Uso o pacote “Lifecontingencies” do software R e de outras funções para cálculos atuariais do ramo vida[1]

Débora Viana da Silva
Curso de ciências Atuariais UFMG
Aloísio Joaquim Freitas Ribeiro
Departamento de Estatística – UFMG

2016

[1] Este texto foi escrito como resultado do trabalho de Iniciação Científica da aluna do Curso de Ciências Atuariais Débora Viana da Silva, bolsista do CNPq, orientado por Aloísio Joaquim Freitas Ribeiro.

Capitulo 1 - Cálculos atuarias de seguros de vida usando o software R

1) Introdução

O Programa estatístico R possui um pacote chamado *LifeContingencies* construído por Giorgio Alfredo Spedicato, Reinhold Kainhofer, Kevin J. Owens, Christophe Dutang e Ernesto Schirmacher. A documentação do pacote pode ser encontrada https://cran.r-project.org/web/packages/lifecontingencies/lifecontingencies.pdf .

O pacotes possui funções úteis para cálculos atuariais que podem ser agrupadas em 3 classes:

a) Funções para construção da tábua de vida e obtenção de funções da mesma.

b) Funções financeiras.

c) Funções para cálculos atuarias.

Neste trabalho apresentaremos através de exemplos a utilização destas funções iniciando com aquelas relativas à construção da tábua de vida.

É importante ressaltar a seguinte observação dos autores do "lifecontingencies": "This package and functions herein are provided as is, without any guarantee regarding the accuracy of calculations. The author disclaims any liability arising by any losses due to direct or indirect use of this package".

2) Funções para construção da tábua de vida

Uma tábua de vida, também chamada de tábua de mortalidade, é um instrumento muito importante para se realizar cálculos atuariais para planos de previdência e seguro de vida. Uma tábua de vida descreve a sobrevivência futura de uma população. São funções da tábua de vida.

l_x quantidade de pessoas vivas na idade exata x

${}_nd_x$ número de mortes entre as idades x e x+n

${}_nq_x$ probabilidade de morrer entre as idades x e x+n

${}_np_x$ probabilidade de sobreviver entre as idades x e x+n

${}_na_x$ tempo médio vivido das pessoas que morreram no intervalo x a x+n

${}_nT_x$ número de pessoas-anos vividos após idade exata x

${}_nm_x$ taxa de mortalidade central

${}_ne_x$ esperança de vida entre a idade x e x+n

2.1) Construindo uma tábua de vida

Tabuas de vida podem ser construídas utilizando as ferramentas *lifetable e probs2lifetable.* Com a ferramenta *lifetable* são necessárias as informações sobre idade (x) e número de sobreviventes à idade x (l_x). Com a ferramenta *probs2lifetable* é preciso indicar as idades (x) com as respectivas probabilidades de morte (q_x) ou sobrevivência (p_x) associadas.

2.1.1) Usando a ferramenta *lifetable*

Para exemplificar o uso da ferramenta *lifetable* vamos utilizar o conjunto de dados *soaLt*, que contem valores de x e l_x para uma tábua de vida construída pelo SOA ("Society of Actuaries" EUA) e utilizada em vários livros de Ciências Atuariais.

Vários outros conjuntos de dados com funções de tábuas de vida são disponíveis no pacote *lifecontingencies*. Por exemplo o conjunto de dados AF92Lt, que contém a tábua de vida feminina para o Reino Unido, ano de 1992. Para ver estes arquivos de dados, execute o comando

```
> try(data(package="lifecontingencies"))
```

O primeiro passo para construir a tábua de vida consiste em carregar o conjunto de dados *soaLt,* que contém duas variáveis: idade (x) e número de sobreviventes à idade x (l_x).

```
> data(soaLt)              # carrega o arquivo SoaLt
> head(soaLt)              # mostra as primeiras linhas do

arquivo
    x          Ix
1   0    10000000
2   1    9949901
3   2    9899801
4   3    9849702
5   4    9799602
6   5    9749503

> tail(soaLt)            # mostra as ultimas linhas do arquivo
    x     Ix
106  105 1668
107  106  727
108  107  292
109  108  108
110  109   36
111  110   11
```

A ferramenta *lifetable* é um objeto do tipo classe e pode ser gerada usando a função *new*. Antes de gerar a tabua de vida, para sabermos como se caracteriza a classe *lifetable,* utilizamos a função *showClass*.

```
> showClass("lifetable")       # descreve a classe lifetable

Class "lifetable" [package "lifecontingencies"]
Slots
Name:          x        lx      name
Class:   numeric   numeric character
Known Subclasses: "actuarialtable"
```

Depois de carregados os dados, a tábua de vida é gerada a partir das informações de x e l_x fazendo:

```
> TVsoa=new("lifetable",x=soaLt$x,lx=soaLt$Ix, name=" SOA")
> TVsoa     # mostra a tabela de vida TVSoa

Life table SOA
       x        lx        px              ex
1      0     10000000    0.9949901    71.3469199
2      1      9949901    0.9949648    70.7061635
3      2      9899801    0.9949394    70.0639824
4      3      9849702    0.9949136    69.4203550
:
108  107          292     0.3698630    0.5308219
109  108          108     0.3333333    0.4351852
110  109           36     0.3055556    0.3055556
```

Podemos também entrar diretamente com os dados x e l_x. Considere os valores de x de 0 a 10 com valores de l_x variando de 1000 a 500, com intervalos de 50.

```
> x = 0:10
> lx = seq(1000,500,-50)
> TV = new("lifetable",x=x,lx=lx,name="tábua de vida")
> TV

Life table tábua de vida
   x   lx        px        ex
1  0 1000 0.9500000 7.2500000
2  1  950 0.9473684 6.6315789
3  2  900 0.9444444 6.0000000
4  3  850 0.9411765 5.3529412
5  4  800 0.9375000 4.6875000
6  5  750 0.9333333 4.0000000
7  6  700 0.9285714 3.2857143
8  7  650 0.9230769 2.5384615
9  8  600 0.9166667 1.7500000
10 9  550 0.9090909 0.9090909
```

Um cuidado que deve ser tomado é que o vetor l_x não pode conter valores faltantes (NA) ou iguais a zero.

Para obter os "slots" da classe TV que acabamos de criar, que são x, l_x e "name" proceda como segue.

```
> TV@x
 [1]  0  1  2  3  4  5  6  7  8  9 10
> TV@lx
 [1] 1000  950  900  850  800  750  700  650  600  550  500
> TV@name
[1] "tábua de vida"
```

2.1.2) Construindo tábua de vida usando a ferramenta probs2lifetable

Tábuas de mortalidade também podem ser obtidas a partir dos vetores de probabilidades px ou qx usando a função *probs2lifetable*. Os valores de q_x e p_x devem estar entre 0 e 1.

Os argumentos desta função são:

probs	A real valued vector representing either one year survival or death probabilities. The last value in the vector must be either 1 or 0, depending if it represents death or survival probabilities respectively.
radix	The radix of the life table.
Type	Character value either "px" or "qx" indicating how probabilities must be interpreted.
name	The character value to be put in the corresponding slot of returned object.

Para exemplificar, vamos utilizar as probabilidades (JP8587F) de morte, para o sexo feminino, que constam no conjunto de dados *demoJapan* (variável JP8587F). O primeiro passo é carregar o arquivo de dados.

```
> data(demoJapan)   # carrega o objeto do tipo classe
> head(demoJapan)   # mostra as primeiras linhas

   age  JP8587M    JP8587F
1   0    0.00137     0.00126
2   1    0.00098     0.00094
3   2    0.00067     0.00065
4   3    0.00048     0.00044
5   4    0.00039     0.00030
6   5    0.00036     0.00023
```

Uma vez carregado o arquivo de dados, a tábua de vida é construída através do seguinte comando.

```
>    TV    =    probs2lifetable(demoJapan$JP8587F,    type="qx",
radix=10000, name="Japão – sexo Feminino")

> TV # mostra a tabela de vida resultante
Life table Japão – sexo Feminino
      x           lx      px         ex
1     0 1.000000e+04 0.99874 81.5643723
2     1 9.987400e+03 0.99906 80.6672731
3     2 9.978012e+03 0.99935 79.7431717
4     3 9.971526e+03 0.99956 78.7950384
5     4 9.967139e+03 0.99970 77.8297235
6     5 9.964149e+03 0.99977 76.8530794
7     6 9.961857e+03 0.99979 75.8707597
8     7 9.959765e+03 0.99981 74.8866959
9     8 9.957872e+03 0.99982 73.9009271
10    9 9.956080e+03 0.99984 72.9142317
:
109 108 1.686990e-01 0.23348  0.23348010
```

Na tabela acima, "e_x" é a esperança de vida em anos completos.

2.2) Tábuas de comutação

Antes da disponibilidade de computadores, atuários utilizavam uma coleção de funções, denominadas funções de comutação, para calcular valores presentes atuariais de seguros e anuidades de vida. Estas funções são baseadas num modelo de sobrevivência determinístico e numa taxa de juros constante. Com o avanço da tecnologia as limitações das funções de comutação tornaram-se mais evidentes, elas não são adequadas quando temos tabelas de vidas com períodos de seleção, taxas de juros que mudam no tempo e benefícios com valores que crescem de maneira irregular ao longo do tempo.

Apesar de não serem mais necessárias, as funções de comutação ainda são úteis para descrever cálculos atuariais e são encontradas em vários livros e softwares. As funções de comutação são funções da função l_x da tábua de vida e da taxa de desconto $v = (1+i)^{-1}$, onde i é a taxa anual de juros. As funções de comutação são organizadas numa tabela denominada tabela de comutação. As principais funções de comutação são:

$$D_x = v^x l_x \qquad N_x = \sum_{i=x}^{\infty} v^i l_i = \sum_{i=x}^{\infty} D_i$$

$$C_x = v^{x+1} l_x \qquad M_x = \sum_{i=x}^{\infty} v^{i+1} l_i = \sum_{i=x}^{\infty} C_i$$

$$R_x = \sum_{j=x}^{\infty} M_j$$

Uma tabela de comutação *("actuarialtable")* com as funções definidas acima pode ser facilmente obtida no R. Uma *"actuarialtable"* é também um objeto do tipo classe e pode ser gerada usando a função *new*. Para construir uma tábua de comutação no R é preciso especificar os vetores de idade (x) e sobreviventes a idade x (l_x) além da taxa de juros como mostra o resultado do comando *showClass("actuarialtable").*

```
> showClass("actuarialtable")

Class "actuarialtable" [package "lifecontingencies"]
Slots:
Name:   interest         x        lx      name
Class:   numeric   numeric   numeric character
Extends: "lifetable"
```

Considerando uma taxa de juros i = 0,05 e os seguintes vetores de idades e número de sobreviventes, podemos criar uma tábua de comutação como abaixo:

```
> x = 0:10
> lx = seq(1000,500,-50)
>    TA    =    new("actuarialtable",x=x,lx=lx,name="tábua    de
vida",interest=0.05)
> TA

Actuarial table  tábua de vida interest rate  5 %
    x   lx        Dx        Nx        Cx       Mx        Rx
1   0 1000 1000.0000 6753.0458  47.61905 678.4264 5184.4252
2   1  950  904.7619 5753.0458  45.35147 630.8073 4505.9988
3   2  900  816.3265 4848.2839  43.19188 585.4559 3875.1915
4   3  850  734.2620 4031.9574  41.13512 542.2640 3289.7356
5   4  800  658.1620 3297.6954  39.17631 501.1289 2747.4716
6   5  750  587.6446 2639.5334  37.31077 461.9526 2246.3428
7   6  700  522.3508 2051.8888  35.53407 424.6418 1784.3902
8   7  650  461.9429 1529.5380  33.84197 389.1077 1359.7484
9   8  600  406.1036 1067.5951  32.23045 355.2658  970.6407
10  9  550  354.5349  661.4915  30.69566 323.0353  615.3750
11 10  500  306.9566  306.9566 292.33964 292.3396  292.3396
```

2.3) Obtendo probabilidades de sobrevivência e morte a partir dos objetos *lifetable* e *actuarialtable*

As funções *dxt, Lxt, Tx, pxt, qxt, mxt e exn,* do pacote *Lifeconctngencies,* retornam como resultados as respectivas quantidades:

${}_td_x$ – número de mortes entre as idades x e x + t

${}_tL_x$ – número de pessoas-ano vividos após entre as idades x e x + t

T_x – número de pessoas-ano vivido após a idade exata x

${}_tp_x$ - probabilidade de um pessoa com x anos sobreviver até a idade x + t

${}_tq_x$ – probabilidade de uma pessoa com x anos morrer antes da idade x + t

${}_tm_x$– taxa central de mortalidade

$e_{x:n}$ – tempo esperado de vida de uma pessoa com x anos até a idade x + n

Todas estas funções têm como argumento um objeto do tipo *lifetable* ou *actuarialtable.*

```
dxt(object, x, t)
Lxt(object, x, t, fxt = 0.5)
Tx (object, x)
pxt(object, x, t, fractional = "linear")
qxt(object, x, t, fractional = "linear")
mxt(object, x, t)
exn(object, x, n,type="curtate")
```

Object	Um objeto do tipo lifetable
X	Age when the calculation starts.
T	Age when the calculation ends, default=1.
fractional	Assumptions for fractional age. One of "linear", "hyperbolic", "constant force".
Fxt	separation factor, default value is 0.5 (half year).
Type	Either complete or curtate

A esperança de vida em anos completos (“curtate life expectancy”) é o numero esperado de anos completos vividos, enquanto a esperança de vida completa consiste no tempo esperado de vida. Elas são denotadas usualmente por $\overset{o}{e}_x$ e e_x. Pode se mostrar que $\overset{o}{e}_x = e_x + 0.5$. Quando consideramos o tempo esperado de vida até o tempo n, pode-se mostrar que $\overset{o}{e}_{x:\overline{n}|} = e_{x:\overline{n}|} + 0{,}5({}_nq_x)$.

Para exemplificar o uso destas funções, vamos usar a tabua de vida TV obtida acima.

```
> d52 = dxt(TV,x=5,t=2)
>  LT52 =Lxt(TV,x = 5, t = 2, fxt = 0.5)
>  T5  =  Tx(TV,5)
>  q52 = qxt(TV,x=5,t=2)
>  p52 = pxt(TV,x=5,t=2)
>  e52 = exn(TV,x=5,n=2)
>  m52 = mxt(TV, x=5 ,t = 2)
> resultados = round(c(d52,LT52,T5,p52,q52,e52,m52),4)
> resultados
[1] 4.3837 19923.8134 770757.5722 0.9996 0.0004 1.9993
[7] 0.0002
```

Os valores de t nas funções *pxt e qxt* podem ser fracionários. Veja abaixo os argumentos destas funções.

object	A lifetable object.
X	Age of life x.
T	Period until which the age shall be evaluated. Default value is 1.
fractional	Assumptions for fractional age. "linear" for linear interpolation between consecutive ages. "constant" for constant force of mortality "hyperbolic" for Balducci assumptions.

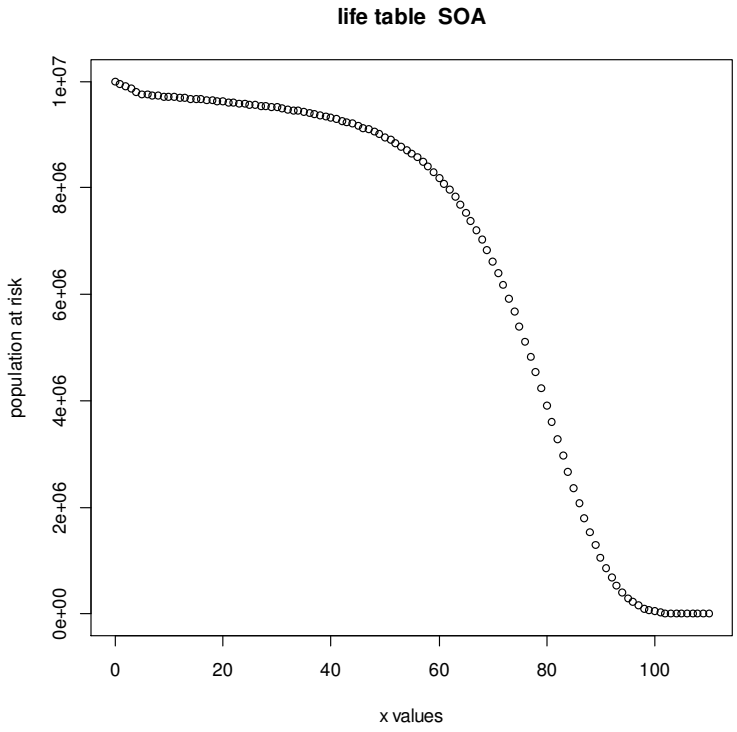
Por exemplo, calculando a probabilidade de morrer entre as idades 5 e 7,5 utilizando suposição "linear".

```
> q52.5 = qxt(TV,x=5,t=2.5,fractional="linear")
> q52.5
[1] 0.1666667
```

2.4) Construindo gráficos utilizando as funções da tabela de vida

A função *plot* pode ser aplicada a um objeto do tipo *lifetable* ou *actuarialtable* para contruir um grafico da função l_x versus x. Por exemplo:

```
> plot(TVsoa)
```



3) Cálculos financeiros

A taxa de juros mede a variação do valor do dinheiro no tempo. Diferentes tipos de taxas de juros são encontrados na literatura. A expressão abaixo apresenta a relação entre os seguintes tipos de taxas de juros:

i – taxa efetiva de juros,

d – taxa efetiva de desconto,

$i^{(m)}$ – taxa nominal de juros (m – número de períodos de capitalização de tamanho 1/m), $d^{(m)}$ – taxa nominal de desconto

δ – força de juros.

$$(1+i)^t = \left(1+\frac{i^{(m)}}{m}\right)^t = \exp(\delta t) = (1-d)^{-t} = \left(1-\frac{d^{(m)}}{m}\right)^{-t}$$

O pacote *Lifencontingencies* possui algumas funções úteis para converter diferentes tipos de taxas.

A função *interest2Discount* converte taxa efetiva de juros em taxa efetiva de desconto e a função *discount2interest* faz o contrário. Por exemplo:

```
>interest2Discount(0.05) #converte a taxa de juros i = 0.05 em
taxa de desconto d
[1] 0.04761905

>discount2Interest(0.05) #converte a taxa de desconto d = 0.05
em taxa de juros i
[1] 0.05263158
```

A funções *real2Nominal* converte taxa efetiva em taxa nominal e função *nominal2Real* converte taxa nominal em taxa efetiva. Os argumentos destas funções são:

i	The rate to be converted.
k	The original / target compounting frequency.
type	Either "interest" (default) or "discount".

Os exemplos seguintes indicam como fazer a conversão de taxas efetiva em nominais e vice-versa. Em todos eles foram considerados capitalização mensal.

a) converter taxa efetiva de juros de 5% ao ano em taxa nominal de juros.

```
> real2Nominal(i=0.05,k=12,type="interest")
[1] 0.04888949.
```

b) converter taxa efetiva de desconto de 5% ao ano em taxa nominal de desconto.

```
>  real2Nominal(i=0.05,k=12,type="discount")
[1] 0.05118383
```

c) converter taxa nominal de juros de 5% ao ano em taxa efetiva de juros.

```
>  nominal2Real(i=0.05,k=12,type="interest")
[1] 0.0511619
```

d) converter taxa nominal de desconto de 5% ao ano em taxa efetiva de desconto

```
>  nominal2Real(i=0.05,k=12,type="discount")
[1] 0.04886993
```

Por último, a função *interest2Intensity* converte taxa efetiva de juros em força de juros e a função *intensity2Interest* faz o contrário, como pode ser visto nos exemplos abaixo:

```
> interest2Intensity(0.06) # converte a taxa efetiva de juros
de 6% ao ano em força de juros
[1] 0.05826891
> intensity2Interest(0.05826891) # converte a força de juros
igual a 0.05826891 em taxa efetiva anual de juros
[1] 0.06
```

3.1) Cálculo de valores presentes

A função *presentValue* calcula o valor presente de um pagamento realizado no tempo t ou de uma série de pagamentos realizados nos tempos indicados.

Os argumentos desta função são:

cashFlows	Vector of cashFlow, must be coherent with timeIds
timeIds	Vector of points of time where cashFlows are due.
interestRates	A numeric value or a time-size vector of interest rate used to discount cash flow.
probabilities	Optional vector of probabilities.

Os vetores com os valores de *cashFlows, timeIds* e *probabilities* devem ter o mesmo tamanho. Caso o vetor de probabilidades não seja especificado considera-se que os pagamentos ocorrerão com certeza.

Antes de apresentarmos exemplos de aplicação da função *presentValue*, vamos recordar rapidamente como calcular o valor presente de pagamento de C unidades monetárias que deverá ser realizado daqui a t anos, considerando taxa anual nominal de juros $i^{(m)}$ por período de tamanho 1/m.

Por exemplo, o valor presente de 10 unidades monetárias pagáveis daqui a 10 anos considerando taxa nominal anual de juros de 5% é

$$VP = 10\left(1+\frac{0{,}05}{12}\right)^{10\,x\,12}$$

Neste caso a taxa efetiva mensal de juros é $\frac{0,05}{12}$ e o pagamento será realizado ao final de 120 meses. Ao utilizar a função *presentValue* estes valores devem ser indicados nos argumentos *interestRates* e *timeIds* como segue.

```
> VP = presentValue(cash=10,interest=0.05/12,timeIds=120)
> VP
6.07161
```

A função *presenteValue* também é útil no cálculo o valor presente de uma série de fluxo de pagamentos. Por exemplo, uma pessoa terá de realizar pagamentos de empréstimos realizados nos valores de 100, 500 e 1000 reais ao final de 12,18 e 24 meses respectivamente. Considerando que as taxas efetivas mensais de juros destes empréstimos são respectivamente 0.5%, 0,6% e 0.7%, em valor presente qual o valor destes empréstimos?

```
>     presentValue(cash=c(100,500,1000),i=c(0.005,0.006,0.007),
timeIds = c(12,18,24))
[1] 1388.998
```

Considere uma série de pagamentos de 1 unidade monetária no início de cada ano por um período de n anos, com primeiro pagamento no tempo 0. Esta série de pagamentos é denominada de **anuidade certa antecipada**. Se os pagamentos forem realizados ao final do ano ela é chamada de **anuidade certa postecipada**. Se o tempo de duração dos pagamentos é infinito, estas séries de pagamentos são chamadas de **perpetuidades.**

A funçao *presentValue* pode ser usada para calcular o valor presente de uma anuidade certa antecipada ou postecipada. Para isto, temos de especificar o vetor de pagamentos unitários, os tempos de realização dos pagamentos e as taxas de juros envolvidas.

Exemplo 3.1.1: Considerando taxa efetiva de juros de 5% ao ano, qual o valor presente de uma anuidade antecipada de duração igual a 10 anos com pagamentos de 1 unidade monetária no início de cada ano?

```
> t=0:9
> presentValue(cash=c, interest=interest,timeIds=t)
[1] 8.107822
```

Considerando unidade certa postecipada, temos de modificar o vetor de tempos de pagamento para t variando de 1 a 10.

```
> t=1:10
> presentValue(cash=c, interest=interest,timeIds=t)
[1] 7.721735
```

Exemplo 3.1.2: Considerando taxa nominal de juros de 5% ao ano, qual o valor presente de uma anuidade antecipada de duração igual a 5 anos com pagamentos mensais de 1 unidade monetária?

```
> t = 0:119
> c=rep(1,120)
> interest=rep(0.05/12,120)
> presentValue(cash=c, interest = interest, timeIds=t)

[1] 94.67419
```

Para o caso de pagamentos imediatos, é só mudar o vetor de tempos.

```
> t=1:120
> presentValue(cash=c, interest = interest, timeIds=t)
[1] 94.28135
```

Valores presentes de anuidades certas são mais facilmente obtidos com a função *annuity,* cujos argumentos são:

I	Effective interest rate expressed in decimal form. E.g. 0.03 means 3%. It can be a vector of interest rates of the same length of periods.
N	Periods for payments. If n = infinity then annuity returns the value of a perpetuity (either immediate or due).
M	Deferring period, whose default value is zero.
K	Yearly payments frequency. A payment of *k^-1* is supposed to be performed at the end of each year.
Type	A string, either "immediate" or "due".

Para os exemplos 1 e 2 acima, os mesmos resultados são obtidos fazendo:

a) Anuidade certa antecipada com pagamentos anuais de 1 u.m. e taxa efetiva anual de juros de 5%

```
> annuity(i=0.05,n=10,type="due")
```

```
[1] 8.107822
```

b) Anuidade certa postecipada com pagamentos anuais de 1 u.m. e taxa efetiva anual de juros de 5% de duração igual a 5 anos

```
> annuity(i=0.05,n=10,type="immediate")
[1] 7.721735
```

a) Anuidade certa antecipada com pagamentos mensais de 1 u.m. e taxa nominal anual de juros de 5% de duração igual a 5 anos.

```
> annuity(i=0.05/12,n=120,type="due")
[1] 94.67419
```

b) Anuidade certa postecipda com pagamentos mensais de 1 u.m. e taxa nominal anual de juros de 5%

```
> annuity(i=0.05/12,n=120,type="immediate")

[1] 94.28135
```

As funções *increasingAnuity* e *decreasingAnnuity* calculam o valor presente de anuidades crescentes e decrescentes com as seguintes características:

1) no caso crescente, o pagamento inicial é de uma unidade monetária e os pagamentos seguintes aumentam uma unidade monetária por ano.
2) no caso decrescente, o pagamento inicial é de n unidades monetárias e os demais pagamentos decrescem uma unidade monetária por ano.

Os argumentos desta função são:

i	A numeric value representing the interest rate.
n	The number of periods.
type	A character value, specifying the annuity type. Either "immediate" or "due". Default value is "immediate".

Exemplo 3.1.3: Considere uma **anuidade antecipada crescente** de duração 10 anos, com pagamento inicial de 1 unidade monetária no tempo 0 e pagamento final de 10 unidades monetárias no tempo 9. Com uma taxa efetiva de juros de 5% ano, o valor presente desta anuidade pode ser obtido do seguinte modo.

```
> increasingAnnuity(i=0.05,n=10,type="due")
[1] 41.34247
```

No caso de uma anuidade antecipada decrescente de duração 10 anos com pagamento inicial de 10 unidades monetárias no tempo 0 e pagamento final de 1 unidade monetária no tempo 10. Com uma taxa efetiva de juros de 5%, o valor presente desta anuidade é obtido do seguinte modo.

```
> decreasingAnnuity(i=0.05,n=10,type="due")
[1] 47.84357
```

3.2) Cálculo de valores futuros de anuidades

Para calcularmos o valor futuro de uma anuidade postecipada ou antecipada podemos multiplicar seu valor presente pelo fator de acumulação do período correspondente, ou usar a função *accumulatedValue* cujos argumentos são os mesmos da função *annuity.*

Considere uma anuidade antecipada de duração igual a 10 anos pagando uma unidade monetária no início de cada ano. Considerando taxa efetiva de juros de 5% ao ano, o valor fituro pode se obtido fazendo:

```
> VF = annuity(i=0.05,n=10,k=1, type="due")*(1+0.05)^10
```

ou

```
> VF=accumulatedValue(i=0.05,n=10,k=1,Type="due")
> VF
[1] 13.20679
```

4) Funções para cálculos atuariais

Nesta seção mostraremos através de exemplos como calcular valores presentes atuariais para seguros e anuidades no caso discreto. Para isto usaremos tábuas de vida e tábuas de comutação.

4.1) Seguros de vida

Para exemplificar o uso das funções atuariais disponíveis para seguros de vida vamos construir uma tabela de comutação com os dados "soaLt" disponibilizados no pacote "*lifecontingencies"*.

```
>data(soaLt)          # carrega o arquivo SoaLt
>soAct=new("actuarialtable",x=soaLt$x,lx=soaLt$Ix,interest=0.0
3,name="SOA") #tabela atuarial com juros de 3%
```

As funções são disponíveis no pacote "*lifecontingencies"* para cálculo de valores presentes atuariais para seguros de vida são:

a) Axn – para seguros de vida temporários ou de vida inteira, com ou sem diferimento
b) Exn – para seguro dotal puro
c) AExn – para seguro dotal misto
d) IAxn – para seguro de vida temporário crescente de duração n anos.
e) DAxn – para seguro de vida temporário decrescente crescente de duração n anos.
f) Axyzn – para seguros de duas vidas.

4.1.1) Seguro de vida inteira ou temporário:

Para obter o valor presente atuarial de seguros destas modalidades podemos usar a função *Axn*, cujos argumentos são:

Actuarialtable	An actuarial table object
X	Age of the insured.
N	Coverage period, if missing the insurance is considered whole life $n=\omega-x-m$
I	Interest rate (overrides the interest rate slot in actuarialtable).
M	Deferring period, even fractional, if missing assumed to be 0.
K	Number of periods per year at the end of which the capital is payable in case of insured event, default=1 (capital payable at the end of death year).
type	A character value, either "EV" (default value) or "ST".
power	The power of the APV. Default is 1 (mean)

Observação: Quando type = "ST", uma amostra de uma unidade do valor presente do benefício é gerada. Quando type = "EV", o valor esperado é retornado.

A seguir apresentamos através de exemplos como utilizar a função Axn em várias situações.

Exemplo 4.1.1.1: Considere um seguro de vida pagável ao final do ano da morte a um segurado com 25 anos na data da aquisição da apólice, cuja soma segurada é de 100.000 reais. Considere a tábua de vida soaLt e taxa de juros de 5% ao ano.

i) Seguro de vida inteira

```
>VPA= 100000*Axn(actuarialtable=soAct, x=25, i=0.05)
>VPA
[1] 11476.49
```

ii) Seguro de vida inteira diferido em 10 anos

```
>VPA= 100000*Axn(actuarialtable=soAct, x=25, m=10, i=0.05)
>VPA
[1] 10333.12
```

iii) Seguro de vida temporário com duração de 30 anos

```
>VPA= 100000*Axn(actuarialtable=soAct, x=25, n=30, i=0.05)
>VPA
[1] 3926.935
```

iv) Seguro de vida temporário com duração de 30 anos diferido por 10 anos

```
>VPA= 100000*Axn(actuarialtable=soAct, x=25, n=30, m=10,
i=0.05)
>VPA
[1] 4791.087
```

Quando o seguro é pago ao final de uma parte do ano em que ocorreu a morte (por exemplo: mês, semestre, etc), é necessário indicar o número de partes do ano através do argumento k.. Por exemplo, se ele for pago ao final do mês da morte, faça k=12; se for pago ao final do semestre da morte faça k=2.

Continuando com o exemplo acima,

v) Seguro de vida temporário com duração de 30 anos diferido por 10 anos pagável ao final do mês da morte.

```
>  VPA=100000*Axn(actuarialtable=soaAct,  x=25,  n=30,  m=10,
k=12, i=0.05)
>VPA
[1] 4899.915
```

O argumento "power" é útil no cálculo da variância do valor presente do benefício (soma segurada a ser paga). Considere o caso de um seguro de vida temporário de duração n pagando uma unidade monetária ao final do ano da morte. Neste caso o valor do benefício, b_k e seu valor presente Z_k são dados por:

$$b_k = \begin{cases} 1 & \text{se K} \le \text{n} \\ 0 & \text{se K} > \text{n} \end{cases} \quad \text{e} \quad Z_k = \begin{cases} 1v^{K+1} & \text{se K} \le \text{n} \\ 0v^{K+1} & \text{se K} > \text{n} \end{cases}$$

onde K = tempo de vida em anos completos.

Para encontrar $\text{VAR(Z)} = \text{E(Z}^2\text{)} - (\text{E(Z)})^2$ podemos usar a regra dos momentos. Pode-se mostrar que nos casos onde o valor do beneficio é 0 ou 1 o valor esperado de Z^h é igual ao valor esperado de Z com fator de desconto $\text{w} = \text{v}^\text{h}$. Por exemplo, para encontrar E(Z)^2 basta encontrar E(Z) considerando fator de desconto $\text{w} = \text{v}^2$.

Exemplo 4.1.1.2: Para o caso do seguro de vida temporário com duração de 30 anos e valor de benefício de uma unidade monetária, o valor esperado de Z^2 é obtido usando a função Axn, como anteriormente especificando o argumento power = 2.

A variância de Z é calculado utilizando o seguinte comando.

```
> VAR = Axn(actuarialtable=soaAct, x=25, n=30, i=0.05, power=2)-
(Axn(actuarialtable=soaAct, x=25, n=30, i=0.05, power=1)^2)
> VAR
[1] 0.01757508
```

No caso do benefício de 100.000 reais basta usar o fato de que $\text{VAR(aZ)} = \text{a}^2\text{VAR(Z)}$, onde a é uma constante. .

```
> 100000^2*VAR
[1] 175750825
```

4.1.2) Seguro dotal puro

Para obter o valor presente atuarial de seguros destas modalidades podemos usar a função *Exn*, cujos argumentos são:

Actuarialtable	An actuarial table object
X	Age of the insured.
N	Length of the pure endowment.
I	Interest rate (overrides the interest rate of the actuarial table object)
Type	A string, eithed "EV" (default value), "ST" (stocastic realization) or "VR" if the value of the variance is needed.
Power	The power of the APV. Default is 1 (mean)

Exemplo 4.2.1: Considere um seguro de vida dotal puro, para um segurado com 25 anos, cuja soma segurada é de 100.000 reais será paga ao final de 30 anos se o segurado estiver vivo. Considere a tábua de vida soaLt e taxa de juros de 5% ao ano. O valor presente atuarial pode ser obtido como:

```
>VPA= 100000*Exn(soAct, x=25, n=30, i=0.05)
>VPA
[1] 20902.21
```

Para encontrar a variância do valor presente do benefício no caso de seguro dotal puro é só usar a regra dos momentos como no exemplo 4.1.1.2.

Para o caso do seguro de vida dotal puro que será pago ao final de 30 anos e valor de benefício de uma unidade monetária, a variância:

```
>VAR=(Exn(actuarialtable=soAct, x=25, n=30, i=0.05, power=2))-
((Exn(actuarialtable=soAct, x=25, n=30, i=0.05, power=1)^2))
> VAR
```

```
[1] 0.004672753
```

No caso do benefício de 100.000 reais:

```
> 100000^2*VAR
[1] 46727535
```

4.3) Seguro dotal misto com duração de n anos:

A função do R para obter o valor presente atuarial para este tipo de seguros é *AExn*, com os seguintes argumentos:

actuarialtable	An actuarial table object.
X	Insured age.
N	Length of the insurance.
I	Rate of interest. When missing the one included in the actuarialtable object is used.
K	Frequency of benefit payment.
Type	Character value, either "EV" or "ST". EV is the default value.
Power	The power of the APV. Default is 1 (mean)

Exemplo 4.3.1: Considere um seguro de vida dotal misto com valor segurado de 100.000 para um segurado com 25 anos na data da emissão da apólice, com duração de 30 anos e pagável ao final do período da morte ou ao final de 30 anos, caso o segurado esteja vivo. Considere a tábua de vida soaLt e taxa de juros de 5% ao ano.

i) Considerando que no caso de morte o seguro é pago ao final do ano.

```
>VPA= 100000*AExn(soAct, x=25,n=30, i=0.05)
>VPA
[1] 24829.15
```

A variância do valor presente do beneficio é obtida através do seguinte comando.

```
>VAR=100000^2*((AExn(actuarialtable=soAct, x=25, n=30, i=0.05,
power=2))-((AExn(actuarialtable=soAct, x=25, n=30, i=0.05,
power=1)^2)))
> VAR
[1] 58315103
```

4.4) Seguro de vida crescente e decrescente

Os argumentos da função *IAxn* são os seguintes.

actuarialtable	The actuarial table used to perform life – contingencies calculations.
X	The age of the insured.
N	The term of life insurance. If missing n is set as $n=\omega - x - m - 1$.
I	Interest rate (overrides the interest rate of the actuarialtable object).
M	The deferring period. If missing, m is set as 0.
K	Number of fractional payments per period. Assumed to be 1 whether missing.
Type	Default value is "EV", where APV is returned. "ST" returns a sample from the underlying present value of benefits distribution.
Power	The power of the APV. Default is 1 (mean).

No caso da função *DAxn*, o argumento n é definido como:

n	Length of the insurance period.

Exemplo 4.4.1) Considere um seguro de vida crescente para uma pessoa com 30 anos e duração igual a 10 anos pagável ao final do ano da morte. Uma unidade monetária será paga se a morte acontece no primeiro ano, 2 unidades se a morte acontece no segundo ano e assim sucessivamente até 10 unidades se a morte ocorre no décimo ano após a realização do contrato. Considerando taxa efetiva de juros de 5% ano e a tábua de vida soaLt, o valor presete atuarial deste seguro é dado por:

```
> IAxn(soAct, x=30, n=10, i=0.05)
[1] 0.083241
```

Exemplo 4.4.2) Considere um seguro de vida decrescente para uma pessoa com 30 anos e duração igual a 10 anos pagável ao final do ano da morte. Cem unidades monetária serão pagas se a morte acontece no primeiro ano, 90 unidades se a morte acontece no segundo ano e assim sucessivamente até 10 unidades se a morte ocorre no décimo ano após a realização do contrato.

Considerando taxa efetiva de juros de 5% ano e a tábua de vida soaLt, o valor presete atuarial deste seguro é dado por:

```
> 10*DAxn(soAct,x=30,n=10,i=0.05)
[1] 0.8119428
```

4.5) Anuidades de vidas

As seguintes funções são disponíveis no pacote "*lifecontingencies"* para cálculo de valores presentes atuariais para anuidades de vida:

axn – para anuidades de vida antecipadas ou postecipadas com ou sem diferimento

axyzn – para anuidades de duas vidas antecipadas ou postecipadas, com ou sem diferimento

Iaxn – para anuidades de vida crescentes

A função *axn* possui os seguintes argumentos:

actuarialtable	An actuarial table object.
X	Age of the annuitant.
N	Number of terms of the annuity, if missing annuity is intended to be paid until death.
I	Interest rate (default value the interest of the life table).
M	Deferring period. Assumed to be 1 whether missing.
K	Number of fractional payments per period. Assumed to be 1 whether missing.
Type	A string, eithed "EV" (default value) or "ST" (stochastic realization).
Power	The power of the APV. Default is 1 (mean)
Payment	Payment type: "advance" default is the annuity due, otherwise annuity immediate.

Para calcular anuidade postecipadas , faça payment = "arrears"

Exemplo 4.5.1) Calcule o valor presente atuarial de uma anuidade de vida inteira pagando uma prestação de 100 unidades monetárias por ano, diferida em 20 anos, para uma pessoa com 40 anos, enquanto estiver viva.

a) Caso antecipado:

```
> VPA=100*(axn(soAct, x = 40, m=20, payment="advance"))
>VPA
 [1] 698.738
```

b) caso postecipado:

```
> VPA=100*(axn(soAct, x = 40, m=20, payment="arrears"))
>VPA
 [1] 650.0592
```

Exemplo 4.5.2) Calcule o valor presente atuarial de uma anuidade de vida de duração igual a 20 anos, pagando uma prestação de 100 unidades monetárias por mês, diferida em 10 anos para uma pessoa com 40 anos

a) caso antecipado:

```
>VPA=100*12*(axn(soAct,n=20,k=12,m=10,x=40,
payment="advance"))
>VPA
 [1] 11837.99
```

b) caso postecipado:

```
>VPA=100*12*(axn(soAct,n=20,k=12,m=10,x=40,payment="arrears"))
>VPA
 [1] 11795.74
```

4.6) Anuidades de vida crescentes

A função *Iaxn* calcula valores presentes atuarias para anuidades de vida crescentes e possui os seguintes argumentos.

actuarialtable	An actuarialtable object.
X	The age of the insured head.
N	The duration of the insurance
I	The interest rate that overrides the one in the actuarialtable object.
M	The deferring period.
Type	Yet only "EV" is implemented.
Power	The power of the APV. Default is 1 (mean)

Exemplo 4.6.1) Calcule o valor presente atuarial de uma anuidade de vida inteira crescente pagando uma prestação de 100 unidades monetárias, diferida em 15 anos, para uma pessoa com 50 anos e taxa de juros de 5% ao ano.

```
> VPA=100*Iaxn(soAct, x=50, i=0.05, m =15)
>VPA
 [1] 3605.468
```

5) Análise Estocástica e Funções de Simulações

5.1) A função rLife

O tempo de vida é uma variável aleatória cuja distribuição pode ser resumida através da tabela de vida. Com a função *rLife* podemos simular o tempo de vida futuro de uma pessoa com x anos, a partir da distribuição do tempo de vida futuro dada pela tabela de vida. Pode-se considerar o tempo de vida futuro em anos completos (K(x)) ou o tempo de vida total (T(x)).

A função *rLife* possui os seguintes argumentos:

N	Number of variates to generate.
object	An object of class lifetable.
X	The attained age of subject x. Default value is 0.
K	Number of periods within the year when it is possible death to happen. Default value is 1.
type	Either "Tx" for continuous future lifetime or "Kx" for curtate future lifetime

Exemplo 5.1.1) Gere uma amostra aleatória de tamanho 5 para o tempo de vida futura de uma pessoa com 45 anos a partir da tabela da SOA.

a) Tempo de vida futuro em anos completos K(x)

```
> rLife(n = 5, object = soAct, x = 45, type = "Kx")
[1] 23 8 37 45 41
```

b) Tempo de vida futuro T(x)

```
> rLife(n = 5, object = soAct, x = 45, type = "Tx")
```

[1] 37.5 26.5 16.5 40.5 28.5

Neste último caso é assumido que o tempo vivido no último ano é igual a 0,5 anos. Uma alternativa seria gerar o tempo de vida futuro vivido no último ano de um U(0,1) e somá-lo aos valores gerados de K(x).

5.2) A função *rLifeContingencies*

Esta função gera amostras aleatórias para o valor presente de alguns benefícios de seguros e anuidades e possui os seguintes argumentos:

N	Size of sample
lifecontingency	A character string, either "Exn" or "Axn" or "axn" or "IAxn" or "DAxn".
Object	An actuarialtable object.
X	Policyholder's age at issue time.
T	The lenght of the insurance. Must be specified according to the present value of benefits definition.
I	The interest rate, whose default value is the actuarialtable interest rate slot value.
M	Deferring period, default value is zero.
K	Fractional payment, default value is 1.
parallel	Uses the parallel computation facility.
Payment	Payment type: "advance" default is the annuity due, otherwise annuity immediate.

Exemplo 5.2.1) Gere uma amostra aleatória de tamanho 1000 para o valor presente de um seguro de vida inteira pagando um unidade monetária a segurado com 40 anos na data da emissão da apólice. Considere taxa efetiva de juros de 5% ano e a tábua de vida soaLt. Obtenha a média dos valores simulados e compare com o valor presente atuarial do seguro.

```
>VPA <-rLifeContingencies(n=1000, lifecontingency="Axn", x=40,
t=getOmega(soAct)-40, object=soAct, i=0.05)

> mean(VPA)                #calcula a média dos valores gerados
[1] 0.2058761
```

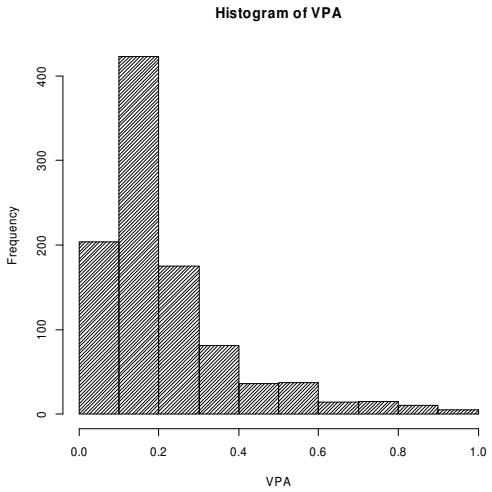
O cálculo do valor presente atuarial obtido usnao a função *Axn* é mostrado abaixo.

```
> VPA= Axn(actuarialtable=soAct, x=40, i=0.05)

> VPA
[1] 0.2079486
```

Para melhor análise, veja a seguir o histograma dos valores gerados:

```
> hist(VPA,density=30)
```



Exemplo 5.2.2) Obtenha por simulação valor presente atuarial um seguro de vida dotal puro para uma amostra de 1000 pessoas e com 25 anos e duração igual a 50 anos. Considere a tábua de vida soaLt.

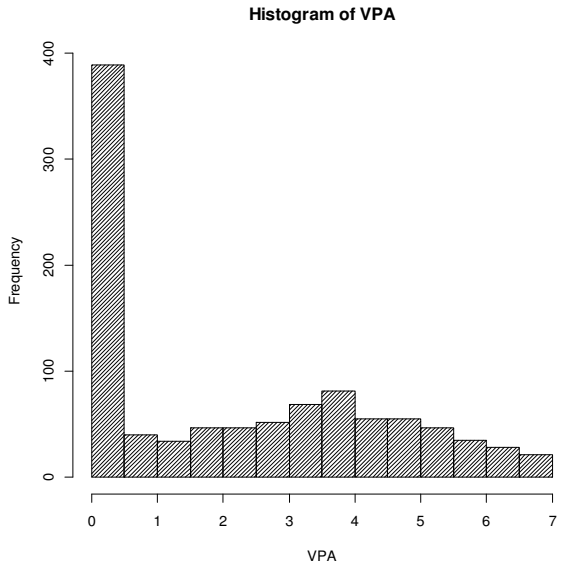
```
>VPA<-rLifeContingencies(n=1000,lifecontingency="Exn",    x=25,
t=50, object=soAct)
> mean(VPA)
[1] 0.04857248
```

O valor presente atuarial também foi obtido usando a função *Exn*.

> VPA= Exn(actuarialtable=soAct, x=25, n=50, i=0.05)
> VPA
[1] 0.04919577

Exemplo 5.2.3) Obtenha por simulação o valor presente atuarial de uma anuidade de vida postecipada, de duração igual a 25 anos, pagando uma prestação mensal e diferida em 15 anos a um pessoa com 60 anos na data da emissão da apólice. Considere uma amostra com 1000 pessoas, taxa efetiva de juros de 5% ano e a tábua de vida soaLt.

```
>VPA<-rLifeContingencies(n=1000,lifecontingency="axn",x=60,
t=25,k=12,m=15, object=soAct, i=0.05,payment="arrears")
> mean(VPA)
[1] 2.203896
> hist(VPA,density=30)
```



Para comparação, valor presente atuarial obtido foi obtido através da função *axn*,

```
> VPA= axn(actuarialtable=soAct, x=60, n=25, k=12, m=15,
i=0.05,payment="arrears")
> VPA
[1] 2.225245
```

6) Usando o Pacote *lifecontingencies* com tabela seletas

As funções do "LifeContingencies" para cálculos de valores presente atuarias não podem ser usadas diretamente com tábuas seletas. Porém usando a função **seleta2padrão**, construída por nós, podemos gerar uma tábua de vida padrão (TV) e uma tábua de comutação (TC) tendo como idade inicial a idade de seleção. As tábuas TV ou TC podem ser usadas como argumentos no cálculo de valores presentes atuariais.

Para exemplificarmos, vamos considerar a tabua seleta apresentado em Dickson et al (2009)[2] cujos valores de l_x constam da matriz TS.

```
TS = matrix(c(20, 99995.08, 99973.75, 99949.71, 22,
21, 99970.04, 99948.40, 99923.98, 23,
22, 99944.63, 99922.65, 99897.79, 24,
23, 99918.81, 99896.43, 99871.08, 25,
24, 99892.52, 99869.70, 99843.80, 26,
25, 99865.69, 99842.38, 99815.86, 27,
26, 99838.28, 99814.41, 99787.20, 28,
27, 99810.20, 99785.70, 99757.71, 29,
28, 99781.36, 99756.17, 99727.29, 30,
29, 99751.69, 99725.70, 99695.83, 31,
30, 99721.06, 99694.18, 99663.20, 32,
31, 99689.36, 99661.48, 99629.26, 33,
32, 99656.47, 99627.47, 99593.83, 34,
33, 99622.23, 99591.96, 99556.75, 35,
34, 99586.47, 99554.78, 99517.80, 36,
35, 99549.01, 99515.73, 99476.75, 37,
36, 99509.64, 99474.56, 99433.34, 38,
37, 99468.12, 99431.02, 99387.29, 39,
38, 99424.18, 99384.82, 99338.26, 40,
39, 99377.52, 99335.62, 99285.88, 41,
40, 99327.82, 99283.06, 99229.76, 42,
41, 99274.69, 99226.72, 99169.41, 43,
42, 99217.72, 99166.14, 99104.33, 44,
43, 99156.42, 99100.80, 99033.94, 45,
44, 99090.27, 99030.10, 98957.57, 46,
45, 99018.67, 98953.40, 98874.50, 47,
46, 98940.96, 98869.96, 98783.91, 48,
47, 98856.38, 98778.94, 98684.88, 49,
48, 98764.09, 98679.44, 98576.37, 50,
49, 98663.15, 98570.40, 98457.24, 51,
50, 98552.51, 98450.67, 98326.19, 52,
51, 98430.98, 98318.95, 98181.77, 53,
52, 98297.24, 98173.79, 98022.38, 54,
53, 98149.81, 98013.56, 97846.20, 55,
54, 97987.03, 97836.44, 97651.21, 56,
55, 97807.07, 97640.40, 97435.17, 57,
56, 97607.84, 97423.18, 97195.56, 58,
57, 97387.05, 97182.25, 96929.59, 59,
58, 97142.13, 96914.80, 96634.14, 60,
```

[2] Dickson, D.C, Hardy, M,R. e Waters, H.R. Actuarial Mathematics for life contingent risks. Cambridge University Press, 2009

```
59, 96870.22, 96617.70, 96305.75, 61,
60, 96568.13, 96287.48, 95940.60, 62,
61, 96232.34, 95920.27, 95534.43, 63,
62, 95858.91, 95511.80, 95082.53, 64,
63, 95443.51, 95057.36, 94579.73, 65,
64, 94981.34, 94551.72, 94020.33, 66,
65, 94467.11, 93989.16, 93398.05, 67,
66, 93895.00, 93363.38, 92706.06, 68,
67, 93258.63, 92667.50, 91936.88, 69,
68, 92551.02, 91894.03, 91082.43, 70,
69, 91764.58, 91034.84, 90133.96, 71,
70, 90891.07, 90081.15, 89082.09, 72,
71, 89921.62, 89023.56, 87916.84, 73,
72, 88846.72, 87852.03, 86627.64, 74,
73, 87656.25, 86555.99, 85203.46, 75,
74, 86339.55, 85124.37, 83632.89, 76,
75, 84885.49, 83545.75, 81904.34, 77,
76, 83282.61, 81808.54, 80006.23, 78,
77, 81519.30, 79901.17, 77927.35, 79,
78, 79584.04, 77812.44, 75657.16, 80,
79, 77465.70, 75531.88, 73186.31, 81,
80, 75153.97, 73050.22, 70507.19, 82), ncol=5, byrow=T)
```

Com a função **seleta2padrao** podemos gerar uma tábua de vida padrão e uma tábua de comutação com os valores de l_x iniciados na idade de seleção [x]. Esta função possui os argumentos: a tábua seleta TS, a idade [x] e a taxa efetiva de juros. Esta função só vale para a tábua seleta do Dickson e deve ser adaptada quando for utilizar uma outra tábua

```
> seleta2padrao=function(idade,TS,i){
x = idade:82
x = x-idade
a = idade-19
b = a+1
lx = c(TS[a,2:4],TS[b:61,4])
TV = new("lifetable",x=x,lx=lx,name="TV")
TC = new("actuarialtable", x=x,lx=lx,interest=i,name="TC")
out = list(TV,TC)
names(out)=c("TV","TC")
return(out)}
```

Considerando [x] = 60 e i = 0.05 (valores de l_x em negrito marcados em negrito na tabela TS)

```
> res = seleta2padrao(idade=60,TS,i=0.05)
```

```
> res
$TV
Life table TV
    x       lx        px         ex
1   0 96568.13 0.9970938 19.9067123
2   1 96287.48 0.9963975 18.9647346
3   2 95940.60 0.9957664 18.0333029
4   3 95534.43 0.9952698 17.1099725
5   4 95082.53 0.9947120 16.1912913
6   5 94579.73 0.9940854 15.2773666
7   6 94020.33 0.9933814 14.3682635
8   7 93398.05 0.9925910 13.4639945
9   8 92706.06 0.9917030 12.5644944
10  9 91936.88 0.9907061 11.6696139
11 10 91082.43 0.9895867 10.7790873
12 11 90133.96 0.9883299  9.8925144
13 12 89082.09 0.9869194  9.0093240
14 13 87916.84 0.9853361  8.1287336
15 14 86627.64 0.9835598  7.2497061
16 15 85203.46 0.9815668  6.3708853
17 16 83632.89 0.9793317  5.4905263
18 17 81904.34 0.9768253  4.6064011
19 18 80006.23 0.9740160  3.7156858
20 19 77927.35 0.9708679  2.8148097
21 20 75657.16 0.9673415  1.8992717
22 21 73186.31 0.9633932  0.9633932

$TC
Actuarial table  TC interest rate  5 %
    x       lx       Dx         Nx         Cx       Mx        Rx
1   0 96568.13 96568.13 1275293.32   267.2857 35839.88 703028.92
2   1 96287.48 91702.36 1178725.19   314.6304 35572.59 667189.05
3   2 95940.60 87020.95 1087022.83   350.8649 35257.96 631616.46
4   3 95534.43 82526.23 1000001.88   371.7792 34907.10 596358.49
5   4 95082.53 78224.63  917475.64   393.9570 34535.32 561451.40
6   5 94579.73 74105.69  839251.01   417.4329 34141.36 526916.08
7   6 94020.33 70159.42  765145.32   442.2428 33723.93 492774.72
8   7 93398.05 66376.25  694985.90   468.3661 33281.68 459050.80
9   8 92706.06 62747.11  628609.65   495.8203 32813.32 425769.11
10  9 91936.88 59263.33  565862.54   524.5582 32317.50 392955.80
11 10 91082.43 55916.71  506599.21   554.5508 31792.94 360638.30
12 11 90133.96 52699.46  450682.50   585.7206 31238.39 328845.36
13 12 89082.09 49604.24  397983.04   617.9570 30652.67 297606.97
14 13 87916.84 46624.18  348378.79   651.1336 30034.71 266954.30
15 14 86627.64 43752.84  301754.62   685.0549 29383.58 236919.59
16 15 85203.46 40984.32  258001.77   719.4962 28698.52 207536.01
17 16 83632.89 38313.19  217017.45   754.1606 27979.03 178837.49
18 17 81904.34 35734.59  178704.26   788.7039 27224.87 150858.47
19 18 80006.23 33244.24  142969.67   822.6834 26436.16 123633.60
20 19 77927.35 30838.50  109725.43   855.6107 25613.48  97197.44
21 20 75657.16 28514.39   78886.93   886.8927 24757.87  71583.96
22 21 73186.31 26269.67   50372.54   915.8568 23870.97  46826.09
23 22 70507.19 24102.87   24102.87 22955.1179 22955.12  22955.12
```

A seguir indicamos como obter alguns valores presentes atuariais, considerando idade de seleção igual a 60 anos.

a) Valor presente atuarial de um seguro de vida inteira pagando 1 unidade monetária ao final do ano da morte para uma pessoa com [60] anos.

```
> Axn(res$TC, x = 0)
[1] 0.0893313
```

Observação: A tábua de vida e de comutação construídas acima tem idade inicial igual a zero embora a idade de seleção considerada como exemplo seja igual a 60. Isto acontece porque as tabelas construídas usando as classes *"lifetable"* e "*actuarialtable*" do pacote "*Lifecontigencies"* têm necessariamente idade inicial igual a zero. Portanto o valor x nas tábuas acima pode ser entendido como o tempo desde a seleção. Um idade igual a 60 corresponde a um tempo desde a seleção igual a zero. Este é o motivo de termos especificado x = 0 como argumento em Axn(res$TC, x = 0) e não x = 60. Isto vale para os exemplos posteriores que fazem uso de tábuas seletas.

b) Valor presente atuarial de um seguro de vida dotal misto pagando uma unidade monetária, com duração de 10 anos para uma pessoa com [60] anos na data da aquisição da apólice.

```
> AExn(res$TC, x = 0, n=10)
[1] 0.6209466
```

c) Anuidade de vida inteira antecipada pagando 1 unidade monetária por mês por no máximo 10 anos com pagamentos iniciando ao [60] anos na data da aquisição da apólice

```
> axn(res$TC, x = 0, n = 10, k=12)
[1] 7.765309
```

Capítulo 2 - Cálculos Atuariais para o caso de múltiplas vidas: status de vida conjunta e status do último sobrevivente usando o software R

1) Introdução

Nesta seção descreveremos como utilizar as funções do pacote *LifeContingencies* para cálculos atuariais para o caso de múltiplas vidas. Vamos deter-nos ao caso de 2 vidas.

Considere que um casal faz um seguro de vida em grupo que deverá pagar uma soma ao sobrevivente ao final do ano da primeira morte observada entre os seus membros. Neste caso a variável aleatória subjacente é o tempo até a ocorrência da primeira morte. Este tempo é chamado na literatura atuarial de tempo de vida futuro do *status de vida conjunta*, isto é, o tempo de vida comum dos elementos do casal.

Podemos considerar também que o seguro, realizado pelo casal, deverá pagar uma soma a um beneficiário do casal ao final do ano da última morte observada entre os elementos do casal. Neste caso, a variável aleatória subjacente é o tempo até a morte do último sobrevivente, que corresponde ao máximo dos tempos de sobrevivência dos elementos do casal. Este tempo é chamado na literatura atuarial de tempo de vida futuro do *status de vida do último sobrevivente.*

O pacote *LifeContingencies* possui um conjunto de funções para obter a distribuição do tempo de vida futuro dos status de vida conjunto e do último sobrevivente, a partir das distribuições dos tempos de vida futuros individuais dos elementos pertencentes ao grupo. Além destas funções, o pacote possui um conjunto de funções para cálculos de valores atuariais de seguros e anuidades envolvendo cada um destes status. Todas estas funções utilizam o pressuposto que os tempos de vida futuro dos elementos do grupo são variáveis aleatórias independentes.

2) Funções para cálculos de probabilidades relacionadas ao tempo de vida futuro do status de vida conjunta e do status do último sobrevivente

A função *pxyt* retorna a probabilidade de que o status em questão esteja ativo no tempo t. No caso do status de vida conjunta, retorna a probabilidade de que ambas as pessoas com idades atuais x e y estejam vivas t anos depois. No caso do status de vida do último sobrevivente retorna à probabilidade de que pelo menos uma das pessoas atualmente com x e y anos esteja viva t anos depois.

A função *qxyt* retorna à probabilidade de que status em questão falhe até o tempo t. No caso do status de vida conjunta, retorna a probabilidade de que pelo menos uma das 2 vidas atualmente com idades x e y morra antes do tempo t. No caso do status de vida do último sobrevivente, retorna a probabilidade de que as duas vidas x e y falhem dentro de t anos.

As funções *qxyt* e *pxyt* têm como argumento um objeto do tipo *lifetable* ou *actuarialtable*. Os demais argumentos são descritos na tabela abaixo. Quando o status não é especificado, o status de vida conjunta é assumido.

objectx	lifetable for life X.
objecty	lifetable for life Y.
x	Age of life X.
y	Age of life Y.
t	Time until survival has to be evaluated.
status	Either "joint" or "last".

Para resolver os exemplos que se seguem, criamos duas tábuas atuariais considerando taxa de juros de 3% ao ano e utilizando os dados das tábuas de vida do Reino unido (United Kingdom - UK), AF92Lt (Tábua com dados femininos) e AM92Lt (Tábua com dados Masculinos).

```
> data(AF92Lt)
>AF92ACT=new("actuarialtable",x=AF92Lt@x,lx=AF92Lt@lx,
interest=0.03, name="UK Female ")   #tabela Fem. atuarial com
juros de 3%

> data(AM92Lt)
> AM92ACT=new("actuarialtable", x =AF92Lt@x,lx=AF92Lt@lx,
interest=0.03, name="UK male")   #tabela Masc. atuarial com
juros de 3%
```

Exemplo 2.1) Calcule a probabilidade de que duas vidas seguradas (marido e mulher) com idades atuais 30 e 40 anos respectivamente, estejam vivas 20 anos após o início da apólice.

```
> pxyt(AF92ACT, AM92ACT, 30, 40, 20, status = "joint")
[1] 0.9682534
```

Se quisermos saber o qxyt neste mesmo caso, basta calcular o complementar de pxyt.

Exemplo 2.2) Calcule a probabilidade de que pelo menos uma das pessoas seguradas (marido e mulher) atualmente com 30 e 40 anos respectivamente, estejam vivas 20 anos depois do início da apólice.

```
> pxyt(AF92ACT, AM92ACT, 30, 40, 20, status = "last")
[1] 0.9997895
```

Exemplo 2.3) Calcule a probabilidade de que pelo menos uma das duas vidas seguradas (marido e mulher) com idades atuais 58 e 70 anos respectivamente, morram antes de 10 anos depois do início da apólice.

```
> qxyt(AF92ACT, AM92ACT, 58, 70, 10, status = "joint")
[1] 0.1447231
```

Exemplo 2.4) Calcule a probabilidade de que duas pessoas seguradas (marido e mulher) atualmente com 58 e 70 anos respectivamente, morram dentro de 10 anos depois do início da apólice.

```
> qxyt(AF92ACT, AM92ACT, 58, 70, 10, status = "last")
[1] 0.004016466
```

Para o caso de 3 ou mais vidas, pode-se usar as funções *pxyzt* e *qxyzt* para obter as respectivamente as probabilidades de sobrevivência e morte dos status de vida conjunta e do último sobrevivente.

3) Funções atuariais

Nesta seção, mostraremos através de exemplos como calcular valores presentes atuariais para seguros e anuidades no caso discreto para o status de 2 vidas. Para isto podemos usar as funções `Axyn` e `axyn`. Ambas têm como argumento um objeto do tipo *lifetable* ou *actuarialtable*.

tablex	Life X lifetable object.
tabley	Life Y lifetable object.
x	Age of life X.
y	Age of life Y.
n	Insured duration. Infinity if missing.
i	Interest rate. Default value is those implied in actuarialtable.
m	Deferring period. Default value is zero.
k	Fractional payments or periods where insurance is payable.
status	Either "joint" or "last" survival status.
type	"EV" (expected value) or "ST" (stochastic)
payment	Payment type: "advance" default is the annuity due, otherwise annuity immediate.

Exemplo 3.1) Um casal formado por uma mulher com 30 anos e um homem com 40 anos contratou um seguro de vida que deverá pagar uma soma de 200000 reais. Calcule o valor presente atuarial deste seguro, assumindo que:

a) O benefício será pago ao sobrevivente do casal ao final do ano da primeira morte entre os elementos do casal.

```
> 200000*Axyn(AF92ACT,30, AM92ACT,40,status = "joint")
[1] 51475.95
```

b) O benefício será pago a um beneficiário ao final do ano da morte do último sobrevivente do casal.

```
>200000 * Axyn(AF92ACT, 30, AM92ACT, 40, status = "last")
[1] 29950.22
```

c) Por este seguro, prêmios deverão ser pagos no início de cada ano. No primeiro caso, os prêmios serão pagos enquanto ambos os elementos do casal estiverem vivos. Encontre o valor do prêmio pelo princípio da equivalência. .

```
> P = 200000 * Axyn(tablex=AF92ACT, x=30, tabley=AM92ACT,
y=40, status = "joint")/ axyn(tablex=AF92ACT, x=30, tabley
=AM92ACT, y=40, status = "joint")
> P
[1] 2018.932
```

d) No segundo caso, um prêmio P deverá ser pago enquanto pelo menos um dos elementos estiver vivo. Após a morte do primeiro, o prêmio P deverá ser reduzido pela metade. Encontre o valor de P.

```
> P = 2*200000 * Axyn(tablex=AF92ACT, x=30, tabley=AM92ACT,
y=40, status = "joint")/ axyn(tablex=AF92ACT, x=30, tabley
=AM92ACT, y=40, status = "last")
> P
[1] 3544.79
```

As funções Axyn e axyn são casos particulares das funções Axyzn e axyzn que consideram as situações de 3 vidas ou mais.

4) Funções para simulação estocástica

A função *rLifexyz é* útil para simular os tempos de vida futuro de mais de uma pessoa assumindo independência. Os argumentos desta função são:

tablesList	A list whose elements are either lifetable or actuarialtable class objects.
x	A vector of the same size of tableList that contains the initial ages.
n	The size of sampled life duration matrix.
status	Either "joint" (for the joint-life status model) or "last".
type	"Tx" for continuous, "Kx" for curtate.
fractional	Fractional lives assumption.
...	Options to be passed to pxt.
k	Fractional frequency option.

Exemplo 4.1) Gere uma amostra de 10 tempos de falha do status de vida conjunta de um casal considerando que o homem tem 50 anos e a mulher 40 anos. Considere que o tempo de vida futuro do homem e da mulher são descritos respectivamente pelas tábuas AF92ACT e AM92ACT. Gere também uma amostra de 10 tempos de falha do status de vida do último sobrevivente.

Para isto gere uma amostra de 10 tempos de vida futuro do homem e outra de 10 tempos de vida futuro da mulher usando a função *rLifexyz, como mostrado abaixo.*

```
> tables=list(AF92ACT,AM92ACT)
> x=c(50,40)
> Tempos = rLifexyz(tablesList=tables,x=x,type="Tx",n=10)
> Tempos
      [,1] [,2]
 [1,] 60.5 26.5
 [2,] 50.5 45.5
 [3,] 52.5 60.5
 [4,] 59.5 37.5
 [5,] 32.5 54.5
 [6,] 43.5 55.5
 [7,] 36.5 55.5
 [8,] 42.5 18.5
 [9,] 11.5 41.5
[10,] 60.5 38.5
```

Utilize a função abaixo para obter o tempo de vida do status de vida conjunta como o mínimo dos tempos de vida dos 2 indivíduos e o tempo de vida do último sobrevivente como o máximo dos 2 tempos de vida.

```
> Tstatus=function(tempox,tempoy){
 n=length(tempox)
 Tjoint=rep(1,n)
 Tlast=rep(1,n)
 for (i in 1:n){
 Tjoint[i]=min(tempox[i],tempoy[i])
 Tlast[i]=max(tempox[i],tempoy[i])}
```

```
 T=list(Tjoint,Tlast)
 names(T)=list("Tjoint", "Tlast")
 return(T)}

> Tstatus(Tempos[,1],Tempos[,2])
$Tjoint
 [1] 26.5 45.5 52.5 37.5 32.5 43.5 36.5 18.5 11.5 38.5
$Tlast
 [1] 60.5 50.5 60.5 59.5 54.5 55.5 55.5 42.5 41.5 60.5
```

Podemos construir a função acima para encontrar as funções de mim e max, ou simplesmente utilizar a função apply, veja abaixo.

```
> Tjoint=apply(Tempos,1,min)
> Tjoint
 [1] 26.5 45.5 52.5 37.5 32.5 43.5 36.5 18.5 11.5 38.5
> Tlast=apply(Tempos,1,max)
> Tlast
 [1] 60.5 50.5 60.5 59.5 54.5 55.5 55.5 42.5 41.5 60.5
```

A função *rLifeContingenciesXyz* gera amostras aleatórias, para dois ou mais indivíduos, do valor presente de alguns benefícios de seguros e anuidades e possui os seguintes argumentos:

n	Sample size
lifecontingency	Either "Axyz" or "axyz"
tablesList	List of tables
x	Ages vector
t	Term
i	Interest rate
m	Deferral period
k	Frequency of payments
status	Either "joint" (default) or "last"
parallel	Use parallel computation
payment	Payment type: "advance" default is the annuity due, otherwise annuity immediate.

Exemplo 4.2) Gere uma amostra de 10 valores do valor presente de um seguro de vida pagando uma unidade monetária ao final do ano da morte do primeiro entre 2 indivíduos de um casal, considerando que o homem tem 50 anos e a mulher tem 40 anos. Considere que o tempo de vida futuro do homem e da mulher são descritos pelas tábuas de vida AF92ACT e AM92ACT.

```
> tables=list(AF92ACT,AM92ACT)
> x=c(50,40)
>    rLifeContingenciesXyz(    n=10,    lifecontingency="Axyz",
tablesList=tables, x=x, status = "joint")
[1]   0.4636947   0.3349829   0.2349503   0.4370768   0.3349829
0.8374843 0.4119868 0.4119868 0.3252262 0.2976280
```

Exemplo 4.2) Gere uma amostra de 10 valores do valor presente de um seguro de vida pagando uma unidade monetária ao final do ano da morte do último sobrevivente de um casal considerando que o homem tem 50 anos e a mulher tem 40 anos. Considere que o tempo de vida futuro do homem e da mulher são descritos pelas tábuas de vida AF92ACT e AM92ACT.

```
> tables=list(AF92ACT,AM92ACT)
> x=c(50,40)
>    rLifeContingenciesXyz(    n=10,    lifecontingency="Axyz",
tablesList=tables, x=x, status = "last")

[1]   0.2349503   0.1697331   0.0000000   0.2150128   0.2026702
0.1647894 0.0000000 0.1748251 0.1599897 0.3157535
```