Package ‘WRI’

November 23, 2020

Type Package

Title Wasserstein Regression Inference

Version 0.1.0

Author Xi Liu [aut, cre], Matthew Coleman [aut], Alexander Petersen [aut]


License GPL-2

Encoding UTF-8

LazyData true

LinkingTo Rcpp, RcppArmadillo

Depends R (>= 3.6.0)

Imports fdapace (>= 0.2.0), fdadensity (>= 0.1.2), Rfast (>= 1.9.8), CVXR (>= 0.99.7), expm (>= 0.999-4), ggplot2 (>= 3.2.1), gridExtra (>= 2.3), stats, Rcpp (>= 1.0.3), locfit (>= 1.5-9.1), mvtnorm (>= 1.1-0), locpol (>= 0.7), modeest (>= 2.4.0), methods, rlang

RoxygenNote 7.0.2

Suggests knitr, rmarkdown, testthat (>= 2.1.0)

VignetteBuilder knitr

NeedsCompilation yes

Maintainer Xi Liu <xiliu@ucsb.edu>

Repository CRAN

Date/Publication 2020-11-23 08:00:06 UTC
**R topics documented:**

- confidenceBands
- den2Q_qd
- globalFtest
- partialFtest
- print.summary_WRI
- quan2den_qd
- simulate_quantile_curves
- strokeCTdensity
- summary_WRI
- wass_R2
- wass_regress

**Index**

<table>
<thead>
<tr>
<th>confidenceBands</th>
<th>Confidence Bands for Wasserstein Regression</th>
</tr>
</thead>
</table>

**Description**

Confidence Bands for Wasserstein Regression

**Usage**

```r
confidenceBands(
  wass_regress_res,
  Xpred_df,
  level = 0.95,
  delta = 0.01,
  type = "density",
  figure = TRUE,
  fig_num = NULL
)
```

**Arguments**

- `wass_regress_res` 
  an object returned by the `wass_regress` function
- `Xpred_df` 
  k-by-p matrix (or dataframe, or named vector) used for prediction. Note that `Xpred_df` should have the same column names with `Xfit_df` used in `wass_regress_res`
- `level` 
  confidence level
- `delta` 
  boundary control value in density band computation. Must be a value in the interval (0, 1/2) (default: 0.01)
- `type` 
  'density', 'quantile' or 'both'
  - 'density': density function bands will be returned (and plotted if `figure` = TRUE)
confidenceBands

- 'quantile': quantile function and CDF bands will be returned (and plotted if figure = TRUE)
- 'both': three kinds of bands, density function, quantile function and CDF bands will be returned (and plotted if figure = TRUE)

figure logical; if TRUE, return a sampled plot (default: TRUE)
fig_num the fig_num-th row of Xpred_df will be used for visualization of confidence bands. If NULL, then fig_num is randomly chosen (default: NULL)

Details

This function computes intrinsic confidence bands for Xpred_df if type = 'quantile' and density bands if type = 'density', and visualizes the confidence and/or density bands when figure = TRUE.

Value

a list containing the following lists:

den_list:  
  - fpred: k-by-m matrix, predicted density function at Xpred_df.
  - f_ux: k-by-m matrix, upper bound of confidence bands of density functions.
  - f_lx: k-by-m matrix, lower bound of confidence bands of density functions.
  - Qpred: k-by-m matrix, f_lx[i, ], f_ux[i, ] and fpred[i, ] evaluated on Qpred[i, ] vector.

quan_list:  
  - Qpred: k-by-m matrix of predicted quantile functions.
  - Q_ux: k-by-m matrix of upper bound of quantile functions.
  - Q_lx: k-by-m matrix of lower bound of quantile functions.
  - t_vec: a length m vector - common grid for all quantile functions.

cdf_list:  
  - fpred: k-by-m matrix, predicted density function.
  - Fpred: k-by-m matrix, predicted cumulative distribution functions.
  - F_ux: k-by-m matrix, upper bound of cumulative distribution functions.
  - F_lx: k-by-m matrix, lower bound of cumulative distribution functions.
  - Fsup: k-by-m matrix, fpred[i, ], F_lx[i, ], F_ux[i, ] and Fpred[i, ] evaluated on Fsup[i, ] vector.

Examples

alpha = 2
beta = 1
n = 50
x1 = runif(n)
t_vec = unique(c(seq(0, 0.05, 0.001), seq(0.05, 0.95, 0.05), seq(0.95, 1, 0.001)))
set.seed(1)
quan_obs = simulate_quantile_curves(x1, alpha, beta, t_vec)
Xfit_df = data.frame(x1 = x1)
res = wass_regress(rightside_formula = ~., Xfit_df = Xfit_df,
                   Ytype = 'quantile', Ymat = quan_obs, Sup = t_vec)
confidence_Band = confidenceBands(res, Xpred_df = data.frame(x1 = c(-0.5, 0.5)),
type = 'both', fig_num = 2)
data(strokeCTdensity)
predictor = strokeCTdensity$predictors
dSup = strokeCTdensity$densitySupport
densityCurves = strokeCTdensity$densityCurve
xpred = predictor[2:3, ]

res = wass_regress(rightside_formula = ~., Xfit_df = predictor,
Ytype = 'density', Ymat = densityCurves, Sup = dSup)
confidence_Band = confidenceBands(res, Xpred_df = xpred, type = 'density', fig_num = 1)

den2Q_qd  

convert density function to quantile and quantile density function

Description

convert density function to quantile and quantile density function

Usage

den2Q_qd(densityCurves, dSup, t_vec)

Arguments

densityCurves  n-by-m matrix of density curves
dSup            length m vector contains the common support grid of the density curves
t_vec            common grid for quantile functions

globalFtest  

global F test for Wasserstein regression

Description

global F test for Wasserstein regression

Usage

globalFtest(  
    wass_regress_res,  
    alpha = 0.05,  
    permutation = FALSE,  
    numPermu = 200,  
    bootstrap = FALSE,  
    numBoot = 200
  )
**Arguments**

- **wass_regress_res**
  - an object returned by the `wass_regress` function
- **alpha**
  - type one error rate
- **permutation**
  - logical; perform permutation global F test (default: FALSE)
- **numPermu**
  - number of permutation samples if permutation = TRUE
- **bootstrap**
  - logical; bootstrap global F test (default: FALSE)
- **numBoot**
  - number of bootstrap samples if bootstrap = TRUE

**Details**

- four methods used to compute p value of global F test
  - **truncated**: asymptotic inference, p-value is obtained by truncating the infinite summation of eigenvalues into the first K terms, where the first K terms explain more than 99.99% of the variance.
  - **satterthwaite**: asymptotic inference, p-value is computed using Satterthwaite’s approximation method of mixtures of chi-square.
  - **permutation**: resampling technique; Wasserstein SSR is used as the F statistic.
  - **bootstrap**: resampling technique; Wasserstein SSR is used as the F statistic.

**Value**

- a list containing the following fields:
  - **wasserstein.F_stat**
    - the Wasserstein F statistic value in Satterthwaite method.
  - **chisq_df**
    - the degree of freedom of the null chi-square distribution.
  - **summary_df**
    - a dataframe containing the following columns:
      - **method**: methods used to compute p value, see details
      - **statistic**: the test statistics
      - **critical_value**: critical value
      - **p_value**: p value of global F test

**Examples**

```r
data(strokeCTdensity)
predictor = strokeCTdensity$predictors
dSup = strokeCTdensity$densitySupport
densityCurves = strokeCTdensity$densityCurve

res = wass_regress(rightside_formula = ~., Xfit_df = predictor,
                    Ytype = 'density', Ymat = densityCurves, Sup = dSup)
globalF_res = globalFtest(res, alpha = 0.05, permutation = TRUE, numPermu = 200)
```
partialFtest  

**Description**

partial F test for Wasserstein regression

**Usage**

```r
partialFtest(reduced_res, full_res, alpha = 0.05)
```

**Arguments**

- `reduced_res`: a reduced model list returned by the `wass_regress` function
- `full_res`: a full model list returned by the `wass_regress` function
- `alpha`: type one error rate

**Details**

two methods used to compute p value using asymptotic distribution of F statistic

- truncated: asymptotic inference, p-value is obtained by truncating the infinite summation of eigenvalues into the first K terms, where the first K terms explain more than 99.99% of the variance.
- satterthwaite: asymptotic inference, p-value is computed using Satterthwaite approximation method of mixtures of chi-square.

**Value**

a dataframe containing the following columns:

- `method`: methods used to compute p value, see details
- `statistic`: the test statistics
- `critical_value`: critical value
- `p_value`: p value of global F test

**Examples**

```r
data(strokeCTdensity)
predictor = strokeCTdensity$predictors
dSup = strokeCTdensity$densitySupport
densityCurves = strokeCTdensity$densityCurve

full_res <- wass_regress(rightside_formula = ~., Xfit_df = predictor,
                        Ymat = densityCurves, Ytype = 'density', Sup = dSup)
reduced_res <- wass_regress(~ log_b_vol + b_shapInd + midline_shift + B_TimeCT, Xfit_df = predictor,
                          Ymat = densityCurves, Ytype = 'density', Sup = dSup)
partialFtable = partialFtest(reduced_res, full_res, alpha = 0.05)
```
Description

print the summary of WRI object

Usage

## S3 method for class "summary.WRI"
print(x, ...)

Arguments

x a 'summary.WRI' object

... further arguments passed to or from other methods.

---

Description

convert density function to quantile and quantile density function

Usage

quan2den_qd(quantileCurves, t_vec)

Arguments

quantileCurves n-by-m matrix of quantile curves

t_vec length m vector contains the common support grid of the quantile curves
simulate_quantile_curves

Simulate quantile curves

Description

This function simulates quantile curves used as a toy example

Usage

simulate_quantile_curves(x1, alpha, beta, t_vec)

Arguments

x1  
\(n\)-by-1 predictor vector

alpha  
parameter in location transformation

beta  
parameter in variance transformation

t_vec  
a length \(m\) vector - common grid for all quantile functions

Value

quan_obs \(n\)-by-\(m\) matrix of quantile functions

References

Wasserstein F-tests and confidence bands for the Frechet regression of density response curves, Alexander Petersen, Xi Liu and Afshin A. Divani, 2019

Examples

alpha = 2
beta = 1
n = 100
x1 = runif(n)
t_vec = unique(c(seq(0, 0.05, 0.001), seq(0.05, 0.95, 0.05), seq(0.95, 1, 0.001)))
quan_obs = simulate_quantile_curves(x1, alpha, beta, t_vec)
**Description**

Stroke data: clinical, radiological scalar variables and density curves of the hematoma of 393 stroke patients

**Format**

a list of the following three fields:

- **densityCurve**: 393-by-101 head CT hematoma densities as distributional response
- **densitySupport**: length 101 common support vector
- **predictors**: 393-by-9 matrix containing 9 scalar predictors

**References**

Wasserstein F-tests and confidence bands for the Frechet regression of density response curves, Alexander Petersen, Xi Liu and Afshin A. Divani, 2019

---

**summary.WRI**

**Summary Function of Wasserstein Regression Model**

**Description**

Summary Function of Wasserstein Regression Model

**Usage**

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

**Arguments**

- **object**: an object returned by the `wass_regress` function
- **...**: further arguments passed to or from other methods.
Value

a list containing the following fields:

- **call**: function call of the Wasserstein regression
- **r.square**: Wasserstein $R^2$, the Wasserstein coefficient of determination
- **global_wasserstein_F_stat**: Wasserstein global F test statistic from the Satterthwaite method
- **global_F_pvalue**: p value of global F test
- **global_wasserstein_F_df**: degrees of freedom of satterthwaite approximated sampling distribution used in global F test
- **partial_F_table**: Partial F test for individual effects

Examples

```r
data(strokeCTdensity)
predictor = strokeCTdensity$predictors
dSup = strokeCTdensity$densitySupport
densityCurves = strokeCTdensity$densityCurve

res <- wass_regress(rightside_formula = ~., Xfit_df = predictor, Ymat = densityCurves, Ytype = 'density', Sup = dSup)
summary(res)
```

---

**wass_R2**

*Compute Wasserstein Coefficient of Determination*

Description

Compute Wasserstein Coefficient of Determination

Usage

`wass_R2(wass_regress_res)`

Arguments

- `wass_regress_res`: an object returned by the `wass_regress` function

Value

Wasserstein $R^2$, the Wasserstein coefficient of determination
wass_regress

References

Frechet regression for random objects with Euclidean predictors, Alexander Petersen and Hans-Georg Müller, 2019

Examples

data(strokeCTdensity)
predictor = strokeCTdensity$predictors
dSup = strokeCTdensity$densitySupport
densityCurves = strokeCTdensity$densityCurve

res = wass_regress(rightside_formula = ~., Xfit_df = predictor,
Ymat = densityCurves, Ytype = 'density', Sup = dSup)
wass_r2 = wass_R2(res)

wass_regress

Perform Frechet Regression with the Wasserstein Distance

Description

Perform Frechet Regression with the Wasserstein Distance

Usage

wass_regress(rightside_formula, Xfit_df, Ytype, Ymat, Sup = NULL)

Arguments

rightside_formula

a right-side formula

Xfit_df

n-by-p matrix (or dataframe) of predictor values for fitting (do not include a column for the intercept)

Ytype

'quantile' or 'density'

Ymat

one of the following matrices:

• if Ytype = 'quantile' Ymat is an n-by-m matrix of the observed quantile functions. Ymat[i, :] is a 1-by-m vector of quantile function values on grid Sup.

• if Ytype = 'density' Ymat is an n-by-m matrix of the observed density functions. Ymat[i, :] is a 1-by-m vector of density function values on grid Sup.

Sup

one of the following vectors:

• if Ytype = 'quantile' Sup is a length m vector - common grid for all quantile functions in Ymat (default: seq(0, 1, length.out = ncol(Ymat))).

• if Ytype = 'density' Sup is a length m vector - common grid for all density functions in Ymat (default: seq(0, 1, length.out = ncol(Ymat))).
Value

a list containing the following objects:

call 
  function call
rformula  
  rightside_formula
predictor_names 
  names of predictors as the colnames given in the xfit matrix or dataframe.
Qfit  
  n-by-m matrix of fitted quantile functions.
xfit 
  design matrix in quantile fitting.
Xfit_df  
  n-by-p matrix (or dataframe) of predictor values for fitting
Yobs  
  a list containing the following matrices:
  • Qobs: n-by-m matrix of the observed quantile functions.
  • qobs: n-by-m matrix of the observed quantile density functions.
  • qobs_prime: n-by-m matrix of the first derivative of the observed quantile density functions.
  • fobs: n-by-m matrix of the observed density functions.
t_vec  
  a length m vector - common grid for all quantile functions in Qobs.

References

Wasserstein F-tests and confidence bands for the Frechet regression of density response curves, Alexander Petersen, Xi Liu and Afshin A. Divani, 2019

Examples

data(strokeCTdensity)
predictor = strokeCTdensity$predictors
dSup = strokeCTdensity$densitySupport
densityCurves = strokeCTdensity$densityCurve

res1 = wass_regress(rightside_formula = ~., Xfit_df = predictor, Ytype = 'density', Ymat = densityCurves, Sup = dSup)
res2 = wass_regress(rightside_formula = ~ log_b_vol * weight, Xfit_df = predictor, Ytype = 'density', Ymat = densityCurves, Sup = dSup)
Index

confidenceBands, 2

den2Q_qd, 4

globalFtest, 4

partialFtest, 6

print.summary.WRI, 7

quan2den_qd, 7

simulate_quantile_curves, 8

strokeCTdensity, 9

summary.WRI, 9

wass_R2, 10

wass_regress, 11