Package ‘stocins’

April 3, 2017

Type Package

Title Stochastic Interest Rate Models for Life Insurance

Version 0.1.0

Author Nathan Esau

Maintainer Nathan <nesau@sfu.ca>

Description This package can be used to analyze the risk of an insurance portfolio using stochastic interest rate models. The focus is on calculating the first three moments of the present value of benefit random variable for a portfolio of endowment or term insurance contracts. Several stochastic interest rate models are implemented including the Wiener process, the Ornstein-Uhlenbeck process and a Second Order Stochastic Differential equation for the force of interest. The references for this package are (1) Parker, Gary (1992) An application of stochastic interest rate models in life assurance and (2) Parker, Gary (1997) Stochastic analysis of the interaction between investment and insurance risks.

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 5.0.1

R topics documented:

`ann.ev` *Expected value of annuity*

Description

Calculates the expected value of an n-period annuity. See the iratemodel class for details and examples.

Usage

```
ann.ev(n, irm)
```

Arguments

`n`	The length of the annuity
`irm`	The interest rate model to use

Value

Present value of annuity

`ann.var` *Variance of annuity*

Description

Calculates the variance of an n-period annuity. See the iratemodel class for details and examples.

Usage

```
ann.var(n, irm)
```

Arguments

`n`	The length of the annuity
`irm`	The interest rate model to use

Value

Variance of annuity

`ar1` *The AR(1) Process*

Description

A class used to model an AR(1) process. See the iratemodel class for details and examples.

Usage

```
## S3 method for class 'ar1'
delta.ev(t, irm)

## S3 method for class 'ar1'
delta.cov(s, t, irm)

## S3 method for class 'ar1'
delta.var(t, irm)

## S3 method for class 'ar1'
y.ev(t, irm)

## S3 method for class 'ar1'
y.var(t, irm)

## S3 method for class 'ar1'
y.cov(s, t, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

`arma` *The ARMA(2,1) Process*

Description

A class used to model the ARMA(2,1) process. See the iratemodel class for details and examples.

Usage

```
## S3 method for class 'arma'
delta.ev(t, irm)

## S3 method for class 'arma'
delta.cov(s, t, irm)

## S3 method for class 'arma'
delta.var(t, irm)

## S3 method for class 'arma'
y.ev(t, irm)

## S3 method for class 'arma'
y.var(t, irm)

## S3 method for class 'arma'
y.cov(s, t, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

`CdnPop2016` *Canadian Population Table for 2016*

Description

Canadian Population for 2016 from Statistics Canada census.

Usage

```
CdnPop2016
```

Format

Male and Female population counts and proportions for different age groups.

Source

Statistics Canada

---

determ *Deterministic interest rate*

Description

A class used to model the deterministic interest rate. See the iratemodel class for details and examples.

Usage

```
## S3 method for class 'determ'
delta.ev(t, irm)

## S3 method for class 'determ'
delta.cov(s, t, irm)

## S3 method for class 'determ'
delta.var(t, irm)

## S3 method for class 'determ'
y.ev(t, irm)

## S3 method for class 'determ'
y.var(t, irm)

## S3 method for class 'determ'
y.cov(s, t, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

endowport — *Endowment insurance portfolio (identical policies)*

Description

A portfolio of c identical endowment policies. See the insurance class for details and examples on how to use this class.

Usage

```
## S3 method for class 'iport.endowport'
z.moment(moment, ins, mort, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

endowsingle — *Single endowment insurance product*

Description

An endowment insurance pays a benefit of d if the policyholder dies within the term of the contract and e if the policyholder survives the term of the contract.

In addition to the functions described in the insurance class, other functions are available for the endowsingle class.

In particular, `z.ev.two.isingle.endowsingle` can be used to calculate $E[Z_1 Z_2]$, `z.ev.three.isingle.endowsingle` can be used to calculate $E[Z_1 Z_2 Z_3]$ and `z.ev.twoone.isingle.endowsingle` can be used to calculate $E[Z_1^2 Z_2]$ in Parker (1992).

Usage

```
## S3 method for class 'isingle.endowsingle'
z.moment(moment, ins, mort, irm)

z.ev.two.isingle.endowsingle(ins, mort, irm)

z.ev.three.isingle.endowsingle(ins, mort, irm)

z.ev.twoone.isingle.endowsingle(ins, mort, irm)

## S3 method for class 'isingle.endowsingle'
z.insrisk(ins, mort, irm)

## S3 method for class 'isingle.endowsingle'
z.invrisk(ins, mort, irm)
```

Details

The `z.insrisk` and `z.invrisk` functions for the `endowsingle` class are implemented such that we are conditioning on K not y. See Parker (1997) for details.

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

Parker, G. (1997). Stochastic analysis of the interaction between investment and insurance risks. North American actuarial journal, 1(2), 55–71.

Examples

```
oumodel = iratemodel(list(delta0 = 0.1, delta = 0.06,
alpha = 0.1, sigma = 0.01), "ou")
mort = mortassumptions(list(x = 40, table = "MaleMort91"))
endow = insurance(list(n = 10, d = 1, e = 1), "isingle", "endow")

z.ev(endow,mort,oumodel) # first moment
z.ev.two.isingle.endowsingle(endow, mort, oumodel)
z.ev.twoone.isingle.endowsingle(endow, mort, oumodel)
z.ev.three.isingle.endowsingle(endow, mort, oumodel)
```

`FemaleMort82` *CA Male Mortality rates from 1980-1982 * 0.9*

Description

This data set contains the mortality rates used in Parker (1997).

Usage

```
FemaleMort82
```

Format

a `x` column for the age from 0 to 102 inclusive and a `qx` column for the mortality rate.

References

Parker, G. (1997). Stochastic analysis of the interaction between investment and insurance risks. North American actuarial journal, 1(2), 55–71.

FemaleMort82Reduced *CA Male Mortality rates from 1980-1982 * 0.75*

Description

This data set contains the mortality rates used in Parker.

Usage

```
FemaleMort82Reduced
```

Format

a x column for the age from 0 to 102 inclusive and a qx column for the mortality rate.

References

Parker, G. (1997). Stochastic analysis of the interaction between investment and insurance risks. North American actuarial journal, 1(2), 55–71.

FemaleMort91 *CA Female Mortality rates from 1990-1992*

Description

This data set contains the 1991 mortality rates from Statistics Canada.

Usage

```
FemaleMort91
```

Format

a x column for the age from 0 to 100 inclusive and a qx column for the mortality rate.

Source

Statistics Canada

igroup *Group of Insurance Portfolios*

Description

A class for calculating the moments, insurance risk and and investment risk for a group of endowment and term insurance portfolios using the formulas described in Parker (1997).

In addition to the functions described in the insurance class some other functions are available for the `igroup` class.

In particular, `z.ev.two.igroup` calculates $E[Z_{1,i}, Z_{i,r}]$ for `i = ind1` and `r = ind2`, `cashflow.ev` calculates the expected cashflow at time `r` and `cashflow.cov` calculates the covariance between the cashflows at time `r` and time `s` as describd in Parker (1997).

For details and examples on how to use the `igroup` class see the insurance class.

Usage

```
z.ev.two.igroup(ind1, ind2, ins, mort, irm)

## S3 method for class 'igroup'
z.moment(moment, ins, mort, irm)

cashflow.ev(r, ins, mort, irm)

cashflow.cov(s, r, ins, mort, irm)

## S3 method for class 'igroup'
z.insrisk(ins, mort, irm)

## S3 method for class 'igroup'
z.invrisk(ins, mort, irm)
```

References

Parker, G. (1997). Stochastic analysis of the interaction between investment and insurance risks. North American actuarial journal, 1(2), 55–71.

Examples

```
oumodel = iratemodel(list(delta0 = 0.1, delta = 0.06,
alpha = 0.1, sigma = 0.01), "ou")
mort = mortassumptions(list(x = 40, table = "MaleMort91"))
mort2 = mortassumptions(list(x = 50, table = "FemaleMort91"))

termins = insurance(list(n = 10, d = 1), "isingle", "term")
endowins = insurance(list(n = 10, e = 1, d = 1), "isingle", "endow")

termport = insurance(list(single = termins, c = 1000), "iport", "term")
```

```
endowport = insurance(list(single = endowins, c = 1000), "iport", "endow")

groupins = insurance(list(termport, endowport),
"igroup") # 1000 term contracts, 1000 endow contracts
groupmort = list(mort, mort2) # term contracts are age 40, endow contracts are age 50
z.moment(1, groupins, groupmort, oumodel) / groupins$c # average cost per policy
z.ev.two.igroup(1, 2, groupins, groupmort, oumodel)
cashflow.ev(5, groupins, groupmort, oumodel)
cashflow.cov(3, 5, groupins, groupmort, oumodel)
```

`insurance` *Insurance product*

Description

A class used to describe the insurance product we are modeling. Three types of insurance policies can be modeled:

1. Insurance policies issued to a single life, i.e. the isingle subclasses. For these policies, the `params` argument should be a list with `n` for the term of the contract, `d` for the death benefit and `e` for the survival benefit. If `e` is not specified it is assumed to be 0. For a policy issued to a single life, the class argument should be "isingle" and the subclass argument should be either "term" for a term insurance or "endow" for an endowment insurance. See the examples below.

2. Identical policies issued to many lives, i.e. the iport subclasses. For these policies, the `params` argument should be a list with `single` being an `isingle` object and `c` being a number specifying how many identical policies are in the portfolio. For a portfolio of policies, the class argument should be "iport" and the subclass argument should either be "term" for a term insurance or "endow" for an endowment insurance. See the examples below.

3. A group of insurance portfolios, i.e. the igroup subclass. For a group of policies, the `params` argument should be a list of iport objects and the class argument should be "igroup". The subclass argument is not needed. See the examples below.

Usage

```
insurance(params, class, subclass = NULL)

z.moment(moment, ins, mort, irm)

z.ev(ins, mort, irm)

z.sd(ins, mort, irm)

z.sk(ins, mort, irm)

z.insrisk(ins, mort, irm)

z.invrisk(ins, mort, irm)
```

```
z.pdf(z, ins, mort, irm)
```

Details

For each class, several functions are available.

The `z.moment` function can be used to calculate the raw moments of the present value of benefit random variable. For the isingle classes all the moments are implemented, for the iport classes the first three moments are implemented and for the igroup class the first two moments are implemented. The formulas from Parker (1992) were used to implement these moments.

The `z.ev` function can be used to calculate the first moment, the `z.sd` function can be used to calculate the standard deviation and the `z.sk` function can be used to calculate the skewness of the present value of benefit random variable.

The `z.insrisk` function can be used to calculate the insurance risk arising from uncertain mortality and `z.invrisk` can be used to calculate the investment risk arising from uncertain investment returns. The formulas from Parker (1997) were used to implement these functions.

The `z.pdf` function can be used to calculate the density function of the present value of benefit random variable for the isingle classes. This function has not been implemented for the iport and igroup classes. For details on how this could be done, refer to Parker (1992) and Parker (1997).

Refer to the examples below for how to use these functions.

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

Parker, G. (1997). Stochastic analysis of the interaction between investment and insurance risks. North American actuarial journal, 1(2), 55–71.

Examples

```
oumodel = iratemodel(list(delta0 = 0.1, delta = 0.06,
alpha = 0.1, sigma = 0.01), "ou")
mort = mortassumptions(list(x = 40, table = "MaleMort91"))
mort2 = mortassumptions(list(x = 50, table = "FemaleMort91"))

## isingle classes
termins = insurance(list(n = 10, d = 1), "isingle", "term")
endowins = insurance(list(n = 10, e = 1, d = 1), "isingle", "endow")

z.ev(termins, mort, oumodel) # first moment
z.moment(2, termins, mort, oumodel) # second moment
z.moment(3, termins, mort, oumodel) # third moment
z.sd(endowins, mort, oumodel) # standard deviation
z.sk(endowins, mort, oumodel) # skewness

plot(function(z) z.pdf(z, termins, mort, oumodel), 0.01, 1.0,
ylim = c(0, 0.15), lty = 1, xlab = "z", ylab = "f(z)")

legend('topleft', leg = c(paste0("P(Z=0) = ", round(kpx(1, mort), 5))),
```

```
lty = 1)

## iport classes
termport = insurance(list(single = termins, c = 1000), "iport", "term")
endowport = insurance(list(single = endowins, c = 1000), "iport", "endow")
z.moment(1, termport, mort, oumodel) / termport$c # average cost
z.sd(termport, mort, oumodel) / termport$c # average standard deviation

## igroup class
groupins = insurance(list(termport, endowport),
"igroup") # 1000 term contracts, 1000 endow contracts
groupmort = list(mort, mort2) # term contracts are age 40, endow contracts are age 50
z.moment(1, groupins, groupmort, oumodel) / groupins$c # average cost per policy
z.insrisk(groupins, groupmort, oumodel) / termport$c^2 # insrisk per policy
z.invrisk(groupins, groupmort, oumodel) / termport$c^2 # invrisk per policy
z.sd(groupins, groupmort, oumodel) / termport$c # sd per policy
```

`iport` *Insurance portfolio (identical policies)*

Description

A portfolio of c identical term or endowment policies. The termport and endowport classes implement the functions for the present value of benefit random variable. See the insurance class for details and examples on how to use this class.

Usage

```
z.moment.iport(moment, ins, mort, irm)

## S3 method for class 'iport'
z.insrisk(ins, mort, irm)

## S3 method for class 'iport'
z.invrisk(ins, mort, irm)

## S3 method for class 'iport'
z.pdf(z, ins, mort, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

iratemodel *Interest Rate Model*

Description

A class used to describe the interest rate we are modeling. Six types of interest rate models can be used to model the force of interest δ_s (see Parker (1992)):

1. Deterministic interest rate, i.e. the determ class. For the determinstic interest rate, the `params` argument should be a list with `delta` for the interest rate. The class argument should be "determ". See the examples below.

2. Wiener process, i.e. the wiener class. For the wiener process, the `params` argument should be a list with `delta` for the long term mean and `sigma` for the local volatility. The class argument should be "wiener". See the examples below.

3. Ornstein-Uhlenbeck process, i.e. the ou class. For the Ornstein-Uhlenbeck process, the `params` should be a list with `delta` for the long term mean, `alpha` for the friction parameter, `sigma` for the local volatility and `delta0` for the initial value of the process. The class argument should be "ou". See the examples below.

4. Second Order Stochastic Differential Equation, i.e. the second class. For the second order process, the `params` should be a list with `delta` for the long term mean, `alpha1` for the first friction parameter, `alpha2` for the second friction parameter, `sigma` for the local volatility, `delta0prime` for the initial derivative of the process and and `delta0` for the initial value of the process. Note that Parker uses the notation of `alpha0` instead of `alpha2` but note that we are referring to the same parameter. The class argument should be "second". See the examples below.

5. AR(1) process, i.e. the ar1 class. For the AR(1) process, the `params` should be a list with `delta` for the long term mean, `delta0` for the initial value of the process, `phi1` for the AR(1) coefficient and `sigma` for the local volatility. The class argument should be "ar1". See the examples below.

6. ARMA(2,1) process, i.e. the arma class. For the ARMA(2,1) process, the `params` should be a list with `delta` for the long term mean, `delta0` for the initial value of the process, `phi1` for the AR(1) coefficient, `phi2` for the AR(2) coefficient, `theta1` for the MA(1) coefficient and `sigma` for the local volatility. The class argument should be "arma".

Usage

```
iratemodel(params, class)

iratemodel.convert(from, to, irm, Delta = 1)

delta.ev(t, irm)

delta.cov(s, t, irm)

delta.var(t, irm)

y.ev(t, irm)
```

```
y.var(t, irm)

y.cov(s, t, irm)

pv.moment(t, moment, irm)

pv.ev(t, irm)

pv.var(t, irm)

pv.cov(s, t, irm)
```

Details

Define:

$$y(t) = int_0^t \delta_s$$

$$pv(t) = exp{-y(t)}$$

For each class, several functions are available.

The `delta.ev` function can be used to calculate the expected value of the interest rate process at time `t`, the `delta.var` function can be used to calculate of the interest rate process at time `t` and `delta.cov` can be used to calculate the covariance of the interest rate process at two times `s` and `t`.

The `y.ev` function can be used to calculate the expected value of the y at time `t`, the `y.var` function can be used to calculate of variance of y at time `t` and `y.cov` can be used to calculate the covariance of y at two times `s` and `t`.

The `pv.moment` can be used to calculate the raw moments of the present value random variable, `pv.ev` can be used to calculate the expected value of the present value random varaible, `pv.var` can be used to calculate the variance of the present value random variable and `pv.cov` can be used to calculate the covariance of the present value random variable at two times `s` and `t`.

The `ann.ev` function can be used to calculate the expected value of an `n`-year annuity and `ann.var` can be used to calculate the variance of an `n`-year annuity.

The `iratemodel.convert` function can be used to convert between these interest rate models using the principle of covariance equivalence.

Refer to the examples below for how to use these functions.

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

Examples

```
## 1. Deterministic interest rate
determodel = iratemodel(list(delta = 0.06), "determ")
delta.ev(5, determodel) # expected value
```

```
## 2. Wiener process
wienermodel = iratemodel(list(delta = 0.05, sigma = 0.01), "wiener")
pv.ev(10, wienermodel) # expected present value
pv.var(10, wienermodel) # variance of present value
pv.cov(5,10,wienermodel) # covariance of present value

## 3. Ornstein-Uhlenbeck process
oumodel = iratemodel(list(delta0 = 0.08, delta = 0.05,
alpha = 0.1, sigma = 0.01), "ou")
delta.ev(5, oumodel)
delta.var(5, oumodel)
y.ev(10, oumodel)
y.var(10, oumodel)
y.cov(5, 10, oumodel)
pv.ev(10, oumodel)
pv.var(10, oumodel)
pv.cov(5, 10, oumodel)

## 4. Second Order Stochastic Differential Equation
secondmodel = iratemodel(params = list(alpha1 = -0.50, alpha2 = -0.04,
delta0prime = 0.05, delta0 = 0.10, delta = 0.06, sigma = 0.01), "second")

## 5. AR(1) Process
ar1model = iratemodel(params = list(delta = 0.05, delta0 = 0.08,
phi1 = 0.90, sigma = 0.01), "ar1")
delta.ev(5, ar1model)
delta.var(5, ar1model)
delta.cov(5, 10, ar1model)
ann.ev(5, ar1model)
ann.var(5, ar1model)

## 6. ARMA(2,1) Process
armamodel = iratemodel(params = list(delta = 0.05, delta0 = 0.08,
phi1 = 1.05, phi2 = -0.095, theta1 = -0.05, sigma = 0.01), "arma")

## Convert from one model to another
sdemodel = iratemodel.convert("arma", "second", armamodel, 5)
armamodel = iratemodel.convert("second", "arma", sdemodel, 5)

oumodel = iratemodel.convert("ar1", "ou", ar1model, 1/12)
```

`isingle` *Single insurance product*

Description

A class used to describe insurance policies issued to a single life. The termsingle and endowsingle classes can be used to model term and endowment insurance contracts respectively. See the insurance class for details and examples.

Usage

```
z.moment.isingle(moment, ins, mort, irm)

z.insrisk.isingle(ins, mort, irm)

z.invrisk.isingle(ins, mort, irm)

z.pdf.isingle(z, ins, mort, irm)

## S3 method for class 'isingle.endowsingle'
z.pdf(z, ins, mort, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

MaleMort82 *CA Male Mortality rates from 1980-1982*

Description

This data set contains the mortality rates used in Parker (1992).

Usage

```
MaleMort82
```

Format

a x column for the age from 0 to 102 inclusive and a qx column for the mortality rate.

Source

Appendix B in Parker (1992).

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

`MaleMort82Reduced` *CA Male Mortality rates from 1980-1982 * 0.8*

Description

This data set contains the mortality rates used in Parker (1992).

Usage

```
MaleMort82Reduced
```

Format

a `x` column for the age from 0 to 102 inclusive and a `qx` column for the mortality rate.

References

Parker, G. (1997). Stochastic analysis of the interaction between investment and insurance risks. North American actuarial journal, 1(2), 55–71.

`MaleMort91` *CA Male Mortality rates from 1990-1992*

Description

This data set contains the 1991 mortality rates from Statistics Canada.

Usage

```
MaleMort91
```

Format

a `x` column for the age from 0 to 100 inclusive and a `qx` column for the mortality rate.

Source

Statistics Canada

mortassumptions *Mortality Assumptions*

Description

A class which describes the mortality for a policyholder. The `params` argument should be a list with `x` for the age and `table` for the table. See the examples below.

Usage

```
mortassumptions(params)

kpx(k, mort)

kdeferredqx(k, mort)

## S3 method for class 'mortassumptions'
kpx(k, mort)

## S3 method for class 'mortassumptions'
kdeferredqx(k, mort)
```

Details

The `kpx` function can be used to calculate survival probabilities and the `kdeferredqx` function can be used to calculate probability of death in a given year. See the examples below.

Examples

```
malemort = mortassumptions(list(x = 50, table = "MaleMort91"))
femalemort = mortassumptions(list(x = 30, table = "FemaleMort91"))

# calculate probabilities
kpx(5, malemort)
kdeferredqx(2, femalemort)
```

ou *The Ornstein-Uhlenbeck Process*

Description

A class used to model the Ornstein-Uhlenbeck process. See the iratemodel class for details and examples.

Usage

```
## S3 method for class 'ou'
delta.ev(t, irm)

## S3 method for class 'ou'
delta.cov(s, t, irm)

## S3 method for class 'ou'
delta.var(t, irm)

## S3 method for class 'ou'
y.ev(t, irm)

## S3 method for class 'ou'
y.var(t, irm)

## S3 method for class 'ou'
y.cov(s, t, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

---

returns91 *Investment Returns from 1991 to 2017*

Description

Company returns observed on monthly basis from 1991 to 2017

Usage

```
returns91
```

Format

A column for month and column for return (annualized). Rates are continuously compounded.

second — *Second Order Stochastic Differential Equation*

Description

A class used to model the Second Order Stochastic Differential Equation for the force of interest. See the iratemodel class for details and examples.

Usage

```
## S3 method for class 'second'
delta.ev(t, irm)

## S3 method for class 'second'
delta.cov(s, t, irm)

## S3 method for class 'second'
delta.var(t, irm)

## S3 method for class 'second'
y.ev(t, irm)

## S3 method for class 'second'
y.var(t, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

termport — *Term insurance portfolio (identical policies)*

Description

A portfolio of c identical term policies. See the insurance class for details and examples on how to use this class.

Usage

```
## S3 method for class 'iport.termport'
z.moment(moment, ins, mort, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

termsingle — *Single term insurance product*

Description

A term insurance pays a benefit of d if the policyholder dies within the term of the contract and 0 if the policyholder survives the term of the contract.

In addition to the functions described in the insurance class, other functions are available for the termsingle class.

In particular, `z.ev.two.isingle.termsingle` can be used to calculate $E[Z_1 Z_2]$, `z.ev.three.isingle.termsingle` can be used to calculate $E[Z_1 Z_2 Z_3]$ and `z.ev.twoone.isingle.termsingle` can be used to calculate $E[Z_1^2 Z_2]$ in Parker (1992).

Usage

```
## S3 method for class 'isingle.termsingle'
z.moment(moment, ins, mort, irm)

z.ev.two.isingle.termsingle(ins, mort, irm)

z.ev.three.isingle.termsingle(ins, mort, irm)

z.ev.twoone.isingle.termsingle(ins, mort, irm)

## S3 method for class 'isingle.termsingle'
z.insrisk(ins, mort, irm)

## S3 method for class 'isingle.termsingle'
z.invrisk(ins, mort, irm)

## S3 method for class 'isingle.termsingle'
z.pdf(z, ins, mort, irm)
```

Details

The `z.insrisk` and `z.invrisk` functions for the termsingle class are implemented such that we are conditioning on K not y. See Parker (1997) for details.

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

Parker, G. (1997). Stochastic analysis of the interaction between investment and insurance risks. North American actuarial journal, 1(2), 55–71.

Examples

```
oumodel = iratemodel(list(delta0 = 0.1, delta = 0.06,
alpha = 0.1, sigma = 0.01), "ou")
mort = mortassumptions(list(x = 40, table = "MaleMort91"))
term = insurance(list(n = 10, d = 1), "isingle", "term")

z.ev(term,mort,oumodel) # first moment
z.ev.two.isingle.termsingle(term, mort, oumodel)
z.ev.twoone.isingle.termsingle(term, mort, oumodel)
z.ev.three.isingle.termsingle(term, mort, oumodel)
```

`wiener` *The Wiener Process*

Description

A class used to model the Wiener process. See the iratemodel class for details and examples.

Usage

```
## S3 method for class 'wiener'
delta.ev(t, irm)

## S3 method for class 'wiener'
delta.cov(s, t, irm)

## S3 method for class 'wiener'
delta.var(t, irm)

## S3 method for class 'wiener'
y.ev(t, irm)

## S3 method for class 'wiener'
y.cov(s, t, irm)

## S3 method for class 'wiener'
y.var(t, irm)
```

References

Parker, Gary. An application of stochastic interest rate models in life assurance. Diss. Heriot-Watt University, 1992.

Index