First steps with R

Practical: first steps with R

Jacques van Helden

2017-01-08

Contents

Scope

In this session we will explore basic manipulations of variables.

- Assigning a value to a variable

- Basic operations on numbers

R is a calculator

Convention:

- Dark boxes: commands to type in RStudio **Console** (bottom-left panel).
- White boxes: the result you should obtain.

Example: compute a simple addition.

```
2 + 5
```

```
[1] 7
```

Assign a value to a variable

In ***R*** **<-** means "create a variable and assign its value."

Example:

- create a variable named a,
- assign the value 2 to this variable,
- ***print*** the result.

```
a <- 2
print(a)
```

```
[1] 2
```

Computing with variables

Example:

- create a variable named b having value 5,
- compute $a + b$ and store the result in a variable named c,
- ***print*** the result.

```
b <- 5
c <- a + b
print(c)
```

```
[1] 7
```

Variables need to be updated

Example:

- change the value of a to 3,
- print the value of c
- Is this the correct result for $c = a + b$? Why?

```
a <- 3 ## Change the value of a
print(c) ## Print the value of c
```

```
[1] 7
```

```
## Check whether c equals a + b
c == a + b
```

```
[1] FALSE
```

Note: `==` is used to test whether two variables have the same content.

Updating variable contents

Example:

- change the value of a to 27,
- recompute and print the value of c

```
a <- 27 ## Change the value of a
c <- a + b
print(c) ## Print the value of c
```

```
[1] 32
```

```
## Check whether c  equals a + b
c == a + b
```

```
[1] TRUE
```

Vectors of values

The simplest data structure in ***R*** is a vector. In the previous example, the variable ***a*** was actually a vector with a single value.

Example: create a variable named ***three.numbers***, and initialize it with a vector with values ***27***, ***12*** and ***3000***.

Tips: - variable names can comprize several parts, separated by dots. - the function ***c()*** combines several values into a vector

```
three.numbers <- c(27,12,3000)
print(three.numbers)
```

```
[1]   27   12 3000
```

Series

The simple way to create a series of numbers. The column operator permits to generate all integer values between two limits.

```
x <- 0:14
print(x)
```

```
 [1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14
```

Computing with vectors

R handles vectors in a very convenient way. An operation on a vector applies to all its elements.

```
x <- 1:10 # Define a series from 1 to 10
print(x)
```

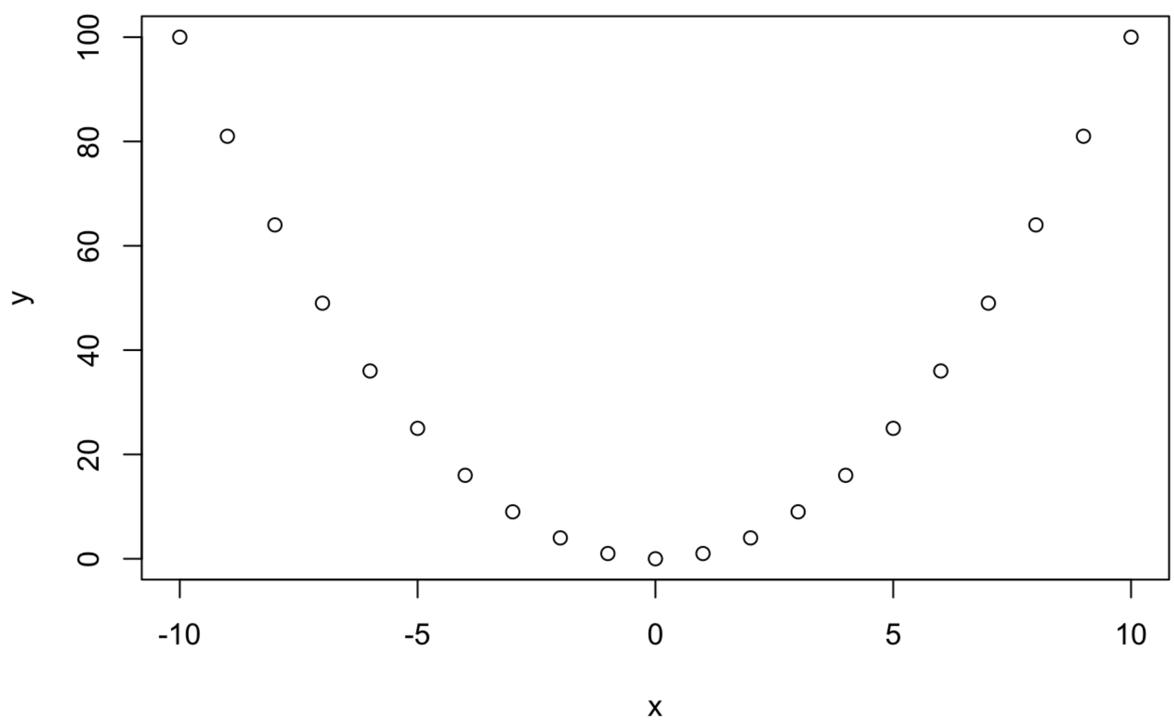
```
 [1]  1  2  3  4  5  6  7  8  9 10
```

```
y <- x^2 # Compute the square of each number
print(y)
```

```
 [1]   1   4   9  16  25  36  49  64  81 100
```

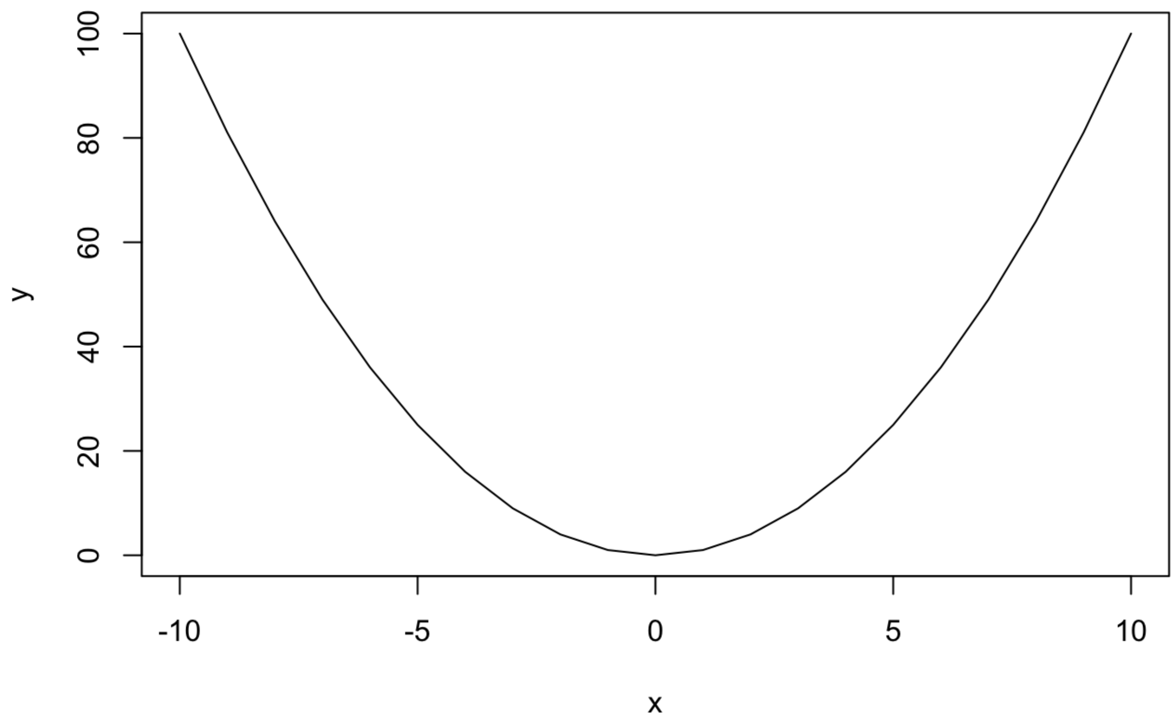
Scatter plot

```
x <- -10:10
y <- x^2
plot(x,y)
```



Line plot

```
x <- -10:10
y <- x^2
plot(x,y, type="l")
```



Variables can also contain strings

```
# The # symbol allows to insert comments in R code

# Define  a vector named "whoami", and
# containing two names
whoami <- c("Denis", "Siméon")
print(whoami) # Comment at the end of a line
```

```
[1] "Denis"  "Siméon"
```

String concatenation

```
# Define  a vector named "names", and
# containing two names
whoami <- c("Denis", "Siméon")

# Paste the values of a vector of string
print(paste(sep=" ", whoami[1], whoami[2]))
```

```
[1] "Denis Siméon"
```

Carl's preferred distribution

The function **`dpois()`** computes the Poisson **density**, i.e. the probability to observe **exactly** x successes in a series of independent trials with equal probability.

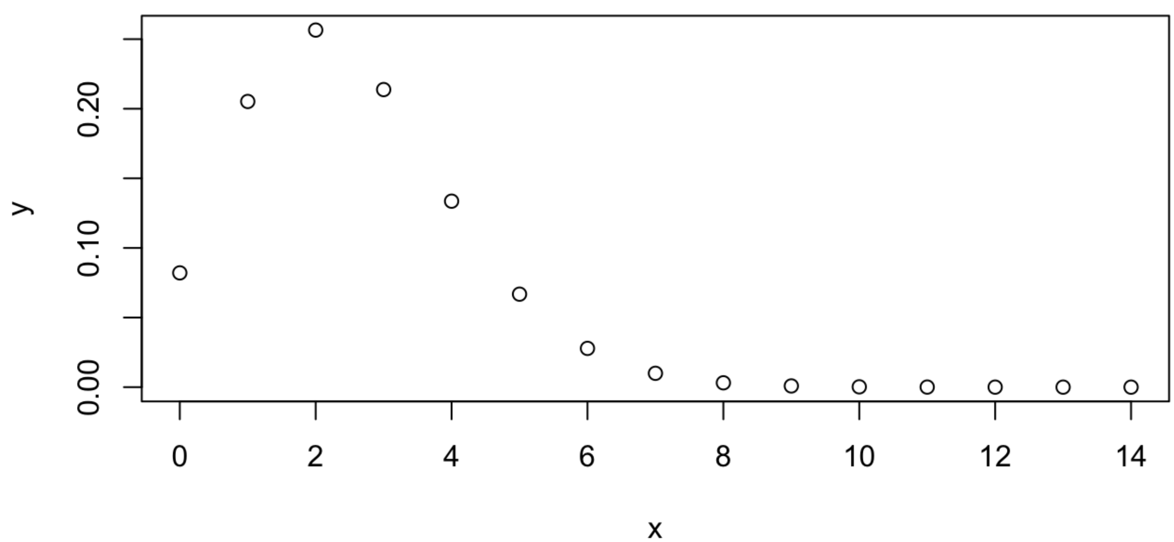
The Poisson distribution is defined by a single parameter: the expected number of successes λ (read "lambda").

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

```
x <- 0:14   # Define the X values from 0 to 14
y <- dpois(x, lambda = 2.5) # Poisson density
print(y) # Check the result
```

Plotting the Poisson distribution

```
x <- 0:14   # Define the X values from 0 to 14
y <- dpois(x, lambda = 2.5) # Poisson density
plot(x,y) # Check the result
```



This first plot is not very nice. Let us get some help to improve it.

Getting help for *R* functions

Need help? Type **`help()`**.

```
help(plot)
```

A question? Type **`?`**

```
?plot
```

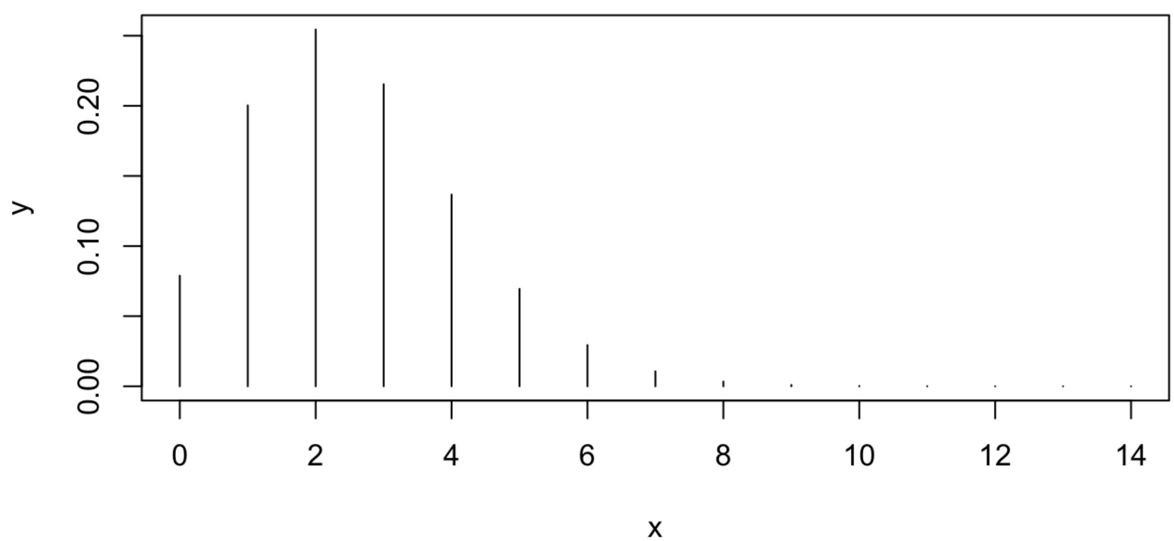
Result: ***R*** displays the help message for the function `dpois()`.

Exercise: improve Poisson density plot

1. Do not (yet) look the next slide.
2. Read the help page for the `dpois()`function.
3. draw a plot that provides a didactic illustration of the Poisson density.

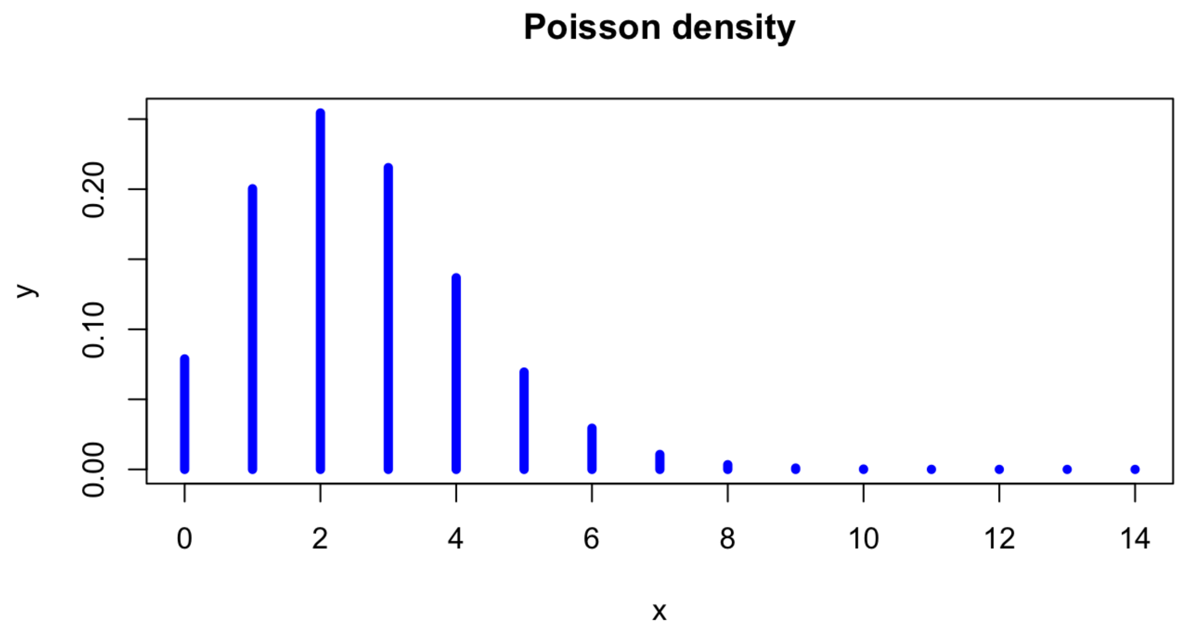
Improve the plot: type = histogram

```
x <- 0:14
lambda <- 2.54
y <- dpois(x, lambda)
plot(x,y, type="h")
```



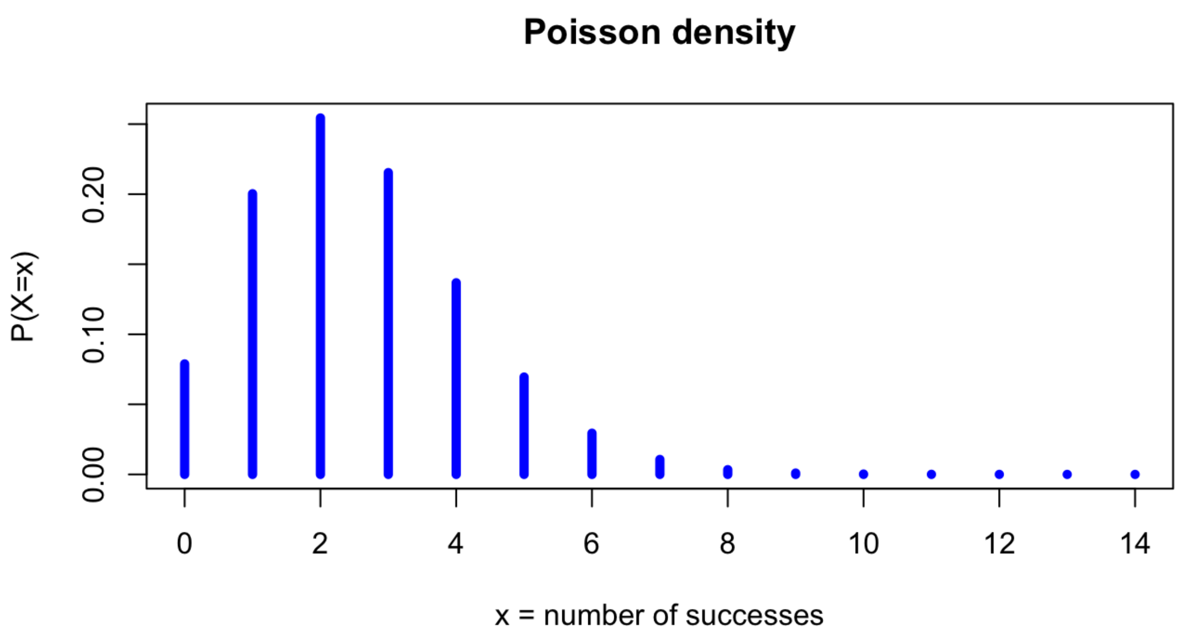
Improve the plot: Add a title

```
plot(x,y, type="h", lwd=5, col="blue",
     main="Poisson density")
```



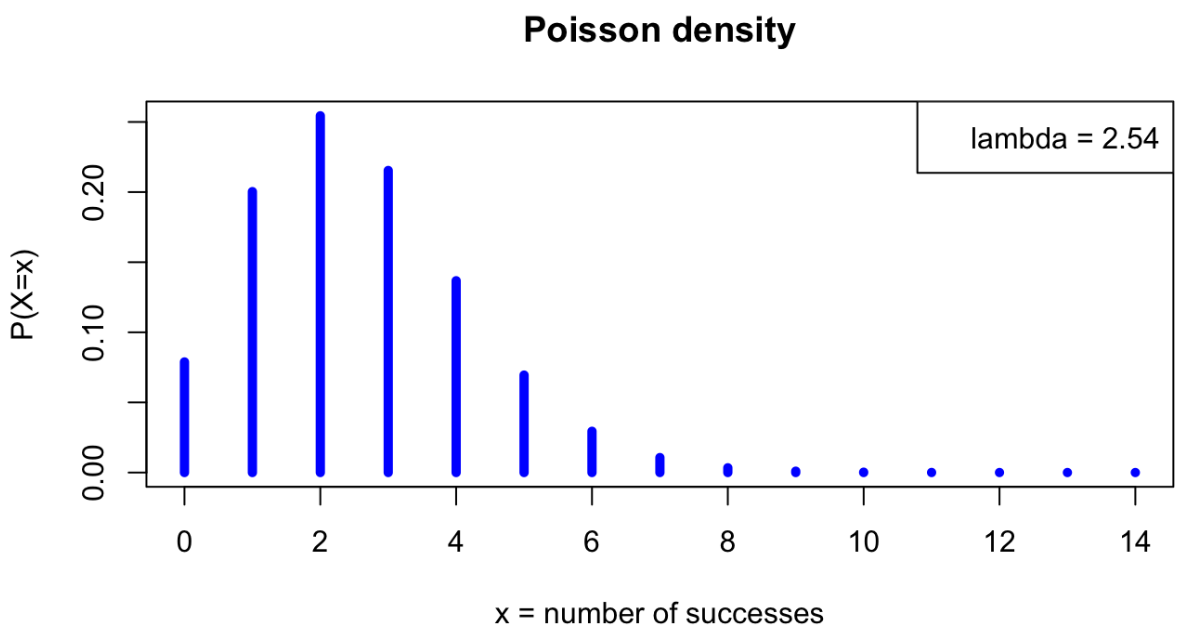
Improve the plot: define axis labels

```
plot(x,y, type="h", lwd=5, col="blue",
     main="Poisson density",
     xlab="x = number of successes",
     ylab="P(X=x)")
```



Improve the plot: add a legend

```
plot(x,y, type="h", lwd=5, col="blue",
     main="Poisson density",
     xlab="x = number of successes",
     ylab="P(X=x)")
legend("topright", paste("lambda =", lambda))
```



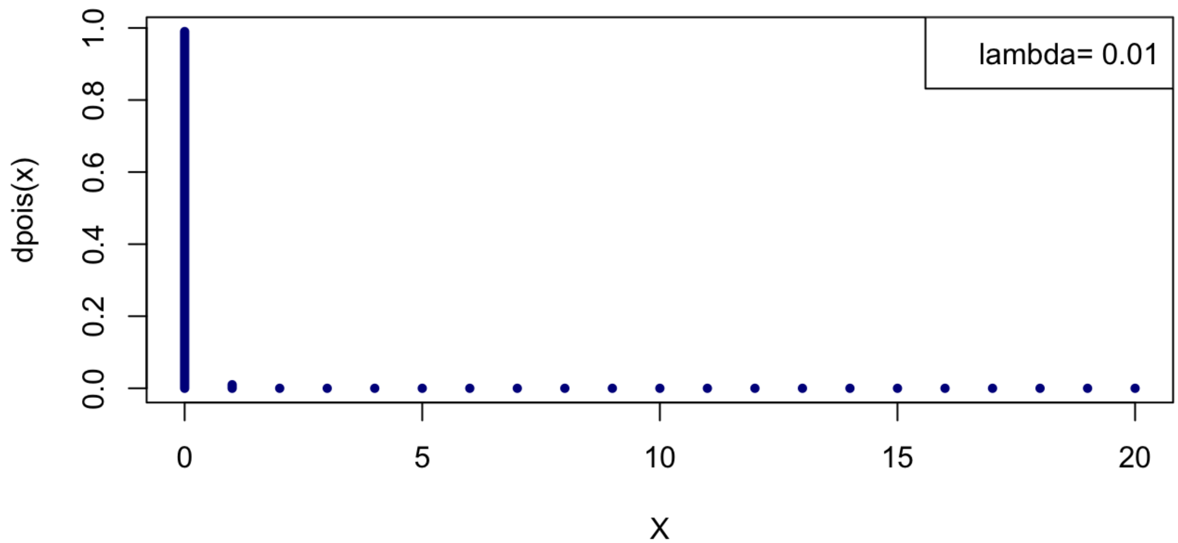
Poisson: a family of curves

Exercice: explore the properties of the Poisson density function, by changing the rang of x values, and the λ parameter.

Solution: a family of Poisson curves

$lambda = 0.01$

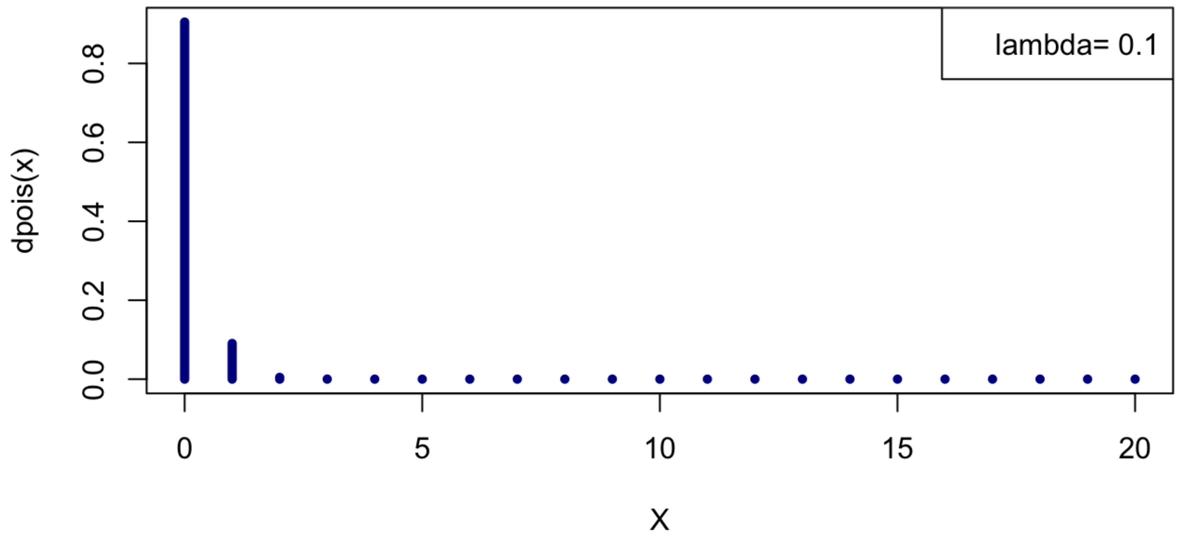
```
lambda <- 0.01
x <- 0:20
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=5, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 0.1$

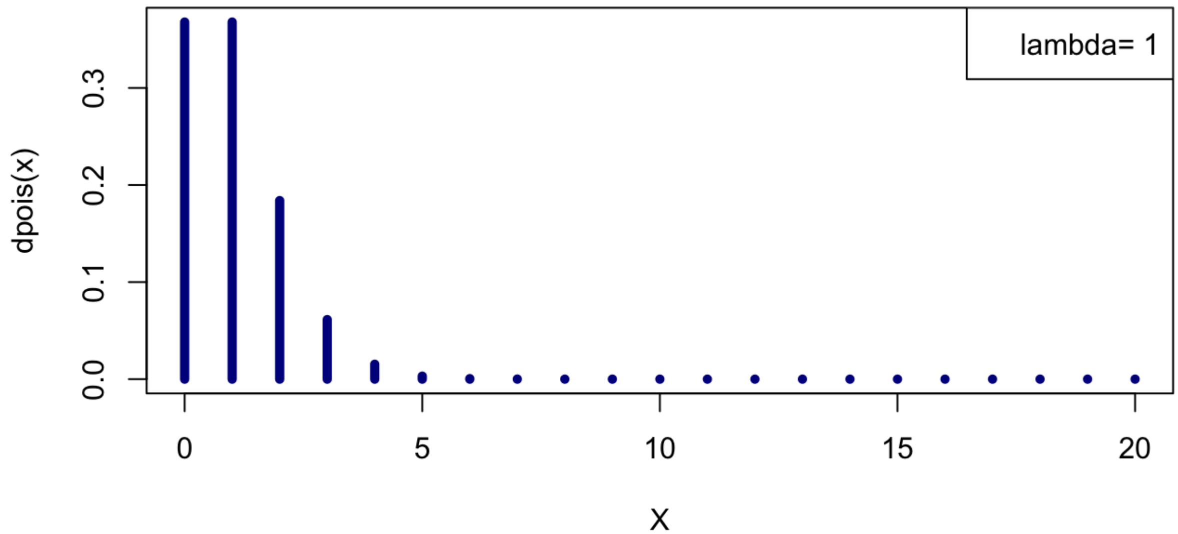
```
lambda <- 0.1
x <- 0:20
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=5, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 1$

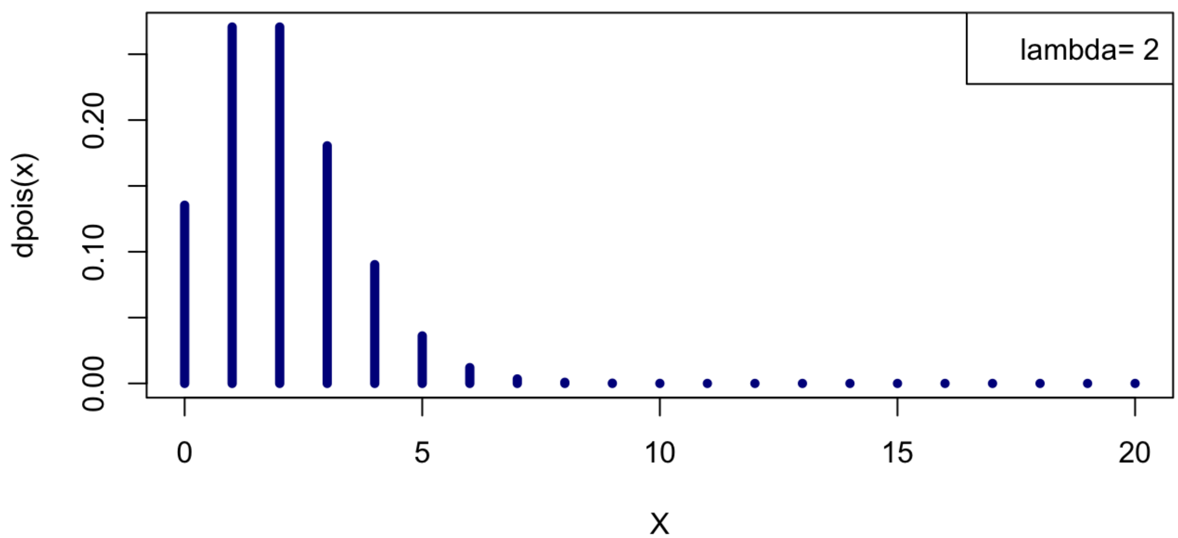
```
lambda <- 1
x <- 0:20
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=5, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 2$

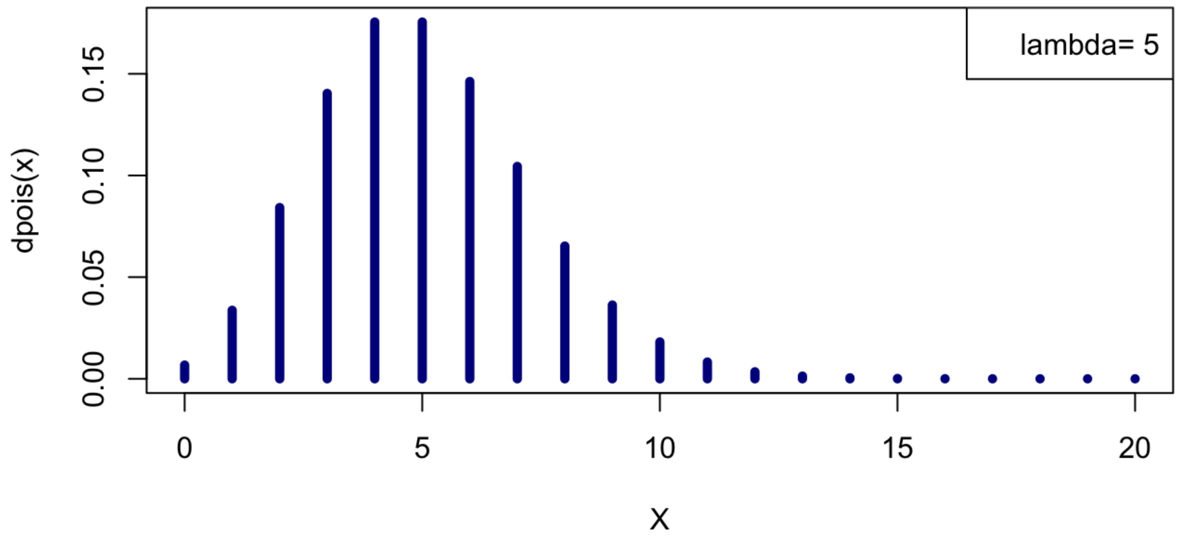
```
lambda <- 2
x <- 0:20
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=5, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 5$

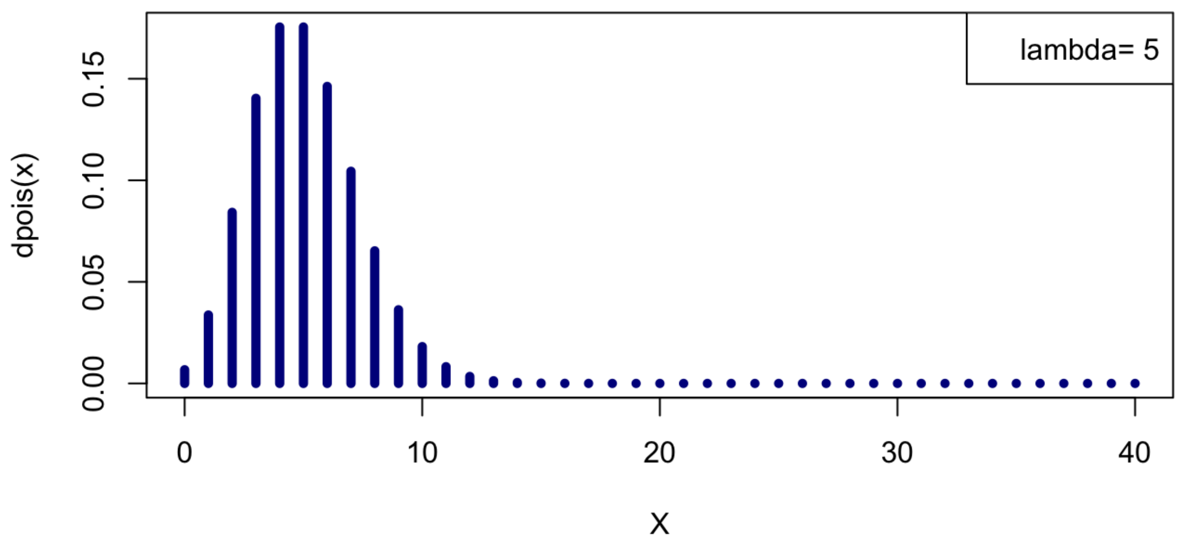
```
lambda <- 5
x <- 0:20
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=5, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 5$

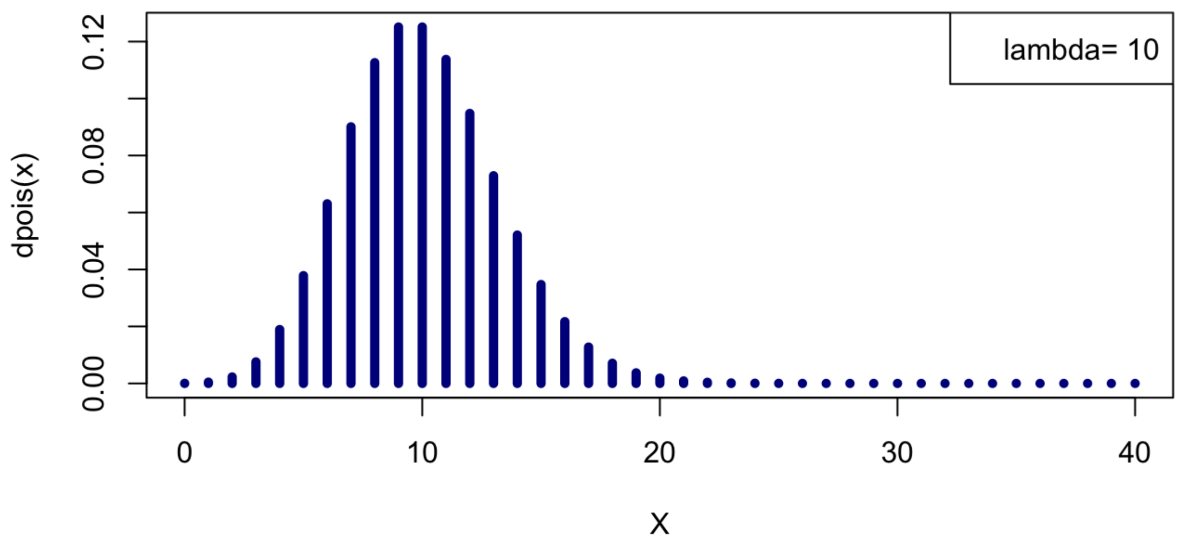
```
lambda <- 5
x <- 0:40
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=5, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 10$

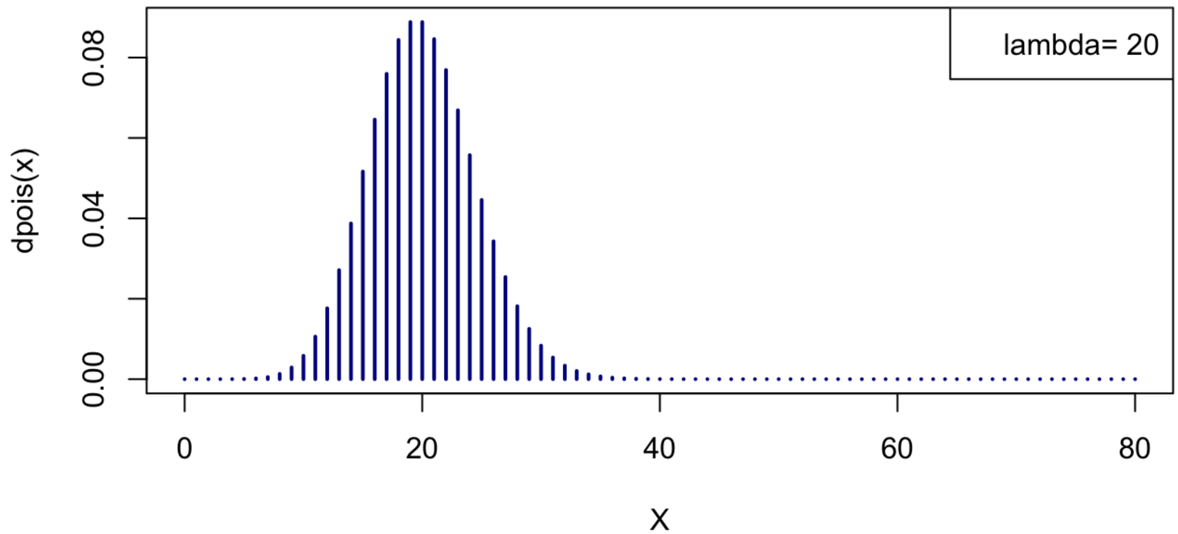
```
lambda <- 10
x <- 0:(4*lambda)
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=5, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 20$

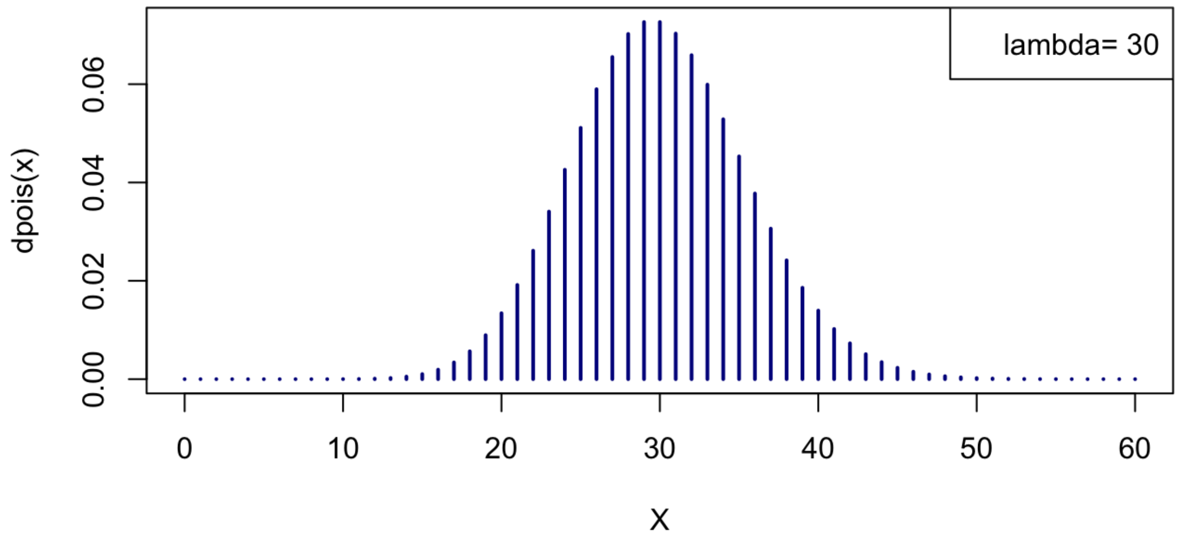
```
lambda <- 20
x <- 0:(4*lambda)
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=2, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 30$

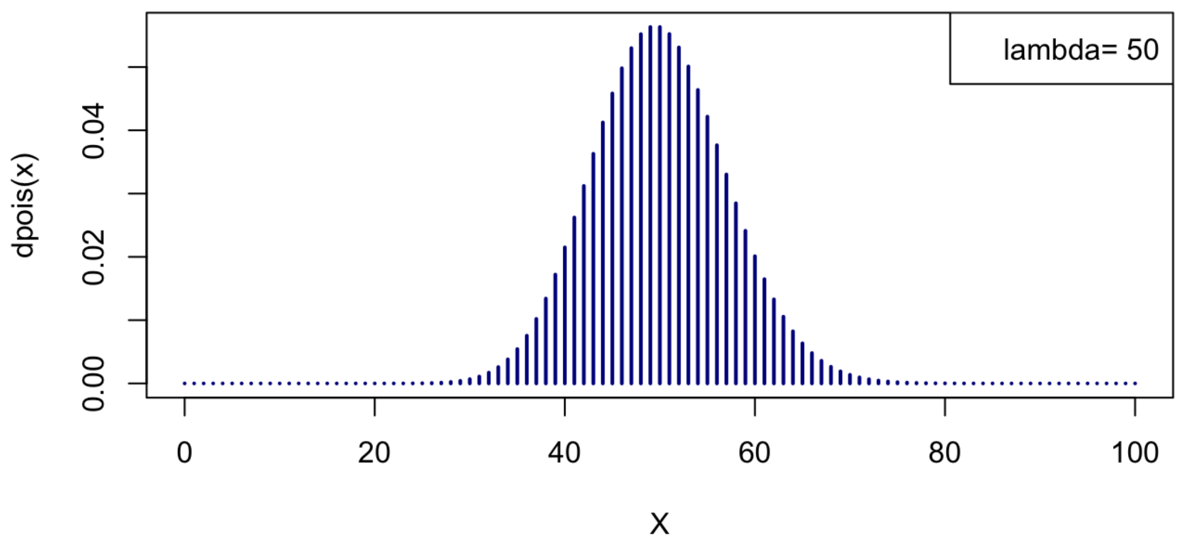
```
lambda <- 30
x <- 0:(2*lambda)
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=2, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 50$

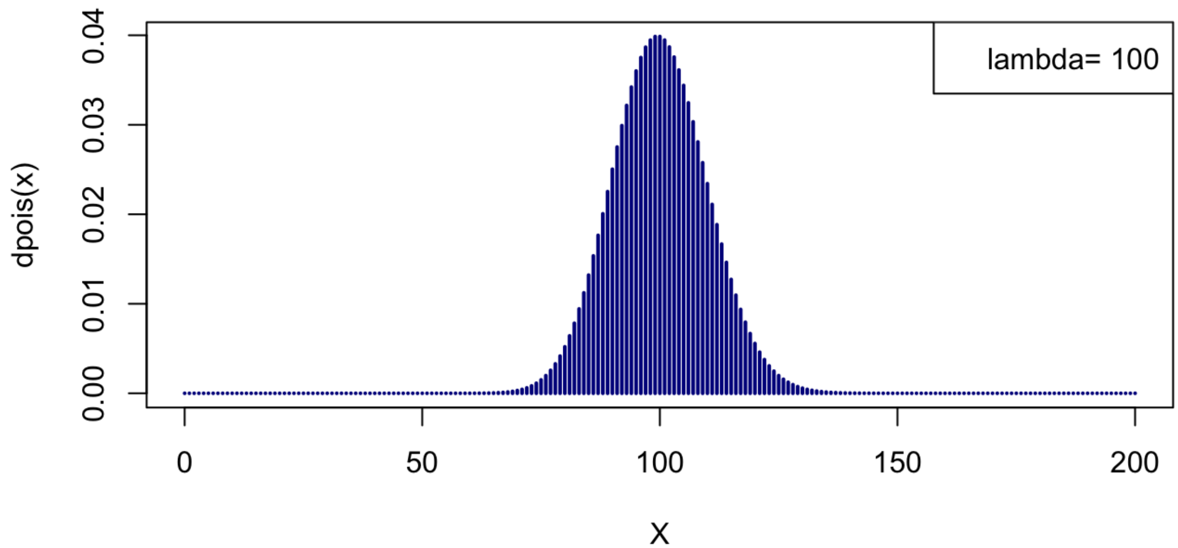
```
lambda <- 50
x <- 0:(2*lambda)
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=2, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 100$

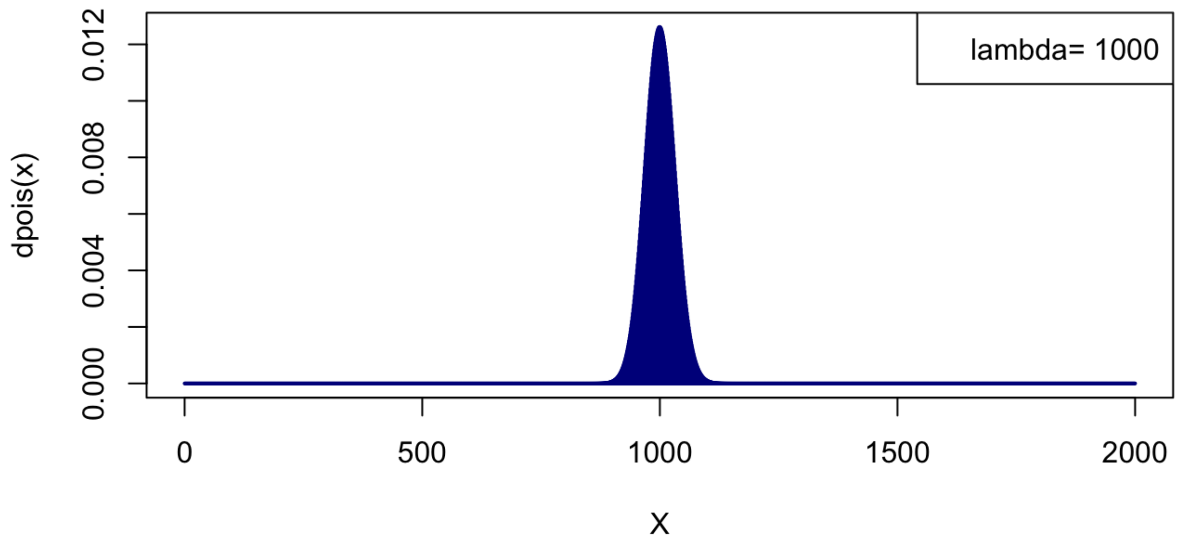
```
lambda <- 100
x <- 0:(2*lambda)
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=2, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 1000$

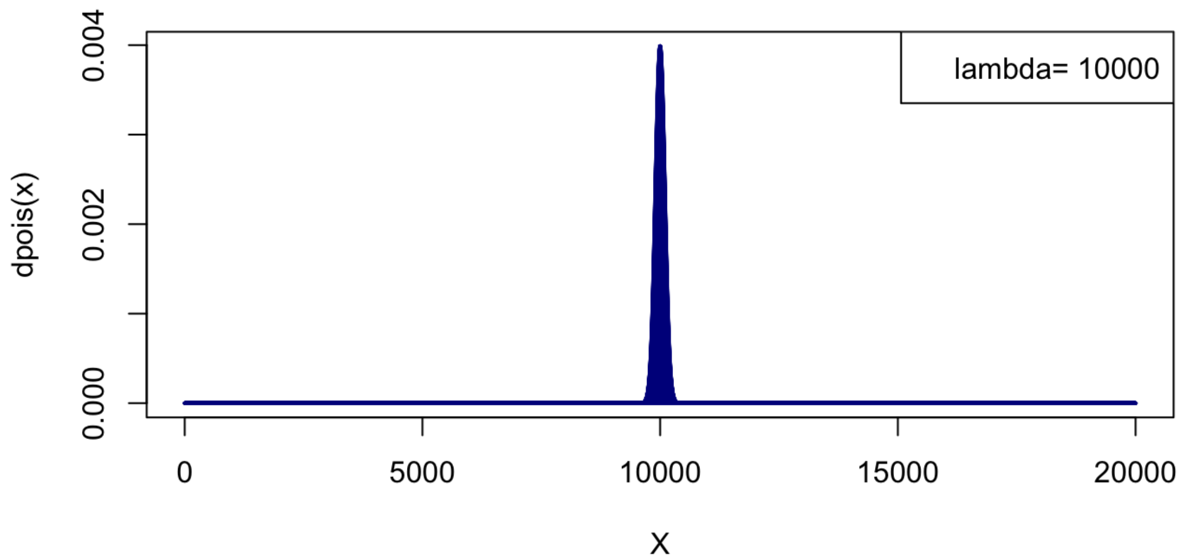
```
lambda <- 1000
x <- 0:(2*lambda)
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=2, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 10000$

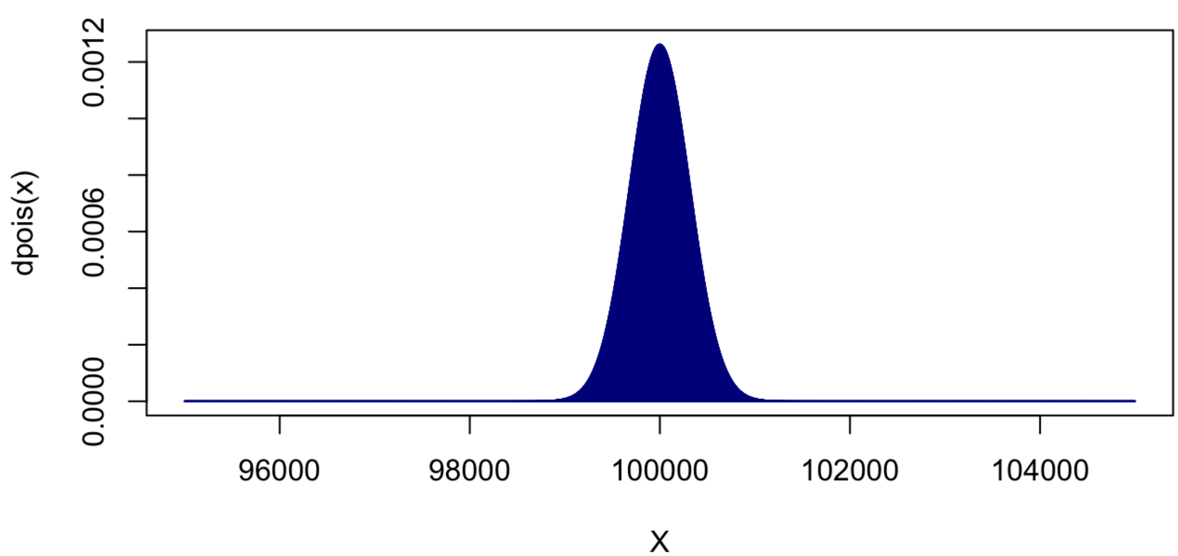
```
lambda <- 10000
x <- 0:(2*lambda)
plot(x, dpois(x,lambda=lambda), type="h",
     col="darkblue", lwd=2, xlab="X",ylab="dpois(x)")
legend("topright", paste("lambda=",lambda))
```



Solution: a family of Poisson curves

$lambda = 100000$

```
plot(95000:105000, dpois(95000:105000,lambda=100000), type="h", col="darkblue", xlab="X",ylab="dpois(x)
```



Before finishing – keep track of your session

Tractability is an important issue in sciences. The ***R*** function `sessionInfo()` summarizes information about the versions of R, the operating system, and all the libraries used during a session.

```
sessionInfo()
```

```
R version 3.3.2 (2016-10-31)
Platform: x86_64-apple-darwin13.4.0 (64-bit)
Running under: macOS Sierra 10.12.2

locale:
[1] en_US.UTF-8/en_US.UTF-8/en_US.UTF-8/C/en_US.UTF-8/en_US.UTF-8

attached base packages:
[1] stats     graphics  grDevices utils     datasets  methods   base

other attached packages:
[1] knitr_1.15.1

loaded via a namespace (and not attached):
 [1] backports_1.0.4 magrittr_1.5    rprojroot_1.1   tools_3.3.2
 [5] htmltools_0.3.5 yaml_2.1.14     Rcpp_0.12.8     stringi_1.1.2
 [9] rmarkdown_1.3   stringr_1.1.0   digest_0.6.10   evaluate_0.10
```