Package: serad (via r-universe)

October 9, 2026

Title Standardized Economic Reporting and Automated Dynamic Writing / Synthèse d'Écrits Avec des Règles Automatisées et Dynamiques

Version 0.2.6

Description Provides tools for generating dynamic and standardized economic narratives in R Markdown documents. The package is primarily designed for French-language statistical and economic publications. It includes functions to describe changes in levels, percentages, trends, accelerations and short-term economic developments using consistent linguistic rules. The package supports automated reporting workflows and reproducible economic writing. Fournit des outils permettant de générer des textes économiques dynamiques et standardisés dans des documents R Markdown. Le package est principalement conçu pour les publications statistiques et économiques en français. Il propose des fonctions permettant de décrire les évolutions de niveaux, de pourcentages, de tendances, d'accélérations et les évolutions conjoncturelles à l'aide de règles linguistiques homogènes. Le package facilite l'automatisation de la rédaction et la reproductibilité des publications économiques.

License MIT + file LICENSE

Encoding UTF-8

RoxygenNote 7.3.3

Imports tibble

Suggests knitr, rmarkdown, testthat (>= 3.0.0)

VignetteBuilder knitr

Config/testthat/edition 3

NeedsCompilation no

Author Alexandre Cazenave-Lacroutz [aut] (ORCID: <https://orcid.org/0000-0003-3488-6092>), Jules Lejas [cre], Direction de l'animation de la recherche, des études et des statistiques (Dares) [cph]

Maintainer Jules Lejas <jules.lejas@gmail.com>

Repository https://julesljs.r-universe.dev

Date/Publication 2026-10-07 16:42:52 UTC

RemoteUrl https://github.com/cran/serad

RemoteRef HEAD

RemoteSha 9895bbbac80886c14e08758178ecc1bf196a422f

Contents

a . . . 2
arrondi_tot . . . 3
comparaison . . . 4
comparaison_taux . . . 6
format_niv . . . 8
format_pts . . . 9
format_taux . . . 10
g . . . 11
g_nom . . . 12
g_nom_evo . . . 13
g_verbe . . . 15
g_verbe_evo . . . 16
get_serad_language . . . 18
gETa_nom . . . 19
gETa_nom_taux . . . 20
gETa_verbe . . . 21
gETa_verbe_taux . . . 22
init_serad . . . 24
libelle_periode . . . 25
pluriel . . . 26
set_serad_language . . . 27

Index **28**

`a` *Calcul d'une accélération*

Description

Calcule l'accélération entre trois niveaux successifs en comparant les taux de variation consécutifs.

Usage

```
a(x1, x2, x3)
```

Arguments

x1	Niveau le plus récent.
x2	Niveau précédent.
x3	Niveau le plus ancien.

Details

L'accélération correspond à la variation du taux entre (x1, x2) et (x2, x3).

Value

Un nombre numérique correspondant à l'accélération en pourcentage.

See Also

g, format_taux

Examples

```
a(4, 2, 1)  # 0
a(6, 2, 1)  # 100
a(2, 1, 1)  # valeur très élevée si taux précédent proche de zéro
```

---

arrondi_tot *Arrondi arithmétique*

Description

Arrondit un nombre selon la règle arithmétique (0.5 vers le haut), contrairement à round() qui utilise l'arrondi bancaire.

Usage

```
arrondi_tot(x, digits = 1)
```

Arguments

x	Nombre à arrondir.
digits	Entier indiquant le nombre de décimales. Positif pour les décimales, négatif pour les dizaines, centaines, etc. Par défaut : 1.

Value

Un nombre numérique arrondi.

See Also

format_niv

Examples

```
arrondi_tot(1877.85, digits = 0)   # 1878
arrondi_tot(1877.85, digits = 1)   # 1877.9
arrondi_tot(1877.85, digits = -1)  # 1880
```

---

comparaison *Comparaison qualitative entre deux niveaux*

Description

Compare deux niveaux successifs et retourne une formulation selon l'évolution observée.

Usage

```
comparaison(
  x1,
  x2,
  hausse_defaut,
  egalite_defaut,
  baisse_defaut,
  seuil = 0.1,
  alt = 0,
  hausse_alt = hausse_defaut,
  egalite_alt = egalite_defaut,
  baisse_alt = baisse_defaut
)
```

Arguments

x1	Niveau le plus récent.
x2	Niveau le plus ancien.
hausse_defaut	Formulation en cas de hausse.
egalite_defaut	Formulation en cas de stabilité.
baisse_defaut	Formulation en cas de baisse.
seuil	Seuil d'égalité en valeur absolue. Par défaut : 0.1.
alt	Indicateur logique permettant d'utiliser une formulation alternative.
hausse_alt	Formulation alternative en cas de hausse.
egalite_alt	Formulation alternative en cas de stabilité.
baisse_alt	Formulation alternative en cas de baisse.

Details

La comparaison repose sur le taux de variation calculé via g. Des cas particuliers sont traités lorsque x2 est nul ou négatif.

Les exemples ci-dessous proposent trois formulations à copier-coller et à adapter selon le contexte. Ces fonctions ne font pas partie du package.

Value

Une chaîne de caractères correspondant à la formulation retenue.

See Also

comparaison_taux, g

Examples

```
comparaison(1.04, 1, "augmente", "reste stable", "diminue")
comparaison(0.9991, 1, "augmente", "reste stable", "diminue")
comparaison(1, 1, "augmente", "reste égal", "diminue", seuil = 0)

# Situer un niveau par rapport à un autre :
# « le niveau est au-dessus de celui de l'année précédente ».
formuler_position <- function(x1, x2, lang = get_serad_language()) {
  if (lang == "en") {
    comparaison(
      x1, x2,
      hausse_defaut = "above",
      egalite_defaut = "at the same level as",
      baisse_defaut = "below",
      seuil = 0
    )
  } else {
    comparaison(
      x1, x2,
      hausse_defaut = "au-dessus de",
      egalite_defaut = "au même niveau que",
      baisse_defaut = "en dessous de",
      seuil = 0
    )
  }
}

formuler_position(104, 100)

# Qualifier une tendance :
# « la tendance est à la hausse ».
formuler_tendance <- function(
    x1, x2, seuil = 0.1, lang = get_serad_language()
) {
  if (lang == "en") {
    comparaison(
```

```
      x1, x2,
      hausse_defaut = "upward",
      egalite_defaut = "stable",
      baisse_defaut = "downward",
      seuil = seuil
    )
  } else {
    comparaison(
      x1, x2,
      hausse_defaut = "à la hausse",
      egalite_defaut = "stable",
      baisse_defaut = "à la baisse",
      seuil = seuil
    )
  }
}

formuler_tendance(1.04, 1)

# Comparer des nombres d'éléments :
# « il y en a davantage ».
formuler_quantite <- function(x1, x2, lang = get_serad_language()) {
  if (lang == "en") {
    comparaison(
      x1, x2,
      hausse_defaut = "more",
      egalite_defaut = "as many",
      baisse_defaut = "fewer",
      seuil = 0
    )
  } else {
    comparaison(
      x1, x2,
      hausse_defaut = "davantage",
      egalite_defaut = "autant",
      baisse_defaut = "moins",
      seuil = 0
    )
  }
}

formuler_quantite(104, 100)
```

`comparaison_taux` *Comparaison d'une variation à un seuil*

Description

Analyse un taux et retourne une formulation selon sa valeur.

Usage

```
comparaison_taux(
  g,
  hausse_defaut,
  egalite_defaut,
  baisse_defaut,
  seuil = 0.1,
  alt = 0,
  hausse_alt = hausse_defaut,
  egalite_alt = egalite_defaut,
  baisse_alt = baisse_defaut
)
```

Arguments

g	Variation exprimée en pourcentage (5 signifie 5 %, 0.1 signifie 0.1 %).
hausse_defaut	Mot si hausse (forme par défaut).
egalite_defaut	Mot si égalité (forme par défaut).
baisse_defaut	Mot si baisse (forme par défaut).
seuil	Limite pour l'égalité (0.1 par défaut).
alt	Paramètre supplémentaire égal à 0 ou 1 (par exemple pour distinguer singulier/pluriel).
hausse_alt	Formulation alternative si alt = 1.
egalite_alt	Formulation alternative si alt = 1.
baisse_alt	Formulation alternative si alt = 1.

Details

Fonction interne utilisée par comparaison.

Value

Une chaîne de caractères correspondant à la modalité choisie.

See Also

comparaison

Examples

```
comparaison_taux(5, "augmente", "reste stable", "diminue")
comparaison_taux(0.05, "augmente", "reste stable", "diminue")
comparaison_taux(0, "as", "bs", "cs", seuil = 0, alt = 1,
                 hausse_alt = "a", egalite_alt = "b", baisse_alt = "c")
```

format_niv *Formatage des niveaux en nombres*

Description

Formate un niveau numérique selon les règles d'arrondi définies dans le package. Peut afficher explicitement le signe d'une variation en niveau.

Usage

```
format_niv(
  y,
  signe = FALSE,
  detail = getOption("serad")$arrondi_niv,
  lang = get_serad_language()
)
```

Arguments

y	Le niveau ou la variation à formater.
signe	Indicateur logique : TRUE pour afficher le signe plus des valeurs positives, FALSE sinon (par défaut).
detail	Précision d'arrondi. Par défaut, on utilise getOption("serad")$arrondi_niv.
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères correspondant à la valeur formatée.

See Also

arrondi_tot

Examples

```
format_niv(365484)                        # "365 500"
format_niv(365484, lang = "en")           # "365,500"
format_niv(365484 - 300000, signe = TRUE) # "+65 500"
format_niv(300000 - 365484, signe = TRUE) # "-65 500"
```

format_pts *Formatage des variations en points*

Description

Formate une variation exprimée en points selon les règles d'arrondi et d'affichage du package.

Usage

```
format_pts(
  y,
  signe = TRUE,
  detail = getOption("serad")$arrondi_pourcent,
  abrev = FALSE,
  lang = get_serad_language()
)
```

Arguments

y	La variation à formater.
signe	Indicateur logique : TRUE pour afficher le signe, FALSE pour le retirer (par défaut : TRUE).
detail	Nombre de chiffres après la virgule. Par défaut, on utilise getOption("serad")$arrondi_pourcent.
abrev	Indicateur logique : TRUE pour utiliser l'abréviation ("pt"/"pts"), FALSE pour afficher "point(s)" (par défaut : FALSE).
lang	Langue de sortie : "fr" ou "en".

Details

Le symbole "moins" peut être personnalisé via getOption("serad")$moins.

Value

Une chaîne de caractères correspondant à la variation formatée (ex. "+5,4 points", "+5.4 points").

See Also

format_taux arrondi_tot

Examples

```
format_pts(5.3654, signe = FALSE)   # "5,4 points"
format_pts(1.3654, signe = FALSE)   # "1,4 point"
format_pts(5.3654, abrev = TRUE)    # "+5,4 pts"
format_pts(-5.3654, FALSE)          # "5,4 points"
format_pts(-5.3654)                 # "-5,4 points"
format_pts(-5.3654, detail = 2)     # "-5,37 points"
```

```
format_pts(0.35)                    # "+0,4 points"
```

`format_taux` *Formatage des variations en pourcentage*

Description

Formate une variation exprimée en pourcentage selon les règles d'arrondi et d'affichage du package.

Usage

```
format_taux(
  y,
  signe = TRUE,
  detail = getOption("serad")$arrondi_pourcent,
  lang = get_serad_language()
)
```

Arguments

`y`	La variation à formater.
`signe`	Indicateur logique : TRUE pour afficher le signe, FALSE pour le retirer (par défaut : TRUE).
`detail`	Nombre de chiffres après la virgule. Par défaut, on utilise getOption("serad")$arrondi_pourcent.
`lang`	Langue de sortie : "fr" ou "en".

Details

Le symbole "moins" peut être personnalisé via getOption("serad")$moins.

Value

Une chaîne de caractères correspondant à la variation formatée (ex. "+5,4%", "-5,4%", "+5.4%").

See Also

g, arrondi_tot

Examples

```
format_taux(5.3654, signe = FALSE)          # "5,4 %"
format_taux(5.3654)                         # "+5,4 %"
format_taux(-5.3654, FALSE)                 # "5,4 %"
format_taux(-5.3654)                        # "-5,4 %"
format_taux(-5.3654, detail = 2)            # "-5,37 %"
format_taux(0.35)                           # "+0,4 %"
format_taux(5.3654, lang = "en")            # "+5.4%"
```

g *Calcul d'une variation relative*

Description

Calcule la variation relative entre `x1` et `x2`, exprimée en pourcentage.

Usage

```
g(x1, x2, eps = 1e-08)
```

Arguments

`x1`	Niveau le plus récent.
`x2`	Niveau le plus ancien.
`eps`	Valeur utilisée à la place de `x2` lorsque `x2 = 0`, afin d'éviter une division par zéro. Par défaut : `1e-8`.

Details

Si `x2 = 0`, la valeur epsilon définie par `getOption("serad")$eps` est utilisée afin d'éviter une division par zéro. Un message d'avertissement est émis dans ce cas.

Il est possible de modifier cette valeur :

```
serad0 <- getOption("serad") serad0$eps <- 0 options(serad = serad0)
```

Value

Une valeur numérique correspondant à la variation en pourcentage (par exemple, si x1 = 2 * x2, la fonction retourne 100).

See Also

`format_taux`

Examples

```
g(2, 1)  # 100
g(2, 0)  # valeur très élevée et avertissement
```

g_nom *Évolution nominale non suivie d'une valeur*

Description

Décrit une évolution sous forme nominale à partir de deux niveaux, sans ajouter la valeur de variation.

Usage

```
g_nom(
  x1,
  x2,
  evolution = c("pourcents", "points"),
  avec_evolution = FALSE,
  titre = FALSE,
  lang = get_serad_language()
)
```

Arguments

x1	Le niveau le plus récent.
x2	Le niveau le plus ancien.
evolution	Type d'évolution : "pourcents" (variation relative, par défaut) ou "points".
avec_evolution	Indicateur logique : TRUE pour ajouter la valeur de l'évolution à la formulation nominale, FALSE pour retourner uniquement la formulation.
titre	Indicateur logique : TRUE pour supprimer l'article initial et mettre une majuscule, notamment en début de titre.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord une évolution à partir de x1 et x2 :

- si evolution = "pourcents", elle utilise g ;
- si evolution = "points", elle calcule x1 - x2.

La valeur obtenue est ensuite transmise à g_nom_evo, qui détermine la formulation à partir des conditions définies dans la table getOption("serad")$evo_simple.

Value

Une chaîne de caractères correspondant à la formulation nominale retenue (par exemple : "une forte hausse").

Personnalisation

Les formulations utilisées par cette fonction proviennent de la table `getOption("serad")$evo_simple`.

Cette table doit notamment contenir une colonne `condition` et une colonne `nom`. Les conditions doivent être disjointes, c'est-à-dire qu'une seule condition doit être vraie pour une valeur donnée.

Pour modifier les conditions ou les libellés, voir `init_serad`.

See Also

`g_nom_evo`, `g`, `init_serad`

Examples

```
g_nom(1.04, 1)
g_nom(1.01, 1)
g_nom(1.004, 1)
g_nom(1.001, 1)
g_nom(1, 1)
g_nom(0.997, 1)
g_nom(0.95, 1)
g_nom(0.95, 1, evolution = "points")
```

---

g_nom_evo *Évolution nominale*

Description

Décrit une évolution sous forme nominale, avec ou sans la valeur de l'évolution.

Usage

```
g_nom_evo(
  g,
  evolution = c("pourcents", "points"),
  avec_evolution = FALSE,
  titre = FALSE,
  lang = get_serad_language()
)
```

Arguments

`g`	Valeur de l'évolution.
`evolution`	Type d'évolution : `"pourcents"` (par défaut) ou `"points"`.
`avec_evolution`	Indicateur logique : `TRUE` pour ajouter la valeur de l'évolution à la formulation nominale, `FALSE` pour retourner uniquement la formulation.
`titre`	Indicateur logique : `TRUE` pour supprimer l'article initial et mettre une majuscule, notamment en début de titre.
`lang`	Langue de sortie : `"fr"` ou `"en"`.

Details

La fonction sélectionne, dans la table `getOption("serad")$evo_simple`, la ligne dont la condition est vérifiée par la valeur de g.

La table `evo_simple` doit contenir une colonne `condition`, composée de chaînes de caractères évaluables par R, par exemple : `"g >= -0.10 & g <= 0.10"`.

Les conditions doivent être disjointes : pour une valeur donnée de g, une seule condition doit être vraie.

La fonction renvoie ensuite la colonne `nom` correspondante.

Si `titre = TRUE`, l'article initial est supprimé et la première lettre restante est mise en majuscule.

Si `avec_evolution = TRUE`, la valeur absolue de g est ajoutée à la formulation avec l'unité correspondant à `evolution`. La valeur absolue est utilisée car le sens de l'évolution est déjà exprimé par le nom : par exemple, `"une baisse de 4 %"` et non `"une baisse de -4 %"`.

Value

Une chaîne de caractères décrivant l'évolution, avec ou sans sa valeur (par exemple : `"une forte hausse"`, `"une forte hausse de 4 %"` ou `"une forte hausse de 2 points"`).

Personnalisation

Les formulations utilisées par cette fonction proviennent de la table `getOption("serad")$evo_simple`.

Pour modifier les conditions ou les libellés, voir `init_serad`.

See Also

`g_nom`, `g_verbe_evo`, `pluriel`, `init_serad`

Examples

```
g_nom_evo(4)
g_nom_evo(4, avec_evolution = TRUE)

g_nom_evo(1.5, evolution = "points")
g_nom_evo(1.5, evolution = "points", avec_evolution = TRUE)
g_nom_evo(-2, evolution = "points", avec_evolution = TRUE)

g_nom_evo(4, avec_evolution = TRUE, titre = TRUE)
g_nom_evo(4, avec_evolution = TRUE, lang = "en")
```

g_verbe *Évolution verbale*

Description

Décrit une évolution sous forme verbale à partir de deux niveaux, sans tenir compte d'une éventuelle accélération, avec ou sans la valeur de variation.

Usage

```
g_verbe(
  x1,
  x2,
  sing = TRUE,
  evolution = c("pourcents", "points"),
  avec_evolution = TRUE,
  stable_sans_valeur = TRUE,
  lang = get_serad_language()
)
```

Arguments

x1	Le niveau le plus récent.
x2	Le niveau le plus ancien.
sing	Indicateur logique : TRUE si le sujet est singulier (par défaut), FALSE sinon.
evolution	Type d'évolution : "pourcents" (variation relative, par défaut) ou "points".
avec_evolution	Indicateur logique : TRUE pour ajouter la valeur de l'évolution après le verbe. Par défaut, FALSE retourne uniquement le verbe.
stable_sans_valeur	Indicateur logique : TRUE (par défaut) pour ne pas ajouter la valeur lorsque la catégorie correspond à une stabilité. Si FALSE, la valeur est ajoutée après la formulation de stabilité, à condition que avec_evolution = TRUE.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord une évolution à partir de x1 et x2 :

- si evolution = "pourcents", elle utilise g ;
- si evolution = "points", elle calcule x1 - x2.

La valeur obtenue est ensuite transmise à g_verbe_evo, qui détermine la formulation à partir des conditions définies dans la table getOption("serad")$evo_simple.

Cette table doit notamment contenir les colonnes condition, verbe_sing et verbe_plur.

Les conditions doivent être disjointes : pour une valeur donnée, une seule condition doit être vraie.

Value

Une chaîne de caractères correspondant à la formulation verbale retenue, avec ou sans la valeur de l'évolution, par exemple : `"augmente"` ou `"augmente de 10,0 %"`.

Personnalisation

Les formulations utilisées par cette fonction proviennent de la table `getOption("serad")$evo_simple`.

Pour modifier les conditions ou les libellés, voir `init_serad`.

See Also

`g_verbe_evo`, `g`, `init_serad`

Examples

```
g_verbe(1.1, 1)
g_verbe(1.1, 1, avec_evolution = TRUE)

g_verbe(1.04, 1)
g_verbe(1.01, 1, sing = FALSE)

g_verbe(0.999, 1)
g_verbe(
  0.999,
  1,
  avec_evolution = TRUE,
  stable_sans_valeur = FALSE
)

g_verbe(0.96, 1)
g_verbe(0.79, 1)

g_verbe(
  12,
  10,
  evolution = "points",
  avec_evolution = TRUE
)
```

---

g_verbe_evo *Évolution verbale d'un taux*

Description

Décrit une évolution sous forme verbale, sans tenir compte d'une éventuelle accélération, avec ou sans la valeur formatée.

Usage

```
g_verbe_evo(
  g,
  sing = TRUE,
  evolution = c("pourcents", "points"),
  avec_evolution = TRUE,
  stable_sans_valeur = TRUE,
  lang = get_serad_language()
)
```

Arguments

g	Valeur de l'évolution.
sing	Indicateur logique : TRUE si le sujet est singulier (par défaut), FALSE sinon.
evolution	Type d'évolution : "pourcents" (variation relative, par défaut) ou "points".
avec_evolution	Indicateur logique : TRUE pour ajouter la valeur de l'évolution après le verbe, FALSE pour retourner uniquement le verbe.
stable_sans_valeur	Indicateur logique : TRUE (par défaut) pour ne pas ajouter la valeur lorsque la catégorie correspond à une stabilité. Si FALSE, la valeur est ajoutée après la formulation de stabilité, à condition que avec_evolution = TRUE.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction sélectionne, dans la table getOption("serad")$evo_simple, la ligne dont la condition est vérifiée par la valeur de g.

La table evo_simple doit contenir une colonne condition, composée de chaînes de caractères évaluables par R, par exemple : "g >= -0.10 & g <= 0.10".

Les conditions doivent être disjointes : pour une valeur donnée de g, une seule condition doit être vraie.

Si sing = TRUE, la fonction utilise la colonne verbe_sing. Sinon, elle utilise la colonne verbe_plur.

Si avec_evolution = FALSE, la préposition placée à la fin de la formulation verbale ("de", "à", "by" ou "at") est supprimée.

Si avec_evolution = TRUE, la valeur est formatée avec format_taux lorsque evolution = "pourcents" et avec format_pts lorsque evolution = "points".

Lorsqu'une évolution appartient à la catégorie de stabilité et que stable_sans_valeur = TRUE, seule la formulation verbale est retournée.

Value

Une chaîne de caractères décrivant l'évolution, avec ou sans sa valeur.

Personnalisation

Les formulations utilisées par cette fonction proviennent de la table `getOption("serad")$evo_simple`.

Pour modifier les conditions ou les libellés, voir `init_serad`.

See Also

`g_verbe`, `g_nom_evo`, `format_taux`, `format_pts`, `init_serad`

Examples

```
g_verbe_evo(10)
g_verbe_evo(10, avec_evolution = FALSE)

g_verbe_evo(2, evolution = "points")
g_verbe_evo(2, evolution = "points", avec_evolution = FALSE)

g_verbe_evo(0.1)
g_verbe_evo(-0.1)
g_verbe_evo(-0.1, stable_sans_valeur = FALSE)

g_verbe_evo(10, sing = FALSE)
g_verbe_evo(10, lang = "en")
```

`get_serad_language` *Récupère la langue de serad*

Description

Retourne la langue actuellement utilisée par {serad}.

Usage

```
get_serad_language()
```

Value

Une chaîne de caractères indiquant la langue courante ("fr" ou "en").

gETa_nom *Évolution nominale tenant compte de l'accélération*

Description

Décrit l'évolution sous forme nominale à partir de trois niveaux, en tenant compte de l'accélération entre deux variations successives.

Usage

```
gETa_nom(x1, x2, x3, titre = FALSE, alea = 0, lang = get_serad_language())
```

Arguments

x1	Niveau le plus récent.
x2	Niveau précédent.
x3	Niveau le plus ancien.
titre	Indicateur logique : TRUE pour supprimer l'article initial et mettre une majuscule, notamment en début de titre.
alea	Paramètre numérique compris entre 0 et 1 contrôlant l'utilisation de formulations alternatives. Si alea = 0, la formulation est déterministe. Si alea = 1, la formulation alternative est toujours utilisée. Des valeurs intermédiaires permettent un tirage aléatoire.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord deux évolutions successives : g1 <- serad::g(x1, x2) et g2 <- serad::g(x2, x3).

Ces deux variations sont ensuite transmises à gETa_nom_taux, qui calcule leur accélération à l'aide de a et détermine la formulation nominale appropriée à partir de la table getOption("serad")$evo_accel.

Une formulation alternative peut être utilisée via la table getOption("serad")$evo_accel_alt. Le choix entre la formulation principale et la variante dépend du paramètre alea.

Value

Une chaîne de caractères correspondant à la formulation nominale retenue.

Personnalisation

Les formulations utilisées par cette fonction proviennent des tables getOption("serad")$evo_accel et getOption("serad")$evo_accel_alt.

Pour modifier les seuils, les conditions ou les libellés, voir init_serad.

See Also

gETa_nom_taux, g, a, init_serad

Examples

```
gETa_nom(1.00049, 1, 0.9996)
gETa_nom(1.1, 1, 0.99)
gETa_nom(1.003, 1, 0.99)
gETa_nom(0.96, 1, 1.01)
gETa_nom(0.8, 1, 0.99)
gETa_nom(1.1, 1, 0.99, alea = 0.5)
```

---

gETa_nom_taux *Évolution nominale tenant compte de l'accélération*

Description

Décrit l'évolution sous forme nominale en tenant compte de l'accélération entre deux variations successives.

Usage

```
gETa_nom_taux(g1, g2, titre = FALSE, alea = 0, lang = get_serad_language())
```

Arguments

g1	Dernière évolution, exprimée en pourcentage.
g2	Évolution précédente, exprimée en pourcentage.
titre	Indicateur logique : TRUE pour supprimer l'article initial et mettre une majuscule, notamment en début de titre.
alea	Paramètre numérique compris entre 0 et 1 contrôlant l'utilisation de formulations alternatives. Si alea = 0, la formulation est déterministe. Si alea = 1, la formulation alternative est toujours utilisée. Des valeurs intermédiaires permettent un tirage aléatoire.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord l'accélération entre g1 et g2 à l'aide de a.

Elle parcourt ensuite la table getOption("serad")$evo_accel ligne par ligne. Chaque ligne contient un ensemble de conditions sur g1, g2 et l'accélération. La première ligne dont toutes les conditions sont vérifiées détermine la formulation retenue.

Une formulation alternative peut être utilisée via la table getOption("serad")$evo_accel_alt, qui contient une variante pour chaque ligne de evo_accel. Le choix entre la formulation principale et la variante dépend du paramètre alea.

Si titre = TRUE, l'article initial est supprimé et la première lettre restante est mise en majuscule.

Value

Une chaîne de caractères correspondant à la formulation nominale retenue (par exemple : "une accélération", "une stabilisation").

Personnalisation

Les formulations utilisées par cette fonction proviennent des tables `getOption("serad")$evo_accel` et `getOption("serad")$evo_accel_alt`.

Pour modifier les seuils, les conditions ou les libellés, voir `init_serad`.

See Also

`a`, `gETa_verbe_taux`, `init_serad`

Examples

```
gETa_nom_taux(0.049, 0.049)
gETa_nom_taux(10, 1)
gETa_nom_taux(-4, 1, titre = TRUE)
gETa_nom_taux(-21, 1)
gETa_nom_taux(10, 1, alea = 0.5)
```

---

gETa_verbe *Évolution verbale tenant compte de l'accélération*

Description

Décrit l'évolution sous forme verbale à partir de trois niveaux, en tenant compte de l'accélération entre deux variations successives.

Usage

```
gETa_verbe(x1, x2, x3, sing = TRUE, alea = 0, lang = get_serad_language())
```

Arguments

`x1`	Niveau le plus récent.
`x2`	Niveau précédent.
`x3`	Niveau le plus ancien.
`sing`	Indicateur logique : TRUE si le sujet est singulier (par défaut), FALSE sinon.
`alea`	Paramètre numérique compris entre 0 et 1 contrôlant l'utilisation de formulations alternatives. Si `alea = 0`, la formulation est déterministe. Si `alea = 1`, la formulation alternative est toujours utilisée. Des valeurs intermédiaires permettent un tirage aléatoire.
`lang`	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord deux évolutions successives : `g1 <- serad::g(x1, x2)` et `g2 <- serad::g(x2, x3)`.

Ces deux variations sont ensuite transmises à gETa_verbe_taux, qui calcule leur accélération à l'aide de g et détermine la formulation verbale appropriée à partir de la table `getOption("serad")$evo_accel`.

Une formulation alternative peut être utilisée via la table `getOption("serad")$evo_accel_alt`. Le choix entre la formulation principale et la variante dépend du paramètre `alea`.

Value

Une chaîne de caractères correspondant à la formulation verbale retenue.

Personnalisation

Les formulations utilisées par cette fonction proviennent des tables `getOption("serad")$evo_accel` et `getOption("serad")$evo_accel_alt`.

Pour modifier les seuils, les conditions ou les libellés, voir init_serad.

See Also

gETa_verbe_taux, g, init_serad

Examples

```
gETa_verbe(1.00049, 1, 0.9996)
gETa_verbe(1.1, 1, 0.99)
gETa_verbe(1.003, 1, 0.99, sing = FALSE)
gETa_verbe(0.96, 1, 1.01)
gETa_verbe(1.1, 1, 0.99, alea = 0.5)
```

---

gETa_verbe_taux *Évolution verbale tenant compte de l'accélération*

Description

Décrit l'évolution sous forme verbale en tenant compte de l'accélération entre deux variations successives.

Usage

```
gETa_verbe_taux(g1, g2, sing = TRUE, alea = 0, lang = get_serad_language())
```

Arguments

g1	Dernière évolution, exprimée en pourcentage.
g2	Évolution précédente, exprimée en pourcentage.
sing	Indicateur logique : TRUE si le sujet est singulier (par défaut), FALSE sinon.
alea	Nombre réel compris entre 0 et 1 contrôlant l'utilisation de formulations alternatives. Si alea = 0, la formulation est déterministe. Si alea = 1, la formulation alternative est toujours utilisée. Des valeurs intermédiaires permettent un tirage aléatoire.
lang	Langue de sortie : "fr" ou "en".

Details

La fonction calcule d'abord l'accélération entre g1 et g2 à l'aide de a.

Elle parcourt ensuite la table getOption("serad")$evo_accel ligne par ligne. Chaque ligne contient un ensemble de conditions sur g1, g2 et l'accélération. La première ligne pour laquelle toutes les conditions sont vérifiées détermine la formulation retenue.

Une formulation alternative peut être utilisée via la table getOption("serad")$evo_accel_alt. Le choix entre la formulation principale et la variante dépend du paramètre alea.

Si sing = TRUE, la fonction renvoie la colonne verbe_sing. Sinon, elle renvoie verbe_plur.

Value

Une chaîne de caractères correspondant à la formulation verbale retenue (par exemple : "accélère", "se stabilise").

Personnalisation

Les formulations utilisées par cette fonction proviennent des tables getOption("serad")$evo_accel et getOption("serad")$evo_accel_alt.

Pour modifier les seuils, les conditions ou les libellés, voir init_serad.

See Also

a, gETa_nom_taux, init_serad

Examples

```
gETa_verbe_taux(0.049, 0.049)
gETa_verbe_taux(10, 1)
gETa_verbe_taux(4, 1, FALSE)
gETa_verbe_taux(-4, 1)
gETa_verbe_taux(-21, 1)
gETa_verbe_taux(10, 1, alea = 0.5)
```

`init_serad` *Initialisation des règles de rédaction de {serad}*

Description

Les fonctions `init_serad_fr()` et `init_serad_en()` appliquent les règles de rédaction par défaut de {serad}, respectivement en français et en anglais.

Elles initialisent notamment :

- les règles d'arrondi ;
- le symbole de signe négatif ;
- les seuils d'évolution ;
- les tables de correspondance entre évolutions et formulations ;
- les règles de gestion de l'accélération.

Usage

```
init_serad_en()

init_serad_fr()
```

Details

Les règles sont enregistrées dans l'option `serad` et peuvent être consultées avec `getOption("serad")`.

Les principales structures utilisées sont :

- `evo_simple` : utilisée pour les évolutions simples (`g_nom_evo`, `g_verbe_evo`) ;
- `seuil` : seuils utilisés pour les évolutions tenant compte de l'accélération ;
- `evo_accel` : formulations utilisées pour les évolutions tenant compte de l'accélération (`gETa_nom_taux`, `gETa_verbe_taux`) ;
- `evo_accel_alt` : variantes utilisées lorsque l'argument `alea` est activé.

Value

La liste des règles appliquées, renvoyée de manière invisible. Les fonctions enregistrent également cette liste dans l'option `serad`.

Personnalisation

Le package fournit deux scripts modèles commentés : `config_serad_fr.R` et `config_serad_en.R`.

Pour personnaliser les règles, copier le modèle de la langue souhaitée dans son projet :

```
file.copy(
  system.file("modeles", "config_serad_fr.R", package = "serad"),
  "config_serad_perso.R"
)
```

Cette copie est réalisée une seule fois. Pour utiliser le modèle anglais, remplacer `config_serad_fr.R` par `config_serad_en.R`.

Modifier ensuite les arrondis, les seuils, les conditions ou les formulations dans le fichier copié, puis le charger au début du programme :

```
library(serad)
source("config_serad_perso.R")
```

Le script applique directement les règles personnalisées avec `options(serad = serad0)`. Aucun appel supplémentaire à une fonction d'initialisation n'est nécessaire.

Le fichier personnalisé peut être conservé et versionné avec le programme de production.

Un appel ultérieur à `init_serad_fr()` ou `init_serad_en()` remplace les règles personnalisées par les règles par défaut de la langue choisie.

See Also

g_nom_evo, g_verbe_evo, gETa_nom_taux, gETa_verbe_taux

---

libelle_periode *Produit le libellé d'une période*

Description

Formate un mois ou un trimestre, éventuellement décalé dans le temps, en français ou en anglais.

Usage

```
libelle_periode(
  numero,
  annee,
  periode = c("mois", "trimestre"),
  decalage = 0L,
  format = c("lettres", "chiffres"),
  avec_annee = TRUE,
  lang = get_serad_language()
)
```

Arguments

numero	Numéro du mois (1 à 12) ou du trimestre (1 à 4).
annee	Année de la période.
periode	Type de période : "mois" ou "trimestre".
decalage	Nombre de périodes à ajouter. Une valeur positive avance dans le temps et une valeur négative permet de revenir en arrière.
format	Format du libellé : "lettres" ou "chiffres". Pour les mois, cet argument ne modifie pas le résultat.
avec_annee	Indicateur logique : TRUE pour afficher l'année, FALSE pour ne pas l'afficher.
lang	Langue de sortie : "fr" ou "en".

Value

Une chaîne de caractères représentant la période demandée.

Examples

```
libelle_periode(8, 2026, periode = "mois")
libelle_periode(8, 2026, periode = "mois", decalage = 23)
libelle_periode(4, 2026, periode = "trimestre")
libelle_periode(
  4,
  2026,
  periode = "trimestre",
  decalage = 1,
  format = "chiffres"
)
libelle_periode(
  1,
  2026,
  periode = "trimestre",
  avec_annee = FALSE,
  lang = "en"
)
```

---

pluriel *Accord au pluriel*

Description

Retourne une forme singulière ou plurielle selon une valeur numérique et la langue.

Usage

```
pluriel(x, sing = "", plur = "s", lang = get_serad_language())
```

Arguments

x	Valeur numérique.
sing	Forme utilisée au singulier. Par défaut : chaîne vide.
plur	Forme utilisée au pluriel. Par défaut : "s".
lang	Langue de sortie : "fr" ou "en".

Details

En français, le pluriel s'applique à partir de 2. En anglais, seul 1 (ou -1) est au singulier.

Value

Une chaîne de caractères correspondant à la forme correctement accordée. Renvoie NA_character_ lorsque x vaut NA.

Examples

```
pluriel(-7.5)
pluriel(-2)
pluriel(1.4, "chat parle", "chats parlent")
pluriel(-2, "chat parle", "chats parlent")
pluriel(1.97)
pluriel(1.5, "point", "points", lang = "fr")
pluriel(1.5, "point", "points", lang = "en")
```

---

set_serad_language *Définit la langue de serad*

Description

Modifie la langue utilisée par {serad} et réinitialise les règles de rédaction associées.

Usage

```
set_serad_language(lang = c("fr", "en"))
```

Arguments

lang	Langue de sortie : "fr" ou "en".

Value

NULL invisiblement.

Index

a, 2, *19–21*, *23*
arrondi_tot, 3, *8–10*

comparaison, 4, *7*
comparaison_taux, *5*, 6

format_niv, *4*, 8
format_pts, 9, *17, 18*
format_taux, *3*, *9*, 10, *11*, *17, 18*

g, *3*, *5*, *10*, 11, *12, 13*, *15, 16*, *20*, *22*
g_nom, 12, *14*
g_nom_evo, *12*, 13, *13*, *18*, *24, 25*
g_verbe, 15, *18*
g_verbe_evo, *14, 15*, 16, *16*, *24, 25*
get_serad_language, 18
gETa_nom, 19
gETa_nom_taux, *19*, 20, *20*, *23–25*
gETa_verbe, 21
gETa_verbe_taux, *21*, 22, *22*, *24, 25*

init_serad, *13, 14*, *16*, *18–23*, 24
init_serad_en *(*init_serad*)*, 24
init_serad_fr *(*init_serad*)*, 24

libelle_periode, 25

pluriel, *14*, 26

set_serad_language, 27