Package ‘PanelIFE’

December 2, 2024

Title Panels with Interactive Fixed Effects

Version 0.0

Date 2024-11-01

Author Timothy B. Armstrong [aut],
Chen-Wei Hsiang [aut, cre],
Martin Weidner [aut],
Andrei Zeleneev [aut]

Maintainer Chen-Wei Hsiang <chen-wei.hsiang.21@ucl.ac.uk>

Description
Provides tools for estimation and inference for a regression coefficient in panel data with interactive fixed effects (i.e., with a factor structure), and either with strong factors or weak factors.

License MIT + file LICENSE

Encoding UTF-8

Imports pracma,
RcppDE,
stats

Roxygen list(markdown = TRUE)

RoxygenNote 7.3.2

Depends R (>= 2.10)

LazyData true

Contents

`empirical_data` *Empirical Dataset (Divorce Rate in the USA from 1956 to 1988)*

Description

Data of the divorce rate in the USA from 1956 to 1988, originally from Friedberg (1998, "Did Unilateral Divorce Raise Divorce Rates? Evidence from Panel Data") and Wolfers (2006, "Did Unilateral Divorce Laws Raise Divorce Rates? A Reconciliation and New Results"). Here, this empirical dataset follows Kim and Oka (2014, "Divorce Law Reforms and Divorce Rates in the USA: An Interactive Fixed-Effects Approach") and use their data to construct a balanced panel with `N = 48` states and `T = 33` years.

Usage

```
empirical_data
```

Format

A `data.frame` with 1,650 rows and 88 variables:

st Two-letter state code

id_st Numeric ID for the states

year Year of observation

div_rate_rev01 Divorces per 1000 people, 1956-1998

div_rate_rev02 Divorces per 1000 people, 1956-1998

stpop State population

unilateral Dummy for unilateral law in Friedberg (1998)

dyn_uni2 Dynamic treatment effects (years 1-2): dummy for year of unilateral law in Friedberg (1998)

dyn_uni3 Dynamic treatment effects (years 3-4): dummy for year of unilateral law in Friedberg (1998)

dyn_uni4 Dynamic treatment effects (years 5-6): dummy for year of unilateral law in Friedberg (1998)

dyn_uni5 Dynamic treatment effects (years 7-8): dummy for year of unilateral law in Friedberg (1998)

dyn_uni6 Dynamic treatment effects (years 9-10): dummy for year of unilateral law in Friedberg (1998)

dyn_uni7 Dynamic treatment effects (years 11-12): dummy for year of unilateral law in Friedberg (1998)

dyn_uni8 Dynamic treatment effects (years 13-14): dummy for year of unilateral law in Friedberg (1998)

dyn_uni9 Dynamic treatment effects (years 15+): dummy for year of unilateral law in Friedberg (1998)

dyn_gruber_2 Dynamic treatment effects (years 1-2): dummy for year of divorce law reform according to Gruber (2004)

dyn_gruber_3 Dynamic treatment effects (years 3-4): dummy for year of divorce law reform according to Gruber (2004)

dyn_gruber_4 Dynamic treatment effects (years 5-6): dummy for year of divorce law reform according to Gruber (2004)

dyn_gruber_5 Dynamic treatment effects (years 7-8): dummy for year of divorce law reform according to Gruber (2004)

dyn_gruber_6 Dynamic treatment effects (years 9-10): dummy for year of divorce law reform according to Gruber (2004)

dyn_gruber_7 Dynamic treatment effects (years 11-12): dummy for year of divorce law reform according to Gruber (2004)

dyn_gruber_8 Dynamic treatment effects (years 13-14): dummy for year of divorce law reform according to Gruber (2004)

dyn_gruber_9 Dynamic treatment effects (years 15+): dummy for year of divorce law reform according to Gruber (2004)

dyn_johnson_2 Dynamic treatment effects (years 1-2): dummy for year of divorce law reform according to Johnson and Mazingo (2000)

dyn_johnson_3 Dynamic treatment effects (years 3-4): dummy for year of divorce law reform according to Johnson and Mazingo (2000)

dyn_johnson_4 Dynamic treatment effects (years 5-6): dummy for year of divorce law reform according to Johnson and Mazingo (2000)

dyn_johnson_5 Dynamic treatment effects (years 7-8): dummy for year of divorce law reform according to Johnson and Mazingo (2000)

dyn_johnson_6 Dynamic treatment effects (years 9-10): dummy for year of divorce law reform according to Johnson and Mazingo (2000)

dyn_johnson_7 Dynamic treatment effects (years 11-12): dummy for year of divorce law reform according to Johnson and Mazingo (2000)

dyn_johnson_8 Dynamic treatment effects (years 13-14): dummy for year of divorce law reform according to Johnson and Mazingo (2000)

dyn_johnson_9 Dynamic treatment effects (years 15+): dummy for year of divorce law reform according to Johnson and Mazingo (2000)

dyn_mechoulan_2 Dynamic treatment effects (years 1-2): dummy for year of divorce law reform according to Mechoulan (2001)

dyn_mechoulan_3 Dynamic treatment effects (years 3-4): dummy for year of divorce law reform according to Mechoulan (2001)

dyn_mechoulan_4 Dynamic treatment effects (years 5-6): dummy for year of divorce law reform according to Mechoulan (2001)

dyn_mechoulan_5 Dynamic treatment effects (years 7-8): dummy for year of divorce law reform according to Mechoulan (2001)

dyn_mechoulan_6 Dynamic treatment effects (years 9-10): dummy for year of divorce law reform according to Mechoulan (2001)

dyn_mechoulan_7 Dynamic treatment effects (years 11-12): dummy for year of divorce law reform according to Mechoulan (2001)

dyn_mechoulan_8 Dynamic treatment effects (years 13-14): dummy for year of divorce law reform according to Mechoulan (2001)

dyn_mechoulan_9 Dynamic treatment effects (years 15+): dummy for year of divorce law reform according to Mechoulan (2001)

dyn_ellmanlohr1_2 Dynamic treatment effects (years 1-2): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition a)

dyn_ellmanlohr1_3 Dynamic treatment effects (years 3-4): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition a)

dyn_ellmanlohr1_4 Dynamic treatment effects (years 5-6): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition a)

dyn_ellmanlohr1_5 Dynamic treatment effects (years 7-8): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition a)

dyn_ellmanlohr1_6 Dynamic treatment effects (years 9-10): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition a)

dyn_ellmanlohr1_7 Dynamic treatment effects (years 11-12): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition a)

dyn_ellmanlohr1_8 Dynamic treatment effects (years 13-14): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition a)

dyn_ellmanlohr1_9 Dynamic treatment effects (years 15+): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition a)

dyn_ellmanlohr2_2 Dynamic treatment effects (years 1-2): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition b)

dyn_ellmanlohr2_3 Dynamic treatment effects (years 3-4): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition b)

dyn_ellmanlohr2_4 Dynamic treatment effects (years 5-6): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition b)

dyn_ellmanlohr2_5 Dynamic treatment effects (years 7-8): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition b)

dyn_ellmanlohr2_6 Dynamic treatment effects (years 9-10): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition b)

dyn_ellmanlohr2_7 Dynamic treatment effects (years 11-12): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition b)

dyn_ellmanlohr2_8 Dynamic treatment effects (years 13-14): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition b)

dyn_ellmanlohr2_9 Dynamic treatment effects (years 15+): dummy for year of divorce law reform according to Ellman and Lohr (1998) (definition b)

dyn_brinigbuckley_2 Dynamic treatment effects (years 1-2): dummy for year of divorce law reform according to Brinig and Buckley (1998)

dyn_brinigbuckley_3 Dynamic treatment effects (years 3-4): dummy for year of divorce law reform according to Brinig and Buckley (1998)

dyn_brinigbuckley_4 Dynamic treatment effects (years 5-6): dummy for year of divorce law reform according to Brinig and Buckley (1998)

dyn_brinigbuckley_5 Dynamic treatment effects (years 7-8): dummy for year of divorce law reform according to Brinig and Buckley (1998)

dyn_brinigbuckley_6 Dynamic treatment effects (years 9-10): dummy for year of divorce law reform according to Brinig and Buckley (1998)

dyn_brinigbuckley_7 Dynamic treatment effects (years 11-12): dummy for year of divorce law reform according to Brinig and Buckley (1998)

dyn_brinigbuckley_8 Dynamic treatment effects (years 13-14): dummy for year of divorce law reform according to Brinig and Buckley (1998)

dyn_brinigbuckley_9 Dynamic treatment effects (years 15+): dummy for year of divorce law reform according to Brinig and Buckley (1998)

dyn_nakonezny_2 Dynamic treatment effects (years 1-2): dummy for year of divorce law reform according to Nakonezny, Shull and Rodgers (1995)

dyn_nakonezny_3 Dynamic treatment effects (years 3-4): dummy for year of divorce law reform according to Nakonezny, Shull and Rodgers (1995)

dyn_nakonezny_4 Dynamic treatment effects (years 5-6): dummy for year of divorce law reform according to Nakonezny, Shull and Rodgers (1995)

dyn_nakonezny_5 Dynamic treatment effects (years 7-8): dummy for year of divorce law reform according to Nakonezny, Shull and Rodgers (1995)

dyn_nakonezny_6 Dynamic treatment effects (years 9-10): dummy for year of divorce law reform according to Nakonezny, Shull and Rodgers (1995)

dyn_nakonezny_7 Dynamic treatment effects (years 11-12): dummy for year of divorce law reform according to Nakonezny, Shull and Rodgers (1995)

dyn_nakonezny_8 Dynamic treatment effects (years 13-14): dummy for year of divorce law reform according to Nakonezny, Shull and Rodgers (1995)

dyn_nakonezny_9 Dynamic treatment effects (years 15+): dummy for year of divorce law reform according to Nakonezny, Shull and Rodgers (1995)

divx1 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx2 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx3 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx4 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx5 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx6 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx7 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx8 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx9 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx10 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx11 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx12 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx13 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx14 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx15 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx16 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

divx17 Friedberg (1998)'s dummies for coding breaks; see appendix of the paper

Note

Examples section provides replication code for Table 5 and 9 in Armstrong, Weidner, and Zeleneev (2024, "Robust Estimation and Inference in Panels with Interactive Fixed Effects").

References

For the detail of the data and the contruction of the balanced panel, see Friedberg (1998, "Did Unilateral Divorce Raise Divorce Rates? Evidence from Panel Data"), Wolfers (2006, "Did Unilateral Divorce Laws Raise Divorce Rates? A Reconciliation and New Results"), and Kim and Oka (2014, "Divorce Law Reforms and Divorce Rates in the USA: An Interactive Fixed-Effects Approach").

Examples

```
# Following replication requires some time for computation

# # Replication of Table 5 in Armstrong, Weidner, and Zeleneev (2024)
# ## Data cleaning
# df <- PanelIFE::empirical_data[ , -1]       # Load data without two-letter state code
# df <- df[df$id_st != 16 & df$id_st != 33, ] # Drop IN (id == 16) and NM (id == 33)
# XX <- df[ , grep("dyn_uni", colnames(df))]
# # ===
# T <- max(df$year) - min(df$year) + 1
# N <- nrow(df) / T
# K <- ncol(XX)
# # ===
# Y <- matrix(df$div_rate_rev02, nrow = N, ncol = T, byrow = TRUE)
# X <- array(0, dim = c(K, N, T))
# for(k in 1:K) {
#   X[k, , ] <- matrix(XX[ , k], nrow = N, ncol = T, byrow = TRUE)
# }
# # ===
# D <- array(0, dim = c(1, N, T))             # Using one regressor only
# D[1, , ] <- apply(X, c(2, 3), sum)
# X <- D
# K <- 1
# ## Setting estimation parameters
# set.seed(1)
# beta0 <- rep(0, K)
# alpha <- 0.05                               # 1 - confidence level
# Rmax <- 6                                   # Maximum number of the factors
# lambda_known <- matrix(rep(1, N))           # Standard time effects
# f_known <- cbind(rep(1, T), (1:T), (1:T)^2) # Individual effects + time trend + time trend^2
# ## Least squares estimation of linear panel data model with interactive fixed effects (LS factors)
# LS_summary <- setNames(data.frame(matrix(NA, nrow = Rmax, ncol = 4)),
#                        c("R", "beta", "CI_LB", "CI_UB"))
# for(R in 1:Rmax) {
#   res <- ls_factor(Y = Y, X = X, R = R,
#                    lambda_known = lambda_known, f_known = f_known,
#                    report = "silent", precision_beta = 10^(-8), method = "m1",
#                    start = beta0, repMIN = 30, repMAX = 300, M1 = 2, M2 = 2)
#   LS_summary$R[R] <- R
#   LS_summary$beta[R] <- res$beta - res$bcorr2 - res$bcorr3
#  LS_summary$CI_LB[R] <- LS_summary$beta[R] - sqrt(diag(res$Vbeta2)) * qnorm(1 - alpha/2)
#  LS_summary$CI_UB[R] <- LS_summary$beta[R] + sqrt(diag(res$Vbeta2)) * qnorm(1 - alpha/2)
# }
# ## Robust estimation and inference in panels with interactive fixed effects (honest weak factors)
# Debiased_summary <- setNames(data.frame(matrix(NA, nrow = Rmax * 3, ncol = 5)),
#                              c("R", "Rw", "beta", "CI_LB", "CI_UB"))
# for(R in 1:Rmax) {
# res <- honest_weak_factors(Y = Y, X = X, R = R,
#                            Gamma_LS = NULL, alpha = 0.05, clustered_se = FALSE,
#                            lambda_known = lambda_known, f_known = f_known,
#                            itermax = 75, reltol = 10^(-6))
#   Debiased_summary$R[(3*R-2):(3*R)] <- R
#   Debiased_summary$Rw[(3*R-2):(3*R)] <- c(0, 1, R)
#   Debiased_summary$beta[(3*R-2):(3*R)] <- res$beta
#   Debiased_summary$CI_LB[(3*R-2):(3*R)] <- res$LB[c(1:2, R+1), 2]
#   Debiased_summary$CI_UB[(3*R-2):(3*R)] <- res$UB[c(1:2, R+1), 2]
```

```
# }
# ## Print results
# print(round(LS_summary, 3), row.names = FALSE)
# print(round(Debiased_summary, 3), row.names = FALSE)

# # Replication of Table 10 in Armstrong, Weidner, and Zeleneev (2024)
# ## Data cleaning
# df <- PanelIFE::empirical_data[ , -1]       # Load data without two-letter state code
# df <- df[df$id_st != 16 & df$id_st != 33, ] # Drop IN (id == 16) and NM (id == 33)
# XX <- df[ , grep("dyn_uni", colnames(df))]
# # ===
# T <- max(df$year) - min(df$year) + 1
# N <- nrow(df) / T
# K <- ncol(XX)
# # ===
# Y <- matrix(df$div_rate_rev02, nrow = N, ncol = T, byrow = TRUE)
# X <- array(0, dim = c(K, N, T))
# for(k in 1:K) {
#   X[k, , ] <- matrix(XX[ , k], nrow = N, ncol = T, byrow = TRUE)
# }
# # ===
# D <- array(0, dim = c(4, N, T))             # Using four regressors
# D[1, , ] <- apply(X[1:2, , ], c(2, 3), sum)
# D[2, , ] <- apply(X[3:4, , ], c(2, 3), sum)
# D[3, , ] <- apply(X[5:6, , ], c(2, 3), sum)
# D[4, , ] <- apply(X[7:8, , ], c(2, 3), sum)
# X <- D
# K <- 4
# ## Setting estimation parameters
# set.seed(1)
# beta0 <- rep(0, K)
# alpha <- 0.05                               # 1 - confidence level
# Rmax <- 6                                   # Maximum number of the factors
# lambda_known <- matrix(rep(1, N))           # Standard time effects
# f_known <- cbind(rep(1, T), (1:T), (1:T)^2) # Individual effects + time trend + time trend^2
# ## Least squares estimation of linear panel data model with interactive fixed effects (LS factors)
# LS_summary <- setNames(data.frame(matrix(NA, nrow = Rmax * K, ncol = 5)),
#                        c("R", "k", "beta", "CI_LB", "CI_UB"))
# for(R in 1:Rmax) {
#   res <- ls_factor(Y = Y, X = X, R = R,
#                    lambda_known = lambda_known, f_known = f_known,
#                    report = "silent", precision_beta = 10^(-8), method = "m1",
#                    start = beta0, repMIN = 30, repMAX = 300, M1 = 2, M2 = 2)
#   LS_summary$k[(K*R-(K-1)):(K*R)] <- 1:K
#   LS_summary$R[(K*R-(K-1)):(K*R)] <- R
#   LS_summary$beta[(K*R-(K-1)):(K*R)] <- as.numeric(res$beta - res$bcorr2 - res$bcorr3)
#   LS_summary$CI_LB[(K*R-(K-1)):(K*R)] <-
#     LS_summary$beta[(K*R-(K-1)):(K*R)] - sqrt(diag(res$Vbeta2)) * qnorm(1 - alpha/2)
#   LS_summary$CI_UB[(K*R-(K-1)):(K*R)] <-
#     LS_summary$beta[(K*R-(K-1)):(K*R)] + sqrt(diag(res$Vbeta2)) * qnorm(1 - alpha/2)
# }
# ## Robust estimation and inference in panels with interactive fixed effects (honest weak factors)
# Debiased_summary <- setNames(data.frame(matrix(NA, nrow = Rmax * K * 3, ncol = 6)),
#                              c("R", "k", "Rw", "beta", "CI_LB", "CI_UB"))
# for(R in 1:Rmax) {
#   res <- honest_weak_factors(Y = Y, X = X, R = R,
#                              Gamma_LS = NULL, alpha = 0.05, clustered_se = FALSE,
```

```
#                              lambda_known = lambda_known, f_known = f_known,
#                              itermax = 50, reltol = 10^(-4))
#   Debiased_summary$k[(3*K*R-(3*K-1)):(3*K*R)] <- rep(1:K, each = 3)
#   Debiased_summary$R[(3*K*R-(3*K-1)):(3*K*R)] <- R
#   Debiased_summary$Rw[(3*K*R-(3*K-1)):(3*K*R)] <- rep(c(0, 1, R), K)
#   Debiased_summary$beta[(3*K*R-(3*K-1)):(3*K*R)] <- rep(res$beta, each = 3)
#  Debiased_summary$CI_LB[(3*K*R-(3*K-1)):(3*K*R)] <- c(as.matrix(res$LB[c(1:2, R+1), -1]))
#  Debiased_summary$CI_UB[(3*K*R-(3*K-1)):(3*K*R)] <- c(as.matrix(res$UB[c(1:2, R+1), -1]))
# }
# ## Print results
# print(round(LS_summary, 3), row.names = FALSE)
# print(round(Debiased_summary, 3), row.names = FALSE)
```

---

honest_weak_factors *Robust Estimation and Inference in Panels with Interactive Fixed Effects*

Description

This method considers estimation and inference for a regression coefficient in panels with interactive fixed effects (i.e., with a factor structure). As the previously developed estimators and confidence intervals (CIs) might be heavily biased and size-distorted when some of the factors are weak, this method has estimators with improved rates of convergence and bias-aware CIs that are uniformly valid regardless of whether the factors are strong or not.

Usage

```
honest_weak_factors(
  Y,
  X,
  R,
  Gamma_LS = NULL,
  alpha = 0.05,
  clustered_se = FALSE,
  lambda_known = NA,
  f_known = NA,
  itermax = 75,
  reltol = 10^(-6)
)
```

Arguments

Y	$N \times T$ matrix of outcomes
X	$K \times N \times T$ tensor of regressors
R	A positive integer, indicates the number of interactive fixed effects in the estimation. Note that this number does not include the known factors and loadings defined below
Gamma_LS	(Optional) A preliminary LS estimate of the matrix of fixed effects; it will be computed if not provided

`alpha` (Optional) It determines the `1 - alpha` coverage of the constructed confidence interval, where default is set to `alpha = 0.05`

`clustered_se` (Optional) Whether or not performing clustered standard error, where default is set to `clustered_se = FALSE`

`lambda_known` (Optional) $N \times Rex1$ matrix of known factor loadings, e.g., `lambda_known = matrix(rep(1, N), nrow = N, ncol = 1)` to control standard time dummies. Default is set to `lambda_known = NA` (or equivalently `matrix(NA, nrow = N, ncol = 0)`), i.e., there is no known factor loadings

`f_known` (Optional) $T \times Rex2$ matrix of known factor, e.g., `f_known = matrix(rep(1, T), nrow = T, ncol = 1)` to control standard individual specific fixed effects. Default is set to `f_known = NA` (or equivalently `matrix(NA, nrow = T, ncol = 0)`), i.e., there is no known factor

`itermax` (Optional) Maximum iteration allowed while optimizing for the weights A, where default is set to `itermax = 50` for faster computation

`reltol` (Optional) Relative convergence tolerance for the optimization algorithm to stop if it is unable to reduce the value by `reltol * (abs(val) + reltol)` after `0.75 * itermax` steps. , where default is set to `reltol = 10^(-4)` for faster computation

Details

Disclaimer: This function is the implementation of Armstrong, Weidner, and Zeleneev (2024, "Robust Estimation and Inference in Panels with Interactive Fixed Effects"). This code is offered with no guarantees. Not all features of this code were properly tested. Please let me know if you find any bugs or encounter any problems while using this code. All feedback is appreciated.

Linear panel regression model with weak factors:

- We consider a linear panel regression model of the form

$$Y_{it} = X_{it}\beta + \sum_{k=1}^{K} Z_{k,it}\delta_k + \Gamma_{it} + U_{it}$$

  where
  - Y_{it}, $X_{i,t}$, and $Z_{k,it}$ are the observed outcome variable and covariates,
  - Γ_{it} is the error component that can be correlated with X_{it} and $Z_{k,it}$,
  - U_{it} is the error component modelled as a mean-zero random shock,
  - and large panel is considered, where both N and T are relatively large.
- Model for Γ_{it} is referred to as a factor model with factor loadings λ_{ir} and factors f_{tr}, with R being the number of factors.
- Other than having some strong factors, which requires λ_{ir} and f_{tr} to have sufficient variation across i and over t, model here allows weak factors.
- Proposed method provides the bias-aware confidence intervals that are uniformly valid regardless of whether the factors are strong or not.

Debiasing approach:

- Access the preliminary estimate $\hat{\Gamma}_{pre}$ along with a bound $\hat{C}$ on the nuclear norm $\|\Gamma - \hat{\Gamma}_{pre}\|_*$.
- Then consider the regression with the augmented outcomes $\tilde{Y}_{it} := Y_{it} - \hat{\Gamma}_{pre,it}$.

- Construct the confidence interval with the preliminary estimate $\hat{\Gamma}_{pre}$ and bound $\hat{C}$ on the nuclear norm of its estimation error. Such confidence interval is bias-aware, that is, using the bound $\hat{C}$ will take any remaining bias into account after the previous debias procedure.
- For detailed implementation, please refer to Armstrong, Weidner, and Zeleneev (2024, "Robust Estimation and Inference in Panels with Interactive Fixed Effects").

Value

A list of results, where

- `beta` is the point estimate
- `bias` is the worst-case bias
- `LB` are the lower bounds of the `1 - alpha` confidence intervals, from number of weak factors being 0 to R
- `UB` are the upper bounds of the `1 - alpha` confidence intervals, from number of weak factors being 0 to R
- `A` is the matrix of weights
- `parameter` is the list of input parameters, including `Gamma_LS`, `alpha`, and `clustered_se`

Note

We assume that all provided input parameters have values and dimensions as described above.

References

For a description of the model see Armstrong, Weidner, and Zeleneev (2024, "Robust Estimation and Inference in Panels with Interactive Fixed Effects").

Examples

```
dt <- sample_data(N = 100, T = 20, R = 1, kappa = c(0.5))
res <- honest_weak_factors(Y = dt$Y, X = dt$X, R = dt$R,
                           Gamma_LS = NULL, alpha = 0.05)
sum_res <- summary(res)
print(sum_res)
```

---

ls_factor — *Least Squares Estimation of Linear Panel Data Models with Interactive Fixed Effects*

Description

This function estimates least squares estimator in a linear panel regression model with factors appearing as interactive fixed effects.

Usage

```
ls_factor(
  Y,
  X,
  R,
  lambda_known = NA,
  f_known = NA,
  report = "report",
  precision_beta = 10^(-8),
  method = "m1",
  start,
  repMIN,
  repMAX,
  M1 = 1,
  M2 = 0,
  DoF_adj = FALSE
)
```

Arguments

Y	$N \times T$ matrix of outcomes, where we assume a balanced panel, i.e. all elements of Y are known
X	$K \times N \times T$ tensor of regressors, where we assume a balanced panel, i.e. all elements of X are known
R	A positive integer, indicates the number of interactive fixed effects in the estimation; this R does not include the number of known factors and loadings
lambda_known	(Optional) $N \times Rex1$ matrix of known factor loadings, e.g., lambda_known = matrix(rep(1, N), nrow = N, ncol = 1) to control standard time dummies. Default is set to lambda_known = matrix(NA, nrow = N, ncol = 0), i.e., there is no known factor loadings
f_known	(Optional) $T \times Rex2$ matrix of known factor, e.g., f_known = matrix(rep(1, T), nrow = T, ncol = 1) to control standard individual specific fixed effects. Default is set to f_known = matrix(NA, nrow = T, ncol = 0), i.e., there is no known factor
report	(Optional) Whether or not to report the progress. "silent" has the program running silently; "report" has the program reporting what it is doing
precision_beta	(Optional) Defines stopping criteria for numerical optimization, namely optimization is stopped when difference in beta relative to previous opimtization step is smaller than "precision_beta" (uniformly over all K components of beta). Note that the actual precision in beta will typically be lower than precision_beta, depending on the convergence rate of the procedure.
method	(Optional) Optimization method option of choice. Options include "m1" and "m2"
start	(Optional) $K \times 1$ vector, first starting value for numerical optimization
repMIN	(Optional) Minimal number, which is a positive integer, of runs of optimization with different starting point
repMAX	(Optional) Maximal number, which is a positive integer, of runs of optimization (in case numerical optimization doesn't terminate properly, we do multiple runs even for repMIN = 1)

M1
: (Optional) A positive integer, bandwidth for bias correction for dynamic bias (bcorr1), M1 is the number of lags of correlation between regressors and errors that is corrected for in dynamic bias correction

M2
: (Optional) A non-negative integer, bandwidth for bias correction for time-serial correlation (bcorr3), M2 = 0 only corrects for time-series heteroscedasticity, while M2 > 0 corrects for time-correlation in errors up to lag M2

DoF_adj
: (Optional) Whether or not to adjust for degree of freedom, where default is set to DoF_adj = FALSE

Details

Disclaimer: This function is translated and modified from the Matlab function LS_factor.m by Martin Weidner, and the documentation details are also mainly from the original function. This code is offered with no guarantees. Not all features of this code were properly tested. Please let me know if you find any bugs or encounter any problems while using this code. All feedback is appreciated.

Different computational methods:

- Method 1 (recommended default method) iterates the following two steps until convergence:
  - Step 1: forgiven beta compute update for lambda and f as principal components of Y - beta * X (same as in method 1)
  - Step 2: forgiven lambda and f run a pooled OLS regression of Y - lambda * t(f) on X to update beta
- Method 2 (described in Bai, 2009) iterates the following two steps until convergence:
  - Step 1: forgiven beta compute update for lambda and f as principal components of Y - beta * X (same as in method 1)
  - Step 2: forgiven lambda and f run a pooled OLS regression of Y - lambda * t(f) on X to update beta

  The procedure is repeated multiple times with different starting values.

Comments:

- Another method would be to use step 1 as in Method 1 & 2, but to replace step 2 with a regression of Y on either M_lambda * X or X * M_f, i.e. to only project out either lambda or f in the step 2 regression. Bai (2009) mentions this method and refers to Ahn, Lee, and Schmidt (2001), Kiefer (1980) and Sargan (1964) for this. We have not tested this alternative method, but we suspect that Method 1 performs better in terms of speed of convergence.
- This alternative method and the method proposed by Bai (2009), i.e. "method 2" here, have the property of reducing the LS objective function in each step. This is not true for Method 1 and may be seen as a disadvantage of Method 1. However, we found this to be a nice feature, because we use this property of Method 1 as a stopping rule: if the LS objective function does not improve, then we know we are "far away" from a proper minimum, so we stop the iteration and begin the iteration with another randomly chosen starting value. Note that multiple runs with different starting values are required anyways for all methods (because the LS objective function may have multiple local minimal).
- We recommend method 1, because each iteration step is fast and its rate of convergence in our tests was very good (faster than method 2). However, we have not much explored the relative sensitivity of the different methods towards the choice of starting value. Note that by choosing the quickest method (method 1) one can try out more different starting values of the procedure in the same amount of time. Nevertheless, it may well be that method 2 or the alternative method described above perform better in certain situations.

Value

A list of results, where

- `beta` is the parameter estimate
- `exitflag` = 1 if iteration algorithm properly converged at optimal beta, and `exitflag` = -1 if iteration algorithm did not properly converge at optimal beta
- `lambda` is the estimate for factor loading
- `f` is the estimate for factors
- `Vbeta1,2,3` are estimated variance-covariance matrices of beta, assuming
  1. homoscedasticity of errors in both dimensions
  2. heteroscedasticity of errors in both dimensions
  3. allowing for time-serial correlation up to lag M2 (i.e. if M2 == 0, then Vbeta2 == Vbeta3)
- `bcorr1,2,3` are estimates for the three different bias components (needs to be subtracted from beta to correct for the bias), where
  1. is bias due to pre-determined regressors
  2. is bias due to cross-sectional heteroscedasticity of errors
  3. is bias due to time-serial heteroscedasticity and time-serial correlation of errors
- `parameter` is the list of input parameters, including `Gamma_LS`, `alpha`, and `clustered_se`

Note

We assume that all provided input parameters have values and dimensions as described above. The program could be improved by checking that this is indeed the case.

References

For a description of the model and the least squares estimator see e.g. Bai (2009, "Panel data models with interactive fixed effects"), or Moon and Weidner (2017, "Dynamic Linear Panel Regression Models with Interactive Fixed Effects"; 2015, "Linear Regression for Panel with Unknown Number of Factors as Interactive Fixed Effects")

Examples

```
dt <- sample_data(N = 100, T = 20, R = 1, kappa = c(0.5))
res <- ls_factor(Y = dt$Y, X = dt$X, R = dt$R, report = "silent",
                 precision_beta = 10^(-8), method = "m1",
                 start = c(0), repMIN = 3, repMAX = 10, M1 = 2, M2 = 2)
sum_res <- summary(res)
print(sum_res)
```

sample_data *Generate Sample Data with Interactive Fixed Effects*

Description

This function generates sample data with interactive fixed effects.

Usage

```
sample_data(N = 100, T = 20, R = 1, kappa = c(0.5))
```

Arguments

N	Number of total individuals.
T	Number of total time periods.
R	Number of factors.
kappa	Strength of each factor.

Details

The design follows that data generating process of the simulation in Armstrong, Weidner, and Zeleneev (2024, "Robust Estimation and Inference in Panels with Interactive Fixed Effects").

$$Y_{i,t} = X_{i,t}\beta + \sum_{r=1}^{R} \kappa_r \lambda_{i,r}, f_{t,r} + U_{i,t}$$

$$X_{i,t} = \sum_{r=1}^{R} \lambda_{i,r}, f_{t,r} + V_{i,t}$$

where κ_r controls the strength of factor $f_{t,r}$, and R is the number of factors. The factors, loadings, and error terms follows the distributions of:

$$\lambda_i \sim N(0, I_R) \perp f_t \sim N(0, I_R) \perp \begin{pmatrix} U_{i,t} \\ V_{i,t} \end{pmatrix} \sim N\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \sigma_U^2 & 0 \\ 0 & \sigma_V^2 \end{pmatrix} \right).$$

We consider the setting of $(\beta, \sigma_U^2, \sigma_V^2) = (0, 1, 1)$, various of N, T, and different strength of factors $\kappa_r \in [0, 1]$. In this simple simulation data, we only consider $R \in \{1, 2\}$.

Value

A list of simple simulation data, where

- Y is the $N \times T$ matrix of outcomes.
- X is the $K \times N \times T$ tensor of regressors.
- R is the number of total individuals.
- N is the number of total individuals.
- T is the number of total time periods.
- kappa is the vector of strength factors.

`summary.honest_weak_factors` *Summarizing Robust Estimation and Inference in Panels with Interactive Fixed Effects*

Description

`summary` method for object of class `honest_weak_factors` and returned object is of class summary.honest_weak_factors

Usage

```
## S3 method for class 'honest_weak_factors'
summary(object, ...)

## S3 method for class 'summary.honest_weak_factors'
print(x, digits = 4, labels = TRUE, ...)
```

Arguments

`object`	The `honest_weak_factors` fitted result of class `honest_weak_factors`.
`...`	Confidence level `alpha` can be set for computing confidence intervals. Default is set to 0.05.
`x`	The summarized result of class `summary.ls_factor`.
`digits`	Number of digits to use when printing. Default is set to 4.
`labels`	Option of whether or not to print labels in the summary. Default is set to `TRUE`. Only parameters estimates will be printed when it's set to `FALSE`.

Details

Summarizing the fitted `honest_weak_factors` object and computing

- Debiased parameter estimate.
- Standard error.
- Worst-case bias of the estimator.
- With and without bias-aware confidence intervals.

Value

A list of results, where

- `beta` is the debiased parameter estimate.
- `se` is the standard errors of the parameters estimator.
- `bias` is the worst-case bias of the estimator.
- `CI` is the bias-aware confidence interval.
- `CI_unadj` is the confidence interval without bias adjustment.
- `A` is the choice of weight matrix.

`summary.ls_factor` *Summarizing Least Squares Estimation of Linear Panel Data Models with Interactive Fixed Effects*

Description

`summary` method for object of class `ls_factor` and returned object is of class summary.ls_factor

Usage

```
## S3 method for class 'ls_factor'
summary(object, ...)

## S3 method for class 'summary.ls_factor'
print(x, digits = 4, labels = TRUE, ...)
```

Arguments

`object`	The `ls_factor` fitted result of class `ls_factor`.
`...`	Confidence level `alpha` can be set for computing confidence intervals. Default is set to 0.05.
`x`	The summarized result of class `summary.ls_factor`.
`digits`	Number of digits to use when printing. Default is set to 4.
`labels`	Option of whether or not to print labels in the summary. Default is set to `TRUE`. Only parameters estimates will be printed when it's set to `FALSE`.

Details

Summarizing the fitted `ls_factor` object and computing

- Estimates under different bias correction schemes.
- Variance-covariance matrix under different assumptions.
- Confidence intervals and their lengths under different settings.

Value

A list of results, where

- `beta` is the table of parameters estimates under different bias correction schemes.
- `CI` is the table of confidence intervals and their lengths under different settings.
- `var_cov` are the variance-covariance matrices under different assumptions.

Index