R doc: Continuous probability models

Josep L. Carrasco
Bioestadística. Departament de Fonaments Clínics
Universitat de Barcelona

Continuous uniform

Example. In a neurological test the subject has to press a button when a light signal is activated. The signal may appear in any time within an interval of 10 seconds.

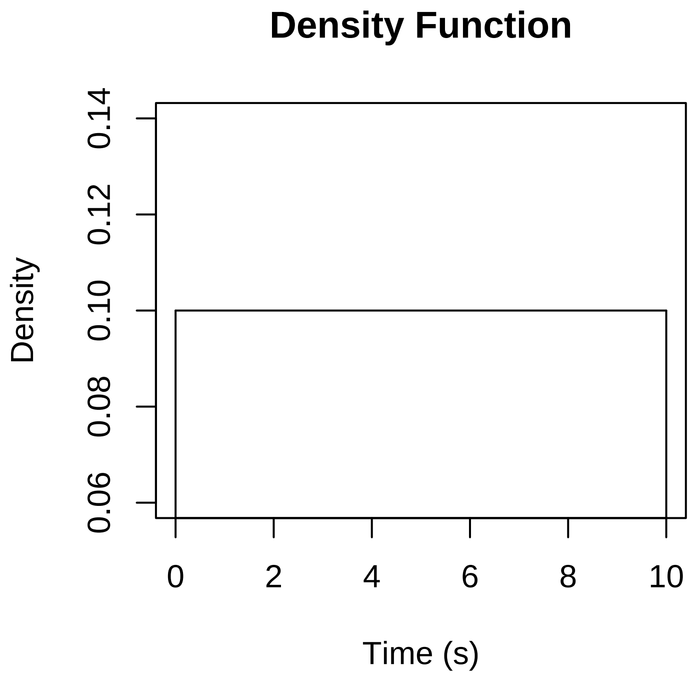
Probability density and distribution functions

Probability density function (pdf) is obtained by applying the *dunif(x,min,max)* where:

- *x*. Value where the function is evaluated.
- *min, max*: lower and upper limits of the distribution.

Let's plot the pdf for the uniform distribution of the model.

```
x=seq(from=0,to=10,by=0.1)
dx=dunif(x,0,10)
plot(x,dx,type="l",main="Density Function",xlab="Time (s)",ylab="Density")
lines(c(0,0),c(0,0.1))
lines(c(10,10),c(0,0.1))
```



The distribution function is obtained with the function *punif(x,min,max)*.

- What is the probability that the signal appears between 2 and 4 seconds?

$$P\left(2 < X < 4\right) = P\left(X < 4\right) - P\left(X < 2\right)$$

```
punif(4,0,10)-punif(2,0,10)
```

```
[1] 0.2
```

Normal

Example. The time that a cell needs to divide in two genetically identical cells (mitosis) is distributed as a Normal with mean of 1 hour and a standard deviation of 5 minutes.

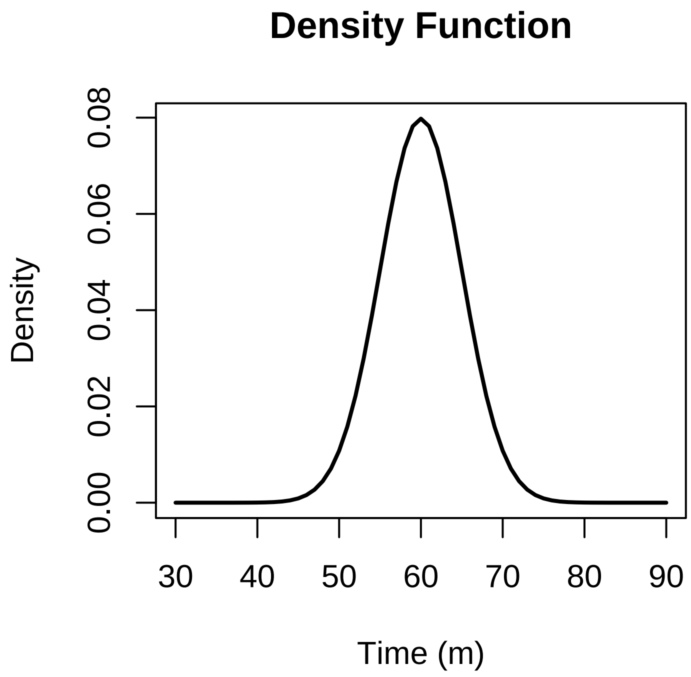
Probability density and distribution functions

The function *dnorm(x,mean,sd)* gives the probability density of a Normal model where:

- *x* is the value where the density function is evaluated.
- *mu* is the mean of the variable.
- *sd* stands for the standard deviation of the variable.

Let's plot the density function of the example's Normal model.

```
x=seq(from=30,to=90,by=1)
dx=dnorm(x,60,5)
plot(x,dx,type="l",main="Density Function",xlab="Time (m)",
     ylab="Density",lwd=2)
```



The distribution function is obtained by using the *pnorm(x,mu,sd)* function.

- What is the probability that the mitosis is complete before 45 minutes?

$$P\left(X < 45\right)$$

```r
pnorm(45,60,5)
```

```
[1] 0.001349898
```

The quantiles can be obtained with the *qnorm(p,mu,sd)* function, where p is the quantile.

- The 99% of the cells, how much time will they need to be divided?

```r
qnorm(0.99,60,5)
```

```
[1] 71.63174
```

Log-normal

Example. The logarithm of the blood triglycerides concentration in healthy people follows a Normal distribution with mean 5.5 and standard deviation 0.5.

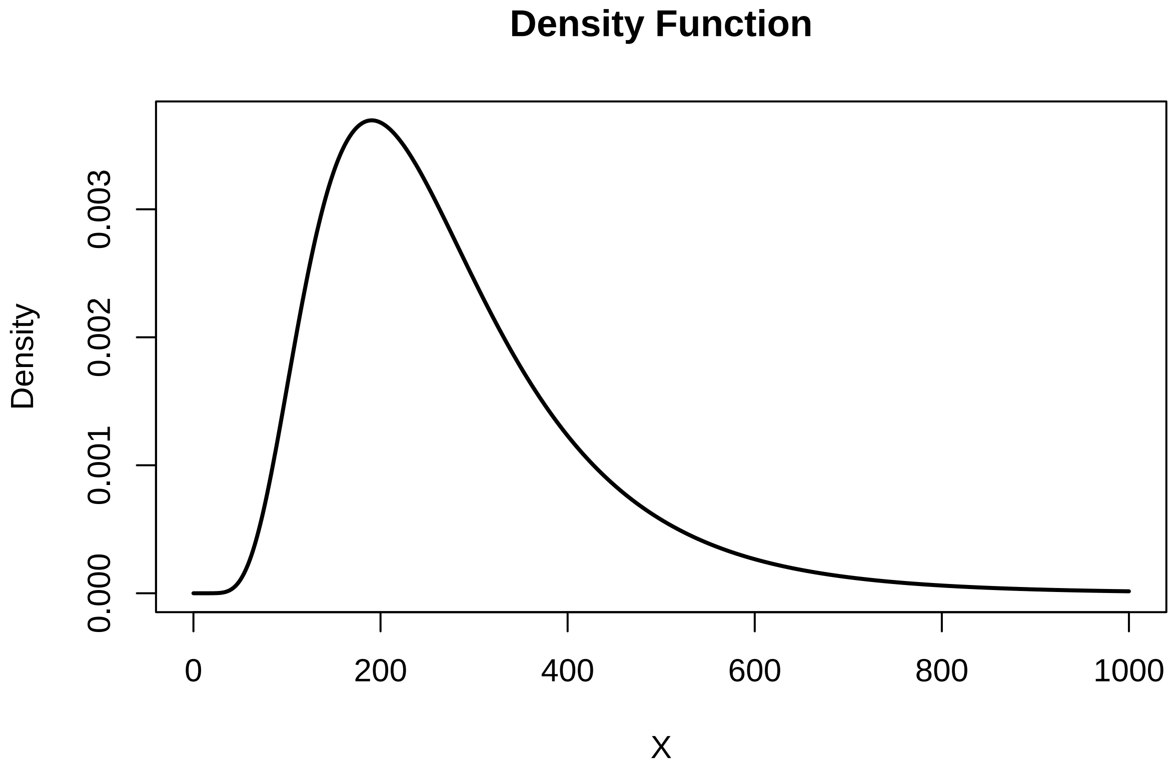
Probability density and distribution functions

Use the functions *dlnorm(x,meanlog,sdlog)* and *plnorm(x,meanlog,sdlog)* to get the density and distribution functions.

- *x*. The value where the density/distribution function is evaluated.
- *meanlog*. The mean of the logarithm of the variable.
- *sdlog*. The standard deviation of the logarithm of the variable.

Let's plot the probability density function from the example.

```
x=seq(from=0,to=1000,by=1)
dx=dlnorm(x,5.5,0.5)
plot(x,dx,type="l",main="Density Function",xlab="X",
     ylab="Density",lwd=2)
```



- If a subject is chosen at random, what is the probability that its blood triglyceride concentration is greater than 400?

$$P\left(X>400\right)=1-P\left(X<400\right)$$

```
1-plnorm(400,5.5,0.5)
```

```
[1] 0.1628212
```

The quantiles are obtained with the function *qlnorm*.

- Between which values it is possible to find the 90% of the blood triglyceride concentrations.

```
qlnorm(0.05,5.5,0.5)
```

```
[1] 107.5089
```

```
qlnorm(0.95,5.5,0.5)
```

```
[1] 556.9229
```

Actually we could define infinite intervals containing the 90% of the probability. This one is that symmetric in relation to the probability, so that there is a 5% left at each side.

Exponential

Example. The number of defective components that a machine produces every month follows a Poisson process with rate 30 components per month (30 days). So that, the production time to next defective component follows an Exponential distribution with parameter equal to 1 component a day.

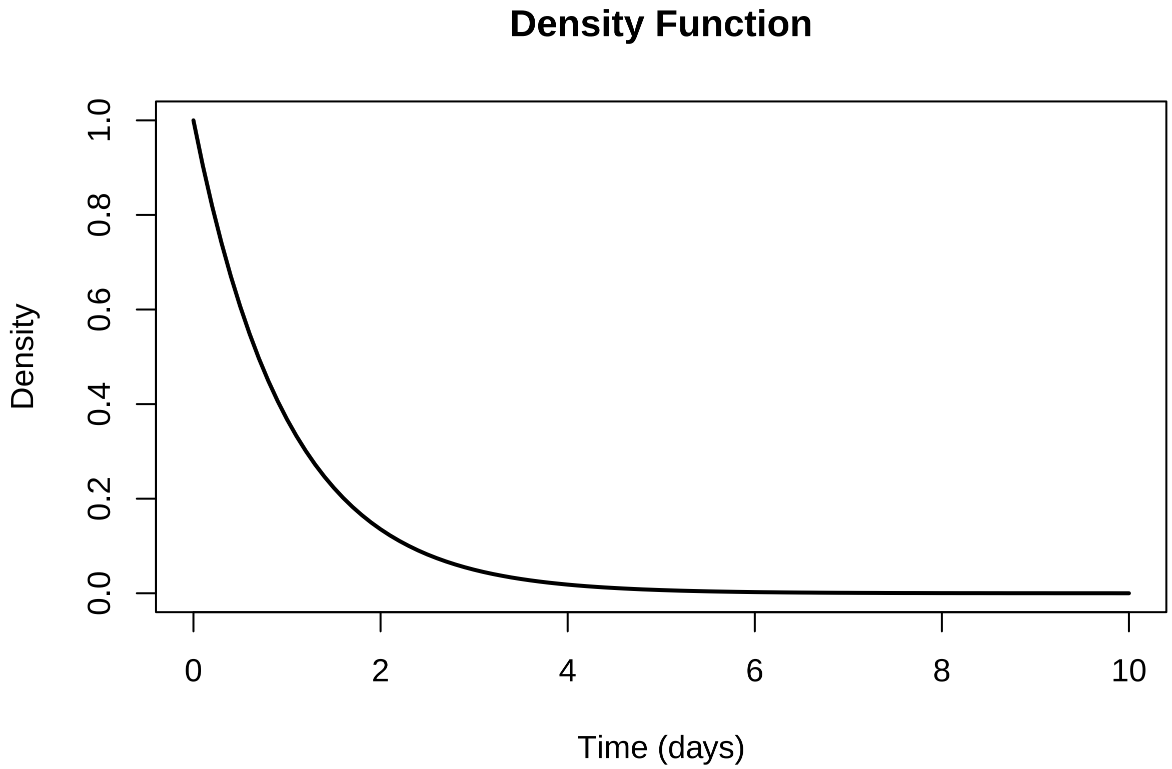
Probability density and distribution functions

Use *dexp(x,rate)* and *pexp(x,rate)* to get the probability density and distribution functions.

- *x*. The value where the density/distribution function is evaluated.
- *rate*. The rate of events.

Let's plot the density function of the example.

```
x=seq(from=0,to=10,by=0.1)
dx=dexp(x,1)
plot(x,dx,type="l",main="Density Function",xlab="Time (days)",
     ylab="Density",lwd=2)
```



- What is the probability that the next defective component takes 3 days or less?

$$P\left(X < 3\right)$$

```
pexp(3,1)
```

```
[1] 0.9502129
```

The quantiles are obtained with the function *qexp(p,rate)*.

- With a probability of 90%, how long will it take to produce a defective component?

```
qexp(0.9,1)
```

```
[1] 2.302585
```

Gamma

Example. Continuing with the previous example, let's define now X as the time needed to produce 3 defective components.

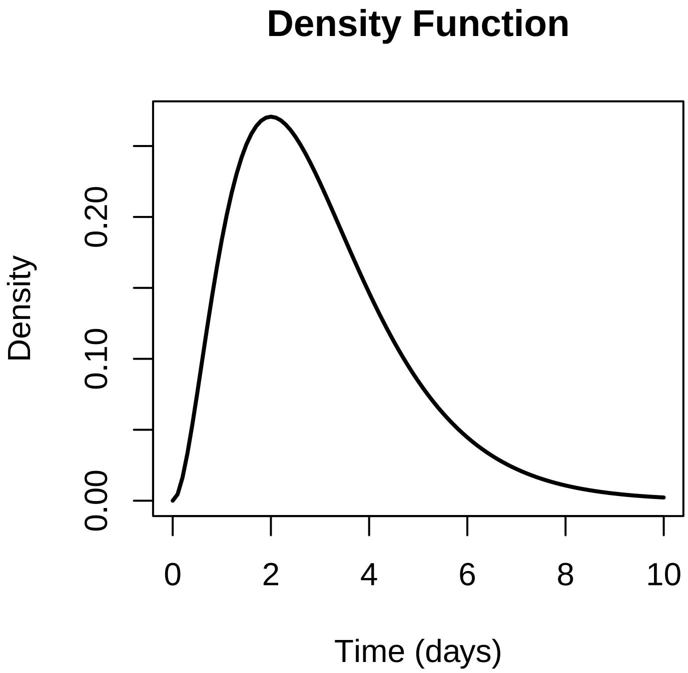
Probability density and distribution functions

Use *dgamma(x,shape,rate)* and *pgamma(x,shape,rate)* to get the probability density and distribution functions.

- *x.* The value where the density/distribution function is evaluated.
- *shape.* Shape parameter.
- *rate.* The rate of events.

Let's plot the density function of the example.

```
x=seq(from=0,to=10,by=0.1)
dx=dgamma(x,shape=3,rate=1)
plot(x,dx,type="l",main="Density Function",xlab="Time (days)",
     ylab="Density",lwd=2)
```



- What is the probability of producing 3 defective components in 3 days?

$P\left(X<3\right)$

```
pgamma(3,shape=3,rate=1)
```

```
[1] 0.5768099
```

Use the function *qgamma(p,shape,rate)* to optain the p*th* quantile.

- With a probability of 90%, how long will it take to find 3 defective components?

```
qgamma(0.9,shape=3,rate=1)
```

```
[1] 5.32232
```

Weibull

Example. Some researchers are using computational intensive methods in a computer to perform a simulation study. The computer memory gets overloaded every 200 hours in mean. On the other hand, this overloading increases with the time of use of the computer. It is assumed that the time to the computer failure follows a Weibull model with parameters $\beta = 2$ and $\delta = 200$.

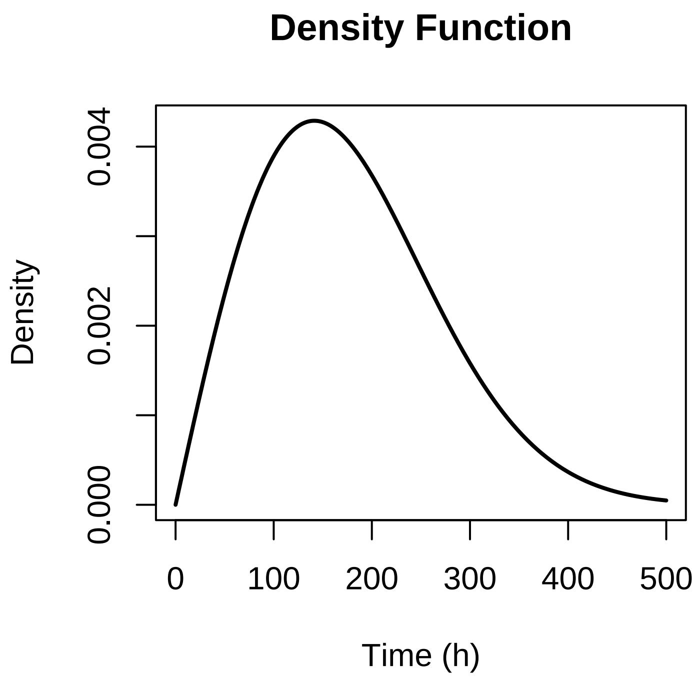
Probability density and distribution functions

Use *dweibull(x,shape,scale)* and *pweibull(x,shape,scale)* to get the probability density and distribution functions.

- *x.* The value where the density/distribution function is evaluated.
- *shape.* Shape parameter.
- *scale.* Scale parameter.

Let's plot the density function of the example.

```
x=seq(from=0,to=500,by=1)
dx=dweibull(x,shape=2,scale=200)
plot(x,dx,type="l",main="Density Function",xlab="Time (h)",
     ylab="Density",lwd=2)
```



- What is the probability that the memory gets overloaded through the first day of use? \ $P(X < 24)$

```
pweibull(24,shape=2,scale=200)
```

```
[1] 0.01429682
```

Use the function *qweibull(p,shape,scale)* to obtain the p*th* quantile.

- In the 90% of cases, before what time the memory will get overloaded?

```
qweibull(0.9,shape=2,scale=200)
```

```
[1] 303.4854
```

Q-Q plots

Let's randomly generate 50 data from a Weibull distribution with parameters shape = 20 and scale =200.

```
set.seed(20) # To set the random seed
x=rweibull(50,shape=2,scale=200)
```

Let's assess if the data comes from a log-normal distribution. Firstly we have to determine the parameters of the theoretical model (log-normal distribution in this case). There are two options:

1) Preset values of the parameters.
2) Obtain the parameters values from the data. This is the most common option.

To get the parameters from the data we could use the function *fitdistr(x,densfun)* from the *MASS* package where:

- x: the vector data
- densfun: model to fit

```
# install.packages("MASS") #Set up the package
library(MASS) #load the package once it is set up
par.ln<-fitdistr(x,"lognormal")
par.ln$estimate
```

```
  meanlog     sdlog
5.0519452 0.6494587
```

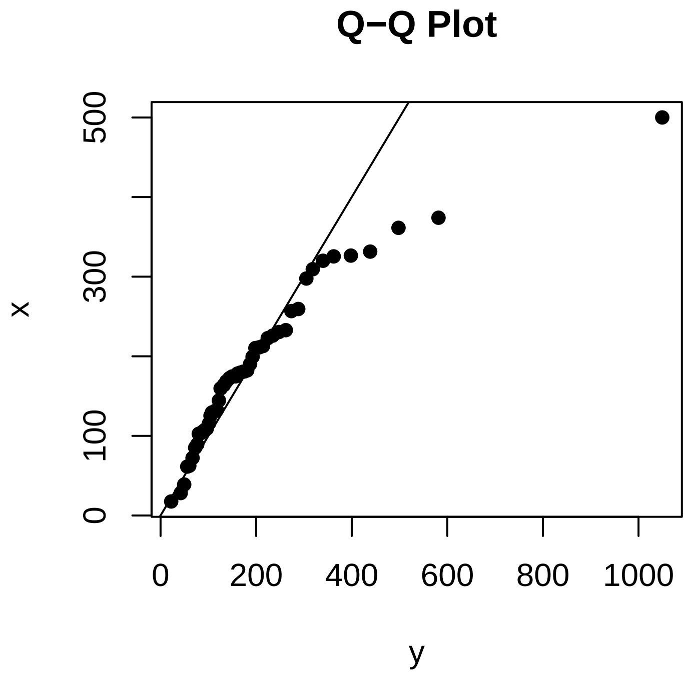
Let's draw the Q-Q plot. The function is *qqplot(y,x)* where:

- y: theoretical values
- x: data values

Theoretical values are created by generating a large number of data from the theoretical model.

```
par(pch=16) # Symbol to draw. A filled circle in this case
y<-rlnorm(1000,par.ln$estimate[1],par.ln$estimate[2])

qqplot(y,x, main="Q-Q Plot")
abline(0,1) # Draw the concordance line
```



Some points are too far from the concordance line to assume that data comes from a log-normal model.

Let's try now the Weibull model. First we must estimate the parameters

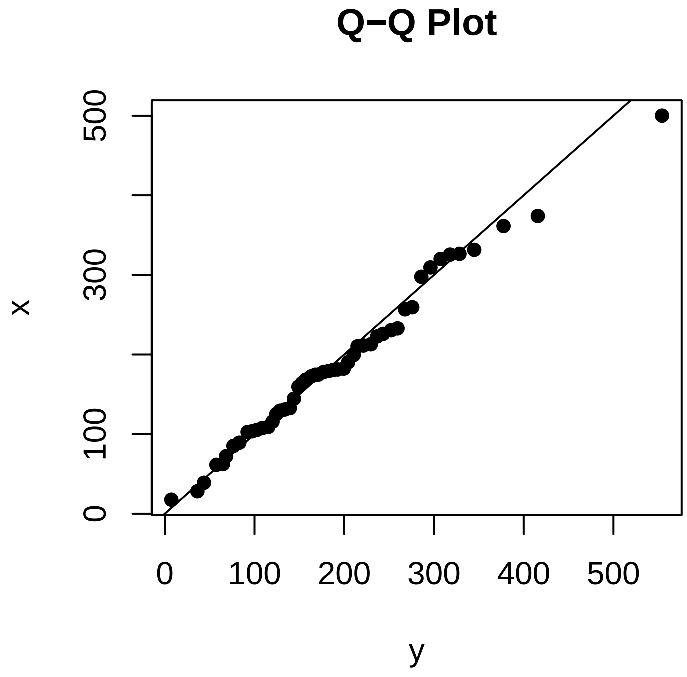
```
par.w<-fitdistr(x,"weibull")
par.w$estimate
```

```
     shape       scale 
  1.969649 209.662650 
```

Draw the Q-Q plot

```
y<-rweibull(1000,par.w$estimate[1],par.w$estimate[2])

qqplot(y,x, main="Q-Q Plot",pch=16)
abline(0,1)
```



In this case it is quite acceptable to assume data comes from a Weibull model (which actually is the model used to generate the data).