« Data frame » pour la séquence de paramètres d'OSQP

Data frame contenant une séquence de paramètres d'OSQP pour `tsbalancing()` spécifié avec l'argument `osqp_settings_df`. La librairie inclut deux *data frames* prédéfinis de séquences de paramètres d'OSQP qui sont présentés ci-dessous.

Data frame `default_osqp_sequence`

Séquence rapide et efficace de paramètres d'OSQP qui devrait permettre de résoudre avec précision la plupart des problèmes d'équilibrage de séries chronologiques. C'est la valeur par défaut de l'argument `osqp_settings_df` de `tsbalancing()`.

```
#>   max_iter    sigma  eps_abs  eps_rel eps_prim_inf eps_dual_inf polish scaling
#> 1    4,000 1.00e-09 1.00e-06 1.00e-06     1.00e-07     1.00e-07   TRUE       0
#> 2   10,000 1.00e-15 1.00e-12 1.00e-12     1.00e-13     1.00e-13   TRUE       0
#> 3   10,000 1.00e-15 1.00e-12 1.00e-12     1.00e-13     1.00e-13   TRUE       0
#> 4   10,000 2.22e-16 2.22e-16 2.22e-16     2.22e-16     2.22e-16   TRUE       0
#>   prior_scaling require_polished
#> 1          TRUE             TRUE
#> 2          TRUE             TRUE
#> 3         FALSE            FALSE
#> 4          TRUE            FALSE
```

Data frame `alternate_osqp_sequence`

Séquence alternative plus lente de paramètres d'OSQP qui pourrait aider à atteindre une plus grande précision, si nécessaire, en particulier lorsque combinée avec l'argument `full_sequence = TRUE`.

```
#>    max_iter    sigma  eps_abs  eps_rel eps_prim_inf eps_dual_inf polish scaling
#> 1    10,000 1.00e-15 1.00e-12 1.00e-12     1.00e-13     1.00e-13   TRUE       0
#> 2    10,000 1.00e-15 1.00e-12 1.00e-12     1.00e-13     1.00e-13   TRUE      10
#> 3    10,000 1.00e-12 1.00e-09 1.00e-09     1.00e-10     1.00e-10   TRUE       0
#> 4    10,000 1.00e-12 1.00e-09 1.00e-09     1.00e-10     1.00e-10   TRUE      10
#> 5    10,000 1.00e-09 1.00e-06 1.00e-06     1.00e-07     1.00e-07   TRUE       0
#> 6    10,000 1.00e-09 1.00e-06 1.00e-06     1.00e-07     1.00e-07   TRUE      10
#> 7    10,000 1.00e-06 1.00e-03 1.00e-03     1.00e-04     1.00e-04   TRUE       0
#> 8    10,000 1.00e-06 1.00e-03 1.00e-03     1.00e-04     1.00e-04   TRUE      10
#> 9    10,000 2.22e-16 2.22e-16 2.22e-16     2.22e-16     2.22e-16   TRUE       0
#> 10   10,000 2.22e-16 2.22e-16 2.22e-16     2.22e-16     2.22e-16   TRUE      10
#> 11   10,000 1.00e-15 1.00e-12 1.00e-12     1.00e-13     1.00e-13   TRUE       0
#> 12   10,000 1.00e-12 1.00e-09 1.00e-09     1.00e-10     1.00e-10   TRUE       0
#> 13   10,000 1.00e-09 1.00e-06 1.00e-06     1.00e-07     1.00e-07   TRUE       0
#> 14   10,000 1.00e-06 1.00e-03 1.00e-03     1.00e-04     1.00e-04   TRUE       0
#> 15   10,000 2.22e-16 2.22e-16 2.22e-16     2.22e-16     2.22e-16   TRUE       0
#>    prior_scaling require_polished
#> 1          FALSE             TRUE
#> 2          FALSE             TRUE
#> 3          FALSE             TRUE
#> 4          FALSE             TRUE
#> 5          FALSE             TRUE
```

```
#> 6          FALSE            TRUE
#> 7          FALSE            TRUE
#> 8          FALSE            TRUE
#> 9          FALSE            TRUE
#> 10         FALSE            TRUE
#> 11          TRUE            TRUE
#> 12          TRUE            TRUE
#> 13          TRUE            TRUE
#> 14          TRUE            TRUE
#> 15          TRUE            TRUE
```

Détails

À l'exception de `prior_scaling` et `require_polished`, toutes les colonnes du *data frame* doivent correspondre à un paramètre d'OSQP. Les valeurs par défaut d'OSQP sont utilisées pour tout paramètre non spécifié dans ce *data frame*. Visitez https://osqp.org/docs/interfaces/solver_settings.html pour connaître tous les paramètres d'OSQP disponibles. Notez que le paramètre d'OSQP `verbose` est en fait contrôlé par les arguments `quiet` et `display_level` de `tsbalancing()` (c'est à dire que la colonne `verbose` dans un *data frame pour la séquence de paramètres d'OSQP* serait ignorée).

Chaque enregistrement (ligne) d'un *data frame pour la séquence de paramètres d'OSQP* représente une tentative de résolution d'un problème d'équilibrage avec les paramètres d'OSQP correspondants. La séquence de résolution s'arrête dès qu'une solution valide est obtenue (une solution pour laquelle tous les écarts de contraintes sont inférieurs ou égaux à la tolérance spécifiée avec l'argument `validation_tol` de `tsbalancing()`) à moins que la colonne `require_polished = TRUE`, auquel cas une solution *raffinée* d'OSQP (`status_polish = 1`) serait également nécessaire pour arrêter la séquence. Les écarts de contraintes correspondent à $\max(0, l - Ax, Ax - u)$ avec des contraintes définies comme $l \leq Ax \leq u$. Dans le cas où une solution satisfaisante ne peut être obtenue après avoir parcouru toute la séquence, `tsbalancing()` renvoie la solution qui a généré le plus petit total d'écarts de contraintes parmi les solutions valides, le cas échéant, ou parmi toutes les solutions, dans le cas contraire. Notez que l'exécution de la séquence de résolution entière peut être *forcée* en spécifiant l'argument `full_sequence = TRUE` avec `tsbalancing()`. Les enregistrements avec la colonne `prior_scaling = TRUE` ont les données du problème mises à l'échelle (redimensionnées) avant la résolution avec OSQP, en utilisant la moyenne des valeurs libres (non contraignantes) du problème comme facteur d'échelle.

En plus de spécifier un *data frame pour la séquence de paramètres d'OSQP* personnalisé avec l'argument `osqp_settings_df`, on peut aussi spécifier `osqp_settings_df = NULL` ce qui résultera en une seule tentative de résolution avec les valeurs par défaut d'OSQP pour tous les paramètres et avec `prior_scaling = FALSE` et `require_polished = FALSE`. Il est cependant recommandé d'essayer d'abord les *data frames* `default_osqp_sequence` et `alternate_osqp_sequence`, avec `full_sequence = TRUE` si nécessaire, avant d'envisager d'autres alternatives.

Approche recommandée

Commencez par la séquence de résolution par défaut de `tsbalancing()` (`osqp_settings_df = default_osqp_sequence` et `full_sequence = FALSE`). Ensuite, si plus de précision est nécessaire, essayez avec :

1. `full_sequence = TRUE`
2. `osqp_settings_df = alternate_osqp_sequence`
3. `osqp_settings_df = alternate_osqp_sequence` et `full_sequence = TRUE`

En pratique, spécifier `full_sequence = TRUE` devrait suffire lorsque plus de précision est nécessaire (au détriment du temps d'exécution, évidemment). Ce n'est qu'en de rares occasions que vous devrez utiliser le *data frame* `alternate_osqp_sequence`, qui sera souvent encore plus coûteux en termes de temps d'exécution, en particulier lorsqu'il est combiné avec `full_sequence = TRUE`.

Principes fondateurs

Ce qui suit est un résumé des leçons apprises lors du développement de `tsbalancing()` et des expérimentations avec le solveur OSQP. Il s'agit des principes fondateurs qui ont conduit aux deux *data frames pour la séquence de paramètres d'OSQP* présentés précédemment. Notez que ces observations s'appliquent aux ***problèmes d'équilibrage de séries chronologiques*** tels que résolus par `tsbalancing()` et peuvent ne pas s'appliquer directement à d'autres types de *problèmes quadratiques*.

- Les options de préconditionnement des données disponibles dans OSQP (avec le paramètre `scaling`) ne sont pas suffisantes pour certains problèmes (mal échelonnés). Une mise à l'échelle externe (préalable) des données (`prior_scaling = TRUE`) est parfois nécessaire pour que OSQP converge à un rythme décent et génère des solutions suffisamment précises dans un nombre raisonnable d'itérations.
- La mise à l'échelle préalable des données (`prior_scaling = TRUE`) permet souvent de réduire le temps d'exécution (le nombre d'itérations nécessaires pour atteindre la précision spécifiée) et d'augmenter considérablement la probabilité d'obtenir des solutions *raffinées* (`status_polish = 1`).
- Les solutions *raffinées* sont toujours très précises, même lorsqu'une mise à l'échelle préalable des données est effectuée (c'est-à-dire que la solution dans l'échelle d'origine sera généralement encore *suffisamment précise*).
- Si les solutions *raffinées* avec mise à l'échelle préalable des données sont généralement plus précises que les solutions *non raffinées* sans mise à l'échelle préalable des données, les solutions les plus précises correspondent aux solutions *raffinées* sans mise à l'échelle préalable des données.
- Des paramètres `sigma` et de tolérance (`eps_*`) plus petits permettent d'obtenir des solutions plus précises, mais leur exécution est plus longue (plus d'itérations sont nécesssaires).
- Une précision suffisante est généralement obtenue après 10 000 itérations avec des petites valeurs pour les paramètres `sigma` et de tolérance (`eps_*`).
- Les valeurs par défaut d'OSQP pour `alpha` et les divers paramètres associés à ρ (`*rho*`) sont suffisants (ils fonctionnent bien). Réduire les paramètres `sigma` et de tolérance (`eps_*`) et effectuer une mise à l'échelle préalable des données est suffisant pour obtenir des solutions précises dans un nombre raisonnable d'itérations.
- Garder la même échelle entre les paramètres `sigma` et de tolérance (`eps_*`) que les valeurs par défaut d'OSQP fonctionne bien et est utilisé dans les deux *data frames pour la séquence de paramètres d'OSQP*, c'est à dire :
  - `eps_abs = eps_rel = 1000 * sigma`
  - `eps_prim_inf = eps_dual_inf = 100 * sigma`
  - (et par conséquent) `eps_abs = eps_rel = 10 * eps_prim_inf = 10 * eps_dual_inf`
- L'*epsilon de la machine* (`.Machine$double.eps`) pour les paramètres `sigma` et de tolérance (`eps_*`), qui force essentiellement le nombre maximum d'itérations, est utilisé en *dernier recours* dans les deux *data frames pour la séquence de paramètres d'OSQP*.

Résumé - séquence par défaut (*data frame* `default_osqp_sequence`)

- Orientée vers l'obtention de solutions à la fois rapides et précises.
- Essayer d'abord d'obtenir des solutions *raffinées* (rapides) avec mise à l'échelle préalable des données avant d'essayer sans mise à l'échelle préalable des données.
- Faire une dernière tentative avec mise à l'échelle préalable des données et l'*epsilon de la machine* pour les paramètres `sigma` et de tolérance (`eps_*`).

Résumé - séquence alternative (*data frame* `alternate_osqp_sequence`)

- Orientée vers l'obtention de solutions précises au détriment du temps d'exécution.
- Un peu similaire à une approche par « force brute » ou « essayer tout ».
- Les paramètres `sigma` et de tolérance (`eps_*`) sont d'abord petits et augmentent progressivement, avec l'*epsilon de la machine* comme dernière tentative.
- Des solutions *raffinées* sont requises pour chaque étape de la séquence (la meilleure solution *non raffinée* est renvoyée si aucune solution *raffinée* n'a pu être obtenue à la fin de la séquence).
- Maximum de 10 000 itérations pour chaque étape de la séquence.

- On essaie d'abord d'obtenir des solutions *raffinées* sans mise à l'échelle préalable des données (les solutions les plus précises), puis on essaie avec mise à l'échelle préalable des données.