Basic Course in R for Historical Demography

Francesco Scalone
Società italiana di Demografia Storica
(Italian Society of Historical demography)
francesco.scalone@unibo.it

July 7, 2023

Contents

Part I

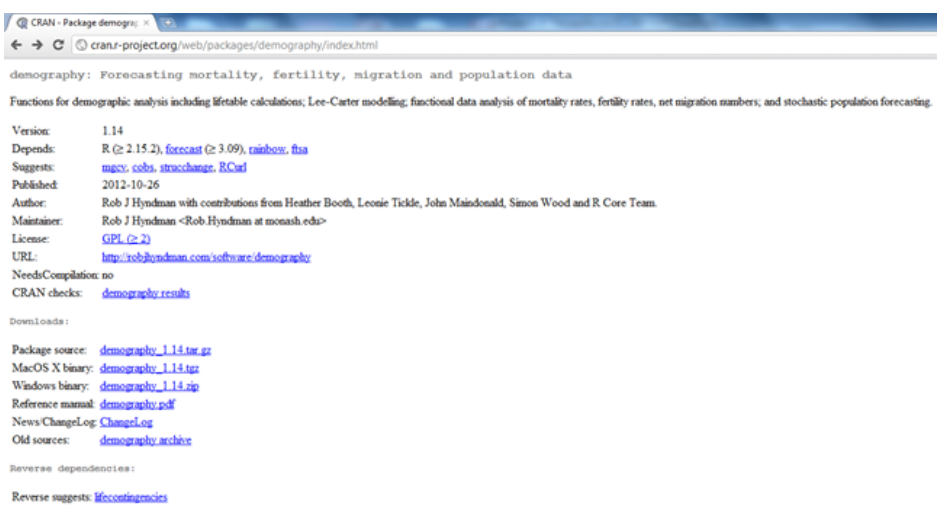
Introduction to the Demography Package

I.1 The package Demography

The package "Demography" by provides several functions and commands to carry out specific calculations in the demographic field such as computing lifetables and fertility measures. In addition, the Demography package estimates the Lee-Carter model to project death probabilities by ages with prediction intervals. Future mortality and fertility rates are also forecasted according to the functional data analysis approach.

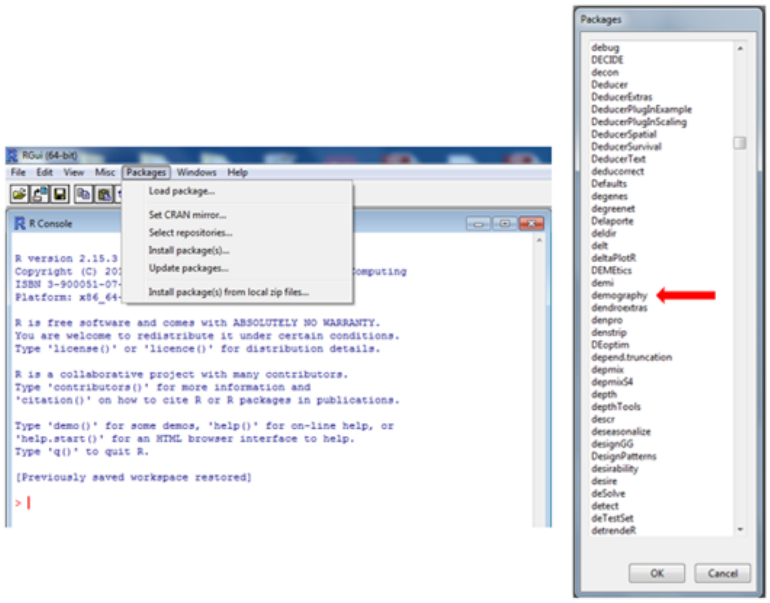
The package is primarily maintained by Rob J Hyndman, with contributions from Heather Booth, Leonie Tickle, John Maindonald, Simon Wood, and the R Core Team. The maintainer of the package can be contacted at Rob.Hyndman@monash.edu. Their expertise and dedication have been instrumental in creating and maintaining this valuable resource for demographic analysis in R.

This concise introduction provides a beginner-friendly introduction to utilizing the R Demography package for historical demography research, tailored for researchers who are new to using R. We will explore some fundamental concepts and methodologies in demographic analysis, with a specific focus on age-specific death rates, life tables, and total fertility rates. Through the accessible and user-friendly features of the R Demography package, we will show the essential tools to conduct insightful analyses of historical demography.



I.2 Install the Demography Package

The "Demography" package can be installed either manually or, as recommended, using the specific commands in R. After installation, it is important to remember to load the package using the "library" command.



To download and install the "Demography" package from the comprehensive R Archive Network, use the following specific commands:

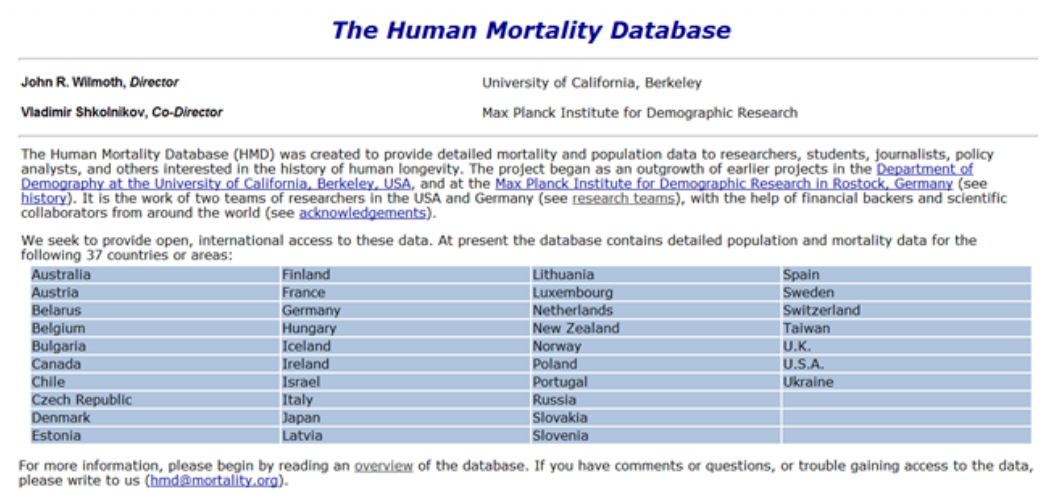
```
install.packages("demography", dependencies =
    TRUE)
library(demography)
```

I.3 The Human Mortality Database

This package also offers the possibility to download directly from the Human Mortality Database (henceforth, HMD) a set of mortality rates related to a wide range of countries. Before being able to directly download mortality data from the HMD, it is necessary to create an account and password at

the following website: http://www.mortality.org/

The Human Mortality Database provides mortality data considering a number of countries around the world situated in Europe, Northern America, Asia and Australia. The countries were selected since they offer a high-quality demographic data spanning on a very long period. In some cases, the mortality data begun before the demographic transition (for example, the Swedish data starts from the middle of the Eighteen century). Basing on detailed information on annual deaths and population amounts by age and sex, the HMD make available a collection of lifetables computed according a common protocol and therefore directly comparable. In addition, both period and cohort lifetables are also available.



The Human Mortality Database

John R. Wilmoth, *Director* University of California, Berkeley

Vladimir Shkolnikov, *Co-Director* Max Planck Institute for Demographic Research

The Human Mortality Database (HMD) was created to provide detailed mortality and population data to researchers, students, journalists, policy analysts, and others interested in the history of human longevity. The project began as an outgrowth of earlier projects in the Department of Demography at the University of California, Berkeley, USA, and at the Max Planck Institute for Demographic Research in Rostock, Germany (see history). It is the work of two teams of researchers in the USA and Germany (see research teams), with the help of financial backers and scientific collaborators from around the world (see acknowledgements).

We seek to provide open, international access to these data. At present the database contains detailed population and mortality data for the following 37 countries or areas:

Australia	Finland	Lithuania	Spain
Austria	France	Luxembourg	Sweden
Belarus	Germany	Netherlands	Switzerland
Belgium	Hungary	New Zealand	Taiwan
Bulgaria	Iceland	Norway	U.K.
Canada	Ireland	Poland	U.S.A.
Chile	Israel	Portugal	Ukraine
Czech Republic	Italy	Russia	
Denmark	Japan	Slovakia	
Estonia	Latvia	Slovenia	

For more information, please begin by reading an overview of the database. If you have comments or questions, or trouble gaining access to the data, please write to us (hmd@mortality.org).

I.4 Importing data from the Human Mortality Database

The function `hmd.mx` is a function of the "Demography" package in R that allows access to demographic data from the Human Mortality Database (HMD). By using this function, you can retrieve information on life tables and other demographic indicators for different populations and specific time

periods. `hmd.mx` reads mortality rates from the HMD, constructs a demogdata object suitable for calculating and plotting life tables.
Having an internet connection, we can import directly from the HMD's website a set of age specific mortality rates for a range of years and a specific country. In order to read the data, users are required to create their account via the HMD website (http://www.mortality.org), and obtain a valid username and password.

Afterwards, it is necessary to indicate the abbreviation of the country under study from the HMD database, username and password that were previously registered on the HMD's website, and finally a label referring to country name which the data are taken. Here there is an example for Italy.

```
italy <- hmd.mx("ITA", "username", "password",
                "Italy")
```

Arguments:

country	Directory abbreviation from the HMD. For instance, Italy = "ITA"
username	HMD username
password	HMD password
label	Character string giving name of country from which the data are taken

The table at this web address lists the populations currently included in the HMD: https://www.mortality.org/Data/DataAvailability. The table also specifies the years covered by the life tables and the corresponding label to be used in the "hmd.mx" command.

The `hmd.mx` returns an object of class demogdata containing elements of different types – like strings, numbers, vectors and another list inside it. Considering the previous example, we can show the content of the Italy object.

More specifically these components refer to:

- year: Vector of years
- age: Vector of ages
- rate: A list containing rate matrices with one age group per row and one column per year
- pop: A list of the same form as rate but containing population numbers instead of demographic rates
- type: Type of object: "mortality", "fertility" or "migration"
- label: label

The first vector contains the historical years of the series, while the second component refers to the age of the specific rates. Both year and age ideally represent the columns and rows of the matrices containing respectively the demographic rates (rate vector) and the population amounts (pop vector) exposed to the risk of dying at each age. The type indicates the kind of rates considered (age-specific mortality, fertility or migration rates). Finally, label refers to the country abbreviation.

I.5 Importing other country data

At the same way, it is possible to import mortality data of other countries from the HMD, such as Sweden, England or France.

```
sweden <- hmd.mx("SWE", "username", "password", "
   Sweden")
france <- hmd.mx("FRATNP", "username", "password"
   , "France")
```

To display the years and ages covered by the mortality data, it is necessary typing the name of demographic object in the R console, as in the following cases:

```
> italy
Mortality data for Italy
Series: female male total
Years: 1872 - 2019
Ages:  0 - 110


> sweden
Mortality data for sweden
Series: female male total
Years: 1751 - 2022
Ages:  0 - 110


> france
Mortality data for France
Series: female male total
Years: 1816 - 2020
Ages:  0 - 110
```

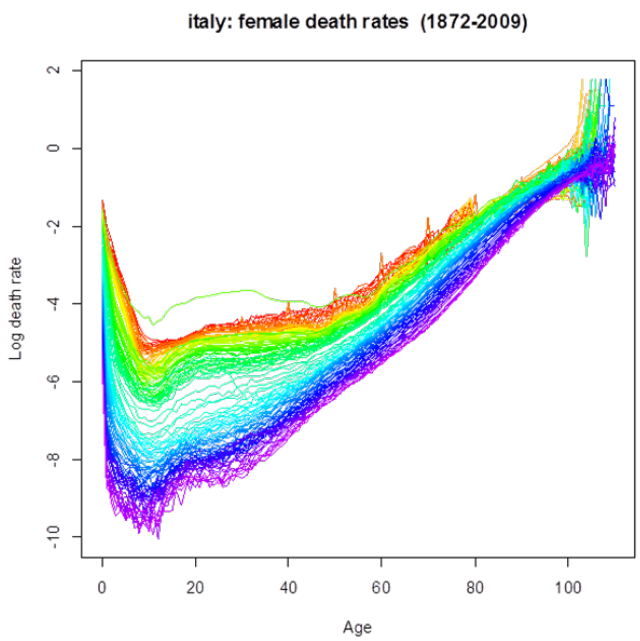
In the Italian case, the output shows that age-specific mortality data refer to male, female and total population from age 0 to 110, while the series covers a time interval longer than a century. Since the mortality data of Italy, Sweden and France cover a long historical interval, they can be very useful to explore the survival evolution in the long term during the mortality transition.

I.6 Plotting the age-specific death rates

Once a demographic data has been created by means of `hmd.mx`, it is possible to apply a set of specific demographic instructions. By using plot, the logarithms of the mortality rate by age are shown as in the figure 1. Indeed, the ordinate axis is named "Log death rate". Without setting any other option, the graph shows only the female rates, as it is automatically reported in the title.

The first series in the plotted figure refer to the late nineteenth century and their lines are shown in red. These red lines appear on higher levels, over the other ones, as they registered higher mortality. The levels of logarithm lines progressively reduced due to the general improvements of the survival conditions occurred in Italy during the considered period. As the lines are drawn on lower and lower levels, the colours progressively range from red, to yellow, to shades of green and light blue, deep blue, and finally to purple.
In general, all the lines considered together follow the typical "j" shape of the log death risks by age, with higher mortality in the infantile ages, an evident reduction in the young rages and a constant acceleration in the adult ages. It is evident the random variability affecting the log death rates in the oldest age due to the small amounts of the exposures to risk of dying.
It is possible to note one green line that exceeded the mortality level of the first red lines of the mortality rates collection. As a matter of fact, the isolate green line in the graph refers to 1919, an exceptional year of mortality due to Spanish influenza.

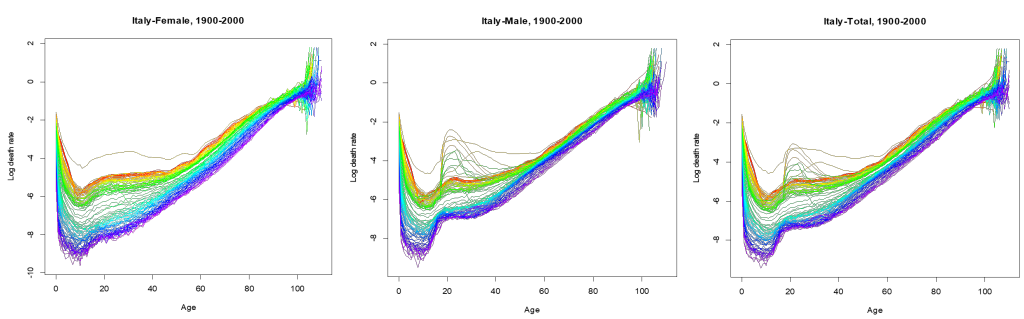
```
plot(italy)
```



It is also possible to show only the age-specific mortality rates for a given time interval and sex. In the following examples, by modifying the parameters for series and year, the log death rates from 1900 to 2000 are plotted respectively for female, male and total population:

```
plot(italy, series=names(italy$rate)[1], year=1900:2000,
   main="Italy-Female,1900-2000")

plot(italy, series=names(italy$rate)[2], year=1900:2000,
   main="Italy-Male,1900-2000")

plot(italy, series=names(italy$rate)[3], year=1900:2000,
   main="Italy-Total,1900-2000")
```

Comparing the following graphs, male log-curves shows a typical mortality humps around 20-30 years. This mortality excess is due to the accidental causes of death more frequently affecting the male population. Evident increase during War World I and II are evident in the male graph as yellow and light green lines show.



I.7 The life expectancy series

As it is possible to download the age-specific death rates from the HMD, `hmd.e0` directly imports the series of the life expectancy at birth (e_0) from a period life table. Again, as main parameters, this command requires country

abbreviation, username and password.

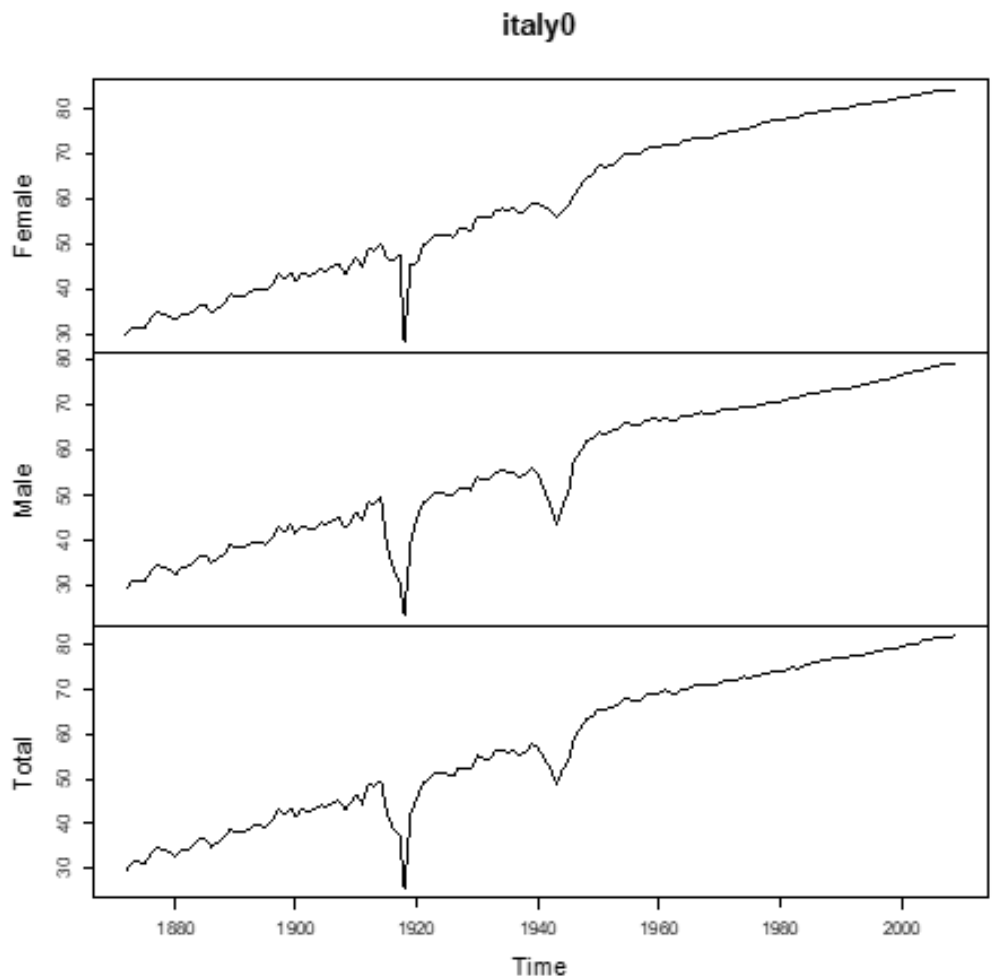
```
italy0 <- hmd.e0("ITA", "username", "password")
```

In the following example, the Italian series of life expectancy is imported for male, female and total population. The italy0 object is a time-series object ranging from 1872 to 2018 with a frequency of one year. By directly typing the italy0 name in the R console starting and ending years are display and the entire series are also shown (the output is interrupted by three dots for sake of brevity).

```
> italy0
Time Series:
Start = 1872
End = 2019
Frequency = 1
Female  Male Total
1872  30.20 29.22 29.69
1873  31.79 31.44 31.60
1874  31.97 31.56 31.75
1875  31.61 31.07 31.31
...
2019  85.41 81.14 83.35
```

The plot command shows the life expectancy series of italy0 in a graph with three panels referring to male, female and total population. In this Italian example, the progressive increase of the life expectancy during the considered period are shown. Two evident decrease in the life expectancy levels occurred during WWI and WWII. The Spanish influenza also caused an evident mortality crisis in 1919. Looking at the y-axes of the graphs, it is possible to assess that the female life expectancy remains constantly on higher level than their male one, showing the advantage of women in terms of survivorship.

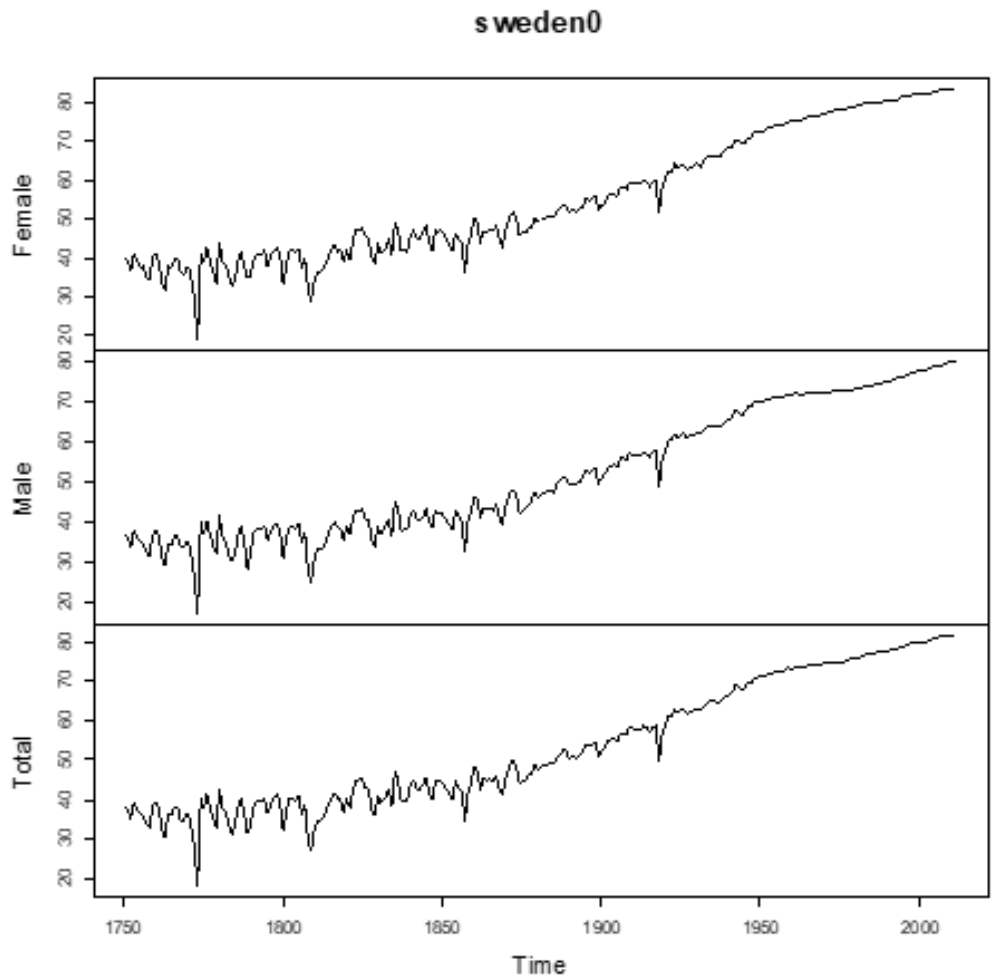
```
plot(italy0)
```



As already said the HMD also provide mortality data for several countries. In the next example, life expectancy series at birth (e_0) for Sweden is imported and shown. In these case, the data cover an historical interval even longer with respect to the Italian one, starting in 1751. No decrease in the Swedish survivorship was observed in the years of World War II as the country remained neutral.

```
sweden0 <- hmd.e0("SWE", "username", "password")
```

```
plot(sweden0)
```



I.8 Life tables from mortality rates

The `lifetable` function allows to compute period and cohort lifetables from mortality rates for multiple years. In the following example, it turns an object containing Italian mortality rates from HMD in a single year life table ranging from 1872 to 2019, respectively for female, male and total population.

The last open age interval is defined after `max.age`. To close the lifetable at a lower age, the first occurrence after min should reduce to values less than 110. By changing the type option, it is also possible to construct period or cohort lifetable.

In the following examples, a set of life tables for all the available years is created resplectively for the female, male and total populations.

Female population:

```
italyf.lt <- lifetable(italy, series = names(italy$rate)
   [1], years = 1872:2019, ages = italy$age, max.age =
   99, type = c("period"))
```

Male population:

```
italym.lt <- lifetable(italy, series = names(italy$rate)
   [2], years = 1872:2019, ages = italy$age, max.age =
   99, type = c("period"))
```

Total population:

```
italyt.lt <- lifetable(italy, series = names(italy$rate)
   [3], years = 1872:2019, ages = italy$age, max.age =
   99, type = c("period"))
```

The following table reports the main arguments of the Lifetable function.

Arguments:

data:	Demogdata object
series:	Name of series to use. Default is the first series in data$rate
years:	Vector indicating which years to include in the tables
ages:	Vector indicating which ages to include in table
max.age:	Age for last row
type:	Type of lifetable: period or cohort

Once you have used the `lifetable` function in R to calculate life tables, you can utilize the `print` function to display the resulting life tables. The `print` function allows you to view the life table information in a formatted and readable manner. By calling the `print` function on the life table object, you can easily access and examine the calculated life table data, including age-specific mortality rates, life expectancy, and other relevant demographic measures. This feature is particularly useful for inspecting and analyzing the life table results generated by the Demography package in R.

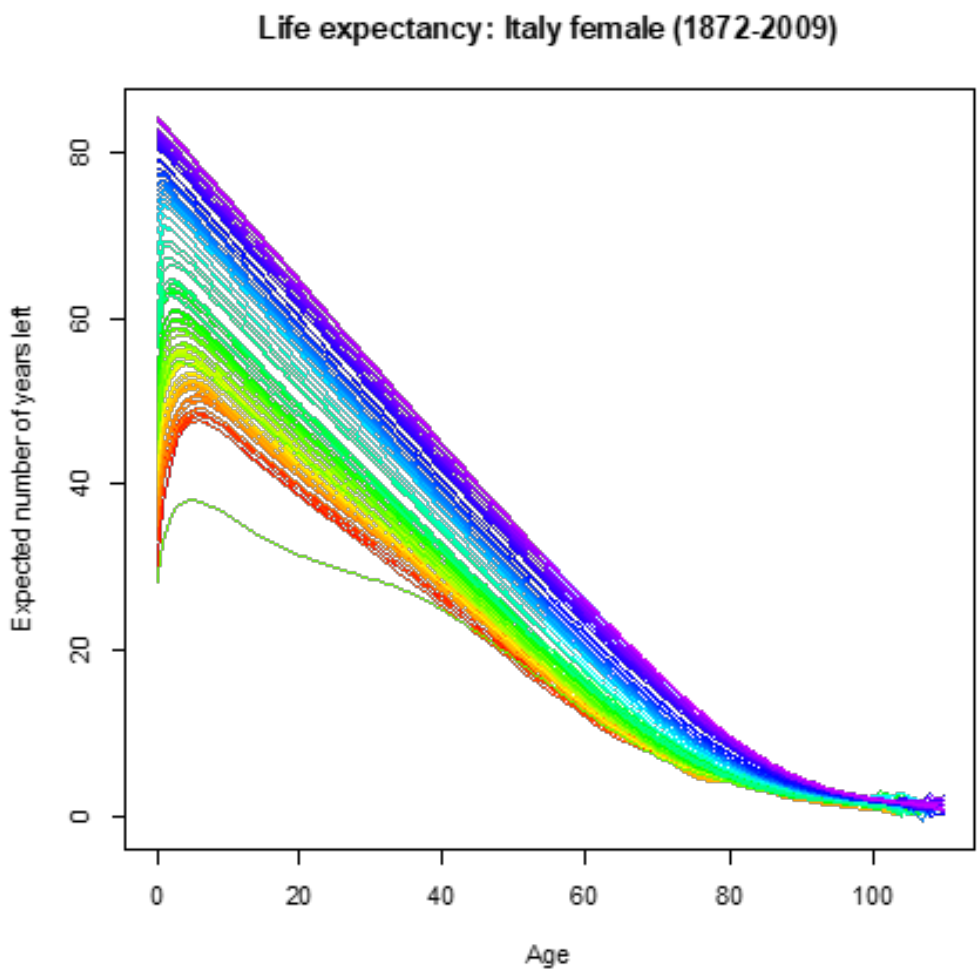
```
lifetable_fem <- print(italyf.lt)

lifetable_mal <- print(italym.lt)

lifetable_tot <- print(italyt.lt)
```

I.9 Plotting e_x curves (Life expectancy at age x)

By combining the `lifetable` and `plot` functions in R, you can easily visualize the survival function, or remaining life expectancy at each age x, of the constructed mortality tables. This can be achieved by passing the life table object obtained from the `lifetable` function as an argument to the `plot` function. This powerful combination allows you to generate insightful plots that represent the biometric function e_x, providing a clear representation of the remaining life expectancy at different ages. Additionally, it is worth

noting that the e_x function will exhibit a descending pattern only for those mortality tables in which infant mortality has significantly decreased.

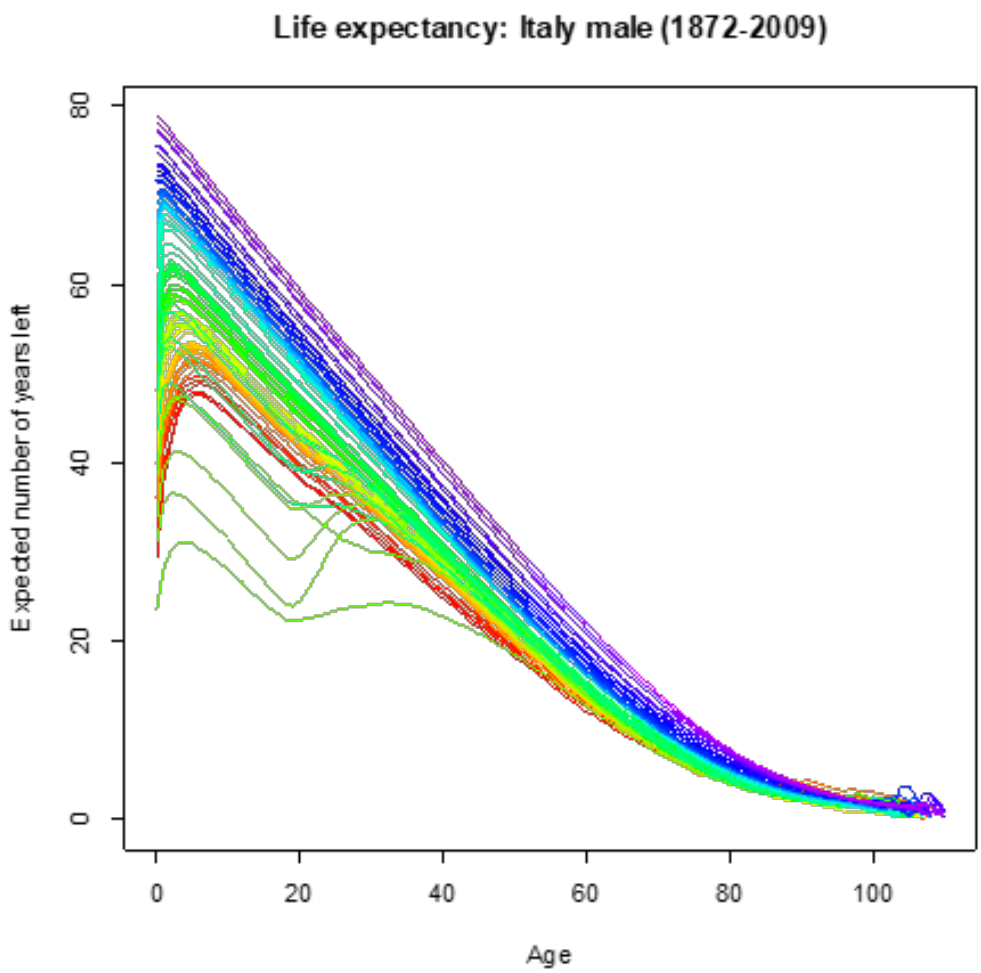
```
plot(italyf.lt)
```



In the graph illustrating life expectancy at age x for females, the green line corresponding to 1919, during the Spanish flu epidemic, is visible. In that year of catastrophic mortality, there was a generalized decrease in life

expectancy at all ages. On the other hand, in the case of males, it is possible to observe a decline in the lines corresponding to the war years.

```
plot(italym.lt)
```



I.10 Plotting l_x curves

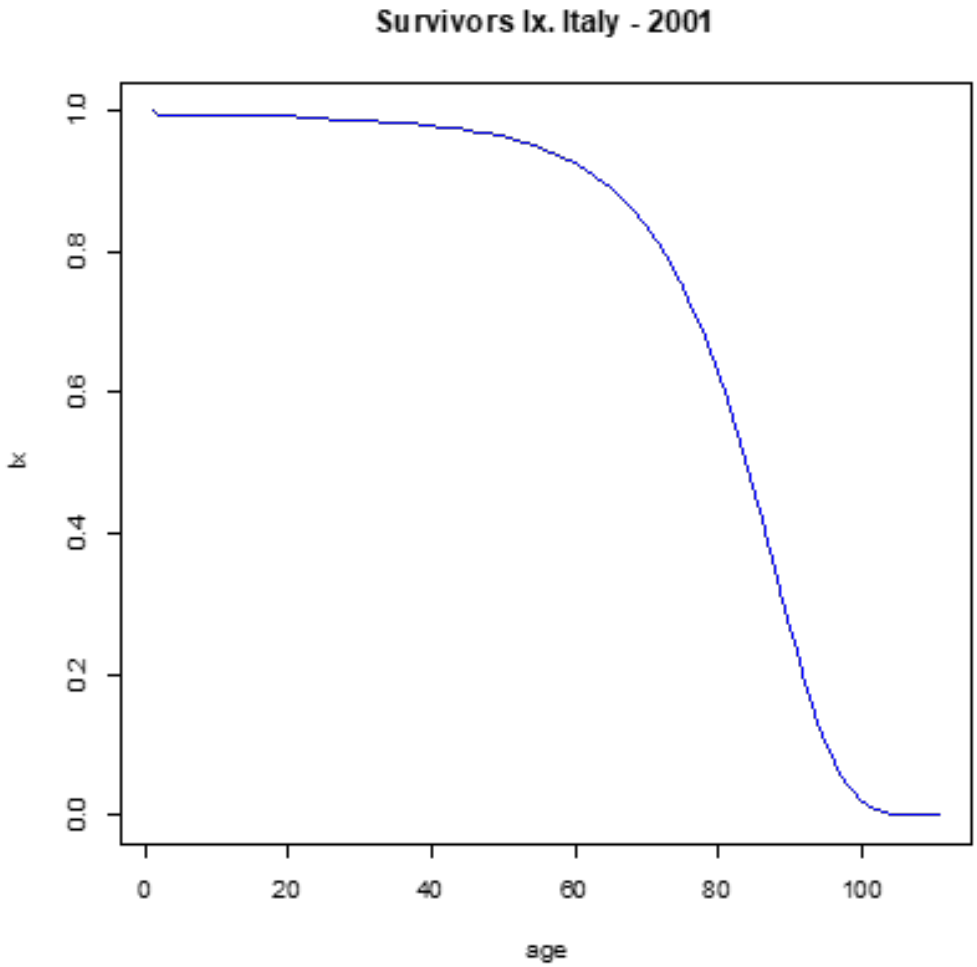
The provided R code calculates the life table for Italy in the year 2001 using the `lifetable` function. It selects the mortality rate series from the input

data, specifies the desired years and age range, and sets the type of life table as "period". The resulting life table is stored in `italyt_2001.lt`. The code then extracts the survivors (`lx`) from the life table and assigns them to the vector `lx_2001`. Finally, it plots the survivors against age using a blue solid line (`lty=1`) and adds appropriate labels and a title to the plot.

Different types of plots can be drawn. Possible type options are:

- "p" for points,
- "l" for lines,
- "o" for both 'overplotted',
- "h" for 'histogram' like vertical lines,
- "s" for stair steps,

```
italyt_2001.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = 2001, ages = italy$
   age, max.age = 99, type = c("period"))

lx_2001 <-italyt_2001.lt$lx

plot(lx_2001, col="blue", xlab="age", ylab="lx",
   type="l", lty=1, main = "Survivors lx. Italy -
    2001")
```



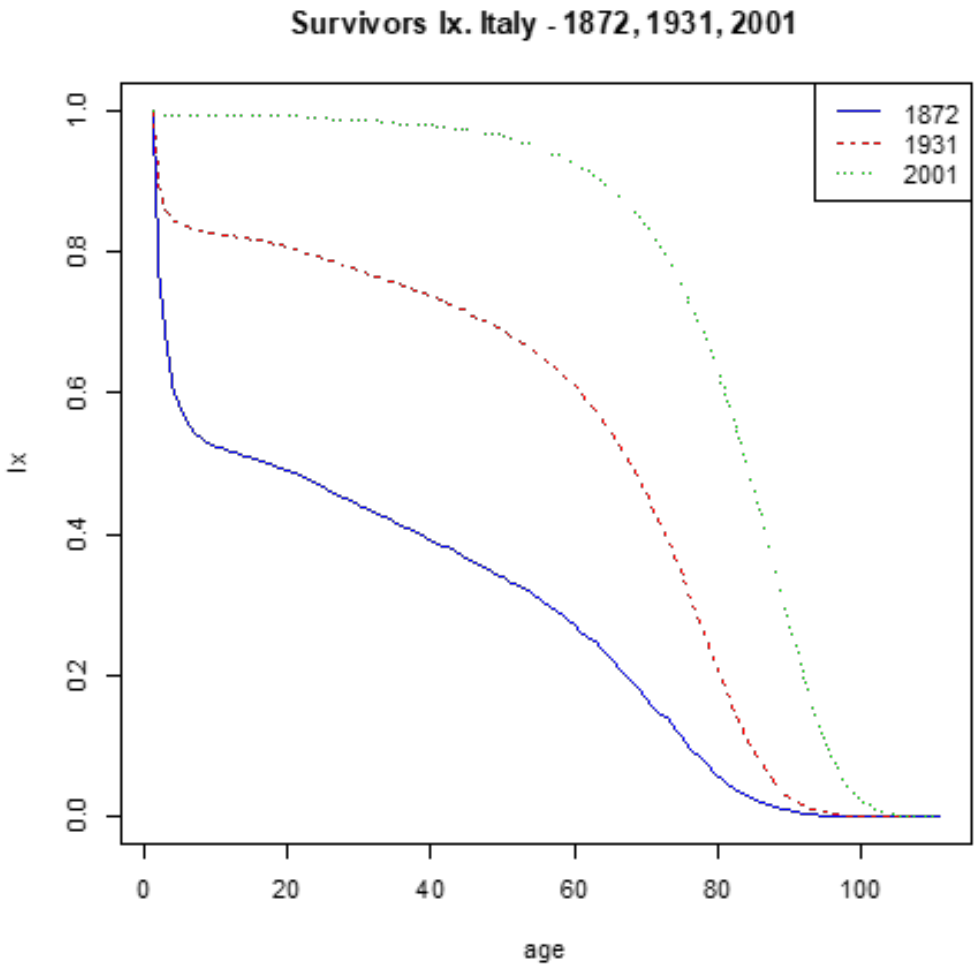
I.11 Plotting l_x curves at 1872, 1931 and 2001

The following R code is used to illustrate the phenomenon of rectangularization of survivorship curves that occurred in Italy due to a drastic reduction in mortality over the examined years. The code calculates the life tables for Italy in the years 1872, 1931, and 2001 using the `lifetable` function. It then extracts the survivors (`lx`) from each life table and assigns them to respective vectors.

```
italyt_1872.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = italy$year[1], ages =
    italy$age, max.age = 99, type = c("period"))
lx_1872 <-italyt_1872.lt$lx

italyt_1931.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = italy$year[60], ages
   = italy$age, max.age = 99, max(italy$age)),
   type = c("period"))
lx_1931 <-italyt_1931.lt$lx

italyt_2001.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = italy$year[130], ages
    = italy$age, max.age = 99, max(italy$age)),
   type = c("period"))
lx_2001 <-italyt_2001.lt$lx
```

The code plots the survivorship curves using different line colours and line types. The blue line represents the survivorship curve for 1872, the red line represents 1931, and the black line represents 2001. Finally, a legend is added to the plot in the bottom left corner, providing labels for the years and corresponding line colours.

```
plot(lx_1872, col="blue", xlab="age", ylab="lx",
   type="l", lty=1, main = "Survivors lx. Italy -
    1872, 1931, 2001")
lines(lx_1931, col="red", lty=2)
lines(lx_2001, col="green", lty=3)
legend("topright",c("1872", "1931", "2001"),col=c
   ("blue","red", "green"), lty=1:3)
```

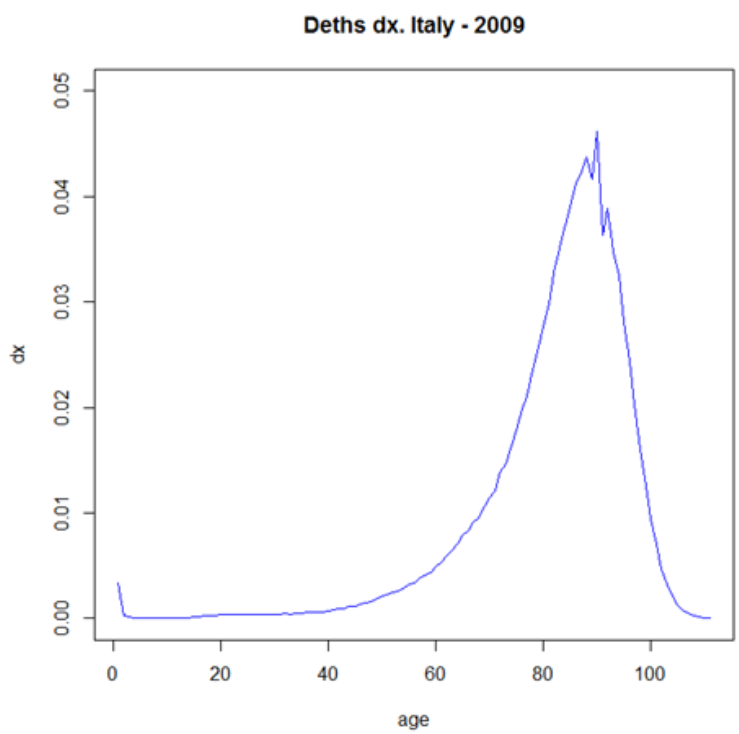


I.12 Plotting d_x curves

The following code plots the deaths (`dx`) column from the Italian life table of 2009. It first calculates the life table using the `lifetable` function. Then, it extracts the deaths (`dx`) column and assigns it to the vector (`dx_2009`). The `plot` function is then used to create a line plot of the Deaths (`dx`) column with the specified y-axis limits, using a blue line. The x-axis is labeled as "Age" and the y-axis as "Deaths (`dx`)". The title of the plot is set as "Deaths

dx. Italy - 2009".

```
italyt_2009.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = 2009, ages = italy$
   age, max.age = 110, type = c("period"))

dx_2009 <-italyt_2009.lt$dx

plot(dx_2009, ylim=c(0, 0.05), col="blue", xlab="
   age", ylab="dx", type="l", lty=1, main = "
   Deths dx. Italy - 2009")
```

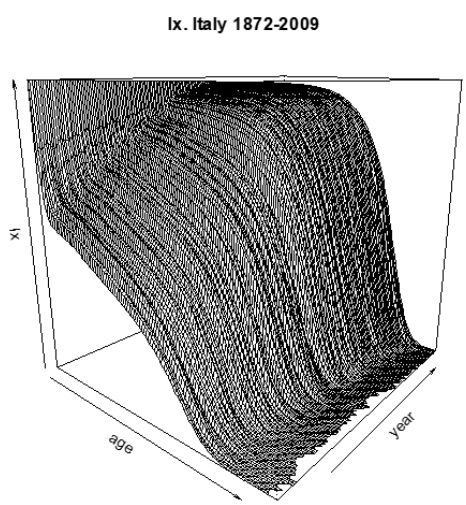


I.13 Plotting 3D mortality surfaces

It is possible to represent the functions of a life table as surfaces in a 3D graph that considers the axes of age and historical time. The provided R

code generates a 3D plot of the survival curves using the `persp` function. The age values are stored in the `age` vector, and the year values are stored in the `year` vector. The `italyt.lt$lx` represents the survivors (`lx`) column. By specifying `age`, `year`, and `italyt.lt$lx` as the x, y, and z coordinates respectively, the `persp` function creates a 3D surface plot. The `theta` parameter sets the angle of the plot, and the `zlab` parameter labels the z-axis as "lx". The `main` parameter adds a title to the plot, indicating it as "lx. Italy 1872-2009".

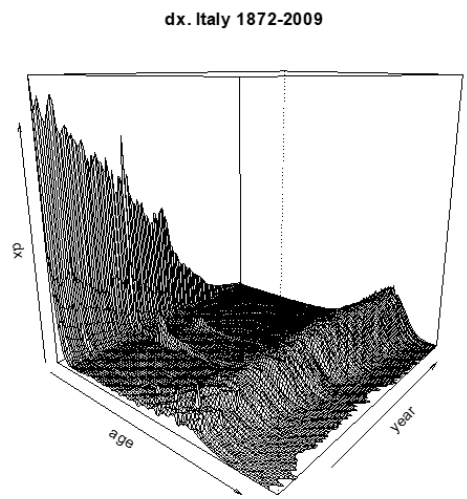
```
age <- italyt.lt$age

year <- italyt.lt$year

persp(age,year,italyt.lt$lx, theta = 40, zlab = "
   lx", main = "lx. Italy 1872-2009")
```



The additional line of code provided allows us to construct a 3D plot for the Deaths (dx) curves. By using the `persp` function and specifying the `age`, `year`, and `italyt.lt$dx` as the x, y, and z coordinates respectively, we can

create a 3D surface plot. Notably, this plot showcases a transfer of deaths from infancy to older adult ages, resulting in a shift in the modal age at death.

```
persp(age,year,italyt.lt$dx, theta = 40, zlab = "
   dx", main = "dx. Italy 1872-2009")
```



I.14 Life expectancy at e_0

We can use the `life.expectancy` function to extract the life expectancy at birth for females, males, and the total population from the mortality data. By setting the `series` option to `names(italy$rate)[1]`, `names(italy$rate)[2]`, and `names(italy$rate)[3]` respectively, we can calculate the life expectancy for each group.

For female (1)

```
e0_f <- life.expectancy(italy, series = names(
   italy$rate)[1],  years = 1872:2019, type = c("
```

```
            period"))
        e0_f

Time Series:
Start = 1872
End = 2009
Frequency = 1
1872     1873     1874     1875     1876
30.20327 31.79222 31.97075 31.60758 33.94496
....
```

For male (2)

```
        e0_m <- life.expectancy(italy, series = names(
            italy$rate)[2],  years = 1872:2019, type = c("
            period"))
```

For total population (3)

```
        e0_t <- life.expectancy(italy, series = names(
            italy$rate)[3],  years = 1872:2019, type = c("
            period"))
```

I.15 Evolution of life expectancy at 0, 60 e 80

Furthermore, it is possible to specify the `age` option in the `life.expectancy` command to obtain life expectancy not only at birth but also at specific ages. For example, the following code calculates the life expectancy at ages 60 and 80:

```
        e_0 <- life.expectancy(italy, series = names(
            italy$rate)[3],  years = 1872:2009, type = c("
            period"))

        e_60 <- life.expectancy(italy, series = names(
            italy$rate)[3],  years = 1872:2009, type = c("
            period"), age=60)
```

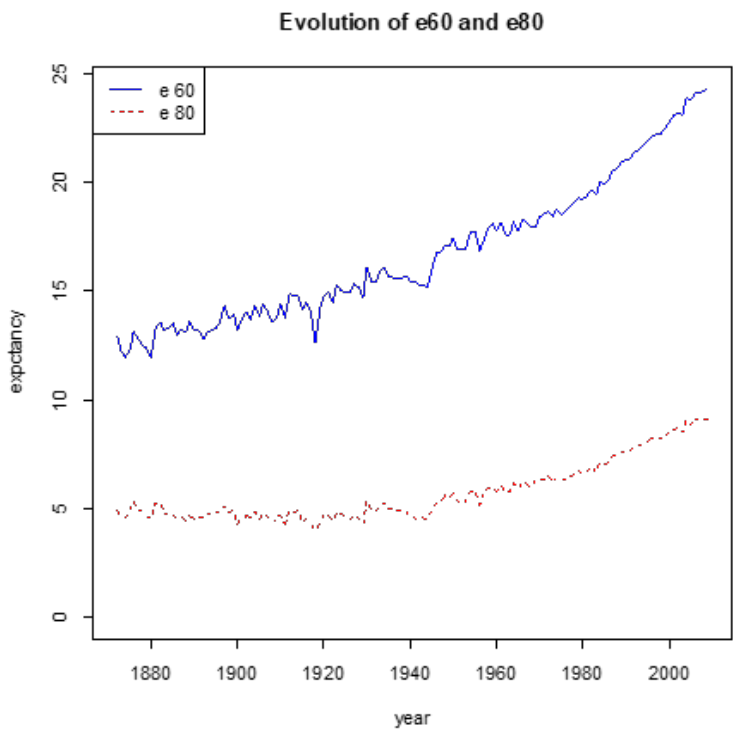
```
e_80 <- life.expectancy(italy, series = names(
   italy$rate)[3],  years = 1872:2009, type = c("
   period"), age=80)
```

By including specific age values, we can analyse life expectancy at different ages. The figure below illustrates the series of life expectancy at age 80 (`e80`) for Italy over an extended period, ranging from the late 1800s to the early 2000s.

```
> e_80
Time Series:
Start = 1872
End = 2009
Frequency = 1
    1872     1873     1874     1875     1876     1877     1878     1879     1880     1881     1882     1883     1884
4.934047 4.683522 4.566781 4.959424 5.317258 4.943171 4.846586 4.625030 4.511352 5.289949 5.261403 4.808902 4.776482
    1885     1886     1887     1888     1889     1890     1891     1892     1893     1894     1895     1896     1897
4.754067 4.571725 4.612600 4.483966 4.709793 4.533718 4.582369 4.583624 4.726800 4.835951 4.831086 4.901472 5.104242
    1898     1899     1900     1901     1902     1903     1904     1905     1906     1907     1908     1909     1910
4.823453 4.925790 4.332362 4.562840 4.681957 4.567316 4.888684 4.527875 4.810896 4.643633 4.397411 4.478055 4.739009
    1911     1912     1913     1914     1915     1916     1917     1918     1919     1920     1921     1922     1923
4.321012 4.934728 4.858839 4.896434 4.366893 4.564213 4.309872 3.998494 4.376384 4.716177 4.686588 4.408357 4.838599
    1924     1925     1926     1927     1928     1929     1930     1931     1932     1933     1934     1935     1936
4.711929 4.603216 4.545320 4.695251 4.581588 4.325699 5.406819 4.883357 4.900566 5.132590 5.271829 4.992035 4.964633
    1937     1938     1939     1940     1941     1942     1943     1944     1945     1946     1947     1948     1949
4.922630 4.887412 4.911342 4.654454 4.585228 4.523711 4.578785 4.497132 4.891487 5.301804 5.253014 5.594445 5.483341
    1950     1951     1952     1953     1954     1955     1956     1957     1958     1959     1960     1961     1962
5.715794 5.309326 5.309431 5.331673 5.772758 5.847504 5.190637 5.663463 5.931932 6.093944 5.799213 6.073740 5.733556
    1963     1964     1965     1966     1967     1968     1969     1970     1971     1972     1973     1974     1975
5.783519 6.198574 5.930651 6.241874 6.162887 5.953114 6.336530 6.338374 6.346399 6.510507 6.199690 6.437487 6.292536
    1976     1977     1978     1979     1980     1981     1982     1983     1984     1985     1986     1987     1988
6.396396 6.471217 6.629444 6.702958 6.647554 6.718460 6.956329 6.677432 7.115204 7.047262 7.156651 7.430654 7.437972
    1989     1990     1991     1992     1993     1994     1995     1996     1997     1998     1999     2000     2001
7.656108 7.629189 7.663259 7.890404 7.921083 8.017785 8.139038 8.277517 8.273656 8.233013 8.373458 8.504407 8.668905
    2002     2003     2004     2005     2006     2007     2008     2009
8.701066 8.476930 9.090291 8.886016 9.110017 9.060909 9.055276 9.130091
```

As previously mentioned, it is possible to plot multiple vectors representing life expectancy at different ages on the same graph using the `plot` command. This allows us to visualize the evolution of life expectancy at two particularly significant ages, such as 60 and 80. The graph provides insight into the changes in life expectancy over time at these specific ages.

```
plot(e_60, col="blue", ylim=c(0, 25), xlab="year"
   , ylab="expctancy", main = "Evolution of e60
   and e80")
lines(e_80, col="red", lty=2)
legend("topleft",c("e 60", "e 80"),col=c("blue","
   red"), lty=1:2)
```


Evolution of e60 and e80
e 60
e 80
expctancy
year
0
5
10
15
20
25
1880
1900
1920
1940
1960
1980
2000

Part II

Other Useful Functions and Scripts

II.1 Extract some ages from a demogdata object

The `extra.ages` command allows us to extract specific age groups from the mortality data. In this case, it is used to extract the age group 0-3 from the `italy` dataset.

The provided code demonstrates the usage of `extra.ages` by extracting the age group 0-3 from the `italy` dataset and assigning it to the `italy_0_3` object. `combine.upper` argument in the `extract.ages` command determines whether to combine the upper age group boundary with the next lower age group or not. When `combine.upper` is set to `FALSE`, each age group is treated separately and not combined with the next lower age group. This means that the upper age boundary of each specified age group will be exclusive, and the resulting extracted age groups will have distinct ranges.

The `plot` function is then used to create a line plot of the log mortality rates (`ln(mx)`) for the age group 0-3 in Italy for the year 1872. The plot is shown in blue color with a solid line.

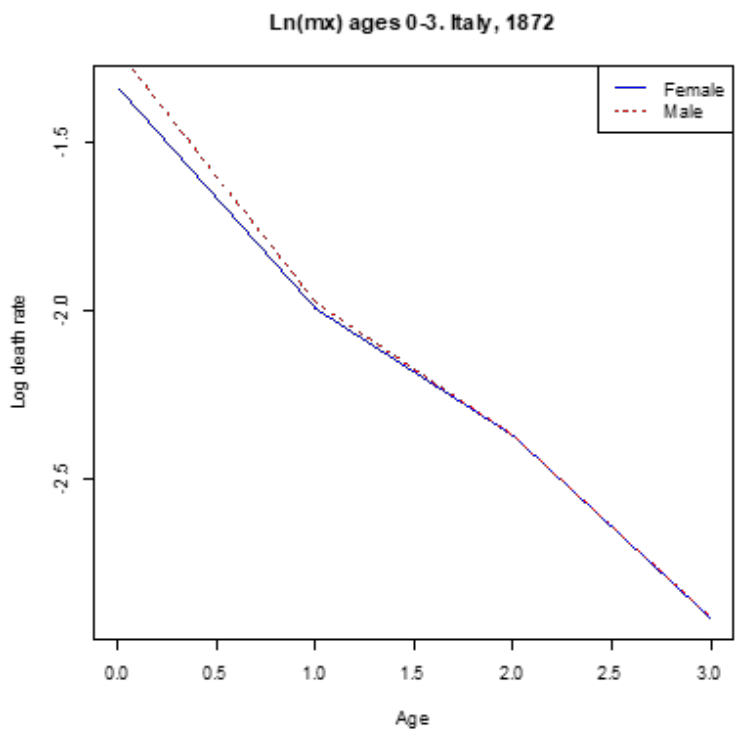
Furthermore, the `lines` function overlays another line on the plot, representing the mortality rates for the same age group but for the opposite gender (in this case, males). The line is shown in red color with a dashed line.

Finally, a legend is added to the plot in the top-right corner, indicating the color and line type for each gender.

This example is important as it illustrates how to extract specific age groups from the mortality data and visualize them separately. By plotting the mortality rates for different age groups and genders, we can gain insights into the variations and trends in mortality patterns across different subpopulations.

```
italy_0_3 <- extract.ages(italy, 0:3, combine.
   upper=FALSE)
plot(italy_0_3, series=names(italy$rate)[1], year
   =1872, type="l", col="blue", lty=1, main="Ln(
   mx) ages 0-3. Italy, 1872")
lines(italy_0_3, series=names(italy$rate)[2],
   year=1872, type="l", col="red", lty=2,)
```

```
legend("topright",c("Female", "Male"),col=c("blue
    ","red"), lty=1:2)
```



II.2 Analyzing fertility data

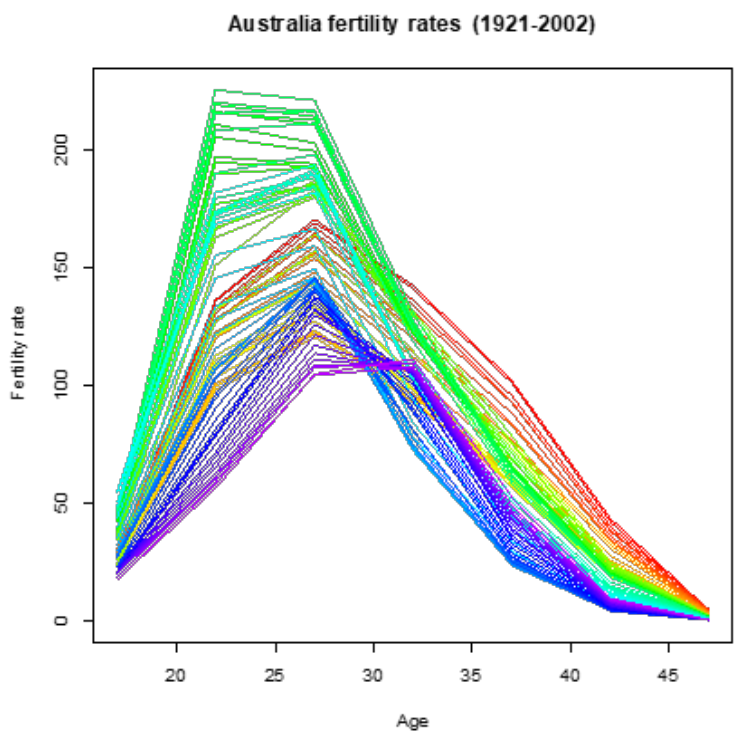
Similarly to what we have seen for mortality data, the Demography package also provides functionality to handle fertility and migration rates. Within the package, there are already included fertility data specifically related to Australia, which can be plotted. Additionally, the `tfr` function allows us to calculate the total fertility rate and visualize its temporal trend.

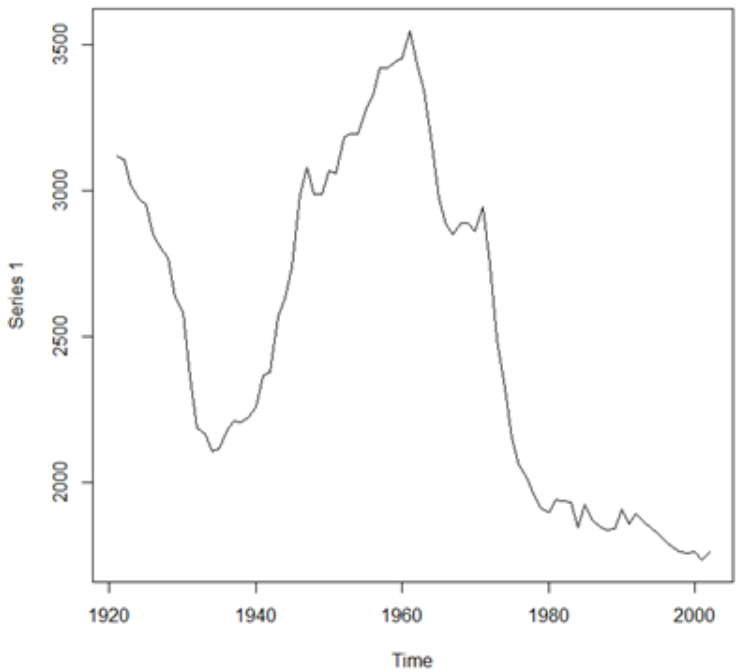
The provided code demonstrates the usage of these features. First, the fertility data for Australia, stored in the `aus.fert` object, is plotted. This provides an initial visualization of the fertility rates. Next, the `tfr` function is applied to the `aus.fert` object, calculating the total fertility rate and storing it in the `tfr` variable. The resulting value of the total fertility rate

is then displayed. Finally, the `plot` function is used again to generate a plot specifically for the total fertility rate.

This code showcases the capability of the Demography package to handle fertility data, allowing users to analyze and explore fertility rates, including the calculation and visualization of the total fertility rate.

```
aus.fert
plot(aus.fert)
tfr <- tfr(aus.fert)
tfr
plot(tfr(aus.fert))
```

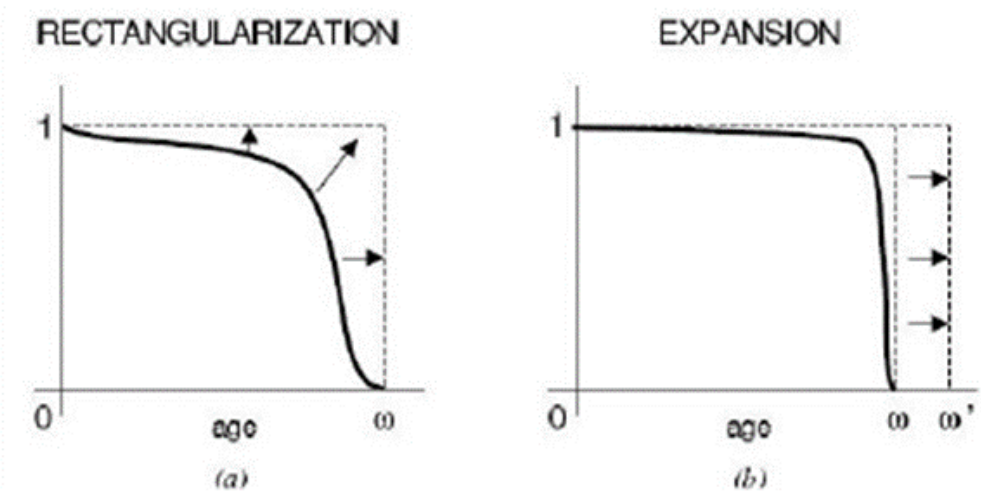




II.3 Rectangularization

The phenomenon of "rectangularization" and "expansion" of survival curves (`lx`) in life tables refers to the trend of decreasing mortality rates over time. As mortality levels decline, the survival proportion becomes more similar across different age groups, resulting in a rectangular shape when plotted against age. This indicates a compression of mortality towards older ages and an expansion of the survival curve.

In the following sections, we will introduce various measures and visualization techniques to represent and analyze this phenomenon. These measures and visualizations provide insights into the changing patterns of mortality and life expectancy, allowing us to examine the trends, inequalities, and potential implications for population health and aging.



II.4 H index for rectangularization

The rectangularization index is a measure that quantifies the degree of rectangularization in a mortality pattern. It provides an indication of how closely the survival curves (`lx`) resemble a rectangular shape.

The formula for the rectangularization index is as follows:

$$H = -\frac{\sum_0^{\omega}(log \quad lx)lx}{\sum_0^{\omega} lx} \qquad 0 < H < 1$$

A lower rectangularization index indicates a greater degree of rectangularization.

The rectangularization index is a useful tool for measuring and comparing the degree of rectangularization across different populations or time periods. It provides insights into changes in mortality patterns and can help identify trends in population health and aging.

$H = 0$ if l_x is rectangular, $H = 1$ if it is not.

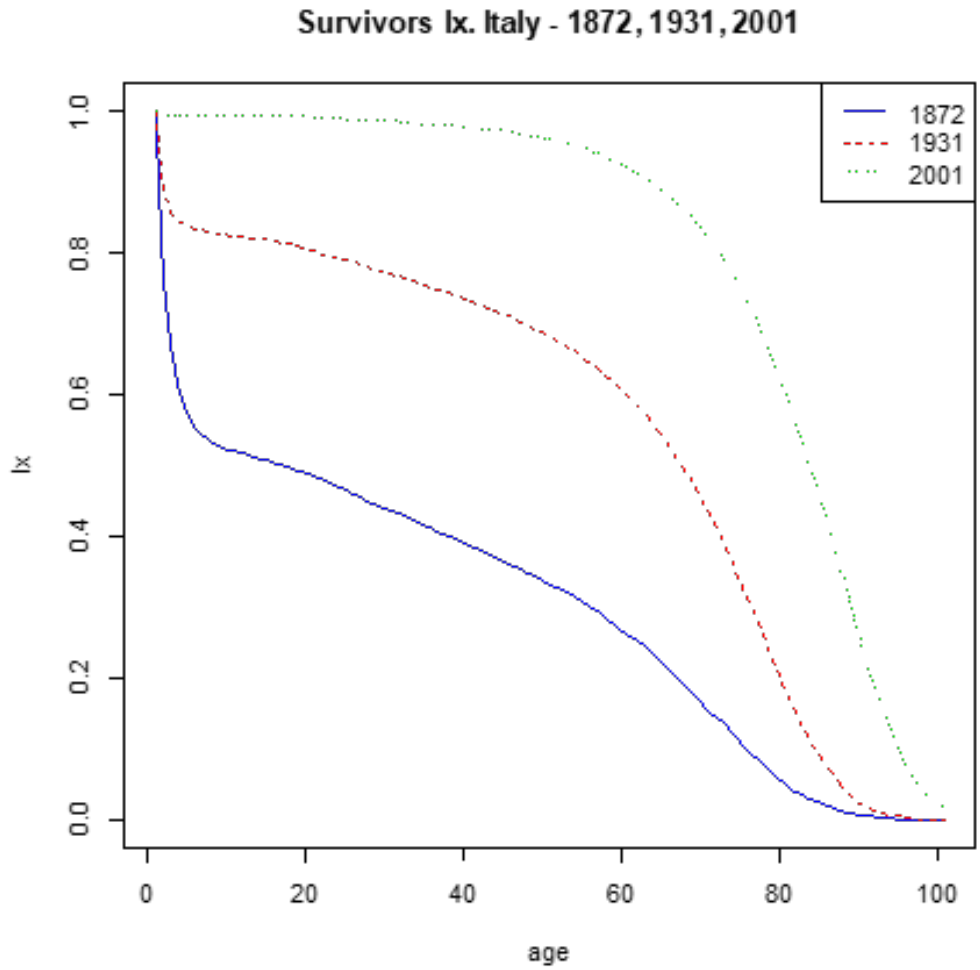
II.5 H index in only 3 year (1872, 1931 and 2001)

This paragraph presents an R code that first visually compares the survival curves for the Italian life tables of 1872, 1931, and 2001. Subsequently, the code calculates and displays the recently mentioned H index for these three years.

```
italyt_1872.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = 1872, ages = italy$
   age, max.age = min(100, max(italy$age)), type
   = c("period"))
lx_1872 <-italyt_1872.lt$lx

italyt_1931.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = 1931, ages = italy$
   age, max.age = min(100, max(italy$age)), type
   = c("period"))
lx_1931 <-italyt_1931.lt$lx

italyt_2001.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = 2011, ages = italy$
   age, max.age = min(100, max(italy$age)), type
   = c("period"))
lx_2001 <-italyt_2001.lt$lx

plot(lx_1872, col="blue", xlab="age", ylab="lx",
   type="l", lty=1, main = "Survivors lx. Italy -
    1872, 1931, 2001")
lines(lx_1931, col="red", lty=2)
lines(lx_2001, col="green", lty=3)
legend("topright",c("1872", "1931", "2001"),col=c
   ("blue","red", "green"), lty=1:3)

#H index for 1872
ln_lx_1872 <- log(lx_1872)
ln_lx_1872
product_1872 <- ln_lx_1872 * lx_1872
```

```
product_1872
sum_product_1872 <- sum(product_1872)
sum_product_1872
sum_dem_1872 <- sum(lx_1872)
sum_dem_1872
H_1872 <- - (sum_product_1872 / sum_dem_1872)
H_1872

# H index for 1931
ln_lx_1931 <- log(lx_1931)
ln_lx_1931
product_1931 <- ln_lx_1931 * lx_1931
product_1931
sum_product_1931 <- sum(product_1931)
sum_product_1931
sum_dem_1931 <- sum(lx_1931)
sum_dem_1931
H_1931 <- - (sum_product_1931 / sum_dem_1931)
H_1931

#H index for 2001
ln_lx_2001 <- log(lx_2001)
ln_lx_2001
product_2001 <- ln_lx_2001 * lx_2001
product_2001
sum_product_2001 <- sum(product_2001)
sum_product_2001
sum_dem_2001 <- sum(lx_2001)
sum_dem_2001
H_2001 <- - (sum_product_2001 / sum_dem_2001)
H_2001
H_3 <- c(H_1872, H_1931, H_2001)
H_3

Anni <- c("1872", "1931", "2001")
H_3 <- c(H_1872, H_1931, H_2001)
table <- rbind (Anni, H_3)
table

[,1]                [,2]                [,3]
```

```
Anni "1872"                "1931"                "
   2001"
H_3  "0.902573294369772" "0.387468668894437" "
   0.127667717632917"
```



Calculating an H rectangularization index for each individual year is not incorrect, but it can be laborious and prone to errors. Therefore, it is necessary to consider a new programming strategy to streamline the code, making it more concise and less repetitive.

II.6 *H* index from 1872 to 2009. Italy

We can calculate the H index for each year in the available life table dataset by simply setting up a for loop using a loop variable to iterate through each year and update the H index calculation accordingly.

In general, a for loop is a programming construct that allows you to repeatedly execute a block of code for a specified number of iterations. It is commonly used when you need to perform a certain operation multiple times or iterate through a collection of elements.

In the context of R programming, a for loop follows a specific syntax. It typically consists of three parts: initialization, condition, and iteration. The initialization part sets the initial value of a loop variable. The condition part defines the condition that must be true for the loop to continue executing. The iteration part specifies how the loop variable should be updated after each iteration.

During each iteration of the loop, the code within the loop block is executed. This allows you to perform specific operations or calculations repeatedly until the specified condition becomes false.

Using a for loop in R, you can easily iterate over a sequence of values, such as a vector or a range of numbers. By utilizing the loop variable, you can perform actions or computations that depend on the current iteration. This provides a convenient way to automate repetitive tasks and process data efficiently.

The following R code calculates the Rectangularization index (H index) for Italy from 1872 to 2009.

- First, a vector H is created with 138 elements, all initialized to 1, representing the index values.
- *Anno* is a vector that stores the years from 1872 to 2009.
- A for loop is used to iterate over the index values from 1 to 138.
- Within each iteration:
  - The `lifetable` function is applied to calculate the life table for the corresponding year using the mortality data from the "italy" dataset.

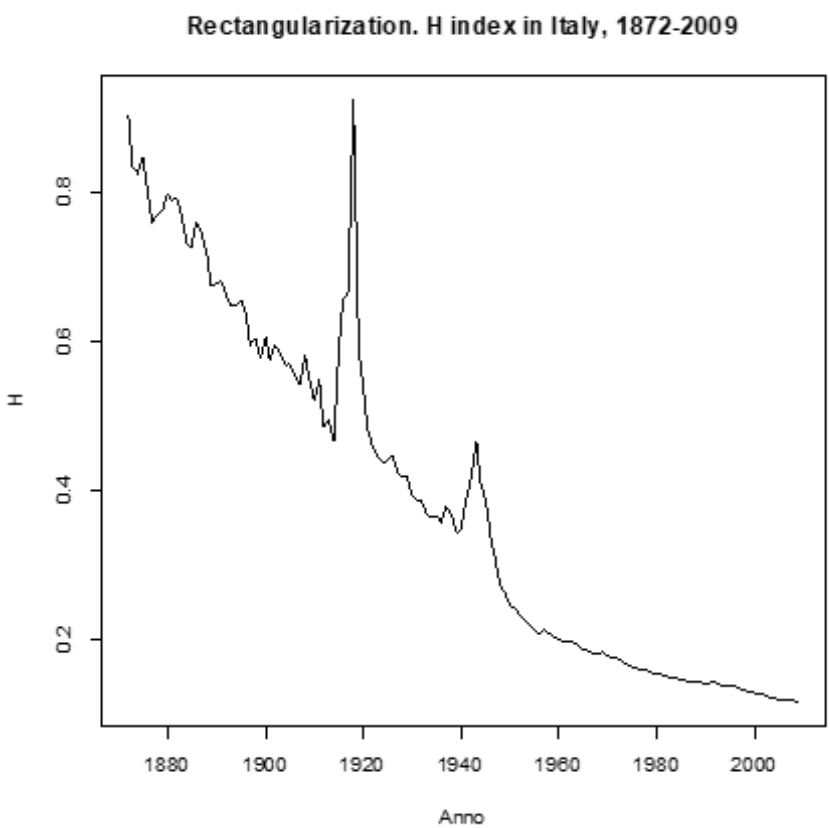
- lx_{year} is extracted from the life table, representing the survivors at each age.
- $ln_{lx_{\text{year}}}$ is calculated as the natural logarithm of lx_{year}.
- $product_{\text{year}}$ is obtained by multiplying $ln_{lx_{\text{year}}}$ with lx_{year}.
- The sum of $product_{\text{year}}$ is calculated and stored in $sum_{\text{product}_{\text{year}}}$.
- $sum_{\text{dem}_{\text{year}}}$ is calculated as the sum of lx_{year}.
- H_{year} is computed as $-(sum_{\text{product}_{\text{year}}}/sum_{\text{dem}_{\text{year}}})$, representing the Rectangularization index for the specific year.
- H_{year} is assigned to the corresponding element in the H vector.

- Finally, the H vector and the corresponding years are combined into the *serie* matrix.

- The `plot` function is used to create a line plot, displaying the H index values over the years, with the title "Rectangularization: H index in Italy, 1872-2009".

```
H <- rep(1,138)
Anno <- seq(1872,2009,1)
for (index in 1:138) {
        italyt_year.lt <- lifetable(italy, series
            = names(italy$rate)[3], years = italy
           $year[index], ages = italy$age, max.
           age = 100, type = c("period"))
        lx_year <-italyt_year.lt$lx
        ln_lx_year <- log(lx_year)
        ln_lx_year
        product_year <- ln_lx_year * lx_year
        product_year
        sum_product_year <- sum(product_year)
        sum_product_year
        sum_dem_year <- sum(lx_year)
        sum_dem_year
        H_year <- - (sum_product_year / sum_dem_
           year)
        H_year
```

```
        H[index] = H_year
        }

H

serie <- cbind(H, Anno)

serie

plot(Anno, H, type = "l", main="
   Rectangularization. H index in Italy,
   1872-2009")
```



II.7 Plotting multiple survival curves using a For Loop

The `for` loop can be useful not only for calculating formulas but also for other repetitive actions to be performed automatically in R. One possible application is in constructing a graph where not just three survival curves, as seen in previous examples, but many more are extracted and plotted. Additionally, a legend can be included to indicate the various years being considered. This allows for the visualization of a larger dataset with multiple survival curves, providing a comprehensive representation of the data over different periods.

```
lx_matrix <- matrix(data = seq(1,15318,1), nrow
   =111, ncol=138)
for (index in 1:138) {
        lx_matrix[,index] <- ((lifetable(italy,
           years = italy$year[index], ages =
           italy$age, max.age = 110, type = c("
           period")))$lx)
        }
```

The provided R code initializes a matrix called `lx_matrix` with dimensions 111 rows and 138 columns. The values of the matrix are initially set as a sequence from 1 to 15,318.

A for loop is then used to iterate over the index values from 1 to 138. Within each iteration, the `lifetable` function is applied to calculate the life table for the corresponding year using the mortality data from the "italy" dataset. The resulting survivors (`lx`) values are extracted from the life table and assigned to the corresponding column of the `lx_matrix`.

The final output is the `lx_matrix`, which contains the survivors (`lx`) values for each year across different age groups. This code efficiently generates the matrix by automatically populating the data using the `lifetable` function, providing a convenient way to store and access mortality information over multiple years and age groups.

```
years <- seq(1872,2009,5)
plot(lx_matrix[,1], xlab="age", ylab="lx", type="
   l", lty=1)
```

```
for (index in seq(1, 138, 5)) {
        lines(lx_matrix[,index], col=rainbow(138)
           [index], lty=1)
        }
legend("bottomright", legend=years, col=rainbow
   (24), lty=1, cex = 0.9, bty ="n")
```

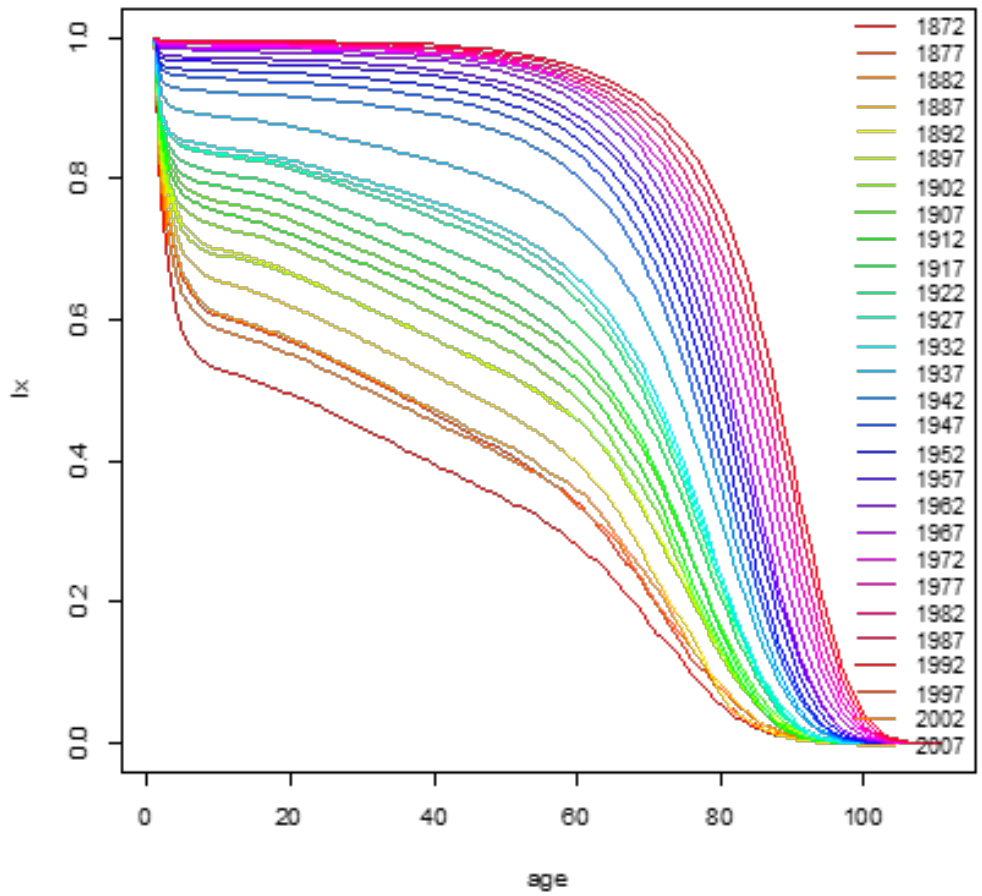
The given R code generates a line plot of survivorship (`lx`) using the `plot` function and a `for` loop.

- First, the initial survivorship values from the `lx_matrix` are plotted for the first column using `plot`. The x-axis is labeled as "age" and the y-axis as "lx". The plot is displayed with a solid line.

- The `for` loop is used to iterate over the index values from 1 to 138 with a step size of 5. Within each iteration, the `lines` function is called to overlay additional lines on the plot. These lines correspond to the survivorship values from the `lx_matrix` at the specified index, with different colors based on the `rainbow` palette.

- The legend is added to the bottom-right corner of the plot, displaying the years in the `years` vector. The colors of the legend entries are chosen from the `rainbow` palette, and a solid line type is used. The `cex` argument adjusts the size of the legend text, and `bty` removes the legend box.

This code allows for the comparison of survivorship curves at specific intervals, showcasing the changes in survivorship over time. The legend provides a clear representation of the corresponding years for a group of lines in the plot.



II.8 Comparison of male and female survival: temporal advantage of women in Italy

The code compares male and female survival in Italy for specific years. It calculates life expectancy and survivorship for males in 2009 and females in 1986. The results are plotted, showing survivorship curves for both genders. The graph is interesting as it demonstrates the temporal advantage of women

over men in terms of survival. In Italy, men reached the same life expectancy that women had in 1986 only in 2009.

```
# Italy Male 2009
italy_M_2009.lt <- lifetable(italy, series =
   names(italy$rate)[2], years = 2009, ages =
   italy$age, max.age = min(110, max(italy$age)),
    type = c("period"))
lx_M_2009 <-italy_M_2009.lt$lx
e_0_M_2009 <- life.expectancy(italy, series =
   names(italy$rate)[2], years = 2009, type = c("
   period"))
e_0_M_2009

# Italy Female 1985
italy_F_1986.lt <- lifetable(italy, series =
   names(italy$rate)[1], years = 1986, ages =
   italy$age, max.age = min(110, max(italy$age)),
    type = c("period"))
lx_F_1986 <-italy_F_1986.lt$lx
e_0_F_1986 <- life.expectancy(italy, series =
   names(italy$rate)[1], years = 1986, type = c("
   period"))
e_0_F_1986

confronto_Italia <- matrix(c(e_0_F_1986, e_0_M_
   2009), nrow = 1, ncol = 2, byrow = TRUE,
   dimnames = list(c("e0"), c("Female 1986", "
   Male 2009")))
plot(lx_F_1986, col="red", xlab="age", ylab="lx",
    type="l", lty=1, main = "Survivors lx by Sex.
    Italy, Males 2009 and Females 1986")
lines(lx_M_2009, col="blue", lty=2)
legend("topright",c("Female", "Male"),col=c("red"
   ,"blue"), lty=1:2)
```

The provided R code compares the survival of males and females in Italy, highlighting the temporal advantage of women.

- **Italy Male 2009**
  The code calculates the life table and life expectancy for males in Italy

in 2009. It extracts the survivorship values (`lx`)and life expectancy at birth (`ex`) for males in that year.

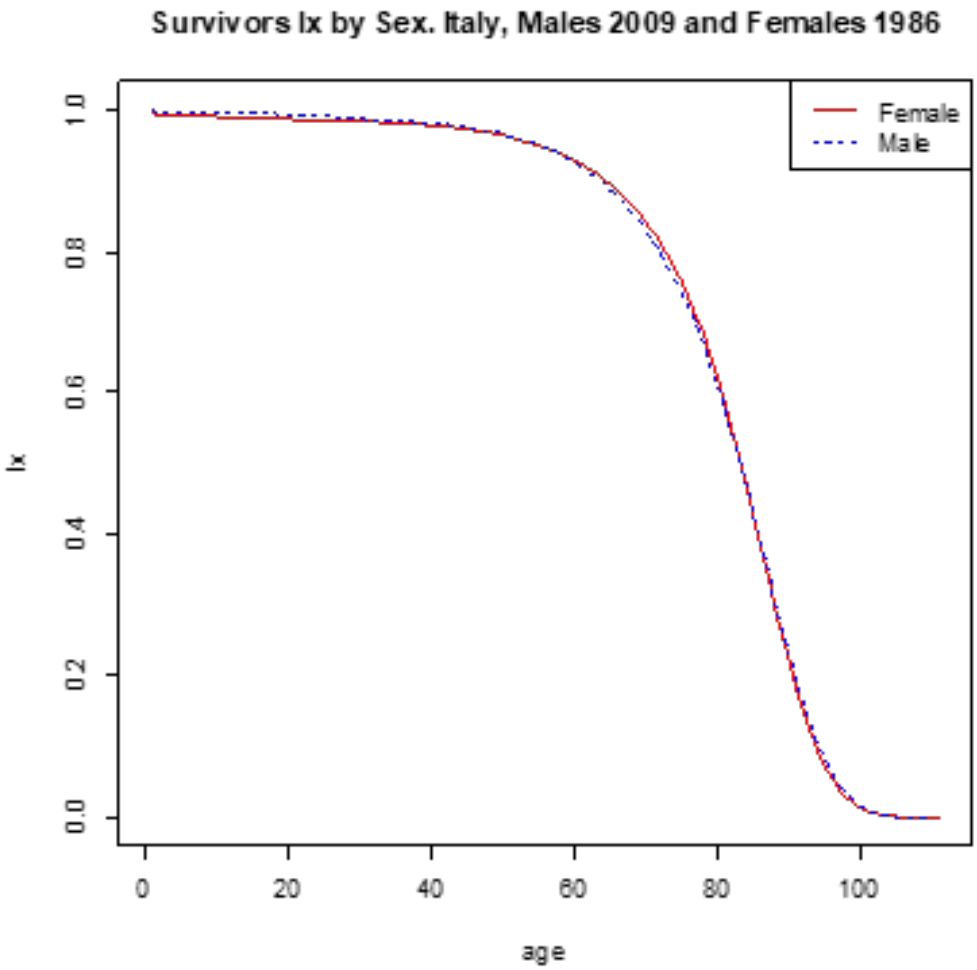
- **Italy Female 1986**
  The code calculates the life table and life expectancy for females in Italy in 1986. It extracts the survivorship values and life expectancy for females in that year.

- **Comparison Matrix**
  A matrix, (`confronto_Italia`), is created to compare the life expectancies of females in 1986 and males in 2009. The matrix has one row and two columns, representing the life expectancies of females and males.

- **Line Plot**
  The code generates a line plot to visualize the survivorship of females in 1986 (represented by a red line) and males in 2009 (represented by a blue line). The x-axis represents age, and the y-axis represents survivorship. The plot is titled "Survivors lx by Sex. Italy, Males 2009 and Females 1986."

- **Legend**
  A legend is added to the top-right corner of the plot, indicating the colors and line types for females and males. The legend includes the labels "Female" and "Male" with red and blue colors, respectively.

```
> confronto_Italia
Female 1986 Male 2009
e0    79.13512  79.22049
```



II.9 Comparing probability of death in reproductive ages. Italy 1881

The following R code aims to compare the probability of death between men and women during the reproductive ages in Italy in 1881. By examining mortality rates within these ages, we can gain insights into potential gender disparities in health and mortality within the reproductive age group. The

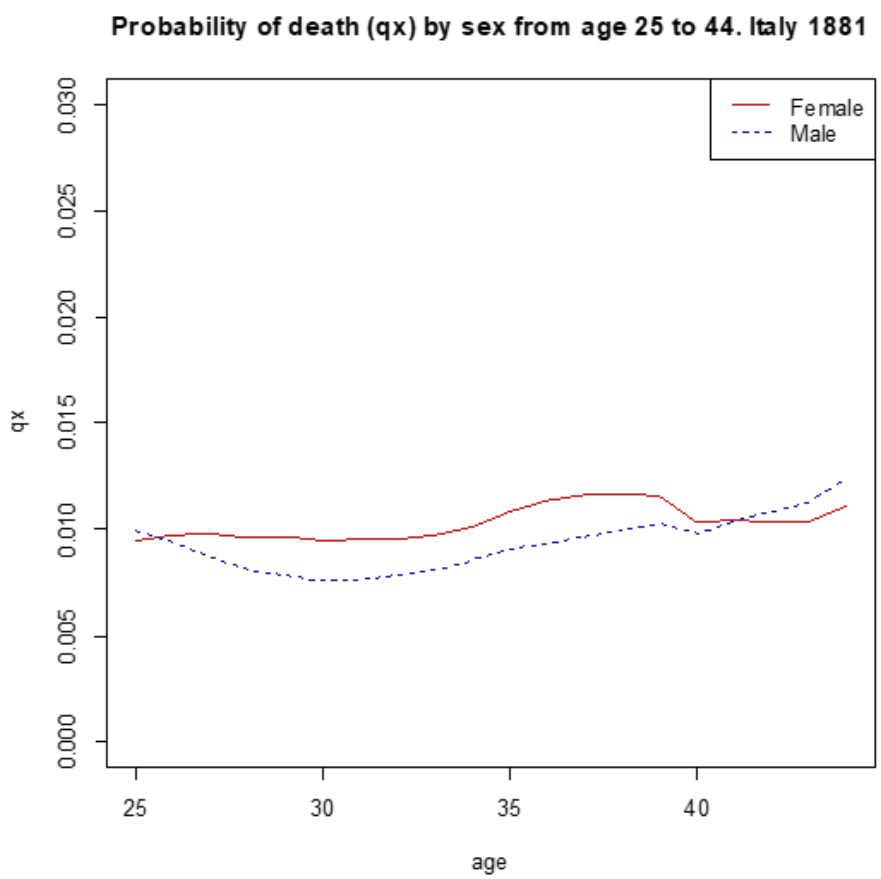
comparison between genders provides a deeper understanding of the mortality risks due to maternal causes of death.

This R code compares the probability of death between men and women in the reproductive ages (25 to 44) in Italy in the year 1881.

```
italy_F_1881.lt <- lifetable(italy, series =
   names(italy$rate)[1], years = 1881, ages =
   italy$age, max.age = 100, type = c("period"))

italy_M_1881.lt <- lifetable(italy, series =
   names(italy$rate)[2], years = 1881, ages =
   italy$age, max.age = 100, type = c("period"))

qx_F_1881_15_44 <- italy_F_1881.lt$qx[25:44]

qx_M_1881_15_44 <- italy_M_1881.lt$qx[25:44]

age <- seq(25,44)

plot( age, qx_F_1881_15_44, ylim=c(0, 0.03), col=
   "red", xlab="age", ylab="qx", type="l", lty=1,
    main = "Probability of death (qx) by sex from
    age 25 to 44. Italy 1881")

lines( age, qx_M_1881_15_44, col="blue", lty=2)

legend("topright",c("Female", "Male"),col=c("red"
   ,"blue"), lty=1:2)
```

- First, the `lifetable` function is used to calculate the life table for females and males in Italy in 1881, considering the mortality rates from the *italy* dataset.
- The `qx` values (probability of death) for females and males in the age range of 25 to 44 are extracted from the respective life tables.

- A sequence of ages from 25 to 44 is created.
- The `plot` function is then used to generate a line plot, showing the `qx` values for females (red line) and males (blue dashed line) on the y-axis against the corresponding ages on the x-axis. The plot is labeled with appropriate axis labels and a title.
- Finally, a legend is added to the top-right corner of the plot, indicating the colors and line types associated with females and males.



II.10 Computing sex ratios from mortality rates

The `sex.ratio` command is a function in the R package *demography* that calculates the male-to-female ratios from historical mortality rates. This information can be valuable for studying gender disparities and examining the impact of mortality on gender dynamics. The `sex.ratio` command is a useful tool for researchers and demographers working with mortality data to explore gender-related phenomena.

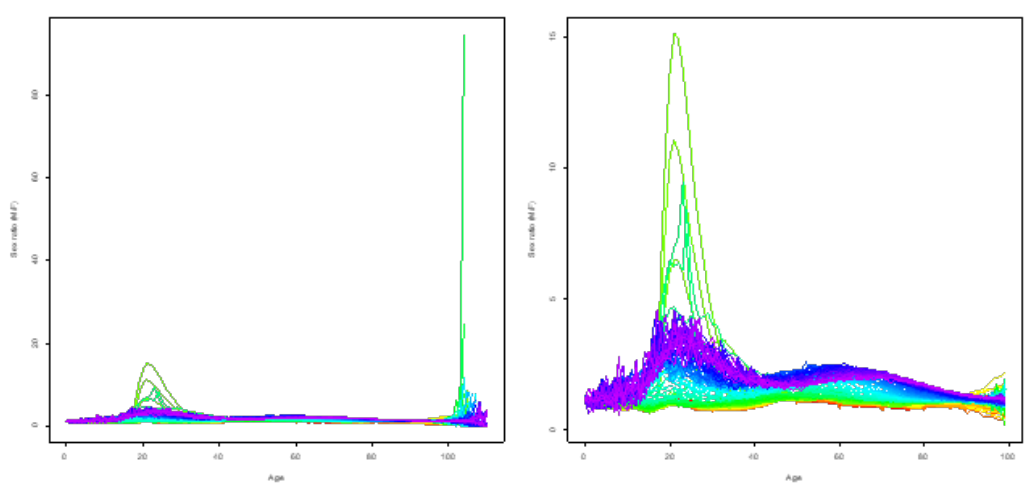
```
par(mfrow=c(1,2))
```

We can calculate the male/female ratios from historical mortality rates in all the age group.

```
plot(sex.ratio(italy))
```

However it is better to focus on a more limited interval of ages, to avoid random fluctuations in the oldest ages due to small numbers.

```
plot(sex.ratio(extract.ages(italy, 0:99, combine.
   upper=FALSE)))
```



II.11 Smoothing Demographic Data in R

Smoothing splines are a powerful tool used to smooth and regularize rates or functions in mortality tables, addressing issues related to small numbers and random fluctuations. They provide a flexible and intuitive approach to handling demographic data. A spline can be considered a smooth curve constructed by connecting several smaller curve segments, called basis functions. These basis functions are typically polynomials of a certain degree, such as cubic splines.

Cubic splines are especially common in demographic analysis. They consist of piecewise cubic functions joined at specific points, called knots, ensuring smoothness and continuity. In addition, smoothing splines also require the definition of a smoothing parameter. This parameter controls the trade-off between the goodness of fit and the smoothness of the resulting curve.

Analysts can effectively smooth and regularise demographic data by employing smoothing splines and tuning the smoothing parameter, reducing noise and highlighting underlying patterns and trends.

II.12 `smooth.spline`

The `smooth.spline` function is part of the R package *demography* and provides a convenient way to apply smoothing splines to demographic data. The `smooth.spline` function fits a smoothing spline to the given data, estimating a smooth curve that best represents the underlying pattern. It takes into account the local variations in the data while ensuring overall smoothness. The function allows users to specify the degree of smoothness through a smoothing parameter `spar`.

By utilizing `smooth.spline`, demographers can effectively handle demographic data by smoothing out noise and emphasizing underlying trends. This facilitates the analysis and interpretation of demographic patterns and provides a valuable tool for studying population dynamics.

The following code performs smoothing of demographic data in R.

First, it calculates the lifetable for Italy in the year 1900 using the mortality rates from the "italy" dataset. The mortality data is specifically extracted

using the series name corresponding to the third column of the dataset (total population).

Next, the code creates a smoothing spline for the age-specific mortality rates (`dx_1900`) using the `smooth.spline` function. Four different smoothing splines are generated with varying degrees of smoothness. The `spar` parameter is adjusted to control the level of smoothness. The values used in this code are 0, 0.8, 1, and 2 for `spar`.

The resulting smoothing splines (`dx_1900_0.spl`, `dx_1900_03.spl`, `dx_1900_1.spl`, `dx_1900_6.spl`) represent smoothed curves that capture the underlying patterns in the age-specific mortality rates. These smoothed curves can be used for further analysis and visualization, to examine and interpret demographic trends with reduced noise and fluctuations.

```
italyt_1900.lt <- lifetable(italy, series = names
   (italy$rate)[3], years = 2009)
dx_1900 <-italyt_1900.lt$dx
dx_1900_0.spl <- smooth.spline(dx_1900, spar=0)
dx_1900_03.spl <- smooth.spline(dx_1900, spar
   =0.8)
dx_1900_1.spl <- smooth.spline(dx_1900, spar=1)
dx_1900_6.spl <- smooth.spline(dx_1900, spar=2)
```

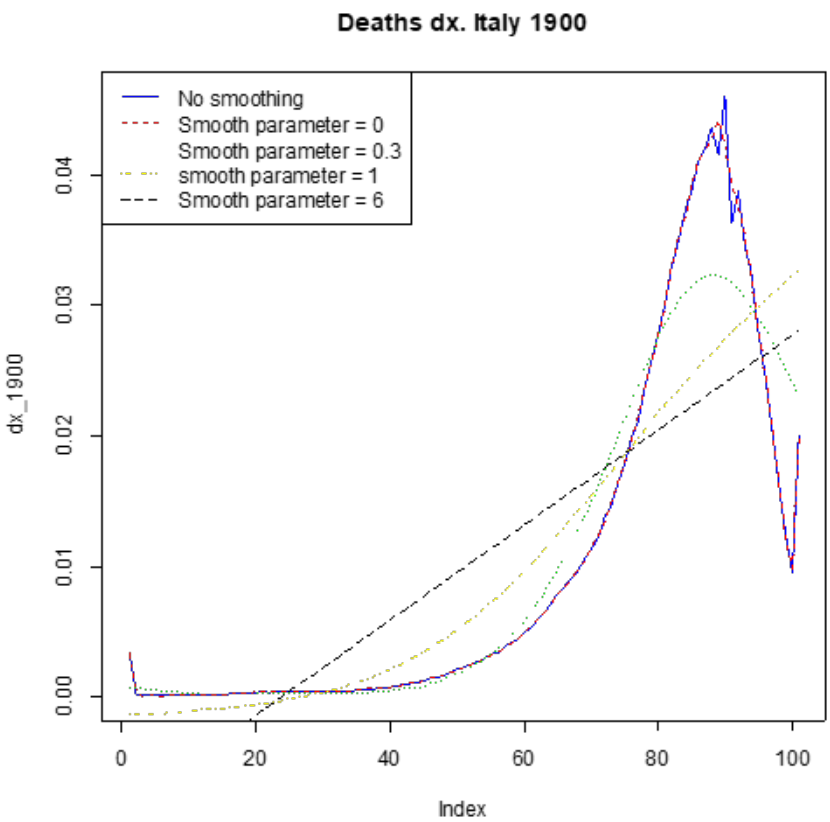
Spar is the smoothing parameter:

```
spar = 0 lowest smoothing
spar = 2 higher smoothing
```

In the following graph, we compare the different splines and the deaths from the 1900 life table.

```
plot(dx_1900, type="l", col="blue", lty=1, main =
    "Deaths dx. Italy 1900")
lines(dx_1900_0.spl, col="red", lty=2)
lines(dx_1900_03.spl, col="green", lty=3)
lines(dx_1900_1.spl, col="yellow", lty=4)
lines(dx_1900_6.spl, col="black", lty=5)
```

```
legend("topleft",c("No smoothing", "Smooth
   parameter = 0", "Smooth parameter = 0.3", "
   smooth parameter = 1", "Smooth parameter = 6")
   , col=c("blue", "red", "green", "yellow", "
   black"), lty=1:5)
```



Part III

Importing External Demographic Data

III.1 Importing data from text files

The `read.demogdata` function in the R package *demography* allows for the import of mortality rates and exposures to risk data from the Human Mortality Database (HMD) directly into R. While we previously learned how to load the data directly from the HMD website using an internet connection, it is also possible to download the mortality rates and exposures to risk data from the HMD website to the local hard drive and then import this data into the R memory.

By utilizing the `read.demogdata` function, demographers have the flexibility to work with the mortality data offline, without the need for an internet connection. This approach enables the construction of a demographic object in R, which can be used to construct life tables and set up the graphs we have seen thus far.

This process allows for efficient data management and analysis within R, as researchers can work with the local copy of the data, ensuring data availability even without an internet connection. This functionality provides demographers with greater flexibility in their data exploration and enables the creation of comprehensive demographic analyses using the powerful tools available in the *demography* package.

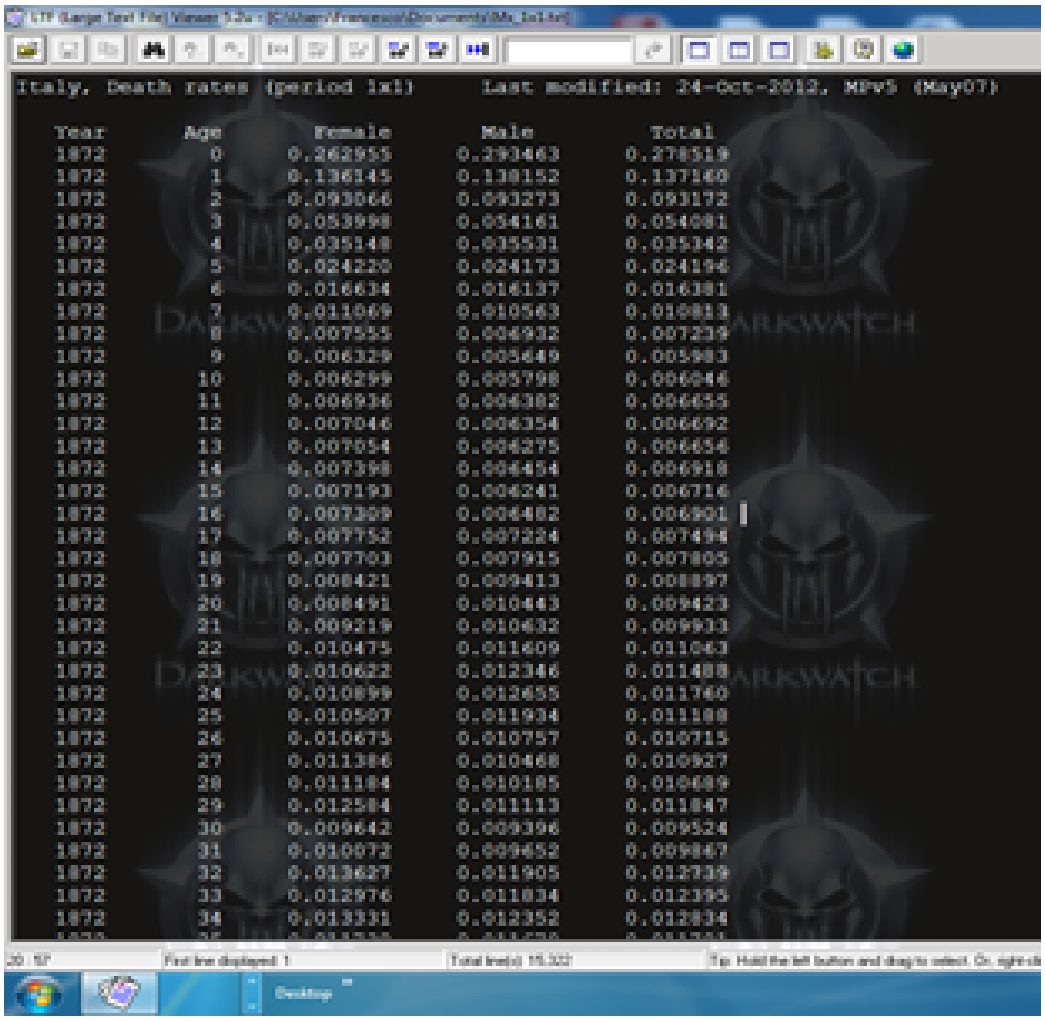
The following line of R code reads demographic data from two files, `Mx_1x1.txt` and `Exposures_1x1.txt`, and assigns it to the object `italy2`. The `Mx_1x1.txt` and `Exposures_1x1.txt` files have been downloaded from the Human Mortality Database (HMD) website and saved locally on the disk. The data being read is of type "mortality" and corresponds to the label "Italy". The `skip=2` argument indicates that the first two rows of the data files should be skipped during the import process.

In summary, this line of code imports mortality data and exposures to risk data for Italy from the specified files and stores it in the `italy2` variable for further analysis and manipulation.

```
italy2 <- read.demogdata("Mx_1x1.txt", "Exposures
   _1x1.txt", type="mortality", label="Italy",
   skip=2)
```

Here, you can see the content of the two `Mx_1x1.txt` and `Exposures_1x1.txt` files.

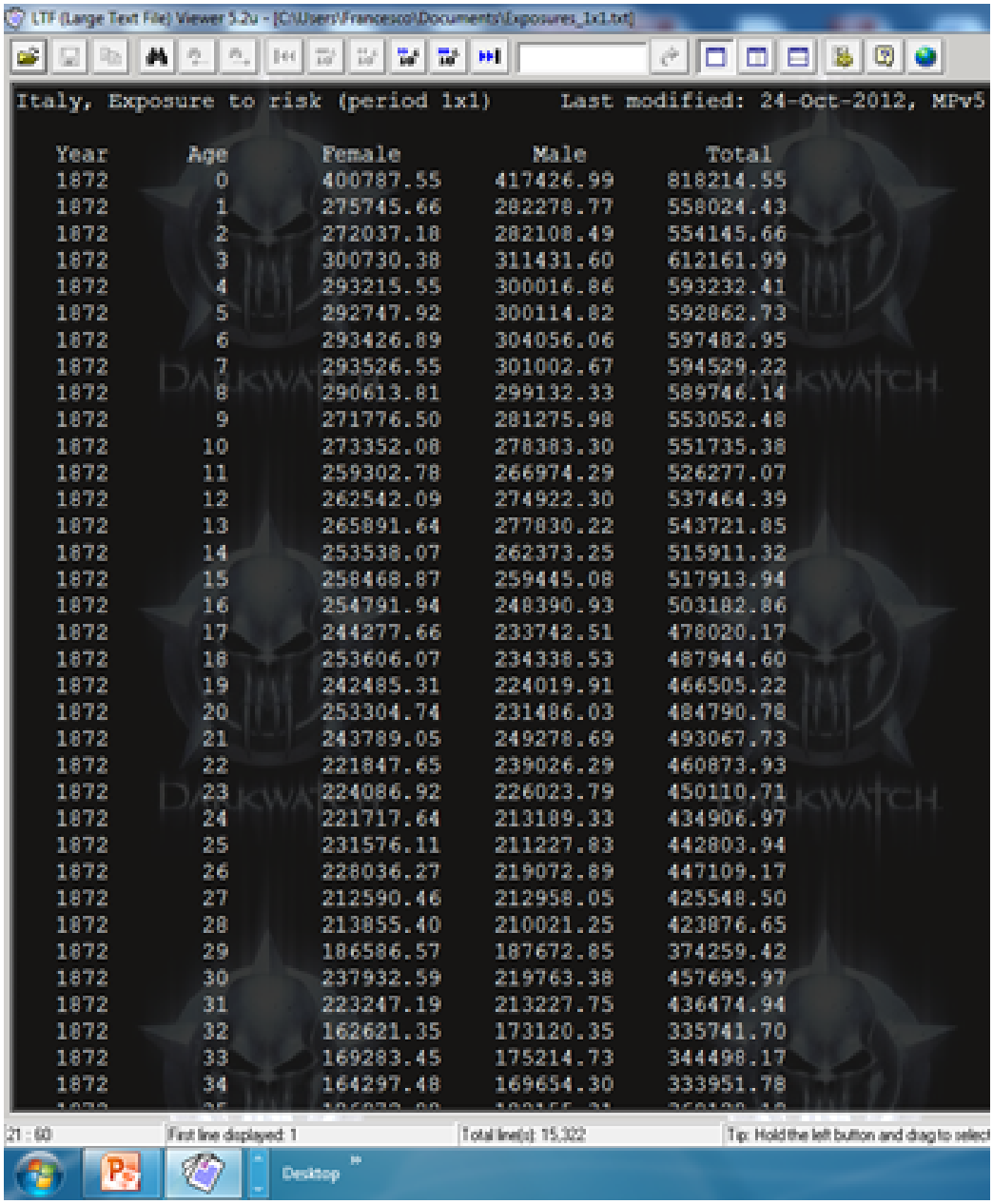
"Mx_1x1.txt"



Italy, Death rates (period 1x1) Last modified: 24-Oct-2012, MPv5 (May07)

Year	Age	Female	Male	Total
1872	0	0.262955	0.293463	0.278519
1872	1	0.136145	0.138152	0.137160
1872	2	0.093066	0.093273	0.093172
1872	3	0.053998	0.054161	0.054081
1872	4	0.035148	0.035531	0.035342
1872	5	0.024220	0.024173	0.024196
1872	6	0.016634	0.016137	0.016381
1872	7	0.011069	0.010563	0.010813
1872	8	0.007555	0.006932	0.007239
1872	9	0.006329	0.005649	0.005983
1872	10	0.006299	0.005798	0.006046
1872	11	0.006936	0.006382	0.006655
1872	12	0.007046	0.006354	0.006692
1872	13	0.007054	0.006275	0.006654
1872	14	0.007398	0.006454	0.006918
1872	15	0.007193	0.006241	0.006716
1872	16	0.007309	0.006482	0.006901
1872	17	0.007752	0.007224	0.007494
1872	18	0.007703	0.007915	0.007805
1872	19	0.008421	0.009413	0.008897
1872	20	0.008491	0.010443	0.009423
1872	21	0.009219	0.010632	0.009933
1872	22	0.010475	0.011609	0.011063
1872	23	0.010622	0.012346	0.011488
1872	24	0.010899	0.012655	0.011760
1872	25	0.010507	0.011934	0.011188
1872	26	0.010675	0.010757	0.010715
1872	27	0.011386	0.010468	0.010927
1872	28	0.011184	0.010185	0.010689
1872	29	0.012584	0.011113	0.011847
1872	30	0.009642	0.009396	0.009524
1872	31	0.010072	0.009652	0.009867
1872	32	0.013627	0.011905	0.012739
1872	33	0.012976	0.011834	0.012395
1872	34	0.013331	0.012352	0.012834

"Exposures_1x1.txt"



Italy, Exposure to risk (period 1x1) Last modified: 24-Oct-2012, MPv5

Year	Age	Female	Male	Total
1872	0	400787.55	417426.99	818214.55
1872	1	275745.66	282278.77	558024.43
1872	2	272037.18	282108.49	554145.66
1872	3	300730.38	311431.60	612161.99
1872	4	293215.55	300016.86	593232.41
1872	5	292747.92	300114.82	592862.73
1872	6	293426.89	304056.06	597482.95
1872	7	293526.55	301002.67	594529.22
1872	8	290613.81	299132.33	589746.14
1872	9	271776.50	281275.98	553052.48
1872	10	273352.08	278383.30	551735.38
1872	11	259302.78	266974.29	526277.07
1872	12	262542.09	274922.30	537464.39
1872	13	265891.64	277830.22	543721.85
1872	14	253538.07	262373.25	515911.32
1872	15	258468.87	259445.08	517913.94
1872	16	254791.94	248390.93	503182.86
1872	17	244277.66	233742.51	478020.17
1872	18	253606.07	234338.53	487944.60
1872	19	242485.31	224019.91	466505.22
1872	20	253304.74	231486.03	484790.78
1872	21	243789.05	249278.69	493067.73
1872	22	221847.65	239026.29	460873.93
1872	23	224086.92	226023.79	450110.71
1872	24	221717.64	213189.33	434906.97
1872	25	231576.11	211227.83	442803.94
1872	26	228036.27	219072.89	447109.17
1872	27	212590.46	212958.05	425548.50
1872	28	213855.40	210021.25	423876.65
1872	29	186586.57	187672.85	374259.42
1872	30	237932.59	219763.38	457695.97
1872	31	223247.19	213227.75	436474.94
1872	32	162621.35	173120.35	335741.70
1872	33	169283.45	175214.73	344498.17
1872	34	164297.48	169654.30	333951.78

To download the Mx and Exposures files for a specific country, you need

to visit the web page of the Human Mortality Database (HMD) at `https://www.mortality.org/Data/DataAvailability`. From there, click on the country of interest, which will take you to a new page. On this page, you will find the Death Rates 1x1 and Exposure-to-risk 1x1 sections.

Manual downloading is necessary if you want to work with specific subpopulations in countries like France, UK, or Germany, as the `hmd.mx` command may not function correctly. In such cases, you will need to manually download these files to your local disk for further processing.

III.2 `demogdata`: create demogdata object from raw data matrices

The `demogdata` command in the R package "demography" serves as a useful tool for creating demography objects from raw data matrices. This function allows users to organize and structure their demographic data into a coherent and manageable format.
With the `demogdata` command, users can input raw data matrices representing various demographic indicators such as mortality rates or fertility rates, and exposures to risk. The command then processes and transforms these raw data matrices into a demography object, which provides a structured framework for analyzing and visualizing demographic information.
By creating a demography object, users can take advantage of the functionalities offered by the "demography" package, including generating life tables, constructing age-specific mortality curves, or estimating life expectancies.

```
demogdata(data, pop, ages, years, type, label,
   name)
```

Arguments:

data	Matrix of data: either mortality rates or fertility rates
pop	Matrix of population values of same dimension as data. These are population numbers as of 30 June of each year (i.e., the "exposures"). So, for example, the number of deaths is data ×pop if data contains mortality rates
ages	Vector of ages corresponding to rows of data
years	Vector of years corresponding to columns of data
type	Character string showing type of demographic series: either "mortality", "fertility" or "migration"
label	Name of area from which the data are taken
name	Name of series: usually male, female or total

III.3 Importing data matrices from an external source (Istat website)

In this paragraph, we will explore how to import two data matrices in TXT format that we previously downloaded from the website of ISTAT (Italian National Institute of Statistics). The first matrix contains age-specific mortality rates, while the second matrix contains the number of exposures to risk at various ages from 1992 to 2012. We can perform various demographic analyses by importing these data matrices into R and utilizing the functionalities of the "demography" package.

mx_italia_1992_2012 - Notepad

File Edit Format View Help

1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
0.008955	0.008670	0.008203	0.007492	0.006773	0.006297	0.006004	0.005721	0.005236	0.005003	0.004670	0.004388	0.004005	0.003763	0.003673	0.003532	0.003431	0.003451	0.003481	0.003431	0.003290
0.000515	0.000464	0.000454	0.000453	0.000453	0.000403	0.000352	0.000302	0.000292	0.000291	0.000291	0.000271	0.000261	0.000251	0.000261	0.000251	0.000241	0.000231	0.000231	0.000241	0.000241
0.000363	0.000373	0.000363	0.000363	0.000343	0.000322	0.000292	0.000262	0.000231	0.000211	0.000211	0.000211	0.000201	0.000191	0.000191	0.000191	0.000181	0.000181	0.000171	0.000171	0.000171
0.000283	0.000303	0.000293	0.000292	0.000272	0.000262	0.000242	0.000221	0.000181	0.000171	0.000171	0.000171	0.000161	0.000151	0.000151	0.000141	0.000141	0.000141	0.000131	0.000131	0.000130
0.000232	0.000252	0.000242	0.000242	0.000222	0.000222	0.000211	0.000191	0.000151	0.000141	0.000141	0.000141	0.000131	0.000131	0.000121	0.000110	0.000110	0.000110	0.000110	0.000100	0.000110
0.000212	0.000222	0.000212	0.000212	0.000202	0.000202	0.000181	0.000171	0.000141	0.000131	0.000131	0.000141	0.000121	0.000110	0.000110	0.000110	0.000100	0.000100	0.000100	0.000090	0.000100
0.000222	0.000222	0.000202	0.000202	0.000192	0.000191	0.000171	0.000151	0.000131	0.000131	0.000131	0.000121	0.000121	0.000111	0.000100	0.000100	0.000100	0.000090	0.000090	0.000090	0.000090
0.000222	0.000222	0.000202	0.000202	0.000182	0.000171	0.000161	0.000141	0.000131	0.000121	0.000131	0.000121	0.000111	0.000100	0.000100	0.000110	0.000100	0.000090	0.000090	0.000080	0.000080
0.000212	0.000212	0.000202	0.000192	0.000171	0.000161	0.000161	0.000131	0.000121	0.000111	0.000131	0.000121	0.000121	0.000100	0.000090	0.000100	0.000100	0.000090	0.000080	0.000080	0.000070
0.000202	0.000202	0.000202	0.000192	0.000172	0.000161	0.000161	0.000141	0.000121	0.000111	0.000131	0.000131	0.000121	0.000100	0.000090	0.000090	0.000090	0.000090	0.000080	0.000090	0.000070
0.000192	0.000192	0.000192	0.000182	0.000161	0.000161	0.000151	0.000141	0.000131	0.000121	0.000131	0.000121	0.000111	0.000101	0.000090	0.000090	0.000090	0.000090	0.000090	0.000090	0.000080
0.000202	0.000202	0.000192	0.000192	0.000182	0.000182	0.000171	0.000161	0.000141	0.000131	0.000121	0.000131	0.000111	0.000111	0.000101	0.000090	0.000090	0.000111	0.000111	0.000100	0.000090
0.000233	0.000243	0.000222	0.000222	0.000212	0.000222	0.000202	0.000191	0.000171	0.000161	0.000141	0.000141	0.000131	0.000131	0.000131	0.000121	0.000111	0.000131	0.000121	0.000121	0.000100
0.000304	0.000314	0.000293	0.000283	0.000273	0.000272	0.000252	0.000242	0.000232	0.000211	0.000181	0.000181	0.000161	0.000171	0.000161	0.000151	0.000151	0.000161	0.000151	0.000141	0.000131
0.000435	0.000425	0.000405	0.000384	0.000374	0.000353	0.000343	0.000333	0.000302	0.000272	0.000262	0.000242	0.000231	0.000211	0.000211	0.000211	0.000211	0.000231	0.000201	0.000191	0.000161
0.000618	0.000587	0.000567	0.000526	0.000505	0.000475	0.000444	0.000424	0.000383	0.000363	0.000353	0.000342	0.000312	0.000282	0.000282	0.000292	0.000292	0.000302	0.000261	0.000261	0.000221
0.000801	0.000780	0.000750	0.000698	0.000667	0.000626	0.000576	0.000535	0.000504	0.000484	0.000474	0.000463	0.000433	0.000392	0.000382	0.000382	0.000382	0.000382	0.000342	0.000342	0.000292
0.000934	0.000913	0.000892	0.000851	0.000820	0.000768	0.000717	0.000656	0.000615	0.000595	0.000585	0.000574	0.000544	0.000513	0.000483	0.000473	0.000473	0.000463	0.000423	0.000402	0.000352
0.001016	0.001005	0.000984	0.000963	0.000932	0.000880	0.000819	0.000768	0.000747	0.000706	0.000686	0.000666	0.000635	0.000614	0.000574	0.000544	0.000534	0.000523	0.000483	0.000463	0.000412
0.001068	0.001057	0.001026	0.001015	0.001004	0.000962	0.000911	0.000880	0.000859	0.000818	0.000747	0.000716	0.000706	0.000695	0.000635	0.000584	0.000564	0.000554	0.000534	0.000503	0.000453
0.001109	0.001079	0.001037	0.001026	0.001015	0.001014	0.000973	0.000962	0.000930	0.000879	0.000798	0.000737	0.000716	0.000726	0.000686	0.000635	0.000605	0.000605	0.000564	0.000524	0.000473
0.001162	0.001120	0.001048	0.001007	0.000985	0.001005	0.000994	0.001013	0.000982	0.000931	0.000829	0.000758	0.000737	0.000727	0.000726	0.000666	0.000625	0.000615	0.000554	0.000524	0.000484
0.001214	0.001173	0.001070	0.001008	0.000996	0.001036	0.001025	0.001045	0.001003	0.000942	0.000860	0.000779	0.000748	0.000737	0.000747	0.000696	0.000646	0.000616	0.000555	0.000534	0.000504
0.001236	0.001164	0.001051	0.000988	0.000977	0.001027	0.001047	0.001046	0.001004	0.000943	0.000881	0.000789	0.000759	0.000748	0.000758	0.000727	0.000677	0.000626	0.000565	0.000545	0.000514
0.001278	0.001196	0.001062	0.000989	0.000978	0.001008	0.001027	0.001006	0.000985	0.000944	0.000902	0.000800	0.000759	0.000749	0.000758	0.000728	0.000667	0.000647	0.000576	0.000565	0.000535
0.001331	0.001228	0.001094	0.001000	0.000999	0.001009	0.000998	0.000967	0.000955	0.000924	0.000893	0.000811	0.000780	0.000780	0.000769	0.000738	0.000667	0.000637	0.000586	0.000576	0.000545
0.001354	0.001281	0.001146	0.001053	0.001031	0.001010	0.000999	0.000947	0.000936	0.000915	0.000884	0.000822	0.000781	0.000790	0.000790	0.000769	0.000698	0.000658	0.000607	0.000576	0.000545

pop_italia_1992_2012 - Notepad

File Edit Format View Help

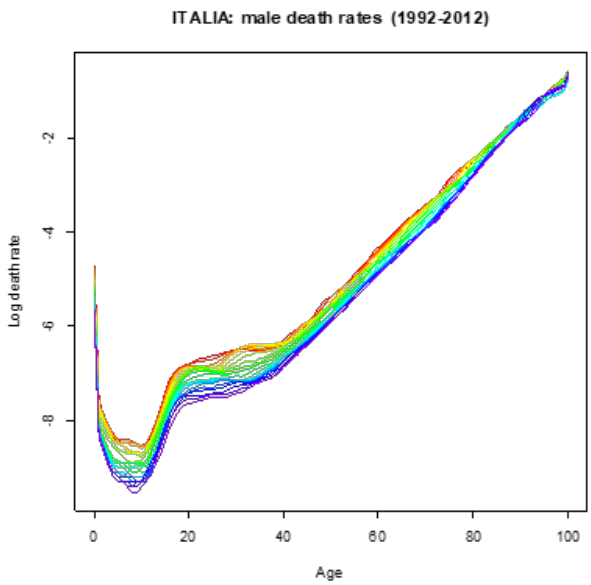
1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002	2003	2004	2005	2006	2007	2008	2009	2010	2011	2012
262256	293100	281875	274171	269122	273472	273404	270959	272792	274518	268267	274839	278291	286322	283649	286202	287891	291773	287649	282221	272408
267914	264589	292378	280911	273119	268728	271549	270301	269376	271686	271974	269702	275726	279519	286257	284033	287156	288161	290245	285931	279765
270613	270215	266124	291954	280284	272417	268475	270552	268321	268009	270631	273695	271783	276848	280728	286512	285277	287923	288058	288905	284756
273681	271914	271628	267907	291372	279718	271667	268351	269711	266437	266702	271945	275782	273928	278129	281678	287552	286476	288199	287787	287411
267722	275090	272863	273160	270243	290732	278860	271017	268274	268841	264789	268442	273495	277983	276100	278873	283412	288375	286963	288157	287514
270798	268585	276125	273918	275334	273657	290059	278324	270494	268120	268104	266673	270436	275159	280324	277776	280721	284716	288450	286978	288213
281702	272056	269269	277167	275287	278264	277272	289696	277828	269896	268064	269763	268896	272827	276859	282150	280583	282094	285399	288263	286615
286983	282668	272963	269983	278499	277396	281425	280841	289156	277278	269478	269696	271604	271141	275206	278220	285031	282972	282891	285879	287655
295917	287936	283481	273862	270912	280479	279695	284385	284236	288717	276891	271109	271262	273428	273738	277065	280675	287388	284693	283381	285572
305422	296877	288683	284348	275101	272273	282659	281842	287390	287799	288467	278498	272994	273193	275705	275699	279872	282725	289474	286107	282782
313202	306476	297623	289499	285485	276998	273896	284707	284025	290480	291140	289801	280240	275433	275365	277478	278607	282536	284244	291290	285992
319714	313920	307341	298411	290571	287181	279050	275401	286776	286194	293288	291924	291333	282354	277786	277058	280137	281223	284690	285519	290793
332506	320499	314535	308262	299422	292133	289045	280996	276864	288884	288214	294022	292878	293196	284573	279596	279623	282727	283504	286436	284887
355326	333136	321195	315215	309420	300935	293872	290802	282929	278359	290940	288944	294870	294387	295388	286207	282400	282155	284693	285365	285832
370807	355619	333623	321877	316024	311102	302586	295479	292507	284992	279903	291815	289821	296132	295870	296944	288777	285166	284292	286425	284775
392536	370835	355802	334114	322758	317236	313079	304113	297087	294336	286718	280760	292905	291152	297807	296981	299351	291438	287502	285994	285905
416912	392494	370836	355944	334759	324159	318655	314848	305639	298797	295918	286921	281932	294491	293102	299054	298899	301954	293921	289613	285523
438772	416679	392342	370776	356135	335784	325578	319965	316652	307354	300342	296294	287491	283999	296763	294856	301397	301376	304540	296276	289699
441511	438217	415420	392178	370711	356566	336904	326986	321417	318587	308785	300986	297339	289746	287316	299705	298614	305203	304987	307830	296890
450360	440749	437756	416166	391945	370810	356999	337796	328420	322854	320017	308580	302619	298482	291075	289540	304206	302932	308730	307860	308994
460945	449410	440231	437221	415881	392036	370999	357356	338934	329992	323843	319093	310655	304060	298452	291200	294533	308300	305988	310824	309180
454934	459623	448771	439609	436640	415933	392106	371060	357946	340339	331166	322665	321786	312481	304182	298444	295978	299362	311222	308553	312864
468311	453827	458786	448107	438898	436312	416019	392081	371410	358858	341330	330093	326150	324394	312557	304537	303783	300876	302605	314359	312052
463382	467361	453117	457906	447249	438513	435955	416003	392272	371939	359463	340888	335181	329516	324666	312941	310424	309146	304180	306561	318568

The given code imports two data matrices containing age-specific mortality rates and exposures to risk for the Italian male population from 1992 to 2012.
The `read.table` function is used to read the data from text files, and the resulting data is assigned to the matrices `data` and `pop`. The vectors `ages` and `years` are created to represent the age groups and years included in the data. Finally, the `demogdata` function is used to create a demography object, `italy_istat`, which combines the imported data with additional information such as the data type, location (ITALIA), and gender (male).

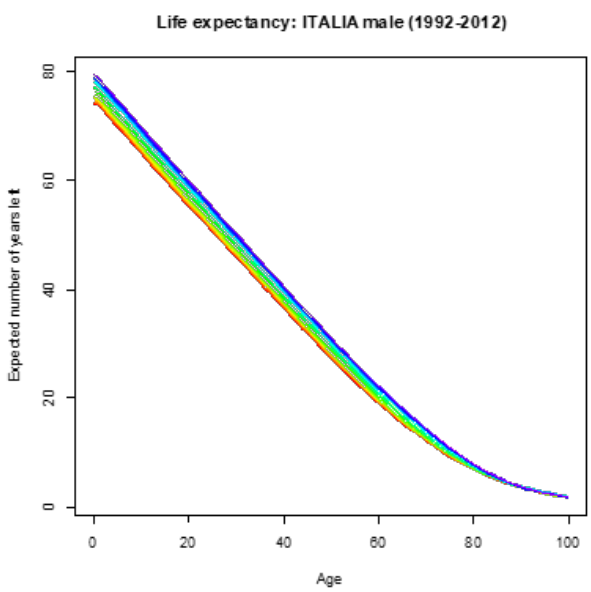
```
data <-  read.table (file="C:/Users/Francesco/
   Documents/Didattica/mx_italia_1992_2012.txt",
   header=TRUE, sep="\t", dec=".")

pop <- read.table (file="C:/Users/Francesco/
   Documents/Didattica/pop_italia_1992_2012.txt",
    header=TRUE, sep="\t", dec=".")

ages <- (0:100)

years <- (1992:2012)

italy_istat <- demogdata(data, pop, ages, years,
   "mortality", "ITALIA", "male")
```

Once the demography object `italy_istat` is created, we can apply the same functions from the `demography` package to construct and plot log age-specific mortality rates, life tables, life expectancy, and survivors.
The following lines of code and the corresponding plots serve as an example.

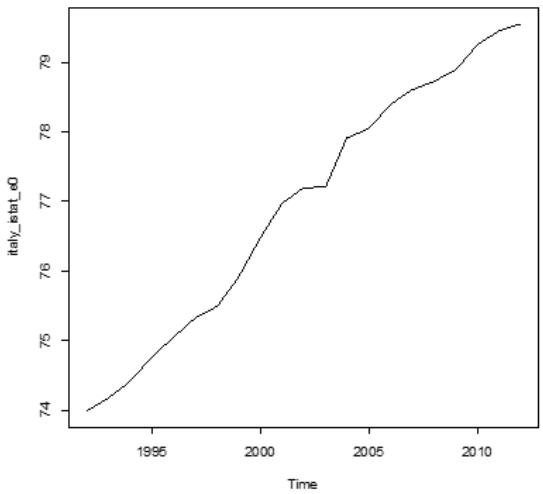
```
plot(italy_istat)
```




```
plot(lifetable(italy_istat))
```



```
plot(life.expectancy(italy_istat))
```




```
plot((lifetable(italy_istat, years = 2012, ages =
    italy_istat$age, max.age = 100)), type = c("
   period")))$lx)
```