An introduction to lifecontingencies package

Giorgio A. Spedicato, Ph.D

September 17, 2011

1 Overview

I've decided to submit to the cran package to perform life contingencies calculation in order to fill a current lack within the CRAN archive. The structure of the vignette document (uncompleted yet) is:

1. Section 2 provides an overview of R usage within actuarial fields and describes the package structure.
2. Section 3 gives a wide choice of lifecontingencies packages example.
3. Finally section will provide a discussion of results and further potential developments.

The accuracy of calculation have been verified by checkings with numerical examples reported in [Bowers et al., 1997]. The package numerical results are identical to those reported in the [Bowers et al., 1997] for most function, with the exception of fractional payments annuities where the accuracy leads only to the 5th decimal. The reason of such inaccuracy is due to the fact that the package calculates the APV by directly sum of fractional survival probabilities, while the formulas reported in [Bowers et al., 1997] uses an analytical formula. This package and functions herein are provided as is, without any guarantee regarding the accuracy of calculations. The author disclaims any liability arising by eventual losses due to direct or indirect use of this package.

2 Lifecontingencies package description

2.1 Current R actuarial packages

R [R Development Core Team, 2011] represents a powerful enviromnent for statistical analysis and simulation. Thus many packages dedicated to P&C actuarial software have been available from some years. Among those we shall cite:

- The package **actuar** [Dutang et al., 2008] provides functions to fit relevant loss distribution and perform credibility analysis. It represents the computational side of the classical book [?].
- The package **ChainLadder** [Gesmann and Zhang, 2011] provides functions to estimate non-life loss reserve.

The choice of statistical functions to perform rate - making is more wide as R provides a wide range of statistical function to perform classification and predictive modelling task (e.g. GLMs, data - mining techniques) performed by pricing actuaries.

Life actuaries works more with demographic and financial data. While R has a dedicated view to packages dedicated to financial analysis and few packages exists to perform demographic analysis (see for examples [with contributions from Heather Booth et al., 2011]) as of August 2011 no package exists to perform life contingencies calculation.
This package aims to represent the R computational support of the concepts developed in the classical life contingencies book [Bowers et al., 1997].

2.2 The structure of the package

The package contains R function, classes and methods to perform classical financial mathematics calculations, working with lifetable objects and classical life contingencies calculations.
Functions are available to evaluate actuarial present values for life contingencies functions as $\ddot{a}_{x:\overline{n}|}^{(m)}$, $A_{x:\overline{n}|}^{1}$, $A_{x:\overset{1}{\overline{n}|}}$, $(DA)_{x:\overline{n}|}^{1}$ and $(IA)_{x:\overline{n}|}^{1}$.
Some functions allows to return the simulated value of most life contingencies functions.
Demos and vignettes (like this document) are also available.

The package is load within the R working environment as follows:

```
> library(lifecontingencies)
```

3 Examples

3.1 Classical financial mathematics example

Two examples will show classical financial mathematics applications of package lifecontingencies: present value analysis, the amortization of a loan and a savings account projection.
Functions to switch between nominal and effective interest rates have been also developed.

The present value of a cash flow series,$\bar{c_T}$ is $PV\left(\bar{c}_T\right) = \sum\limits_{t \in T} c_t {v_{t_i}}^t$ and if we consider probabilities the latter formula became an actuarial present value, $PV\left(\bar{c}_T\right) = \sum\limits_{t \in T} p_{t_i} c_t {v_{t_i}}^t$.

3.1.1 Present value analysis

```
> capitals = c(-1000, 200, 500, 700)
> times = c(-2, -1, 4, 7)
> presentValue(cashFlows = capitals, timeIds = times, interestRates = 0.03)

[1] 158.5076

> presentValue(cashFlows = capitals, timeIds = times, interestRates = c(0.04,
+     0.02, 0.03, 0.057))

[1] 41.51177

> presentValue(cashFlows = capitals, timeIds = times, interestRates = c(0.04,
+     0.02, 0.03, 0.057), probabilities = c(1, 1, 1, 0.5))

[1] -195.9224
```

3.1.2 Loan amortization

```
> capital = 1e+05
> interest = 0.05
> payments_per_year = 2
> effectiveRate = (1 + interest)^(1/payments_per_year) - 1
> years = 10
> installment = capital/annuity(i = effectiveRate, periods = years *
+     payments_per_year)
> installment

[1] 6396.251

> balance_due = numeric(years * payments_per_year)
> balance_due[1] = capital * (1 + effectiveRate) - installment
> for (i in 2:length(balance_due)) {
+     balance_due[i] = balance_due[i - 1] * (1 + effectiveRate) -
+         installment
+     cat("Payment ", i, " balance due:", round(balance_due[i]),
+         "\n")
+ }
```

```
Payment  2  balance due: 92050
Payment  3  balance due: 87926
Payment  4  balance due: 83702
Payment  5  balance due: 79372
Payment  6  balance due: 74936
Payment  7  balance due: 70390
Payment  8  balance due: 65733
Payment  9  balance due: 60960
Payment  10  balance due: 56069
Payment  11  balance due: 51057
Payment  12  balance due: 45922
Payment  13  balance due: 40659
Payment  14  balance due: 35267
Payment  15  balance due: 29742
Payment  16  balance due: 24080
Payment  17  balance due: 18279
Payment  18  balance due: 12334
Payment  19  balance due: 6242
Payment  20  balance due: 0
```

3.1.3 Saving account projection

```
> cumulatedSavings <- function(amount, rate, periods) {
+     service_charge = 1
+     service_fee = (0.01 * min(100, amount) + 0.005 * max(0, min(50,
+         amount - 100)))
+     invested_amount = amount - service_charge - service_fee
+     out = invested_amount * accumulatedValue(interestRates = rate,
+         periods = periods)
+     return(out)
+ }
> savings_sequence = seq(from = 50, to = 300, by = 10)
> periods = 30 * 12
> yearly_rate = 0.025
> montly_effective_rate = (1 + yearly_rate)^(1/12) - 1
> cumulated_value = sapply(savings_sequence, cumulatedSavings,
+     montly_effective_rate, periods)
```

3.2 Functions to switch between nominal and effective interest rates

```
> nominal2Real(0.04, 4)

[1] 0.04060401

> real2Nominal(0.04, 4) * 100

[1] 3.941363
```

3.3 Working with lifetable and actuarial table objects

Lifetable objects represent the basic class designed to handle life table calculations needed to evaluate life contingencies. Actuarialtable class inherits from lifetable class.
Both have been designed using the S4 class framework. To build a lifetable class object three items are needed:

1. The years sequence, that is an integer sequence $0, 1, \ldots, \omega$. It shall starts from zero and going to the ω age (the age x that $p_x = 0$).
2. The l_x vector, that is the number of subjects living at the beginning of age x.
3. The name of the life table.

```
> x_example = seq(from = 0, to = 9, by = 1)
> lx_example = c(1000, 950, 850, 700, 680, 600, 550, 400, 200,
+     50)
> fakeLt = new("lifetable", x = x_example, lx = lx_example, name = "fake lifetable")
```

A print (or show - equivalent) method is also available, reporting the x, lx, px and ex in tabular form.

```
> print(fakeLt)
```

```
Life table fake lifetable

   x   lx        px       ex
1  0 1000 0.9500000 5.980000
2  1  950 0.8947368 5.242105
3  2  850 0.8235294 4.741176
4  3  700 0.9714286 4.542857
5  4  680 0.8823529 3.647059
6  5  600 0.9166667 3.000000
7  6  550 0.7272727 2.181818
8  7  400 0.5000000 1.625000
9  8  200 0.2500000 1.250000
10 9   50 0.0000000 1.000000
```

An actuarialtable class inherits from the lifecontingencies class, but contains and additional slot: the interest rate slot.

```
> irate = 0.03
> fakeAct = new("actuarialtable", x = fakeLt@x, lx = fakeLt@lx,
+     interest = irate, name = "fake actuarialtable")
```

Currently just one method, **getOmega** has been implemented for lifetable and actuarialtable S4 classes, that provides the ω age.

```
> getOmega(fakeAct)
```

```
[1] 9
```

3.4 Survival distribution and life tables

After a lifecontingencies table has been created, basic probability calculations may be performed. Below calculations for ${}_tp_x$, ${}_tq_x$ and $\mathring{e}_{x:\overline{n}|}$.

```
> pxt(fakeLt, 2, 1)

[1] 0.8235294

> qxt(fakeLt, 3, 2)

[1] 0.1428571

> exn(fakeLt, 5, 2)

[1] 1.583333
```

Fractional survival probabilities can also be calculated according with linear interpolation, constant force of mortality and hyperbolic assumption.

```
> data(soa08Act)
> pxt(soa08Act, 80, 0.5, "linear")

[1] 0.9598496

> pxt(soa08Act, 80, 0.5, "constant force")

[1] 0.9590094

> pxt(soa08Act, 80, 0.5, "hyperbolic")

[1] 0.9581701
```

Analysis of two heads survival probabilities are possible:

```
> pxyt(fakeLt, fakeLt, x = 6, y = 7, t = 2)

[1] 0.04545455

> pxyt(fakeLt, fakeLt, x = 6, y = 7, t = 2, status = "last")

[1] 0.4431818
```

If we want a more real example, lets use the IPS55 Italian population life table

```
> lxIPS55M <- with(demoita, IPS55M)
> pos2Remove <- which(lxIPS55M %in% c(0, NA))
> lxIPS55M <- lxIPS55M[-pos2Remove]
> xIPS55M <- seq(0, length(lxIPS55M) - 1, 1)
> lxIPS55F <- with(demoita, IPS55F)
> pos2Remove <- which(lxIPS55F %in% c(0, NA))
> lxIPS55F <- lxIPS55F[-pos2Remove]
> xIPS55F <- seq(0, length(lxIPS55F) - 1, 1)
> ips55M = new("lifetable", x = xIPS55M, lx = lxIPS55M, name = "IPS 55 Males")
> ips55F = new("lifetable", x = xIPS55F, lx = lxIPS55F, name = "IPS 55 Females")
> getOmega(ips55M)
```

```
[1] 117
```

```
> getOmega(ips55F)
```

```
[1] 118
```

```
> exyt(ips55M, ips55F, x = 65, y = 63, status = "joint")
```

```
[1] 19.1983
```

3.5 Classical actuarial mathematics examples

We will now show some classical actuarial mathematics example regarding the evaluation of actuarial present value (APV) of some life insurance benefits, benefit premiums and benefit reserves for classical life insurances.
For all reported examples, we will use the SOA illustrative life table and the insured amount is considered equal to 1 unless otherwise specified.

3.5.1 Life insurance examples

Following examples show APV for a series of life insurances.

```
> Axn(soa08Act, 30, 10, i = 0.04)
```

```
[1] 0.01577283
```

```
> Axn(soa08Act, x = 30, n = 10, i = 0.04, k = 12)
```

```
[1] 0.01605995
```

```
> Axn(soa08Act, 40)
```

```
[1] 0.1613242
```

```
> Axn(actuarialtable = soa08Act, x = 40, n = 10, m = 5, i = 0.05)
```

```
[1] 0.03298309
```

```
> DAxn(soa08Act, 50, 5)
```

```
[1] 0.08575918
```

```
> IAxn(soa08Act, 40, 10)
```

```
[1] 0.1551456
```

while following examples evaluate pure endowments

```
> Exn(soa08Act, x = 30, n = 35, i = 0.06)
```

```
[1] 0.1031648
```

```
> Exn(soa08Act, x = 30, n = 35, i = 0.03)
```

```
[1] 0.2817954
```

3.5.2 Life annuities examples

Following examples show annuities (immediate, due, with fractional payments provision, deferred, etd ...) APV calculations.

```
> axn(soa08Act, x = 65, m = 1)
```

```
[1] 8.896928
```

```
> axn(soa08Act, x = 65)
```

```
[1] 9.896928
```

```
> 12 * 1000 * axn(soa08Act, x = 65, k = 12)
```

```
[1] 113179.1
```

```
> 12 * 1000 * axn(soa08Act, x = 65, k = 12, n = 20)
```

```
[1] 108223.5
```

```
> 12 * 1000 * axn(soa08Act, x = 65, k = 12, n = 20, m = 1/12)
```

```
[1] 107321.1
```

3.5.3 Benefit premiums examples

Lifecontingencies package functions can be used to evaluate benefit premium for life contingencies, using the formula ${}_hP^1_{x:\overline{n}|} = APV\ddot{a}_{x:\overline{h}|}$.

```
> data(soa08Act)
> Pa = 1e+05 * Axn(soa08Act, x = 30, n = 35, i = 0.025)/axn(soa08Act,
+     x = 30, n = 15, i = 0.025)
> Pa
```

```
[1] 921.5262
```

```
> Pm = 1e+05 * Axn(soa08Act, x = 30, n = 35, i = 0.025)/axn(soa08Act,
+     x = 30, n = 15, i = 0.025, k = 12)
> Pm
```

```
[1] 932.9836
```

```
> APV = 10000 * (Axn(soa08Act, 50, 20) + Exn(soa08Act, 50, 20))
> P = APV/axn(soa08Act, 50, 20, k = 2)
```

3.5.4 Benefit reserves examples

Now we will evaluate the benefit reserve for a 20 year life insurance of 100,000, whith benefits payable at the end of year of death, whith level benefit premium payable at the beginning of each year. Assume 3% of interest rate and SOA life table to apply.

The benefit premium is P, determined from equation

$$P\ddot{a}_{40:\overline{20}|} = 100000A^{\ 1}_{40:\overline{20}|}$$

. The benefit reserve is ${}_kV^{\ 1}_{40+t:\overline{n-t}|} = 100000A^{\ 1}_{40+t:\overline{20-t}|} - P\ddot{a}_{40+t:\overline{20-t}|}$ for $t = 0 \ldots 19$.

```
> P = 1e+05 * Axn(soa08Act, x = 40, n = 20, i = 0.03)/axn(soa08Act,
+     x = 40, n = 20, i = 0.03)
> for (t in 0:19) cat("At time ", t, " benefit reserve is ", 1e+05 *
+     Axn(soa08Act, x = 40 + t, n = 20 - t, i = 0.03) - P * axn(soa08Act,
+     x = 40 + t, n = 20 - t, i = 0.03), "\n")
```

```
At time  0  benefit reserve is  0
At time  1  benefit reserve is  306.9663
At time  2  benefit reserve is  604.0289
At time  3  benefit reserve is  889.0652
At time  4  benefit reserve is  1159.693
At time  5  benefit reserve is  1413.253
At time  6  benefit reserve is  1646.808
At time  7  benefit reserve is  1857.044
At time  8  benefit reserve is  2040.286
At time  9  benefit reserve is  2192.436
At time  10  benefit reserve is  2308.88
At time  11  benefit reserve is  2384.513
At time  12  benefit reserve is  2413.576
At time  13  benefit reserve is  2389.633
At time  14  benefit reserve is  2305.464
At time  15  benefit reserve is  2152.963
At time  16  benefit reserve is  1922.973
At time  17  benefit reserve is  1605.162
At time  18  benefit reserve is  1187.872
At time  19  benefit reserve is  657.8482
```

The benefit reserve for a whole life annuity with level annual premium is ${}_kV({}_{n|}\ddot{a}_x)$, that equals ${}_{n|}\ddot{a}_x - \bar{P}({}_{n|}\bar{a}_x)\ddot{a}_{x+k:\overline{n-k|}}$ when $x \ldots n$, $\ddot{a}_{x+k}$ otherwise. The figure is shown in 3.

3.5.5 Insurance and annuities on two heads

Lifecontingencies package provides function to evaluate life insurance and annuities on two lifes. Following examples will check the equality $a_{\overline{xy}} = a_x + a_y - a_{xy}$.

```
> axn(soa08Act, x = 65, m = 1) + axn(soa08Act, x = 70, m = 1) -
+     axyn(soa08Act, soa08Act, x = 65, y = 70, status = "joint",
+         m = 1)
```

```
[1] 10.35704
```

```
> axyn(soa08Act, soa08Act, x = 65, y = 70, status = "last", m = 1)
```

```
[1] 10.35704
```

Reversionary annuity (annuities payable to life y upon death of x), $a_{x|y} = a_y - a_{xy}$ are also evaluable.

```
> axn(soa08Act, x = 60, m = 1) - axyn(soa08Act, soa08Act, x = 65,
+     y = 60, status = "joint", m = 1)
```

```
[1] 2.695232
```

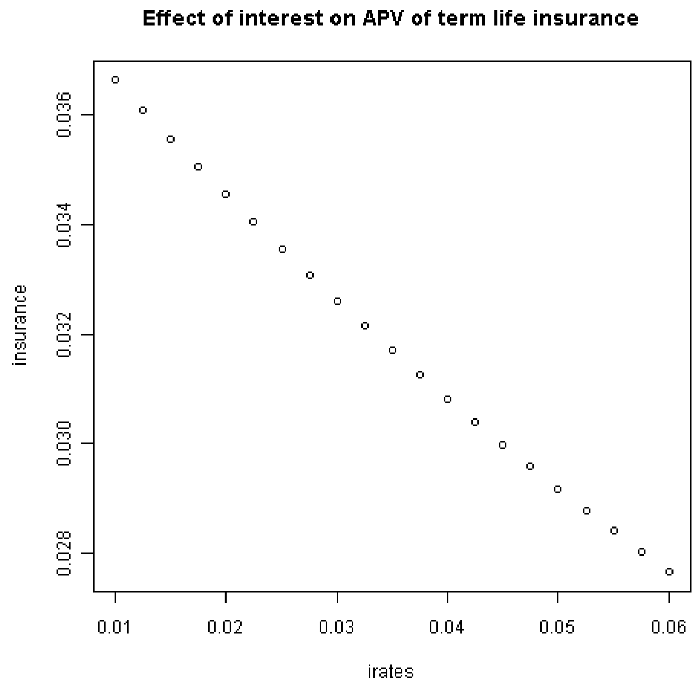

Figure 1: Interest rate effect on life insurance

3.5.6 Other examples

Figure 1 shows the effect of changing interest rates on the APV of $A^{1}_{40:\overline{10}|}$. The APV is a present value of a random variable that represent a composite function between the discount amount and indicator variables regarding the life status of the insured. Figure 2 shows the stochastic distribution of $\ddot{a}_{65}$.

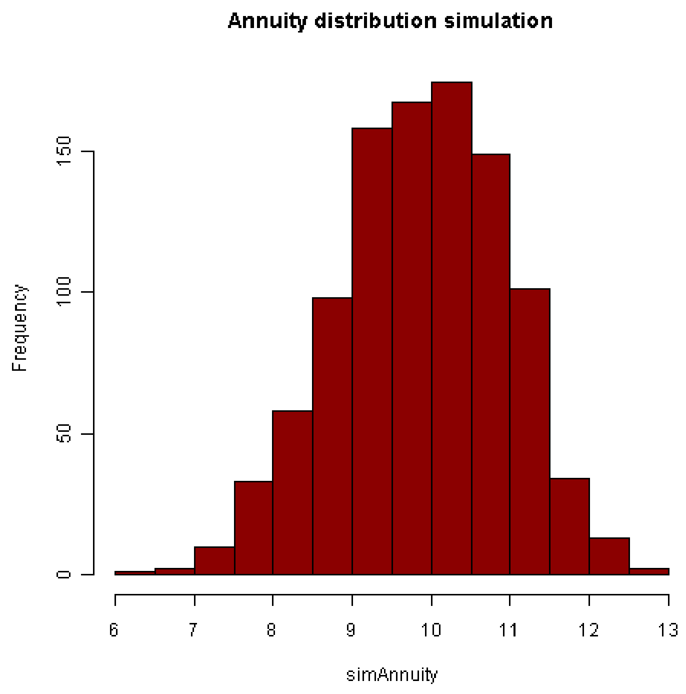

Figure 2: Stochastic distribution of $\ddot{a}_65$

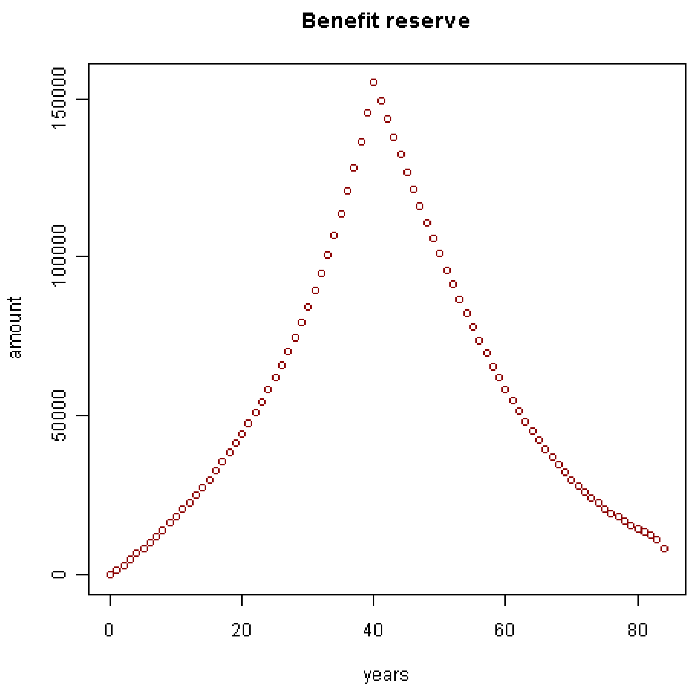

Figure 3: Benefit reserve of $\ddot{a}_{65}$

4 Discussion

Lifecontingencies package allows practictioner actuaries to evaluate actuarial present values functions by means of the R system framework. The lifecontingencies packages offers the basic tools to manipulate life tables, time value of cash flows. These tools are used to evaluate standard life contingencies present values by code already binded to the package as long as to build own function to perform day to day actuarial analysis.

Future work spans in multiple directions. Carefull check of the APV functions will be performed, expecially in the computation of stochastic values. C++ fragments will be tested and addedd to the package whether performance shows to improve.
Finally coerce functions will be written. We wish to provide input and output convenience functions for lifecontingencies objects toward package specialized in demographic analysis. Moreover the use of stochastic interest rate within the actuarial analysis will be facilitated allowing the package to interact with specialized packages.

References

[Bowers et al., 1997] Bowers, N., Gerber, H., Hickman, J., Jones, D., and Nesbitt, C. (1997). Actuarial mathematics. schaumburg. *IL: Society of Actuaries*, pages 79–82.

[Dutang et al., 2008] Dutang, C., Goulet, V., and Pigeon, M. (2008). actuar: An r package for actuarial science. *Journal of Statistical Software*, 25(7):38.

[Gesmann and Zhang, 2011] Gesmann, M. and Zhang, Y. (2011). *ChainLadder: Mack, Bootstrap, Munich and Multivariate-chain-ladder Methods.* R package version 0.1.4-3.4.

[R Development Core Team, 2011] R Development Core Team (2011). *R: A Language and Environment for Statistical Computing.* R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0.

[with contributions from Heather Booth et al., 2011] with contributions from Heather Booth, R. J. H., Tickle, L., and Maindonald, J. (2011). *demography: Forecasting mortality, fertility, migration and population data.* R package version 1.09-1.