

# Package: vamac (via r-universe)

September 9, 2024

**Type** Package

**Title** A Monte Carlo Valuation Framework for Variable Annuities

**Version** 0.2.1

**Description** Implementation of a Monte Carlo simulation engine for valuing synthetic portfolios of variable annuities, which reflect realistic features of common annuity contracts in practice. It aims to facilitate the development and dissemination of research related to the efficient valuation of a portfolio of large variable annuities. The main valuation methodology was proposed by Gan (2017) [doi:10.1515/demo-2017-0021](https://doi.org/10.1515/demo-2017-0021).

**Depends** R (>= 3.3.0)

**License** GPL-2

**LazyData** true

**Suggests** knitr, rmarkdown, testthat

**VignetteBuilder** knitr

**RoxygenNote** 7.0.2

**Imports** stats (>= 3.3.0), utils (>= 3.3.0), Rdpack (>= 0.4)

**RdMacros** Rdpack

**NeedsCompilation** no

**Encoding** UTF-8

**Author** Hengxin Li [aut, cph], Ben Feng [aut, cph], Mingyi Jiang [aut, cph, cre], GuoJun Gan [ctb]

**Maintainer** Mingyi Jiang <m64jiang@uwaterloo.ca>

**Date/Publication** 2020-02-28 12:00:02 UTC

**Repository** <https://my-jiang.r-universe.dev>

**RemoteUrl** <https://github.com/cran/vamac>

**RemoteRef** HEAD

**RemoteSha** 0769de7b1c146bed0f1c78bfb7695ee8c6f1b0b5

## Contents

ageOnePolicy . . . . .	2
agePortfolio . . . . .	4
buildCurve . . . . .	5
calcMortFactors . . . . .	7
cForwardCurve . . . . .	8
fundMap . . . . .	8
genFundScen . . . . .	9
genIndexScen . . . . .	9
genPortInception . . . . .	10
histDates . . . . .	11
histIdxScen . . . . .	12
indexNames . . . . .	12
indexScen . . . . .	13
mCov . . . . .	13
mortTable . . . . .	14
swapRate . . . . .	14
valueOnePolicy . . . . .	15
valuePortfolio . . . . .	15
vamc . . . . .	16
VAPort . . . . .	17
<b>Index</b>	<b>19</b>

---

ageOnePolicy	<i>Age One Policy</i>
--------------	-----------------------

---

### Description

Age a VA policy specified in inPolicy from currentDate (specified in inPolicy) to targetDate. The ageing scenario is given in fundScen. The time step length is specified in dT. Here we input a rather irrelevant parameter df to "hack" for a more flexible user-defined projection function.

### Usage

```
ageOnePolicy(
  inPolicy,
  mortTable,
  fundScen,
  scenDates,
  dT = 1/12,
  targetDate,
  df
)
```

**Arguments**

inPolicy	A vector containing 45 attributes of a VA policy, usually a row of a VA portfolio dataframe.
mortTable	A dataframe with three columns of doubles representing the mortality table.
fundScen	A numScen-by-numStep-by-numFund array of doubles of return factors (i.e., $\exp(\mu \cdot dt)$ ) in each period.
scenDates	A vector containing strings in the format of "YYYY-MM-DD" of dates corresponding to each period in fundScen.
dT	A double of stepsize in years; $dT = 1 / 12$ would be monthly.
targetDate	A string in the format of "YYYY-MM-DD" of valuation date of the portfolio.
df	A vector of doubles of risk-free discount rates of different tenor (not forward rates), should have length being numStep.

**Value**

Outputs a vector containing 45 attributes of a VA policy, where currentDate, gbAmt, GMWBbalance, withdrawal, & fundValue could be updated as a result of aging. Usually a row of a VA portfolio dataframe.

**Note**

Target date MUST be PRIOR to the last date of historical scenario date, Current date MUST be LATER than the first date of historical scenario date.

**Examples**

```

exPolicy <- VAPort[1, ]
targetDate <- "2016-01-01"
histFundScen <- genFundScen(fundMap, histIdxScen)
ageOnePolicy(exPolicy, mortTable, histFundScen, histDates, dT = 1 / 12,
targetDate, cForwardCurve)
## Not run:
targetDate <- "2001-01-01"
histFundScen <- genFundScen(fundMap, histIdxScen)
ageOnePolicy(exPolicy, mortTable, histFundScen, histDates, dT = 1 / 12,
targetDate, cForwardCurve)

## End(Not run)
## Not run:
exPolicy <- VAPort[1, ]
exPolicy[1, c("currentDate", "issueDate")] <- c("2001-01-01", "2001-01-01")
histFundScen <- genFundScen(fundMap, histIdxScen)
ageOnePolicy(exPolicy, mortTable, histFundScen, histDates, dT = 1 / 12,
targetDate, cForwardCurve)

## End(Not run)

```

---

 agePortfolio

*Age a Portfolio*


---

### Description

Age a portfolio of VA policies specified in each inPolicy of inPortfolio from currentDate (specified in inPolicy) to targetDate. The ageing scenario is given in fundScen. The time step length is specified in dT. Here we input a rather irrelevant parameter df to "hack" for a more flexible user-defined projection function.

### Usage

```
agePortfolio(
  inPortfolio,
  mortTable,
  fundScen,
  scenDates,
  dT = 1/12,
  targetDate,
  df
)
```

### Arguments

inPortfolio	A dataframe containing numPolicy rows and 45 attributes of each VA policy.
mortTable	A dataframe with three columns of doubles representing the mortality table.
fundScen	A numScen-by-numStep-by-numFund array of doubles of return factors (i.e., $\exp(\mu_t dt)$ ) in each period.
scenDates	A vector containing strings in the format of "YYYY-MM-DD" of dates corresponding to each period in fundScen.
dT	A double of stepsize in years; $dT = 1 / 12$ would be monthly.
targetDate	A string in the format of "YYYY-MM-DD" of valuation date of the portfolio.
df	A vector of doubles of risk-free discount rates of different tenor (not forward rates), should have length being numStep.

### Value

Outputs a dataframe containing numPolicy rows and 45 attributes of each VA policy, where currentDate, gbAmt, GMWBbalance, withdrawal, & fundValue of each policy could be updated as a result of aging.

### Note

Target date MUST be PRIOR to the last date of historical scenario date, Current date MUST be LATER than the first date of historical scenario date.

**Examples**

```

targetDate <- "2016-01-01"
histFundScen <- genFundScen(fundMap, histIdxScen)
agePortfolio(VAPort[1:2, ], mortTable, histFundScen, histDates, dT = 1 / 12,
targetDate, cForwardCurve)
## Not run:
targetDate <- "2001-01-01"
histFundScen <- genFundScen(fundMap, histIdxScen)
agePortfolio(VAPort, mortTable, histFundScen, histDates, dT = 1 / 12,
targetDate, cForwardCurve)

## End(Not run)
## Not run:
VAPort[1, c("currentDate", "issueDate")] <- c("2001-01-01", "2001-01-01")
histFundScen <- genFundScen(fundMap, histIdxScen)
agePortfolio(VAPort, mortTable, histFundScen, histDates, dT = 1 / 12,
targetDate, cForwardCurve)

## End(Not run)

```

---

buildCurve

*Build Curve*


---

**Description**

Bootstrap discount factors from a yield curve.

**Usage**

```

buildCurve(
  swapRates,
  tenors,
  fixFreq = 6,
  fixDCC = "Thirty360",
  fltFreq = 6,
  fltDCC = "Thirty360",
  calendar = "General",
  bdc = c("Actual", "Preceding", "Following", "Modified_Prec", "Modified_Foll"),
  curveDate,
  numSetDay,
  yieldCurveDCC = "Thirty360",
  holidays = NULL
)

```

**Arguments**

swapRates      A vector of doubles of swap rates.  
tenors            A vector of integers of corresponding tenors.

fixFreq	An integer of fixed leg frequency of payment in months. Default is 6, semi-annual payments.
fixDCC	A string of fixed leg day count convention from four options: "Thirty360", "ACT360", "ACT365", or "ACTACT". Default is "Thirty360".
fltFreq	An integer of floating leg frequency of payment in months. Default is 6, semi-annual payments.
fltDCC	A string of floating leg day count convention from four options: "Thirty360", "ACT360", "ACT365", or "ACTACT". Default is "Thirty360".
calendar	A string of the desired calendar convention from two options: <ul style="list-style-type: none"> <li>• "NY": New York holiday calendar</li> <li>• "General": all weekdays are business days</li> </ul>
bdc	A string of business day convention from five options: <ul style="list-style-type: none"> <li>• "Actual": No rolling on the date applied even if it is a non-business day</li> <li>• "Preceding": 1st business day before holiday</li> <li>• "Following": 1st business day after holiday</li> <li>• "Modified_Prec": Same as "Preceding" unless it belongs to a different month, in which case 1st business day after holiday</li> <li>• "Modified_Foll": Same as "Following" unless it belongs to a different month, in which case 1st business day before holiday</li> </ul> Default is "Actual".
curveDate	A string in the format of "YYYY-MM-DD" of yield curve date.
numSetDay	An integer of settlement days from yield curve date.
yieldCurveDCC	A string of yield curve day count convention from four options: "Thirty360", "ACT360", "ACT365", or "ACTACT". Default is "Thirty360".
holidays	An optional vector dates of user-defined holidays. If provided, within the given holidays range, the calendar provided in the parameter "calendar" will not be applied; If the date is not in the given holidays range, it will follow the calendar provided in the "calendar" parameter

### Value

Outputs a data frame of strings of discount dates and doubles of discount factors.

### Examples

```
rate <- c(0.69, 0.77, 0.88, 1.01, 1.14, 1.38, 1.66, 2.15) * 0.01
tenor <- c(1, 2, 3, 4, 5, 7, 10, 30)
fixFreq <- 6
fixDCC <- "Thirty360"
fltFreq <- 6
fltDCC <- "ACT360"
```

```

calendar <- "NY"
bdc <- "Modified_Foll"
curveDate <- "2016-02-08"
numSetDay <- 2
yieldCurveDCC <- "Thirty360"
holidays <- NULL
buildCurve(rate, tenor, fixFreq, fixDCC, fltFreq, fltDCC, calendar, bdc,
           curveDate, numSetDay, yieldCurveDCC, holidays)

```

---

calcMortFactors	<i>Calculate Mortality Factors</i>
-----------------	------------------------------------

---

### Description

Calculates the mortality factors  $(t - 1)px q(x + t - 1)$  and  $tpx$  required to value the inPolicy. Extract gender, age (birth date & current date), valuation date (current date), and maturity date from inPolicy, mortality rates from mortTable.

### Usage

```
calcMortFactors(inPolicy, mortTable, dT = 1/12)
```

### Arguments

inPolicy	A vector containing 45 attributes of a VA policy, usually a row of a VA portfolio dataframe.
mortTable	A dataframe with three columns of doubles representing the mortality table.
dT	A double of stepsize in years; $dT = 1 / 12$ would be monthly.

### Value

Outputs a two-column data frame of doubles of mortFactors  $(t - 1)px q(x + t - 1)$  and  $tpx$ .

### Examples

```

exPolicy <- VAPort[1, ]
calcMortFactors(exPolicy, mortTable, dT = 1 / 12)

```

---

cForwardCurve	<i>Constant Forward Curve</i>
---------------	-------------------------------

---

**Description**

A dataset containing 2 percent continuously compounded annual interest rate for illustration purposes.

**Usage**

cForwardCurve

**Format**

A vector with 360 elements:

**rate** discount rate ...

---

fundMap	<i>Fund Map for 10 Funds</i>
---------	------------------------------

---

**Description**

A dataset containing a default mapping from five indices to ten different funds.

**Usage**

fundMap

**Format**

A matrix with 10 rows and 5 columns:

**index name** name for each index

**fund number** proportion of fund allocated to a particular index ...



---

genFundScen	<i>Generate Fund Scenerio</i>
-------------	-------------------------------

---

**Description**

Calculate numScen-by-numStep-by-numFund fund scenarios based on given index scenarios indexScen and fund map fundMap that maps indices to funds.

**Usage**

```
genFundScen(fundMap, indexScen)
```

**Arguments**

fundMap	A numFund-by-numIndex matrix of doubles, mapping indices to funds.
indexScen	A numScen-by-numStep-by-numIndex array of doubles, index scenarios.

**Value**

Outputs a numScen-by-numStep-by-numFund array of doubles of fund scenarios.

**Examples**

```
genFundScen(fundMap, indexScen)
```

---

genIndexScen	<i>Generate Index Scenerio</i>
--------------	--------------------------------

---

**Description**

Simulate a 3D array, numScen-by-numStep-by-numIndex, of Black-Scholes return factors for numIndex indices in each of numStep time steps and each of numScen scenarios. Covariances among indices are specified in covMatrix. Step size is given as dT and interpolated discount factors are given in vDF. Random seed is optional for reproducibility.

**Usage**

```
genIndexScen(
  covMatrix,
  numScen,
  numStep,
  indexNames,
  dT = 1/12,
  forwardCurve,
  seed
)
```

**Arguments**

covMatrix	A numIndex-by-numIndex matrix of doubles of covariances among numIndex indices.
numScen	An integer of number of scenario (sample paths) to be simulated.
numStep	An integer of number of periods to be simulated.
indexNames	A vector of strings containing index names.
dT	A double of stepsize in years; dT = 1 / 12 would be monthly.
forwardCurve	A vector of doubles of discount rates at each time step.
seed	An integer of the deterministic seed for random sampling.

**Value**

Outputs a 3D array (numScen-by-numStep-by-numIndex) of index scenarios

**Examples**

```
genIndexScen(mCov, 100, 360, indexNames, 1 / 12, cForwardCurve, 1)
```

---

genPortInception      *Generate Portfolio at Inception*

---

**Description**

Generate a portfolio of VA contracts at inception based on given attribute ranges and investment fund information.

**Usage**

```
genPortInception(
  birthDayRng = c("1950-01-01", "1980-01-01"),
  issueRng = c("2001-08-01", "2014-01-01"),
  matRng = c(15, 30),
  acctValueRng = c(50000, 5e+05),
  femPct = 0.4,
  fundFee = c(30, 50, 60, 80, 10, 38, 45, 55, 47, 46),
  baseFee = 200,
  prodPct = rep(1/19, 19),
  prodType = c("DBRP", "DBRU", "DBSU", "ABRP", "ABRU", "ABSU", "IBRP", "IBRU", "IBSU",
    "MBRP", "MBRU", "MBSU", "WBRP", "WBRU", "WBSU", "DBAB", "DBIB", "DBMB", "DBWB"),
  riderFee = c(25, 35, 35, 50, 60, 60, 60, 70, 70, 50, 60, 60, 65, 75, 75, 75, 85, 75,
    90),
  rollUpRate = rep(5, 19),
  withdrawalRate = rep(5, 19),
  numPolicy = 10
)
```

**Arguments**

birthDayRng	A vector of two strings in 'YYYY-MM-DD' of birthday range.
issueRng	A vector of two strings in 'YYYY-MM-DD' of issue date range.
matRng	A vector of two integers, range of policy maturity.
acctValueRng	A vector of two doubles, range of initial account values.
femPct	A double, percentage of female policyholders in the portfolio.
fundFee	A vector of doubles, fees charged by each fund in bps.
baseFee	A double, base fee for all funds in bps.
prodPct	A vector of non-negative doubles, proportions of rider types.
prodType	A vector of strings, names of different rider types.
riderFee	A vector of doubles, rider fees for different riders in bps.
rollUpRate	A vector of doubles, roll up rates for different rider types in bps.
withdrawalRate	A vector of doubles, withdrawal rates for different rider types in bps.
numPolicy	An integer, number of each type of policies to be generated.

**Value**

Outputs a data frame of 45 columns of attributes in an annuity contract.

**Examples**

```
genPortInception(c("1980-01-01", "1990-01-01"), c("2001-08-01", "2014-01-01"),
c(15, 30), c(5e4, 5e5), 0.4, c(30, 50, 60, 80, 10, 38, 45, 55, 47, 46),
200, rep(1 / 4, 4), c("WBRP", "WBRU", "WBSU", "DBWB"),
riderFee = c(25, 35, 35, 50), rep(5, 4), rep(5, 4), 100)
## Not run:
genPortInception()

## End(Not run)
```

---

histDates

*Historical Scenario Dates*


---

**Description**

A dataset containing the dates at which historical returns for different indices were observed.

**Usage**

```
histDates
```

**Format**

A vector with 175 elements:

**date** each observation date of the historical scenarios ...

---

histIdxScen	<i>Historical Index Scenario for 5 Indices over 175 Months</i>
-------------	--

---

**Description**

A dataset containing a matrix, number of indices (5) by number of time steps (175), of observed historical returns for each index in each of time step in the past.

**Usage**

histIdxScen

**Format**

A data frame with dimensions 175 rows and 10 columns:

**FIXED** historical return for index "FIXED" in one month

**INT** historical return for index "INT" in one month

**MONEY** historical return for index "MONEY" in one month

**SMALL** historical return for index "SMALL" in one month

**US** historical return for index "US" in one month ...

**Remark**

These historical index scenarios were assessed on 2008-09-12

**Source**

<http://www.math.uconn.edu/~gan/software.html>

---

indexNames	<i>Index Names</i>
------------	--------------------

---

**Description**

A dataset containing names for each index.

**Usage**

indexNames

**Format**

A vector with 5 elements:

**name** name of the index ...

---

indexScen

*5 Indices for 10 Scenarios over 360 Months*

---

### Description

A dataset containing a 3D array, number of scenarios (10) by number of indices (5) by number of time steps (360), of Black-Scholes return factors for each index in each of time step and each of scenario.

### Usage

indexScen

### Format

A 3D array with dimensions 10x360x5:

**scenario** scenario number

**month** month since valuation date

**index number** monthly return for a particular index in one scenario one month ...

---

mCov

*Covariance Matrix for 5 Indices*

---

### Description

A dataset containing the covariance matrix among the returns of five indices.

### Usage

mCov

### Format

A matrix with 5 rows and 5 columns:

**index number** number for each index ...

---

mortTable	<i>Mortality Rate for Male and Female from Ages 5 to 115</i>
-----------	--

---

**Description**

A dataset containing the mortality rates for male and female from ages 5 to 115 (table IAM 1996 from the Society of Actuaries).

**Usage**

mortTable

**Format**

A data frame with 110 rows and 3 columns:

**age** individual's age

**male** mortality of a male at a particular age ranging from 5 to 115

**female** mortality of a female at a particular age ranging from 5 to 115 ...

**Source**

<https://mort.soa.org>

---

swapRate	<i>Swap Rates across 30 Years</i>
----------	-----------------------------------

---

**Description**

A dataset containing US swap rates for various maturities.

**Usage**

swapRate

**Format**

A vector with 8 elements:

**rate** swap rate ...

**Remark**

These swap rates were assessed on 2016-02-08

**Source**

<http://www.federalreserve.gov>

valuateOnePolicy      *Valuate One Policy*

**Description**

Valuate a VA policy specified in inPolicy based on the simulated fund scenarios fundScen. The time step length is specified in dT and the discount rate for each period is specified in df.

**Usage**

```
valuateOnePolicy(inPolicy, mortTable, fundScen, dT = 1/12, df)
```

**Arguments**

- inPolicy      A vector containing 45 attributes of a VA policy, usually a row of a VA portfolio dataframe.
- mortTable    A dataframe with three columns of doubles representing the mortality table.
- fundScen     A numScen-by-numStep-by-numFund array of doubles of return factors (i.e.,  $\exp(\mu_t dt)$ ) in each period.
- dT            A double of stepsize in years;  $dT = 1 / 12$  would be monthly.
- df            A vector of doubles of risk-free discount rates of different tenor (not forward rates), should have length being numStep.

**Value**

Outputs a list of doubles of policyValue, the average discounted payoff of the VA, and riskCharge, the average discounted risk charges.

**Examples**

```
fundScen <- genFundScen(fundMap, indexScen)[1, , ]
exPolicy <- VAPort[1, ]
valuateOnePolicy(exPolicy, mortTable, fundScen, 1 / 12, cForwardCurve)
```

valuatePortfolio      *Valuate a Portfolio*

**Description**

Valuate a portfolio VA policies specified in each curPolicy of inPortfolio based on the simulated fund scenarios fundScen. The time step length is specified in dT and the discount rate for each period is specified in df.

**Usage**

```
valuePortfolio(inPortfolio, mortTable, fundScen, dT = 1/12, df)
```

**Arguments**

<code>inPortfolio</code>	A dataframe containing <code>numPolicy</code> rows and 45 attributes of each VA policy.
<code>mortTable</code>	A dataframe with three columns of doubles representing the mortality table.
<code>fundScen</code>	A <code>numScen</code> -by- <code>numStep</code> -by- <code>numFund</code> array of doubles of return factors (i.e., $\exp(\mu \cdot dt)$ ) in each period.
<code>dT</code>	A double of stepsize in years; $dT = 1 / 12$ would be monthly.
<code>df</code>	A vector of doubles of risk-free discount rates of different tenor (not forward rates), should have length being <code>numStep</code> .

**Value**

Outputs a list of doubles of `portVal`, the sum of average discounted payoff of the VAs in `inPortfolio`, `portRC`, the sum of average discounted risk charges of the VAs in `inPortfolio`, and vectors of doubles of these average discounted values for each policy.

**Examples**

```
fundScen <- genFundScen(fundMap, indexScen)[1, , ]
valuePortfolio(VAPort[1:2, ], mortTable, fundScen, 1 / 12, cForwardCurve)
```

---

vamc

*vamc: A package for pricing a pool of variable annuities.*


---

**Description**

The `vamc` package provides a Monte Carlo engine for valuating a pool of variable annuities. The key steps are: `YieldCurveGeneration`, `ScenarioGeneration`, `PolicyGeneration`, and `MonteCarloValuation`.

**YieldCurveGeneration functions**

`YieldCurveGeneration` generates a forward curve from swap rates. The forward curve is obtained by solving for swap rates that equates values of floating and fixed notes.

**ScenarioGeneration functions**

`ScenarioGeneration` generates a random fund scenario under Black-Scholes. After simulating random index scenarios, a `fundMap` is used to allocate returns of indices to each fund according to proportion of investment.



### PolicyGenerationI functions

PolicyGenerationI randomly generates a pool of variable annuities for user-input birthday range, issue-date range, maturity range, account value range, female percentage, fund management fee, fund base fee, product types, rider fee of each type, roll-up-rate for roll-up featured guarantees, withdrawal rate for GMWB, and number of policies to be generated for each type.

### MonteCarloValuation functions

MonteCarloValuation discounts cash flow from living and death benefits, as well as risk charges for each policy in the portfolio.

### References

Gan G, Valdez EA (2017). “Valuation of Large Variable Annuity Portfolios: Monte Carlo Simulation and Synthetic Datasets.” *Dependence Modeling*, 5, 354–374. doi: [10.1515/demo20170021](https://doi.org/10.1515/demo20170021).

---

 VAPort

---

*A Randomly Generated Pool of Variable Annuities*


---

### Description

A dataset containing information of the policy and the policy holder.

### Usage

VAPort

### Format

A data frame with 19 row and 45 columns:

**recordID** Unique identifier of the policy  
**survivorShip** Positive weighting number  
**gender** Gender of the policyholder  
**productType** Product type  
**issueDate** Issue date  
**matDate** Maturity date  
**birthDate** Birth date of the policyholder  
**currentDate** Current date  
**baseFee** M&E (Mortality & Expense) fee  
**riderFee** Rider fee  
**rollUpRate** Roll-up rate  
**gbAmt** Guaranteed benefit  
**gmwbBalance** GMWB balance

**wbWithdrawalRate** Guaranteed withdrawal rate  
**withdrawal** Withdrawal so far  
**fundNum1** Fund number of the 1st investment fund  
**fundNum2** Fund number of the 2nd investment fund  
**fundNum3** Fund number of the 3rd investment fund  
**fundNum4** Fund number of the 4th investment fund  
**fundNum5** Fund number of the 5th investment fund  
**fundNum6** Fund number of the 6th investment fund  
**fundNum7** Fund number of the 7th investment fund  
**fundNum8** Fund number of the 8th investment fund  
**fundNum9** Fund number of the 9th investment fund  
**fundNum10** Fund number of the 10th investment fund  
**fundValue1** Fund value of the 1st investment fund  
**fundValue2** Fund value of the 2nd investment fund  
**fundValue3** Fund value of the 3rd investment fund  
**fundValue4** Fund value of the 4th investment fund  
**fundValue5** Fund value of the 5th investment fund  
**fundValue6** Fund value of the 6th investment fund  
**fundValue7** Fund value of the 7th investment fund  
**fundValue8** Fund value of the 8th investment fund  
**fundValue9** Fund value of the 9th investment fund  
**fundValue10** Fund value of the 10th investment fund  
**fundFee1** Fund management fee of the 1st investment fund  
**fundFee2** Fund management fee of the 2nd investment fund  
**fundFee3** Fund management fee of the 3rd investment fund  
**fundFee4** Fund management fee of the 4th investment fund  
**fundFee5** Fund management fee of the 5th investment fund  
**fundFee6** Fund management fee of the 6th investment fund  
**fundFee7** Fund management fee of the 7th investment fund  
**fundFee8** Fund management fee of the 8th investment fund  
**fundFee9** Fund management fee of the 9th investment fund  
**fundFee10** Fund management fee of the 10th investment fund ...

# Index

## \* datasets

- cForwardCurve, 8
- fundMap, 8
- histDates, 11
- histIdxScen, 12
- indexNames, 12
- indexScen, 13
- mCov, 13
- mortTable, 14
- swapRate, 14
- VAPort, 17

- ageOnePolicy, 2
- agePortfolio, 4

- buildCurve, 5

- calcMortFactors, 7
- cForwardCurve, 8

- fundMap, 8

- genFundScen, 9
- genIndexScen, 9
- genPortInception, 10

- histDates, 11
- histIdxScen, 12

- indexNames, 12
- indexScen, 13

- mCov, 13
- mortTable, 14

- swapRate, 14

- valueOnePolicy, 15
- valuePortfolio, 15
- vamc, 16
- VAPort, 17