BIOMETRICS 000, 000–000 DOI: 000
000 0000

Codes for Semiparametric Mixture Regression for Asynchronous Longitudinal Data Using Multivariate Functional Principal Component Analysis

Ruihan Lu*, Yehua Li, and Weixin Yao*****

Department of Statistics, University of California, Riverside, CA, U.S.

**email:* rlu014@ucr.edu

***email:* yehua.li@ucr.edu

****email:* weixin.yao@ucr.edu

This paper has been submitted for consideration for publication in *Biometrics*

Package 'MR.ALDA'

Title Semiparametric Mixture Regression for Asynchronous Longitudinal Data

Description Estimation of asynchronous longitudinal data using multivariate functional principal component analysis; Estimation of the mixed models including latent class; Mixed models for multivariate longitudinal outcomes using a maximum likelihood estimation method

1 SWAN folder

- basis1.csv file provides the nonorthogonal basis for $\boldsymbol{X}_1$, which is glucose.
- basis2.csv file provides the nonorthogonal basis for $\boldsymbol{X}_2$, which is triglyceride.
- basis3.csv file provides the nonorthogonal basis for $\boldsymbol{X}_2$, which is systolic blood pressure.
- mu1.csv file provides the initial value for the mean function of glucose.
- mu2.csv file provides the initial value for the mean function of triglyceride.
- mu3.csv file provides the initial value for the mean function of systolic blood pressure.
- psi1.csv file provides the initial value for the eigenfunction of glucose;
- psi2.csv file provides the initial value for the eigenfunction of triglycerides.
- psi3.csv file provides the initial value for the eigenfunction of systolic blood pressure.
- probability1.csv, probability2.csv, probability3.csv, probability4.csv, and probability5.csv files provide the initial value of classification probability (π_{ic}) according to different number of clusters.
- new1class.py, new2class.py, new3class.py, new4class.py, and new5class.py files consist of functions **Emerald** need and return a text result file.
- W1.csv, W2.csv, W3.csv, and Ymeasure.csv are data files we would like to analyze.
- bootstrap for SWAN.py calculate the standard errors for each parameter using bootstrap data.
- result_1cluster_hpcc.txt, result_2cluster_hpcc.txt, result_3cluster_hpcc.txt, result_4cluster_hpcc.txt, and result_5cluster_hpcc.txt files are the results from **Emerald**.
- After determining the number of subgroups, return the results as fit_beta.csv, fit_gamma.csv, fit_mu.csv, fit_psi.csv, and fit_score.csv.
- parametric bootstap.R generates the bootstrap sample for SWAN data.

2 Simulation study folder

2.1 2 subgroup folder

- Corrected Simulation Data for 2 classes.ipynb simulates data for 2-subgroup.
- basis1.csv, basis2.csv provide the initial value of nonorthogonal basis for $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, respectively.
- mu1.csv, mu2.csv provide the initial value of the mean function for $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, respectively.
- psi1.csv, psi2.csv provide the initial value of the mean function for $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, respectively.
- probability2.csv provides the initial value of classification probability, which is π_{ic}.
- result_2cluster_hpcc.txt is the result file.

2.2 3 subgroup

- Corrected Simulation Data for 3 classes.ipynb simulates data for 3-subgroup.
- basis1.csv, basis2.csv provide the initial value of nonorthogonal basis for $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, respectively.
- mu1.csv, mu2.csv provide the initial value of the mean function for $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, respectively.
- psi1.csv, psi2.csv provide the initial value of the mean function for $\boldsymbol{X}_1$, $\boldsymbol{X}_2$, respectively.
- probability3.csv provides the initial value of classification probability, which is π_{ic}.
- result_3cluster_hpcc.txt is the result file.
- X1measure, X2measure, and Ymeasure folder contain datafile for 200 simulation runs.

3 Detail function in ***Python*** file

- fun_df2dict (data): transfer each column into a nested dictionary.
- fun_diag_blocks (data): transfer a $p \times 1$ vector into a $p \times p$ squared matrix.
- fun_b_t_N_orth_b_t (n_spline): construct a orthogonal matrix. The number of knots is required.

- fun_theta_mu (data_basis, data_original_theta_mu): calculate the matrix of observed mean function for variable v, which is $\boldsymbol{B}_{iv}\boldsymbol{\theta}_{\mu v}$.
- fun_theta_psi (data_orth_basis, data_original_theta_psi, data_sigma, data_pick_n_pc): after determining the number of principal components, calculate the matrix of observed eigenfunction for each variable v, which is $\boldsymbol{B}_{iv}\boldsymbol{\Theta}_{\psi v}$.
- fun_matrix_B_iv (n, dx, data_id, data_W, data_orth_b_t): calculate basis matrices for the observed time in all $\boldsymbol{W}$ variables. Then return to the orthogonal basis.
- fun_matrix_unorth_B_iv (n, dx, data_id, data_W, data_orth_b_t): calculate basis matrices for the observed time in all $\boldsymbol{W}$ variables. Then return to the nonorthogonal basis.
- fun_matrix_tilde_mu_iv (n, dx, data_theta_mu, data_B_unorth_iv): given the nonorthogonal spline basis matrix, calculate the mean matrix for each subject within each variable of $\boldsymbol{W}$.
- fun_matrix_tilde_psi_iv(n, dx, data_theta_psi, data_B_iv): given the orthogonal spline basis matrix, calculate the eigenfunction matrix for each subject within each variable of $\boldsymbol{W}$.
- fun_matrix_B_star_iv (n, dx, data_id, data_Y, data_orth_b_t): calculate basis matrices for the observed time in $\boldsymbol{Y}$ variable. Then return to the orthogonal basis.
- fun_matrix_tilde_mu_star_iv(n, dx, data_theta_mu, data_B_unorth_star_iv): given the nonorthogonal spline basis matrix, calculate the mean matrix for each subject within $\boldsymbol{Y}$.
- fun_matrix_tilde_mu_star_ic_v(n, C, dx, data_beta, data_tilde_mu_star_iv): multiple group-specific effect β_c with mean matrix for each subject for all variables in $\boldsymbol{W}$.
- fun_matrix_tilde_psi_star_iv (n, dx, data_theta_psi, data_B_star_iv): given the orthogonal spline basis matrix, calculate the eigenfunction matrix for each subject within $\boldsymbol{Y}$.
- fun_matrix_tilde_psi_star_ic_v (n, C, dx, data_beta, data_tilde_psi_star_iv): multiple group-specific effects β_c with a discrete matrix of eigenfunction for each subject for all variables in $\boldsymbol{W}$.

- fun_matrix_tilde_W_iv (n, dx, data_id, data_W, data_tilde_mu_iv): calculate the value of $\widetilde{\boldsymbol{W}}_{iv}$, which is $\boldsymbol{W}_{iv} - \boldsymbol{B}^*_{iv}\boldsymbol{\theta}_{\mu v}$.
- fun_matrix_tilde_Y_ic (n, C, dz, data_id, data_Z, data_Y, data_beta, data_tilde_mu_star_ic_v): calculate the value of $\widetilde{\boldsymbol{Y}}_{ic}$, which is $\boldsymbol{Y}_i - \beta_{0,c}\mathbf{1}_{m_{y,i}} - \sum_{v=1}^{d_x} \beta_{x,cv}\boldsymbol{B}_{iv}\boldsymbol{\theta}_{\mu v} - \boldsymbol{Z}_i\boldsymbol{\beta}_{z,c}$.
- fun_matrix_G_y_ic (n, C, data_id, data_Y, data_var_Y): construct variance-covariance matrix for each subject in the cth subgroup.
- fun_list_invert_lambda_i (n, dx, data_id, data_W, data_var_W): construct the inverse matrix of variance in $\boldsymbol{W}$ for each subject.