Scientific Introduction: Performing Evolutionary Transcriptomics with R

In the past years, a large body of scientific studies aimed at investigating the molecular basis of variation and conservation within biological processes. These transcriptomics studies allowed us to get a glimpse into the molecular patterns and processes that underly complex biological processes such as development or cell differentiation.

Although powerful for investigating the molecular mechanisms that determine the biological process of interest, these datasets rarely capture the evolutionary history of how these expression patterns emerged or to what extent they are possibly constrained. By combining transcriptomics studies with a comparative genomics approach, however, we can capture some evolutionary signatures that allow us to understand the variability or constraint of particular sets of genes. Finding such transcriptome conservation or variability patterns within a biological process of interest, allows us to investigate the evolutionary processes that might have caused the constraints found in transcriptomes.

The myTAI package provides an analytics tool to perform evolutionary transcriptomics studies and is designed to detect transcriptome conservation patterns in transcriptome data. It furthermore, seeks to provide a consistent way to design a computationally reproducible analytics process that achieves a high degree of transparency when conducting evolutionary transcriptomics studies.

Installation

Users can download myTAI from CRAN:

```r
# install myTAI package
source("http://bioconductor.org/biocLite.R")
biocLite('myTAI')
```

Since most myTAI functions are optimized and internally implemented in C++ please make sure that you have a running version of C++ installed on your machine.

Motivation

Using seven stages of embryo development of the model plant *Arabidopsis thaliana* as an example, we ask the user to imagine how one would investigate the differences of developmental transcriptomes across
developmental stages.

Figure 1: Gene expression distributions (= developmental transcriptome) throughout seven stages of Arabidopsis thaliana embryo development. Embryo development is divided into three phases: early embryogenesis (purple), mid-embryogenesis (green), and late embryogenesis (brown). This boxplot illustrates that the overall distributions of log2 expression levels (y-axis) hardly differ between developmental stages (x-axis) although the difference on the global scale is statistically significant (Kruskal-Wallis Rank Sum Test: p < 2e-16). Hence, a clear visual pattern of gene expression differences between early, mid, and late embryogenesis on the global scale can not be inferred.

The objective of performing evolutionary transcriptomics studies is to classify a transcriptome into different categories of genes sharing similar evolutionary origins (detectable homologs) or genes that share similar phylogenetic relationships (orthologous genes) and to study the overall expression patterns of these classified genes throughout the biological process of interest. Thus, by introducing a phylogenetic or taxonomic variable to a transcriptome dataset, we can determine stages or time points that more constrained than others, indicating switches between gene expression programs.

Conceptually, the idea behind evolutionary transcriptomics studies is to combine the phylogenetic relationship between species (usually retrieved from comparative genomics studies in terms of sequence homology) with transcriptome data of a reference species quantifying a particular biological process of interest (e.g. mutant gene expression versus WT gene expression, stress responses, cell differentiation, development, etc.). Usually, transcriptome data comes from Next Generation Sequencing technologies such as RNA-Seq or from Microarray experiments.

Phylogenetic Information + Transcriptome Data

Or in other words:

Comparative Genomics + Transcriptomics = Evolutionary Transcriptomics

In theory, any published or newly generated transcriptome dataset can be used to capture transcriptome conservation patterns with myTAI.

myTAI is designed to receive phylogenetic information obtained from comparative genomics data and transcriptome data as input and internally combines these datasets to perform evolutionary transcriptomics analyses.
Figure 2: Gene expression distributions (= developmental transcriptome) throughout seven stages of A. thaliana embryo development classified into distinctive age categories. Each box represents the developmental stage during A. thaliana embryogenesis, the y-axis denotes the log2 expression levels of genes that fall into the corresponding age category shown on the x-axis. Hence, each boxplot represents the gene expression distribution of genes that are classified into the corresponding age class during a specific developmental stage. The gene age distribution of A. thaliana genes range from PS1 to PS12 where PS1 represents the evolutionarily most distant age category (cellular org.) and PS12 the evolutionary most recent age category (A. thaliana specific). Yellow dots in the boxplots denote the mean expression level of the corresponding expression distribution. This visualization illustrates that although the global gene expression distributions do not change visually between developmental stages (Fig. 1), the global gene expression distributions of age categories differ between stages of A. thaliana embryo development, and thus, allow studying the effect of transcriptome evolution and conservation on embryo development.
Figure 3: Workflow describing the input and output of the myTAI package. The myTAI package takes phylogenetic information such as phylogenetic trees (see Dunn, 2013), genomic phylostratigraphy based gene age inference (see Domazet-Loso et al., 2007; Capra et al., 2013; Liebeskind et al., 2016), by dNdS estimation of orthologous genes (see Quint, Drost et al., 2012 and Drost et al., 2015), or phylogenetic reconciliation (see Doyon et al., 2011) and a RNA-Seq or Microarray based transcriptome dataset as input. Internally, myTAI then combines the phylogenetic data and the transcriptome data an provides numerous functions to perform evolutionary transcriptomics analyses. Here, we examplify the output of the functions PlotSignature(), PlotRE() and PlotCategoryExpr().
Retrieval of phylogenetic or taxonomic information

For the comparative genomics part there are different methods and tools to quantify sequence homology between genes of a reference species and related species. For example, for phylogenetic or taxonomic information retrieval such as phylogenetic trees, genomic phylostratography based gene age inference, dNdS estimation of orthologous genes or phylogenetic reconciliation can be used. Below users can find the most recent tools and resources for retrieving or computing phylogenetic or taxonomic relationships for an organism of interest.

Phylogenetic reconciliation

Phylogenetic reconciliation models use species trees for modeling individual gene evolution. Some models account for gene duplications and losses. In general, reconciliation is a popular approach for inferring orthology relationships, even though its accuracy strongly depends on the phylogeny reliability Doyon, 2011. If applicable and useful for their study, users can infer gene age information using established phylogenetic reconciliation methods and can then transform the corresponding output in the specified phylostratigraphic map data format for use with myTAI.

Genomic phylostratography based gene age inference

As intensely discussed in recent years (Capra et al., 2013; Altenhoff et al., 2016; Liebeskind et al., 2016), gene age inference isn’t a trivial task and existing approaches are under ongoing improvement and research (Liebeskind et al., 2016).

In particular, Moyers & Zhang argue that genomic phylostratigraphy 1) underestimates gene age for a considerable fraction of genes, 2) is biased for rapidly evolving proteins which are short, and/or their most conserved block of sites is small, and 3) these biases create spurious nonuniform distributions of various gene properties among age groups, many of which cannot be predicted a priori (Moyers & Zhang, 2015; Moyers & Zhang, 2016; Liebeskind et al., 2016). However, these arguments were based on simulated data and were inconclusive due to errors in their analyses. Furthermore, Domazet-Loso et al., 2017 provide further evidence that there is no phylostratigraphic bias (but see also Moyers & Zhang, 2017). In general, however, an objective benchmarking set representing the tree of life is still missing and therefore any procedure aiming to quantify gene ages will be biased to some degree.

Based on this debate a recent study suggested to perform gene age inference by combining thirteen common orthology inference algorithms to create gene age datasets and then characterize the error around each age-call on a per-gene and per-algorithm basis. Using this approach systematic error was found to be a large factor in estimating gene age, suggesting that simple consensus algorithms are not enough to give a reliable point estimate (Liebeskind et al., 2016). However, by generating a consensus gene age and quantifying the possible error in each workflow step, Liebeskind et al., 2016 provide a very useful database of consensus gene ages for a variety of genomes.

Alternatively, Stephen Smith, 2016 argues that de novo gene birth/death and gene family expansion/contraction studies should avoid drawing direct inferences of evolutionary relatedness from measures of sequence similarity alone, and should instead, where possible, use more rigorous phylogeny-based methods. If users choose to follow the advise given by Smith (2016), we recommend to consult the phylomedb database to retrieve phylogeny-based gene orthology relationships and use these age estimates in combination with myTAI.

Evidently, these advancements in gene age inference research are very recent and inferring gene age is a very young and active field of genomic research. Therefore, many more studies need to address the robust and realistic estimation of gene age and propose a community standard.

Despite the ongoing debate about how to correctly infer gene age, users of myTAI can perform any gene age inference method they find most appropriate for their biological question and pass this gene age inference table as input to myTAI. To do so, users need to follow the following data format specifications to use their gene age inference table with myTAI.
The rational behind gene age inference is to assign each protein coding gene of an organism of interest with an evolutionary age estimate which aims to quantify its potential origin within the tree of life (detectable sequence homolog; orphan gene (see Tautz & Domazet-Loso, 2011)). Hence, gene age inference generates a table storing the gene age in the first column and the corresponding gene id of the organism of interest in the second column. This table is named **phylostratigraphic map**.

### Generate or retrieve phylostratigraphic maps using BLAST based methods

- agalma: **Python** scripts to perform the phylogenetic comparative method for gene age inference
- createPSmap.pl: **Perl** script to generate a phylostratigraphic map using phylostratigraphy (implemented by Alexander Gabel)
- phylostratigraphy.pl: **Perl** script to generate a phylostratigraphic map using phylostratigraphy (implemented by Cheng et al. 2015)
- phylo_pipeline.py: **Python** script to generate a phylostratigraphic map using phylostratigraphy (implemented by Shuqing Xu)
- ORFanFinder: web server and command line tool to generate a phylostratigraphic map
- Protein Historian: a database storing pre-computed phylostratigraphic maps
- download pre-computed and published phylostratigraphic maps from previous studies (only selected species)
- Liebeskind et al., 2016: use a gene age consensus approach to estimating gene ages for model organisms
- orthoscape: a cytoscape application for grouping and visualization KEGG based gene networks by taxonomy and homology principles (implemented by Mustafin et al., 2017)
- RecBlast: Cloud-Based Large Scale Orthology Detection (by Efrat Rapoport and Moran Neuhof)

### dNdS estimation of orthologous genes

We recently proposed to use the classical dNdS measure to quantify the sequence conservation of protein coding genes between closely related species. This way, we combine the information about the selective pressure acting on a particular gene with its expression level during a particular time point or condition. We refer to this approach as Divergence Stratigraphy (Drost et al., 2015 *Mol. Biol. Evol.*). Analogous to other gene age inference methods, divergence stratigraphy generates a table storing the sequence conservation estimate in the first column and the corresponding gene id of the organism of interest in the second column. This table is named **divergence stratigraphic map**.

### Generate or retrieve divergence stratigraphic maps

- orthologr: **R** package to generate a divergence stratigraphic map (implemented by Hajk-Georg Drost)
- compute_dNdS.pl: **Perl** script to generate a divergence stratigraphic map (implemented by Cheng et al. 2015)
- MetaPhOrs: retrieve phylogeny-based orthology and paralogy predictions
- online computation and published divergence stratigraphic maps
- orthoscape: a cytoscape application for grouping and visualization KEGG based gene networks by taxonomy and homology principles (implemented by Mustafin et al., 2017)
- RecBlast: Cloud-Based Large Scale Orthology Detection (by Efrat Rapoport and Moran Neuhof)

### Generate custom table

In general, users can construct their own gene age assignment methods and are not limited to the methods listed above. After formatting the corresponding results to the phylostratigraphic map specification (age assignment in the first column and gene id in the second column), users can use any function in myTAI with their custom gene age assignment table.
Getting Started with myTAI

myTAI takes a phylostratigraphic map and an expression dataset as input and combines both tables to quantify and detect transcriptome conservation patterns for the biological process of interest.

Defining input data standards

The following code illustrates an example structure of a gene age map. Here we choose genomic phylostratigraphy and dNdS estimation as method to generate a phylostratigraphic map and divergence stratigraphic map:

```r
data(PhyloExpressionSetExample)
data(DivergenceExpressionSetExample)

head(PhyloExpressionSetExample[, c("Phylostratum","GeneID")])

Phylostratum GeneID
1 1 atig01040.2
2 1 atig01050.1
3 1 atig01070.1
4 1 atig01080.2
5 1 atig01090.1
6 1 atig01120.1

In detail, a phylostratigraphic map stores the gene age assignment generated with e.g. phylostratigraphy in the first columns and the corresponding gene id in the second column.

Analogously, a divergence stratigraphic map stores the gene age assignment generated with e.g. divergence stratigraphy in the first column and the corresponding gene id in the second column:

```r
data(DivergenceExpressionSetExample)

head(DivergenceExpressionSetExample[, c("Divergence.stratum","GeneID")])

Divergence.stratum GeneID
1 1 atig01050.1
2 1 atig01120.1
3 1 atig01140.3
4 1 atig01170.1
5 1 atig01230.1
6 1 atig01540.2

Hence, myTAI relies on pre-computed gene age maps fulfilling the aforementioned data format for all downstream analyses. It does not matter whether or not gene age maps contain categorized age values like in phylostratigraphic maps or e.g. phylogenetic distance values generated by phylogenetic inference.

Expression data format specification

The aim of any evolutionary transcriptomics study is to quantify and detect transcriptome conservation patterns in biological processes. For this purpose, users need to provide the transcriptome dataset of their biological process of interest.
In the following examples we will use a gene expression dataset covering seven stages of *Arabidopsis thaliana* embryo development. This data format is defined as ExpressionMatrix in the myTAI data format specification.

```
# gene expression set
```

| GeneID     | Zygote | Quadrant | Globular | Heart | Torpedo | Bent | Mature |
|------------|--------|----------|----------|-------|---------|------|--------|
| at1g01040.2 | 2173.6352 | 1911.2001 | 1152.5553 | 1291.4224 | 1000.2529 | 962.9772 | 1696.4274 |
| at1g01050.1 | 1501.0141 | 1817.3086 | 1665.3089 | 1564.7612 | 1496.3207 | 1114.6435 | 1071.6555 |
| at1g01070.1 | 1212.7927 | 1233.0023 | 939.2000 | 929.6195 | 864.2180 | 877.2060 | 894.8189 |
| at1g01080.2 | 1016.9203 | 936.3837 | 1181.3381 | 1329.4734 | 1392.6429 | 1287.9746 | 861.2605 |
| at1g01090.1 | 11424.5667 | 16778.1685 | 34366.6493 | 39775.6405 | 56231.5689 | 66980.3673 | 7772.5617 |
| at1g01120.1 | 844.0414 | 787.5929 | 859.6267 | 931.6180 | 942.8453 | 870.2625 | 792.7542 |

The function `MatchMap()` allows users to join a *phylostratigraphic map* with an ExpressionMatrix to obtain a joined table referred to as *PhyloExpressionSet*. In some cases, the GeneIDs stored in the ExpressionMatrix and in the *phylostratigraphic map* do not match. This is due to GeneID mappings between different databases and annotations. To map non matching GeneIDs between databases and annotations, please consult the Functional Annotation Vignette in the biomartr package. The *biomartr* package allows users to map GeneIDs between database annotations.

After matching a *phylostratigraphic map* with an ExpressionMatrix using the `MatchMap()` function, a standard *PhyloExpressionSet* is returned storing the phylostratum assignment of a given gene in the first column, the gene id of the corresponding gene in the second column, and the entire gene expression set (time series or treatments) starting with the third column. This format is crucial for all functions that are implemented in the myTAI package.

```
library(myTAI)

# load the example data set
data(PhyloExpressionSetExample)

# construct an example Phylostratigraphic Map
Example.PhylostratigraphicMap <- PhyloExpressionSetExample[, 1:2]

# construct an example ExpressionMatrix
Example.ExpressionMatrix <- PhyloExpressionSetExample[, 2:9]

# join a PhylostratigraphicMap with an ExpressionMatrix using MatchMap()
Example.PhyloExpressionSet <- MatchMap(Example.PhylostratigraphicMap, Example.ExpressionMatrix)

# look at a standard PhyloExpressionSet
head(Example.PhyloExpressionSet, 3)
```

| Phylostratum | GeneID     | Zygote | Quadrant | Globular | Heart | Torpedo | Bent | Mature |
|--------------|------------|--------|----------|----------|-------|---------|------|--------|
| 1            | at1g01010.1 | 878.2158 | 828.2301 | 776.0703 | 753.9589 | 775.3377 | 756.2460 | 999.9118 |
| 2            | at1g01020.1 | 1004.9710 | 1106.2621 | 1037.5141 | 939.0830 | 961.5249 | 871.4684 | 997.5953 |
| 3            | at1g01030.1 | 819.4880 | 771.6396 | 810.8717 | 866.7780 | 773.7893 | 747.9941 | 785.6105 |

Analogous to a standard *PhyloExpressionSet*, a standard *DivergenceExpressionSet* is a *data.frame* storing the divergence stratum assignment of a given gene in the first column, the gene id of the corresponding gene in the second column, and the entire gene expression set (time series or treatments) starting with the third column.

The following *DivergenceExpressionSet* example illustrates the standard *DivergenceExpressionSet* data set format.

```
# head of an example standard DivergenceExpressionSet
head(DivergenceExpressionSetExample, 3)
```
A DivergenceExpressionSet defines the joined table between a divergence stratigraphic map and an ExpressionSet. A DivergenceExpressionSet can be generated analogous to a PhyloExpressionSet by joining a divergence stratigraphic map with an ExpressionMatrix using the MatchMap() function. In some cases, the GeneIDs stored in the ExpressionMatrix and in the divergence stratigraphic map do not match. This is due to GeneID mappings between different databases and annotations. To map non matching GeneIDs between databases and annotations, please consult the Functional Annotation Vignette in the biomart package.

Each function implemented in myTAI checks internally whether or not the PhyloExpressionSet or DivergenceExpressionSet standard is fulfilled.

# used by all myTAI functions to check the validity of the PhyloExpressionSet standard
is.ExpressionSet(PhyloExpressionSetExample)

[1] TRUE

In case the PhyloExpressionSet standard is violated, the is.ExpressionSet() function will return FALSE and the corresponding function within the myTAI package will return an error message.

# used a non standard PhyloExpressionSet
head(PhyloExpressionSetExample[ , 2:5], 2)

GeneID Zygote Quadrant Globular
1 at1g01040.2 2173.635 1911.200 1152.555
2 at1g01050.1 1501.014 1817.309 1665.309

is(ExpressionSet(PhyloExpressionSetExample[, 2:5])

Error in is.ExpressionSet(PhyloExpressionSetExample[, 2:5]) :
  The first column of the ExpressionSet needs to store numeric values.

The PhyloExpressionSet and DivergenceExpressionSet formats are crucial for all functions that are implemented in the myTAI package.

Keeping these standard data formats in mind will provide users with the most important requirements to get started with myTAI.

Note, that within the code of each function, the argument ExpressionSet always refers to either a PhyloExpressionSet or a DivergenceExpressionSet, whereas in specialized functions some arguments are specified as PhyloExpressionSet when they take an PhyloExpressionSet as input data set, or specified as DivergenceExpressionSet when they take an DivergenceExpressionSet as input data set.

Performing a Standard Workflow for Evolutionary Transcriptomics Analyses

The main goal of any evolutionary transcriptomics study is to quantify transcriptome conservation at a particular stage or treatment. This is achieved by computing the average age of genes that contribute to the transcriptome at that stage or treatment. In other words, by multiplying the gene age value with the expression level of the corresponding gene and averaging over all genes, we obtain the mean age of the transcriptome. Hence, we can say that at a particular stage genes that are most expressed at this stage or treatment have (on average) the evolutionary age XY.

To obtain this mean age value, several measures were introduced:
Transcriptome Age Index

The most prominent measure, Transcriptome Age Index (TAI), was introduced by Domazet-Loso and Tautz, 2010 and represents a weighted arithmetic mean of the transcriptome age during a corresponding developmental stage \( s \).

\[
TAI_s = \sum_{i=1}^{n} \frac{p_{si} \cdot e_{is}}{\sum_{i=1}^{n} e_{is}}
\]

where \( p_{si} \) denotes the gene age assignment of gene \( i \) and \( e_{is} \) denotes the gene expression level of gene \( i \) at developmental time point \( s \). A lower value of TAI describes an older transcriptome age, whereas a higher value of TAI denotes a younger transcriptome age.

The following figure shows the TAI computations for the seven stages of \( A. \ thaliana \) embryo development.

data(PhyloExpressionSetExample)
# Plot the Transcriptome Age Index of a given PhyloExpressionSet
# Test Statistic : Flat Line Test (default)
PlotSignature( ExpressionSet = PhyloExpressionSetExample,
               measure = "TAI",
               TestStatistic = "FlatLineTest",
               xlab = "Ontogeny",
               ylab = "TAI" )

\[ p_{flt} = 4.5 \times 10^{-10} \]

The x-axis shows the seven stages of \( A. \ thaliana \) embryo development and the y-axis shows the corresponding mean transcriptome age (TAI) value. The smaller the TAI value the older the mean transcriptome age and the higher the TAI value the younger the mean transcriptome age.
The interpretation of the TAI values on the y-axis is given by the next figure.

In this example, a TAI value of 3.5 quantifies that genes that contribute most the transcriptome at a particular stage emerged on average between phylostratum 3 and phylostratum 4. Due to the nature of the arithmetic mean, this value does not represent the true origin of individual genes, and thus the TAI measure is only helpful to screen for stages that express (on average) older or younger genes. Subsequent analyses such as mean expression of age categories, relative expression levels, and gene expression level distributions for each age category will then reveal which exact genes or age categories generate the overall TAI value.

To obtain a more detailed overview of which age category contributes how much transcripts to each developmental stage, the gene expression level distributions for each age category and each developmental stage can be visualized (using the `PlotCategoryExpr()` function).

```r
data(PhyloExpressionSetExample)
# category-centered visualization of PS
# specific expression level distributions (log-scale)
PlotCategoryExpr(ExpressionSet = PhyloExpressionSetExample,
                  legendName = "PS",
                  test.stat = TRUE,
                  type = "category-centered",
                  distr.type = "boxplot",
                  log.expr = TRUE)
```

```output
Zygote Quadrant Globular Heart Torpedo Bent Mature
category-centered "***" "***" "***" "***" "***" "***"
```
This figure shows that in all developmental stages, genes coming from PS1-3 are (on average) more expressed than genes coming from PS4-12. Interestingly, the gene expression level distributions of PS4-12 become more equally distributed towards the Torpedo stage which has been marked as the most conserved stage by TAI analysis. This general trend can be visualized using the `PlotMeans()` function.

```r
data(PhyloExpressionSetExample)
# plot evolutionary old PS (PS1-3) vs evolutionary young PS (PS4-12)
PlotMeans(PhyloExpressionSetExample,
          Groups = list(c(1:3), c(4:12)),
          legendName = "PS",
          adjust.range = TRUE)
```
Here, users will observe that indeed PS1-3 genes are (on average) higher expressed than PS4-12 genes.

Using a linear transformation of the mean expression levels into the interval $[0, 1]$ (Quint et al., 2012 and Drost et al., 2015) we can compare mean expression patterns between Phylostrata independent from their actual mean expression magnitude. A relative expression level of 0 denotes the minimum mean expression level compared to all other stages and a relative expression level of 1 denotes the maximum mean expression level compared to all other stages.

The following figure illustrates the average gene expression profile for each phylostratum.

```r
data(PhyloExpressionSetExample)
# plot evolutionary old PS (PS1-3) vs evolutionary young PS (PS4-12)
PlotRE(PhyloExpressionSetExample,
      Groups = list(c(1:3), c(4:12)),
      legendName = "PS",
      adjust.range = TRUE)
```
Users will observe, that PS4-12 genes are down-regulated towards the Torpedo stage (marked as most conserved by TAI analysis) and up-regulated after the Torpedo stage.

Next, we can cluster gene expression levels of PS4-12 genes to retrieve sets of genes that follow the expression level patterns high-low-low, high-low-high, and low-high-high. For this purpose, we group *A. thaliana* embryogenesis into three developmental modules (early, mid, and late) and cluster young genes (PS4-12) according to their fold-change pattern: High-Low-Low, High-Low-High, and Low-Low-High.
High−Low−Low

57 Genes

High−Low−High

9 Genes

Low−Low−High

69 Genes
As a result, we find that there are two distinct sets of young genes: High-high-low and low-low-high. Almost none of the genes have a high-low-high pattern.

Finally, users can perform KEGG or GO Term enrichment analyses with biomartr to obtain the annotated functions of these gene sets.

**Transcriptome Divergence Index**

Analogous to the TAI measure, the *Transcriptome Divergence Index* (TDI) was introduced by Quint et al., 2012 and Drost et al., 2015 as a measure of average transcriptome selection pressure where $s$ denotes the corresponding developmental stage.

$$TDI_s = \sum_{i=1}^{n} \frac{d_{si} \times c_{is}}{\sum_{i=1}^{n} c_{is}}$$

where $d_{si}$ denotes the divergence stratum assignment of gene $i$ and $c_{is}$ denotes the gene expression level of gene $i$ at developmental time point $s$. A lower value of TDI describes a more conserved transcriptome (in terms of sequence dissimilarity), whereas a higher value of TDI denotes a more variable transcriptome.

To assess the statistical significance of all introduced measures and analyses, we developed several test statistics that are introduced in the following sections.

**Statistical Assessment and Quantification of Transcriptome Conservation Patterns**

Transcriptome conservation patterns can be quantified by computing transcriptome indices at different stages of development, combining these values to a transcriptome index profile across the measured stages, and comparing the resulting profile with a flat line. A profile not significantly deviating from a flat line indicates the absence of significant variations of the computed transcriptome index from stage to stage. In contrast, a profile significantly deviating from a flat line indicates the presence of significant variations from stage to stage. We refer to any transcriptome index profile significantly deviating from a flat line as phylotranscriptomic pattern or evolutionary signature.

Previously, we introduced three statistical tests to quantify the significance of observed TAI or TDI patterns: Flat Line Test, Reductive Hourglass Test, and Reductive Early Conservation Test (Drost et al., 2015).

The `PlotSignature()` function introduced in the following sections is the main analytics function of myTAI. `PlotSignature()` allows users to visualize TAI or TDI patterns and internally performs the following statistical tests to assess their significance.

**Flat Line Test**

The `PlotSignature()` function with option `TestStatistic = "FlatLineTest"`, first computes the TAI (given a PhyloExpressionSet and argument specification `measure = "TAI"`) or the TDI (given a Divergence-ExpressionSet and argument specification `measure = "TDI"`) profile as well as their standard deviation, and statistical significance.

```r
data(PhyloExpressionSetExample)
# Plot the Transcriptome Age Index of a given PhyloExpressionSet
# Test Statistic : Flat Line Test (default)
PlotSignature( ExpressionSet = PhyloExpressionSetExample,
               measure = "TAI",
               TestStatistic = "FlatLineTest",
               xlab = "Ontogeny",
               ylab = "TAI" )
```
The p-value ($p_{flt}$) above the TAI curve is returned by the FlatLineTest. As described in the documentation of PlotSignature() (?PlotSignature or ?FlatLineTest), the FlatLineTest is the default statistical test to quantify the statistical significance of the observed phylotranscriptomic pattern. In detail, the test quantifies any statistically significant deviation of the phylotranscriptomic pattern from a flat line. Here, we define any significant deviation of a phylotranscriptomic pattern from a flat line as evolutionary signature. Furthermore, we define corresponding stages of deviation as evolutionary conserved or variable (less conserved) depending on the magnitude of TAI values and the corresponding p-value returned by the FlatLineTest. In summary, the FlatLineTest only tests whether or not a particular transcriptome conservation pattern is present in the dataset of interest, but does not test what kind of pattern can be detected. For testing specific transcriptome conservation patterns we developed more specialized tests such as the Reductive Hourglass Test or Reductive Early Conservation Test.

### Reductive Hourglass Test

In case the observed phylotranscriptomic pattern not only significantly deviates from a flat line but also visually resembles an hourglass shape (high-low-high pattern), one can obtain a p-value quantifying the statistical significance of a visual hourglass pattern based on the ReductiveHourglassTest (?ReductiveHourglassTest).

Since the ReductiveHourglassTest has been defined for a priori biological knowledge (Drost et al., 2015), the modules argument within the ReductiveHourglassTest() function needs to be specified.

Three modules need to be specified: an **early-module**, **phytotypic module** (mid), and a **late-module**.

For this example we divide *A. thaliana* embryo development stored within the PhyloExpressionSetExample into the following three modules:
• early = stages 1 - 2 (Zygote and Quadrant)
• mid = stages 3 - 5 (Globular, Heart, and Torpedo)
• late = stages 6 - 7 (Bent and Mature)

# Plot the Transcriptome Age Index of a given PhyloExpressionSet
# Test Statistic : Reductive Hourglass Test

PlotSignature(ExpressionSet = PhyloExpressionSetExample,
               measure = "TAI",
               TestStatistic = "ReductiveHourglassTest",
               modules = list(early = 1:2, mid = 3:5, late = 6:7),
               xlab = "Ontogeny",
               ylab = "TAI",
               shaded.area = TRUE )

#> Plot signature: ' TAI ' and test statistic: ' ReductiveHourglassTest '.
#> Modules:
#> early = { Zygote Quadrant }
#> mid = { Globular Heart Torpedo }
#> late = { Bent Mature }
#> Significance status of signature: significant.

p_rht = 2.16e−08

The corresponding p-value \( p_{\text{rht}} \) now denotes the p-value returned by the \texttt{ReductiveHourglassTest} which is different from the p-value returned by the \texttt{FlatLineTest} \( (p_{\text{flt}}) \).

To make sure that correct modules have been selected to perform the \texttt{ReductiveHourglassTest}, users can use the \texttt{shaded.area} argument to visualize chosen modules:

# Visualize the phylotypic period used for the Reductive Hourglass Test
PlotSignature(ExpressionSet = PhyloExpressionSetExample,
Note that for defining a priori knowledge for the `ReductiveHourglassTest` using the `modules` argument, modules need to start at stage 1, ..., N and do not correspond to the column position in the `PhyloExpressionSet/DivergenceExpressionSet` which in contrast would start at position 3, ... N + 2.

In a biological context, it is not always clear which stages could define the early, mid, and late modules. For animal embryogenesis, it has been suggested to choose early, mid, and late modules according to the morphological conservation of animal embryos (e.g. mid-stage vertebrate embryos seem to be morphologically conserved). For plants, in contrast no morphological conservation has been reported and thus the average expression of embryo defective genes has been used to define modules (Drost et al. 2015).

**Reductive Early Conservation Test**

The third test statistic that is implemented in the `myTAI` package is the `EarlyConservationTest`.
The `EarlyConservationTest` tests whether an observed phylotranscriptomic pattern follows a low-high-high pattern (monotonically increasing function) supporting the Early Conservation Model of embryogenesis.

Analogous to the `ReductiveHourglassTest`, the `EarlyConservationTest` needs a priori biological knowledge. So again three modules have to be specified for the `EarlyConservationTest()` function.

Three modules need to be specified: an early-module, phylotypic module (mid), and a late-module.

For this example we divide *A. thaliana* embryo development stored within the `PhyloExpressionSetExample` into the following three modules:

- early = stages 1 - 2 (Zygote and Quadrant)
- mid = stages 3 - 5 (Globular, Heart, and Torpedo)
- late = stages 6 - 7 (Bent and Mature)

```r
# Plot the Transcriptome Age Index of a given PhyloExpressionSet
# Test Statistic : Reductive Early Conservation Test
PlotSignature( ExpressionSet = PhyloExpressionSetExample, measure = "TAI", TestStatistic = "EarlyConservationTest", modules = list(early = 1:2, mid = 3:5, late = 6:7), xlab = "Ontogeny", ylab = "TAI", shaded.area = TRUE )
```

```r
#> Plot signature: ' TAI ' and test statistic: ' EarlyConservationTest '.
#> Modules:
#> early = { Zygote Quadrant }
#> mid = { Globular Heart Torpedo }
#> late = { Bent Mature }
#> Significance status of signature: not significant (= no evolutionary signature in the transcriptome).
```
The corresponding p-value \( p\_ect \) now denotes the p-value returned by the EarlyConservationTest which is different from the p-value returned by the FlatLineTest (\( p\_flt \)) and ReductiveHourglassTest (\( p\_rht \)).

Since the present TAI pattern of the PhyloExpressionSetExample doesn’t support the Early Conservation Hypothesis, the p-value \( p\_ect = 1 \).

Again note that for defining a priori knowledge for the EarlyConservationTest using the modules argument, modules need to start at stage 1, \( \ldots \), \( N \) and do not correspond to the column position in the PhyloExpressionSet/DivergenceExpressionSet which in contrast would start at position 3, \( \ldots \) \( N + 2 \).

To obtain the numerical TAI values, the TAI() function can be used:

```r
# Compute the Transcriptome Age Index values of a given PhyloExpressionSet
TAI(PhyloExpressionSetExample)
```

```
Zygote Quadrant Globular Heart Torpedo Bent Mature
3.229942 3.225614 3.107135 3.116693 3.073993 3.176511 3.390334
```

**Transcriptome Divergence Index Analyses**

Analogous to the TAI computations and visualization, the TDI computations can be performed in a similar fashion:

```r
data(DivergenceExpressionSetExample)
# Plot the Transcriptome Divergence Index of a given DivergenceExpressionSet
# Test Statistic : Flat Line Test (default)
PlotSignature( ExpressionSet = DivergenceExpressionSetExample,
               measure = "TDI",
               TestStatistic = "FlatLineTest",
```

21
Again, for the **ReductiveHourglassTest** we divide *A. thaliana* embryo development into three modules:

- **early**: stages 1 - 2 (Zygote and Quadrant)
- **mid**: stages 3 - 5 (Globular, Heart, and Torpedo)
- **late**: stages 6 - 7 (Bent and Mature)

```r
plot_signature = PlotSignature(
  ExpressionSet = DivergenceExpressionSetExample,
  measure = "TDI",
  TestStatistic = "ReductiveHourglassTest",
  modules = list(early = 1:2, mid = 3:5, late = 6:7),
  xlab = "Ontogeny",
  ylab = "TDI",
  shaded.area = TRUE)
```

- > Plot signature: 'TDI' and test statistic: 'ReductiveHourglassTest'.
- > Modules:
  - early = {Zygote, Quadrant}
  - mid = {Globular, Heart, Torpedo}
  - late = {Bent, Mature}
- > Significance status of signature: significant.

And for the **EarlyConservationTest** we again divide *A. thaliana* embryo development into three modules:
• early = stages 1 - 2 (Zygote and Quadrant)
• mid = stages 3 - 5 (Globular, Heart, and Torpedo)
• late = stages 6 - 7 (Bent and Mature)

data(DivergenceExpressionSetExample)

# Plot the Transcriptome Divergence Index of a given DivergenceExpressionSet
# Test Statistic : Reductive Early Conservation Test

PlotSignature( ExpressionSet = DivergenceExpressionSetExample,
               measure      = "TDI",
               TestStatistic = "EarlyConservationTest",
               modules      = list(early = 1:2, mid = 3:5, late = 6:7),
               xlab         = "Ontogeny",
               ylab         = "TDI",
               shaded.area  = TRUE )

#> Plot signature: ' TDI ' and test statistic: ' EarlyConservationTest '.
#> Modules:
#>   early = { Zygote Quadrant }
#>   mid   = { Globular Heart Torpedo }
#>   late  = { Bent Mature }
#> Significance status of signature: not significant (= no evolutionary signature in the transcriptome).

p_ect = 0.973

To obtain the numerical TDI values for a given DivergenceExpressionSet simply run:

# Compute the Transcriptome Divergence Index values of a given DivergenceExpressionSet

TDI(DivergenceExpressionSetExample)

| Ontogeny  | TDI     |
|-----------|---------|
| Zygote    | 4.532029|
| Quadrant  | 4.563200|
| Globular  | 4.485705|
| Heart     | 4.500868|
| Torpedo   | 4.466477|
| Bent      | 4.530704|
| Mature    | 4.690292|
Phylostratum or Divergence Stratum specific contribution to the global transcriptome index profile

Another way to visualize the cumulative contribution of each Phylostratum or Divergence Stratum to the global Transcriptome Age Index or Transcriptome Divergence Index profile was introduced by Domazet-Loso and Tautz, 2010 (Fig. 1b). The advantage of visualizing the cumulative contribution of each Phylostratum or Divergence Stratum to the global pattern is to study how the final (global) TAI or TDI profile emerges from the cumulative TAI/TDI distribution of each Phylostratum or Divergence Stratum. This Phylostratum or Divergence Stratum specific contribution on the global TAI or TDI pattern can be visualized using PlotContribution():

Example: Phylostrata

```
data(PhyloExpressionSetExample)
# visualize phylostrata contribution to the global TAI pattern
PlotContribution( ExpressionSet = PhyloExpressionSetExample,
                 legendName = "PS",
                 xlab = "Ontogeny",
                 ylab = "Transcriptome Age Index",
                 y.ticks = 10)
```

The y.ticks argument allows users to adjust the number of ticks that shall be visualized on the y-axis. The exact values of the Phylostratum specific cumulative TAI profiles can be obtained using the pTAI() function:

```
pTAI(PhyloExpressionSetExample)
```
Example: Divergence Strata

Analogously, the Divergence Stratum specific influence on the global TDI pattern can be visualized using:

```r
# visualize divergence stratum contribution to global TDI
PlotContribution(ExpressionSet = DivergenceExpressionSetExample, legendName = "DS")
```

The exact values of the Divergence Stratum specific cumulative TDI values can be obtained using the `pTDI()` function:

```r
pTDI(DivergenceExpressionSetExample)
```
In both cases (Phylostrata and Divergence Strata) the pTAI() and pTDI() functions return a numeric matrix storing the cumulative TAI or TDI values for each Phylostratum and Divergence Stratum. Note, that the TAI values of Phylostratum 12 (in the pTAI() matrix) are equivalent to TAI(PhyloExpressionSetExample).

This can be explained by the definition of the TAI. Here the sum of all partial TAI values over all Phylostrata is equal to the global TAI values:

```
# show that the cumulative TAI value of PS 12 is
# equivalent to the global TAI() values
pTAI(PhyloExpressionSetExample)[12,]
# > 3.229942 3.225614 3.107135 3.116693 3.073993 3.176511 3.390334
```

```
# show that the colSum() of partial TAI values
# over all Phylostrata equals the global TAI() values
apply(pStrata(PhyloExpressionSetExample), 2, sum)
# > 3.229942 3.225614 3.107135 3.116693 3.073993 3.176511 3.390334
```

Now, the PlotContribution() only differs from apply(pStrata(PhyloExpressionSetExample), 2, sum) by exchanging the sum() by cumsum().

```
# show that apply(pStrata(PhyloExpressionSetExample), 2, cumsum)
# is equivalent to pTAI()
apply(pStrata(PhyloExpressionSetExample), 2, cumsum)
```

```
Zygote Quadrant Globular Heart Torpedo Bent Mature
1 0.3929533 0.3935308 0.4142106 0.4115399 0.4216806 0.4178302 0.3883815
2 0.9521021 0.9547833 0.9748522 0.9674842 0.9632213 0.9285336 0.8661883
 1 0.502814 1.0477016 1.0576104 1.0556300 1.0549156 1.0187303 0.9722031
 4 1.3861830 1.3810595 1.3837548 1.3928030 1.3876006 1.3632945 1.3656170
 5 1.5527473 1.5489929 1.5360986 1.5500708 1.5468767 1.5531699 1.5769758
 6 1.8114772 1.8111167 1.7806803 1.7968463 1.8020203 1.8223456 1.9207232
 7 1.8766090 1.8739449 1.8317327 1.8530276 1.8573216 1.8776292 1.9952325
 8 1.9417254 1.9387499 1.8774361 1.9089144 1.9117438 1.9478303 2.0778287
 9 2.0339129 2.0294517 1.9823091 1.9982676 1.9915249 2.0413791 2.1878074
10 2.4215866 2.4361137 2.3489524 2.3347670 2.3028346 2.3797112 2.5142773
11 2.4900201 2.5079107 2.4104647 2.3980760 2.3651571 2.442093 2.5963136
12 3.2299418 3.2256139 3.1071348 3.1166934 3.0739935 3.1765113 3.3903336
```
This pTAI() matrix is what is being visualized inside PlotContribution().

Note that the pStrata() function returns the partial TAI or TDI values for each Phylostratum or Divergence Stratum, whereas pMatrix() returns the partial TAI or TDI value for each gene.

# compute partial TAI values for each Phylostratum
pStrata(PhyloExpressionSetExample)

# compute partial TAI values for each gene
dplyr::glimpse(pMatrix(PhyloExpressionSetExample))
Mean Expression and Relative Expression of Single Phylostrata or Divergence Strata

TAI or TDI patterns are very useful to gain a first insight into the mean transcriptome age or mean sequence divergence of genes being most active during the corresponding developmental stage or experiment.

To further investigate the origins of the global TAI or TDI pattern it is useful to visualize the mean gene expression of each Phylostratum or Divergence-Stratum class.

Mean Expression Levels of a PhyloExpressionSet and DivergenceExpressionSet

Visualizing the mean gene expression of genes corresponding to the same Phylostratum or Divergence Stratum class allows users to detect biological process specific groups of Phylostrata or Divergence Strata that are most expressed during the underlying biological process. This might lead to correlating specific groups of Phylostrata or Divergence Strata sharing similar evolutionary origins with common functions or functional contributions to a specific developmental process.

Here we see that the mean gene expression of Phylostratum group: PS1-3 (genes evolved before the establishment of embryogenesis in plants) are more expressed during A. thaliana embryogenesis than PS4-12 (genes evolved during or after the establishment of embryogenesis in plants).

In different biological processes different Phylostratum groups or combination of groups might resemble the majority of expressed genes.
The `PlotMeans()` function takes an `PhyloExpressionSet` or `DivergenceExpressionSet` and visualizes for each Phylostratum the mean expression levels of all genes that correspond to this Phylostratum. The `Groups` argument takes a list storing the Phylostrata (classified into the same group) that shall be visualized on the same plot.

For this example we separate groups of Phylostrata into **evolutionary old Phylostrata** (PS1-3) in one plot versus **evolutionary younger Phylostrata** (PS4-12) into another plot:

```r
# Visualizing the mean gene expression of each Phylostratum class # in two separate plots (groups)
PlotMeans( ExpressionSet = PhyloExpressionSetExample,
           Groups = list(group_1 = 1:3, group_2 = 4:12),
           legendName = "PS"
)
```

To obtain the numerical values (mean expression levels for all Phylostrata) run:

```r
# Using the age.apply() function to compute the mean expression levels # of all Phylostrata
age.apply( ExpressionSet = PhyloExpressionSetExample,
           FUN = colMeans )
```

```
Zygote Quadrant Globular Heart Torpedo Bent Mature
1 2607.882 2579.372 2604.856 2525.704 2554.825 2622.757 2696.331
2 2597.258 2574.745 2467.679 2388.045 2296.410 2243.716 2321.709
3 2528.272 2363.159 2019.436 2099.079 2155.642 2196.875 2855.866
4 1925.320 1887.078 1771.399 1787.175 1740.823 1867.981 2358.893
5 2378.883 2368.593 2061.729 2077.087 2076.693 2564.904 3157.761
6 1658.253 1697.242 1485.401 1462.613 1492.861 1631.741 2304.683
```
Here the \texttt{age.apply()} function (\texttt{?age.apply}) takes a function as argument that itself receives a \texttt{data.frame} as argument (e.g. \texttt{colMeans()}).

Users may also specify a shaded area corresponding to the modules that were specified when using the \texttt{PlotSignature()} function.

\begin{verbatim}
# Visualizing the mean gene expression of each Phylostratum class
# and draw an shaded area for the mid-module.
PlotMeans( ExpressionSet = PhyloExpressionSetExample,
            Groups = list(1:12),
            modules = list(1:2,3:5,6:7),
            legendName = "PS")
\end{verbatim}

The \texttt{PlotMedians()} and \texttt{PlotVars()} functions can be used to visualize the median and variance of expression profiles that correspond to the age categories of interest.

For a DivergenceExpressionSet run:

\begin{verbatim}
data(DivergenceExpressionSetExample)
# Visualizing the mean gene expression of each Divergence Stratum class
PlotMeans( ExpressionSet = DivergenceExpressionSetExample,
            Groups = list(1:10),
            legendName = "DS")
\end{verbatim}
To obtain the numerical values (mean expression levels for all Divergence Strata) run:

```r
# Using the age.apply() function to compute the mean expression levels # of all Divergence Strata
age.apply(ExpressionSet = DivergenceExpressionSetExample,
          FUN = colMeans)
```

```
Zygote    Quadrant Globular    Heart Torpedo Bent Mature
1 5222.189 5230.547 5254.464 4911.494 4807.936 4654.683 4277.490
2 3146.510 3020.156 2852.072 2807.367 2845.025 3002.967 3237.315
3 2356.008 2239.344 2257.539 2272.270 2360.816 2529.276 2912.164
4 2230.350 2180.706 2050.895 2049.035 2001.043 2127.165 2608.903
5 2014.600 1994.640 1884.899 1851.554 1858.913 1920.185 2210.391
6 2096.593 2018.440 1938.765 1961.828 1905.246 2005.523 2339.767
7 1836.290 1832.815 1734.319 1719.186 1659.044 1736.141 2201.981
8 1784.470 1762.151 1635.529 1624.682 1590.489 1711.439 1983.607
9 1649.254 1659.455 1522.214 1485.560 1453.689 1584.176 1767.276
10 1660.750 1735.086 1605.275 1473.854 1398.067 1438.258 1541.633
```

The PlotMedians() and PlotVars() functions can be used to visualize the median and variance of expression profiles that correspond to the age categories of interest. Values can be retrieved via age.apply(PhyloExpressionSetExample, function(x) apply(x, 2, median)) and age.apply(PhyloExpressionSetExample, function(x) apply(x, 2, var)).
Relative Expression Levels of a PhyloExpressionSet and DivergenceExpressionSet

Introduced by Domazet-Loso and Tautz, (2010), relative expression levels are defined as a linear transformation of the mean expression levels (of each Phylostratum or Divergence-Stratum) into the interval \([0,1]\) (Quint et al., 2012 and Drost et al., 2015). This procedure allows users to compare mean expression patterns between Phylostrata or Divergence Strata independent from their actual magnitude. Hence, relative expression profiles aim to correlate the mean expression profiles of groups of Phylostrata or Divergence Strata due to the assumption that genes or groups of genes sharing a similar expression profile might be regulated by similar gene regulatory mechanisms or contribute to similar biological processes.

The `PlotRE()` function can be used (analogous to the `PlotMeans()` function) to visualize the relative expression levels of a given PhyloExpressionSet and DivergenceExpressionSet:

```r
# Visualizing the mean gene expression of each Phylostratum class
PlotRE( ExpressionSet = PhyloExpressionSetExample,
          Groups = list(1:12),
          legendName = "PS",
          xlab = "Ontogeny")
```

```r
# Visualizing the mean gene expression of each Divergence Stratum class
PlotRE( ExpressionSet = DivergenceExpressionSetExample,
          Groups = list(1:10),
          legendName = "DS",
          xlab = "Ontogeny")
```
or again by assigning Phylostratum or Divergence Stratum groups that shall be visualized in different plots:

```r
# Visualizing the mean gene expression of each Phylostratum class
PlotRE( ExpressionSet = PhyloExpressionSetExample,
        Groups = list(group_1 = 1:3, group_2 = 4:12),
        legendName = "PS",
        xlab = "Ontogeny")
```
Users may also specify a shaded area corresponding to the modules that were specified when using the `PlotSignature()` function.

```r
# Visualizing the mean gene expression of each Phylostratum class
PlotRE( ExpressionSet = PhyloExpressionSetExample,
  Groups = list(1:12),
  modules = list(1:2,3:5,6:7),
  legendName = "PS",
  xlab = "Ontogeny")
```
The relative expression levels can be obtained using the `REMatrix()` function:

```r
# Getting the relative expression levels for all Phylostrata
REMatrix(PhyloExpressionSetExample)
```

```
Zygote  Quadrant  Globular  Heart  Torpedo  Bent  Mature
1  0.4816246  0.3145330  0.4638918  0.0000000  0.17067495  0.56880234  1.0000000
2  1.0000000  0.9363209  0.6334838  0.14904726  0.0000000  0.2206630  0.56880234  1.0000000
3  0.6083424  0.4109402  0.0000000  0.14904726  0.17067495  0.4638918  1.0000000
4  0.2985050  0.4109402  0.0000000  0.14904726  0.17067495  0.4638918  1.0000000
5  0.2893657  0.2799777  0.0000000  0.14904726  0.17067495  0.4638918  1.0000000
6  0.2323316  0.2799777  0.0000000  0.14904726  0.17067495  0.4638918  1.0000000
7  0.5669797  0.2620602  0.0000000  0.14904726  0.17067495  0.4638918  1.0000000
8  0.4203039  0.3092784  0.09237036  0.05442042  0.45520558  0.0000000  0.6278838
9  0.4586261  0.4686613  0.39738003  0.2206630  0.6278838  0.14904726  0.0000000  0.56880234  1.0000000
10  0.8811321  1.0000000  0.53841500  0.0000000  0.17067495  0.4638918  1.0000000
11  0.4015809  0.4877111  0.04849782  0.07133814  0.17067495  0.4638918  1.0000000
12  0.5052572  0.3359211  0.07100055  0.09489782  0.0000000  0.2206630  0.0000000  0.25811214  1.0000000
```

# Getting the relative expression levels for all Divergence-Strata
```r
REMatrix(DivergenceExpressionSetExample)
```

```
Zygote  Quadrant  Globular  Heart  Torpedo  Bent  Mature
1  0.9696943  0.9755188  1.0000000  0.64894653  0.54294759  0.3860827  0.0000000
2  0.7888009  0.4949178  0.10397567  0.0000000  0.08758660  0.459387  1.0000000
3  0.1739530  0.0000000  0.20704324  0.0489782  0.07133814  0.17067495  0.4638918  1.0000000
4  0.3772372  0.2955661  0.08201140  0.07895260  0.0000000  0.2206630  0.0000000  0.25811214  1.0000000
```

## Ontogeny

The relative expression levels can be obtained using the `REMatrix()` function:

```r
# Getting the relative expression levels for all Phylostrata
REMatrix(PhyloExpressionSetExample)
```

```r
# Getting the relative expression levels for all Divergence-Strata
REMatrix(DivergenceExpressionSetExample)
```
The same result could also be obtained by using the `age.apply()` function in combination with the `RE()` function:

```r
Zygote  Quadrant  Globular  Heart  Torpedo  Bent  Mature
1  0.4816246  0.3145330  0.46389184  0.00000000  0.17067495  0.56880234  1.0000000
2  1.0000000  0.9363209  0.63348381  0.40823711  0.14904726  0.00000000  0.2206063
3  0.6083424  0.4109402  0.00000000  0.09521758  0.16284114  0.21213845  1.0000000
4  0.2985050  0.2366309  0.07499453  0.00000000  0.03592044  0.00000000  0.45908792
5  0.2893657  0.2799777  0.00000000  0.1401191  0.01365328  0.21213845  1.0000000
6  0.2323316  0.2786335  0.02706119  0.00000000  0.20573325  0.00000000  1.0000000
7  0.5666979  0.2620602  0.00000000  0.12099252  0.03592044  0.00000000  1.0000000
8  0.4203039  0.3927846  0.05452042  0.00000000  0.21213845  0.00000000  1.0000000
9  0.4586261  0.4668613  0.30973803  0.23205534  0.00000000  0.03592044  0.45908792
10 0.8811321  1.0000000  0.53841500  0.22974016  0.30490542  0.00000000  0.4881046
11 0.4015809  0.4877111  0.04846721  0.05741594  0.00000000  0.03592044  0.25811214
12 0.5052572  0.3359211  0.07100055  0.09489782  0.00000000  0.03592044  1.0000000
```

In Quint et al. (2012) we introduced an additional way of visualizing the difference of relative expression levels between groups of Phylostrata/Divergence-Strata.

This bar plot comparing the mean relative expression levels of one Phylostratum/Divergence-Stratum group with all other groups can be plotted analogous to the `PlotMeans()` and `PlotRE()` functions:

```r
# Visualizing the mean relative expression of two Phylostratum groups
```
Here the argument Groups = list(1:3, 4:12) corresponds to dividing Phylostrata 1-12 into Phylostratum groups defined as origin before embryogenesis (group one: PS1-3) and origin during or after embryogenesis (group two: PS4-12). A Kruskal-Wallis Rank Sum Test is then performed to test the statistical significance of the different bars that are compared. The '*' corresponds to a statistically significant difference.

Additionally the ratio between both values represented by the bars to be compared can be visualized as function within the bar plot using the ratio = TRUE argument:

```r
# Visualizing the mean relative expression of two Phylostratum groups
PlotBarRE( ExpressionSet = PhyloExpressionSetExample,
           Groups = list(group_1 = 1:3, group_2 = 4:12),
           ratio = TRUE,
           xlab = "Ontogeny",
           ylab = "Mean Relative Expression",
           cex = 1.5 )
```
It is also possible to compare more than two groups:

# Visualizing the mean relative expression of three Phylostratum groups

```r
PlotBarRE( ExpressionSet = PhyloExpressionSetExample,
    Groups = list(group_1 = 1:3, group_2 = 4:6, group_3 = 7:12),
    wLength = 0.05,
    xlab = "Ontogeny",
    ylab = "Mean Relative Expression",
    cex = 1.5 )
```

For the corresponding statistically significant stages, a Posthoc test can be performed to detect the combinations of differing bars that cause the global statistical significance.

**Visualize age distributions**

Users can visualize the age distributions using the `PlotDistribution()` function.

For this purpose, the `PlotDistribution()` function was implemented:

```r
# Display the phylostratum distribution (gene frequency distribution)
# of a PhyloExpressionSet as absolute frequency distribution
PlotDistribution( PhyloExpressionSet = PhyloExpressionSetExample,
    xlab = "Phylostratum" )
```
or display it as relative frequencies:

```r
# Plot phylostrata as relative frequency distribution
PlotDistribution( PhyloExpressionSet = PhyloExpressionSetExample, 
as.ratio = TRUE, 
xlab = "Phylostratum")
```
Saving Plots on Local Machine

To save plots generated with myTAI on a local machine users can use the following functions implemented in the R language: `png`, `pdf`, `svg`.

```r
# save the TAI profile to a local machine with png()
png("ExampleTAIProfile.png", width = 800, height = 600)

PlotPattern(ExpressionSet = PhyloExpressionSetExample, 
  type = "l",
  lwd = 6,
  xlab = "Ontogeny",
  ylab = "TAI",
  cex = 1,
  cex.lab = 1,
  cex.axis = 1.2 )

dev.off()
```

When using ggplot2 based functions such as `PlotSignature()`, `PlotMeans()`, `PlotRE()`, etc. users can rely on the cowplot package.

```r
# store ggplot2 graphic in variable p
p <- PlotSignature(ExpressionSet = PhyloExpressionSetExample, 
  ylab = "Transcriptome Age Index")
```
More Details about the Statistical Quantification of Transcriptome Conservation Patterns

We discussed three methods to quantify the statistical significance of observed transcriptome conservation patterns:

- Flat Line Test
- Reductive Hourglass Test
- Reductive Early Conservation Test

Here, we will build the test statistic of each test step by step so that future modifications or new test statistics can be built upon the existing methods implemented in the myTAI package.

Details about the Flat Line Test

The Flat Line Test is a permutation test quantifying the statistical significance of an observed phylotranscriptomic pattern. The goal is to detect any evolutionary signal within a developmental time course that significantly deviates from a flat line.

To build the test statistic we start with a standard PhyloExpressionSet. The myTAI package provides an example PhyloExpressionSet named PhyloExpressionSetExample:

```r
library(myTAI)

data(PhyloExpressionSetExample)

# look at the standardized data set format
head(PhyloExpressionSetExample, 3)
```

| Phylostratum | GeneID   | Zygote | Quadrant | Globular | Heart | Torpedo | Bent | Mature |
|--------------|----------|--------|----------|----------|-------|---------|------|--------|
| 1            | at1g01040.2 | 2173.635 | 1911.200 | 1152.555 | 1291.4224 | 1000.253 | 962.9772 | 1696.4274 |
| 2            | at1g01050.1 | 1501.014 | 1817.309 | 1665.309 | 1564.7612 | 1496.321 | 1114.6435 | 1071.6555 |
| 3            | at1g01070.1 | 1212.793 | 1233.002 | 939.200 | 929.6195 | 864.218 | 877.2060 | 894.8189 |

Users will observe that the first column of the PhyloExpressionSetExample stores the Phylostratum assignments of the corresponding genes. The permutation test is based on random sampling of the Phylostratum assignment of genes. The underlying assumption is that the TAI profile of correctly assigned Phylostrata is significantly deviating from TAI profiles based on randomly assigned Phylostrata.

```r
# TAI profile of correctly assigned Phylostrata
TAI(PhyloExpressionSetExample)
```

Visualization:

```r
data(PhyloExpressionSetExample)

# Visualize the TAI profile of correctly assigned Phylostrata
```
PlotSignature( ExpressionSet = PhyloExpressionSetExample, 
  p.value = FALSE )

#> Plot signature: ' TAI ' and test statistic: ' FlatLineTest '.

Visualization:

# Visualize the TAI profile based on randomly assigned Phylostrata
PlotSignature( ExpressionSet = randomPhyloExpressionSetExample, 
  p.value = FALSE )

Users will observe that the visual pattern of the correctly assigned TAI profile and the randomly assigned TAI profile differ qualitatively.

Now we investigate the variance of the two observed patterns.

#> [1] 0.01147725

[1] 0.01147725

#> [1] 0.0005164134

[1] 0.0004102549

We observe that the variance of the randomly assigned TAI profile is much smaller than the variance of the correctly assigned TAI profile. Here we use the variance to quantify the flatness of a given TAI profile.
In theory the variance of a perfect flat line would be zero. So any TAI profile that is close to zero would resemble a flat line. But how exactly are the variances of randomly assigned TAI profiles distributed? For this purpose the `bootMatrix()` function was implemented.

The `bootMatrix()` takes an PhyloExpressionSet or DivergenceExpressionSet as input and computes N TAI or TDI profiles based on randomly assigned Phylostrata or Divergence-Strata.

```
Zygote Quadrant Globular Heart Torpedo Bent Mature
#> 1 3.654682 3.636179 3.637716 3.630673 3.627429 3.585649 3.522012
#> 2 3.471181 3.471773 3.492310 3.481089 3.500058 3.504816 3.481668
#> 3 3.556062 3.556415 3.588464 3.586417 3.611023 3.584442 3.551205
#> 4 3.509540 3.491617 3.483350 3.483370 3.486415 3.531470 3.548840
#> 5 3.583865 3.581659 3.532107 3.526030 3.518202 3.535185 3.570773
#> 6 3.637009 3.619531 3.640805 3.652182 3.664245 3.645142 3.644842
```

Based on this `bootMatrix` we can compute the variance of each random TAI profile.

```
# compute the variance of the random TAI profile for each row
variance_vector <- apply(randomTAIs, 1, var)

# and visualize the distribution of variances
hist(variance_vector, breaks = 100)
```

43
Now it is interesting to see where we can find the variance of the correctly assigned TAI.

```r
# variance of the TAI profile based on correctly assigned Phylostrata
var_real <- var(TAI(PhyloExpressionSetExample))

# visualize the distribution of variances
hist(x = c(variance_vector, var_real), breaks = 100, xlab = "variance", main = "Histogram of variance_vector")

# and plot a red line at the position where we can find the real variance
abline(v = var_real, lwd = 5, col = "red")
```
This plot illustrates that variances based on random TAI profiles seem to have a smaller variance than the variance based on the correct TAI profile. To obtain a p-value that now quantifies this difference, we need to fit the histogram of `variance_vector` with a specific probability distribution.

Visually it would be possible to choose a gamma distribution to fit the histogram of `variance_vector`. To validate this choice a Cullen and Frey graph provided by the `fitdistrplus` package can be used.

```r
# install.packages("fitdistrplus")

# plot a Cullen and Frey graph
fitdistrplus::descdist(variance_vector)
```
Based on the observation that a gamma distribution is a suitable fit for `variance_vector`, we can now estimate the parameters of the gamma distribution that fits the data.

```r
# estimate the parameters: shape and rate using 'moment matching estimation'
gamma_MME <- fitdistrplus::fitdist(variance_vector, distr = "gamma", method = "mme")
# estimate shape:
shape <- gamma_MME$estimate[1]
# estimate the rate:
rate <- gamma_MME$estimate[2]
# define an expression written as function as input for the curve() function
gamma_distr <- function(x){ return(dgamma(x = x, shape = shape, rate = rate)) }

# plot the density function and the histogram of variance_vector
curve( expr = gamma_distr, 
   xlim = c(min(variance_vector), max(c(variance_vector, var_real))),
   col = "steelblue",
   lwd = 5,
   xlab = "Variances",
   col = "steelblue",
   lwd = 5,
   xlab = "Variances",
)
```
Using the gamma distribution with estimated parameters the corresponding p-value of var_real can be computed.

```r
# p-value of var_real
pgamma(var_real, shape = shape, rate = rate, lower.tail = FALSE)
```

```
> [1] 2.319058e-09
```

Hence, the variance of the correct TAI profile significantly deviates from random TAI profiles and this allows us to assume that the underlying TAI profile captures a real evolutionary signal.

**Using the FlatLineTest() Function**

This entire procedure of computing the p-value having the variance of TAI profiles as test statistic is done by the FlatLineTest() function.

```r
# Perform the FlatLineTest
FlatLineTest( ExpressionSet = PhyloExpressionSetExample,
              permutations = 1000 )
```

This function returns the p-value of the test statistic.
Additionally the `FlatLineTest()` function allows to investigate the goodness of the test statistic.

```r
# perform the FlatLineTest and investigate the goodness of the test statistic
FlatLineTest( ExpressionSet = PhyloExpressionSetExample,
              permutations = 1000,
              plotHistogram = TRUE )
```

The `plotHistogram` argument specifies whether analytics plots shall be drawn to quantify the goodness of the test statistic returned by the `FlatLineTest`.

The three resulting plots show:

- a Cullen and Frey graph
- a histogram of the test statistic and the corresponding gamma distribution that was fitted to the test statistic
- a plot showing the p-values (`p_flt`) for 10 individual runs. Since the underlying test statistic is generated by a permutation test, the third plot returned by `FlatLineTest()` shows the influence of different permutations to the corresponding p-value

In other words, to test whether or not the underlying permutation of the permutation test is causing the significance of the p-value, you can specify the `runs` argument within the `FlatLineTest()` function to perform several independent runs. In case there exists a permutation that causes a previous significant p-value to become non-significant, the corresponding phylotranscriptomic pattern shouldn’t be considered as statistically significant.

### Details about the Reductive Hourglass Test

The **Reductive Hourglass Test** has been developed to statistically evaluate the existence of a phylotranscriptomic hourglass pattern based on TAI or TDI computations. The corresponding p-value quantifies the probability that a given TAI or TDI pattern (or any phylotranscriptomics pattern) does not follow an hourglass like shape. A p-value < 0.05 indicates that the corresponding phylotranscriptomics pattern does indeed follow an hourglass (high-low-high) shape.

To build the test statistic again we start with a standard PhyloExpressionSet.

```r
#> Phylostratum GeneID Zygote Quadrant Globular Heart Torpedo Bent Mature
#> 1 1 atig01040.2 2173.635 1911.200 1152.555 1291.4224 1000.253 962.9772 1696.4274
#> 2 1 atig01050.1 1501.014 1817.309 1665.309 1564.7612 1496.321 1114.6435 1071.6555
#> 3 1 atig01070.1 1212.793 1233.002 939.200 929.6195 864.218 877.2060 894.8189
```

And again compute the `TAI()` profile of the `PhyloExpressionSetExample`.

```r
# TAI profile of correctly assigned Phylostrata
TAI(PhyloExpressionSetExample)
```

```r
Zygote Quadrant Globular Heart Torpedo Bent Mature
3.229942 3.225614 3.107135 3.116693 3.073993 3.176511 3.390334
```

Visualization:

```r
# visualize the TAI profile of correctly assigned Phylostrata
PlotSignature( ExpressionSet = PhyloExpressionSetExample,
               p.value = FALSE )
```

48
The Reductive Hourglass Test is a permutation test based on the following test statistic.

1) A set of developmental stages is partitioned into three modules - early, mid, and late - based on prior biological knowledge (see Drost et al., 2015 for details).

2) The mean TAI or TDI value for each of the three modules $T_{\text{early}}$, $T_{\text{mid}}$, and $T_{\text{late}}$ are computed.

3) The two differences $D_1 = T_{\text{early}} - T_{\text{mid}}$ and $D_2 = T_{\text{late}} - T_{\text{mid}}$ are calculated.

4) The minimum $D_{\text{min}}$ of $D_1$ and $D_2$ is computed as final test statistic of the Reductive Hourglass Test.

In order to determine the statistical significance of an observed minimum difference $D_{\text{min}}$ the following permutation test was performed. Based on the $\text{bootMatrix()}$ $D_{\text{min}}$ is calculated from each of the permuted TAI or TDI profiles, approximated by a Gaussian distribution with method of moments estimated parameters returned by $\text{fitdistrplus::fitdist()}$, and the corresponding p-value is computed by $\text{pnorm}$ given the estimated parameters of the Gaussian distribution. The goodness of fit for the random vector $D_{\text{min}}$ is statistically quantified by a Lilliefors (Kolmogorov-Smirnov) test for normality.

To perform the Reductive Hourglass Test you can use the $\text{ReductiveHourglassTest()}$ function. Using this function you need to divide the given phylotranscriptomic pattern into three developmental modules:

- early module
- mid module
- late module

This can be done using the $\text{modules}$ argument: $\text{module} = \text{list}(\text{early} = 1:2, \text{mid} = 3:5, \text{late} = 6:7)$.

In this example ($\text{PhyloExpressionSetExample}$) we divide the corresponding developmental process into the three modules:

- early module: Stages 1 - 2 = Zygote and Quadrant
- mid module: Stages 3 - 5 = Globular, Heart, and Torpedo
- late module: Stages 6 - 7 = Bent and Mature

```
# Perform the Reductive Hourglass Test
\text{ReductiveHourglassTest( ExpressionSet = PhyloExpressionSetExample, }
\text{ modules \hspace{0.2cm} = \text{list(early = 1:2, mid = 3:5, late = 6:7)},}
\text{ lillie.test \hspace{0.2cm} = \text{TRUE} )}
```

The corresponding output shows the p-value returned by the Reductive Hourglass Test, the standard deviation of randomly permuted TAI profiles returned by $\text{bootMatrix()}$ ($\text{apply(bootMatrix(PhyloExpressionSetExample) , 2 , sd)}$) and in case the argument $\text{lillie.test = TRUE}$, a logical value representing the goodness of fit statistic returned by the Lilliefors (Kolmogorov-Smirnov) test for normality. In case $\text{lillie.test}$ is TRUE the corresponding Lilliefors (Kolmogorov-Smirnov) test passed the goodness of fit criterion. In case $\text{lillie.test}$ is FALSE the corresponding goodness of fit by a normal distribution is not statistically significant.

Analogous to the $\text{plotHistogram}$ argument that is present in the FlatLineTest() function, the $\text{ReductiveHourglassTest()}$ function also takes an argument $\text{plotHistogram}$. When $\text{plotHistogram} = \text{TRUE}$, the $\text{ReductiveHourglassTest()}$ function returns a multi-plot showing:

- A Cullen and Frey skewness-kurtosis plot. This plot illustrates which distributions seem plausible to fit the resulting permutation vector $D_{\text{min}}$. Here a normal distribution seems most plausible.
- A histogram of $D_{\text{min}}$ combined with the density plot is visualized. $D_{\text{min}}$ is then fitted by a normal distribution. The corresponding parameters are estimated by moment matching estimation.
- A plot showing the p-values for N independent runs to verify that a specific p-value is biased by a specific permutation order.
- A bar plot showing the number of cases in which the underlying goodness of fit (returned by Lilliefors (Kolmogorov-Smirnov) test for normality) has shown to be significant (TRUE) or not significant (FALSE). This allows to quantify the permutation bias and their implications on the goodness of fit.
# perform the Reductive Hourglass Test and plot the test statistic

```r
ReductiveHourglassTest( ExpressionSet = PhyloExpressionSetExample,
modules = list(early = 1:2, mid = 3:5, late = 6:7),
plotHistogram = TRUE,
lillie.test = TRUE )
```

![Cullen and Frey graph](image)

The corresponding output shows the summary statistics of the fitted normal distribution as well as the p-value, standard deviation, and Lilliefors (Kolmogorov-Smirnov) test result.

This example output nicely illustrates that although the Lilliefors (Kolmogorov-Smirnov) test for normality is violated for some permutations, the Cullen and Frey graph shows that there is no better approximation than a normal distribution (which is also supported visually by investigating the fitted frequency distribution). The corresponding p-value returned by `ReductiveHourglassTest()` is significant and illustrates that the observed
The Early Conservation Test has been developed to statistically evaluate the existence of a monotonically increasing phylotranscriptomic pattern based on TAI or TDI computations. The corresponding p-value quantifies the probability that a given TAI or TDI pattern (or any phylotranscriptomic pattern) does not follow an early conservation like pattern. A p-value < 0.05 indicates that the corresponding phylotranscriptomics pattern does indeed follow an early conservation (low-high-high) shape.

To build the test statistic again we start with a standard PhyloExpressionSet.

```r
library(myTAI)

# load an example PhyloExpressionSet stored in the myTAI package
data(PhyloExpressionSetExample)

# look at the standardized data set format
head(PhyloExpressionSetExample, 3)

#> Phylostratum GeneID Zygote Quadrant Globular Heart Torpedo Bent Mature
#> 1 1 at1g01040.2 2173.635 1911.200 1152.555 1291.4224 1000.253 962.9772 1696.4274
#> 2 1 at1g01050.1 1501.014 1817.309 1665.309 1564.7612 1496.321 1114.6435 1071.6555
#> 3 1 at1g01070.1 1212.793 1233.002 939.200 929.6195 864.218 877.2060 894.8189

And again compute the TAI() profile of the PhyloExpressionSetExample.

```r
# TAI profile of correctly assigned Phylostrata
TAI(PhyloExpressionSetExample)

Zygote Quadrant Globular Heart Torpedo Bent Mature
3.229942 3.225614 3.107135 3.116693 3.073993 3.176511 3.390334

Visualization:

```r
# Visualize the TAI profile of correctly assigned Phylostrata
PlotSignature( ExpressionSet = PhyloExpressionSetExample,
               p.value = FALSE )

#> Plot signature: ' TAI ' and test statistic: ' FlatLineTest '.
The reductive early conservation test is a permutation test based on the following test statistic.

1) A set of developmental stages is partitioned into three modules - early, mid, and late - based on prior biological knowledge.

2) The mean TAI or TDI value for each of the three modules $T_{\text{early}}$, $T_{\text{mid}}$, and $T_{\text{late}}$ are computed.

3) The two differences $D_1 = T_{\text{mid}} - T_{\text{early}}$ and $D_2 = T_{\text{late}} - T_{\text{early}}$ are calculated.

4) The minimum $D_{\text{min}}$ of $D_1$ and $D_2$ is computed as final test statistic of the Reductive Early Conservation Test.

In order to determine the statistical significance of an observed minimum difference $D_{\text{min}}$ the following permutation test was performed. Based on the `bootMatrix()` $D_{\text{min}}$ is calculated from each of the permuted TAI or TDI profiles, approximated by a Gaussian distribution with method of moments estimated parameters returned by `fitdistrplus::fitdist()`, and the corresponding p-value is computed by `pnorm` given the estimated parameters of the Gaussian distribution. The goodness of fit for the random vector $D_{\text{min}}$ is statistically quantified by an Lilliefors (Kolmogorov-Smirnov) test for normality.

To perform the Reductive Early Conservation Test you can use the `EarlyConservationTest()` function. Using this function you need to divide the given phylotranscriptomics pattern into three developmental modules:

- early module
- mid module
- late module

This can be done using the `modules` argument: `module = list(early = 1:2, mid = 3:5, late = 6:7).

In this example (PhyloExpressionSetExample) we divide the corresponding developmental process into the three modules:

- early module: Stages 1 - 2 = Zygote and Quadrant
• mid module: Stages 3 - 5 = Globular, Heart, and Torpedo
• late module: Stages 6 - 7 = Bent and Mature

# Perform the Reductive Early Conservation Test

```r
EarlyConservationTest( ExpressionSet = PhyloExpressionSetExample,
                        modules = list(early = 1:2, mid = 3:5, late = 6:7),
                        lillie.test = TRUE )
```

```r
#> $p.value
#> [1] 0.9998982
```

```r
#> $std.dev
#> [1] 0.05524279 0.05374393 0.05205015 0.05040596 0.04988670 0.05261761 0.05809457
```

```r
#> $lillie.test
#> [1] FALSE
```

Analogous to the `plotHistogram` argument that is present in the `FlatLineTest()` and `ReductiveHourglassTest()` function, the `EarlyConservationTest()` function also takes an argument `plotHistogram`. When `plotHistogram = TRUE`, the `EarlyConservationTest()` function returns a multi-plot showing:

- A Cullen and Frey skewness-kurtosis plot. This plot illustrates which distributions seem plausible to fit the resulting permutation vector $D_{min}$. Again a normal distribution seems most appropriate.

- A histogram of $D_{min}$ combined with the density plot is visualized. $D_{min}$ is then fitted by a normal distribution. The corresponding parameters are estimated by moment matching estimation.

- A plot showing the p-values for N independent runs to verify that a specific p-value is biased by a specific permutation order.

- A bar plot showing the number of cases in which the underlying goodness of fit (returned by Lilliefors (Kolmogorov-Smirnov) test for normality) has shown to be significant (TRUE) or not significant (FALSE). This allows to quantify the permutation bias and their implications on the goodness of fit.

# perform the Reductive Early Conservation Test and plot the test statistic

```r
EarlyConservationTest( ExpressionSet = PhyloExpressionSetExample,
                        modules = list(early = 1:2, mid = 3:5, late = 6:7),
                        plotHistogram = TRUE,
                        lillie.test = TRUE )
```

53
This example output nicely illustrates that although the Lilliefors (Kolmogorov-Smirnov) test for normality is violated, the Cullen and Frey graph shows that there is no better approximation than a normal distribution (which is also supported visually by investigating the fitted frequency distribution). The corresponding p-value returned by the `EarlyConservationTest()` is highly non-significant and illustrates that the observed phylotranscriptomic pattern of `PhyloExpressionSetExample` does not follow the Early Conservation Model assumption.

This example shall illustrate that finding the right test statistic is a multi-step process of investigating different properties of the underlying permutation test. Although single aspects might fit or fit not corresponding criteria, the overall impression (sum of all individual analyses) must be considered to obtain a valid p-value.
**Data Transformations**

Motivated by the discussion raised by Piasecka et al., 2013, the influence of gene expression transformation on the global phylotranscriptomic patterns does not seem negligible. Hence, different transformations can result in qualitatively different TAI or TDI patterns.

Initially, the TAI and TDI formulas were defined for absolute expression levels. So using the initial TAI and TDI formulas with transformed expression levels can result in qualitatively different patterns when compared with non-transformed expression levels, but might also belong to a different class of models, since different valid expression level transformation functions result in different patterns.

The purpose of the `tf()` function is to allow users to study the qualitative impact of different transformation functions on the global TAI and TDI pattern, or on any subsequent phylotranscriptomic analysis.

The examples using the `PhyloExpressionSetExample` data set show that using common gene expression transformation functions: log2 (Quackenbush, 2001 and 2002), sqrt (Yeung et al., 2001), boxcox, or inverse hyperbolic sine transformation, each transformation results in qualitatively different patterns. Nevertheless, for each resulting pattern the statistical significance can be tested using either the `FlatLineTest()`, `ReductiveHourglassTest()`, or `EarlyConservationTest()` (Drost et al., 2015) to quantify the significance of observed patterns.

The `tf()` function takes a standard `PhyloExpressionSet` or `DivergenceExpressionSet` and transformation function and returns the corresponding `ExpressionSet` with transformed gene expression levels.

```r
library(myTAI)
data(PhyloExpressionSetExample)

# a simple example is to transform the gene expression levels of a given PhyloExpressionSet
# using a sqrt or log2 transformation

PES.sqrt <- tf(PhyloExpressionSetExample, sqrt)
head(PES.sqrt)

Phylostratum GeneID Zygote Quadrant Globular Heart Torpedo Bent Mature
1 1 at1g01040.2 46.6226 43.71728 33.94930 31.62678 31.03187 41.18771
2 1 at1g01050.1 38.74292 42.62990 40.80820 39.55706 38.68230 33.38628 32.73615
3 1 at1g01070.1 34.82517 35.11413 30.64637 30.48966 29.39759 26.61766 25.91352
4 1 at1g01080.2 31.88919 30.60039 34.37060 36.46195 37.31813 35.88369 29.34724
5 1 at1g01090.1 106.88576 129.53057 185.38244 199.43831 237.13197 258.80566 88.16213
6 1 at1g01120.1 29.05239 28.06409 29.31939 30.52242 30.70579 29.50021 28.15589

PES.log2 <- tf(PhyloExpressionSetExample, log2)
head(PES.log2)

Phylostratum GeneID Zygote Quadrant Globular Heart Torpedo Bent Mature
1 1 at1g01040.2 11.08589 10.900263 10.170620 10.334745 9.966149 9.911358 10.728284
2 1 at1g01050.1 10.551722 10.827588 10.701574 10.611727 10.547204 10.122367 10.065625
3 1 at1g01070.1 10.244117 10.267960 9.875289 9.860497 9.755251 9.776772 9.805452
4 1 at1g01080.2 9.989991 9.870956 10.206206 10.376639 10.443610 10.330888 9.750306
5 1 at1g01090.1 106.88576 129.53057 185.38244 199.43831 237.13197 258.80566 88.16213
6 1 at1g01120.1 9.721170 9.621306 9.747567 9.863595 9.880877 9.765307 9.630730
```

# in case a given PhyloExpressionSet already stores gene expression levels
# that are log2 transformed and need to be re-transformed to absolute
# expression levels, to perform subsequent phylotranscriptomics analyses
(that are defined for absolute expression levels), one can re-transform a PhyloExpressionSet like this:

```r
PES.absolute <- tf(PES.log2 , function(x) 2^-x)
```

# which should be the same as PhyloExpressionSetExample :

```r
head(PhyloExpressionSetExample)
head(PES.absolute)
```

> head(PhyloExpressionSetExample)

| Phylostratum | GeneID      | Zygote   | Quadrant | Globular | Heart    | Torpedo | Bent    | Mature   |
|--------------|-------------|----------|----------|----------|----------|---------|---------|----------|
| 1            | at1g01040.2 | 2173.6352| 1911.2001| 1152.5553| 1291.4224| 1000.2529| 962.9772| 1696.4274|
| 2            | at1g01050.1 | 1501.0141| 1817.3086| 1665.3089| 1564.7612| 1496.3207| 1114.6435| 1071.6555|
| 3            | at1g01070.1 | 1212.7927| 1233.0023| 939.2000 | 929.6195 | 864.2180 | 877.2060 | 894.8189 |
| 4            | at1g01080.2 | 1016.9203| 936.3837  | 1181.3381| 1329.4734| 1392.6429| 1287.9746| 861.2605 |
| 5            | at1g01090.1 | 11424.5667| 16778.1685| 34366.6493| 39775.6405| 56231.5689| 66980.3673| 7772.5617|
| 6            | at1g01120.1 | 844.0414 | 787.5929  | 859.6267 | 931.6180 | 942.8453 | 870.2625 | 792.7542 |

When transforming the ExpressionMatrix of the PhyloExpressionSetExample using different transformation functions, the resulting phylotranscriptomic patterns qualitatively differ:

**log2 transformation (TAI)**

```r
data(PhyloExpressionSetExample)
# plotting the TAI using log2 transformed expression levels
# and performing the Flat Line Test to obtain the p-value
PlotSignature( ExpressionSet = tf(PhyloExpressionSetExample, log2),
              TestStatistic = "FlatLineTest",
              xlab = "Ontogeny",
              ylab = "TAI" )
```

>` Plot signature: ' TAI ' and test statistic: ' FlatLineTest '.  
> Significance status of signature: significant.
sqrt transformation (TAI)

data(PhyloExpressionSetExample)
# plotting the TAI using sqrt transformed expression levels
# and performing the Flat Line Test to obtain the p-value
PlotSignature( ExpressionSet = tf(PhyloExpressionSetExample, sqrt),
    TestStatistic = "FlatLineTest",
    xlab = "Ontogeny",
    ylab = "TAI" )

#> Plot signature: ' TAI ' and test statistic: ' FlatLineTest '.
#> Significance status of signature: significant.
For the `PhyloExpressionSetExample` all transformations result in a significant phylotranscriptomics pattern deviating from a flat line.

Nevertheless, it is not clear which transformation is the most appropriate one since the original TAI and TDI measure were defined for absolute expression levels.

The same accounts for TDI profiles:

**log2 transformation (TDI)**

```r
data(DivergenceExpressionSetExample)
# plotting the TDI using log2 transformed expression levels
# and performing the Flat Line Test to obtain the p-value
PlotSignature( ExpressionSet = tf(DivergenceExpressionSetExample, log2),
               TestStatistic = "FlatLineTest",
               xlab = "Ontogeny",
               ylab = "TDI" )
```

> Plot signature: ' TAI ' and test statistic: ' FlatLineTest '.
> Significance status of signature: significant.
sqrt transformation (TDI)

data(DivergenceExpressionSetExample)
# plotting the TDI using sqrt transformed expression levels
# and performing the Flat Line Test to obtain the p-value
PlotSignature( ExpressionSet = tf(DivergenceExpressionSetExample, sqrt),
TestStatistic = "FlatLineTest",
xlab = "Ontogeny",
ylab = "TDI")

#> Plot signature: 'TAI' and test statistic: 'FlatLineTest'.
#> Significance status of signature: significant.
As a result, observed patterns should always be quantified using statistical tests (for ex. FlatLineTest(), ReductiveHourglassTest(), and EarlyConservationTest()). In case the observed pattern is significant, qualitative differences of the observed patterns based on different data transformations must be investigated in more detail, since most data transformations are known to cause different effects on a measure that isn’t robust against data transformations.

References

Domazet-Loso, T. & Tautz, D. A phylogenetically based transcriptome age index mirrors ontogenetic divergence patterns. *Nature* **468**, 815-8 (2010).

Quint, M. *et al.* A transcriptomic hourglass in plant embryogenesis. *Nature* **490**, 98-101 (2012).

Drost, H.G. *et al.* Evidence for Active Maintenance of Phylotranscriptomic Hourglass Patterns in Animal and Plant Embryogenesis. *Mol. Biol. Evol.* **32** (5), 1221-1231 (2015).

Piasecka, B. *et al.* The hourglass and the early conservation models - co-existing patterns of developmental constraints in vertebrates. *PLoS Genet.* **9**:e1003476 (2013).

Quackenbush, J. Computational Analysis of Microarray Data. *Nature Reviews*. **2**, 418-427 (2001).

Quackenbush, J. Microarray data normalization and transformation. *Nature Genetics*. **32**, 496-501 (2002).

Yeung, K.Y. *et al.* Model-based clustering and data transformations for gene expression data. *Bioinformatics*. **17**, 977-987 (2001).

Yeung, K.Y. *et al.* Supplement to Model-based clustering and data transformations for gene expression data - Data Transformations and the Gaussian mixture assumption. *Bioinformatics*. **17**, 977-987 (2001).