Package 'prevR'

November 30, 2007

Type Package

Title Prevalence estimation with DHS data - Estimation des prévalences à partir des données EDS

Version 1.11

Date 2007-11-30

Author Joseph LARMARANGE <joseph@larmarange.net> IRD - Centre Muraz with supports of ANRS

Maintainer Joseph LARMARANGE <joseph@larmarange.net>

Depends fields, sp, gstat, maptools

Description Ce package permet d'importer des données de type EDS, de les formater, puis de cartographier les variations spatiales de la prévalence d'un phénomène par estimation de la prévalence de chaque point enquêté et interpolation spatiale. Les résultats peuvent ensuite être exportés vers d'autres logiciels de statistique ou de cartographie. La documentation n'est disponible pour le moment qu'en français.

License CeCILL-C - http://www.cecill.info/

URL http://www.ceped.org/prevR/, http://joseph.larmarange.net/prevR/

R topics documented:

`N.optim` *Propose une valeur optimale pour le paramètre N.*

Description

Calcule et propose une valeur optimale pour le paramètre *N* à partir des caractéristiques de l'échantillon (nombre total d'observations, prévalence globale et nombre de clusters).

Usage

```
N.optim(
   n.total,
   global.prevalence,
   nb.clusters,
   b0 = 14.172,
   b1 = 0.419,
   b2 = -0.361,
   b3 = 0.037
   c = -91.011)
```

Arguments

`n.total`	numeric. Nombre total d'observations valides.
`global.prevalence`	numeric. Prévalence globale de l'échantillon. En pourcents.
`nb.clusters`	numeric. Nombre total de clusters.
`b0`	Paramètre b_0.
`b1`	Paramètre b_1.
`b2`	Paramètre b_2.
`b3`	Paramètre b_3.
`c`	Constante c.

Details

La valeur de $N_{optimal}$ est obtenue à partir de l'équation suivante :

$$N_{optimal} = b_0 * n.total^{b_1} * global.prevalence^{b_2} * nb.clusters^{b_3} + c$$

Les coefficients b_0, b_1, b_2 et b_3 ont été obtenus à partir de la simulation de 24500 enquêtes sur un pays modèle, enquêtes présentant une prévalence globale de 1%, 2%, 5%, 10%, 15%, 30% ou 45%, un effectif total de 5000, 6000, 7200, 8640, 10368, 12442 et 14930, réparti selon 300, 360, 432, 518 ou 622 clusters.

Value

Integer.

References

Joseph LARMARANGE, *Prévalences du VIH en Afrique : validité d'une mesure*, thèse de doctorat en démographie, sous la direction de Benoît FERRY, université Paris Descartes, 2007.
Disponible en ligne sur (http://joseph.larmarange.net/).

Examples

```
data(alicante)

infos.prev(alicante.clust)
N.optim(8000,10.16,401)
```

---

alicante *Données issues d'une simulation d'EDS.*

Description

Ces données sont issues de la simulation d'une Enquête Démographique et de Santé (EDS) sur un pays fictif présentant une prévalence nationale de 10 %. 8000 personnes ont été enquêtées, réparties en 401 clusters. Le pays virtuel a été appelé Alicante, d'où le nom du fichier de données.

Usage

```
data(alicante)
```

Value

Charge 5 objets :

`alicante.bounds`
: data.frame. Frontières d'Alicante. Résultat type de `make.boundary.dcw`.

`alicante.cities`
: data.frame. Principales villes d'Alicante. Résultat type de `make.cities.csv`.

`alicante.clust`
: data.frame. Résultats d'enquêtes. Résultat type de `make.clust.dbf` et de la fonction `calcul.dist.cities`.

`alicante.prev`
: data.frame. Estimation des prévalences à partir des données d'enquêtes. Résultat type de `estimate.prev`.

`alicante.krige`
: SpatialPixelsDataFrame. Interpolations spatiales réalisées à partir des estimations de prévalence. Résultat type de `krige.prev`.

Source

- Joseph Larmarange et al., 2006, 'Cartographier les données des enquêtes démographiques et de santé à partir des coordonnées des zones d'enquête', Chaire Quételet, 29 novembre au 1er décembre 2006, Université Catholique de Louvain, Louvain-la-Neuve, Belgique.
Disponible en ligne à (http://www.uclouvain.be/13881.html).

- Joseph LARMARANGE, *Prévalences du VIH en Afrique : validité d'une mesure*, thèse de doctorat en démographie, sous la direction de Benoît FERRY, université Paris Descartes, 2007.
Disponible en ligne sur (http://joseph.larmarange.net/).

`calcul.dist.cities` *Calcule la distance à la ville la plus proche et recode le milieu de résidence.*

Description

Cette fonction calcule la distance de chaque cluster à la ville la plus proche et recode le milieu de résidence selon l'appartenance ou non à une agglomération urbaine. Un cluster est considéré comme appartenant à une agglomération urbaine s'il est à la fois urbain et situé à une distance inférieure à `dist` de la ville la plus proche.

Usage

```
calcul.dist.cities(
   clust,
   cities,
   dist = 15,
   type = "ask",
   lang = "en",
   var.cities = c("x", "y", "city.name"),
   var.clust = c("x", "y", "residence"),
   urban.code = "Urban",
   dist.func = "rdist.earth",
   miles = FALSE)
```

Arguments

`clust`	data.frame. Une observation par cluster. Typiquement le résultat de la fonction `make.clust.dbf`.
`cities`	data.frame. Une observation par ville. Typiquement le résultat de la fonction `make.cities.csv`.
`dist`	numeric. Distance en dessous de laquelle un cluster urbain est considéré comme appartenant à une agglomération urbaine. Il ne s'agit pas de la taille d'une ville mais d'un paramètre permettant de distinguer les clusters urbains d'une agglomération urbaine des autres clusters urbains. Voir les détails pour l'unité à utiliser.
`type`	character. Prend la valeur `'ask'` ou `'all'`. Voir les détails.
`lang`	character. Permet de choisir la langue des messages utilisateur. `fr` pour le français, `en` pour l'anglais.

var.cities	character vector. Liste spécifiant, dans l'ordre, le nom des variables de cities correspondant à la longitude, la latitude et le nom de chaque ville.
var.clust	character vector. Liste spécifiant, dans l'ordre, le nom des variables de clust correspondant à la longitude, la latitude et le milieu de résidence. Le milieu de résidence doit être de type factor ou character.
urban.code	character. Valeur texte spécifiant comment est codée la valeur *urbain* de la variable *milieu de résidence* dans clust.
dist.func	character. Valeur texte spécifiant le nom de la fonction utilisée pour calculer les distances. Utilisez rdist.earth si vous utilisez des coordonnées longitude/latitude en degrés décimaux. Utilisez rdist pour des distances euclidiennes.
miles	logical. TRUE ou FALSE. Variable transmise à rdist.earth si cette fonction est appelée. Calcul des distances en kilomètres si FALSE et en miles si TRUE.

Details

Si type = 'all', les distances de chaque cluster à l'ensemble des villes spécifiées dans cities sont calculées, puis la ville la plus proche de chaque cluster est sélectionnée. Si type = 'ask', une fenêtre présentant l'ensemble des villes contenues dans cities apparaîtra et vous serez invité à sélectioner les villes que vous souhaitez retenir pour la définition des agglomérations urbaines.

Si vous utilisez la fonction rdist, dist doit être exprimé dans la même unité que les coordonnées des clusters et des villes. Si vos coordonnées sont exprimées en degrés décimaux, utilisez la fonction rdist.earth et précisez avec le paramètre miles si vous souhaitez que les résultats soient exprimés en kilomètres ou en miles. dist devra être exprimé en kilomètres ou en miles selon le cas.

Value

La fonction renvoie clust en lui ajoutant trois variables nommées *dist.city*, *city.name* et *urban.area* (voir les détails ci-dessous). Si clust comporte déjà ces trois variables, les anciennes valeurs sont remplacées par les nouvelles valeurs calculées.

dist.city	numeric. Distance à la ville la plus proche, exprimée avec la même unité que dist.
city.name	character. Nom de la ville la plus proche.
urban.area	factor with 2 levels : *in urban area*, *outside urban area*. Indique si le cluster appartient ou non à l'agglomération urbaine de city.name. Est considéré comme appartenant à une agglomération urbaine tout cluster à la fois urbain et situé à moins de dist de la ville considérée.

Note

Pour un pays de taille moyenne, 15 kilomètres est souvent un bon compromis pour dist. Pour le choix des agglomérations urbaines, voir le tutoriel de prevR ('tutoriel.prevR.pdf').

See Also

verif.urb et infos.prev pour obtenir des statistiques sur les fichiers de données.
map.clust et map.cities pour réaliser des cartes.

Examples

```
## Chargement des données
data(alicante)

## Carte des villes de plus de 100.000 habitants
map.cities(alicante.cities, alicante.bounds, min.population=100000,
   new.window=FALSE, main='Main cities of Alicante')

## Carte des clusters par milieu de résidence
x11()
map.clust(alicante.clust, alicante.bounds, type='urb', new.window=FALSE,
   main='Clusters by type of residence')

## Sélection des villes de plus de 100.000 habitants
main.cities <- alicante.cities[alicante.cities$population > 100000, ]

## Calcul des distances à la ville de plus de 100.000 habitants
## la plus proche et vérification des résulats
alicante.clust <- calcul.dist.cities(alicante.clust, main.cities,
   type='all', dist=25)
verif.urb(alicante.clust)

## Sélection des villes de plus de 250.000 habitants
main.cities <- alicante.cities[alicante.cities$population > 250000, ]

## Calcul des distances à la ville de plus de 250.000 habitants
## la plus proche et vérification des résulats
alicante.clust <- calcul.dist.cities(alicante.clust, main.cities,
   type='all', dist=25)
verif.urb(alicante.clust)
infos.prev(alicante.clust)

## Carte des clusters selon leur appartenance à une agglomération urbaine
x11()
map.clust(alicante.clust, alicante.bounds, type='urb', new.window=FALSE,
   var.urb='urban.area', inverse=TRUE,
        main='Clusters in urban agglomeration')
```

`check.names` *Permet de renommer et de supprimer les colonnes d'un tableau de données.*

Description

Cette fonction vérifie si les noms de colonnes d'un tableau de données dépassent une certaine longueur. Si c'est le cas, l'utilisateur est invité à renommer les noms de colonnes du tableau de données. Il peut également supprimer certaines colonnes par la même occasion. Cette fonction est particulièrement indiquée avant d'exporter des données, notamment aux formatx *dbf* et *shapefile* dans la mesure où les noms de colonnes ne doivent pas dépasser les 10 caractères pour ce type de fichier.

Usage

```
check.names(data, lang='en', size.max = 10)
```

Arguments

`data`	data.frame. Dont on veut vérifier les noms de colonnes.
`lang`	character. Permet de choisir la langue des messages utilisateur. `fr` pour le français, `en` pour l'anglais.
`size.max`	character vector. Nombre maximum de caractères autorisés pour le nom des variables.

Value

Renvoie `data` après avoir renommé les différentes colonnes. Si une colonne a été renommée NULL, alors elle sera supprimée.

Si tous les noms de colonnes de `data` ont une longueur inférieure ou égale à `size.max`, alors `data` est directement renvoyé tel quel par la fonction.

Note

Si, après une saisie par l'utilisateur de nouveaux noms, certains ont toujours une longueur supérieure à `size.max`, alors l'utilisateur sera contraint de modifier à nouveau les noms de colonnes concernés. Afin de repérer facilement les noms de colonnes trop longs, la longueur (en nombre de caractères) de chaque nom est indiquée dans une colonne *size*.

Cette fonction est notamment appelée par `write.prev.shp`.

Examples

```
## Not run:
data(alicante)

alicante.clust.check <- check.names(alicante.clust)
write.dbf(alicante.clust.check, 'alicante_clust.dbf')
## End(Not run)
```

`estimate.prev` *Estime la prévalence de chaque cluster par la méthode des cercles.*

Description

Estime la prévalence de chaque cluster par la méthode des cercles selon trois paramètre *N* (effectif minimum), *R* (rayon maximum) et *U* (prise en compte de l'appartenance à une agglomération urbaine). Plusieurs estimations peuvent être réalisées simultanément, une pour chaque combinaison des paramètres *N*, *R* et *U*.

Usage

```
estimate.prev(
   clust,
   N = seq(100, 500, 50),
   R = Inf,
   U = FALSE,
   dist.func = "rdist.earth",
   miles = FALSE,
```

```
  lang = "en",
  progression = TRUE,
  merge.result = FALSE,
  var.clust = c("x", "y", "n", "nweight", "obs.prevalence",
                                    "urban.area", "city.name"),
  urban.area.code = "in urban area")
```

Arguments

clust	data.frame. Une observation (cluster) par ligne. Typiquement le résultat de la fonction make.clust.dbf. Si le paramètre *U* est utilisé, ce fichier doit comprendre également la distance à la ville, le nom de la ville la plus proche et l'appartenance ou non à une agglomération urbaine, typiquement le résultat de la fonction calcul.dist.cities.
N	integer vector. Entier ou liste d'entiers représentant l'effectif minimum des cercles. Voir les fonctions c et seq pour la génération de listes d'entiers. Si l'on ne souhaite pas utiliser le paramètre *N*, attribuez à N la valeur Inf.
R	integer vector. Entier ou liste d'entiers représentant le rayon maximum des cercles. Voir les fonctions c et seq pour la génération de listes d'entiers. Si l'on ne souhaite pas utiliser le paramètre *R*, attribuez à R la valeur Inf.
U	logical or integer. Spécifie si l'appartenance à une agglomération urbaine doit être prise en compte. TRUE pour la prise en compte, FALSE pour la non prise en compte, 2 si l'on souhaite réaliser les estimations avec et sans prise en compte de ce paramètre.
dist.func	character. Nom de la fonction utilisée pour calculer les distances. Utilisez rdist.earth si vous utilisez des coordonnées longitude/latitude en degrés décimaux. Utilisez rdist pour des distances euclidiennes.
miles	logical. Transmise à rdist.earth si cette fonction est appelée. Calcul des distances en kilomètres si FALSE et en miles si TRUE.
lang	character. Permet de choisir la langue des messages utilisateur. fr pour le français, en pour l'anglais.
progression	logical. Si TRUE, affiche la progression du calcul.
merge.result	logical. Si TRUE, applique la fonction merge.prev aux résultats. Voir *Value*.
var.clust	character vector. Liste correspondant aux noms des différentes variables de clust. Dans l'ordre : longitude, latitude, effectif, effectif pondéré, prévalence observée, appartenance à une agglomération urbaine et nom de la ville la plus proche. Les deux derniers noms n'ont pas besoin d'être précisés si U = FALSE. Les noms par défaut correspondent aux noms de variables produits par make.clust.dbf et calcul.dist.cities.
urban.area.code	character. Valeur texte spécifiant comment est codée la valeur *appartient à une agglomération urbaine* de la variable *appartenance à une agglomération urbaine* dans clust. La valeur par défaut correspond à une sortie de calcul.dist.cities.

Details

Estime la prévalence de chaque cluster pour chaque combinaison des paramètres *N*, *R* et *U*. Pour une combinaison de paramètres, la fonction calcule la distance du cluster à l'ensemble des autres clusters, puis trie les clusters par distance croissante. Si *U* est requis et que le cluster appartient à

une agglomération urbaine, seuls les clusters de la même agglomération urbaine sont sélectionnés. Si *U* est requis et que le cluster n'appartient pas à une agglomération urbaine, seuls les clusters hors agglomération urbaine sont sélectionnés. Si *N* est spécifié (différent de `Inf`), seuls les clusters les plus proches sont sélectionnés de manière à ce que le nombre total de personnes enquêtées (variable *n*) soit au moins égal à *N*. Si *R* est spécifié (différent de `Inf`), seuls les clusters situés à une distance inférieure à *R* sont retenus. La prévalence du cluster est alors estimée sur l'ensemble des clusters sélectionnés en tenant compte de la pondération de chaque cluster (variable *nweight*).

Pour plus de détails, voir le tutoriel de prevR ('tutoriel.prevR.pdf').

Value

Renvoie `clust` en lui ajoutant 8 nouvelles variables, décrites ci-dessous. S'il y a plusieurs combinaisons des trois paramètres *N*, *R* et *U*, `clust` est répété autant de fois qu'il y a de combinaisons.

`est.prevalence`
: Prévalence estimée.

`circle.count`
: Effectif total sur lequel l'estimation a été effectuée.

`circle.radius`
: Rayon du cercle de lissage dans lequel sont contenus les clusters retenus pour l'estimation.

`circle.nb.clusters`
: Nombre de clusters retenus pour l'estimation.

`quality.indicator`
: Indicateur de qualité de l'estimation. Il est obtenu pour chaque cluster selon l'équation suivante :

$$quality.indicator = \frac{circle.radius^2}{\sqrt{circle.count}}$$

Plus la valeur de cet indicateur est élevée, plus l'estimation est incertaine.

`N.parameter`
: Valeur du paramètre *N* pour cette estimation. `Inf` si le paramètre n'a pas été appliqué.

`R.parameter`
: Valeur du paramètre *R* pour cette estimation. `Inf` si le paramètre n'a pas été appliqué.

`U.parameter`
: 0 si le paramètre *U* n'a pas été appliqué. Le nombre d'agglomérations urbaines retenues sinon.

Si `merge.result = TRUE`, `clust` est renvoyé en lui ajoutant autant de fois qu'il y a de combinaisons des trois paramètres, les variables *est.prevalence*, *circle.count*, *circle.radius*, *circle.nb.clusters* et *quality.indicator*. Les noms de ces variables prennent alors comme suffixe :.*Nvaleur-de-n.Rvaleur-de-r.Uvaleur-de-u* (voir `merge.prev`).

Warning

Le temps de calcul de cette fonction peut prendre plusieurs minutes selon la puissance de votre machine. Soyez donc patient.

Note

Pour plus d'informations sur le choix des paramètres, voir le tutoriel de prevR ('tutoriel.prevR.pdf'). La valeur optimale de *N* peut être déterminée à partir du nombre total de personnes testées, du nombre total de clusters et de la prévalence globale. Voir `infos.prev` et `N.optim`. Un bon

compromis pour choisir R consiste à retenir la valeur du neuvième décile de *circle.radius* lorsque seul le paramètre *N* est appliqué avec sa valeur optimale. Voir `infos.prev`.

Il est possible, si `merge.result = FALSE`, d'extraire du data.frame renvoyé les éléments correspondant à une seule combinaison des trois paramètres à l'aide de la fonction `extract.data`.

Les résultats peuvent être exportés au format *texte tabulé* à l'aide de `write.txt`, au format *dbf* à l'aide de `write.dbf` ou encore au format *shape* pour importation dans un logiciel de cartographie à l'aide de `write.prev.shp`.

References

- Joseph Larmarange et al., 2006, 'Cartographier les données des enquêtes démographiques et de santé à partir des coordonnées des zones d'enquête', Chaire Quételet, 29 novembre au 1er décembre 2006, Université Catholique de Louvain, Louvain-la-Neuve, Belgique.
Disponible en ligne à (http://www.uclouvain.be/13881.html).

- Joseph LARMARANGE, *Prévalences du VIH en Afrique : validité d'une mesure*, thèse de doctorat en démographie, sous la direction de Benoît FERRY, université Paris Descartes, 2007.
Disponible en ligne sur (http://joseph.larmarange.net/).

See Also

`infos.prev` pour obtenir des infos sur le résultat produit, `extract.data` et `merge.prev` pour manipuler les résultats, `write.prev.shp`, `write.txt` et `write.dbf` pour exporter les résultats, `krige.prev` pour réaliser une interpolation spatiale.

Examples

```
data(alicante)

#  Premier lissage - juste selon N,
## pour les valeurs 100, 150, 200, 250, 300, 350 et 400

alicante.prev <- estimate.prev(
        alicante.clust,
        N=seq(100, 400, 50),
        R=Inf, U=FALSE)
str(alicante.prev)

alicante.prev.n250 <- extract.data(
        alicante.prev,
        value=c(250, Inf, 0))
str(alicante.prev.n250)
infos.prev(alicante.prev.n250)

str(merge.prev(alicante.prev))

# Second lissage - En utilisant les trois paramètres

alicante.prev <- estimate.prev(
        alicante.clust,
        N=c(250),
        R=c(128,Inf),
        U=2,
        merge.result=TRUE)
str(alicante.prev)
```

`extract.col` *Extrait les colonnes d'un data.frame selon une condition sur leur nom.*

Description

Cette fonction permet d'extraire d'un data.frame les colonnes dont le nom contient une certaine chaîne de caractère.
Typiquement, cette fonction est pratique pour afficher les résultats de `krige.prev` avec `spplot` (voir exemple).

Usage

```
extract.col(
   x,
   value='.pred'
```

Arguments

`x`	data.frame ou SpatialPixelsDataFrame. Dont on veut extraire des colonnes.
`value`	character. Chaîne de caractères recherchées dans le nom des colonnes.

Value

Renvoie `data` avec uniquement les colonnes dont le nom contient `value`.

Examples

```
data(alicante)

spplot(extract.col(alicante.krige))
```

`extract.data` *Extrait les observations d'un data.frame selon une ou plusieurs conditions.*

Description

Cette fonction permet d'extraire d'un data.frame les observations dont les valeurs à une ou plusieurs variables données sont égales à des valeurs passées en paramètre de la fonction.
Typiquement, cette fonction permet de sélectionner les observations correspondant à une combinaison précise des paramètres *N*, *R* et *U* d'un résultat de la fonction `estimate.prev` appelée avec `merge.result = FALSE`.

Usage

```
extract.data(
   data,
   value = "ask",
   lang = "en",
   var.data = c("N.parameter", "R.parameter", "U.parameter"))
```

Arguments

data	data.frame. Dont on veut extraire des observations.
value	character or vector. Si value="ask", la fonction liste les différentes valeurs présentes dans le fichier et propose à l'utilisateur de choisir celles qui lui conviennent. Sinon, liste des valeurs pour lesquelles on sélectionne les observations. Doit alors être de même longueur que var.data
lang	character. Permet de choisir la langue des messages utilisateur. fr pour le français, en pour l'anglais.
var.data	character vector. Liste des variables sur lesquelles porte la sélection.

Details

Seules sont sélectionnées les observations dont les valeurs aux variables var.data correspondent à value. Voir le tutoriel de prevR ('tutoriel.prevR.pdf') pour une démonstration avec value="ask".

Value

Renvoie data avec uniquement les observations sélectionnées.

Note

Cette fonction sert typiquement à sélectionner, dans les résultats de estimate.prev avec merge.result = FALSE, les observations correspondant à une combinaison donnée des trois paramètres.

Elle peut également servir à sélectionner, par exemple, les clusters ruraux ou bien les clusters appartenant à une agglomération donnée. Voir les exemples ci-dessous.

See Also

[.data.frame.

Examples

```
data(alicante)

## Extraction à partir d'un résultat de estimate.prev

alicante.prev <- estimate.prev(
        alicante.clust,
        N=seq(200, 300, 50),
        R=Inf,
        U=FALSE)
str(alicante.prev)
alicante.prev.n250 <- extract.data(
        alicante.prev,
```

```
        value=c(250, Inf, 0))
str(alicante.prev.n250)

## Sélection des clusters ruraux

rur <- extract.data(
        alicante.clust,
        value='Rural',
        var.data='residence')
str(rur)

## Sélection des clusters appartenant à l'agglomération urbaine de la ville A

urb.A <- extract.data(
        alicante.clust,
        value=c('in urban area','A'),
        var.data=c('urban.area','city.name'))
str(urb.A)
```

`infos.prev` *Fournit des informations à propos d'un data.frame de type clust ou prev.*

Description

Affiche le nombre de clusters, d'observations, la prévalence globale, une valeur optimale pour N, le nombre d'agglomérations urbaines et leurs noms, ainsi que les quantiles des rayons des cercles de lissage.

Usage

```
infos.prev(
   prev,
   lang = "en",
   var.n = "n",
   var.nweight = "nweight",
   var.obs.prevalence = "obs.prevalence",
   var.city.name = "city.name",
   var.circle.radius = "circle.radius")
```

Arguments

`prev`	data.frame. Chaque ligne doit correspondre à un cluster.
`lang`	character. Permet de choisir la langue des messages utilisateur. `fr` pour le français, `en` pour l'anglais.
`var.n`	character. Nom de la variable correspondanta aux effectifs observés dans chaque cluster.
`var.nweight`	character. Nom de la variable correspondants aux effectifs pondérés de chaque cluster.

`var.obs.prevalence`
: character. Nom de la variable correspondant à la prévalance observée dans chaque cluster.

`var.city.name`
: character. Optionnel. Nom de la variable correspondant au nom de la ville la plus proche.

`var.circle.radius`
: character. Optionnel. Nom de la variable correspondant aux rayons des cercles de lissage.

Details

Cette fonction affiche les renseignements suivants :

- **nombre de clusters** correspond au nombre de lignes dans `data`.
- **nombre d'observations valides** correspond à la somme de la colonne définie par `var.n`.
- **prévalence globale** correspondant à la moyenne de la colonne définie par `var.obs.prevalence`, pondérée par la colonne définie par `var.nweight`.
- **valeur optimale proposée pour *N*** calculée à partir de la fonction `N.optim`.
- **nombre et nom des villes** si une colonne valide est entrée pour `var.city.name`, le nombre de villes trouvées ainsi que leurs noms.
- **quantile du rayon des cercles** si une colonne valide est entrée pour `var.circle.radius`, les quantiles à 50%, 75%, 80%, 85%, 90%, 95% et 99% de cette colonne.

Value

Cette fonction renvoie la valeur `NULL`.

Warning

Si vous souhaitez appliquer cette fonction à un résultat obtenu avec `estimate.prev` appelée avec plusieurs combinaisons des trois paramètres et `merge.result = FALSE`, utilisez `extract.data` ou `merge.prev` avant d'utiliser `infos.prev`.

See Also

`N.optim` pour les détails concernant le calcul d'une valeur optimale de N.

Examples

```
data(alicante)

infos.prev(alicante.clust)

infos.prev(alicante.prev,var.circle.radius='circle.radius.N250.RInf.U6')
```

`krige.prev` *Réalise des interpolations spatiales par krigeage et/ou selon une pondération inverse à la distance.*

Description

Cette fonction met en forme les données contenues dans un data.frame et permet de réaliser des interpolations spatiales sur une grille régulière en ayant recours aux fonctions du package **gstat**. Elle permet également de guider l'utilisateur pas à pas pour choisir le modèle de variogramme utilisé.

Usage

```
krige.prev(
   data,
   formula = est.prevalence ~ 1,
   locations = ~x + y,
   type = "ask",
   boundary = NULL,
   cell.size = 0.05,
   ask.cell.size = TRUE,
   lang = "en",
   model = vgm(1, "Exp", 1),
   idp = 2,
   show.variogram = TRUE, ...)
```

Arguments

`data`	data.frame. Chaque ligne doit correspondre à une observation ponctuelle. ATTENTION : deux observations ne peuvent avoir les mêmes coordonnées.
`formula`	formula or formulae list. Formule ou liste de formules spécifiant les interpolations à réaliser. Chaque formule définit la variable dépendante comme un modèle linéaire de variables indépendantes. Pour un krigeage ordinaire ou une interpolation selon l'inverse de la distance, utilisez une formule du type `z~1`. Voir `krige` pour plus de détails.
`locations`	formula. Spécifie sous la forme d'une formule les colonnes de `data` correspondant aux coordonnées géographiques des observations. Si, par exemple, la longitude correspond à la colonne `x` et la latitude à la colonne `y`, entrez `~x+y`.
`type`	character. Peut prendre les valeurs *ask*, *auto*, *model* ou *idw*. Voir les détails.
`boundary`	data.frame. Optionnel. Frontières de la zone géographique étudiée, sous la forme d'une série de points dessinant un polygone fermé. Les coordonnées des points doivent être inscrites dans des colonnes ayant les mêmes noms que ceux définis dans `locations` ou bien nommées *x* et *y*. Les unités doivent évidemment être identiques.
`cell.size`	numeric. Permet de définir la taille de chaque cellule de la grille sur laquelle l'interpolation spatiale sera réalisée. S'exprime dans la même unité que celle utilisée pour les coordonnées des points de `data`.
`ask.cell.size`	logical. Si `TRUE`, calcule et affiche la taille de la grille engendrée par la valeur de `cell.size` et permet à l'utilisateur de modifier cette dernière.

`lang` character. Permet de choisir la langue des messages utilisateur. `fr` pour le français, `en` pour l'anglais.

`model` object or list of objects of class `variogramModel`. Typiquement un résultat de la fonction `vgm`. S'il s'agit d'une liste de modèles de variogramme, elle doit avoir la même longueur que `formula`. Pour générer cette liste, ayez recours à la fonction `list`. Le recours à la fonction `c` provoquerait une erreur. Les différents types de modèles de variogrammes peuvent être visualisés à l'aide de la fonction `show.vgms`. Non requis si `type = 'idw'`. Voir les détails.

`idp` numeric or numeric vector. Requis uniquement si `type = 'idw'`. Spécifie la puissance de la pondération selon l'inverse de la distance. Voir `idw`. S'il s'agit d'un vecteur, il doit avoir la même longueur que `formula`.

`show.variogram` logical. Indique si les graphiques des variogrammes empiriques et des modèles utilisés doivent être affichés à la fin du calcul. Voir les exemples. Sans effet si `type = 'idw'`.

`...` Autres arguments passés à chaque appel de `krige` ou d'`idw` selon le cas. Permet de réaliser des interpolations locales et/ou par blocs. Voir la documentation de ces fonctions.

Details

Si `type='model'`, une interpolation spatiale par krigeage est réalisée pour chaque élément de `formula` en utilisant le même modèle de variogramme pour tous si un seul modèle est fourni dans `model`. Utilise chaque modèle respectivement s'il s'agit d'une liste de modèles de variogrammes.

Si `type='auto'`, les interpolations spatiales par krigeage sont réalisées pour chaque élément de `formula` en utilisant les modèles obtenus par la fonction `fit.variogram` qui permet d'ajuster un modèle de variogramme à un variogramme expérimental. Les valeurs de `model` sont passées à `fit.variogram` et servent de valeurs de départ pour l'ajustement. Pour plus de détails, voir l'aide de cette fonction.

Si `type='ask'`, pour chaque élément de `formula`, un modèle ajusté est calculé à partir du variogramme expérimental, `model` servant de valeurs de départ. Le variogramme expérimental et le modèle ajusté sont ensuite représentés graphiquement. Une invite utilisateur permet d'accepter le modèle ajusté ou bien d'effectuer un ajustement manuel du modèle. Voir le tutoriel de prevR ('tutoriel.prevR.pdf') pour des exemples.

Si `type='idw'`, une interpolation spatiale par pondération selon l'inverse de la distance est réalisée pour chaque élément de `formula`, les valeurs de `idp` étant passées à la fonction `idw`.

Value

Objet de la classe `SpatialPixelsDataFrame`. Pour chaque variable interpolée, deux variables sont produites : la prédiction de l'interpolation et la variance de cette prédiction (krigeage uniquement). Elles portent le nom de la variable suivi respectivement des suffixes *.pred* et *.var*. Si `boundary` a été spécifié, les limites de la grille des résultats correspondent aux limites de la frontière et les points situés en-dehors du polygone défini par `boundary` sont attribués d'une valeur manquante `NA`. Sinon, les limites de la grille correspondent aux coordonnées limites des points contenus dans `data`.

Warning

Le temps de calcul de cette fonction peut prendre plusieurs minutes selon la puissance de votre machine. Soyez donc patient.

Note

Les résultats de cette fonction peuvent être cartographiés à l'aide la fonction `spplot` et exportés au format *asc* pour importation dans un logiciel de cartographie à l'aide de la fonction `write.asciigrid`.Voir les exemples.

Pour plus de détails, notamment sur l'utilisation de la fonction lorsque `type='ask'` et le choix d'un modèle de variogramme, voir le tutoriel de prevR ('tutoriel.prevR.pdf').

See Also

`krige` pour les détails concernant le krigeage, `idw` pour l'interpolation selon l'inverse de la distance, `write.asciigrid` pour l'exportation des résultats, `spplot` pour représenter graphiquement les résultats, `vgm` pour spécifier un modèle de variogramme, `show.vgms` pour afficher les différents types de modèles de variogrammes et `extract.col` pour extraire certaines colonnes.

Examples

```
data(alicante)

## Krigeage

show.vgms()
alicante.krige <- krige.prev(alicante.prev, boundary=alicante.bounds,
        formula=c(est.prevalence.N100.RInf.U0~1,
                est.prevalence.N250.RInf.U0~1,
                est.prevalence.N250.R128.U0~1,
                est.prevalence.N250.R128.U6~1,
                circle.radius.N250.R128.U6~1,
                quality.indicator.N250.R128.U6~1),
        model=list(vgm(53.58,'Exp',0.56),
                vgm(48.97,'Exp',1.06),
                vgm(49.19,'Exp',1.07),
                vgm(42.47,'Exp',0.58),
                vgm(1056,'Exp',0.89),
                vgm(150000,'Exp',2.1)),
        type='model', ask.cell.size=FALSE,
                cell.size=0.075, show.variogram=FALSE)

## Représentation graphique

spplot(extract.col(alicante.krige))

spplot(alicante.krige,
        zcol = c('est.prevalence.N100.RInf.U0.pred',
                'est.prevalence.N250.RInf.U0.pred',
                'est.prevalence.N250.R128.U0.pred',
                'est.prevalence.N250.R128.U6.pred'),
        col.regions=prevR.colors.red(21),
        cuts=20,
        main='Estimated prevalence with several parameters')
x11()
spplot(alicante.krige,
        zcol = 'circle.radius.N250.R128.U6.pred',
        col.regions=prevR.colors.blue(21),
        cuts=20,
        main='Radius of circles - N=250 R=128 U=6')
```

```
x11()
spplot(alicante.krige,
        zcol = 'quality.indicator.N250.R128.U6.pred',
        col.regions=prevR.colors.green.inverse(21),
        cuts=20,
        main='Quality indicator - N=250 R=128 U=6')

## Exportation au format asc

## Not run:
write.asciigrid(alicante.krige, 'alicante-krige.asc')
## End(Not run)
```

`make.boundary.dcw` *Extrait les coordonnées des frontières d'un pays à partir d'un fichier de points du DCW.*

Description

Permet d'importer les coordonnées longitude/latitude des frontières d'un pays à partir d'un fichier de points fourni par le Digital Chart of the World (`http://www.maproom.psu.edu/dcw/`).

Usage

```
make.boundary.dcw(file, progression = TRUE, lang = "en")
```

Arguments

`file`	character. Nom du fichier texte récupéré par le Digital Chart of the World. Typiquement, ces fichiers sont nommés *country-name2pts.txt*.
`progression`	logical. Permet d'afficher des messages indiquant la progression de l'analyse du fichier.
`lang`	character. Permet de choisir la langue des messages utilisateur. `fr` pour le français, `en` pour l'anglais.

Details

Lit le fichier texte. Si celui-ci contient les coordonnées de plusieurs polygones, les affiche et demande à l'utilisateur de choisir le polygone principal.

Value

data.frame avec deux colonnes, *x* et *y*.

Note

La lecture ligne à ligne du fichier texte peut être longue. Soyez patient.

Pour un exemple concret, voir le tutoriel de prevR ('tutoriel.prevR.pdf').

Examples

```
## Not run:

burkina.bounds <- make.boundary.dcw('burkina_faso2pts.txt')

## End(Not run)
```

`make.cities.csv` *Génère un data.frame avec les coordonnées de villes à partir d'un fichier csv.*

Description

Importe un fichier csv, liste les différentes colonnes et demande à l'utilisateur de sélectionner les données correspondant à la longitude, la latitude, le nom et la population de chaque ville. Permet typiquement d'importer des données à partir du *Global Rural - Urban Mapping Project (GRUMP)*, disponibles gratuitement à http://sedac.ciesin.columbia.edu/gpw/.

Usage

```
make.cities.csv(file, lang = "en")
```

Arguments

`file`	character. Nom du fichier csv à importer. Les données doivent être séparées par des virgules, et le séparateur de décimales être le point.
`lang`	character. Permet de choisir la langue des messages utilisateur. `fr` pour le français, `en` pour l'anglais.

Details

Une interface textuelle guide l'utilisateur.

Value

Renvoie un data.frame avec 4 colonnes :

`city.name`	Nom de la ville
`x`	Longitude du centre de la ville
`y`	Latitude du centre de la ville
`population`	Population de la ville

Les données sont triées par population décroissante

Note

Pour un exemple concret, voir le tutoriel de prevR ('tutoriel.prevR.pdf').

Examples

```
## Not run:
burkina.cities <- make.cities.csv('bfapv1.csv')
## End(Not run)
```

`make.clust.dbf` *Lit et met en forme les données d'EDS.*

Description

Cette fonction lit les données GPS d'une Enquête Démographique et de Santé (EDS), fournies sous la forme d'un fichier *dbf* sur `http://www.measuredhs.com`, puis agrège les données individuelles préparées avec `make.ind.spss` pour produire un data.frame exploitable par prevR.

Usage

```
make.clust.dbf(file, ind, lang = "en")
```

Arguments

`file`	character. Nom du fichier *dbf* comportant les coordonnées de chaque cluster.
`ind`	data.frame. Données individuelles préparées avec `make.ind.spss`.
`lang`	character. Permet de choisir la langue de l'interface utilisateur. `fr` pour le français, `en` pour l'anglais.

Details

Lit le fichier `file` et demande à l'utilisateur les variables suivantes : *numéro du cluster*, *longitude*, *latitude*, *milieu de résidence*, *code de la région* et *nom des régions*. La fonction prépare les données sous forme d'un data.frame et agrège les données de `ind` pour chaque cluster. Si les variables *sex* et *age* sont présentes dans `ind`, il est possible de restreindre l'analyse à une tranche d'âges et/ou à un sexe donné.

Value

Renvoie un data.frame, où chaque ligne correspond à un cluster, avec les colonnes suivantes :

`cluster`	Identifiant du cluster.
`x`	Longitude.
`y`	Latitude.
`residence`	Milieu de résidece.
`region`	Code de la région.
`region.name`	Nom de la région.
`n`	Effectif, soit le nombre de personnes observées dans le cluster (résultats indéterminés et personnes ayant une pondération nulle exclue).
`nweight`	Effectif pondéré, soit la somme des poids de chaque individu du cluster.
`obs.prevalence`	Prévalence observée dans le cluster, en tenant compte des pondérations. Est exprimée en pourcents.

Note

Voir le tutoriel de prevR ('tutoriel.prevR.pdf') pour plus de renseignements.

See Also

`make.ind.spss`.

Examples

```
## Not run:
burkina.ind <- make.ind.spss('burkina_hiv.sav', lang='fr')
burkina.clust <- make.clust.dbf('burkina_gps.dbf', burkina.ind, lang='fr')
## End(Not run)
```

`make.ind.spss` *Prépare les données individuelles d'une EDS.*

Description

Cette fonction lit les données individuelles d'une Enquête Démographique et de Santé (EDS), fournies sous la forme d'un fichier *SPSS* sur `http://www.measuredhs.com`, pour produire un data.frame exploitable par *make.clust.dbf*.

Usage

```
make.ind.spss(file, lang = "en")
```

Arguments

`file`	character. Nom du fichier de données individuelles. Doit être au format *sav* de SPSS.
`lang`	character. Permet de choisir la langue de l'interface utilisateur. `fr` pour le français, `en` pour l'anglais.

Details

Lit le fichier `file` et demande à l'utilisateur les variables suivantes : *identifiant des individus*, *numéro du cluster de l'individu*, *âge*, *sexe*, *variable analysée* et *poids statistique de chaque individu*. Si *numéro du cluster de l'individu* n'est pas renseigné, il sera calculé à partir de l'*identifiant*.

L'utilisateur sera invité à spécifier les codes de la *variable analysée* correspondant à un résultat positif, un résultat négatif et à un résultat indéterminé.

âge et *sexe* sont optionnels. Il s'agit de deux variables, respectivement numérique et qualitative, pouvant être utilisées par `make.clust.dbf` pour restreindre l'analyse à une sous-population.

Si la *pondération statistique* n'est pas précisée, chaque individu aura un poids égal à 1. Si cette variable est spécifiée, il est possible de lui appliquer un facteur multiplicatif. Souvent, dans les EDS, la pondération est indiquée dans une variable qui doit être divisée par 1 000 000.

Value

Renvoie un data.frame, où chaque ligne correspond à un individu, avec les colonnes suivantes :

`id`	Identifiant de l'individu.
`cluster`	Numéro du cluster.
`age`	Variable de type *integer*.

`sex` Variable de type *factor*.

`original.result`
Valeurs de la variable analysée telles qu'enregistrées dans `file`.

`weight` Poids statistique de chaque individu.

`result` Variable analysée recodée en *Negative*, *Positive* ou *NA*.

Si certaines variables n'ont pas été spécifiées par l'utilisateur, les colonnes correspondantes seront inexistantes dans le data.frame retourné.

Note

Voir le tutoriel de prevR ('tutoriel.prevR.pdf') pour plus de renseignements.

See Also

`make.clust.dbf`.

Examples

```
## Not run:
burkina.ind <- make.ind.spss('burkina_hiv.sav', lang='fr')
burkina.clust <- make.clust.dbf('burkina_gps.dbf', burkina.ind, lang='fr')
## End(Not run)
```

---

map.cities *Représente les villes contenues dans un data.frame sur une carte.*

Description

Permet de représenter sur une carte les points contenus dans un data.frame ainsi que le nom de chaque point.

Usage

```
map.cities(
   cities,
   boundary,
   var.cities = c("x", "y", "city.name", "population"),
   min.population = NULL,
   new.window = TRUE,
   ...)
```

Arguments

`cities` data.frame. Contient les villes/points à représenter.

`boundary` data.frame. Contient deux colonnes, *x* et *y*. Série de points définisant un polygone correspondant aux limites de la zone étudiée.

`var.cities` character vector. Liste des noms des variabales de `cities` correspondant, dans l'ordre, à la longitude, la latitude, le nom et la population associés à chaque ville/point. Si `min.population = NULL`, la quatrième valeur de `var.cities` n'est pas requise.

`min.population`
: numeric. Si précisé, seules les villes ayant une population au moins égale à cette valeur seront représentées.

`new.window`
: logical. Indique si la carte doit être effectuée dans une nouvelle fenêtre.

`...`
: Paramètres supplémentaires passés à la fonction `title`. Voir l'aide de cette fonction.

Details

Dessine le polygone défini par `boundary`, puis les différents points sélectionnés de `cities` et ajoute leur nom à côté de chacun d'eux.

Examples

```
data(alicante)

map.cities(
        alicante.cities,
        alicante.bounds,
        main='Cities of Alicante')

map.cities(
        alicante.cities,
        alicante.bounds,
        min.population=100000,
        main='Main cities of Alicante')
```

---

map.clust *Permet de cartographier les zones d'enquêtes ou clusters.*

Description

Affiche sur une carte les clusters enquêtés selon leur milieu de résidence, le nombre de personnes enquêtées ou selon le nombre de personnes présentant la caractéristique étudiée (nombre de cas positifs).

Usage

```
map.clust(
   clust,
   boundary,
   type = "urb",
   lang = "en",
   var.urb = "residence",
   var.n = "n",
   var.obs.prevalence = "obs.prevalence",
   var.coords = c("x", "y"),
   inverse = FALSE,
   add.legend = TRUE,
   legend.location = "bottomright",
   factor.size = 0.2,
   new.window = TRUE,
   ...)
```

Arguments

clust	data.frame. Il doit comporter une ligne par cluster. Typiquement le résultat de make.clust.dbf.
boundary	data.frame. Doit contenir deux colonnes, *x* et *y*. Série de points définissant un polygone correspondant aux limites de la zone étudiée.
type	character. Peut prendre les valeurs *urb*, *flower* ou *count*. Voir les détails.
lang	character. Permet de choisir la langue de l'interface utilisateur. fr pour le français, en pour l'anglais.
var.urb	character. Nom de la colonne de clust utilisée pour représenter le type des clusters si type= 'urb'. Non requise sinon.
var.n	character. Nom de la colonne de clust utilisée pour représenter la taille des clusters si type= 'count'. Requise également si type= 'flower'.
var.obs.prevalence	character. Nom de la colonne de clust correspondant à la prévalence observée de chaque cluster. Requise uniquement si type= 'flower'.
var.coords	character vector. Liste des noms des deux variables de clust correspondant à la longitude et la latitude de chaque cluster.
inverse	logical. Utilisé si type= 'urb'. Si TRUE, inverse l'ordre des modalités. Voir détails.
add.legend	logical. Indique si une légende doit être affichée.
legend.location	Position de la légende. Variable transmise à legend. Voir la section *Details* de l'aide de cette fonction. Le plus simple consiste à utiliser les termes *bottomright*, *bottom*, *bottomleft*, *left*, *topleft*, *top*, *topright*, *right* and *center*.
factor.size	numeric. Utilisé si type= 'count'. Facteur d'agrandissement ou de réduction de la taille des cercles.
new.window	logical. Indique si la carte doit être dessinnée dans une nouvelle fenêtre.
...	Paramètres supplémentaires passés à la fonction title. Voir l'aide de cette fonction.

Details

Dessine tout d'abord le polygone défini par boundary. Si type = 'urb', représente les clusters, selon leur milieu de résidence, par des points verts et rouges. Par défaut, le vert est utilisé pour coder la première modalité, d'un point de vue alphabétique, de la colonne définie par var.urb. Si inverse = TRUE, le vert sera utilisé pour la dernière modalité, d'un point de vue alphabétique.

Si type = 'flower', la carte générée représentera le nombre d'obervations positives par cluster. Un cluster sans cas positif sera représenté par un point vert, un cluster avec un seul cas positif par un point mauve, un cluster avec plusieurs cas positifs par un point mauve et autant de 'rayons' rouges que de cas positifs. Voir sunflowerplot pour plus de détails sur ce type de graphiques.

Si type = 'count', la carte représente le nombre d'observations de chaque cluster par un cercle proportionnel au nombre d'observatuibs du cluster. La taille des cercles peut être contrôlée par factor.size. Pour un exemple d'exportation des résultats vers un logiciel de dessin afin d'appliquer une transparence aux cercles, voir le tutoriel de prevR ('tutoriel.prevR.pdf').

Examples

```
data(alicante)

map.clust(alicante.clust, alicante.bounds,
        type='count',
        new.window=FALSE,
        main='Number of tested persons by cluster',
        factor.size=0.15)

map.clust(alicante.clust, alicante.bounds,
        type='flower',
        main='Number of HIV positive persons by cluster')

map.clust(alicante.clust, alicante.bounds,
        type='urb',
        main='Clusters by type of residence')

map.clust(alicante.clust, alicante.bounds,
        type='urb',
        var.urb='urban.area',
        inverse=TRUE,
        main='Clusters in urban agglomeration')
```

---

merge.prev *Réorganise les résultats de estimate.prev.*

Description

Réorganise un data.frame produit par la fonction *estimate.prev* en un data.frame comportant un seul cluster par ligne, afin de permettre, par exemple, de réaliser plusieurs interpolations spatiales simultanément avec *krige.prev*.

Usage

```
merge.prev(data)
rename.variables.parameters(data)
```

Arguments

`data`	data.frame. L'appelation des colonnes doit correspondre à celui produit par `estimate.prev`.

Details

La fonction `rename.variables.parameters` supprime les colonnes *N.parameter*, *R.parameter* et *U.parameter* de `data` ; renomme les colonnes *est.prevalence*, *circle.count*, *circle.radius*, *circle.nb.clusters* et *quality.indicator* en *xxx.Naa.Rbb.Ucc* où *xxx* correspond à l'ancien nom de la colonne, *aa* à la valeur de *N.parameter*, *bb* à la valeur de *R.parameter* et *cc* à la valeur de *U.parameter*.

La fonction `merge.prev` extrait les données de `data` pour chaque combinaison des paramètres *N*, *R* et *U*, leur applique `rename.variables.parameters` et fusionne les données en un seul data.frame.

Value

data.frame avec un seul cluster par ligne. Voir les exemples.

Note

`merge.prev` est appliquée directement au résultat de `estimate.prev` si `merge.result = TRUE`.

See Also

`estimate.prev`.

Examples

```
data(alicante)

alicante.prev <- estimate.prev(
        alicante.clust,
        N=c(100,250),
        R=c(128,Inf),
        U=2)
str(alicante.prev)

alicante.prev.n250 <- extract.data(
        alicante.prev,
        value=c(250, Inf, 0))
str(alicante.prev.n250)

alicante.prev.n250 <- rename.variables.parameters(alicante.prev.n250)
str(alicante.prev.n250)

alicante.prev <- merge.prev(alicante.prev)
str(alicante.prev)
```

---

prevR-package *Prevalence estimation with DHS data - Estimation des prévalences à partir des données EDS.*

Description

Ce package permet d'importer des données de type EDS (Enquête Démographique et de Santé), de les formater, puis de cartographier la prévalence d'un phénomène par estimation de la prévalence de chaque point enquêté et interpolation spatiale. Les résultats peuvent ensuite être exportés vers d'autres logiciels de statistiques ou de cartographie (SIG). La documentation n'est disponible pour le moment qu'en français.

Details

Package:	prevR
Type:	Package
Version:	1.11
Date:	2007-11-30
License:	CeCILL-C - http://www.cecill.info/
Web:	http://www.ceped.org/prevR/, http://joseph.larmarange.net/prevR/
License:	CeCILL-C - http://www.cecill.info/

Pour plus d'informations sur la manière d'utiliser ce package, nous vous conseillons la lecture du tutoriel de prevR ('tutoriel.prevR.pdf').

Author(s)

Joseph LARMARANGE <joseph@larmarange.net> IRD - Centre Muraz with supports of ANRS

Maintainer: Joseph LARMARANGE <joseph@larmarange.net>

References

- Joseph Larmarange et al., 2006, 'Cartographier les données des enquêtes démographiques et de santé à partir des coordonnées des zones d'enquête', Chaire Quételet, 29 novembre au 1er décembre 2006, Université Catholique de Louvain, Louvain-la-Neuve, Belgique.
Disponible en ligne à (http://www.uclouvain.be/13881.html).

- Joseph LARMARANGE, *Prévalences du VIH en Afrique : validité d'une mesure*, thèse de doctorat en démographie, sous la direction de Benoît FERRY, université Paris Descartes, 2007.
Disponible en ligne sur (http://joseph.larmarange.net/).

See Also

Packages nécessaires pour l'exécution de prevR : fields sp gstat maptools

Examples

```
## Pour une démonstration des possibilités de prevR

demo(prevR)
```

---

prevR.colors *Palettes de couleurs continues.*

Description

Fonctions générant des palettes de couleurs utilisables par les fonctions graphiques de R, en particulier spplot. Elles créent des palettes de couleurs continues, les contrastes étant renforcés par l'éclaircissement ou l'assombrissement des valeurs extrêmes.

Usage

```
prevR.demo.pal(n)
prevR.demo.pal(n, border, main, ch.col)
prevR.colors.red(n)
prevR.colors.red.inverse(n)
prevR.colors.blue(n)
prevR.colors.blue.inverse(n)
prevR.colors.green(n)
prevR.colors.green.inverse(n)
prevR.colors.gray(n)
prevR.colors.gray.inverse(n)
```

Arguments

n	integer. Nombre de couleurs constituant la palette.
border	color. Couleur de la bordure des cases.
main	character. Titre du graphique.
ch.col	character vector. Liste des fonctions à représenter.

Details

Le code de prevR.demo.pal a été repris sur celui de la fonction demo.pal décrite dans les exemples de la documentation de rainbow.

prevR.colors.red réalise un gradient allant du blanc/jaune au rouge/rouge foncé.
prevR.colors.blue réalise un gradient allant du bleu pâle au bleu foncé.
prevR.colors.green réalise un gradient allant du vert pâle au vert foncé.
prevR.colors.gray réalise un gradient allant du blanc/gris clair au gris foncé/noir.

Les fonctions avec le suffixe *.inverse* réalisent les mêmes gradients mais en partant des couleurs foncées vers les couleurs claires.

Value

prevR.demo.pal affiche les différentes palettes. Les autres fonctions renvoient une liste de couleurs codées de manière hexadécimale. Pour récupérer la liste des couleurs au format RGB (pour Red Green Blue), utilisez la fonction col2rgb.

See Also

D'autres palettes de couleurs existent sous R. Voir rainbow ainsi que le package **RColorBrewer**.

Examples

```
## Affiche les différentes palettes
prevR.demo.pal(25)

## Exemples d'utilisation avec spplot()

data(alicante)

# Représentation graphique
```

```
x11()
spplot(alicante.krige,
        c('est.prevalence.N100.RInf.U0.pred',
                'est.prevalence.N250.RInf.U0.pred',
                'est.prevalence.N250.R128.U0.pred',
                'est.prevalence.N250.R128.U6.pred'),
        col.regions=prevR.colors.red(21),
        cuts=20,
        main='Estimated prevalence with several parameters')

x11()
spplot(alicante.krige,
        'circle.radius.N250.R128.U6.pred',
        col.regions=prevR.colors.blue(21),
        cuts=20,
        main='Radius of circles - N=250 R=128 U=6')

x11()
spplot(alicante.krige,
        'quality.indicator.N250.R128.U6.pred',
        col.regions=prevR.colors.green.inverse(21),
        cuts=20,
        main='Quality indicator - N=250 R=128 U=6')
```

`read.spss2` *Lire un fichier SPSS au format sav.*

Description

Cette fonction est identique à `read.spss` exceptée sur un point. En effet, si dans le fichier SPSS, une variable comporte des étiquettes de valeurs non présentes dans les données, `read.spss` renvoie une colonne de type *atomic* tandis que `read.spss2` renvoie une colonne de type factor.

Usage

```
read.spss2(
   file,
   use.value.labels = TRUE,
   to.data.frame = FALSE,
   max.value.labels = Inf,
   trim.factor.names = FALSE)
```

Arguments

`file`	character. Nom du fichier *sav* à lire et transformer en data.frame.
`use.value.labels`	logical. Convertir les variables avec des étiquettes de valeurs en colonnes de type *factor* ?
`to.data.frame`	logical. Renvoyer un data.frame ? Renvoie une liste sinon.

`max.value.labels`
: logical. Seules les variables ayant au maximum ce nombre de modalités seront converties en *factor*.

`trim.factor.names`
: logical. Supprimer les espaces des *factor levels* ?

Details

Cette fonction est identique à read.spss exceptée sur un point. Si, dans le fichier SPSS, une variable comporte des étiquettes de valeurs non présentes dans les données, read.spss renvoie une colonne de type *atomic* tandis que `read.spss2` renvoie une colonne de type factor.

See Also

`read.spss` pour plus de détails sur cette fonction.

Examples

```
## Not run:
read.spss('file.sav')
## End(Not run)
```

---

verif.urb *Calcule des statistiques sur les agglomérations urbaines.*

Description

Calcule, pour chaque agglomération urbaine présente dans le fichier de donnée, la prévalence observée dans l'agglomération considérée et permet de comparer ces données avec celles provenant d'autres sources.

Usage

```
verif.urb(
   clust,
   conf.level = 0.9,
   add = FALSE,
   lang = "en",
   var.clust = c("n", "nweight", "obs.prevalence",
                                    "urban.area", "city.name"),
   urban.area.code = "in urban area")
```

Arguments

`clust`	data.frame. Chaque ligne doit correspondre à un cluster.
`conf.level`	numeric. Niveau de confiance pour les tests statistiques.
`add`	logical. Si `TRUE`, invite l'utilisateur à ajouter manuellement des données provenant d'une autre source.
`lang`	character. Permet de choisir la langue des messages utilisateur. `fr` pour le français, `en` pour l'anglais.

`var.clust` character vector. Noms des variables de `clust` correspondant dans l'ordre au nombre d'observations valides, à l'effectif pondéré, à la prévalence observée, à l'appartenance à une agglomération urbaine et au nom de la ville la plus proche de chaque cluster.

`urban.area.code` character. Valeur du facteur indiquant l'appartenance à une agglomération urbaine.

Details

Retient pour chaque agglomération urbaine les clusters appartenant à la dite agglomération et calcule une prévalence observée sur ces clusters.

Les intervalles de confiance sont calculés à l'aide de la fonction `prop.test`. Les comparaisons de deux proportions sont calculées à l'aide de la fonction `fisher.test`. Les effectifs pondérés de chaque cluster sont pris en compte.

Value

data.frame avec les colonnes suivantes :

`city` factor. Nom de l'agglomération.

`city.nb.cluster` numeric. Nombre de clusters appartenant à l'agglomération.

`city.n` numeric. Nombre d'observations valides de l'agglomération.

`city.prevalence` numeric. Prévalence observée dans l'agglomération, en pourcents.

`city.low` numeric. Valeur basse de l'intervalle de confiance de la prévalence observée.

`city.high` numeric. Valeur haute de l'intervalle de confiance de la prévalence observée.

`conf.level` numeric. Niveau de confiance pour les intervalles.

`add.prevalence` numeric. Prévalence de la source additionnelle saisie, en pourcents.

`add.n` numeric. Effectif de la source additionnelle.

`add.low` numeric. Valeur basse de l'intervalle de confiance de la prévalence additionnelle.

`add.high` numeric. Valeur haute de l'intervalle de confiance de la prévalence additionnelle.

`p.value.comparaison` numeric. Résultat du test de comparaison des deux proportions.

Note

Pour des exemples d'utilisation de cette fonction, en particulier avec `add = TRUE`, voir le tutoriel de prevR ('tutoriel.prevR.pdf').

Examples

```
data(alicante)

main.cities <- alicante.cities[alicante.cities$population > 100000, ]
alicante.clust <- calcul.dist.cities(
        alicante.clust,
```

```
        main.cities,
        type='all',
        dist=25)
verif.urb(alicante.clust)

main.cities <- alicante.cities[alicante.cities$population > 250000, ]
alicante.clust <- calcul.dist.cities(
        alicante.clust,
        main.cities,
        type='all',
        dist=25)
verif.urb(alicante.clust)
```

`write.boundary.shp` *Exporter les frontières au format shape.*

Description

Exporte un polygone fermé, défini dans un data.frame à deux colonnes, au format shape.

Usage

```
write.boundary.shp(boundary, file, country = file)
```

Arguments

`boundary`	data.frame. Typiquement un résultat de `make.boundary.dcw`.
`file`	character. Nom du fichier à créer.
`country`	character. Nom de la zone. Par défaut, prend le nom du fichier.

Note

Ce type de fichier peut être facilement importé dans un logiciel de cartographie.

See Also

`writePolyShape`.

Examples

```
## Not run:
data(alicante)
write.boundary.shp(alicante.bounds,'alicante-bounds','alicante')
## End(Not run)
```

`write.prev.shp` *Exporter des points au format shape.*

Description

Exporte les points contenus dans un data.frame ainsi que les données qui leur sont associées au format shape file.

Usage

```
write.prev.shp(x, file, coords = ~x+y, check=TRUE, lang='en')
```

Arguments

`x`	data.frame. Données à exporter, un point par ligne.
`file`	character. Nom du fichier à créer.
`coords`	formula. Variables définissant la longitude et la latitude des points, exprimées en tant que formule.
`check`	logical. Si vrai, alors la fonction `check.names` sera appliquée à `x` avant l'exportation, afin de renommer les noms de colonnes de plus de 10 caractères.
`lang`	character. Permet de choisir la langue des messages utilisateur. `fr` pour le français, `en` pour l'anglais.

Note

Ce type de fichier peut être facilement importé dans un logiciel de cartographie.

Le contenu du tableau de données est exporté au format *dbf*. Or ce type de format n'accepte pas les noms de variables de plus de 10 caractères. Au moment de l'export, les noms de colonnes de plus de 10 caractères seront donc tronqués, ce qui peut poser problème lorsque les 10 premiers caractères de deux noms de colonnes sont identiques. Si `check = TRUE`, vous serez invité à renommer les noms de variable de plus de 10 caractères.

See Also

`writePointsShape`, `check.names`.

Examples

```
## Not run:
data(alicante)
write.prev.shp(alicante.prev,'alicante-prev', lang='fr')
## End(Not run)
```

`write.txt` *Exporter un data.frame au format texte tabulé.*

Description

Permet d'exporter un data.frame au format texte, les valeurs étant séparées par des tabulations.

Usage

```
write.txt(x, file, dec = ".")
```

Arguments

`x`	data.frame. Données à exporter.
`file`	character. Nom du fichier à créer.
`dec`	character. Charactère de séparation des décimales.

Details

Est équivalent à
`write.table(x, file = file, sep="\t", row.names = FALSE, quote = FALSE, dec = dec)`.

See Also

`write.table` pour plus d'options d'export.

Examples

```
## Not run:
data(alicante)

write.txt(alicante.clust,'alicante-clusters.txt')

#Pour importer dans Excel version française :
write.txt(alicante.clust,'alicante-clusters.txt', dec=',')
## End(Not run)
```

Index