tca_split

Synopsis

- This vignette shows the behavior of `tca()` and `tca_split()`.
- If tau is estimated from the data, either set `vars.mle = TRUE` or the chunk size must be big enough. Do both is the best practice because a big enough chunk size would minimize the overhead of parallel.
- Used appropriately, `tca_split()` and `tca()` returns highly correlated results `(cor > 0.99)`.

```r
library(TCA)
library(furrr)
#> Loading required package: future
```

Fit

- First we run a sequential `TCA::tca()` and one with `vars.mle = TRUE`

```r
set.seed(1234)
n_features <- 150
data <- test_data(50, n_features, 3, 2, 2, 0.03, verbose = FALSE)
```

```r
tca_seq <- tca(
  X = data$X, W = data$W, C1 = data$C1, C2 = data$C2,
  log_file = NULL, verbose = FALSE
)

tca_seq_mle <- tca(
  X = data$X, W = data$W, C1 = data$C1, C2 = data$C2,
  log_file = NULL, verbose = FALSE, vars.mle = TRUE,
  max_iters = 20
)
```

- Then we do the same for `tca_split()` under the following scenarios:
  - There are as many chunks as there are parallel workers, in this case 7.
  - There are as many chunks as there are features, in this case 150.
  - There are as many chunks as there are features, `vars.mle = TRUE`

```r
split_X_7 <- split_input(data$X, 7, shuffle = TRUE) # Split X into 7 chunks
split_X <- split_input(data$X, n_features) # Split X into as many chunks as feat
```

- Fit with `TCA::tca_split()`

```r
# Not actually ran to save time
plan(multisession, workers = 7)
# There are as many chunks as there are parallel workers
tca_par_7 <- tca_split(
  split_X_7, W = data$W, C1 = data$C1, C2 = data$C2,
  log_file_prefix = NULL, verbose = FALSE
)
# There are as many chunks as there are features
tca_par <- tca_split(
  split_X, W = data$W, C1 = data$C1, C2 = data$C2,
  log_file_prefix = NULL, verbose = FALSE
)
# There are as many chunks as there are features, `vars.mle = TRUE`
tca_par_mle <- tca_split(
  split_X, W = data$W, C1 = data$C1, C2 = data$C2,
  log_file_prefix = NULL, verbose = FALSE,
  vars.mle = TRUE, max_iters = 20
)
plan(sequential)
```

Results

- We see that for 7 chunks, the `tca_split()` and `tca()` fits are very correlated.

```
compare_fit_corr(tca_seq, tca_par_7) # tca_seq vs tca_split with 7 Chunks of X
#> $mus_hat
#> [1] 0.9999 1.0000 0.9999
#>
#> $sigmas_hat
#> [1] 0.9394 0.9777 0.9809
#>
#> $deltas_hat
#> [1] 1 1
#>
#> $gammas_hat
#> [1] 0.9988 0.9988 0.9997 0.9992 0.9992 0.9990
#>
#> $deltas_hat_pvals
#> [1] 0.9951 0.9992
#>
#> $gammas_hat_pvals
#> [1] 0.9886 0.9850 0.9955 0.9868 0.9895 0.9901
#>
#> $gammas_hat_pvals.joint
#> [1] 0.9966 0.9913
```

- However, for as many chunks as there are features, the correlation drops significantly. Especially for `sigmas_hat` and `gammas_hat_pvals`.

```
# tca_seq vs tca_split with as many chunks of X as there is features
compare_fit_corr(tca_seq, tca_par)
#> $mus_hat
#> [1] 0.9997 0.9998 0.9998
#>
#> $sigmas_hat
#> [1] 0.8935 0.9300 0.9211
#>
#> $deltas_hat
#> [1] 0.9999 0.9999
#>
```

```
#> $gammas_hat
#> [1] 0.9967 0.9965 0.9992 0.9978 0.9977 0.9968
#>
#> $deltas_hat_pvals
#> [1] 0.9874 0.9966
#>
#> $gammas_hat_pvals
#> [1] 0.9516 0.9575 0.9874 0.9570 0.9605 0.9695
#>
#> $gammas_hat_pvals.joint
#> [1] 0.9850 0.9676
```

- `tau_hat` are close enough.

```
unname(tca_seq$tau_hat)
#> [1] 0.04017212
mean(tca_par_7$tau_hat)
#> [1] 0.03717156
mean(tca_par$tau_hat)
#> [1] 0.03310071
```

- For the as many chunks as there are features `vars.mle = TRUE`, the correlation stays high.

```
# tca_seq vs tca_split with as many chunks of X as there is features, vars.mle = TRUE
compare_fit_corr(tca_seq_mle, tca_par_mle)
#> $mus_hat
#> [1] 0.9999 0.9997 0.9999
#>
#> $sigmas_hat
#> [1] 0.9941 0.9892 0.9943
#>
#> $deltas_hat
#> [1] 0.9999 0.9999
#>
#> $gammas_hat
#> [1] 0.9975 0.9984 0.9987 0.9985 0.9975 0.9984
#>
#> $deltas_hat_pvals
#> [1] 0.9985 0.9984
#>
#> $gammas_hat_pvals
#> [1] 0.9807 0.9644 0.9661 0.9509 0.9682 0.9720
#>
#> $gammas_hat_pvals.joint
#> [1] 0.9989 0.9971
```