Package 'rGV'

December 19, 2024

Version 0.0.5

Date 2024-12-19

Title Analysis of Continuous Glucose Monitor Data

Author Evan Olawsky [aut, cre],
Yuan Zhang [ctb],
Lynn Eberly [ctb]

Maintainer Evan Olawsky <evanolawsky@gmail.com>

Depends R (>= 3.2.0), chron

Description Reads in continuous glucose monitor data of many different formats, calculates a host of glycemic variability metrics, and plots glucose over time.

License MIT + file LICENSE

Encoding UTF-8

Suggests testthat (>= 3.0.0)

Config/testthat/edition 3

RoxygenNote 7.1.1

NeedsCompilation no

Repository CRAN

Date/Publication 2024-12-19 21:10:02 UTC

Contents

`adrr` *Calculate Average Daily Risk Range (ADRR)*

Description

Calculate Average Daily Risk Range (ADRR)

Usage

```
adrr(x, times, unit = "mg", method = "manuscript")
```

Arguments

`x`	vector of glucose readings
`times`	vector of corresponding times, in minutes
`unit`	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".
`method`	"manuscript", "corrected", or "easy". Null value is "manuscript".

Value

The numeric ADRR value for a given dataset of glucose measurements and times.

Examples

```
adrr(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60),
unit="mg", method='manuscript')
```

bgi *Calculate Low / High Blood Glucose Index (LBGI, HBGI)*

Description

Calculate Low / High Blood Glucose Index (LBGI, HBGI)

Usage

```
bgi(x, unit = "mg", method = "manuscript")
```

Arguments

x	vector of glucose readings
unit	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".
method	"manuscript", "corrected", or "easy". Null value is "manuscript".

Value

A list containing the LBGI and HBGI values for a given dataset of glucose measurements.

Examples

```
bgi(x=c(rep(100, 10), rep(120, 10), 105, 85), unit='mg', method='manuscript')
```

cgm_auc *Calculate area under the curve (AUC)*

Description

Calculate area under the curve (AUC)

Usage

```
cgm_auc(x, times, thresh = 100, above = TRUE)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
thresh	threshold above (or below) which you wish to calculate the AUC. Default is 100.
above	logical indicating whether you wish to calculate area above the threshold value (TRUE) or below it (FALSE). Default is TRUE.

Value

The numeric area under the curve value for a given dataset of glucose measurements and times.

Examples

```
cgm_auc(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60), thresh=110, above=TRUE)
```

`cgm_plot` *Plot glucose values over time*

Description

Plot glucose values over time

Usage

```
cgm_plot(x, times, unit = "mg")
```

Arguments

`x`	vector of glucose readings
`times`	vector of corresponding times, in minutes
`unit`	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".

Value

A plot of glucose values over time.

Examples

```
cgm_plot(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60), unit='mg')
```

`conga` *Calculate continuous overall net glycemic action (CONGA)*

Description

Calculate continuous overall net glycemic action (CONGA)

Usage

```
conga(x, times, n = 1, s = 1, method = "manuscript")
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
n	number of hours between "partner" observations. Null value is 1.
s	number of minutes of slack used when searching for partners. Null value is 1.
method	"manuscript" or "easy". Null value is "manuscript".

Value

The numeric CONGA value for a given dataset of glucose measurements and times.

Examples

```
conga(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60),
n=1, s=1, method="manuscript")
```

---

cv *Calculate coefficient of variation (CV)*

Description

Calculate coefficient of variation (CV)

Usage

```
cv(x, times, overall = TRUE, interval = 1)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
overall	a logical, equal to TRUE you want the CV for the entire dataset, or equal to FALSE if you would prefer many CV values over a moving window
interval	size (in hours) of the moving window to be used if overall is false. Null value is 1.

Value

Either a numeric coefficient of variation over the entire dataset or a vector of CV values over windows of the data.

Examples

```
cv(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60), overall=TRUE)
```

diff_plot *Plots glucose changes over time*

Description

Plots glucose changes over time

Usage

```
diff_plot(x, times, n = 1, s = 1, unit = "mg")
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
n	number of hours between "partner" observations. Null value is 1.
s	number of minutes of slack used when searching for partners. Null value is 1.
unit	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".

Value

A plot of n-hour glucose differences over time.

Examples

```
diff_plot(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60), n=1, s=1, unit='mg')
```

dist_travelled *Calculate distance travelled*

Description

Calculate distance travelled

Usage

```
dist_travelled(x)
```

Arguments

x	vector of glucose readings

Value

The numeric distance travelled value for a given dataset of glucose measurements.

Examples

```
dist_travelled(x=c(rep(100, 10), rep(120, 10), 105, 85))
```

---

gmi *Calculate Glucose Management Indicator (GMI)*

Description

Calculate Glucose Management Indicator (GMI)

Usage

```
gmi(x, unit = "mg")
```

Arguments

x	vector of glucose readings
unit	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".

Value

The numeric GMI value for a given dataset of glucose measurements.

Examples

```
gmi(x=c(rep(100, 10), rep(120, 10), 105, 85), unit='mg')
```

---

grade *Calculate Glycemic Risk Assessment Diabetes Equation (GRADE)*

Description

Calculate Glycemic Risk Assessment Diabetes Equation (GRADE)

Usage

```
grade(
  x,
  unit = "mg",
  method = "manuscript",
  c1 = ifelse(unit == "mg", 70.2, 3.9),
  c2 = ifelse(unit == "mg", 140.4, 7.8)
)
```

Arguments

x	vector of glucose readings
unit	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".
method	"manuscript" or "easy". Null value is "manuscript".
c1	glucose value below which readings are considered hypoglycemic. Default is 70.2 mg/dL.
c2	glucose value above which readings are considered hyperglycemic. Default is 140.4 mg/dL.

Value

A list containing the GRADE value and the percentage of the GRADE value due to euglycemia, hypoglycemia, and hyperglycemia for a given dataset of glucose measurements

Examples

```
grade(x=c(rep(100, 10), rep(120, 10), 105, 85), unit='mg', method='manuscript', c1=70.2, c2=140.4)
```

GV *Calculate all glycemic variability metrics*

Description

Calculate all glycemic variability metrics

Usage

```
GV(
  x,
  times,
  unit = "mg",
  m_index = 120,
  k = 60,
  s = 1,
  conga_n = 1,
  interval = 1,
  thresh = 100,
  event_thresh = 55
)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
unit	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".
m_index	a value to be considered a 'standard' blood glucose value for calculation of M-value, in mg/dL. Null value is 120.
k	length of time (in minutes) used to find partners. Null value is 60.
s	number of minutes of slack used when searching for partners. Null value is 1.
conga_n	number of hours between "partner" observations. Null value is 1.
interval	size (in hours) of the moving window to be used if overall is false. Null value is 1.
thresh	threshold above (or below) which you wish to calculate the AUC. Default is 100.
event_thresh	glucoses below this threshold are considered as part of an episode. Default is 55

Value

A data frame containing the entire suite of rGV metrics for the given dataset.

Examples

```
GV(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60), unit='mg',
m_index=120, k=60, s=1, conga_n=1, interval=1, thresh=100, event_thresh=55)
```

---

gvp *Calculate Glycemic Variability Percentage (GVP)*

Description

Calculate Glycemic Variability Percentage (GVP)

Usage

```
gvp(x, times)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes

Value

The numeric GVP value for a given dataset of glucose measurements and times.

Examples

```
gvp(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60))
```

j_index *Calculate J-index*

Description

Calculate J-index

Usage

```
j_index(x, unit = "mg")
```

Arguments

x	vector of glucose readings
unit	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".

Value

The numeric J-index value for a given dataset of glucose measurements.

Examples

```
j_index(x=c(rep(100, 10), rep(120, 10), 105, 85), unit='mg')
```

li *Calculate the lability index (LI)*

Description

Calculate the lability index (LI)

Usage

```
li(x, times, k = 60, s = 1)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
k	length of time (in minutes) used to find partners. Null value is 60.
s	number of minutes of slack used when searching for partners. Null value is 1.

Value

The numeric value of the lability index for a given dataset of glucose measurements and times.

Examples

```
li(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60), k=60, s=1)
```

---

mag *Calculate Mean Absolute Glucose (MAG)*

Description

Calculate Mean Absolute Glucose (MAG)

Usage

```
mag(x, times)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes

Value

The numeric MAG value for a given dataset of glucose measurements and times.

Examples

```
mag(x=c(rep(100, 10),rep(120, 10), 105, 85), times=seq(0, 1260, 60))
```

---

mage *Calculate Mean Amplitude of Glycemic Excursions (MAGE)*

Description

Calculate Mean Amplitude of Glycemic Excursions (MAGE)

Usage

```
mage(x, times)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes

Value

The numeric MAGE value for a given dataset of glucose measurements and times.

Examples

```
mage(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60))
```

---

modd *Calculate Mean of Daily Differences (MODD)*

Description

Calculate Mean of Daily Differences (MODD)

Usage

```
modd(x, times, s = 1, method = "manuscript")
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
s	number of minutes of slack used when searching for partners. Null value is 1.
method	"manuscript" or "easy". Null value is "manuscript".

Value

The numeric MODD value for a given dataset of glucose measurements and times.

Examples

```
modd(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60), s=1, method='manuscript')
```

---

m_value *Calculate M-value*

Description

Calculate M-value

Usage

```
m_value(x, unit = "mg", index = 120, method = "manuscript")
```

Arguments

x	vector of glucose readings
unit	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".
index	value to be considered a 'standard' blood glucose value, in mg/dL. Null value is 120.
method	"manuscript", "corrected", or "easy". Null value is "manuscript".

Value

The numeric M-value for a given dataset of glucose measurements.

Examples

```
m_value(x=c(rep(100, 10), rep(120, 10), 105, 85), unit='mg', index=120, method='manuscript')
```

---

num_events *Find number of episodes below a given glucose value for a given amount of time*

Description

Find number of episodes below a given glucose value for a given amount of time

Usage

```
num_events(x, times, thresh = 55, len = 15, gap = 5)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
thresh	glucoses below this threshold are considered as part of an episode. Default is 55
len	minimum length of an episode. Default is 15
gap	typical gap between CGM measurements, in minutes. Default is 5

Value

The integer number of events for a given dataset of glucose measurements and times.

Examples

```
num_events(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60),
thresh=55, len=15, gap=5)
```

read_cgm *Read in continuous glucose monitor data*

Description

Read in continuous glucose monitor data

Usage

```
read_cgm(
  file,
  timezero = "first",
  na.rm = TRUE,
  skip = 0,
  calib_col = NA,
  calib_tag,
  mult_sensors = FALSE,
  sensor_times = NA,
  sensor_gap = 120,
  time_col,
  gluc_col,
  time_sep = " ",
  time_format = c(dates = "m/d/y", times = "h:m:s"),
  high_ind = "High",
  high_value = 400,
  low_ind = "Low",
  low_value = 40
)
```

Arguments

file	name of the CSV file to be read in
timezero	set to "first" if the first glucose reading should be considered time zero and set to "midnight" if midnight of the day of the first reading should be considered time zero. Default is "first".
na.rm	a logical that is TRUE if you wish to exclude all readings that are missing glucose values or time stamps and FALSE if not. Default is TRUE.
skip	the number of lines in the data file to skip before beginning to read in data
calib_col	the number or name of the column containing information regarding calibration status of each glucose entry
calib_tag	the character value used to denote calibration rows in calib_col
mult_sensors	a logical that is TRUE if you wish to split the data set into parts corresponding to different CGM sensors and FALSE if not. Default is FALSE.

sensor_times	a vector of times (in the same format as the time data) that correspond to the beginning of a new CGM sensor. These times are used to split the data between multiple sensors if mult_sensors is TRUE. If sensor_times is NA, the data is split automatically at every gap of sensor_gap or more minutes.
sensor_gap	a number specifying the minimum gap (in minutes) for which we should split the data into two pieces. Default is 120.
time_col	the number or name of the column containing time data
gluc_col	the number or name of the column containing glucose data
time_sep	character that separates date from time in your time data
time_format	specify date and time formats according to the specification used in the chron package. Default is c(dates = "m/d/y", times = "h:m:s").
high_ind	character value that identifies high glucose values in the data. Default is "High".
high_value	numeric value by which to replace glucose values equal to "high_ind". Default is 400.
low_ind	character value that identifies low glucose values in the data. Default is "Low".
low_value	numeric value by which to replace glucose values equal to "low_ind". Default is 40.

Value

A data frame with two columns: glucose values and time. This data frame can then be used with other rGV functions to calculate CGM metrics.

---

st_dev *Calculate standard deviation (SD)*

Description

Calculate standard deviation (SD)

Usage

```
st_dev(x, times, overall = TRUE, interval = 1)
```

Arguments

x	vector of glucose readings
times	vector of corresponding times, in minutes
overall	a logical, equal to TRUE you want the CV for the entire dataset, or equal to FALSE if you would prefer many CV values over a moving window
interval	size (in hours) of the moving window to be used if overall is false. Null value is 1.

Value

Either a numeric standard deviation over the entire dataset or a vector of SD values over windows of the data.

Examples

```
st_dev(x=c(rep(100, 10), rep(120, 10), 105, 85), times=seq(0, 1260, 60), overall=TRUE)
```

`symm_plot` *Plot the symmetrized glucose values*

Description

Plot the symmetrized glucose values

Usage

```
symm_plot(x, times, unit = "mg")
```

Arguments

`x`	vector of glucose readings
`times`	vector of corresponding times, in minutes
`unit`	"mg" if the units are mg/dL or "mmol" if the units are mmol/L. Null value is "mg".

Value

A plot of symmetrized glucose values over time. These symmetrized values are used in the calculation of BGI and ADRR.

Examples

```
symm_plot(x=c(rep(100, 10),rep(120, 10), 105, 85), times=seq(0, 1260, 60), unit='mg')
```

time_on *Calculate amount of time that the CGM was active*

Description

Calculate amount of time that the CGM was active

Usage

```
time_on(times, gap = 5)
```

Arguments

times	vector of corresponding times, in minutes
gap	typical gap between CGM measurements, in minutes. Default is 5

Value

The numeric amount of time that the CGM device was active in a given dataset.

Examples

```
time_on(times=seq(0, 1260, 60), gap=5)
```

tir *Calculate time in range (TIR)*

Description

Calculate time in range (TIR)

Usage

```
tir(x, low = 70, high = 180)
```

Arguments

x	vector of glucose readings
low	lower bound of the range. Default is 70
high	upper bound of the range. Default is 180

Value

The numeric time in range value for a given dataset of glucose measurements and times.

Examples

```
tir(x=c(rep(100, 10), rep(120, 10), 105, 85), low=70, high=80)
```

Index