

Package: AQLTools (via r-universe)

September 21, 2026

Title AQLT Toolkit

Version 0.1.0

Date septembre 2017

Description Boîte à outils de fonctions, surtout autour des séries temporelles.

Depends ggplot2, highcharter,

Imports grid, rsdmx, reshape2, scales, stats, zoo, XLConnect

Suggests knitr, rmarkdown

VignetteBuilder knitr

License GPL-2

Encoding UTF-8

LazyData true

RoxygenNote 7.1.1

Config/pak/sysreqs cmake libglpk-dev make default-jdk libicu-dev libuv1-dev libxml2-dev libssl-dev

Repository <https://aqlt.r-universe.dev>

Date/Publication 2023-10-30 22:26:45 UTC

RemoteUrl <https://github.com/AQLT/AQLTools>

RemoteRef HEAD

RemoteSha 3c14f0ffc990e427c5a69ee5c149d6700d0f23cd

Contents

ctrl_c	2
ctrl_v	3
data_to_ts	4
facteurStatique	5
graph_ts	6
hc_ts	7
lectureBDM	9
ts_utils	10
ts2xls	10
volatilite	11

ctrl_c	<i>Exporter des données vers le presse-papier</i>
--------	---

Description

Fonction qui permet d'exporter des données (un vecteur ou un [data.frame](#) par exemple) vers le presse-papier (pour ensuite les coller dans un tableur par exemple).

Usage

```
ctrl_c(data, col.names = TRUE, row.names = FALSE, memory = 2^5, dec = ",", ...)
```

Arguments

data	les données à exporter.
col.names	Booléen indiquant si l'on souhaite exporter les noms des colonnes (par défaut col.names = TRUE)
row.names	Booléen indiquant si l'on souhaite exporter les noms des lignes (par défaut row.names = FALSE)
memory	entier indiquant la taille maximale de l'objet à copier dans le presse-papier. Par défaut la limite est de 32 Kb (=2^5 Kb).
dec	le caractère utilisé comme séparateur décimal dans le tableur où l'on souhaite exporter les données. Par défaut la virgule est utilisée : utiliser le point pour si l'on utilise tableur non francophone.
...	autres paramètres de read.table (uniquement si les données du presse-papier contiennent plusieurs colonnes).

Details

Il est possible qu'il y ait une ligne en trop si la fonction est utilisée sous Mac Os.

See Also

[ctrl_v](#) pour importer depuis le presse-papier.

Examples

```
ctrl_c(c("line 1", "line 2 \n and line 3"))
```

ctrl_v*Importer des données depuis le presse-papier*

Description

Fonction qui permet d'importer sous R des données depuis le presse-papiers. Ces données peuvent par exemple venir d'un tableur (Excel, Calc, etc.) : copier les données du tableur et ensuite lancer sous R la fonction `ctrl_v()`

Usage

```
ctrl_v(header = TRUE, row.names, stringsAsFactors = FALSE, dec = ",", ...)
```

Arguments

header	booléen indiquant si la première ligne doit être interprétée comme les noms des colonnes (TRUE par défaut). Paramètre utilisé uniquement si les données du presse-papier contiennent plusieurs colonnes.
row.names	un vecteur contenant les noms des lignes. Cela peut être un vecteur contenant les noms des lignes, un entier donnant le numéro de la colonne contenant les noms des lignes ou une chaîne de caractères donnant le nom de la colonne qui contient les noms des lignes. Lorsque <code>row.names = NULL</code> alors les lignes sont numérotées.
stringsAsFactors	booléen indiquant si l'on souhaite que les chaînes de caractères soient transformées en facteur dans la table de sortie (FALSE par défaut). Paramètre utilisé uniquement si les données du presse-papier contiennent plusieurs colonnes.
dec	le caractère utilisé comme séparateur décimal dans les données du presse-papiers. Par défaut la virgule est utilisée : utiliser le point pour si les données proviennent d'un tableur non francophone. Paramètre utilisé uniquement si les données du presse-papier contiennent plusieurs colonnes.
...	autres paramètres de read.table (uniquement si les données du presse-papier contiennent plusieurs colonnes).

Value

Un vecteur de caractères s'il n'y a qu'une colonne, sinon un [data.frame](#).

See Also

[ctrl_c](#) pour exporter vers le presse-papier.

Examples

```
# Après avoir copié des données depuis un tableur
ctrl_v()
```

Description

Fonctions qui permettent de convertir des tableaux en [ts](#).

Usage

```
ymd_ts(x, col_date = 1, sep_date, frequence = NULL)
```

```
dmy_ts(x, col_date = 1, sep_date, frequence = NULL)
```

Arguments

x	table contenant au moins deux colonnes dont une avec des dates.
col_date	numéro ou nom de la colonne contenant les dates (par défaut c'est la première colonne).
sep_date	caractère utilisé pour séparé le jour, le mois et l'année. Détection automatique par défaut.
frequence	fréquence de l'objet ts . Par défaut la fonction fait une recherche automatique (il peut y avoir des problèmes si l'on n'a pas d'année entière connue).

Details

Les fonctions `ymd_ts()` et `dmy_ts()` permettent de convertir des tableaux (par exemple des [data.frame](#)) en [ts](#). Pour cela il faut dans le tableau d'entrée une colonne contenant les dates qui est au format YYYY-MM-JJ (fonction `ymd_ts()`) ou au format JJ-MM-YYYY (fonction `dmy_ts()`). Le séparateur utilisé entre le jour, le mois et l'année peut être modifié par le paramètre `sep_date`.

Value

Un [ts](#).

Exemples

```
x <- data.frame(dates=paste(rep(2000:2002, each=4),
                           rep(c("01", "04", "07", "10"),2) , "01", sep = "-"),
               donnees = 1:12, stringsAsFactors = FALSE)

ymd_ts(x)
```

facteurStatique	Analyse factorielle statique
-----------------	------------------------------

Description

Fonction qui permet de calculer un indicateur synthétique par une analyse factorielle statique.

Usage

```
facteurStatique(  
  data,  
  date_deb,  
  date_fin,  
  normalise = FALSE,  
  retard_contrib = -1  
)
```

Arguments

data	un objet de type <code>ts</code> contenant les données.
date_deb	date de début d'estimation.
date_fin	date de fin d'estimation.
normalise	booléen indiquant s'il faut renormaliser les données à une moyenne 100 et écart-type 10.
retard_contrib	nombre de retards à prendre en compte pour le calcul des contribs.

Value

Un objet de même type que celui en entrée.

Exemples

```
series_cvs <- c(`Capacité d'épargne actuelle` = "000857195",  
               `Capacité d'épargne future` = "000857198",  
               `Évolution future du chômage` = "000857190",  
               `Niveau de vie futur en France` = "000857189",  
               `Niveau de vie passé en France` = "000857188",  
               `Opportunité de faire des achats importants` = "000857193",  
               `Situation financière future` = "000857197",  
               `Situation financière passée` = "000857196")  
data <- lectureBDM(series_cvs)  
facteurStatique(data)
```

graph_ts

*Graphique d'une série temporelle***Description**

Fonction qui permet de faire des graphiques de séries temporelles à partir d'un objet de type `ts`.

Usage

```
graph_ts(
  data,
  titre = NULL,
  sous_titre = NULL,
  legende = NULL,
  afficheVolatilite = FALSE,
  cex = 0.6,
  diviserParPeriode = FALSE,
  x_lab = NULL,
  x_lab_month = FALSE,
  y_lab = NULL,
  outDec = ", ",
  n_xlabel = length(time(data))%24,
  n_ylabel = 12,
  prec_plot,
  size = 0.7
)
```

Arguments

<code>data</code>	un objet de type <code>ts</code> contenant une ou plusieurs séries à grapher.
<code>titre</code>	titre du graphique (par défaut pas de titre).
<code>sous_titre</code>	sous-titre du graphique (par défaut pas de sous-titre).
<code>legende</code>	légende du graphique. Par défaut le nom des variables dans l'objet <code>data</code> en entrée. S'il n'y a qu'une série à tracer alors aucune légende n'est affichée.
<code>afficheVolatilite</code>	booléen indiquant si l'on souhaite afficher la volatilité des séries graphées (c'est-à-dire l'écart-type de leur variation). Par défaut <code>afficheVolatilite = FALSE</code> : on n'affiche rien.
<code>cex</code>	nombre indiquant le montant par lequel le texte associé à la volatilité doit être mis à l'échelle par rapport à la valeur par défaut. Par défaut <code>cex = 0.6</code> : le texte est donc 60 % plus petit que la valeur par défaut. Par exemple, pour des séries trimestrielles cela permet d'avoir un graphique par trimestre traçant chacun les valeurs associées à son trimestre.
<code>diviserParPeriode</code>	diviser le graphique en fonction de la période : il y aura en sortie autant de graphiques que de périodes dans la série de départ.

x_lab	titre axe des abscisses (par défaut aucun titre).
x_lab_month	boolean indiquant si on veut que les x_labels soit formatés avec le mois
y_lab	titre axe des ordonnées (par défaut "Date").
outDec	séparateur décimal utilisé pour dans la légende des axes (par défaut la virgule).
n_xlabel	nombre de labels pour l'axe des abscisses (par défaut une année sur deux).
n_ylabel	nombre de labels pour l'axe des ordonnées (par défaut 12).
prec_plot	graphique à ajouter avant le geom_line
size	taille des points.

Value

Un graphique [ggplot](#).

Examples

```
data <- lectureBDM("001585942", "001585980")
legende <- c("Carnets de commandes globaux", "Carnets de commandes étrangers")
titre <- "Soldes d'opinion sur les carnets de commandes dans l'industrie manufacturière"
graph_ts(data, titre = titre, legende = legende, afficheVolatilite = TRUE)
graph_ts(data, titre = titre, legende = legende, diviserParPeriode = TRUE)
```

hc_ts

Graphiques interactifs d'une série temporelle avec highcharter

Description

Fonctions qui permettent de faire des graphiques interactifs de séries temporelles à partir d'un objet de type `ts`.

Usage

```
hc_lines(
  data,
  titre = NULL,
  sous_titre = NULL,
  legende = NULL,
  affiche_legende = TRUE,
  x_lab = NULL,
  y_lab = NULL,
  outDec = ",",
  useHTML = FALSE,
  digits = 1
)
```

```
hc_stocks(
  data,
```

```

    titre = NULL,
    sous_titre = NULL,
    legende = NULL,
    affiche_legende = TRUE,
    x_lab = NULL,
    y_lab = NULL,
    outDec = ",",
    useHTML = FALSE,
    type = NULL,
    color = NULL,
    digits = 1
  )

```

Arguments

<code>data</code>	un objet de type <code>ts</code> contenant une ou plusieurs séries à grapher.
<code>titre</code>	titre du graphique (par défaut pas de titre).
<code>sous_titre</code>	sous-titre du graphique (par défaut pas de sous-titre).
<code>legende</code>	légende du graphique. Par défaut le nom des variables dans l'objet <code>data</code> en entrée. S'il n'y a qu'une série à tracer alors aucune légende n'est affichée.
<code>affiche_legende</code>	booléen indiquant si l'on affiche la légende.
<code>x_lab</code>	titre axe des abscisses (par défaut aucun titre).
<code>y_lab</code>	titre axe des ordonnées (par défaut "Date").
<code>outDec</code>	séparateur décimal utilisé pour dans la légende des axes (par défaut la virgule).
<code>useHTML</code>	Booléen indiquant si l'on veut utiliser du code HTML dans les titres.
<code>digits</code>	nombre de chiffres après la virgule.
<code>type</code>	vecteur contenant le type à utiliser pour les séries (par défaut "lines").
<code>color</code>	vecteur contenant les couleurs à utiliser (par défaut on garde les couleurs de <code>highchart</code>).

Value

Un graphique `highcharter`.

Examples

```

data <- lectureBDM("001585942","001585980")
legende <- c("Carnets de commandes globaux","Carnets de commandes étrangers")
titre <- "Soldes d'opinion sur les carnets de commandes dans l'industrie manufacturière"
hc_lines(data, titre = titre, legende = legende)
hc_stocks(data, titre = titre, legende = legende)

```

Description

Fonction qui permet d'importer des séries de la Banque de données macro-économiques (BDM) de l'Insee à partir de son idbank. L'idbank d'une série peut être trouvé sur le site de la bdm (<https://www.bdm.insee.fr/bdm2/index>). permet également l'utilisation de régressions quantiles.

Usage

```
lectureBDM(idbank, ...)
```

Arguments

idbank	un vecteur d'un ou plusieurs idbank à télécharger.
...	d'autres vecteurs d'un ou plusieurs idbank à télécharger.

Details

La fonction permet de télécharger des séries temporelles présentes la Banque de données macro-économiques (BDM) de l'Insee à partir de son idbank. Elle utilise pour cela le service web SDMX de l'Insee (voir <http://www.bdm.insee.fr/bdm2/statique?page=sdmx> pour plus d'informations). Elle utilise le package **r sdmx** pour la lecture des données et **reshape2** pour la mise en forme. Les idbank en paramètre doivent correspondre à des séries qui ont la même périodicité. Si un des idbank en entrée n'est pas trouvé alors la fonction renvoie également un message d'avertissement.

Value

Retourne un objet de type `ts` ou `mts`, en fonction du nombre d'idbank en entrée, de la même fréquence que les séries en entrée.

Examples

```
# Pour télécharger l'IPI manufacturier CVS : 001654241
lectureBDM("001654241")
# Pour télécharger les IPI dans l'industrie manufacturière et dans l'agroalimentaire :
lectureBDM(c("001654241", "001654289"))
# ou bien :
lectureBDM("001654241", "001654289")
```

ts_utils

*Évolution et glissement annuel***Description**

Fonctions qui permettent, pour une série temporelle, de calculer l'évolution par rapport à la dernière période (*i.e.* : évolution mensuelle pour des séries mensuelles, évolution trimestrielle pour des séries trimestrielles, etc. ; avec `ev`) ou le glissement annuel (avec `ga()`) .

Usage

```
ev(x)
```

```
ga(x)
```

Arguments

`x` un objet de type `ts` ou `xts`.

Value

Un objet de même type que celui en entrée.

Exemples

```
x <- ts(1:10, frequency = 4, start = c(1959, 2))
ev(x)
ga(x)
```

ts2xls

*Exporter données ts vers un fichier excel***Description**

Fonction qui permet d'exporter des données `ts` vers un fichier excel en créant une colonne date qui sera au format date.

Usage

```
ts2xls(x, file, sheet = "Feuille 1", format = "dd/mm/yyyy")
```

Arguments

`x` un objet de type `ts`.
`file` nom du fichier excel (avec extension `.xls` ou `.xlsx`)
`sheet` nom de la feuille
`format` format en sortie dans Excel

`volatilite`*Volatilité*

Description

Fonction qui permet de calculer la volatilité d'une série temporelle (i.e. : l'écart-type de sa différence). Les valeurs manquantes ne sont pas prises en compte.

Usage

```
volatilite(x, lag = 1)
```

Arguments

<code>x</code>	un objet de type <code>ts</code> ou <code>xts</code> .
<code>lag</code>	nombre de retards à utiliser pour le calcul de la différence (par défaut <code>lag = 1</code> : on calcule une différence mensuelle pour les séries mensuelles).

Value

Un objet de même type que celui en entrée.

Examples

```
x <- ts(1:10, frequency = 4, start = c(1959, 2))
volatilite(x)
```

Index

`ctrl_c`, [2](#), [3](#)
`ctrl_v`, [2](#), [3](#)

`data.frame`, [2–4](#)
`data_to_ts`, [4](#)
`dmy_ts (data_to_ts)`, [4](#)

`ev (ts_utils)`, [10](#)

`facteurStatique`, [5](#)

`ga (ts_utils)`, [10](#)
`ggplot`, [7](#)
`graph_ts`, [6](#)

`hc_lines (hc_ts)`, [7](#)
`hc_stocks (hc_ts)`, [7](#)
`hc_ts`, [7](#)
`highcharter`, [8](#)

`lectureBDM`, [9](#)

`read.table`, [2](#), [3](#)

`ts`, [4–8](#), [10](#), [11](#)
`ts2xls`, [10](#)
`ts_utils`, [10](#)

`volatilite`, [11](#)

`xts`, [10](#), [11](#)

`ymd_ts (data_to_ts)`, [4](#)