Package: GSSE (via r-universe)

September 13, 2024

Type Package

Title Genotype-Specific Survival Estimation

Description We propose a fully efficient sieve maximum likelihood method to estimate genotype-specific distribution of time-to-event outcomes under a nonparametric model. We can handle missing genotypes in pedigrees. We estimate the time-dependent hazard ratio between two genetic mutation groups using B-splines, while applying nonparametric maximum likelihood estimation to the reference baseline hazard function. The estimators are calculated via an expectation-maximization algorithm.

Version 0.1

Date 2015-10-17

Author Baosheng Liang, Yuanjia Wang and Donglin Zeng

Maintainer Baosheng Liang <liangbsunc@gmail.com>

Depends R(>= 3.0.1)

Suggests MASS

Imports Iso (>= 0.0-17), splines, stats, zoo (>= 1.7-12)

License GPL (>= 2)

LazyLoad yes

NeedsCompilation no

Repository CRAN

Date/Publication 2015-10-20 08:06:35

Contents

GSSE-package *Genotype-Specific Survival Estimation*

Description

The package 'GSSE' (Genotype-Specific Survival Estimation) is made to implement a fully efficient sieve maximum likelihood method to estimate genotype-specific distribution of time-to-event outcomes under a nonparametric model. The package can handle missing genotypes in pedigrees. We estimate time-dependent hazard ratio between two genetic mutation groups using B-splines and apply nonparametric maximum likelihood estimation to the reference baseline hazard function. The estimators are calculated via an expectation-maximization algorithm.

Details

Package:	GSSE
Type:	Package
Version:	0.1
Date:	2015-10-17
License:	GPL (>= 2)

The main function is `Sieve_NPMLE_Switch()`. See the documentation file with examples.

Author(s)

Baosheng Liang <liangbsunc@gmail.com>, Yuanjia Wang <yw2016@cumc.columbia.edu> and Donglin Zeng <dzeng@email.unc.edu>

Maintainer: Baosheng Liang <liangbsunc@gmail.com>

References

Wang, Y., Clark, L. N., Louis, E. D., Mejia-Santana, H., Harris, J., Cote, L. J., ... & Marder, K. (2008). Risk of Parkinson disease in carriers of parkin mutations: estimation using the kin-cohort method. *Archives of neurology*, **65**(4), 467-474.

Qin, J., Garcia, T., Ma, Y., Tang, M., Marder, K. & Wang, Y. (2014). Combining isotonic regression and EM algorithm to predict genetic risk under monotonicity constraint. *The Annals of Applied Statistics* **8**(2), 1182-1208.

Wang, Y., Liang, B., Tong, X., Marder, K., Bressman, S., Orr-Urtreger, A., Giladi, N. & Zeng, D. (2015). Efficient estimation of nonparametric genetic risk function with censored data. *Biometrika*, **102**(3), 515-532.

EM_PAVA_Func *EM-PAVA function*

Description

This function is used to estimate the genotype-specific distribution of time-to-event outcomes using EM-PAVA algorithm (Qin et al. 2014).

Usage

```
EM_PAVA_Func (q, x, delta, timeval, p, ep = 1e-05, maxiter = 400)
```

Arguments

q	matrix of 2 columns, where the first and second columns are the probabilities of belonging to the carrier p0G and non-carrier groups 1 - p0G, respectively.
x	observed event time or censoring time.
delta	indicator of event.
timeval	grid points at which the distribution function values are estimated.
p	number of groups.
ep	convergence criterion. Here, ep = 1e-5 is used as the default value.
maxiter	maximum number of EM iterations.

Details

Technical details can be found in Qin et al. (2014).

Value

Returns a list of prediction values for classes

Fest	estimated values at the points of timeval.
Fest.all	estimated values of cumulative distribution function on both carrier and non-carrier groups.

References

Qin, J., Garcia, T., Ma, Y., Tang, M., Marder, K. & Wang, Y. (2014). Combining isotonic regression and EM algorithm to predict genetic risk under monotonicity constraint. *The Annals of Applied Statistics* **8**(2), 1182-1208.

See Also

p0G_Func(), Sieve_NPMLE_Switch()

Examples

```
data("Simulated_data");

OY = Simulated_data[,2];
ind = order(OY);
ODelta = Simulated_data[,3];
Op0G = Simulated_data[,4];

Y = OY[ind];
Delta = ODelta[ind];
p0G = Op0G[ind];

Grid = seq(0.01, 3.65, 0.01);
fix_t1 = c(0.288, 0.693, 1.390);
fix_t2 = c(0.779, 1.860, 3.650);

EMpava_result = EM_PAVA_Func ( q = rbind(p0G,1-p0G), x = Y, delta = Delta,
                               timeval = Grid, p = 2, ep = 1e-4 );

all = sort(c(Grid, Y));

F_carr_func = function(x){  EMpava_result$Fest.all[1, which.max(all[all <= x]) ]  };
F_non_func  = function(x){  EMpava_result$Fest.all[2, which.max(all[all <= x]) ]  };

PAVA_F1.hat_fix_t = apply( matrix(fix_t1, ncol=1), 1, F_carr_func );
PAVA_F2.hat_fix_t = apply( matrix(fix_t2, ncol=1), 1, F_non_func );

PAVA_F.hat_fix_t = data.frame( fix_t1 = fix_t1, PAVA_F1.hat = PAVA_F1.hat_fix_t,
                               fix_t2 = fix_t2, PAVA_F2.hat = PAVA_F2.hat_fix_t  );

print(PAVA_F.hat_fix_t);

# plot estimated curves

F_carr = apply( matrix(Grid, ncol=1), 1, F_carr_func );
F_non = apply( matrix(Grid, ncol=1), 1, F_non_func );

plot( Grid, F_carr, type = 's', lty = 1,
      xlab = "Y", ylab = "Estimated Cumulative Distribution Function",
      ylim = c(0,1), col = 'blue' );
lines(Grid, F_non, type='s', lty=2, col='red');
legend("topleft", legend=c("Carrier group", "Non-Carrier group"),
       lty=c(1,2), col=c("blue", "red") );
```

---

p0G_data *Data Set for Illustration of the 'p0G' Calculation*

Description

This is a small data set used for illustration of the 'p0G' calculation. There are 2 columns, 20 rows in this data set.

Usage

```
data("p0G_data")
```

Format

A data frame with 20 observations on the following 2 columns with

`relative` a factor with levels `Child`, `Parent` and `Sibling`,

`proband_pd` a factor with levels `No` and `Yes`.

Source

This data set is a tiny subset of the data set of a Parkinson disease research project supported by Michael J. Fox Foundation.

References

Wang, Y., Clark, L. N., Louis, E. D., Mejia-Santana, H., Harris, J., Cote, L. J., ... & Marder, K. (2008). Risk of Parkinson disease in carriers of parkin mutations: estimation using the kin-cohort method. *Archives of neurology*, **65**(4), 467-474.

Wang, Y., Liang, B., Tong, X., Marder, K., Bressman, S., Orr-Urtreger, A., Giladi, N. & Zeng, D. (2015). Efficient estimation of nonparametric genetic risk function with censored data. *Biometrika*, **102**(3), 515-532.

See Also

`p0G_Func()`

Examples

```
data("p0G_data")
```

`p0G_Func` *Probability Calculation of Relative's Mutation Status*

Description

This function is used to calculate the probability of a relative being a carrier.

Usage

```
p0G_Func (p, status, relative, model)
```

Arguments

p	population frequency of mutation.
status	proband's carrier status (status = 0, non-carrier; = 1, homozygous; = 2, heterozygous).
relative	relative's relationship to the proband (relative = 1, parents; = 2, sibling; = 3, offspring).
model	assumed genetic model ("dominant" or "recessive").

Details

Technical details can be found in Wang et al. (2008).

Value

The function p0G_Func returns the probability of a relative being a carrier under the given genetic model.

References

Wang, Y., Clark, L. N., Louis, E. D., Mejia-Santana, H., Harris, J., Cote, L. J., ... & Marder, K. (2008). Risk of Parkinson disease in carriers of parkin mutations: estimation using the kin-cohort method. *Archives of neurology*, **65**(4), 467-474.

Wang, Y., Liang, B., Tong, X., Marder, K., Bressman, S., Orr-Urtreger, A., Giladi, N. & Zeng, D. (2015). Efficient estimation of nonparametric genetic risk function with censored data. *Biometrika*, **102**(3), 515-532.

See Also

p0G_data, Sieve_NPMLE_Switch(), Sieve_NPMLE_Bootstrap(), EM_PAVA_Func() and Permutation_Test().

Examples

```
#  Example 1  #

 data("p0G_data");
 attach(p0G_data);

 n = dim(p0G_data)[1];
 Status = as.numeric( proband_pd == "Yes" );
 Relative = rep(1, n) + 1*(relative == "Sibling") + 2*(relative == "Child");

 detach(p0G_data);

 p0G = rep(0, n);

 for(i in 1:n)
 {
  p0G[i] = p0G_Func(p = 0.02, status = Status[i], relative = Relative[i], model="dominant");
```

```
 }

 data.frame(pd_status = Status, relatives = Relative, prob = p0G);

#  Example 2  #

 n = 50;
 status = sample(x=c(0,1), size = n, replace = TRUE, prob = c(0.6, 0.4) );
 relative = sample(x=1:3, size = n, replace = TRUE, prob = c(1/3, 1/3, 1/3) );

 p0G = rep(0, n);

 for(i in 1:n)
 {
  p0G[i] = p0G_Func(p = 0.2, status = status[i], relative = relative[i], model="dominant");
 }

 data.frame(status = status, relative = relative, p0G = p0G);
```

`Permutation_Test` *Permutation Test*

Description

Permutation test for the Sieve-NPMLE switch method with null hypothesis H_0: F_carr = F_non and alternative hypothesis H_1: F_carr is not equal to F_non.

Usage

```
Permutation_Test (Grid, F_carr, F_non, OY, ODelta, Op0G, nperm)
```

Arguments

`Grid`	time points at which the distribution function values are estimated.
`F_carr`	a vector of distribution function values at given grid points of the carrier group.
`F_non`	a vector of distribution function values at given grid points of the non-carrier group.
`OY`	observed event times.
`ODelta`	observed indicators of right censoring.
`Op0G`	observed probability values of carrier and non-carrier groups.
`nperm`	replication number used in permutation.

Details

Technical details can be found in Wang et al. (2015).

Value

This function returns a list of prediction values for classes,

`Test_Stat`	value of the Kolmogorov-Smirnov statistic with observed data.
`Pvalues`	p-value of the permutation test.
`Permutation.value`	values of Kolmogorov-Smirnov statistics under all permutations.

References

Wang, Y., Liang, B., Tong, X., Marder, K., Bressman, S., Orr-Urtreger, A., Giladi, N. & Zeng, D. (2015). Efficient estimation of nonparametric genetic risk function with censored data. *Biometrika*, **102**(3), 515-532.

See Also

`test_stat()` and `Sieve_NPMLE_Switch()`.

Examples

```
data("Simulated_data");

OY = Simulated_data[,2];
ind = order(OY);
ODelta = Simulated_data[,3];
Op0G = Simulated_data[,4];

Y = OY[ind];
Delta = ODelta[ind];
p0G = Op0G[ind];

Grid = seq(0.2, 3.65, 0.05);
fix_t1 = c(0.288, 0.693, 1.390);
fix_t2 = c(0.779, 1.860, 3.650);
px = seq(0.1, 3, 0.1);

SieveNPMLE_result = Sieve_NPMLE_Switch( Y=Y, Delta=Delta, p0G=p0G,
                                        px=px, Grid=Grid, Knot=7,
                                        degree=3 );

Lambda_1.hat = cumsum( SieveNPMLE_result$lamb1.hat );
Lambda_2.hat = cumsum( SieveNPMLE_result$lamb2.hat );

F_carr_func = function(x){ 1 - exp( max( Lambda_1.hat[Y <= x] ) ) }
F_non_func  = function(x){ 1 - exp( max( Lambda_2.hat[Y <= x] ) ) }

F_carr = apply( matrix(px, ncol=1), 1, F_carr_func );
F_non = apply( matrix(px, ncol=1), 1, F_non_func );

# Permutation test #
```

```
Permutation_Test( Grid=Grid, F_carr=F_carr, F_non=F_non,
                  OY=OY, ODelta=ODelta, Op0G=Op0G,
                  nperm=10 );
```

`Sieve_NPMLE_Bootstrap` *Sieve_NPMLE_Bootstrap function*

Description

This function is used for calculating standard error estimates and 95% confidence bands in quantile using the bootstrap method.

Usage

```
Sieve_NPMLE_Bootstrap ( fam_ID, Y0, Delta0, p0G0, fix_t1, fix_t2,
                        Grid, Knot, degree=3, Bn, maxiter=400, ep=1e-05)
```

Arguments

`fam_ID`	family ID numbers.
`Y0`	observed event times or censoring times.
`Delta0`	indicators of event.
`p0G0`	probabilities of being a carrier.
`fix_t1`	a vector of fixed points at which the carrier's cumulative distribution function values are estimated.
`fix_t2`	a vector of fixed points at which the non-carrier's cumulative distribution function values are estimated.
`Grid`	a vector of grid points used for plotting the estimated distribution functions of carrier and non-carrier groups.
`Knot`	number of knots of the B-spline base functions.
`degree`	degree of the B-spline base functions.
`Bn`	number of bootstrap samples.
`maxiter`	maximum number of iterations.
`ep`	convergence criterion, default is ep= 1e-05.

Details

Using bootstrap for standard error estimation and 95% confidence bands calculation. We do the Bootstrap resample according to `fam_ID`.

Value

This function returns a list

`Boot.L1`	estimated cumulative hazard function for the carrier group.
`Boot.L2`	estimated cumulative hazard function for the non-carrier group.
`SE_F1_fix_t`	estimated standard errors for the carrier group at given points `fix_t1`.
`SE_F2_fix_t`	estimated standard errors for the non-carrier group at given points `fix_t2`.

References

Wang, Y., Liang, B., Tong, X., Marder, K., Bressman, S., Orr-Urtreger, A., Giladi, N. & Zeng, D. (2015). Efficient estimation of nonparametric genetic risk function with censored data. *Biometrika*, **102**(3), 515-532.

See Also

`p0G_Func()`, `Sieve_NPMLE_Switch()`.

Examples

```
data("Simulated_data");

OID = Simulated_data[,1];
OY = Simulated_data[,2];
ind = order(OY);
ODelta = Simulated_data[,3];
Op0G = Simulated_data[,4];

Y = OY[ind];
Delta = ODelta[ind];
p0G = Op0G[ind];

Grid = seq(0.2, 3.65, 0.05);
fix_t1 = c(0.288, 0.693, 1.390);
fix_t2 = c(0.779, 1.860, 3.650);
px = seq(0.1, 3, 0.1);

SieveNPMLE_result = Sieve_NPMLE_Switch( Y=Y, Delta=Delta, p0G=p0G, px=px,
                                        Grid=Grid, Knot=7, degree=3  );

Lambda_1.hat = cumsum( SieveNPMLE_result$lamb1.hat );
Lambda_2.hat = cumsum( SieveNPMLE_result$lamb2.hat );

F_carr_func = function(x){ 1 - exp( - max( Lambda_1.hat[Y <= x] ) ) }
F_non_func  = function(x){ 1 - exp( - max( Lambda_2.hat[Y <= x] ) ) }

est.f1 = apply(matrix(fix_t1, ncol=1), 1, F_carr_func );
est.f2 = apply(matrix(fix_t2, ncol=1), 1, F_non_func  );

# ---------------- #
#    Bootstrap     #
```

```
# ---------------- #

 Boot = Sieve_NPMLE_Bootstrap( fam_ID=OID, Y0=OY, Delta0=ODelta, p0G0=Op0G,
                               fix_t1=fix_t1, fix_t2=fix_t2, Grid = Grid,
                               Knot=6, degree =3, Bn=10  );

 SE1 = Boot$SE_F1_fix_t;
 SE2 = Boot$SE_F2_fix_t;

 estp = data.frame( fix_t1 = fix_t1, F1.hat = est.f1, SE_F1 = SE1,
                    fix_t2 = fix_t2, F2.hat = est.f2, SE_F2 = SE2  );

 print(estp)
```

`Sieve_NPMLE_Switch` *Sieve_NPMLE_Switch function*

Description

This function is used to estimate the genotype-specific distribution of time-to-event outcomes with the Sieve-NPMLE switch algorithm.

Usage

```
Sieve_NPMLE_Switch (Y, p0G, Delta, px, Grid, Knot, degree, maxiter=400, ep=1e-05)
```

Arguments

`Y`	observed event times or censoring times.
`p0G`	probabilities of being a carrier.
`Delta`	indicators of event.
`px`	grid points at which the distribution function values are estimated.
`Grid`	grid points used for plots. `Grid` could be either exactly the same as `px` or different from `px`.
`Knot`	number of knots.
`degree`	degree of the B-spline base functions.
`maxiter`	maximum number of iterations.
`ep`	convergence criterion, default is `ep= 1e-05`.

Details

This function is used to estimate the distribution of the age-at-onset of Parkinsons disease for carriers' and non-carriers' mutations in the leucine-rich repeat kinase 2, LRRK2, gene. Please refer to Wang et al. (2015) for more details.

Value

This function returns a list of prediction values for classes

`lamb1.hat`	estimated values of hazard function for the carrier group.
`lamb2.hat`	estimated values of hazard function for the non-carrier group.
`Lamb1`	estimated values of cumulative hazard function for the carrier group.
`Lamb2`	estimated values of cumulative hazard function for the non-carrier group.
`Converge`	convergence status.

References

Wang, Y., Liang, B., Tong, X., Marder, K., Bressman, S., Orr-Urtreger, A., Giladi, N. & Zeng, D. (2015). Efficient estimation of nonparametric genetic risk function with censored data. *Biometrika*, **102**(3), 515-532.

See Also

`p0G_Func()`, `Sieve_NPMLE_Bootstrap()` and `Permutation_Test()`.

Examples

```
data("Simulated_data");

OY = Simulated_data[,2];
ind = order(OY);
ODelta = Simulated_data[,3];
Op0G = Simulated_data[,4];

Y = OY[ind];
Delta = ODelta[ind];
p0G = Op0G[ind];

Grid = seq(0.2, 3.65, 0.05);
fix_t1 = c(0.288, 0.693, 1.390);
fix_t2 = c(0.779, 1.860, 3.650);
px = seq(0.01, 3.65, 0.01);

SieveNPMLE_result = Sieve_NPMLE_Switch( Y=Y, Delta=Delta, p0G=p0G,
                                        px=px, Grid=Grid,
                                        Knot=6, degree=3  );

# 'Sieve_NPMLE_Switch' only returns values of hazard function
# or cumulative hazard function of mutation groups, in practice, we only
# need to do a little bit more as follows for calculating the
# cumulative distribution values of mutation groups.

Lambda_1.hat = cumsum( SieveNPMLE_result$lamb1.hat );
Lambda_2.hat = cumsum( SieveNPMLE_result$lamb2.hat );

F_carr_func = function(x){ 1 - exp(- max( Lambda_1.hat[Y <= x] ) ) }
```

```
F_non_func  = function(x){ 1 - exp(- max( Lambda_2.hat[Y <= x] ) ) }

est.f1 = apply(matrix(fix_t1, ncol=1), 1, F_carr_func );
est.f2 = apply(matrix(fix_t2, ncol=1), 1, F_non_func  );

estp = data.frame( fix_t1 = fix_t1, F_carr_t1 = est.f1,
                   fix_t2 = fix_t2, F_non_t2 = est.f2  );

print(estp);

# plot estimated curves

F_carr = apply( matrix(px, ncol=1), 1, F_carr_func );
F_non = apply( matrix(px, ncol=1), 1, F_non_func );

plot( px, F_carr, type='s', lty=1, ylim=c(0, 1), xlab="Y",
      ylab="Estimated Cumulative Distribution Function", col='blue' );
lines(px, F_non, type='s', lty=2, col='red');
legend("topleft", legend=c("Carrier group", "Non-Carrier group"),
       lty=c(1,2), col=c("blue","red") );
```

`Simulated_data` *Simulated Parkinson's disease data*

Description

This is an artificial data set used for the illustration of the current R-package. There are 4 columns, 268 rows in this data set.

Usage

```
data("Simulated_data")
```

Format

A data frame with 268 observations on the following 4 columns with.

`FamID` a factor with levels `Child`, `Parent` and `Sibling`

`Y` a factor with levels `No` and `Yes`

`delta` a numeric vector

`p0G` a numeric vector, probability distribution value.

Source

This is a simulated data set based on the real data analysis results in Wang et al. (2015).

References

Wang, Y., Liang, B., Tong, X., Marder, K., Bressman, S., Orr-Urtreger, A., Giladi, N. & Zeng, D. (2015). Efficient estimation of nonparametric genetic risk function with censored data. *Biometrika*, **102**(3), 515-532.

Marder, K., Wang, Y., Alcalay, R. N., Mejia-Santana, H., Tang, M. X., Lee, A., ... & Miravite, J. (2015). Age-specific penetrance of LRRK2 G2019S in the Michael J. Fox Ashkenazi Jewish LRRK2 Consortium. *Neurology*, **85**(1), 89–95.

Qin, J., Garcia, T., Ma, Y., Tang, M., Marder, K. & Wang, Y. (2014). Combining isotonic regression and EM algorithm to predict genetic risk under monotonicity constraint. *The Annals of Applied Statistics* **8**(2), 1182-1208.

See Also

`Sieve_NPMLE_Switch()`, `Sieve_NPMLE_Bootstrap()`, `EM_PAVA_Func()`.

Examples

```
data("Simulated_data")
```

---

test_stat *Kolmogorov-Smirnov Test Statistic*

Description

This function is used to calculate the Kolmogorov-Smirnov statistic: $D(F_1, F_2) = \sup_x |F_1(x) - F_2(x)|$.

Usage

```
test_stat (F1, F2)
```

Arguments

`F1`	vector of one distribution function value.
`F2`	vector of the other distribution function value.

Details

It is necessary to point out that `F1` and `F2` are based on common grids. Other technical details can be found in Wang et al. (2015).

Value

This function returns a list of prediction values for classes,

`out`	value of Kolmogorov-Smirnov statistics.

References

Wang, Y., Liang, B., Tong, X., Marder, K., Bressman, S., Orr-Urtreger, A., Giladi, N. & Zeng, D. (2015). Efficient estimation of nonparametric genetic risk function with censored data. *Biometrika*, **102**(3), 515-532.

See Also

`Permutation_Test()`

Examples

```
X = seq(0.01, 1, 0.05);
Func_1 = function(x){ x + runif(length(x), min=0, max=1) };
Func_2 = function(x){ x + 2*runif(length(x), min=0, max=1) };

U1 = Func_1(X);
U2 = Func_2(X);

test_stat (F1 = U1, F2 = U2);
```

Index