

# Package ‘Rpu2’

October 28, 2015

**Type** Package

**Title** Routines pour RPU

**Version** 0.1.2

**Date** 2015-09-27

**Maintainer** Jean-Claude Bartier <jeanclaudio.bartier@gmail.com>

**Description** Ensemble de fonctions destin<3><a9>es <c3><a0> faciliter l'analyse des RPU.

**Depends** R (>= 3.1.0), lubridate, xtable, openintro, plotrix

**License** GNU

**LazyData** true

**Imports** dplyr, lubridate, xtable, openintro, plotrix

**NeedsCompilation** no

**Author** JC Bartier [aut, cre]

## R topics documented:

add.territoire . . . . .	3
analyse_type_etablissement . . . . .	3
barplot.week.variations . . . . .	4
completude . . . . .	5
completude.time . . . . .	5
copyright . . . . .	6
count.CIM10 . . . . .	7
datetime . . . . .	8
df.duree.pas . . . . .	8
duree.passage2 . . . . .	9
evolution . . . . .	10
factor2table . . . . .	10
finess2territoires . . . . .	11
format.n . . . . .	11
horaire . . . . .	12
is.present.at . . . . .	12
mn2h . . . . .	13

n.isna	14
p.isna	14
passage	15
passages.en.moins.de.4h	15
passages2	16
pdsa	17
plot.xts2	17
print.summary.rpu	18
print.table.rpu	18
pyramide.age	19
radar.completude	19
reorder.dataframe.fedoru	20
reorder.vector.fedoru	20
resume.age	21
resume.age.sexe	21
resume.ccmu	22
resume.dateheure	22
resume.dp	23
resume.entree	23
resume.mode.sortie	24
resume.motif	24
resume.passages	25
resume.rpu	25
resume.sexe	26
resume.transport	27
rpu.par.jour	27
rpu.par.jour2	28
rpu.par.mois	28
rpu2xts	29
summary.cp	30
summary.destination	30
summary.duree.passage	31
summary.orientation	31
summary.wday	32
synthese.completude	32
tab.completude	33
tarru	34
teste.radar	35
week.rpu	35
week.variations	36

---

add.territoire	NA
----------------	----

---

**Description**

Ajoute une colonne TERRITOIRE à un dataframe qui contient une colonne FINESS

**Usage**

```
add.territoire(dx)
```

**Arguments**

dx un dataframe ayant une colonne FINESS renseignée

**Value**

un dataframe

---

analyse_type_etablissement	<i>Analyse etablissement</i>
----------------------------	------------------------------

---

**Description**

fournit une liste d'indicateur à partir des données d'un établissement ou d'un groupe d'établissements.  
Voir rapport 2014: Analyse par type d'établissement

**Usage**

```
analyse_type_etablissement(es)
```

**Arguments**

es dataframe RPU (es = établissement de santé)

**Value**

"n.passages", "n.age.ren", "n.infl1an", "n.infl15ans", "n.75ans", "n.cp.rens", "n.etrangers", "n.lun",  
"n.mar", "n.mer", "n.jeu", "n.ven", "n.sam", "n.dim", "n.nuit", "n.pds", "n.h.rens", "n.trans.rens",  
"n.fo", "n.heli", "n.perso", "n.smur", "n.vsav", "n.ambu", "n.ccmu.rens", "n.ccmu1", "n.ccmu2",  
"n.ccmu3", "n.ccmu4", "n.ccmu5", "n.ccmuP", "n.ccmuD", "n.ccmu45", "n.sorties.conf", "mean.passage",  
"median.passage", "n.passage4", "n.hosp.passage4", "n.dom.passage4", "n.dom", "n.hosp", "n.transfert",  
"n.deces", "n.mode.sortie", "n.mutation2"

**Examples**

```
# es non SAMU, siège de SMUR
# es <- dx[dx$FINESS %in% c("Wis", "Hag", "Sav", "Sel", "Col"),]
# analyse_type_etablissement(es)
```

---

```
barplot.week.variations
```

*Variation du nombre de RPU par semaine*

---

**Description**

Variation du nombre de RPU par semaine

**Usage**

```
barplot.week.variations()
```

**Arguments**

x	vecteur du nombre de RPU pr semaine (voir week.rpu)
coltitre	bool, si TRUE la valeur de la barre est inscrite au dessus ou en dessous
colmoins	couleur des barres négatives. Red par défaut
colplus	couleur des barres positives. Blue par défaut
xlab	nom pour l'axe des X. 'Semaines' par défaut
cex.names	échelle pour le titre des barres (n° de la semaine). 0.8 par défaut
cex.col	échelle pour les valeurs des colonnes. Utile que si coltitre = TRUE. Défaut 0.8
dx	écart entre le sommet de la barre et l'affichage de sa valeur. Utile que si coltitre = TRUE. Défaut 3.
...	autres paramètres pour boxplot

**Value**

le vecteur des abscisses des colonnes

**Examples**

```
v <- week.variations(dx[dx$FINESS == "3Fr",])
barplot.week.variations(v[-length(v)], las = 2, main = "test", ylim = c(min(v[-length(v)])-10, max(v[-length(v)])
ylab = "Variations hebdomadaires")

###
v <- week.variations(week.rpu(dx[dx$FINESS == "Col",]))
barplot.week.variations(v[-length(v)], las = 2, main = "CH Colmar - 2015",
ylim = c(min(v[-length(v)])-10, max(v[-length(v)])+10), ylab = "Variations hebdomadaires", dx = 5)
```

---

completude	<i>taux de completude global.</i>
------------	-----------------------------------

---

**Description**

Pour chacune des rubriques RPU calcule le taux de réponse (complétude)

**Usage**

```
completude(dx, calcul = "percent", tri = FALSE)
```

**Arguments**

dx	Un dataframe
calcul	2 options "percent" (défaut) ou "somme". Somme = nb de réponses non nulles. Percent = % de réponses non nulles.
tri	si tri = TRUE (default) les colonnes sont triées par ordre croissant.

**Details**

todo

**Value**

vecteur des taux de complétude

**Author(s)**

JcB 2013-02-01

**See Also**

Other RPU: [radar.completude](#)

---

completude.time	<i>Pour un etablissement donne, calcule le aux de completude par mois, semaine, jours</i>
-----------------	---

---

**Description**

Pour un etablissement donne, calcule le aux de completude par mois, semaine, jours

**Usage**

```
completude.time(dx, finess, time = "month")
```

## Arguments

<code>dx</code>	un dataframe de type RPU
<code>finess</code>	établissement concerné ('Wis', 'Hag', 'Sav', ...)
<code>time</code>	factor de découpage

## Details

Au départ on dispose d'un dataframe de type RPU. Ce dataframe est splité en sous groupes sur une base temporelle (mois, jour, semaine...). Sur chacun des sous-groupes on applique la fonction "completude". Retourne un dataframe où chaque ligne correspond à une période et chaque colonne à un élément du RPU. Utilise "ddply" qui fonctionne comme tapply mais s'applique à un DF au lieu d'un vecteur et retourne un DF. TODO: extension à plusieurs établissements simultanément; limitation à certaines colonnes.

## Exemples

```
load("~/Documents/Resural/Stat Resural/RPU_2014/rpu2015d0112_provisoire.Rda")
# old
sav <- d15[d15$FINESS == "Sav",] # Saverne 2015
t3 <- ddply(sav, .(month(as.Date(sav$ENTREE))), completude) # completude par mois

# new
library(xts)
t3 <- completude.time(d15, "Sav", "day")
a <- seq(as.Date("2015-01-01"), length.out = nrow(t3), by = 1)
x <- xts(t3, order.by = a)
plot(x[, "DP"], main = "CH Saverne - DIAGNOSTIC PRINCIPAL", ylab = "% de complétude")

# TODO: tableau de complétude par mois et par Finess:
t3 <- ddply(dx, .(dx$FINESS, month(as.Date(dx$ENTREE))), completude)
# Application: rpu2014/Analyse/Completude/Analyse_completude
```

---

copyright

copyright

---

## Description

Place un copyright Resural sur un graphique. Par défaut la phrase est inscrite verticalement sur le bord droit de l'image

## Usage

```
copyright(an = "2013-2015", side=4, line=-1, cex=0.8, titre = "Resural")
```

**Arguments**

an (str) année du copyright (par défaut 2013)  
side coté de l'écriture (défaut = 4)  
line distance par rapport au bord. Défaut=-1, immédiatement à l'intérieur du cadre  
cex taille du texte (défaut 0.8)  
titre

**Value**

"© 2012 Resural"

**Author(s)**

JcB

---

count.CIM10	<i>Combien de codes CIM10</i>
-------------	-------------------------------

---

**Description**

examine un vecteur de caractères et compte le nombre de mots compatibles avec un code CIM10  
NA n'est pas compté comme un code CIM10

**Usage**

```
count.CIM10(vx)
```

**Arguments**

vx un vecteur de character

**Value**

n nombre de codes CIM1

**Author(s)**

JcB

**Examples**

```
count.CIM10(dx[dx$FINISS == "Col", "MOTIF"])
```

---

datetime	<i>met une string date au format YYYY-MM-DD HH:MM:SS</i>
----------	--

---

**Description**

met une string date au format YYYY-MM-DD HH:MM:SS

**Usage**

datetime(date)

**Arguments**

date                    une chaine de caractère de type Date

**Value**

un vecteur date time (lubridate)

**Note**

nécessite lubridate

**See Also**

horaire, passage.nuit

**Examples**

Transforme des rubriques ENTREE et SORTIE en objet datetime

---

df.duree.pas	NA
--------------	----

---

**Description**

fabrique à partir d'un dataframe de type RPU, un dataframe de type duree\_passage comportant les colonnes suivantes: date/heure d'ebtree, date/heure de sortie, durée de passage (en minutes par défaut), l'heure d'entrée (HMS), l'heure de sortie

**Usage**

df.duree.pas(dx, unit = "mins", mintime = 0, maxtime = 3)



**Arguments**

dx	un dataframe de type RPU
unit	unité de temps. Défaut = mins
mintime	défaut = 0. Durée de passage minimale
maxtime	défaut = 3 (72 heures). Durée de passage maximale

**Value**

dataframe de type duree\_passage

**Examples**

```
df <- df.duree.pas(dx)
```

---

duree.passage2	<i>Calcul de la duree de passage</i>
----------------	--------------------------------------

---

**Description**

todo

**Usage**

```
duree.passage2(dx, h1 = 0, h2 = 4320, hors_uhcd = TRUE)
```

**Arguments**

dx	dataframe RPU
h1	durée minimale en minutes (par défaut > 0)
h2	durée maximale en minutes (par défaut 4320 = 72 heures)
hors_uhcd	si TRUE (défaut) on retire les enregistrements où ORIENTATION = UHCD

**Value**

dataframe à 4 colonnes: entree, sortie, mode\_sortie, duree (en mn), he (heure d'entrée), hs (heure de sortie)

---

evolution

*Evolution d'une année sur l'autre*

---

**Description**

calcule l'évolution entre 2 chiffres

**Usage**

```
evolution(a, b)
```

**Arguments**

a                    chiffre de l'année courante  
b                    chiffre de l'année précédente

**Value**

pourcentage d'augmentation ou de diminution

**Examples**

```
evolution(n.rpu, n.rpu.2013)
```

---

factor2table

*NA*

---

**Description**

crée une table à 2 colonnes: fréquence et pourcentage

**Usage**

```
factor2table(vx, pc = TRUE)
```

**Arguments**

vx                    un vecteur de facteurs ou d'entiers  
pc                    si TRUE crée une colonne de %

**Value**

une table

**Examples**

```
a <- c(1,2,3,4,5,5,5,5,1,1,2); factor2table(a); print.table.rpu(a)
#           Fréq.      %
#           1      3 27.27
#           2      2 18.18
#           3      1  9.09
#           4      1  9.09
#           5      4 36.36
#
#           factor2table(pop18$GRAVITE, TRUE)
```

---

finess2territoires	NA
--------------------	----

---

**Description**

réorganiser les FINESS par territoires de santé

**Usage**

```
finess2territoires(finess)
```

**Arguments**

finess                      code finess de létablissement

**Examples**

```
dx$FINESS <- finess2territoires(dx)
```

---

format.n	<i>formate un nombre</i>
----------	--------------------------

---

**Description**

formate un nombre en ajoutant un espace pour les milliers une virgule décimale pas de notation scientifique deux chiffres significatifs

**Usage**

```
format.n(x)
```

**Arguments**

x                              un nombre entier ou décimal

**Examples**

```
format.n(7890.14) -> "7 890,14"
```

---

horaire	<i>extrait l'heure d'une date AAAA-MM-DD HH:MM:SS</i>
---------	---

---

### Description

extrait l'heure d'une date AAAA-MM-DD HH:MM:SS

### Usage

```
horaire(date)
```

### Arguments

date	une date ou un vecteur au format DATE
------	---------------------------------------

### Value

un vecteur d'heures au format HH:MM:SS

### Examples

```
e <- datetime(dx$ENTREE); he <- horaire(e)
```

---

is.present.at	NA
---------------	----

---

### Description

Crée le vecteur des personnes présentes à une heure donnée

### Usage

```
is.present.at((dp, heure = "15:00:00"))
```

### Arguments

dp	dataframe de type duree_passage
heure	heure au format HH:MM:SS. C'es l'heure à laquelle on veut mesurer les passages

### Value

np vecteur de boolean: TRUE si présent à l'heure analysee et FALSE sinon

## Examples

```
dp <- df.duree.pas(dx)
dp$present.a.15h <- is.present.at(dp)
# nombre moyen de patients présents à 15h tous les jours
n.p15 <- tapply(dp$present.a.15h, yday(as.Date(dp$ENTREE)), sum)
summary(n.p15)
sd(n.p15)
# transformation en xts
xts.p15 <- xts(n.p15, order.by = unique(as.Date(dp$ENTREE)))
plot(xts.p15, ylab = "Nombre de patients à 15h", main = "Nombre de patients présents à 15 heures")
lines(rollmean(x = xts.p15, k = 7), col = "red", lwd = 2)

# à 2h du matin
dp$present.a.2h <- is.present.at(dp, "02:00:00")
n.p2 <- tapply(dp$present.a.2h, yday(as.Date(dp$ENTREE)), sum)
summary(n.p2)
xts.p2 <- xts(n.p2, order.by = unique(as.Date(dp$ENTREE)))
plot(xts.p2, ylab = "Nombre de patients présents", main = "Nombre de patients présents à 2 heures du matin")
lines(rollmean(x = xts.p2, k = 7), col = "red", lwd = 2)
# pour les données de 2015, noter le pic à 2 heures du matin

# à 8 heures
present.a.8h <- is.present.at(dp, "08:00:00")
n.p8 <- tapply(present.a.8h, yday(as.Date(dp$ENTREE)), sum)
summary(n.p8)
xts.p8 <- xts(n.p8, order.by = unique(as.Date(dp$ENTREE)))
plot(xts.p8, ylab = "Nombre de patients présents", main = "Nombre de patients présents à 8 heures du matin")
lines(rollmean(x = xts.p8, k = 7), col = "red", lwd = 2)
```

---

mn2h

*transforme des minutes en heure/mn*


---

## Description

transforme des minutes en heure/mn

## Usage

mn2h(x)

## Arguments

x                      integer = nombre de minutes

## Value

char

---

`n.isna`*Nombre de NA*

---

**Description**

Nombre de NA dans un vecteur

**Usage**`n.isna(x)`**Arguments**

`x` un vecteur quelconque

**Value**

en entier

---

`p.isna`*Pourcentage de NA*

---

**Description**

Pourcentage de NA dans un vecteur

**Usage**`p.isna(x)`**Arguments**

`x` un vecteur quelconque

**Value**

un pourcentage

---

passage	<i>Horaires de passages</i>
---------	-----------------------------

---

**Description**

Horaires de passages

**Usage**

```
passage(he, horaire = "nuit")
```

**Arguments**

he	vecteur time de type hms
horaire	= 'nuit', 'nuit profonde', 'jour'

**Value**

un vecteur avec 2 éléments: le nombre de passages et le pourcentage en fonction de la période (jour, nuit)

**Note**

necessite lubridate. Prend en compte toutes les heures et pas seulement celles comprises entre 0 et 72h (voir passage2)

**See Also**

horaire

**Examples**

```
e <- datetime(dx$ENTREE); he <- horaire(e); nuit <- passage(he, "nuit")
```

---

passages.en.moins.de.4h	<i>Analyse les passages de moins de 4 heures.</i>
-------------------------	---

---

**Description**

analyse les durée de passage de moins de 4 heures par rapport aux durées de passage conformes (c'est à dire de moins de 72 heures).

**Usage**

```
passages.en.moins.de.4h(dx)
```

**Arguments**

dx un dataframe de type RPU

**Value**

n.so.conforme.dom, n.duree.passage.inf4h.dom, p.passages.en.moins.de.4h.dom, n.so.conforme.hosp, n.duree.passage.inf4h.hosp, p.duree.passage.inf4h.hosp

**Warning**

Cette fonction n'est pas terminée.

---

passages2	<i>Nombre de RPU sur une plage horaire donnee</i>
-----------	---

---

**Description**

Détermine le nombre de RPU sur une plage horaire donnée et le pourcentage par rapport au nombre total de passages contenus dans vx.

**Usage**

```
passages2(vx, h1, h2 = NULL)
```

**Arguments**

vx vecteur de type datetime (dx\$ENTREE, dx\$SORTIE par exemple). Transformé par ymd\_hms Transform dates stored as character or numeric vectors to POSIXct objects

h1 char heure de début ou période: 'nuit', 'nuit\_profonde', 'jour', 'pds', 'soir', '08:00:00'

h2 char heure de fin. h2 doit être > h1

**Details**

nécessite lubridate library(lubridate)

**Value**

2 objets: nombre de RPU et pourcentage

**Author(s)**

jcb

**Examples**

```
n.passages.nuit <- passages2(pop18$ENTREE, "nuit"); n.passages.nuit[1]; n.passages.nuit[2]
```



---

pdsa	<i>Détermine si on est en horaire de PDS.</i>
------	---

---

**Description**

Détermine si on est en horaire de PDS de WE (PDSWE) ou de semaine (PDSS) ou hors horaire de PDS (NPDS) à partir d'une date.

**Usage**

```
pdsa(dx)
```

**Arguments**

dx	vecteur date/heure au format YYYY-MM-DD HH:MM:SS
----	--

**Details**

REM sur xps les jours commencent par une minuscule alors que sur le Mac c'est une majuscule ?

**Value**

un vecteur de factor NPDS, PDSS, PDSW

**Examples**

```
x <- "2009-09-02 12:23:33"; weekdays(as.Date(x)); pds(x) # NPDS
```

---

plot.xts2	<i>plot.xts en couleur</i>
-----------	----------------------------

---

**Description**

La méthode plot.xts compte un bug qui empêche l'affichage de courbes en couleur. Cette version corrige le bug.

**Usage**

```
plot.xts2(x, y = NULL, type = "l", auto.grid = TRUE, major.ticks = "auto", minor.ticks = TRUE, major.f  
bar.col = "grey", candle.col = "white", ann = TRUE, axes = TRUE, col = "black", ...)
```

**Author(s)**

Roman Luštrik (<http://stackoverflow.com/users/322912/roman-lu>)

**Source**

<http://stackoverflow.com/questions/9017070/set-the-color-in-plot-xts>

---

```
print.summary.rpu
```

*Imprime un summary.rpu*

---

### Description

imprime un objet de type summary.rpu, en ligne ou en colonne (défaut) avec xtable.

### Usage

```
print.summary.rpu(x, sens = "colonne", cnames = NULL, rnames = NULL,
                  caption = "", type = "latex", ref = "")
```

### Arguments

x	un vecteur nommé
sens	'colonne' = vertical, 'ligne' = horizontal
cnames	noms des colonnes
rnames	noms des lignes

### Exemples

```
x <- ummary.wday(es$ENTREE))
print.summary.rpu(x, cnames = c("Jour", "n"), caption = "Nombre de RPU par jour de semaine")
```

---

```
print.table.rpu
```

*Imprime une table avec xtable.*

---

### Description

imprime une table avec xtable. Par défaut l'environnement est du type latex, le séparateur de milliers est l'espace et la virgule décimale

### Usage

```
print.table.rpu(t, caption = "", type = "latex", ref = "")
```

### Arguments

t	un objet de type table
caption	une légende. Mettre c("légende", "sommaire") si nécessaire
type	"latex" ou "html"
ref	référence du tableau (latex)

**Examples**

```
print.table.rpu(t)
print.table.rpu(t, "table de test")
print.table.rpu(t, "table de test", "html")
```

---

pyramide.age

*pyramide des ages*


---

**Description**

pyramide des ages

**Usage**

```
pyramide.age(dx, cut = 5, gap = 1, cex = 0.8, col.h = "light green", col.f = "khaki1")
```

**Arguments**

dx	dataframe RPU ou DF à 2 colonnes: AGE et SEXE
cut	intervalles. Par défaut tranche d'age de 5 ans, borne sup exclue: [0-5[ ans
gap	largeur de la colonne age (N = 1, varie de 0 à ...)
col.h	couleur pour les hommes
col.f	couleur pour les femmes

**Details**

pyramid nécessite epicalc, pyramid.plot nécessite plotrix

---

radar.completude

*dessine un graphe en etoile*


---

**Description**

dessine un graphe en étoile à partir des données retournées par "completude"

**Usage**

```
radar.completude(completude, finess = NULL, titre = NULL)
```

**Arguments**

completude	taux de completude global calculé par la fonction completude
finess	character: nom de l'établissement. NULL (default) => tout le datafame

**Value**

diagramme en étoile

**Author(s)**

JcB 2013-02-01

**See Also**

Other RPU: [completude](#)

**Examples**

```
radar.completude(completude(dx))
```

---

```
reorder.dataframe.fedoru
```

*Reordonne les colonnes du dataframe RPU dans l'ordre défini par la FEDORU.*

---

**Description**

Permet une meilleure cohérence du diagramme en étoile

**Usage**

```
reorder.dataframe.fedoru(dx)
```

**Arguments**

dx                    un dataframe de type RPU

---

```
reorder.vector.fedoru NA
```

---

**Description**

On part d'un vecteur contenant les intitulés du RPU et on le réordonne pour que les intitulés soient mis dans l'ordre du rapport FEDORU (proposition de GillesFaugeras)

**Usage**

```
reorder.vector.fedoru(dx)
```

**Arguments**

dx                    un dataframe du typr RPU

**Value**

un dataframe

---

resume.age	<i>Resume du vecteur des AGE</i>
------------	----------------------------------

---

**Description**

résumé du vecteur vx des AGE

**Usage**

summary.age(vx)

**Arguments**

vx                      vecteur char AGE

**Value**

"n", "n.na", "p.na", "n.rens", "p.rens", "n.inf1an", "n.inf15ans", "n.inf18ans", "n.75ans", "n.85ans", "n.90ans", "p.inf1an", "p.inf15ans", "p.inf18ans", "p.75ans", "p.85ans", "p.90ans", "mean.age", "sd.age", "median.age", "min.age", "max.age", "q1", "q3")

**Examples**

summary.dp(dx\$AGE)

---

resume.age.sexe	<i>NA</i>
-----------------	-----------

---

**Description**

résumé des vecteurs AGE et SEXE

**Usage**

summary.age.sexe(dx)

**Arguments**

dx                      dataframe RPU

**Value**

moyenne, écart-type, médiane par sexe

**Examples**

```
summary.age.sexe(dx)
```

---

```
resume.ccmu
```

*Resume du vecteur vx des CCMU*

---

**Description**

résumé du vecteur vx des CCMU

**Usage**

```
summary.ccmu(vx)
```

**Arguments**

vx                      vecteur de factor CCMU

**Value**

"n", "n.na", "p.na", "n.rens", "p.rens", "n.ccmu1", "n.ccmu2", "n.ccmu3", "n.ccmu4", "n.ccmu5",  
"n.ccmup", "n.ccmud", "p.ccmu1", "p.ccmu2", "p.ccmu3", "p.ccmu4", "p.ccmu5", "p.ccmup", "p.ccmud")

**Examples**

```
summary.ccmu(dx$GRAVITE)
```

---

```
resume.dateheure
```

*Resume du vecteur des ENTREE ou SORTIE*

---

**Description**

résumé du vecteur vx des ENTREE ou SORTIE

**Usage**

```
summary.dateheure(vx)
```

**Arguments**

vx                      vecteur ENTREE ou SORTIE

**Value**

"n", "n.na", "p.na", "n.rens", "p.rens"

**Examples**

```
summary.ccmu(dx$SORTIE)
```

---

resume.dp	<i>Resume du vecteur DP (diagnostic principal)</i>
-----------	--

---

**Description**

résumé du vecteur vx des DP (diagnostic principal)

**Usage**

```
summary.dp(vx)
```

**Arguments**

vx	vecteur char DP
----	-----------------

**Value**

"n", "n.na", "p.na", "n.rens", "p.rens"

**Examples**

```
summary.dp(dx$DP)
```

---

resume.entree	<i>analyse du vecteur ENTREE ou SORTIE</i>
---------------	--

---

**Description**

analyse du vecteur ENTREE ou SORTIE

**Usage**

```
summary.entree(vx)
```

**Arguments**

vx	vecteur de Date ou de DateTime
----	--------------------------------

**Value**

vecteur nommé: "n", "n.na", "n.rens", "p.rens", "min", "max", "range"

**Note**

min et max ne s'affichent pas sous forme de date. Que donne hms

**Examples**

```
summary.entree(as.Date(pop75$ENTREE))
```

---

resume.mode.sortie	<i>Resume du vecteur vx des MODE_SORTIE</i>
--------------------	---

---

**Description**

résumé du vecteur vx des MODE\_SORTIE

**Usage**

```
summary.mode.sortie(vx)
```

**Arguments**

vx	vecteur char MODE_SORTIE
----	--------------------------

**Value**

"n", "n.na", "p.na", "n.rens", "p.rens", "n.dom", "n.hosp", "n.transfert", "n.mutation", "n.deces",  
"p.dom", "p.hosp", "p.transfert", "p.mutation", "p.deces")

**Examples**

```
summary.mode.sortie(dx$MODE_SORTIE)
```

---

resume.motif	<i>analyse un vecteur de MOTIF</i>
--------------	------------------------------------

---

**Description**

retourne: le nombre d'éléments du vecteur (NA inclus), le nombre de NA, nombre et pourcentage de valeurs renseignées,

**Usage**

```
summary.motif(vx)
```

**Arguments**

vx	vecteur de Char (motif)
----	-------------------------

**Value**

vecteur nommé: "n.na", "p.na", "n.rens", "p.rens"



---

resume.passages	<i>analyse un objet de type duree.passage2</i>
-----------------	--

---

**Description**

analyse un objet de type duree.passage2

**Usage**

summary.passages(dp)

**Arguments**

dp	un objet de type duree.passage2. Correspond à un dataframe d'éléments du RPU dont la durée de passage est conforme cad non nulle et inférieure à 72 heures
----	--

**Value**

n.conforme NB de durées conformes (>0 mn et < 72 heures) duree.moyenne.passage durée moyenne d'un passage en minutes duree.mediane.passage durée médiane d'un passage en minutes duree.moyenne.passage.dom durée moyenne d'un passage en minutes si retour dom duree.mediane.passage.dom durée médiane d'un passage en minutes duree.moyenne.passage.hosp durée moyenne d'un passage en minutes si hospit. duree.mediane.passage.hosp durée médiane d'un passage en minutes n.passage4 nombre de passages de moins de 4 heures n.hosp.passage4 nombre de passages de moins de 4 heures suivi d'hospitalisation n.domicile nombre de retours à domicile n.dom.passage4 nombre de passages de moins de 4 heures suivi d'un retour à domicile n.dom nombre de retours à domicile

---

resume.rpu	<i>calcule le nombre de RPU par SU, territoire de sante et departement.</i>
------------	---

---

**Description**

calcule le nombre de RPU par SU, territoire de santé et département à partir d'un dataframe RPU. Deux colonnes sont indispensables: ENTREE et FINISS

**Usage**

summary.rpu(dx)

**Arguments**

dx	un dataframe RPU ou un dataframe réduit à 2 colonnes: ENTREE et FINISS
----	--

**Details**

v1.0 24/08/2015

**Value**

un objet "list" n nombre total de RPU n.tx total RPU du territoire x n.67 total pour le 67 n.68 total pour 68 n.xxx total pour le Finess xxx p.tx

**Author(s)**

JcB - 2015-08-24

**Source**

summary\_rpu.R

**Examples**

```
s <- summary.rpu(d15); s[1]; s$debut; s$n
```

---

resume.sexe	NA
-------------	----

---

**Description**

retourne: le nombre d'éléments du vecteur (NA inclus), le nombre de NA, nombre et pourcentage de valeurs renseignées, nombre et pourcentage d'hommes et de femmes, sex ratio et taux de masculinité.

**Usage**

```
summary.sexe(vx)
```

**Arguments**

vx                      vecteur de Char (sexe)

**Value**

vecteur nommé: "N", "n.na", "n.rens", "p.rens", "n.hommes", "n.femmes", "p.hommes", "p.femmes", "sex.ratio", "tx.masculinité"

---

resume.transport	<i>analyse du vecteur TRANSPORT</i>
------------------	-------------------------------------

---

**Description**

analyse du vecteur TRANSPORT

**Usage**

```
summary.transport(vx)
```

**Arguments**

vx	vecteur de Factor
----	-------------------

**Value**

"n", "n.na", "p.na", "n.rens", "p.rens", "n.fo", "n.heli", "n.perso", "n.smur", "n.vsav", "n.ambu",  
"p.fo", "p.heli", "p.perso", "p.smur", "p.vsav", "p.ambu"

**Examples**

```
summary.transport(pop75$TRANSPORT)
```

---

rpu.par.jour	<i>Nombre de RPU par jour et par FINESS</i>
--------------	---

---

**Description**

retourne une table contenant le nombre de RPU par jour et par FINESS

**Usage**

```
rpu.par.jour(dx)
```

**Arguments**

dx	un dataframe de type rpu ayant un minimum 2 colonnes ENTREE et FINESS
----	---

**Examples**

```
rpu.par.jour(d04)
```

---

rpu.par.jour2	<i>A partir d'un vecteur de dates, calcule le nombre de RPU par jour</i>
---------------	--

---

**Description**

A partir d'un vecteur de dates, calcule le nombre de RPU par jour

**Usage**

```
rpu.par.jour(d, roll = 7)
```

**Arguments**

d	vecteur de dates compatible avec le format Date
roll:	nb de jours pour la moyenne lissée. Défaut = 7

**Details**

RAJOUTER LES SOMMES CUMuLEES. Nécessite xts, lubridate

**Value**

un dataframe de 4 colonnes: date calendaire, nb de RPU du jour, le n° du jour de l'année (1 à 365), la moyennne lissée

**Examples**

```
p2013 <- rpu.par.jour(j2013$ENTREE)
plot(p2013$V2, type="l") # les RPU
lines(p2013$V3, p2013$V4) # moyenne mobile
```

---

rpu.par.mois	<i>Nombre de RPU par mois</i>
--------------	-------------------------------

---

**Description**

Calcule le nombre de RPU par mois entre deux dates sous forme brute ou corrigée en mois constants de 30 jours.

**Usage**

```
rpu.par.mois(dx, standard = FALSE)
```

**Arguments**

dx	dataframe (au minimum la colonne ENTREE)
standard	(boolean) si true retourne par mois corrigés de 30j sinon le nombre brut de RPU

**Value**

un vecteur nommé: nom du mois, nb de RPU

**Examples**

```
tc1 <- rpu.par.mois(d15, FALSE)
tc2 <- rpu.par.mois(d15, TRUE)
a <- rbind(tc1, tc2)
par(mar=c(5.1, 4.1, 8.1, 2), xpd=TRUE)
barplot(a, beside = TRUE, cex.names = 0.8)
legend("topleft", inset = c(0, -0.1), legend = c("Brut", "Standardisé"), bty = "n", col = c("black", "gray80"), pch
```

rpu2xts

*Transforme RPU eb XTS***Description**

A partir du fichier habituel des RPU retourne un objet xts ayant autant de colonnes qu'il y a de SU dans d plus 2 colonnes supplémentaires: - date de type 'Date' qui sert d'index à xts - total nombre total de RPU par jour

**Usage**

```
rpu2xts(dx)
```

**Arguments**

dx                      un datafrale de type RPU comportant au moins une colonne ENTREE

**Value**

un dataframe avec une colonne 'total'

**Examples**

```
ts <- rpu2xts(d0106p); plot(ts$total); lines(rollapply(ts$total, 7, mean), col="red")
```

---

summary.cp	<i>resume du vecteur CODE_POSTAL (cp)</i>
------------	---

---

**Description**

résumé du vecteur vx des CODE\_POSTAL (cp)

**Usage**

```
summary.cp(vx)
```

**Arguments**

vx	vecteur char CODE_POSTAL
----	--------------------------

**Details**

NECESSITE LA BIBLIOTHEQUE RPU\_Doc/mes.constants

**Value**

- nb de CP renseignés - nb de résidents alsaciens - nb d'étrangers

**Examples**

```
summary.cp(dx$CODE_POSTAL)
```

---

summary.destination	<i>Resume de la DESTINATION</i>
---------------------	---------------------------------

---

**Description**

résumé du vecteur vx des DESTINATION. En cas d'hospitalisation, il y a quatre destinations possibles: MCO, SSR, SLD et PSY. En ca de sortie au domicile: HAD et Structure médico-sociale (EHPAD)

**Usage**

```
summary.destination(dx, correction = TRUE)
```

**Arguments**

dx	dataframe RPU
correction	= TRUE: on ne retient que les destinations correspondant à une hospitalisation

**Details**

MANQUE LE SUMMARY DU VECTEUR.

**Value**

"n", "n.na", "p.na", "n.rens", "p.rens"

---

summary.duree.passage *Resume de la Duree de passage.*

---

**Description**

Résumé de dp. dp est produit par duree.passages2 et se présente sous forme d'un data.frame à 4 colonnes

**Usage**

```
summary.duree.passage(dp)
```

**Arguments**

dp                      un objet de type duree.passage2

**Value**

- nb de durées - min durée - max durée - durée moyenne - durée médiane - écart-type - 1er quartile  
- 3ème quartile

---

summary.orientation *Resume de ORIENTATION*

---

**Description**

résumé du vecteur vx des ORIENTATION

**Usage**

```
summary.orientation(dx, correction = TRUE)
```

**Arguments**

dx                      dataframe RPU  
correction              = TRUE: on ne retient que les orientation correspondant à une hospitalisation

**Value**

"n", "n.na", "p.na", "n.rens", "p.rens", "n.chir", "n.med", "n.obst", "n.si", "n.sc", "n.rea", "n.uhcd",  
 "n.ho", "n.hdt", "n.reo", "n.scam", "n.psa", "p.chir", "p.med", "p.obst", "p.si", "p.sc", "p.rea", "p.uhcd",  
 "p.ho", "p.hdt", "p.reo", "p.scam", "p.psa"

---

summary.wday	Nombre de RPU par jour de semaine
--------------	-----------------------------------

---

**Description**

à partir du vecteur vx des ENTREE, retourne le nombre de RPU pour chaque jour de la semaine

**Usage**

```
summary.wday(vx)
```

**Arguments**

vx	vecteur datetime
----	------------------

**Details**

La semaine américaine est modifiée pour correspondre à la semaine française commençant un lundi.

**Value**

vecteur nommé commençant le lundi

**Examples**

```
summary.wday(dx$ENTREE)
```

---

synthese.completude	Calcule le tableau des taux de completude de l'ensemble des Finess.
---------------------	---

---

**Description**

A partir du dataframe initial (dx) calcule le tableau des taux de complétude de l'ensemble des Finess présents dans dx.

**Usage**

```
synthese.completude(dx)
```



**Arguments**

dx                      dataframe de type RPU

**Details**

Le tableau comporte en ordonnée le nom des établissements, en abscisse les différents items du RPU et à l'intersection ligne/colonne la complétude correspondante. dx peut compter un ou plusieurs Finess et concerner une période variable (semaine, mois, année...) Nécessite la librairie plyr pour la fonction dplyr()

**Value**

un dataframe

**Examples**

```
synthese.completude(dx)
synthese.completude(dx[dx$FINESS == "Hag",]) pour un seul établissement
```

---

tab.completude	<i>tableau de complétude par jour</i>
----------------	---------------------------------------

---

**Description**

faire un tableau de complétude par jour pendant une période donnée Permet de suivre les taux de complétude pour une structure et par période

**Usage**

```
tab.completude(dx, d1, d2, finess = NULL)
```

**Arguments**

dx	dataframe de type RPU
d1	date de début
d2	date de fin
finess	= NULL ou un des finess abrégés autorisés. Si NULL, dx doit être spécifique d'un établissement.

### Examples

```

hus <- d15[d15$FINISS == hus,]
d1 <- as.Date("2015-01-01")
d2 <- as.Date("2015-01-31")
t <- tab.completude(hus, d1, d2)
plot(t[, "DATE DE SORTIE"], type = "l", main = "Mode de sortie", ylab = "Taux de completude")
t.zoo <- zoo(t) # nécessite la librairie zoo
plot(xts(t.zoo$DP, order.by = as.Date(rownames(t.zoo))), las = 2,
     main = "Diagnostic principal", ylab = "Taux de completude", cex.axis = 0.8)
boxplot(t, las = 2, cex.axis = 0.8, ylab = "% de completude", main = "Complétude RPU")

```

---

tarru

*Taux de Recours Regional aux Urgences*


---

### Description

Les RPU générés par les habitants de la région sont comptés à partir du vecteur des codes postaux. Le rapport est calculé en divisant le nombre de RPU régionaux par la population de la région.

### Usage

```
tarru(cp, pop.region, rpu.region)
```

### Arguments

cp	vecteur des codes postaux. Détermine le nb de RPU générés par des Alsaciens
pop.region	population régionale de référence

### Value

un pourcentage

### Examples

```

pop.region <- pop.als.tot.2014 <- 1868773
tarru(dx$CODE_POSTAL, pop.als.tot.2014)

```

---

teste.radar	NA
-------------	----

---

**Description**

data pour créer automatiquement un radar RPU et faire des test

**Usage**

```
teste.radar()
```

**Examples**

```
teste.radar()
```

---

week.rpu	<i>Calcule le nombre de RPU par mois</i>
----------	--

---

**Description**

Calcule le nombre de RPU par mois de tous les ES présents dans le dataframe

**Usage**

```
week.rpu(dx)
```

**Arguments**

dx                    un dataframe de type RPU. Doit comporter au moins une colonne ENTREE

**Details**

Nécessite Lubridate. dx peut regrouper tous les ES ou ne converner qu'un ES Particulier.

**Value**

un vecteur du nombre de RPU par mois

**Examples**

```
s <- week.rpu(dx)
tot <- sum(s) # nombre total de RPU
p = s/tot # % de RPU par semaine
summary(p)
```

---

week.variations	<i>Variation du nombre de RPU par semaine</i>
-----------------	---

---

**Description**

Variation du nombre de RPU par semaine

**Usage**

```
week.variations(vx, last = FALSE)
```

**Arguments**

vx	vecteur du nombre de RPU pr semaine (voir week.rpu)
last	boolean Si TRUE, on élimine la dernière semaine qui est souvent incomplète. FALSE par défaut.

**Value**

un vecteur d'entiers positifs ou négatifs

**Examples**

```
# d3 <- week.rpu(dx[dx$FINISS == "3Fr",])  
# v <- week.variations(d3)
```

# Index

- \*Topic **étoile**
  - radar.completude, 19
- \*Topic **completude**
  - completude, 5
- \*Topic **diagramme**
  - radar.completude, 19
- \*Topic **spider**,
  - radar.completude, 19
- add.territoire, 3
- analyse\_type\_etablissement, 3
- barplot.week.variations, 4
- completude, 5, 20
- completude.time, 5
- copyright, 6
- count.CIM10, 7
- datetime, 8
- df.duree.pas, 8
- duree.passage2, 9
- evolution, 10
- factor2table, 10
- finess2territoires, 11
- format.n, 11
- horaire, 12
- is.present.at, 12
- mn2h, 13
- n.isna, 14
- p.isna, 14
- passage, 15
- passages.en.moins.de.4h, 15
- passages2, 16
- pdsa, 17
- plot.xts2, 17
- print.summary.rpu, 18
- print.table.rpu, 18
- pyramide.age, 19
- radar.completude, 5, 19
- reorder.dataframe.fedoru, 20
- reorder.vector.fedoru, 20
- resume.age, 21
- resume.age.sexe, 21
- resume.ccmu, 22
- resume.dateheure, 22
- resume.dp, 23
- resume.duree.passage
  - (summary.duree.passage), 31
- resume.entree, 23
- resume.mode.sortie, 24
- resume.motif, 24
- resume.passages, 25
- resume.rpu, 25
- resume.sexe, 26
- resume.transport, 27
- rpu.par.jour, 27
- rpu.par.jour2, 28
- rpu.par.mois, 28
- rpu2xts, 29
- summary.cp, 30
- summary.destination, 30
- summary.duree.passage, 31
- summary.orientation, 31
- summary.wday, 32
- synthese.completude, 32
- tab.completude, 33
- taru, 34
- teste.radar, 35
- week.rpu, 35
- week.variations, 36