Gamma\left(\frac{n}{2}\right)} x^{\left(\frac{n}{2}-1\right)} e^{-\frac{x}{2}}$$

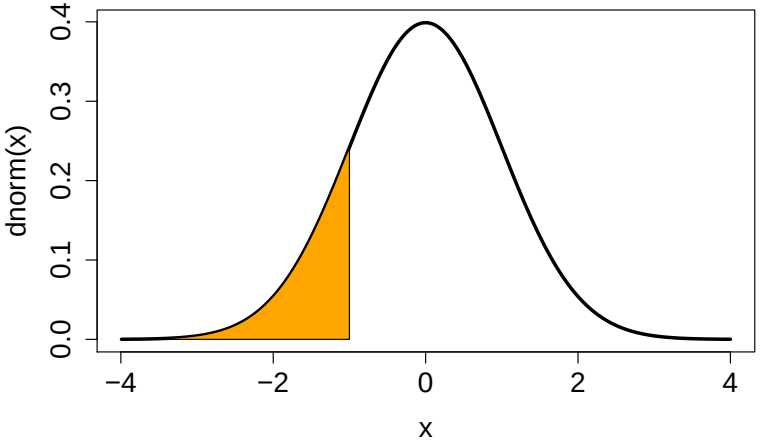
Chi-squared

Some distribution terminology



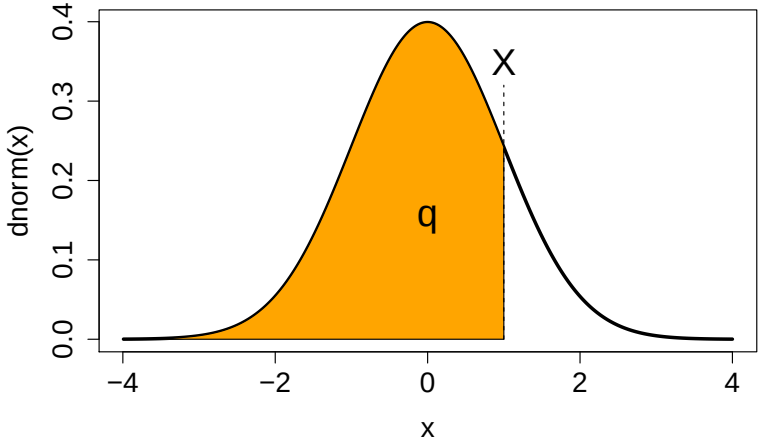
Probability density function (PDF), $\rho(x)$: the function which determines probability of getting a particular value, X, from a continuous distribution, $P(x = X) = \rho(x)dx$.

Some distribution terminology, continued



Cummulative distribution function (CDF): the probability of getting a particular value of x below a certain value, $P(x < X) = \int_{-\infty}^{X} \rho(x)dx$.

Some distribution terminology, continued more



Quantile function (QFn): given a probability q, the particular value of X such that $P(x < X) \leq q$.

Some more distribution terminology

Some terms you're all likely aware of.

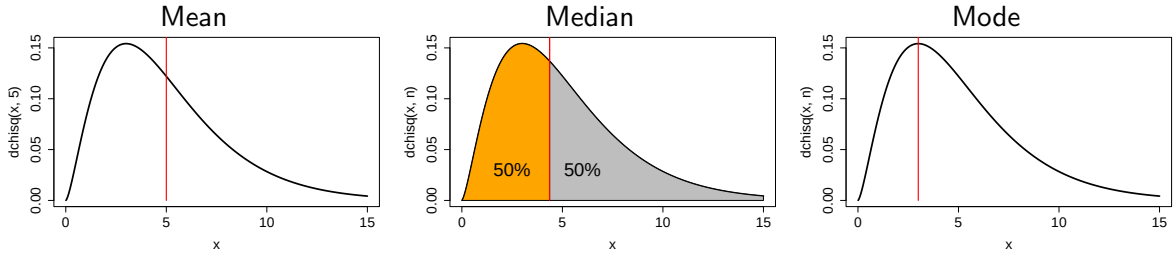
- the Expectation value: $\langle f \rangle = E(f(x)) = \int_{-\infty}^{\infty} f(x)\rho(x)dx$.
- the mean: $\langle x \rangle = \int_{-\infty}^{\infty} x\rho(x)dx$.
- variance: $\sigma^2(x) = \left\langle (x - \langle x \rangle)^2 \right\rangle$
- Standard deviation: square root of the variance.

R comes with many built-in functions to calculate the usual quantities.

- mean(), sd(), var()
- min(), max()
- range()
- summary()

As usual, type "help(mean)" to learn how these functions can be used. Do NOT use these function names as variables!

Characteristic statistics



A warning: many distributions are not centred. If the distribution of your data is not centred the concept of a 'mean' may not be meaningful.

The chi-squared distribution, shown here, is not centered or symmetric around its peak.

The summary function

One of the more useful functions is'summary'. It prints out basic statistics about the quantity in question.

```
>
> summary(trees)
     Girth           Height       Volume
 Min.   : 8.30   Min.   :63   Min.   :10.20
 1st Qu.:11.05   1st Qu.:72   1st Qu.:19.40
 Median :12.90   Median :76   Median :24.20
 Mean   :13.25   Mean   :76   Mean   :30.17
 3rd Qu.:15.25   3rd Qu.:80   3rd Qu.:37.30
 Max.   :20.60   Max.   :87   Max.   :77.00
>
```

Built-in datasets, an aside

R contains built-in datasets that can be used for practicing.

```
> data()
Data sets in package ‘datasets’:

 AirPassengers          Monthly Airline Passenger Numbers 1949-1960
 BJsales                Sales Data with Leading Indicator
 BJsales.lead (BJsales) Sales Data with Leading Indicator
 BOD                    Biochemical Oxygen Demand
 ⋮
>
> str(faithful)
data.frame’:   272 obs.  of 2 variables:
$ eruptions: num 3.6 1.8 3.33 2.28 4.53 ...
$ waiting  : num 79 54 74 62 85 55 88 85 51 85 ...
>
```

Type 'q' to get out of the 'data' menu. DO NOT use 'data' as a variable.

R distribution functions

R has a tonne of distributions built into it. The syntax for using the distributions is fairly consistent. To access a particular distribution, you use the following suffixes:

- Uniform: unif
- Normal: norm
- Binomial: binom
- Poisson: pois
- and many many others

To access particular functions associated with those distributions, you use the prefixes:

- Probability Distribution Function (PDF): d
- Cummulative Distribution Function (CDF): p
- Quantile Function (QFn): q
- Random sampling from the distribution: r

SciNet compute • calcul CANADA

R distribution functions, continued

Distribution	suffix	PDF (d)	CDF (p)	QFn (q)	Sample (r)
Normal	norm	dnorm	pnorm	qnorm	rnorm
Uniform	unif	dunif	punif	qunif	runif
Exponential	exp	dexp	pexp	qexp	rexp
Poisson	pois	dpois	ppois	qpois	rpois
Binomial	binom	dbinom	pbinom	qbinom	rbinom

Note that most distributions take optional arguments which control their behaviour (use the 'help' function to get details on how to use these functions).

R distribution functions, examples

```
>
> # Normal distribution probabilities with
> # default values (mean = 0, sd = 1)
> dnorm(c(-2, 0, 2))
[1] 0.05399097 0.39894228 0.05399097
>
> # Normal distribution probabilities with
> # mean = 1, sd = 2
> dnorm(c(-2, 0, 2), mean = 1, sd = 2)
[1] 0.0647588 0.1760327 0.1760327
>
> # Value of normal distribution which
> # has a probability of 0.025
> qnorm(0.025)
[1] -1.959964
>
```

```
>
> # 4 random samples from the normal distribution
> # with default values (mean = 0, sd = 1)
> rnorm(4)
[1] -0.01890732 -1.51366406 1.00561637 -0.27690594
>
> # 4 random samples from the normal distribution
> # with mean = 1, sd = 2
> rnorm(4, mean = 1, sd = 2)
[1] 1.70841356 2.96691253 0.07857346 -0.87288538
>
> # 4 random samples from the Poisson distribution
> # with lambda = 20
> rpois(4, lambda = 20)
[1] 26 19 25 15
>
```

SciNet compute • calcul CANADA

Applied examples

Suppose that a given collection of insects have weights that are normally distributed with a

- mean of 17.46 grams and
- variance of 75.67 grams^2.

What is the probability that a randomly chosen insect within the collection weighs more than 19 grams?

We follow these steps:

- We use the cummulative distribution function (CDF) to get the probability of being less than 19 grams.
- We then subtract this from 1 to get the probability of being greater than 19 grams.

```
>
> p <- pnorm(19, mean = 17.46, sd = sqrt(75.67))
>
> 1 - p
[1] 0.4297405
>
```

Answer: 43%

Applied examples, continued

Suppose that IQ scores are normally distributed with a

- mean of 100 and
- standard deviation of 15.

What is the 95th percentile of the distribution of IQ scores?

What is the value of x such that x is greater than 95% of the distribution?

```
>
> qnorm(0.95, mean = 100, sd = 15)
[1] 124.6728
>
```

Answer: 124.7

Applied examples, continued more

Suppose some widgets produced at the Acme Factory have a probability of 0.005 of being defective. The widgets are shipped in boxes of 25.

- What is the probability that a box contains exactly 1 defective widget?
- What is the probability that a box has *at most* 1 defective widget?

Use a binomial distribution, since each widget is either defective or not. The binomial distribution takes two optional arguments:

- size (the number of samples),
- prob (the probabilty of something occuring).

```
>
> dbinom(1, size = 25, prob = 0.005)
[1] 0.1108317
>
> pbinom(1, size = 25, prob = 0.005)
[1] 0.9930519
>
```

Answer 1: 11% Answer 2: 99%

The *apply family of functions

The *apply family of functions make it very easy and fast to repeatedly apply a function to a set of individual elements.

Many parallel routines are parallel versions of these higher-level functions.

- lapply: apply a function to each element of a list or vector.
- sapply: like lapply, but return a vector instead of a list.
- apply: apply a function to rows, columns or elements of an array.
- tapply: apply a function to subsets of a list or vector.
- mapply: apply a function to the "transpose" of a list.

Using these functions, rather than regular for loops, can significantly speed up calculations.

SciNet compute • calcul CANADA

lapply

The function lapply will repeatedly apply a function to each element of a list or vector.

Note that you only specify the name of the function to be called as the second argument. You don't give the function any arguments, unless extra arguments are needed by the function, in which case they are supplied to lapply, not the function.

```
> mean.n.rnorm <- function(n) return(mean(rnorm(n)))
>
> ns <- c(1, 10, 100, 1000)
>
> lapply(ns, mean.n.rnorm)
[[1]]
[1] -0.01890732

[[2]]
[1] 0.1327366

[[3]]
[1] -0.1007226

[[4]]
[1] -0.03226481
>
```

lapply, continued

The same could be done in two steps:

- apply rnorm to the list of values
- run mean on the list of vectors

```
>
> ns <- c(1, 10, 100, 1000)
>
> random.nums <- lapply(ns, rnorm)
>
> lapply(random.nums, mean)
[[1]]
[1] 0.341349

[[2]]
[1] 0.3039785

[[3]]
[1] 0.03475841

[[4]]
[1] -0.01775784
>
```

sapply

I usually use sapply, as it returns a vector rather than a list, which I usually find more-convenient to use.

```
>
> ns <- c(1, 10, 100, 1000)
>
> random.nums <- lapply(ns, rnorm)
>
> sapply(random.nums, mean)
[1] 0.34134897 0.30397851 0.03475841 -0.01775784
>
```

SciNet compute • calcul CANADA

apply

The apply function is used on matrices and arrays. It applies a function to the rows (MARGIN = 1), or columns (MARGIN = 2) of an array.

```
>
> A <- matrix(1:9, nrow = 3)
>
> A
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
>
> apply(A, MARGIN = 1, max) # max of each row
[1] 7 8 9
>
> apply(A, MARGIN = 2, max) # max of each column
[1] 3 6 9
>
```

apply, continued

The apply function can also be applied to each element, using MARGIN = 1:2.

If you have more than 2 dimensions then those dimensions can also be specified as an argument to MARGIN.

```
>
> A <- matrix(1:9, nrow = 3)
>
> A
     [,1] [,2] [,3]
[1,]    1    4    7
[2,]    2    5    8
[3,]    3    6    9
>
>
> apply(A, MARGIN = 1:2, function(x) return(x**2))
     [,1] [,2] [,3]
[1,]    1   16   49
[2,]    4   25   64
[3,]    9   36   81
>
```

tapply

The tapply function applies a function to each cell of a ragged array, meaning to each group of values given by a factor level.

Here we're taking the mean of the "Speed" quantities, grouped based on the (factor) value of "Expt".

Similar results can be obtained using the 'split' function.

```
>
> str(morley)
'data.frame':  100 obs.  of 3 variables:
$Expt :  int 1 1 1 1 1 1 1 1 1 1 ...
$Run :  int 1 2 3 4 5 6 7 8 9 10 ...
$Speed:  int 850 740 900 1070 930 850 950 980 ...
>
> tapply(morley$Speed, morley$Expt, mean)
    1     2     3     4     5
909.0 856.0 845.0 820.5 831.5
>
```

Enough to get started

- Today's class went over the very beginning concepts needed to do statistics.
- There are two classes of distributions, continuous and discrete.
- All the distribution commands are built into R already. If they aren't in the base R installation, they exist in a separate package.
- The *apply family functions can be used to operate on vectors, matrices or data frames.

SciNet compute • calcul CANADA