# Package ‘RcmdrPlugin.RiskDemo’

**October 3, 2018**

<table>
<thead>
<tr>
<th>Type</th>
<th>Package</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Title</strong></td>
<td>R Commander Plug-in for Risk Demonstration</td>
</tr>
<tr>
<td><strong>Version</strong></td>
<td>2.0</td>
</tr>
<tr>
<td><strong>Date</strong></td>
<td>2018-10-3</td>
</tr>
<tr>
<td><strong>Author</strong></td>
<td>Arto Luoma</td>
</tr>
<tr>
<td><strong>Maintainer</strong></td>
<td>Arto Luoma <a href="mailto:arto.luoma@wippies.com">arto.luoma@wippies.com</a></td>
</tr>
</tbody>
</table>

**Description**
R Commander plug-in to demonstrate various actuarial and financial risks. It includes valuation of bonds and stocks, portfolio optimization, classical ruin theory and demography.

**Depends**
R (>= 2.10), rgl, demography

**Imports**
Rcmdr, ftsa

**Suggests**
tkrplot

**License**
GPL-2

**LazyData**
no

**LazyLoad**
yes

**NeedsCompilation**
no

**Repository**
CRAN

**Date/Publication**
2018-10-03 21:52:23 UTC

### R topics documented:

<table>
<thead>
<tr>
<th>Topic</th>
<th>Pages</th>
</tr>
</thead>
<tbody>
<tr>
<td>RcmdrPlugin.RiskDemo-package</td>
<td>2</td>
</tr>
<tr>
<td>bondCurve</td>
<td>2</td>
</tr>
<tr>
<td>bondFigure</td>
<td>3</td>
</tr>
<tr>
<td>bondPrice</td>
<td>4</td>
</tr>
<tr>
<td>computeRuin</td>
<td>6</td>
</tr>
<tr>
<td>computeRuinFinite</td>
<td>7</td>
</tr>
<tr>
<td>countries</td>
<td>8</td>
</tr>
<tr>
<td>countries.mort</td>
<td>8</td>
</tr>
<tr>
<td>drawFigure</td>
<td>9</td>
</tr>
<tr>
<td>drawRuin</td>
<td>10</td>
</tr>
<tr>
<td>fin</td>
<td>11</td>
</tr>
</tbody>
</table>
R Commander Plug-in for Risk Demonstration

Description

R Commander plug-in to demonstrate various actuarial and financial risks. It includes valuation of bonds and stocks, portfolio optimization, classical ruin theory and demography.

Details

- **Package**: RcmdrPlugin.RiskDemo
- **Type**: Package
- **Version**: 2.0
- **Date**: 2018-10-3
- **License**: GPL (>= 2)
- **LazyLoad**: yes

Author(s)

Arto Luoma

Maintainer: Arto Luoma <arto.luoma@wippies.com>

bondCurve

*Drawing forward and yield curves*

Description

This function draws forward and yields curves, for AAA-rated central government bonds and/or all central government bonds.
Usage

bondFigure(buyDate, matDate, rateCoupon, yieldToMat = NULL, 
bondPr = NULL, nPay)

Arguments

date1 The date for which the curves are drawn
date2 Optional second date for which the curves are drawn
yield Is the yield curve shown (TRUE/FALSE)?
forward Is the forward curve shown (TRUE/FALSE)?
AAA Are the curves drawn for the AAA-rated bonds (TRUE/FALSE)?
all Are the curves drawn for the bonds with all ratings (TRUE/FALSE)?
params The data frame of curve parameters

Value
No value. Only a figure is produced.

Author(s)
Arto Luoma

References
https://bit.ly/2zfs0G8

Examples

data(params)
bondCurve(as.Date("2004-09-06"),params=params)

Description
This function plots the bond price as a function of interest rate. It also shows, using dotted lines, the yield to maturity rate corresponding to the face value, and the flat price corresponding to the yield to maturity.

Usage

bondFigure(buyDate, matDate, rateCoupon, yieldToMat = NULL, 
bondPr = NULL, nPay)
bondPrice

Arguments

- **buyDate**: the date when the coupon is bought (settlement date)
- **matDate**: maturity date
- **rateCoupon**: coupon rate (in decimals)
- **yieldToMat**: yield to maturity (in decimals)
- **bondPr**: the flat price of the bond
- **nPay**: number of coupon payments per year

Details

Either `yieldToMat` or `bondPr` should be given as input.

Value

This function only plots a figure.

Author(s)

Arto Luoma <arto.luoma@wippies.com>

References


See Also

- `bondPrice`
- `solveYield`

Examples

```
bondFigure("2012-7-31","2018-7-31",rateCoupon=0.0225,yieldToMat=0.0079, nPay=2)
bondFigure("2012-7-31","2018-7-31",rateCoupon=0.0225,bondPr=90,nPay=2)
```

---

bondPrice

*Computing bond prices*

Description

This function computes the bond price, given the yield to maturity.

Usage

```
bondPrice(buyDate, matDate, rateCoupon, yieldToMat, nPay)
```
Arguments

buyDate the date at which the bond is bought (settlement date).
matDate maturity date
rateCoupon annual coupon date
yieldToMat yield to maturity
nPay number of coupon payments per day

Details

All the rates are given in decimals.

Value

A list with the following components:
yieldToMaturity yield to maturity
flatPrice flat price
daysSinceLastCoupon days since previous coupon payment
daysInCouponPeriod days in a coupon period
accruedInterest accrued interest since last coupon payment
invoicePrice invoice price (= flat price + accrued interest)

Note

With Excel functions PRICE, DATE, COUPDAYBS and COUPDAYS you can do the same.

Author(s)

Arto Luoma <arto.luoma@wippies.com>

References


See Also

solveYield

Examples

bondPrice("2012-7-31","2018-7-31",0.0225,0.0079,2)
bondPrice("2012-7-31","2018-7-31",0.0225,0.0079,4)
bondPrice("2012-7-31","2030-5-15",0.0625,0.02117,2)
computeRuin  

*Ruin probability computation with infinite time horizon*

**Description**

This function uses classical ruin theory to compute either ruin probability, safety loading or initial capital, given two of them. The time horizon is infinite. Gamma distribution is used to model claim sizes.

**Usage**

```r
computeRuin(u0 = NULL, theta = NULL, eps = NULL, alpha, beta)
```

**Arguments**

- `u0`: initial capital
- `theta`: safety loading
- `eps`: ruin probability
- `alpha`: shape parameter of gamma distribution
- `beta`: rate parameter of gamma distribution

**Value**

The value is a list with the following components:

- `LundbergExp`: Lundberg's exponent $R$
- `initialCapital`: initial capital
- `safetyLoading`: safety loading
- `ruinProb`: ruin probability

**Author(s)**

Arto Luoma <arto.luoma@wippies.com>

**References**


**See Also**

`computeRuinInfinite`, `solveLund`

**Examples**

```r
computeRuin(u0=1000,theta=0.01,alpha=1,beta=0.1)
computeRuin(eps=0.005,theta=0.01,alpha=1,beta=0.1)
computeRuin(u0=5399.24,eps=0.005,alpha=1,beta=0.1)
```
**computeRuinFinite**

Ruin probability computation with finite time horizon

**Description**

This function uses classical ruin theory to compute either ruin probability, safety loading or initial capital, given two of them. The time horizon is finite. Gamma distribution is used to model claim sizes.

**Usage**

```r
computeRuinFinite(tP, U0 = NULL, theta = NULL, eps = NULL, lambda, alpha, beta)
```

**Arguments**

- `tP`: time horizon (in years)
- `U0`: initial capital
- `theta`: safety loading
- `eps`: ruin probability
- `lambda`: claim intensity (mean number of claims per year)
- `alpha`: shape parameter of gamma distribution
- `beta`: rate parameter of gamma distribution

**Value**

The value is a list with the following components:

- `LundbergExp`: Lundberg’s exponent \( R \)
- `initialCapital`: initial capital
- `safetyLoading`: safety loading
- `ruinProb`: ruin probability

**Author(s)**

Arto Luoma <arto.luoma@wippies.com>

**See Also**

- `computeRuin`
- `solveLund`

**Examples**

```r
computeRuinFinite(T0=100, U0=1000, theta=0.01, lambda=100, alpha=1, beta=0.1)
computeRuinFinite(T0=1, eps=0.005, theta=0.001, lambda=100, alpha=1, beta=0.1)
computeRuinFinite(T0=500, U0=5347, eps=0.005, lambda=100, alpha=1, beta=0.1)
```
countries

**Names of the countries in the Human Mortality Database**

**Description**

Names of the countries for which data are available in this package.

**Usage**

data("countries")

**Format**

vector of character strings

**Examples**

data(countries)
print(countries)

---

countries.mort

**Mortality data**

**Description**

Mortality data for 38 countries (period death rates and exposures) retrieved from Human Mortality Database. Exposures are only included for the Nordic countries, China, U.S., Russia, Japan and Germany.

**Usage**

data("countries.mort")

**Format**

List of objects of class demogdata.

**Source**

Human Mortality Database. University of California, Berkeley (USA), and Max Planck Institute for Demographic Research (Germany). Available at www.mortality.org or www.humanmortality.de (data downloaded May 3, 2017).

**Examples**

data(countries.mort)
plot(countries.mort[[1]])
drawFigure

**Description**

Plots the efficient frontiers of risky investments and all investments. The optimum points corresponding to the risk aversion coefficient are indicated by dots. Further, the function plots a predictive return distribution figure.

**Usage**

```r
drawFigure(symbol, yield, vol, beta, r = 1, 
    total = 1, indexVol = 20, nStocks = 7, balanceInt = 12, A = 10, 
    riskfree = FALSE, bor = FALSE)
```

**Arguments**

- `symbol` character vector of the symbols of the risky investments
- `yield` vector of yields (%)
- `vol` vector of volatilities (%)
- `beta` vector of betas (%)
- `r` risk-free interest rate (%)
- `total` total investment (for example in euros)
- `indexVol` volatility of market portfolio (%)
- `nStocks` number of risky investments in the portfolio
- `balanceInt` balancing interval of the portfolio in months
- `A` risk aversion coefficient (see details)
- `riskfree` is risk-free investment included in the portfolio (logical)
- `bor` is borrowing (negative risk-free investment) allowed (logical)

**Details**

The function uses the single-index model and Markovitz portfolio optimization model to find the optimum risky portfolio. The returns are assumed to be log-normally distributed. The maximized function is mu - 0.5*A*var where mu is expected return, A is risk aversion coefficient, and var is return variance.

**Value**

- `portfolio` allocation of the total investment (in euros)
- `returnExpectation` expected portfolio return
- `returnDeviation` standard deviation of the portfolio
Author(s)
Arto Luoma <arto.luoma@wippies.com>

References

See Also
portfOptim

Examples
```r
data(stockData, package="RcmdrPlugin.RiskDemo")
with(stockData,drawFigure(symbol=rownames(stockData),yield=divYield,
vol=vol,beta=beta,r=1,total=100,indexVol=10,
nStocks=5,balanceInt=12,A=10,riskfree=TRUE,bor=FALSE))
```

---

**drawRuin**

*Plotting simulations of a surplus process*

Description
This function plots simulation paths of a surplus process. The claims are assumed to arrive according to a Poisson process and the claim sizes are assumed to be gamma distributed.

Usage
```r
drawRuin(nsim = 10, Tup = 10, U0 = 1000, theta = 0.01,
lambda = 100, alpha = 1, beta = 0.1)
```

Arguments
- **nsim**: number of simulations
- **Tup**: maximum value in the time axis
- **U0**: initial capital
- **theta**: risk loading
- **lambda**: intensity of claim process (mean number of claims per year)
- **alpha**: shape parameter of gamma distribution
- **beta**: rate parameter of gamma distribution

Value
No value; only a figure is plotted.
**Author(s)**
Arto Luoma <arto.luoma@wippies.com>

**References**

**See Also**
`computeRuinFinite`.

**Examples**
```r
computeRuinFinite(T0=10, U0=1000, eps=0.05, lambda=100, alpha=1, beta=0.1)
drawRuin(nsim=10, Tup=10, U0=1000, theta=0.0125, lambda=100, alpha=1, beta=0.1)
```

---

**Description**
Mortality data for Finland

Series: female male total
Years: 1878 - 2015
Ages: 0 - 110

**Usage**
data("fin")

**Format**
object of class demogdata

**Details**
This is part of the `countries.mort` data (`countries.mort[[11]]`).

**Source**
Human Mortality Database. University of California, Berkeley (USA), and Max Planck Institute for Demographic Research (Germany). Available at [www.mortality.org](http://www.mortality.org) or [www.humanmortality.de](http://www.humanmortality.de) (data downloaded May 3, 2017).

**Examples**
data(fin)
print(fin)
plot(fin)
**fin.fcast**  
*Finnish mortality forecast*

**Description**  
Finnish mortality forecast 50 years ahead (2016-2065) for 0 - 100 years old. The forecast is based on an estimated Lee-Carter model. The kt coefficients were forecast using a random walk with drift. Fitted rates were used as the starting value.

**Usage**  
data("fin.fcast")

**Format**  
An object of class "fmforecast"; for details, see documentation of package "demography".

**Details**  
The forecast was produced using function "forecast.lca" of package "demography".

**Examples**  
data(fin.fcast)  
print(fin.fcast)  
plot(fin.fcast)

---

**fin.lca**  
*Lee-Carter model fit for Finnish data*

**Description**  
Lee-Carter model fit obtained by function "lca" of package "demography". The fit is based on Finnish mortality data for ages from 0 to 100 and years from 1950 to 2015.

**Usage**  
data("fin.lca")

**Format**  
object of class "lca"

**Details**  
Both sexes were included in the input mortality data.
**params**  

**Examples**  

```r  
data(fin.lca)  
plot(fin.lca)  
```

---

**Yield curve parameter data**

**Description**

Yield curve parameters from the European Central Bank (ECB), downloaded on May 2, 2017

**Usage**

```r  
data("params")  
```

**Format**

A data frame with 3239 observations on the following 13 variables.

- **date** a Date  
- **b0** a numeric vector  
- **b1** a numeric vector  
- **b2** a numeric vector  
- **b3** a numeric vector  
- **t1** a numeric vector  
- **t2** a numeric vector  
- **c0** a numeric vector  
- **c1** a numeric vector  
- **c2** a numeric vector  
- **c3** a numeric vector  
- **d1** a numeric vector  
- **d2** a numeric vector

**Details**

The parameters b0 to b3 are the beta-parameters, and t1 and t2 the tau-parameters for AAA-rated government bonds. The parameters c0 to c3 are the beta-parameters, and d1 and d2 the tau-parameters for all government bonds.

**Source**

https://bit.ly/2zfs0G8

**Examples**

```r  
data(params)  
bondCurve(as.Date("2004-09-06"), params=params)  
```
pop.pred  Population forecasting

Description

Population forecasting using mortality forecast and simple time series forecast for age 0 population

Usage

   pop.pred(mort, mort.fcast)

Arguments

   mort      mortality data of class 'demogdata'
   mort.fcast mortality forecast of class 'fmforecast'

Details

ARIMA(0,2,2)-model is used to forecast age 0 population.

Value

   population forecast of class 'demogdata'

Author(s)

   Arto Luoma <arto.luoma@wippies.com>

Examples

   data(fin)
   data(fin.fcast)
   fin.pcast <- pop.pred(fin,fin.fcast)
   plot(fin,plot.type="functions",series="total",transform=FALSE,
        datatype="pop",ages=c(0:100), years=c(1990+0:5*10), xlab="Age")
   lines(fin.pcast,plot.type="functions",series="total",transform=FALSE,
        datatype="pop",ages=c(0:100), years=c(1990+0:5*10), lty=2)
Portfolio optimization for an index model

Description

Finds an optimal portfolio for long-term investments and plots a return distribution.

Usage

```r
portfoptim(i, symbol, yield, vol, beta,
    indexVol = 0.2, nStocks = 7, total = 1, balanceInt = 1,
    C = 0.05, riskProportion = 1, riskfreeRate = 0, sim = FALSE)
```

Arguments

- `i`: vector of the indices of the included risky investments
- `symbol`: character vector of the symbols of the risky investments
- `yield`: vector of expected yields (in euros)
- `vol`: vector of volatilities
- `beta`: vector of betas
- `indexVol`: portfolio index volatility
- `nStocks`: number of stocks in the portfolio
- `total`: total sum invested (in euros)
- `balanceInt`: balancing interval of the portfolio (in years)
- `C`: expected portfolio return (in euros)
- `riskProportion`: proportion of risky investments
- `riskfreeRate`: risk-free interest rate
- `sim`: is the return distribution simulated and plotted (logical value)?

Details

The arguments `vol`, `beta`, `indexVol`, `riskProportion` and `riskfreeRate` are given in decimals. The portfolio is optimized by minimizing the variance of the portfolio yield for a given expected yield. The returns are assumed to be log-normally distributed. The covariance matrix is computed using the single index model and the properties of the log-normal distribution.

Value

- `portfolio`: numeric vector of allocations to each stock (in euros)
- `returnExpectation`: expected value of the return distribution (in euros)
- `returnDeviation`: standard deviation of the return distribution (in euros)
- `VaR`: 0.5%, 1%, 5%, 10% and 50% percentiles of the return distribution (in euros)
Note

This function is usually called by drawFigure.

Author(s)

Arto Luoma <arto.luoma@wippies.com>

References


See Also

drawFigure

Examples

data(stockData, package="RcmdrPlugin.RiskDemo")
with(stockData, portfOptim(i=1:5, symbol=rownames(stockData),
    yield=divYield/100, vol=vol/100, beta=beta/100, total=100, sim=TRUE))

returns

Computing expected returns and their covariance matrix

Description

Computing expected returns and their covariance matrix when the returns are lognormal.

Usage

returns(volvec, indexvol, beta)

Arguments

volvec vector of volatilities
indexvol volatility of the portfolio index
beta vector of betas

Details

The arguments are given in decimals. The single index model is used to compute the covariance matrix of a multivariate normal distribution. The mean vector is assumed to be zero. The properties of the log-normal distribution are then used to compute the mean vector and covariance matrix of the corresponding multivariate log-normal distribution.
solveLund

Value

mean vector of expected returns
cov covariance matrix of returns

Author(s)

Arto Luoma <arto.luoma@wippies.com>

References


Examples

returns(volvec=c(0.1,0.2,0.3),indexvol=0.2, beta=c(0.5,-0.1,1.1))

solveLund(alpha, beta, theta)

Arguments

alpha shape parameter of gamma distribution
beta rate parameter of gamma distribution
theta safety loading

Value

Lundberg’s exponent (or adjustment coefficient)

Author(s)

Arto Luoma <arto.luoma@wippies.com>

References

See Also

`computeruin, computeruinfinite`

Examples

```r
solveLund(1,1,0.1)
```

---

### Description

This function computes the yield to maturity, given the (flat) bond price.

### Usage

```r
solveYield(buyDateL, matDateL, rateCouponL, bondPrL, nPayL)
```

### Arguments

- `buyDate`: settlement date (the date when the bond is bought)
- `matDate`: maturity date
- `rateCoupon`: annual coupon rate
- `bondPr`: bond price. The flat price without accrued interest.
- `nPay`: number of payments per year

### Details

- all the rates are given in decimals

### Value

A list with the following components:

- `yieldToMaturity`: yield to maturity
- `flatPrice`: flat price
- `daysSinceLastCoupon`: days since previous coupon payment
- `daysInCouponPeriod`: days in a coupon period
- `accruedInterest`: accrued interest since last coupon payment
- `invoicePrice`: invoice price (= flat price + accrued interest)
**Note**

With Excel function YIELD you can do the same.

**Author(s)**

Arto Luoma <arto.luoma@wippies.com>

**References**


**See Also**

bondPrice

**Examples**

```r
solveYield("2012-7-31","2018-7-31",0.0225,100,2)
```

---

**Description**

This function computes the intrinsic stock price using the constant growth dividend discount model.

**Usage**

```r
stock.price(dividend, k = NULL, g = NULL, ROE = NULL, b = NULL, riskFree = NULL, marketPremium = NULL, beta = NULL)
```

**Arguments**

- `dividend`: expected dividend(s) for the next year(s) (in euros), separated by commas
- `k`: required rate of return
- `g`: growth rate of dividends
- `ROE`: return on investment
- `b`: plowback ratio
- `riskFree`: riskfree rate
- `marketPremium`: market risk premium
- `beta`: beta
Details

All the above rates are given in percentages (except the dividends). One should provide either k or the following three: riskFree, marketPremium, beta. Further, one should provide either g or the following two: ROE and beta. In the output, k and g are given in decimals.

Value

- dividend: expected dividend(s) for the next year(s) (in euros)
- k: required rate of return
- g: growth rate of dividends
- PVGO: present value of growths opportunities
- stockPrice: intrinsic stock price

Author(s)

Arto Luoma <arto.luoma@wippies.com>

References


Examples

```r
stock.price(dividend=c(1),k=12,g=10)
stock.price(dividend=c(1),ROE=50,b=20,riskFree=5,marketPremium=8, beta=90)
```

Description

Stock data on large companies in Helsinki Stock Exchange, downloaded from Kauppalehti web page (www.kauppalehti.fi), on May 13, 2017

Usage

```r
data("stockData")
```
Format

A data frame with 35 observations on the following 7 variables.

- names  name of the firm
- abbrs  abbreviation of the firm
- quote  closing quote
- vol    volatility (%)
- beta   beta (%)
- div    dividend (eur/stock)
- divYield dividend yield (%)

Source

www.kauppalehti.fi

Examples

data(stockData)
plot(stockData[,-(1:2)])
Index

*Topic **datasets**
  - fin, 11
  - params, 13
*Topic **package**
  - RcmdrPlugin.RiskDemo-package, 2

bondCurve, 2
bondFigure, 3
bondPrice, 4, 4, 19

computeRuin, 6, 7, 18
computeRuinFinite, 6, 7, 11, 18
countries, 8
countries.mort, 8
drawFigure, 9, 16
drawRuin, 10

fin, 11
fin.fcast, 12
fin.lca, 12

params, 13
pop.pred, 14
portfOptim, 10, 15

RcmdrPlugin.RiskDemo
  (RcmdrPlugin.RiskDemo-package), 2
RcmdrPlugin.RiskDemo-package, 2
returns, 16

solveLund, 6, 7, 17
solveYield, 4, 5, 18
stock.price, 19
stockData, 20