Package ‘takos’

October 14, 2022

Type Package

Title Analysis of Differential Calorimetry Scans

Version 0.2.0

Author Serena Berretta [aut] and Giorgio Luciano [aut,cre], Kristian Hovde Liland [ctb]

Maintainer Serena Berretta <serena.berretta@ge.imati.cnr.it>

Description It includes functions for applying methodologies utilized for single-process kinetic analysis of solid-state processes were recently summarized and described in the Recommendation of ICTAC Kinetic Committee. These methods work with the basic kinetic equation. The Methodologies included refers to Avrami, Friedman, Kissinger, Ozawa, OFM, Mo, Starink, isoconversional methodology (Vyazovkin) according to ICATAC Kinetics Committee recommendations as reported in Vyazovkin S, Chrissafis K, Di Lorenzo ML, et al. ICTAC Kinetics Committee recommendations for collecting experimental thermal analysis data for kinetic computations. Thermochim Acta. 2014;590:1-23. <doi:10.1016/J.TCA.2014.05.036> .

Imports MASS,devEMF,segmented,sfsmisc,smoother,deSolve, pracma,data.table,broom,colorRamps, minpack.lm, tools, baseline, graphics

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 6.1.1

URL https://github.com/sere3s/takos

NeedsCompilation no

Repository CRAN

Date/Publication 2020-10-19 09:50:02 UTC

R topics documented:

`addRate` *Title addRate*

Description

add to the thermogram the value of rate(s) for each cycle(s) as provided by the user

Usage

```
addRate(dat, lab_rate, lab_cycles)
```

Arguments

`dat`	matrix
`lab_rate`	rate that corresponds to the cycles of the analysis performed
`lab_cycles`	number of the cycles of the analysis performed

Value

the input matrix with one added column with the values of the rate of the cycles performed

---

`avrami` *Title Avrami*

Description

performs analysis of the thermograms using the avrami method

Usage

```
avrami(mat)
```

Arguments

`mat`	matrix of the all the thermograms checked using the functiom mat.check

Value

models "mod", datable "xy" for plot

References

1. Avrami M. Kinetics of Phase Change. I General Theory. J Chem Phys. 1939;7(12):1103-1112. doi:10.1063/1.1750380.

Examples

```
require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
ar<-testMat(a)
avr<-avrami(ar)
```

checmat *Title checkmat*

Description

Title checkmat

Usage

```
checkmat(dat, header = TRUE, selected = c(0, 1, 2, 0, 0, 0, 4, 0))
```

Arguments

`dat`	MUST be a data.frame where each column represent a parameter of the thermogram you need to check
`header`	present or not in your data.frame
`selected`	a vector that include the coded position of the parameters present in the dataset. 0 equal not present, while if you insert a number its value will refer to the index of the column of the input matrix where the parameter is stored. the coding of the vector selected is the following 1. "time.minutes" 2. "time.seconds" 3."temperature.s" 4."temperature.r" 5."temperature.s.K" 6."temperature.r.K"7."heat.flow"8. "id"

Details

i.e. selected=c(1,0,2,0,0,0,3) means that your first column is time.seconds, the second column is the temperature of the sample and the this column is the heat flow. 0 represents the other column of your files that are not present in your dataset

Value

Checked data frame

Examples

```
npoints=1000
x=seq(1,npoints)
y=(dnorm(x, mean=npoints/2, sd=npoints/10))
x=seq(1,1000)
x2=seq(200,500,length.out=1000)
dat=data.frame(x,x2,y)
colnames(dat) <- c("time.seconds", "temperature.s","heat.flow")
cmat<- checkmat(dat,selected=c(1,0,2,0,0,0,3,0))
```

cutSelect *Title cutSelect*

Description

cut a region of a spectra and substitutes it with a sequence with initial value i.start and end valye i.end

Usage

```
cutSelect(x, i.start, i.end)
```

Arguments

`x`	x to be cut
`i.start`	index value of the starting point for the cut to be performed
`i.end`	index value of the ending point for the cut to be performed

Examples

```
npoints=1000
x=seq(1,npoints)
y=(dnorm(x, mean=npoints/2, sd=npoints/10))
ycut=cutValue(y,10,40,0.003,0.001)
plot(y)
lines(ycut,col="red")
```

cutValue *Title cutValue*

Description

cut a region of a spectra and substitutes it with a sequence with initial value i.start and end valye i.end

Usage

```
cutValue(x, i.start, i.end, value.start, value.end)
```

Arguments

`x`	to be cut
`i.start`	index value of the starting point for the cut to be performed
`i.end`	index value of the ending point for the cut to be performed
`value.start`	desired value at point i.start
`value.end`	desired value at point i.end

Value

x after cut

Examples

```
npoints=1000
x=seq(1,npoints)
y=(dnorm(x, mean=npoints/2, sd=npoints/10))
ycut=cutSelect(y,10,40)
plot(y)
lines(ycut,col="red")
```

dadx *Title dadx*

Description

calculates the ratio of two differential according to the value of d.step

Usage

```
dadx(x, a, d.step = 2)
```

Arguments

`x`	denominator variable for calculating da
`a`	numerator variable for calculating dt
`d.step`	step of differentiation

Value

ratio of two differential of the tw0 input variables

Examples

```
npoints=100
seed=42
x1=round(runif(npoints,0,1), 2)
seed=1234
x2=round(runif(npoints,0,1), 2)
xdiff <- dadx(x1,x2)
```

FRI *Title Friedman*

Description

performs analysis of the thermograms using Friedman method to calculate the activation energy (Ea)

Usage

```
FRI(mat, id = "rate", degree = seq(0.2, 0.8, by = 0.05))
```

Arguments

`mat`	matrix of the all the thermograms checked using the functiom mat.check
`id`	variable choosen for subsetting mat (default = "rate")
`degree`	selected degrees of cristallinity for performing the analysis

Value

models "mod", datable "xy" for plot, "Ea" list of value, datatable "DT" built with the values of mat according to the specified degree

References

H.L. Friedman, Kinetics of thermal degradation of char-forming plastics from thermogravimetry, Appl. Phen. Plastic J. Polym. Sci. Part C: Polym. Symp. 6 (1964)

Examples

```
require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
ar<-testMat(a)
fri<-FRI(ar)
```

gAC *Title sbAC*

Description

Performs simulation according to several kinetic models

Usage

```
gAC(time.start = 0, T0 = 0, T.end = 500, qqq = 50, A = 10^(6.3),
  Ea = 80000, m = 1, n = 2, K = 0, npoints = 10000,
  prec = 10^(-4.30095790876), rmod = "SB", ...)
```

Arguments

`time.start`	Starting time for the simulations
`T0`	Temperature start
`T.end`	End temperature
`qqq`	Heating rate
`A`	Parameter in the equation
`Ea`	Parameter in the equation
`m`	Parameter in the equation
`n`	Parameter in the equation
`K`	Parameter in the equation
`npoints`	Number of points
`prec`	Starting value for the equation "prec"
`rmod`	Kinetic model (default = Isoda)
`...`	Parameters to pass to ode function for choosing solver method

Value

startgin temperature "T","fi",degree of crystallization "alfa",differential alfa in T "dadT",time in seconds "time.s",differential equation solution "sol"

Examples

```
gAC(npoints=5000,prec=10^(-4.30095790876))
```

JMA *Title Johnson-Mehl-Avrami (JMA)*

Description

simulate a thermogram using JMA theory

Usage

```
JMA(A = exp(35), Ea = 120000, q = 50, T0 = -100, T.end = 300,
  npoints = 898, n = 2)
```

Arguments

`A`	pre exponential parameters (1/s)
`Ea`	Activation energy (J/mol)
`q`	= rate of analyis (K/min)
`T0`	= starting temperature of the simulated thermogram expressed in K
`T.end`	= ending temperature of the simulated thermogram expressed in K
`npoints`	desired number of points of the simulate thermogram
`n`	numerical parameter required by JMA model

Value

- T.C = temperature in Celsius
- $fi = d\alpha/dt * q$
- α
- time.s = time in second
- $d\alpha dT$

References

1. Vyazovkin S, Chrissafis K, Di Lorenzo ML, et al. ICTAC Kinetics Committee recommendations for collecting experimental thermal analysis data for kinetic computations. Thermochim Acta. 2014;590:1-23. doi:10.1016/j.tca.2014.05.036.

Examples

```
data <- JMA(A = exp(35),Ea = 120000,q = 50,T0 = -100,T.end = 300,npoints=898,n=2)

require(data.table)
#choose the rates for the simulation of the thermograms
rates=c(0.5,1,2,5,10,20,50)
#first serie of thermograms for all the chosen rate
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
```

```
#setup column names
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,
a[[x]]$T.C, a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
#create a plot using the function thermo
amaxH <- max(sapply(a, function(x) max(x$heat.flow))) # calculate the max
plot(c(0,300),c(0,amaxH),mytitle="dataset A 120/60 0.66/0.33",
ylab="ExothermicHeatFlow", xlab="Temperature")
lapply(a, function(x) lines(x$temperature.s,x$heat.flow,lwd=3))
```

`KAS` *Title KAS*

Description

performs analysis of the thermograms using Kissinger-Akahira-Sunose (KAS) method

Usage

```
KAS(mat, degree = seq(0.2, 0.8, by = 0.05))
```

Arguments

`mat`	matrix of the all the thermograms checked using the functiom mat.check
`degree`	selected degrees of cristallinity for performing the analysis

Value

models "mod", datable "xy" for plot, "Ea" list of value, datatable "DT" built with the values of mat according to the specified degrees

References

1. Akahira, T. Sunose T. Method of determining activation deterioration constant of electrical insulating materials. Res Rep Chiba Inst Technol (Sci Technol). 1971. 2. Kissinger HE. Reaction Kinetics in Differential Thermal Analysis. Anal Chem. 1957;29(11):1702-1706. doi:10.1021/ac60131a045. 3. Starink M. The determination of activation energy from linear heating rate experiments: a comparison of the accuracy of isoconversion methods. Thermochim Acta. 2003;404(1-2):163-176. doi:10.1016/S0040-6031(03)00144-8.

Examples

```
require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
ar<-testMat(a)
kas<-KAS(ar)
```

Kiss *Title Kissinger*

Description

performs analysis of the thermograms using Kissinger method to calculate the activation energy (Ea)

Usage

```
Kiss(mat)
```

Arguments

mat	matrix of the all the thermograms checked using the functiom mat.check

Value

models "mod", datable "xy" for plot, "Ea" list of value, datatable "DT" built with the values of mat according to the specified degrees

References

1. Avrami M. Kinetics of Phase Change. I General Theory. J Chem Phys. 1939;7(12):1103-1112. doi:10.1063/1.1750380. 2. Kissinger HE. Reaction Kinetics in Differential Thermal Analysis. Anal Chem. 1957;29(11):1702-1706. doi:10.1021/ac60131a045.

Examples

```
require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
ar<-testMat(a)
kiss<-Kiss(ar)
```

lavrami *Title Avrami Linearization*

Description

performs analysis of the thermograms using the linearized avrami method in the interval of Xc selected by the user

Usage

```
lavrami(mat, up = 0.9999, low = 1e-04)
```

Arguments

`mat`	matrix of the all the thermograms checked using the functiom mat.check
`up`	max degree of the interval for applying the linearized model default 0.9999
`low`	min degree of the interval for applying the linearized model default 0.0001

Value

models "mod", datable "xy" for plot

References

1. Avrami M. Kinetics of Phase Change. I General Theory. J Chem Phys. 1939;7(12):1103-1112. doi:10.1063/1.1750380.

matzero *Title matzero*

Description

zeroes time (in seconds) according to peak given by the user

Usage

```
matzero(mat, spks = 1, x = mat$time.seconds.zero, colname = "v.check",
  myby = "id_cycle")
```

Arguments

`mat`	matrix of spectra
`spks`	number of the peak selected as the starting point
`x`	variable to be reset according to the position of the selected peak
`colname`	name of the selected column
`myby`	varialbe selected for subsetting the matrix

MO *Title Mo model*

Description

performs analysis of the thermograms using Mo method

Usage

```
MO(mat, degree = seq(0.2, 0.8, by = 0.2))
```

Arguments

`mat`	matrix of the all the thermograms checked using the functiom mat.check
`degree`	selected degrees of cristallinity for performing the analysis

Value

models "mod", datable "xy" for plot, "Ea" list of value, datatable "DT" built with the values of mat according to the specified degrees

References

Liu T, Mo Z, Wang S, Zhang H. Nonisothermal melt and cold crystallization kinetics of poly(aryl ether ether ketone ketone). Polym Eng Sci. 1997;37(3):568-575. doi:10.1002/pen.11700.

Examples

```
require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
ar<-testMat(a)
mo<-MO(ar)
```

OFW *Title OFW*

Description

performs analysis of the thermograms using Ozawa-Flynn and Wall method

Usage

```
OFW(mat, degree = seq(0.2, 0.8, by = 0.05))
```

Arguments

mat	matrix of the all the thermograms checked using the functiom mat.check
degree	selected degrees of cristallinity for performing the analysis

Value

models "mod", datable "xy" for plot, "Ea" list of value, datatable "DT" built with the values of mat according to the specified degrees

References

1. Flynn J, Wall L. Res natl bur standards. Phys Chem. 1966;70:487-492.

Examples

```
require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
ar<-testMat(a)
ofw<-OFW(ar)
```

OZ *Title Ozawa model crystallization*

Description

performs analysis of the thermograms using Ozawa method

Usage

```
OZ(mat, n.step = 1, spks = 1, eps = 0.001)
```

Arguments

mat	matrix of the all the thermograms checked using the functiom mat.check
n.step	number of steps for selecting temperature ranges
spks	id of the peaks selected for applying the method
eps	tollerance for the selection process

Value

models "mod", datable "xy" for plot, "Ea" list of value, datatable "DT" built with the values of mat according to the specified degrees

References

1. Ozawa T. Kinetics of non-isothermal crystallization. Polymer (Guildf). 1971;12(3):150-158. doi:10.1016/0032-3861(71)90041-3.

Examples

```
require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
ar<-testMat(a)
oz<-OZ(ar)
```

`plot_avrami` *Title plot.avrami*

Description

template for plotting results from avrami function

Usage

```
plot_avrami(out, skip = 2)
```

Arguments

`out`	output from avrami function
`skip`	plot symbols every nth points

`plot_fri` *Title plot.fri*

Description

template for plotting results from friedman function

Usage

```
plot_fri(out)
```

Arguments

`out`	from friedman function

`plot_lavrami` *Title plot lavrami*

Description

template for plotting results from avrami function

Usage

```
plot_lavrami(out, skip = 2)
```

Arguments

`out`	output from lavrami function
`skip`	plot symbols every nth points

`plot_mo` *Title plot.mo*

Description

template for plotting results from Mo function

Usage

```
plot_mo(out)
```

Arguments

`out`	from mo function

`plot_ozawa` *Title plot.ozawa*

Description

template for plotting results from avrami function

Usage

```
plot_ozawa(out)
```

Arguments

`out`	from ozawa function

`ri` *Title running integral*

Description

calculate the running integral for the selected peak

Usage

```
ri(x, y, pks, TAP = FALSE, linear = FALSE, ...)
```

Arguments

`x`	x axis for the intergration
`y`	y axis for the intergration
`pks`	selected peak
`TAP`	if TRUE will apply a baseline using tangent area proportional (default=FALSE)
`linear`	if TRUE will apply a linear baseline (default=FALSE)
`...`	parameters in TAPPA function

Value

- ds data frame containing original x and y given as input
- ri running integral
- b.tap baseline calculate if the switch TAP is TRUE
- y.tap = y - b.tap

Examples

```
#' require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
a.dt <-lapply(seq(1,length(a)), function(x) data.table(data.frame(a.check[[x]])))
a<-rbindlist(a.dt)
a$rate<-a$id
a.peaks <- a[,.(res.list = list(findpeaks(heat.flow,sortstr=TRUE,npeaks=2))),by=id]
a.peaks$rate<-a.peaks$id
ref.peak=1
a.peaks <- data.table(data.table(a.peaks$rate),rbindlist((lapply(a.peaks$res.list,
function(x) data.table(t(x[ref.peak,]))))))
colnames(a.peaks)<- c("rate","peak.value","ind.max","left.lim","right.lim")
a.mat<- lapply(unique(a$rate),function(x)
ri(a[a$rate==x]$time.seconds,a[a$rate==x]$heat.flow,a.peaks[rate==x]))
```

`runningIntegral` *Title running integral*

Description

calculates the running integral for customer input

Usage

```
runningIntegral(x, y, integrate.step = 1)
```

Arguments

`x`	variable x use for integration process
`y`	variable y use for integration process
`integrate.step`	= the step used for calculating the integrale the default value is 1

Examples

```
npoints=1000
x=seq(1,npoints)
y=(dnorm(x, mean=npoints/2, sd=npoints/10))
runningIntegral(x,y)
```

`sbAC` *Title sbAC*

Description

Performs Šesták-Berggren (AC) simulations

Usage

```
sbAC(time.start = 0, T0 = 0, T.end = 500, qqq = 50, A = 10^(6.3),
  Ea = 80000, m = 1, n = 2, npoints = 10000,
  prec = 10^(-4.30095790876), ...)
```

Arguments

`time.start`	Starting time for the simulations
`T0`	Temperature start
`T.end`	End temperature
`qqq`	Heating rate
`A`	Parameters in the equation
`Ea`	Parameters in the equation
`m`	Parameter in the equation
`n`	Parameter in the equation
`npoints`	Number of points
`prec`	Starting value for the equation "prec"
`...`	Parameters to pass to ode function for choosing solver method

Value

startgin temperature "T","fi",degree of crystallization "alfa",differential alfa in T "dadT",time in seconds "time.s",differential equation solution "sol"

References

J. Šesták. Thermophysical Properties of Solids, Their Measurements and Theoretical Analysis. Elsevier: Amsterdam, 1984.

Examples

```
res <- sbAC(npoints=5000,prec=10^(-4.30095790876))
```

---

select_degree *Title*

Description

Title

Usage

```
select_degree(mat, degree = seq(0.01, 0.99, by = 0.01))
```

Arguments

mat	matrix of the all the thermograms checked using the functiom mat.check
degree	selected degrees of cristallinity for performing the analysis

Value

"DT" built with the values of mat according to the specified degrees

simG *Title simG*

Description

create a simulated spectra with gaussian shape

Usage

```
simG(vlen, i.start, gheight, shift = 0, wd = 30)
```

Arguments

`vlen`	desired length of the spectra
`i.start`	starting value for the peak
`gheight`	height value
`shift`	shift from 0
`wd`	width of the gaussian curve

Examples

```
y=(simG(500,35,1,0,w=20))
plot(y)
```

smooth.loess *Title smooth.loess*

Description

a wrapper for the loess function included in the R base system

Usage

```
smooth.loess(x, y, safe.start = 5, safe.end = 5, myspan = 0.28)
```

Arguments

`x`	variable x
`y`	variable y
`safe.start`	exclude a the n-th first values from calculation
`safe.end`	exclude a the n-th end values from calculation
`myspan`	span parameter for loess function

Examples

```
npoints=1000
x=seq(1,npoints)
y=(dnorm(x, mean=npoints/2, sd=npoints/10))
y.smooth=smooth.loess(x,y)
plot(x,y)
```

`Starink` *Title Staink*

Description

performs analysis of the thermograms using Starink method

Usage

```
Starink(mat, degree = seq(0.2, 0.8, by = 0.05))
```

Arguments

`mat`	matrix of the all the thermograms checked using the functiom mat.check
`degree`	selected degrees of cristallinity for performing the analysis

Value

models "mod", datable "xy" for plot, "Ea" list of value, datatable "DT" built with the values of mat according to the specified degrees

References

Starink MJ. A new method for the derivation of activation energies from experiments performed at constant heating rate. Thermochim Acta. 1996;288(1-2):97-104. doi:10.1016/S0040-6031(96)03053-5.

Examples

```

require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
ar<-testMat(a)
star<-Starink(ar)
```

summaryTableA *table.avrami*

Description

examples of functions for presenting the results obtained with different methods Title summary-Table A

Usage

```
summaryTableA(mat.mod)
```

Arguments

`mat.mod`	output matrix from avrami function

Value

table with the summary of result of applying the avrami function on the selected thermograms

summaryTableFri *Title summaryTableF*

Description

Title summaryTableF

Usage

```
summaryTableFri(mat.mod)
```

Arguments

`mat.mod`	output matrix from friedman function

Value

table with the summary of result of applying the friedman function on the selected thermograms

`summaryTableKiss` *Title summaryTableK*

Description

Title summaryTableK

Usage

```
summaryTableKiss(mat.mod)
```

Arguments

`mat.mod`	output matrix from Starink function

Value

table with the summary of result of applying the starink function on the selected thermograms

`summaryTableMo` *Title summaryTable Mo*

Description

Title summaryTable Mo

Usage

```
summaryTableMo(mat.mod)
```

Arguments

`mat.mod`	output matrix from Mo function

Value

table with the summary of result of applying the ozawwa function on the selected thermograms

`summaryTableOz` *Title summaryTableOz*

Description

Title summaryTableOz

Usage

```
summaryTableOz(mat.mod)
```

Arguments

`mat.mod`	output matrix from ozawa function

Value

table with the summary of result of applying the ozawwa function on the selected thermograms

`TAPPA` *Title Tangent area proportional method TAPPA*

Description

calculates the background of a thermogram according to Tangent-area-proportional method

Usage

```
TAPPA(T, dAlpha, interval = 10, tol = 0.001)
```

Arguments

`T`	temperature
`dAlpha`	the da/dt values
`interval`	number of points to use for interpolating the two lines that will merge according to the area of the peak
`tol`	tollerance for the iterative process

Value

B baseline values

References

1. Svoboda R. Tangential area-proportional baseline interpolation for complex-process DSC data - Yes or no? Thermochim Acta. 2017;658:55-62. doi:10.1016/J.TCA.2017.10.011.2. Svoboda R. Linear baseline interpolation for single-process DSC data-Yes or no? Thermochim Acta. 2017;655:242-250. doi:10.1016/J.TCA.2017.07.008.

Examples

```
npoints=1000
x=seq(1,npoints)
y=(dnorm(seq(1,npoints), mean=npoints/2, sd=npoints/10)) #simulated peak
y2=y+(dnorm(seq(1,npoints), mean=npoints, sd=npoints/10)) #secondary simulated peak
y2[seq(npoints*0.735,npoints)]=y2[763] #flat the curve at the end of first peak
ytap=TAPPA(x,y2)
plot(x,y2)
lines(x,ytap,col="red")
```

`testMat` *Title testMat*

Description

Title testMat

Usage

```
testMat(a, l.lim = 1, r.lim = NULL, toselect = c(0, 1, 2, 0, 0, 0, 3, 4))
```

Arguments

`a`	list of data tables of the checked thermograms using checkmat , obtained at different rates to change lines
`l.lim`	left lim of running integral
`r.lim`	rigth lim of running integral
`toselect`	vector

Value

data table ready to be used by all the methods for kinetic analysis included in the package

Examples

```
require(data.table)
npoints=1000
x=seq(1,npoints)
y=(dnorm(x, mean=npoints/2, sd=npoints/10))
x=seq(1,1000)
x2=seq(200,500,length.out=1000)
```

```
dat=data.frame(x,x2,y)
colnames(dat) <- c("time.seconds", "temperature.s","heat.flow")
dat=data.table(dat)
dat2=dat
dat$rates=20
dat2$rates=50
toTest=list(dat,dat2)
tested=testMat(toTest)
```

`t_baseline` *Title t.baseline*

Description

a wrapper for the baseline.rdfbaseline function in the package baseline in order to have the output in the same format as the input

Usage

```
t_baseline(y)
```

Arguments

`y`	baseline correction on y

Value

y.baseline returns the corrected y

Examples

```
y.baseline <- t_baseline(y)
```

`VY` *title Vyazovkin*

Description

performs analysis of the thermograms using Vyazovkin isoconversional method to calculate the activation energy (Ea)

Usage

```
VY(T, bet, Ea)
```

Arguments

T	temperature
bet	rate
Ea	estimated Ea to use as a first guess for the iterative process

References

VYAZOVKIN, S. Advanced isoconversional method. Journal of thermal analysis, 1997, 49.3: 1493-1499.

Examples

```
require(data.table)
require(MASS)
rates=c(0.5,1,2,5,10,20,50)
a<-lapply(rates, function(x) JMA(A=exp(35),Ea=120000,T0=0,T.end=300,q=x,npoints=5000,n=2))
a<-lapply(seq(1,length(a)), function(x) data.table(a[[x]]$time.s,a[[x]]$T.C,
a[[x]]$dadT, rates[[x]]))
lapply(seq(1,length(a)), function(x) setnames(a[[x]],
c("time.seconds","temperature.s","heat.flow","rates") ) )
as<-select_degree(ar)
vy<-as[, optimize(function(x) VY(temperature.s.K,rate,x), lower=50,upper=250),by=rit]
```

Index