Package 'Biotech'

April 17, 2020

Title Diverse applications for Biotechnologies

Version 0.0.0.9001

Description Software for Biotechnology students with the aim to popularise the usage of R as great tool for data-handling and manipulation

Author Utz Rieger <s71564@beuth-hochschule.de>

Maintainer Utz Rieger <s71564@beuth-hochschule.de>

License MIT + file LICENSE

Encoding UTF-8

LazyData true

Suggests testthat

RoxygenNote 7.1.0

Imports ggplot2, magrittr

R topics documented:

`c.any` *Compute the concentration of the source salts for specific elements inside a fermentationbroth*

Description

Compute the concentration of the source salts for specific elements inside a fermentationbroth

Usage

```
## S3 method for class 'any'
c(ds, ele = c("P", "N", "K", "Mg"), org = c("Bacteria", "Fungus"))
```

Arguments

`ds`	dry substance mass in g/l
`ele`	element abreveation as string
`org`	organism as string

Value

concentration of salt within the broth

Examples

```
# c.any(ds = 50, ele = 'P', org = 'Fungus')
# c.any(ds = 50, ele = "N", org = 'Fungus')
```

CellVit — *Compute the cell vitality*

Description

Compute the cell vitality

Usage

```
CellVit(unstained, stained)
```

Arguments

unstained	cell count of unstained cells
stained	cell count of stained cells

Value

vitality

Examples

```
# we use a real dataset:
us <- c(55, 65, 49, 57)
st <- c(11, 12, 9, 6)

CellVit(unstained = us, stained = st)
```

conc.4PL — *Compute the concentration of an competitive ELISA based on the 4-Parameter-logistic function*

Description

Compute the concentration of an competitive ELISA based on the 4-Parameter-logistic function

Usage

```
conc.4PL(std.conc, std.resp, mes.resp)
```

Arguments

std.conc	the concentration of the calibrators
std.resp	the measured absorbance of the calibrators
mes.resp	the measured absorbance of the samples of unknown concentration

Value

the model-fit and the concentration of the samples

conc.from.abs — *Functions to compute concentration values in absorption-spectroscopy*

Description

sum This function possesses the ability to return the concentration of an unknown sample by comparing its absorption to a linear regression model

Usage

```
conc.from.abs(abs, epsilon, cuvette = 1, dfac = 1)

conc.from.abs(abs, epsilon, cuvette = 1, dfac = 1)
```

Arguments

abs	measured absorption
epsilon	molar attenuation coefficient (without default)
cuvette	cuvette factor, default is 1
dfac	dilution factor, default is 1

Value

The concentration

conc.per.dil — *Compute the concentration after a dilution step*

Description

Compute the concentration after a dilution step

Usage

```
conc.per.dil(vol.pre, vol.post, conc.pre)
```

Arguments

vol.pre	volume in advance of the dilution step
vol.post	volume after the dilution step
conc.pre	concentration in advance of the dilution

Value

current concentration of the dilution after the dilution step

`conc_eval` *concentration evaluation*

Description

concentration evaluation

Usage

```
conc_eval(abs_P, abs_std, conc_std)
```

Arguments

`abs_P`	absorption of a unknown sample
`abs_std`	absorption of calibrationstandards
`conc_std`	concentration of calibrationstandards

Value

float

`dilfact` *Computing a dilution factor, a sample volume (Vorlagevolumen), concentration after dilution Compute the dilution factor*

Description

Computing a dilution factor, a sample volume (Vorlagevolumen), concentration after dilution Compute the dilution factor

Usage

```
dilfact(vol.1, vol.2)
```

Arguments

`vol.1`	V1 use same metrics
`vol.2`	V2 use same metrics

Value

FV a dilution factor

Examples

```
dilfact(1, 100)
```

dose_response_plot *A dose~response plot with a summary*

Description

A dose~response plot with a summary

Usage

```
dose_response_plot(conc, resp, xlab = "conc", ylab = "resp")
```

Arguments

conc	the concentrations
resp	the response of the system for conc
xlab	lable for the abscissa
ylab	lable for the ordinate

Value

a plot

Eadie_Hofstee *Draw an Eadie-Hofstee graph (and compute the y-axis intercept)*

Description

Draw an Eadie-Hofstee graph (and compute the y-axis intercept)

Usage

```
Eadie_Hofstee(
  vel,
  sub,
  titleEDH = "Eadie-Hostee-Plot",
  xlable = "vel/sub",
  ylable = "sub",
  printfigure = TRUE
)
```

Arguments

vel	Velocity
sub	Substarate concentration
titleEDH	Title of the plot
xlable	lable of the abscissa
ylable	lable of the ordinate

Value

a Plot, the model itself and a summary of the model

Examples

```
# simulate values
sub <-seq(1,20,1)
vel <-((runif(1,14.7,15)*sub)/(runif(1,2.5,3)+sub))+rnorm(20,0,.3)
# plot them
Eadie_Hofstee(vel = vel, sub = sub)
```

---

eG *Compute gas content Keep an eye on the units*

Description

Compute gas content Keep an eye on the units

Usage

```
eG(VG, VL)
```

Arguments

VG	volume of gas
VL	volume of liquid

Value

eG or the gas content

Examples

```
eG( VG = c(1, 2, 3), VL = c(6, 12, 24) )
```

---

eG.empir *Empiric prediction function for the Gas holdup (eG)*

Description

Empiric prediction function for the Gas holdup (eG)

Usage

```
eG.empir(uG)
```

Arguments

uG	superficial velocity

Value

gas holdup prediction

---

eGR_CMC *gas content upstream column for non nowton fluids*

Description

gas content upstream column for non nowton fluids

Usage

```
eGR_CMC(K2 = 0.465, a2 = 0.65, b2 = -1.06, c2 = -0.103, eta.eff, uGR, AD, AR)
```

Arguments

K2	constant of proportionality
a2	specific coefficient
b2	specific coefficient
c2	specific coefficient
eta.eff	dynamic viscosity
uGR	superficial velocity (riser)
AD	surface area downer
AR	surface area riser

Value

eGR gas content upstream column for non nowton fluids

---

eGR_H2O *gas content of upstream column for a bubble column reactor*

Description

gas content of upstream column for a bubble column reactor

Usage

```
eGR_H2O(K1, uGR, m1)
```

Arguments

K1	specific coefficient
uGR	superficial velocity (riser)
m1	specific coefficient

Value

gas content upstream column for water

---

eGT — *compute the total gas content (eGT) of an airlift-reactor with a single riser and a single downer*

Description

compute the total gas content (eGT) of an airlift-reactor with a single riser and a single downer

Usage

```
eGT(eGR, eGD, VDD, VDR)
```

Arguments

eGR	gas content of the riser
eGD	gas content of the downer
VDD	dispersion volume of the downer
VDR	dispersion volume of the riser

Value

total gas content (eGT)

---

eta.eff — *rheological modeling for highly non-newtonian fluids estimate of Ostwald & de Waele*

Description

rheological modeling for highly non-newtonian fluids estimate of Ostwald & de Waele

Usage

```
eta.eff(Kc = (0.3^0.7), uGR, m)
```

Arguments

Kc	consistency index
uGR	superficial velocity (riser)
m	index

Value

dynamic viscosity

`K.Zlokarnik` *Compute K and m from Zlokarnik-Korrelation*

Description

Compute K and m from Zlokarnik-Korrelation

Usage

```
K.Zlokarnik(
  kLa,
  power,
  vol.rea,
  vel.gas,
  rho = 998,
  gforce = 9.81,
  eta = 0.724
)
```

Arguments

`kLa`	kLa
`power`	power input in Watt
`vol.rea`	volume of the reactor vessel in m^3
`vel.gas`	gas velocity in l/min
`rho`	the density of the fermentation broth
`gforce`	the g-force
`eta`	the viscosity

Value

K and m

`kLa.eG` *compute kLa via eG (gas holdup)*

Description

compute kLa via eG (gas holdup)

Usage

```
kLa.eG(eG, K = 0.61, p = 0.83)
```

Arguments

eG	gas holdup (eG)
K	a specific factor, default is 0.61
p	K a specific factor, default is 0.81

Value

the kLa via eG

---

kLaD *Title*

Description

Title

Usage

```
kLaD(kLa, eGt)
```

Arguments

kLa	kLa
eGt	gas content

Value

diffusion coefficient

---

kLaD.korr.H2O *Title*

Description

Title

Usage

```
kLaD.korr.H2O(K3 = 0.076, AD, AR, a3 = -2, uGR, b3 = 0.8)
```

Arguments

K3	constant of proportionality
AD	surface area downer
AR	surface area riser
a3	specific coefficient
uGR	superficial velocity (riser)
b3	specific coefficient

kLaD.korr.nN *non-newton-korrelation model for the kLa*

Description

non-newton-korrelation model for the kLa

Usage

```
kLaD.korr.nN(
  K4 = (0.5 * (10^-2)),
  uGR,
  a4 = 0.65,
  DGL = (2.7 * 10^(-9)),
  b4 = 0.5,
  rho.liq = 1020,
  c4 = 1.03,
  AD,
  AR,
  d4 = 0.85,
  eta.eff,
  e4,
  sigma.liq = 0.071,
  f4
)
```

Arguments

K4	constant of proportionality
uGR	superficial velocity (riser)
a4	specific coefficient
DGL	diffusion coefficient of the gas within the liquid phase
b4	specific coefficient
rho.liq	density of the liquid
c4	specific coefficient
AD	surface area downer
AR	surface area riser
d4	specific coefficient
eta.eff	dynamic viscosity
e4	specific coefficient
sigma.liq	surface tension of the liqud
f4	specific coefficient

Lineweaver_Burk *Plot a Lineweaver-Burk diagram and compute the ordinate intercept*

Description

Plot a Lineweaver-Burk diagram and compute the ordinate intercept

Usage

```
Lineweaver_Burk(
  sub,
  vel,
  title = "Lineweaver-Burk-Plot",
  xlab = "1/sub",
  ylab = "1/vel"
)
```

Arguments

sub	substrate concentration
vel	enzyme velocity
title	title of the plot
xlab	lable of the abscissa
ylab	lable of the ordinate

LinMod_CEv *Linear model for concentration evaluation*

Description

This must be performed in advanvce of usage of the function conc_eval to produce a linear model suiteable for concentration evaluation.

Usage

```
LinMod_CEv(abs, conc)
```

Arguments

abs	list
conc	list

Value

Dataframe

`mix.t.H2O` *mixing time of H2O based on korrelation*

Description

mixing time of H2O based on korrelation

Usage

```
mix.t.H2O(K5, uGR, m5)
```

Arguments

`K5`	constant of proportionality
`uGR`	superficial velocity (riser)
`m5`	specific coefficient

Value

mixing time of H2O based on korrelation

`mix.t.nN` *Title*

Description

Title

Usage

```
mix.t.nN(
  K6 = 571,
  DR,
  a6 = -0.5,
  AD,
  AR,
  b6 = -0.16,
  h.disp.Riser,
  c6 = -1.44,
  uGR,
  d6 = -0.46,
  eta.eff,
  e6 = 0.56
)
```

Arguments

K6	constant of proportionality
DR	diamter of the riser
a6	specific coefficient
AD	surface area downer
AR	surface area riser
b6	specific coefficient
h.disp.Riser	height of the liquid (riser) gased
c6	specific coefficient
uGR	superficial velocity (riser)
d6	specific coefficient
eta.eff	dynamic viscosity
e6	specific coefficient

MM_typ *Plot a non linear regression for Michaelis-Menten type enzyme*

Description

Plot a non linear regression for Michaelis-Menten type enzyme

Usage

```
MM_typ
```

Format

An object of class `formula` of length 3.

---

mt *compute the mixing time*

Description

compute the mixing time

Usage

```
mt(H, D, uG, gforce = 9.81)
```

Arguments

H	height of the reactor
D	diameter of the reactor
uG	superficial velocity
gforce	g-force, default is 9.81

Value

the mixing time

`MultiRate` *Compute the cell vitality after frosting*

Description

Compute the cell vitality after frosting

Usage

```
MultiRate(unstained_defro, unstained_24h)
```

Arguments

`unstained_defro`	cell count of unstained (vital) cells immediately after defrosting
`unstained_24h`	count of unstained (vital) cells 24 hours after defrosting

Value

multiplication count after frosting

Examples

```
# original data is going to be applied
ud  <- c(12, 22, 15, 17)
u24 <-  c(9, 3, 5, 4)

MultiRate(unstained_defro =ud, unstained_24h = u24)
```

`PGVL` *Power input of a bubble column reactor*

Description

Power input of a bubble column reactor

Usage

```
PGVL(
  uGR,
  A.Riser = 0.00528,
  p,
  rho.liq = 997.048,
  gforce = 9.81,
  rho.gas = 1.2041,
  d.gaser = (10^-3),
  d.upcol,
  h.undisp.Riser,
  h.undisp.Downer,
  A.Downer = 0.00196,
  N = 52
)
```

Arguments

uGR	superficial velocity (riser)
A.Riser	column cross-section (riser)
p	room pressure
rho.liq	density of the liquid
gforce	g force
rho.gas	density of the gas
d.gaser	surface area of the gas nozzle
d.upcol	surface area of the riser
h.undisp.Riser	height of the liquid (riser) without gas
h.undisp.Downer	height of the liquid (downer) without gas
A.Downer	column cross-section (downer)
N	count of gas outlets

Value

power input

---

plot_MM_direct *Create a non-linear Regression for a Michaelis-Menten type Enzyme with a little knowlege of its Km and Vmax*

Description

Create a non-linear Regression for a Michaelis-Menten type Enzyme with a little knowlege of its Km and Vmax

Usage

```
plot_MM_direct(
  sub,
  velo,
  title = "title",
  xlab = "abscisaa",
  ylab = "ordinate"
)
```

Arguments

sub	substrate concentration
velo	velocity
title	title of the plot
xlab	lable of the abscissa
ylab	lable of the ordinate

Value

a plot

---

plot_regression *Generic plotting function for usage with any absorption spectroscopic data*

Description

create a graph from obtained data of calibration standards

Usage

```
plot_regression(abs, conc)
```

Arguments

abs	list
conc	list

PM *The power-input via the motor*

Description

The power-input via the motor

Usage

```
PM(P.perc, rpm, F1 = 0.107, F2 = 0.062, FC = 0.753)
```

Arguments

P.perc	motor power in percent
rpm	rotation per minute
F1	armature voltage / rpm
F2	armature current / motor power in percent
FC	degree of efficiency

samp.vol *Compute sample (Vorlage) volumes V_vor = (V_soll * c_soll) / c_ist*

Description

Compute sample (Vorlage) volumes V_vor = (V_soll * c_soll) / c_ist

Usage

```
samp.vol(conc.aim, vol.aim = 1, conc.is)
```

Arguments

conc.aim	aim concentration (C_soll)
vol.aim	volume that is aimed (V_soll)
conc.is	current concentration (C_ist)

Value

sample volumes

uG *compute the superficial velocity*

Description

compute the superficial velocity

Usage

```
uG(velG, A)
```

Arguments

velG	gas velocity
A	flow surface

Value

the superficial velocity

VDi *Compute the dispersion volume mind the units*

Description

Compute the dispersion volume mind the units

Usage

```
VDi(Ai, HDi)
```

Arguments

Ai	column cross-section
HDi	gasified height

Value

the dispersion volume

Examples

```
VDi(Ai = 0.45, HDi = 1.33)
```

velG.cal — *compute the actual gas flow (eliminate the error of the rotameter) of either the big airlift reactor or the smaller one used at the Biotechnikum at Forum Seestraße*

Description

compute the actual gas flow (eliminate the error of the rotameter) of either the big airlift reactor or the smaller one used at the Biotechnikum at Forum Seestraße

Usage

```
velG.cal(velG.mes, p.mes, t.mes, reactor = c("big", "small"))
```

Arguments

velG.mes	measured gas velocity (values read of from rotameter)
p.mes	measured current room pressure
t.mes	measured current room temperature
reactor	either the "big" or "small" reactor (as string)

Value

actual gas flow

Examples

```
velG.cal( velG.mes = 1, p.mes = 1.024, t.mes = 299.9, reactor = "small" )
```

vitality — *Vitality of a cell population*

Description

Vitality of a cell population

Usage

```
vitality(stained, unstained)
```

Arguments

stained	cell count of stained cells
unstained	cell count of unstained cells

Value

Vitality of the cell population

Examples

```
vitality(stained = 3, unstained = 77)
```

vitrate *Vitality rate of a cell population*

Description

Vitality rate of a cell population

Usage

```
vitrate(unstained_prefro, unstained_defro)
```

Arguments

unstained_prefro	cell count of unstained cells prior to frosting
unstained_defro	cell count of unstained cells after frosting

Value

Vitality rate of a cell population

Examples

```
vitrate(unstained_prefro = 33, unstained_defro = 12)
```

vol.act *generic functions to determin protein activity*

Description

All Volumes should be given with same metrics

Usage

```
vol.act(vol.tot = 1, dfac = 1, delta.E, vol.test, epsilon, cuvette = 1)
```

Arguments

vol.tot	total volume, default is 1
dfac	dilution factor, default is 1
delta.E	Delta E (without default)
vol.test	test volume (without default)
epsilon	molar attenuation coefficient (without default)
cuvette	cuvette factor or path length, default is 1

`yield` *Compute yield count*

Description

Compute yield count

Usage

```
yield(unstained_24h, unstained_prefro)
```

Arguments

`unstained_24h`	count of unstained (vital) cells 24 hours after defrosting
`unstained_prefro`	cell count of unstained (vital) cells in advance of frosting

Value

yield count

Examples

```
# original data is going to be applied
uP  <- c(88, 73, 72, 97)
u24 <-  c(9, 3, 5, 4)
yield(unstained_prefro = uP, unstained_24h = u24)
```

Index