GeneGeneInteR vignette
Statistical analysis of the interaction between a pair of genes.

Mathieu Emily and Magalie Houée-Bigot

October 17, 2016

This vignette presents the technical details of the statistical procedure implemented in the package. Readers that would like to have a global overview of the main functions and tools proposed in the package are encouraged to read the vignette **VignetteGeneGeneInteR_Introduction**.

1 Introduction

In this vignette we consider statistical procedures to test for the interaction between two genes in susceptibility with a binary phenotype, typically a case/control disease status. Let $Y \in \{0, 1\}$ be the phenotype, where $Y = 0$ stands for a control and $Y = 1$ a case, and X_1 and X_2 be the two genes for which the interaction is tested.

Let consider a sample of n individuals with n_c controls and n_d cases ($n_c + n_d = n$) and $\mathbf{Y} = [y_1, \ldots, y_n]'$ the vector of the observed binary phenotypes. Each gene is a collection of respectively m_1 and m_2 SNPs. The observed genotypes for gene X_1 can be represented by a $n \times m_1$ matrix: $\mathbf{X_1} = [x^1_{ij}]_{i\in 1\ldots n; j\in 1\ldots m_1}$ where $x^1_{ij} \in \{0; 1; 2\}$ is the number of copies of the minor allele for SNP j carried by individual i. A similar representation is used for gene X_2 where $\mathbf{X_2}$ is a $n \times m_2$ matrix. Let us further introduce $\mathbf{X_1^c}$ and $\mathbf{X_2^c}$ the matrices of observed genotypes among controls for gene 1 and 2 and $\mathbf{X_1^d}$ and $\mathbf{X_2^d}$ among cases for both genes. Thus $\mathbf{X_1^c}$ is a $n_c \times m_1$ matrix, $\mathbf{X_1^d}$ a $n_d \times m_1$ matrix, $\mathbf{X_2^c}$ a $n_c \times m_2$ matrix and $\mathbf{X_2^d}$ a $n_d \times m_2$ matrix. A general setup of the observed values can be presented as follows:

$$\mathbf{Y} = \begin{bmatrix} y_1 \\ \vdots \\ y_{n_c} \\ y_{n_c+1} \\ \vdots \\ y_{n_c+n_d} \end{bmatrix} \mathbf{X_1} = \begin{bmatrix} \mathbf{X_1^c} \\ \\ \mathbf{X_1^d} \end{bmatrix} = \begin{bmatrix} x^1_{11} & \cdots & x^1_{1m_1} \\ \vdots & \ddots & \vdots \\ x^1_{n_c1} & \cdots & x^1_{n_cm_1} \\ x^1_{(n_c+1)1} & \cdots & x^1_{(n_c+1)m_1} \\ \vdots & \ddots & \vdots \\ x^1_{(n_c+n_d)1} & \cdots & x^1_{(n_c+n_d)m_1} \end{bmatrix} \mathbf{X_2} = \begin{bmatrix} \mathbf{X_2^c} \\ \\ \mathbf{X_2^d} \end{bmatrix} = \begin{bmatrix} x^2_{11} & \cdots & x^2_{1m_2} \\ \vdots & \ddots & \vdots \\ x^2_{n_c1} & \cdots & x^2_{n_cm_1} \\ x^2_{(n_c+1)1} & \cdots & x^2_{(n_c+1)m_1} \\ \vdots & \ddots & \vdots \\ x^2_{(n_c+n_d)1} & \cdots & x^2_{(n_c+n_d)m_1} \end{bmatrix}$$

In our package we proposed 10 methods for testing interaction at the gene level. These 10 methods are all based on three main parameters: `Y`, a `numeric` or `factor` vector with exactly two distinct values, `G1` and `G2` two `SnpMatrix` objects as proposed by the `R` **Bioconductor** package **snpStats**. Our implementation is illustrated by the dataset `gene.pair` provided with the **GeneGeneInteR** package and summarized in the following command lines:

```
> library(GeneGeneInteR)
> data(gene.pair)
> head(gene.pair$Y)
```

```
 [1] HealthControl HealthControl HealthControl HealthControl HealthControl HealthControl
 Levels:  HealthControl RheumatoidArthritis
```

```
> gene.pair$G1
```

```
A SnpMatrix with 453 rows and 8 columns
Row names:  Id1 ...  Id453
Col names:  rs1491710 ...  rs2298849
```

```
> gene.pair$G2

   A SnpMatrix with 453 rows and 4 columns
   Row names:  Id1 ...  Id453
   Col names:  rs2057094 ...  rs1005753
```

The 10 methods implemented in our package can be divided into two main families: 6 methods based on a multidimensional modeling of the interaction at the gene level and 4 methods that combine interaction tests at the SNP level into a single test at the gene level.

2 Multidimensional methods at the gene level

In the **GeneGeneInteR** package, 6 multidimensional methods have been implemented that are based on:

- Principal Components Analysis - `PCA.test` function,
- Canonical Correlation Analysis - `CCA.test` function,
- Kernel Canonical Correlation Analysis - `KCCA.test` function,
- Composite Linkage Disequilibrium - `CLD.test` function,
- Partial Least Square Path Modeling - `PLSPM.test` function,
- Gene-Based Information Gain Method - `GBIGM.test` function.

In the remainder of this section, technical and practical details are given regarding these 6 methods.

2.1 PCA-based

In the PCA-based method, a likelihood ratio test is performed to compare the model $\mathcal{M}_{Inter}$ to the model $\mathcal{M}_{No}$, where $\mathcal{M}_{Inter}$ is defined by:

$$\text{logit}\left(\mathbb{P}\left[Y=1|PC^1_{X_1}\dots PC^{n_1}_{X_1}, PC^1_{X_2}\dots PC^{n_2}_{X_2}\right]\right) = \beta_0 + \sum_{i=1}^{n_1} PC^i_{X_1} + \sum_{j=1}^{n_2} PC^j_{X_2} + \sum_{i=1}^{n_1}\sum_{i=2}^{n_2} PC^i_{X_1} PC^j_{X_2}$$

and $\mathcal{M}_{No}$ by:

$$\text{logit}\left(\mathbb{P}\left[Y=1|PC^1_{X_1}\dots PC^{n_1}_{X_1}, PC^1_{X_2}\dots PC^{n_2}_{X_2}\right]\right) = \beta_0 + \sum_{i=1}^{n_1} PC^i_{X_1} + \sum_{j=1}^{n_2} PC^j_{X_2}$$

In models $\mathcal{M}_{Inter}$ and $\mathcal{M}_{No}$, $PC^i_{X_1}$ and $PC^j_{X_2}$ are the i^{th} principal component of $\mathbf{X_1}$ and the j^{th} principal component of $\mathbf{X_2}$. The number of principal components, n_1 and n_2, kept in the interaction test is determined by the percentage of inertia retrieved by the PCA. Such a percentage is defined by the user and corresponds to the `threshold` parameter.

In our package, two distinct Principal Component decomposition are provided by the functions `PCA.test` via the argument `method`. With `method="Std"`, dataset is standardized using variables' standard deviation while with `method="GenFreq"`, dataset is standardized using standard deviation under Hardy-Weinberg equilibrium, as proposed in the **snpStats** package.

When the percentage of inertia asked by the user is high, the number of PCs can be important and fitting logistic models $\mathcal{M}_{Inter}$ and $\mathcal{M}_{No}$ is likely to fail. In that case, the number of PCs in each gene is iteratively reduced until convergence of the `glm` function for fitting models $\mathcal{M}_{Inter}$ and $\mathcal{M}_{No}$.

The following lines provide an example of the PCA.test function:

```
> PCA.test(Y=gene.pair$Y, G1=gene.pair$G1, G2=gene.pair$G2,threshold=0.7,
+ method="GenFreq")

   Gene-Gene Interaction method performed with:
              Principal Component Analysis
   Deviance = 8.215662, df = 6, p-value = 0.2227253
```

```
> PCA.test(Y=gene.pair$Y, G1=gene.pair$G1, G2=gene.pair$G2,threshold=0.7,
+ method="Std")

   Gene-Gene Interaction method performed with:
              Principal Component Analysis
   Deviance = 8.507404, df = 6, p-value = 0.2032347
```

2.2 Canonical Correlation Analysis (CCA)

The CCA test is based on a Wald-type statistic defined as follows (see [?] for details):

$$U_{CCA} = \frac{z_d - z_c}{\sqrt{\mathbb{V}(z_d) + \mathbb{V}(z_c)}}$$

where $z_d = \frac{1}{2}\left(\log(1+r_d) - \log(1-r_d)\right)$ and $z_c = \frac{1}{2}\left(\log(1+r_c) - \log(1-r_c)\right)$ with r_d the maximum canonical correlation coefficient between $\mathbf{X_1^d}$ and $\mathbf{X_2^d}$ and r_c the maximum canonical correlation coefficient between $\mathbf{X_1^c}$ and $\mathbf{X_2^c}$ computed for controls ($Y = 0$). As suggested by [?], the sampled variances $\mathbb{V}(z_d)$ and $\mathbb{V}(z_c)$ were evaluated by applying a bootstrapping method. The number of bootstrap sample used to estimate $\mathbb{V}(z_d)$ and $\mathbb{V}(z_c)$ is determined by the n.boot argument. P-value is then obtained by noting that under the null hypothesis $U_{CCA} \sim \mathcal{N}(0,1)$.

CCA based gene-gene interaction is implemented in the CCA.test function and mainly depends on the cancor function from the **Stats** package [?].

```
   R> set.seed(1234)
```

```
> CCA.test(Y=gene.pair$Y, G1=gene.pair$G1, G2=gene.pair$G2,n.boot=500)

   Gene-Gene Interaction method performed with:
              Canonical Correlation Analysis
   CCU = 0.6030414, p-value = 0.5464811
```

2.3 Kernel Canonical Correlation Analysis (KCCA)

The KCCA based test provides a generalization of CCA test to detect non-linear co-association between $\mathbf{X_1}$ and $\mathbf{X_2}$ [?, ?] and is based on the following Wald-type statistic:

$$U_{KCCA} = \frac{kz_d - kz_c}{\sqrt{\mathbb{V}(kz_d) + \mathbb{V}(kz_c)}}$$

where $kz_d = \frac{1}{2}\left(\log(1+kr_d) - \log(1-kr_d)\right)$ and $kz_c = \frac{1}{2}\left(\log(1+kr_c) - \log(1-kr_c)\right)$ with kr_d the maximum kernel canonical correlation coefficient between $\mathbf{X_1^d}$ and $\mathbf{X_2^d}$ and kr_c the maximum kernel canonical correlation coefficient between $\mathbf{X_1^c}$ and $\mathbf{X_2^c}$.

Similar to the CCA test, $\mathbb{V}(kz_d)$ and $\mathbb{V}(kz_c)$ are estimated using bootstrap techniques [?, ?] and the p-value is obtained using the standard gaussian distribution of U_{KCCA} under the null hypothesis. Since the performance of kernel methods strongly relates to the choice of kernel functions, the default is the Radial Basis kernel Function (RBF) owing to its flexibility in parameter specification. However, other kernel functions, such as linear, polynomial or spline kernels, can be used. Thus, in addition to the three arguments Y, G1 and G2, our implementation of the KCCA test proposes two optional arguments: n.boot that determines the number of bootstrap samples and `kernel` that provides the kernel function to be used. This `kernel` parameter is character string matching one of the kernel name provided by the **kernlab** package [?] such as ”rbfdot”, ”polydot”, ”tanhdot”, ”vanilladot”, ”laplacedot”, ”besseldot”, ”anovadot”, ”splinedot”. Specific arguments, `sigma`, `degree`, `scale`, `offset`and `order`, can also be passed to the `kcca.test` function in order to parameterized the kernel used in the analysis.

KCCA based gene-gene interaction test is implemented in the `KCCA.test` function and mainly depends on the `kcca` function from the **kernlab** package [?].

```
> set.seed(1234)
> KCCA.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,
+ kernel="rbfdot",sigma = 0.05,n.boot=500)

   Gene-Gene Interaction method performed with:
              Kernel Canonical Correlation Analysis
   KCCU = 1.407369, p-value = 0.1593179

> set.seed(1234)
> KCCA.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,
+ kernel="polydot",degree = 1, scale = 1, offset = 1)

   Gene-Gene Interaction method performed with:
              Kernel Canonical Correlation Analysis
   KCCU = -1.413508, p-value = 0.1575063
```

2.4 Partial Least Square Path Modeling (PLSPM)

The PLSPM testing has been introduced by [?] and is based on the Wald-like statistic:

$$U_{PLSPM} = \frac{\beta_d - \beta_c}{\sqrt{\mathbb{V}(\beta_d - \beta_c)}}$$

where β_d (resp. β_c) is the path coefficient between $\mathbf{X_1^d}$ and $\mathbf{X_2^d}$ (resp. $\mathbf{X_1^c}$ and $\mathbf{X_2^c}$). As quoted by [?], the distribution of U_{PLSPM} is unknown and significance can be tested with bootstrapping method.

PLSPM based gene-gene interaction test is implemented in the `PLSPM.test` function and mainly depends on the `plspm` function from the **plspm** package [?].

```
> set.seed(1234)
> PLSPM.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,n.perm=1000)


   Gene-Gene Interaction method performed with:
              Partial Least Squares Path Modeling
   U = 4.093781, p-value = 0.18
```

2.5 Composite Linkage Disequilibrium (CLD)

The CLD method, proposed in [?] is based on the normalized quadratic distance (NQD) and is defined as

$$\delta^2 = \text{tr.}\Big((\tilde{D} - \tilde{C})W^{-1}(\tilde{D} - \tilde{C})W^{-1}\Big)$$

where $\tilde{D}$, $\tilde{C}$ and W are three $(m_1 + m_2) \times (m_1 + m_2)$ matrices of the covariance between the whole set of SNPs that combines SNPs from both genes. More precisely, $\tilde{D}$ and $\tilde{C}$ are defined as follows:

$$\tilde{D} = \begin{bmatrix} W_{11} & D_{12} \\ D_{21} & W_{22} \end{bmatrix} \qquad \tilde{C} = \begin{bmatrix} W_{11} & C_{12} \\ C_{21} & W_{22} \end{bmatrix}$$

where W_{11} (resp. W_{22}) is the pooled estimate of the covariance matrix for $\mathbf{X_1}$ (resp. $\mathbf{X_2}$, $D_{12}(=D'_{21})$ and $C_{12}(=C'_{21})$ are the sample covariance matrix between the two genes estimated from $\left(\mathbf{X_1^d},\mathbf{X_2^d}\right)$ and $(\mathbf{X_1^c},\mathbf{X_2^c})$ respectively. In more details, the sample covariance matrices in cases, denoted by D, and in controls, denoted by C, can be partitioned in 4 blocks as follows:

$$D = \text{Cov}\left(\mathbf{X_1^d},\mathbf{X_2^d}\right) = \begin{bmatrix} D_{11} & D_{12} \\ D_{21} & D_{22} \end{bmatrix} \qquad C = \text{Cov}\left(\mathbf{X_1^c},\mathbf{X_2^c}\right) = \begin{bmatrix} C_{11} & C_{12} \\ C_{21} & C_{22} \end{bmatrix}$$

The pooled estimate of the covariance matrix, W, can thus been obtained by:

$$W = \frac{n_c C + n_d D}{n_c + n_d} = \begin{bmatrix} W_{11} & W_{12} \\ W_{21} & W_{22} \end{bmatrix}$$

Since the distribution of δ^2 is not known under the null hypothesis, significance testing is performed using permutation tests, as proposed by [?]. Such a test has been implemented in our package in the `CLD.test` function where the number of permutations is determined by the argument `n.perm`.

```
> set.seed(1234)
> CLD.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,n.perm=2000)

	Gene-Gene Interaction method performed with:
	          Composite Linkage Disequilibrium
	CLD = 0.4925654, p-value = 0.8865
```

2.6 Gene-Based Information Gain Method (GBIGM)

Introduced by [?], the GBIGM method is based on the information gain rate $\Delta R_{1,2}$. $\Delta R_{1,2}$ is defined as follows:

$$\Delta R_{1,2} = \frac{\min(H_1,H_2) - H_{1,2}}{\min(H_1,H_2)}$$

where H_1, H_2, $H_{1,2}$ are the conditional entropies, given the $\mathbf{Y}$, of $\mathbf{X_1}$, $\mathbf{X_2}$ and the pooled SNP set $(\mathbf{X_1},\mathbf{X_2})$ respectively. Assuming that $H(.)$ is the classical entropy function, we have:

$$\begin{aligned} H_1 &= H(\mathbf{Y},\mathbf{X_1}) - H(\mathbf{X_1}) \\ H_2 &= H(\mathbf{Y},\mathbf{X_2}) - H(\mathbf{X_2}) \\ H_{1,2} &= H(\mathbf{Y},\mathbf{X_1},\mathbf{X_2}) - H(\mathbf{X_1},\mathbf{X_2}) \end{aligned}$$

Since the distribution of $\Delta R_{1,2}$ is unknown, the significance testing is performed by permutations as suggested by [?]. The GBIGM method has been implemented in the `GBIGM.test` function and the number of permutations is defined by the argument `n.perm`.

```
> set.seed(1234)
> GBIGM.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,n.perm=2000)

	Gene-Gene Interaction method performed with:
	          Gene-based Information Gain Method
	DeltaR1,2 = 0.4644093, p-value = 0.441
```

3 From SNP-SNP interaction to Gene-Gene interaction testing

This section provides details of the four statistical methods that proposes a gene-based test from SNP-based tests [?]. Rather than considering multiple SNPs in both gene as part of a joint model, these methods aim at aggregating p-values obtained at the SNP level into a single p-value at a gene level.

Interaction testing at the SNP level
Let consider a pair of SNPs, $(X_{1,j}, X_{2,k})$ where $X_{1,j}$ is the j^{th} SNP of gene X_1 and $X_{2,k}$ the k^{th} SNP of gene X_2 $(1 \leq j \leq m_1$ and $1 \leq k \leq m_2)$. To test for interaction at the SNP level, we used the following Wald statistic:

$$W_{jk} = \frac{\widehat{\beta_3^{j,k}}}{\widehat{\sigma\left(\widehat{\beta_3^{j,k}}\right)}}$$

where $\widehat{\beta_3^{j,k}}$ is an estimate of the interaction coefficient $\beta_3^{j,k}$ of the following logistic model:

$$\log\left(\frac{\mathbb{P}[Y=1|X_{1,j}=x_1, X_{2,k}=x_2]}{1-\mathbb{P}[Y=1|X_{1,j}=x_1, X_{2,k}=x_2]}\right) = \beta_0^{j,k} + \beta_1^{j,k}x_1 + \beta_2^{j,k}x_2 + \beta_3^{j,k}x_1x_2$$

$\widehat{\beta_3^{j,k}}$ is obtained by maximizing the likelihood function on the observed data $\mathbf{Y}$, $\mathbf{X_1}$ and $\mathbf{X_2}$ while $\widehat{\sigma\left(\widehat{\beta_3^{j,k}}\right)}$ is calculating by inverting the Hessian of the likelihood. Since the solution of the maximization of the likelihood function does not have a closed form, we compute W_{jk} according to the iteratively reweighted least squares algorithm proposed in the `glm` function of the **stats** package [?] .

To combine the statistics W_{jk} into a single test, [?] proposed four methods that all account for covariance matrix $\Sigma = [\sigma_{(j,k),(j',k')}]_{\substack{j=1\ldots m_1;k=1\ldots m_2 \\ j'=1\ldots m_1;k'=1\ldots m_2}}$, a $(m_1 \times m_2) \times (m_1 \times m_2)$ symmetric matrix where $\sigma_{(j,k),(j',k')} = Cov(W_{jk}, W_{j',k'})$. As proposed by [?], the covariance between W_{jk} and $W_{j',k'}$ is estimated by:

$$\widehat{\sigma_{(j,k),(j',k')}} = r_{j,j'}r_{k,k'}$$

where $r_{j,j'} = \frac{p_{jj'} - p_j p_{j'}}{\sqrt{p_j(1-p_j)p_{j'}(1-p_{j'})}}$ is the widely used correlation measure between SNP j and SNP j', given that p_j and $p_{j'}$ are the respective allelic frequencies and $p_{jj'}$ is the joint allelic frequency [?].

In the remainder of this section, the four methods: minP (function `minP.test`, GATES (function `gates.test`), tTS (function `tTS.test`) and tProd (function `tProd.test`) are detailed.

3.1 minP

The minP test is based on the minimum p-value that is often used to combine p-values of association (see [?]). Let $W_{\max} = \max|W_{11}|, \ldots, |W_{m_1,m_2}|$ be the maximum of the absolute observed statistics. The minP is then defined by:

$$\text{minP} = 1 - \mathbb{P}\Big[\max(|Z_1|, |Z_2|, \ldots, |Z_{m_1 m_2}|) < W_{\max}\Big]. \tag{1}$$

where $\mathbb{Z} = (Z_1, Z_2, \ldots, Z_{m_1 m_2})$ is a random vector that follows a multivariate normal distribution $\mathbb{Z} \sim \mathcal{N}(\mathbf{0}, \Sigma)$.

The computation of Equation (**??**) requires the calculation of the probability distribution of a multivariate normal random variable. For that purpose, we used the `pmvnorm` function from the R package `mvtnorm` [?].

```
> set.seed(1234)
> minP.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2)

  Gene-Gene Interaction method performed with:
              minP
  minP = 0.009924148, p-value = 0.1795672
```

3.2 GATES

The GATES procedure, proposed by [?], is an extension of the Simes procedure used to assess the gene level association significance. Let $p_{(1)}, \ldots, p_{(m_1 m_2)}$ be the ascending SNP-SNP interaction $m_1 \times m_2$ p-values, GATES p-value is then defined by

$$\mathrm{p}_{GATES} = \min\left(\frac{mep_{(1)}}{me_{(1)}}, \frac{mep_{(2)}}{me_{(2)}}, \ldots, \frac{mep_{(m_1 m_2)}}{me_{(m_1 m_2)}}\right)$$

where m_e is the number of effective tests among the $m_1 \times m_2$ tests and $me_(i)$ the number of effective tests among the i most significative tests associated with the lowest order p-values $p_{(1)}, \ldots, p_{(i)}$. The number of effective tests ought to characterize the number of independent tests equivalent to the correlated tests that are really performed and is often used to account for dependence in a multiple testing correction.

Although no formal definition of the number of effective tests has been formulated in the literature, several procedures have been proposed to estimate such number. All methods are based on a transformation of the set of eigenvalues of the SNP covariance matrix assuming that (1) if the SNPs are independent, the number of effective tests is the number of performed, (2) if the absolute value of the correlation between any pair of SNPs is equal to 1, the number of effective tests is 1. In the **GeneGeneInteR** package, four main methods have been implemented and can be chosen by the user with the argument `merest`: Cheverud-Nyholt method - `me.est="ChevNy"` [?, ?], Keff method -`me.est="Keff"` [?], Li and Ji method - `me.est="LiJi"` [?] and Galwey - `me.est="Galwey"` [?].

```
> set.seed(1234)
> gates.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,me.est="ChevNy")

  Gene-Gene Interaction method performed with:
             GATES
  GATES = 0.009924148, p-value = 0.293932

> set.seed(1234)
> gates.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,alpha=0.05,me.est="Keff")

  Gene-Gene Interaction method performed with:
             GATES
  GATES = 0.01394543, p-value = 0.1899414

> set.seed(1234)
> gates.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,me.est="LiJi")

  Gene-Gene Interaction method performed with:
             GATES
  GATES = 0.01394543, p-value = 0.1255088

> set.seed(1234)
> gates.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,me.est="Galwey")

  Gene-Gene Interaction method performed with:
             GATES
  GATES = 0.01394543, p-value = 0.1596024
```

3.3 tTS and tProd

tTS and tProd procedures are two truncated tail strength methods that aim at combining signals from all single-SNP p-values less than a predefined cutoff value [?]. Denoting by τ the cutoff value, the two truncated p-values are defined as follows [?]:

$$\begin{aligned} tTS &= \frac{1}{m_1 m_2} \sum_{i=1}^{m_1 m_2} \mathbb{I}(p_{(i)} < \tau) \left(1 - p_{(i)} \frac{m_1 m_2 + 1}{i}\right) \\ tProd &= \prod_{i=1}^{m_1 m_2} p_i^{\mathbb{I}(p_i < \tau)} \end{aligned}$$

where $\mathbb{I}$ is the indicator function.

When p-values are correlated, the null distribution of tTS and $tProd$ are unknown. Following the approach proposed by [?], a p-value is obtained in the **GeneGeneInteR** package by computing an empirical null distribution using Monte-Carlo (MC) simulations. For each MC iteration, an empirical value for tTS (or $tProd$) is obtained by simulating a vector of W_{jk} with respect to a multivariate normal distribution with a vector of 0 means and $\widehat{\Sigma}$ as covariance matrix. The empirical p-value is calculated as the proportion of simulated statistics larger than the observed statistic on the "true" set of W_{jk}.

tTS and tProd methods have been implemented in the functions `tTS.test` and `tProd.test` of the **GeneGeneInteR** package. Additional to the mandatory `Y`, G_1 and G_2 arguments, these two functions have two optional arguments: `tau` and `n.sim` that control the cutoff value and the number of simulations used to estimate the empirical value respectively. The following coding lines give an example of the `tTS.test` and `tProd.test`:

```
> set.seed(1234)
> tTS.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,tau=0.5,n.sim=10000)

   Gene-Gene Interaction method performed with:
               Truncated Tail Strength
   tTS = -0.009912706, p-value = 0.5104

> set.seed(1234)
> tProd.test(Y=gene.pair$Y, G1=gene.pair$G1,G2=gene.pair$G2,tau=0.05,n.sim=1000)

   Gene-Gene Interaction method performed with:
       Truncated Product
   tProd = 3.69877e-12, p-value = 0.485
```