MortalityLaws R package

Marius Pascariu

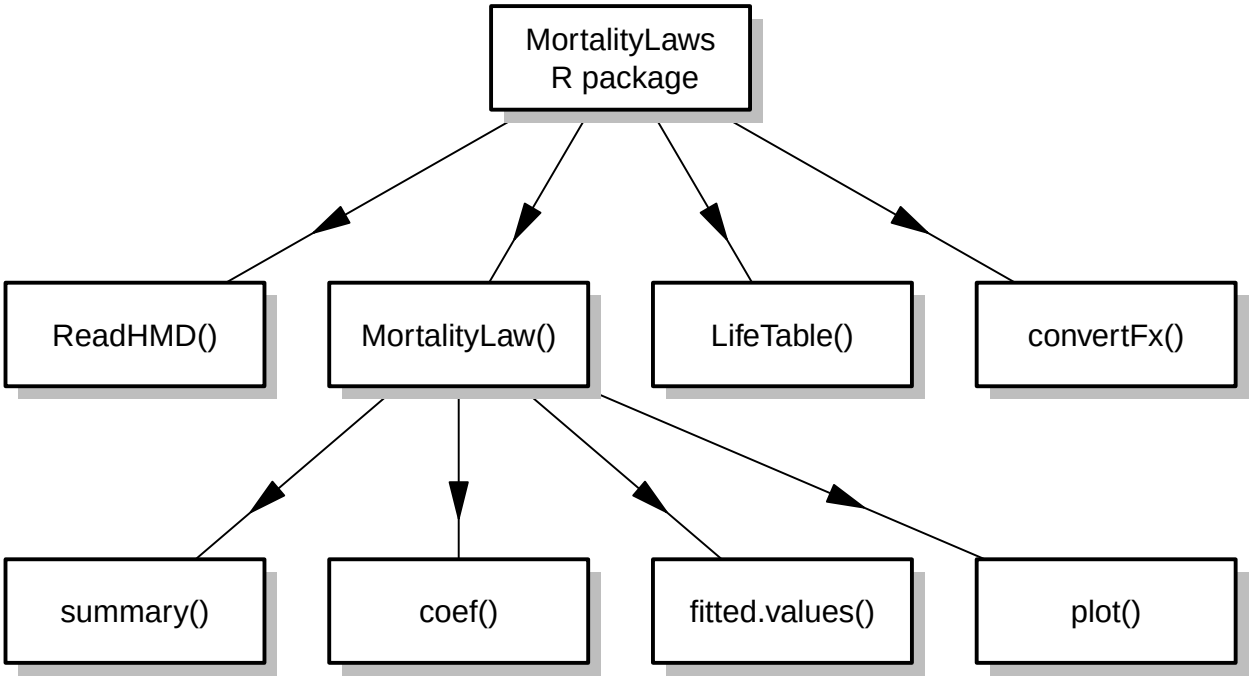
2017-06-07

Warning: This is just a draft and might change at any time

`MortalityLaws` in an R package which exploits the available optimization methods to provide tools for fitting and analyzing a wide range of mortality models.

Three main functions are available in the package: `ReadHMD`, `MortalityLaw`, `LifeTable` and `convertFx`. The package provides also generic functions like `summary`, `plot`, `coef`, `fitted.values` and a small data set for testing purposes `ahmd`.

The structure of the MoralityLaws R package



All functions are documented in the standard way, which means that once you load the package you can just type for example `?MortalityLaw` to see the help file for `MortalityLaw` function.

Installation

`MortalityLaws` is a software in development. The repository containing the development version of the package can be found on GitHub and can be installed by running the following code in the R console:

```
# Make sure you have the most recent version of R and devtools package already installed.
# install.packages("devtools")
devtools::install_github("mpascariu/MortalityLaws")
```

The package is loaded within **R** as follows:

```
library(MortalityLaws)
```

Download HMD Data

Download data form Human Mortality Database using the `ReadHMD` function:

```
# Download HMD data - death counts
HMD_Dx <- ReadHMD(what = "Dx",
                  countries = "SWE",  # HMD country code for Sweden
                  interval = "1x1",             # 1 year - 1 age data
                  username = "user@email.com", # here add your HMD username
                  password = "password",        # here add your password account
                  save = FALSE)                 # Do you what to save the data outside R
```

Here we have downloaded all the registered death counts in Sweden from 1751 until 2014. In the same way one can download the following records: birth counts, population size, lexis triangles, exposures, life tables, life expectancy at birth and death rates for over 38 countries and regions in different formats.

Download life tables all Swedish life tables for female population and save them in an external object:

```
HMD_Ex <- ReadHMD(what = "LT_f",
                  countries = "SWE",
                  interval = "1x1",
                  username = "user@email.com",
                  password = "password",
                  save = TRUE)
```

Model fitting and diagnosis

Once we have data from HMD or other sources we can start analyzing it. For example, let's fit a Heligman-Pollard model under a Poisson setting which is already implemented as one of the standard models in the package[1]. We have to use the `MortalityLaw` function in this regard. In the examples provided below we will use the test data provided in the package `ahmd`.

```
# Select 1 year of data, an age-range and sex
year = 1950
ages = 0:100
deaths <- ahmd$Dx[paste(ages), paste(year)]
exposure <- ahmd$Nx[paste(ages), paste(year)]

fit <- MortalityLaw(x = ages,
                    Dx = deaths, # vector with death counts
                    Ex = exposure, # vector containing exposures
                    law = "HP",
                    how = "poissonL",
                    fit.this.x = 0:65)

ls(fit) # inspect the output object
```

```
## [1] "coefficients"        "fitted.values"       "goodness.of.fit"
## [4] "info"                "input"               "optimization.object"
## [7] "residuals"
```

[1] Check the Appendix to see all the implemented mortality laws.

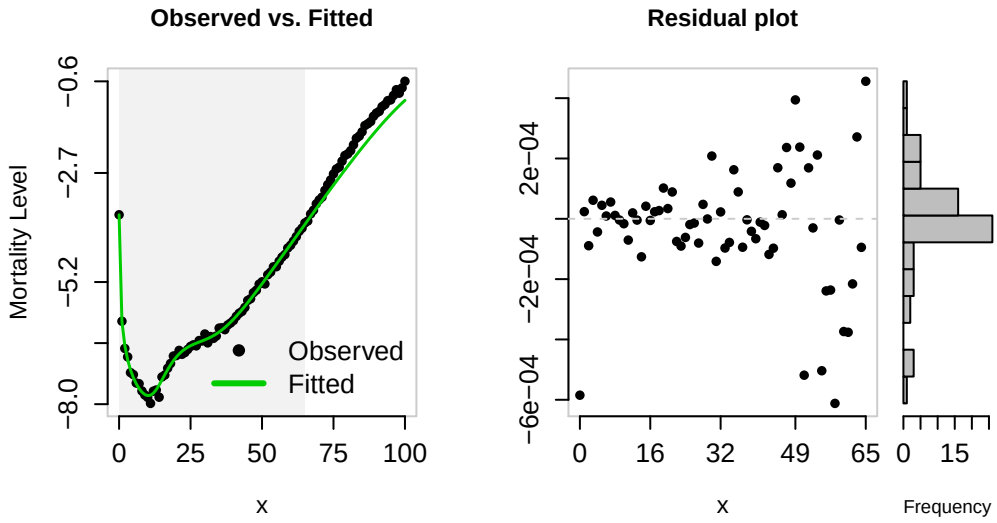


Figure 1: Heligman-Pollard model fitted using the MortalityLaw function

A summary can be obtained using the `summary` function.

```
summary(fit)
```

```
## Heligman-Pollard (1980):
## q(x)/p(x) = A^((x+B)^C) + D*exp(-E*(log(x/F))^2) + G*H^x)
##
## Deviance Residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.00061 -0.00004  0.00005  0.01096  0.01286  0.07611
##
## Coefficients:
##        a        b        c        d        e        f        g        h
##  0.00220  0.01732  0.12851  0.00068  4.47132 24.95710  0.00005  1.09773
##
## Log-Likelihood =  13.07  AIC =  -10.14  BIC =  7.38
```

The standard plot helps us to investigate visually the goodness of fit.

```
plot(fit)
```

The gray area on the plot showing the fitted value indicates the age range used in fitting the model. This is specified in the model design with the help of `fit.this.x` argument, which can be adjusted accordingly.\

We can use the fitted values to compute life tables. This can be achieved using the `lifetable` function.

```
lt <- LifeTable(x = ages, mx = fitted.values(fit))
ls(lt)
```

```
## [1] "lt"           "lt.exact"     "process_date"
```

```
head(lt$lt)
```

```
##   age       mx       qx  ax     lx   dx    Lx      Tx    ex
## 1   0 0.025785 0.025200 0.1 100000 2520 99748 7256965 72.75
## 2   1 0.002220 0.002217 0.5  97480  216 97372 7157217 73.50
## 3   2 0.001292 0.001292 0.5  97264  126 97201 7059846 72.63
## 4   3 0.000930 0.000929 0.5  97138   90 97093 6962645 71.71
```

```
## 5   4 0.000735 0.000735 0.5  97048   71 97012 6865551 70.77
## 6   5 0.000615 0.000615 0.5  96977   60 96947 6768539 69.82
```

Now let's fit a mortality law that is not defined in the package, say a reparametrize version of Gompertz in terms of modal age at death

$$\mu_x = \beta e^{\beta(x-M)}. \tag{1}$$

We have to define a function containing the desired hazard function and then using the `custom.law` argument it can be used in the `MortalityLaw` function.

```
# Select the data that we want to fit
year = 2010
ages = 35:75
deaths <- ahmd$Dx[paste(ages), paste(year)]
exposure <- ahmd$Nx[paste(ages), paste(year)]

# Here we define a function for our new model and provide start parameters
my_gompertz <- function(x, par = c(b = 0.13, M = 45)){
  hx  <- with(as.list(par), b*exp(b*(x - M)) )
  return(as.list(environment())) # return everything inside this function
}

# Use 'custom.law' argument to instruct the MortalityLaw function how to behave
my_model <- MortalityLaw(x = ages,
                         Dx = deaths,
                         Ex = exposure,
                         custom.law = my_gompertz)
summary(my_model) # these are the results
```

```
## Custom Model
##
## Deviance Residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -0.00078 -0.00007  0.00001  0.00001  0.00009  0.00128
##
## Coefficients:
##        b        M
##  0.09196 90.71737
##
## Log-Likelihood =  12.88  AIC =  -21.76  BIC =  -18.33
```

```
plot(my_model)
```

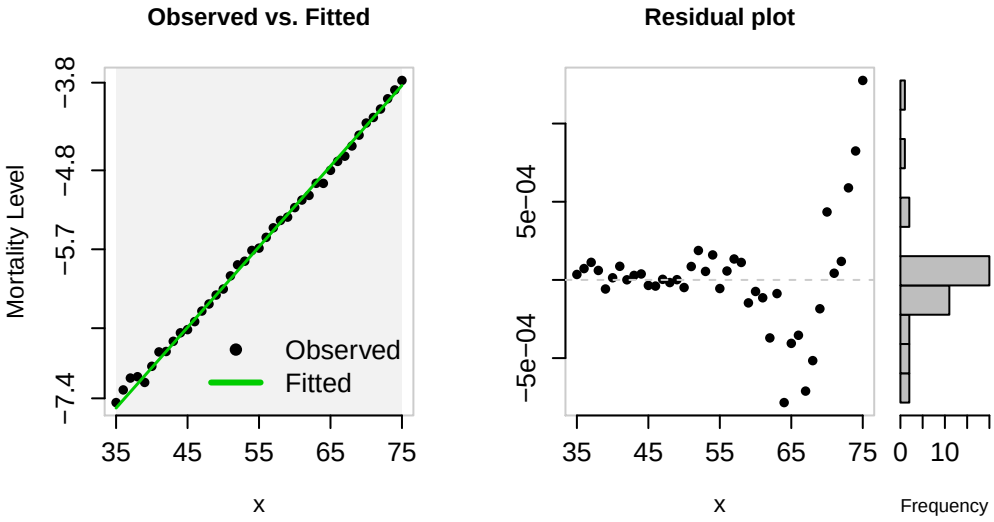


Figure 2: Gompertz model fitted using the MortalityLaw function