Tree-based differential testing using inferential uncertainty for RNA-Seq

Noor Pratap Singh[1], Euphy Y. Wu[2], Jason Fan[1], Michael I. Love[2,3], and Rob Patro[1]

npsingh@cs.umd.edu, euphyw@live.unc.edu, jasonfan@cs.umd.edu, michaelisaiahlove@gmail.com, rob@cs.umd.edu

[1]*Department of Computer Science, University of Maryland, College Park*

[2]*Department of Biostatistics, University of North Carolina-Chapel Hill*

[3]*Department of Genetics, University of North Carolina-Chapel Hill*

[*]*Corresponding author: Rob Patro, rob@cs.umd.edu*

December 25, 2023

Identifying differentially expressed transcripts poses a crucial yet challenging problem in transcriptomics. Substantial uncertainty is associated with the abundance estimates of certain transcripts which, if ignored, can lead to the exaggeration of false positives and, if included, may lead to reduced power. For a given set of RNA-Seq samples, `TreeTerminus` arranges transcripts in a hierarchical tree structure that encodes different layers of resolution for interpretation of the abundance of transcriptional groups, with uncertainty generally decreasing as one ascends the tree from the leaves. We introduce `trenDi`, which utilizes the tree structure from `TreeTerminus` for differential testing. The candidate nodes are determined in a data-driven manner to maximize the signal that can be extracted from the data while controlling for the uncertainty associated with estimating the transcript abundances. The identified candidate nodes can include transcripts and inner nodes, with no two nodes having an ancestor/descendant relationship. We evaluated our method on both simulated and experimental datasets, comparing its performance with other tree-based differential methods as well as with uncertainty-aware differential transcript/gene expression methods. Our method detects inner nodes that show a strong signal for differential expression, which would have been overlooked when analyzing the transcripts alone.

1 Introduction

RNA-Seq has become the *de facto* technology for measuring the expression profiles of different genomic features. Finding the features that differ in expression between the biological samples under different conditions, such as normal versus tumor samples, is called differential analysis and is a fundamental task downstream of quantification. The starting point of differential analysis is to accurately estimate the abundance of the features to be tested, with genes and transcripts being the most common features of interest for most RNA-Seq analysis. A gene can express multiple isoforms or transcripts due to alternative splicing, where the transcripts can comprise an overlapping set of exons and thus share sequences. Commonly used RNA-Seq quantification methods such as `RSEM` [Li and Dewey, 2011], `salmon` [Patro et al., 2017] and `kallisto` [Bray et al., 2016] probabilistically assign each observed fragment to the transcripts using maximum likelihood or Bayesian inference methods. A probabilistic model is required because a substantial fraction of the sequenced reads can multi-map (align similarly or equally-well to multiple reference transcripts) as they arise from the regions and sequences shared between the transcripts. Thus, there is ambiguity concerning the true locus of origin of such reads, leading to uncertainty when trying to quantify transcript abundances. Uncertainty can also exist for gene abundance estimates when the reads belong to shared sequences within the genes (for example, homologous genes); however, in general, gene-level estimates will be more precise compared to transcripts. One approach that estimates the uncertainty associated with the measurement of the abundance estimates is to generate additional samples called inferential replicates through different sampling strategies. These include bootstrap sampling or posterior sampling using MCMC/Gibbs sampling. Such capabilities are provided by several existing quantification methods [Li and Dewey, 2011, Bray et al., 2016, Patro et al., 2017, Turro et al., 2011].

The robustness and accuracy of any downstream analysis, such as differential testing, is directly impacted by the quality of the abundance estimates. The methods designed for differential transcript expression testing that utilize inferential replicates report a more robust performance than the methods that do not include them [Seesi et al., 2014, Mandric et al., 2017, Pimentel et al., 2017, Zhu et al., 2019, Baldoni et al., 2023]. However, if inferential replicates are incorporated, then we might observe a reduced power for the transcripts that exhibit high uncertainty since we might not be confident of the observed differential signal. In such cases, we might be able to improve the differential signal by aggregating transcripts that share reads together into a transcript group. This idea was first proposed in `mmcollapse` [Turro et al., 2014] and further expanded in `Terminus` [Sarkar et al., 2020], where these groups that contain multiple transcripts, rather than single transcripts, form the feature set for differential analysis.

Expanding upon the above motivation, we recently published `TreeTerminus` [Singh et al., 2023], which outputs a forest of transcript trees for the samples in an RNA-Seq experiment. The leaves represent the transcripts, and the in-

ner nodes represent the aggregated group of the constituent transcripts rooted at each node. The tree encodes different layers of resolution for the interpretation of the abundance of transcriptional groups, with uncertainty generally decreasing as the tree is ascended. This tree can be utilized to find differentially expressed nodes between conditions of interest, where the nodes can consist of both leaves (transcripts) and inner nodes (transcript groups). It is possible that if the differential signal exists at a transcript, then it might also propagate at the inner nodes. A finer resolution, i.e., being as close as possible to the leaves, is, however, preferable for many analyses. The inner nodes should be the output when they constitute the subsets of transcripts whose signal gets masked, primarily due to high uncertainty. Thus, a tree-based differential testing method should be able to find the candidate nodes in a data-driven manner.

Several existing tree-based differential testing methods differ in the error rates they control, the underlying objective functions they seek to optimize, and the data modalities they intend to target. One class of methods focuses on testing for the global null hypothesis, meaning that for a given inner node, the null hypothesis for all the descendant leaves is true [Miecznikowski and Wang, 2023]. These methods use a top-down strategy where they descend an inner node only if the null hypothesis at that node is rejected and continue testing until the null hypothesis cannot be rejected or a leaf node is reached [Meinshausen, 2008, Goeman and Solari, 2010, Yekutieli, 2008, Lynch and Guo, 2016, Bogomolov et al., 2021]. While [Meinshausen, 2008, Goeman and Solari, 2010] control the family-wise error rate (FWER) on the rejected set, [Yekutieli, 2008, Lynch and Guo, 2016] control the false discovery rate (FDR), and [Bogomolov et al., 2021] propose and control a new error rate called selective false discovery rate, which is the expectation across level-specific false discovery proportion (FDP). Applying these methods can lead to accepting the null hypothesis at a node in the tree, which can consist of children that contain the differential signal but exhibit a contrasting direction of sign change between the conditions of interest. This is very likely to happen when we do the hypothesis testing or compute the p-value at each node of the tree individually. We might even end up not rejecting the null hypothesis at the root node, even if there is strong evidence of the signal existing at a node situated at a lower height in the tree.

`BOUTH` [Li et al., 2022] proposes an alternative to the above approaches by defining a new hypothesis test called the modified null hypothesis and controlling an error rate called the false selection rate. It employs a bottom-up procedure and examines whether, among descendant nodes not previously detected at the lower level, none are associated. It identifies and outputs "driver" nodes such that none of its ancestor nodes are associated. If the driver nodes were to be analyzed, then the features might provide a coarse-grained level of analysis, especially if a clearer signal already exists at a lower height in the tree. On the other hand, it is not clear which descendant nodes corresponding to a driver node should be analyzed to provide a finer resolution of analysis.

Another category of methods aims to leverage the tree structure to enhance power and control the FDR at the leaf level [Xiao et al., 2017, Huang et al., 2021, Bichat et al., 2022]. `StructFDR` [Xiao et al., 2017]) and `zazou` [Bichat et al.,

2022] initially convert the p-values into z-scores and then propose smoothing on the leaves using a tree-based correlation structure. `treeclimbR` [Huang et al., 2021] proposes multiple candidate sets, performs multiplicity correction on each set, and selects a final candidate set based on a set of criteria. It rejects the null hypothesis for all the leaves belonging to the inner nodes in the selected candidate set. However, there is uncertainty in measuring the abundance estimate of the transcripts, and thus we do not know which transcript corresponding to the node may be differentially expressed. Thus, passing the information from an inner node to all the descendant leaves might lead to an exaggeration of the false positives or missing the branches where there exists a signal at a node located at a higher height in the tree, to control for the false positives on the leaf set.

There exists another class of methods that fit regression models to compositional data for the covariates using a tree-guided regularization in a maximum likelihood (`trac` - [Bien et al., 2021]) or Bayesian setting (`tascCODA` - [Ostner et al., 2021]). For `trac`, the parsimony of feature selection (where more nodes at greater heights result in a lesser number of total nodes) depends on the weight assigned to regularization during model fitting. For both `trac` and `tascCODA`, when an inner node is selected, its learned coefficients are passed equally to its underlying leaves. In addition to some of the limitations mentioned above that also exist for these methods, both of these methods do not produce p-values. This can complicate comparisons with other methods. `trac` requires cross-validation to determine appropriate hyperparameters for the model, thus requiring a substantial number of samples, which may not be feasible for many RNA-Seq experiments. `tascCODA` also does not scale computationally for datasets that contain a large number of features, such as RNA-Seq.

To overcome some of the above limitations, we introduce our differential testing method, `trenDi`, designed to identify differential candidate nodes from a tree. The candidate node can comprise both transcripts and inner nodes, with no two nodes having an ancestor/descendant relationship. It utilizes the tree obtained from `TreeTerminus`, and applies the `Swish` method for hypothesis testing, accounting for inferential uncertainty (though it is conceptually compatible with other differential testing methods that incorporate inferential uncertainty). The candidate nodes are determined in a data-driven fashion by traversing the tree in a top-down manner using a set of rules to maximize the signal that can be extracted from the data while controlling for the uncertainty associated with measuring the abundance estimates. The selected inner node represents a subset of transcripts whose signals are masked at lower levels due to inferential uncertainty, and thus we do not report the differential status of the constituent transcripts. We applied our method to both experimental and simulated datasets, comparing its performance with the other tree-based differential testing methods and uncertainty-aware differential transcript/gene estimation methods. Our method demonstrates improved sensitivity compared to alternative approaches, particularly when the features consist only of transcripts or transcript groups. It identifies inner nodes exhibiting a strong signal for differential expression between the conditions, which might be overlooked

when examining their underlying transcripts alone. The differential signal can often be restricted to only a subset of the annotated isoforms of a gene. Thus, our approach also mitigates the potential over-aggregation of signals if reporting is done at the gene level. The `R` package implementing `trenDi` is freely-available and open-source (under the MIT license) and can be obtained from https://github.com/NPSDC/trenDi.

2 Materials and Methods

2.1 Preliminaries

TreeTerminus - For a given RNA-Seq experiment consisting of M samples, `TreeTerminus` [Singh et al., 2023] outputs a forest of K trees $\mathcal{T} = \{T_1, T_2, ..., T_K\}$, where the leaves of the individual trees comprise the set of quantified transcripts and each internal node n represents an aggregation of the set of transcripts belonging to the subtree rooted at n. The input to `TreeTerminus` is the `salmon` [Patro et al., 2017] quantification estimates, l inferential replicates, and range-factorized equivalence classes corresponding to each RNA-Seq sample m, $m \in \{1, ..., M\}$. The l inferential replicates are produced either through Gibbs sampling or bootstrap sampling, and are denoted by $\mathcal{I}_m = \{I_{m1}, I_{m2}, ..., I_{ml}\}$. The transcripts are arranged in a manner that encodes different layers of resolution for interpretation of the abundance of transcriptional groups, with uncertainty generally decreasing as one ascends up the tree from the leaves. The uncertainty for any node (leaf or inner node) is estimated using the `infRV` metric as defined in [Zhu et al., 2019], over $\mathcal{I}$ as:

$$\text{infRV}_i = \frac{\max(\sigma^2_{I_j} - \mu_{I_j}, 0)}{\mu_{I_j} + pc} + d \tag{1}$$

where μ_{I_j}, σ_{I_j} are the mean, variance over the inferential replicates, pc is a pseudocount (with a default value of 5) and d is a small global shift (with a default value of 0.01).

Unified Tree - Once the forest from `TreeTerminus` is obtained, a unified tree is constructed. For the sake of simplicity, let $\mathcal{T}$ denote the unified tree. The unified tree is constructed using the `R` package beaveR [Singh, 2023]. This tree is constructed by first creating a new root node and assigning all trees in the forest and the remaining transcripts in the transcriptome not covered by the trees as children of this root node.

2.2 Problem Statement

For the given tree $\mathcal{T}$, constructed on the set of P transcripts with all nodes including the leaves labelled as $\mathcal{N} = \{1, ..., N\}$ and $N \geq P$, we want to output a set of candidate nodes $\mathcal{C} \subset \mathcal{N}$ which are differentially expressed between the conditions of interest. We want to leverage the tree to find candidate nodes that may consist of both transcripts and inner nodes, with the inner nodes

determined in a data-dependent manner to maximize the signal from the data while controlling for uncertainty. It is important to note that when `trenDi` reports an inner node as a candidate, nothing specific is implied about the differential status of its underlying descendant transcripts. A candidate inner node might suggest that either the signal did exist in at least one underlying transcript, but was masked due to uncertainty, or that the individual signal was weaker and became stronger upon aggregation. Further, any two nodes in the candidate set do not share an ancestor/descendant relationship, i.e., for any two nodes n_i, n_j in $\mathcal{C}, \Lambda(n_i) \cap \Lambda(n_j) = \emptyset$, where $\Lambda(n_i)$ represents the set of transcripts that are the descendants of the node n_i in the tree $\mathcal{T}$. The inputs to `trenDi` are the tree $\mathcal{T}$, and for all the nodes N, their p-values for differential testing between the conditions, mean inferential relative variance (meanInfRV) across the samples, and the direction of sign-change between conditions.

2.3 trenDi description

We will now outline our method, `trenDi`. We first compute the p-values, meanInfRV and direction of sign change for all the nodes in the tree. Tree nodes with p-values below a specified threshold (`pthresh`) are identified and labelled as significant. The set of candidate nodes, denoted as $\mathcal{C}$, will be a subset of the significant nodes. `trenDi` implements a top-down procedure that first determines the branches that contain at least one significant node. Proceeding with this approach, it traverses each selected branch and, upon encountering a significant inner node, checks for the following criteria:

1. It has at least one non-significant node among its children.
2. The descendant nodes that display a confident sign change have the same direction of sign change between the conditions of interest. A node is assigned a confident sign change if that same sign change occurs in at least a certain proportion of the inferential replicates.
3. All the child nodes have a meanInfRV above a certain threshold (`mthresh`).

If all three criteria are met, then that node is selected as a candidate, and traversal along this branch terminates. Otherwise, this branch is traversed (recursively), checking the above criteria on each newly encountered significant node until either the node is selected or a leaf is reached. If a leaf is significant, it is added to the list of candidate nodes. A small, illustrative example demonstrating the above algorithm is provided in Figure 1.

2.3.1 Explanation of the different criteria

The main intuition behind `trenDi` is to aggregate transcripts in a data-driven manner to recover the signal that could have been lost due to uncertainty. The first criterion ensures a finer resolution of analysis since, if all children of a significant node are also significant, then the signal already exists at a lower level and aggregation is unnecessary. The second criterion helps to find

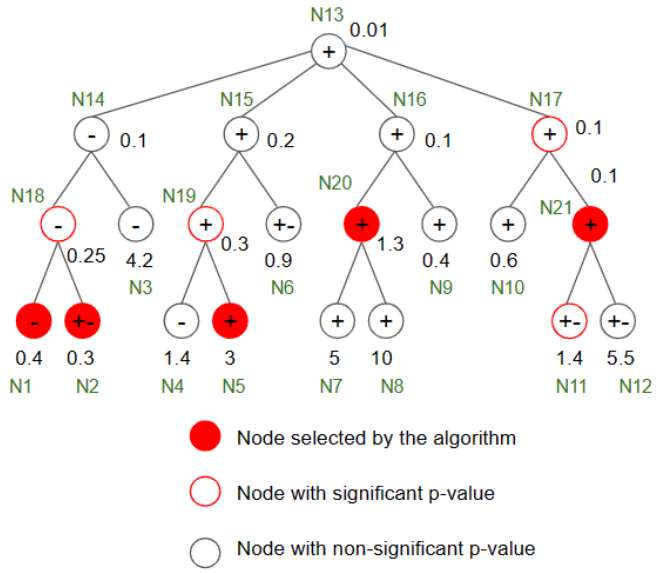

Figure 1: Toy example using a tree with 12 leaves to demonstrate `trenDi`. A highlighted node means that it is selected by `trenDi`, with a red outlined node indicating that the node has a significant p-value. The $+, -$ on a node indicates the direction of sign change between the conditions, whereas $+-$ indicates we are unsure about the direction. The values plotted along the nodes represent their mean inferential relative variance across the samples (meanInfRV). The meanInfRV threshold in this example is 0.4. The nodes N1 and N2 are chosen instead of node N18 since the meanInfRV of one of its children, node N2, is below the threshold of 0.4. The node N5 is chosen instead of the node N19 since the children have opposite sign changes. Node N20 is the only significant node in its branch and meets all the required criteria. Node N21 is picked since it has at least one non-significant child node.

a candidate node that has a consistent direction of the sign change and thus also aids in producing a clearer biological interpretation. Owing to uncertainty, the direction of the sign change of a node can be ambiguous, as it might not be consistent across the inferential replicates. We thus assign a particular direction of sign change (positive/up, negative/down) to the nodes between the conditions of interest, if we are confident. A node is assigned a confident direction of sign change if the same direction appears in at least a certain proportion of the inferential replicates (0.70 by default). Otherwise, the node is assigned an uncertain sign change. Thus, the candidate node consists of descendants where all the nodes whose direction of sign change has been confidently determined are in the same direction, and for the remaining nodes, the direction is not assigned. The third criterion checks for over-aggregation, since if we are certain about the abundance estimates for a group, then we are confident about the assessment of its differential signal.

2.3.2 Computing the p-values

`trenDi` can take p-values produced by any differential analysis tool as input. However, in this study, we have used `Swish` [Zhu et al., 2019] for computing the p-values, since it takes inferential uncertainty into account. The p-values for the leaves and inner nodes are computed separately. The reasoning behind this choice is that the leaves have higher uncertainty compared to the inner nodes, especially the ones that appear higher in the tree (the height of a node refers to the maximum distance between the node and its underlying leaves). This leads to a relatively lower shrinkage of the test-statistic towards 0 for the inner nodes and increases the width of the distribution of the test-statistic used by `Swish` for computing the p-values for the inner nodes compared to those of the leaves. As a result, the p-values of the inner nodes will be smaller, on average, compared to the leaves, when the p-values are computed together (see Section 3.1.1). The p-value threshold that is used for determining the significance of a node is computed by applying a multiple-testing procedure such as Benjamini-Hochberg (`BH`) [Benjamini and Hochberg, 1995] on the leaf nodes that controls the FDR on the leaves at rate α.

2.4 Datasets

We evaluate `trenDi` on four different datasets, two simulated and two experimental, that span different reference organisms.

2.4.1 Simulated human datasets

We used the pipeline as defined in [Love et al., 2018] to create the simulated datasets. Polyester [Frazee et al., 2015] was used to generate the `FASTQ` files corresponding to the paired-end reads for each simulated dataset consisting of 2 groups with 12 samples in each. To simulate samples with realistic ground-truth transcript-level expression, TPM estimates were extracted from the GTEx V8 frontal cortex dataset. The TPM estimates were then used to create the count matrix (`sim.counts.mat`), which was provided as an input to Polyester to simulate the reads. The reads were simulated with a realistic fragment GC bias. The dispersion of transcript-level counts is drawn from the joint distribution of mean and dispersion values estimated from GEUVADIS samples [Lappalainen et al., 2013]. 80% of genes were set to be null genes, with no change in abundance between the two groups. For 10% of the genes, all transcripts were differentially expressed, with all the transcripts belonging to a gene having the same fold change. For the remaining 10% of the genes, a single expressed transcript was selected to be differentially expressed. We created two variations of this simulation by varying the range of fold change: in the first variation, we keep the same range as in [Love et al., 2018] from 2 to 6; and in the second variation, the range is kept between 1.4 and 2.8. The first variation is referred to as `BrSimNorm` and the second is referred to as `BrSimLow`. The second simulation is created to measure the change in performance across the methods when the magnitude of effect sizes between the transcripts is decreased.

2.4.2 Mouse muscle dataset

We also analyzed a dataset taken from the skeletal muscle study `GSE100505` [Terry et al., 2018]. We downloaded RNA-seq data for the 6 samples belonging to `Masseter` (MAST) and the 6 to `Extensor digitorum longus` (EDL), with the accession numbers provided in Table S2. All the samples belong to the organism *Mus musculus*. This dataset is referred to as `MouseMuscle`.

2.4.3 Chimpanzee brain dataset

The final dataset that has been analyzed in this study is the RNA-Seq data from [Sousa et al., 2017], with SynapseID `syn7067053`, collected from 5 Chimpanzees (*Pan Troglodyte*). We refer to this dataset as `ChimpBrain`. We specifically analyze samples obtained from the `cerebellum` and `medial dorsal nucleus of thalamus` tissues of the brain. The batch effects were observed for the specimen ID when visualizing the first two dimensions of the PCA obtained on the normalized counts matrix. We used `svaseq` [Leek, 2014] on the normalized counts using two surrogate variables to create a fit. The fit was then used to correct batch effects using the `removeBatchEffect` function in `limma` [Ritchie et al., 2015], for each scaled and log-transformed inferential replicate. The exponential function was then applied to the batch-effect-corrected inferential replicates, converting counts back to the original scale.

2.5 Experimental Setup

2.5.1 Pipeline to generate trees

For the `ChimpBrain` dataset, only `BAM` files were available as raw data, which were converted into `FASTQ` using the `bamToFastq` command from `bedtools` [Quinlan and Hall, 2010]. The quality control analysis for `ChimpBrain` and `MouseMuscle` datasets was done using `fastqc` [Andrews et al., 2012] and `multiqc` [Ewels et al., 2016]. `salmon` indexes were built on gencode versions v26, vM25, and Pan_tro 3.0 to analyze human, mouse and chimpanzee datasets [Frankish et al., 2019, Cunningham et al., 2022] respectively. `Salmon` was used to quantify and generate Gibbs replicates for the RNA-Seq samples. For each sample, 100 Gibbs replicates were generated using a thinning factor of 100. The trees from `TreeTerminus` were created using the `Consensus` mode. All the pipelines used for analysis in this paper were constructed using `Snakemake` [Mölder et al., 2021].

2.5.2 Differential expression analysis

We compare the performance of carrying out the differential expression analysis using the different feature sets and tree-based methods on the analyzed datasets. The compared features comprise the selected transcripts (`Txps`), genes (`Genes`), `Terminus` groups (`Terminus`), and the candidate/driver nodes obtained by running the different tree-based methods on the unified tree $\mathcal{T}$ from `TreeTerminus`.

The base set for `Terminus` comprises the discrete transcript groups and any remaining transcripts that belong to the transcript set of interest but are not covered by `Terminus` groups. For the tree-based methods, we compare `trenDi`, `treeclimbR`, and `BOUTH`. Median-ratio scaling [Anders, 2010] was used to normalize the counts. When scaling the counts for the nodes in the tree, we first compute the size factor only using the leaf nodes and then divide the bias-corrected scaled counts of the inner nodes by the size factor. This procedure is described in detail in Supplementary Section S1.1. `Swish` is then used for generating p-values for the normalized counts. While `treeclimbR` and `trenDi` require p-values for all the nodes in the tree, `BOUTH` requires the p-values only for the leaf nodes. The p-values were generated separately for the leaf and inner nodes using `Swish`. For `treeclimbR`, the parameter α is set to the nominal threshold, which aims to control the FDR at leaf nodes to extract the candidates. We consider two feature sets for `treeclimbR` in our evaluation. `treeclimbR(N)` refers to the candidate set obtained after running `treeclimbR`, and `treeclimbR(L)` consists of the descendant transcripts for the nodes in `treeclimbR(N)`. We also consider `treeclimbR(L)` in our analysis because `treeclimbR` controls FDR at the leaf level, which, in this case, corresponds to the transcripts.

For `BOUTH`, we set the parameter FAR to the nominal FDR, which controls for false assignment rate, and extract the driver nodes from its output. For `trenDi`, the nominal FDR threshold is used to select the p-value threshold (on the leaves) to obtain the candidate nodes. For the other features (`Txps`, `Genes`, `Terminus`), `Swish` is applied directly to their base sets individually, and the selected set consists of features that have their adjusted p-values less than the given nominal FDR threshold.

2.5.3 Evaluation on the simulated datasets

Since the ground truth is known for the simulated datasets, the features obtained from the different methods are evaluated by computing the True Positive Rate (TPR) and False Discovery Rate (FDR) at several different nominal thresholds $(0.01, 0.05, 0.1)$. When computing these metrics for the features that consist of transcript groups or the inner nodes in the tree, we only consider the status of the differential expression for that node or the transcript group in our evaluation. We do not evaluate the differential status at the underlying descendant transcripts or other inner nodes corresponding to the group or inner node. To obtain the ground truth for differential expression at an inner node, we first create the aggregated counts matrix for the tree $\mathcal{T}$, `sim.tree.counts.mat` using `sim.counts.mat`. The `sim.counts.mat` consists of the true transcript counts corresponding to sample groups, on which we are evaluating the differential expression. This is the same matrix that was provided as an input to `Polyester` to simulate reads. The count of an inner node in `sim.tree.counts.mat` is computed by aggregating the counts of all the descendant transcripts in `sim.counts.mat`. The inner node is differentially expressed if the absolute log fold change (LFC) for that node in `sim.tree.counts.mat` is larger than a threshold. The threshold is set to the LFC obtained at the tree's

root node in the `sim.tree.counts.mat`.

3 Results

3.1 Running trenDi on null simulation

We first evaluated `trenDi` on the simulated dataset that had no differential signal between the conditions to measure the false positive rate. Further, we wanted to demonstrate why we recommend computing the p-values in the specific manner we have described. The null simulation is created using the framework as described in Section 2.4, with the only change being that the fold change between the two conditions is set to 1 for all the transcripts. Thus, by design, the simulation has no true signal for differential expression for the transcripts. We generate a total of 10 such null simulations.

3.1.1 Inner nodes have smaller p-values if the hypotheses testing is carried out on all the nodes of the tree simultaneously

For each null simulation, we ran `Swish` on all the nodes of the tree simultaneously, and then the computed p-values across all the null simulations were concatenated into one vector. As expected, the distribution of the p-values across all the nodes is uniform (Figure 2A). However, if we look at the distribution of the p-values at the leaves, there is a shift in the distribution towards the right for the leaf p-values and the left for the inner node p-values (Figure 2 B,C). Ideally, irrespective of the subset of features that are picked, the distribution should be uniform. This happens because the width of the distribution of the test-statistic which is used by `Swish` is wider for the inner nodes compared to the leaves (Figure 2D). We explain this in Section 2.3.2. Thus, if the p-values are generated together for all nodes, we might inflate false positives for the inner nodes and false negatives for the leaves. We thus compute the p-values for the leaves and inner nodes separately and, as expected, both of these show a uniform distribution (Figure S1).

3.1.2 The number of nodes reported by trenDi on the tree are comparable to running Swish on only the transcripts

We next looked at the number of candidate nodes that are reported by `trenDi` on the null simulation and compared it with the number of transcripts obtained by running `Swish` only on the transcripts (see Section 2.5.2). The total number of nodes and transcripts that are obtained at the different nominal FDR thresholds across the 10 simulations is shown in Table 1. The average number of transcripts across the 10 simulations is around 135000 while the total number of nodes in the tree is around 250000. We observe the total number of reported nodes is comparable across both of the methods. This experiment demonstrates that `trenDi` should not exaggerate the false positive rate.

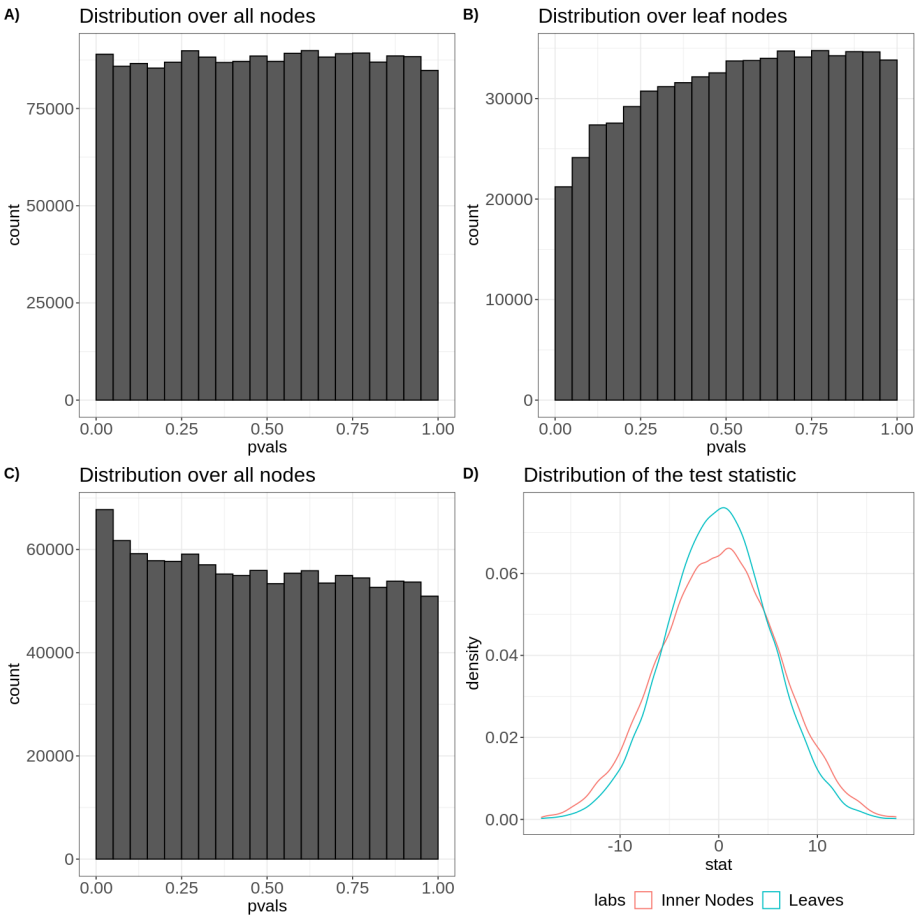

Figure 2: Distribution of p-values and test-statistic for Swish for the null brain simulation. A) Distribution of the p-values on the entire node set over the tree when all hypotheses are tested together using Swish. B) Distribution of the p-values over the leaf nodes. C) Distribution of the p-values over the inner nodes. D) Distribution of the test-statistic on the inner and leaf nodes, which is used by Swish to generate the p-values.

3.2 Comparing the methods on simulated datasets with signal

We next benchmark and analyze the simulated datasets with differential signal i.e. `BrSimNorm` and `BrSimLow` datasets (Section 2.4).

Table 1: Total number of nodes selected by each Swish and `trenDi` on the 10 null simulations

Simulation	Swish (0.01,0.05,0.1)			trenDi (0.01,0.05,0.1)		
Simulation 1	6	6	6	6	6	6
Simulation 2	5	5	5	5	5	5
Simulation 3	6	6	6	6	6	6
Simulation 4	5	5	5	5	5	5
Simulation 5	7	7	7	7	7	7
Simulation 6	6	6	6	7	7	7
Simulation 7	5	5	5	6	6	6
Simulation 8	6	6	6	7	7	7
Simulation 9	5	5	5	5	5	5
Simulation 10	6	6	6	6	6	6

3.2.1 Differential analysis on the different methods

We compare the features obtained after differential testing using different methods. The features consist of genes (`Genes`), transcripts (`Txps`, `treeclimbR(L)`) and transcript-groups (`Terminus`, `trenDi`, `treeclimbR(N)`, `BOUTH`) (see Section 2.5.2). `Txps` refers to the transcripts obtained by running "vanilla" Swish only on the transcripts. When `treeclimbR` selects an inner node as a candidate node, this implies that all of its descendant transcripts are also differentially expressed. Thus, in our analysis, we compare two feature sets corresponding to `treeclimbR`: `treeclimbR(N)`, which consists of only the candidate nodes, and `treeclimbR(L)`, which consists of the underlying transcripts belonging to the nodes. This enables us to compare the performance of *tcr* at both the node and transcript levels. On all the simulated and experimental datasets, `BOUTH` outputs the root node as the driver, and thus we do not show its results in the comparison. We first look at the number of true positive transcripts that are covered by the different methods on the simulated datasets (Figure S2, S3). We observe that `trenDi` yields the largest number of true positives, followed by `treeclimbR`.

We next compute TPR and FDR at different nominal FDR thresholds on the feature sets obtained from the different methods (Figure 3, Section 2.5.3). For the `BrSimNorm` dataset, `treeclimbR(L)` and `trenDi` have the largest TPR, while for the `BrSimLow` dataset, `Genes` has the highest TPR, followed by `treeclimbR(L)` and `trenDi`. `treeclimbR(L)` has an inflated FDR on both datasets (since all transcripts belonging to a candidate node might not be true positives), whereas `treeclimbR(N)`, while controlling for FDR, does not seem to lead to that much overall improvement in sensitivity compared to `Txps`. `trenDi` seems to have a good sensitivity while controlling for the FDR at the same time, with FDR only going slightly beyond the nominal FDR threshold of 0.05 for the `BrSimNorm` dataset. It is important to note that since the underlying features across the

methods are different, these metrics may not be directly comparable. However, we do obtain some information about their respective performances. We next investigate the increased sensitivity that is shown by `trenDi`, by doing a pairwise comparison with the other methods on the `BrSimNorm` dataset.

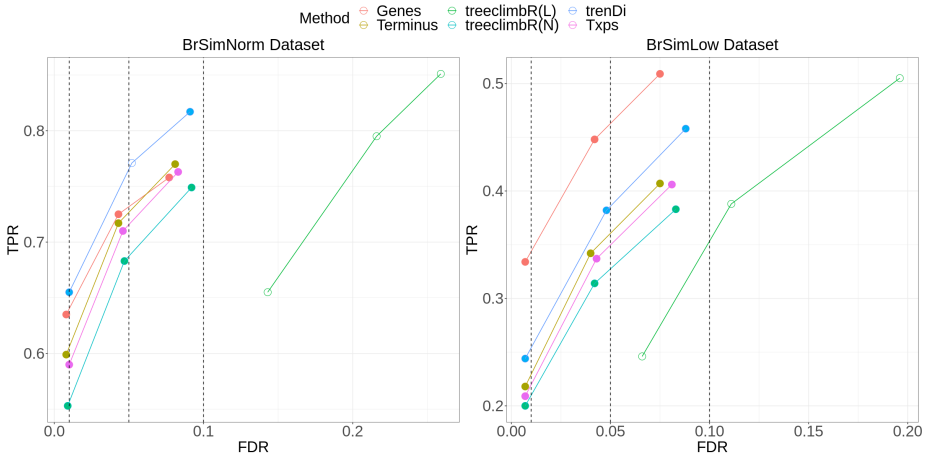

Figure 3: True Positive Rates and Empirical False Discovery Rates at the different nominal FDR thresholds across the different methods and entities for the `BrSimNorm` and `BrSimLow` Dataset.

3.2.2 Comparing Terminus with trenDi

When comparing `trenDi` with `Terminus` at a nominal FDR of 0.01, we find that `Terminus` candidate nodes correspond to only 31 additional unique true positive transcripts, which map to 26 `Terminus` groups. For `trenDi`, transcripts descending from its candidate nodes yield a total of 681 unique true positive transcripts and correspond to 473 nodes. While the general expectation might be that most of these additional candidate nodes in `trenDi` would be as a result of more aggregation, we find 27 nodes for which the signal exists at a lower level in `trenDi` that get lost in `Terminus` due to overaggregation. We provide some examples of such cases in Figure 4, S4 and S5. Similarly, there are 103 such groups (inner nodes in the tree) in `trenDi` that contain at least one underlying descendant transcript, which was differentially expressed. The differential signal is lost for these transcripts in `Terminus`, either because the transcripts were not grouped at all or not aggregated enough to produce the signal. In Figure 5, S6 and S7, we show some examples of such cases.

3.2.3 Comparing treeclimbR with trenDi

We next perform a pairwise comparison between `treeclimbR` and `trenDi` at the 0.01 nominal FDR on the `BrSimNorm` dataset. We first inspect the relative height

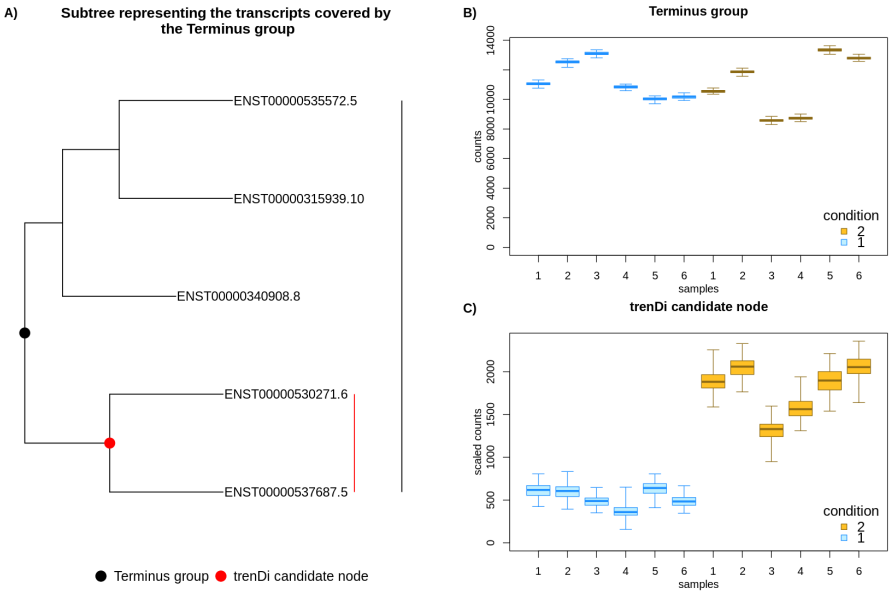

Figure 4: Example of a `trenDi` node that is overaggregated in `Terminus` for the `BrSimNorm` dataset. A) Subtree representing the transcripts covered by the `Terminus` group. B) Inferential replicates for the `Terminus` group. C) Inferential replicates for the candidate node output by `trenDi`.

of the nodes in Table S1. We observe more higher-level nodes for `treeclimbR` compared to `trenDi`, with the maximum height of the candidate node being 19 and 14 in `treeclimbR` and `trenDi`, respectively. It is very hard to compare these two methods when the candidate nodes output by them are located at different heights and when the differential signal can exist at multiple levels within the same branch in the tree. We thus try to identify the branches that are unique to a method using the approach below. For example, imagine that we want to find the nodes that are unique to `trenDi`. To do so, we first extract the candidate nodes that are common to both of these methods (`treeclimbR`, `trenDi`). From the remaining candidate nodes (excluding common nodes) that are output by `trenDi`, we remove all nodes i for which there exists a node j that is a candidate node output of `treeclimbR` and is either an ancestor or descendant of node i in the tree. The candidate nodes that are left represent the unique branches of `trenDi`. We repeat a similar process for `treeclimbR`, to find the unique branches of `treeclimbR`. We find 199 such nodes unique to `trenDi`, while for `treeclimbR` we find only 2 such nodes. Thus `trenDi` outputs more unique branches.

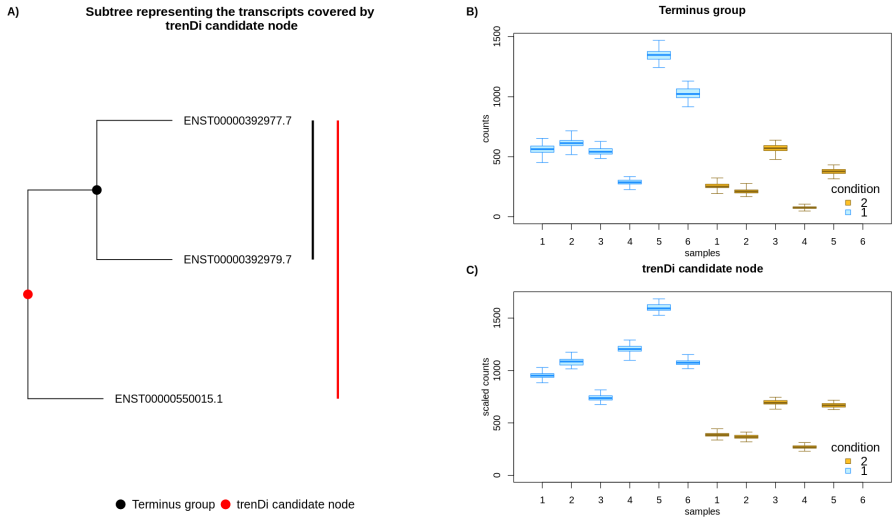

Figure 5: Example of a `trenDi` node that is not aggregated enough in `Terminus` for the `BrSimNorm` dataset. A) Subtree representing the transcripts covered by the `trenDi` group. B) Inferential replicates for the `Terminus` group. C) Inferential replicates for the candidate node output by `trenDi`.

3.2.4 Error rate for the unique nodes

Since each method can output unique candidate nodes, we next measured the error rate only on the unique branches/nodes, when doing a pairwise comparison between methods. The unique nodes are extracted using the approach described above for the different nominal FDR thresholds. Specifically, we extract the nodes for pairwise comparison between `trenDi` versus `treeclimbR`, and `treeclimbR` versus `Txps`. We provide the observed FDR and the total number of unique nodes for these comparisons in Table 2, 3. The multiple hypothesis correction methods aim to control the average false discovery proportion of the overall set of rejected hypotheses. However, the error guarantee might not necessarily hold on a specific subset, which in this case are the unique nodes corresponding to each method. We observe an inflated observed FDR for the unique features corresponding to `Txps` and `trenDi`, although it decreases for lower nominal FDR thresholds. The observed FDR is also inflated for `treeclimbR` on pairwise comparison with `Txps`, and it contains significantly fewer unique nodes. We next try to see if we can reduce the FDR by filtering the unique set based on the threshold of absolute log fold change between the conditions of interest. We plot this in Figure 6, S8 and S9. We see that the observed FDR starts decreasing as the effect size increases and then again starts to increase. The FDR again starts increasing because, after a certain threshold of effect size, only a few candidate nodes are left after filtering. As a result, the false positives get penalized more heavily. We thus recommend using a conservative nominal FDR threshold with an appropriate effect size when the novel

features unique to any method are analyzed.

Table 2: Number of unique nodes that are observed for `trenDi`, `treeclimbR` nodes when making a pairwise comparison for the `BrSimNorm` dataset at different nominal FDR. The number inside the bracket represents the total number of nodes.

Nominal FDR	0.01	0.05	0.10
`trenDi`	0.085 (199)	0.412 (308)	0.606 (381)
`treeclimbR(N)`	0 (2)	0 (4)	0.25(4)

Table 3: Number of unique nodes that are observed for `Txps`, `treeclimbR` nodes when making a pairwise comparison for the `BrSimNorm` dataset at different nominal FDR. The number inside the bracket represents the total number of nodes.

Nominal FDR	0.01	0.05	0.10
`Txps`	0.103 (145)	0.416 (255)	0.590 (339)
`treeclimbR(N)`	0 (15)	0.24 (25)	0.481 (27)

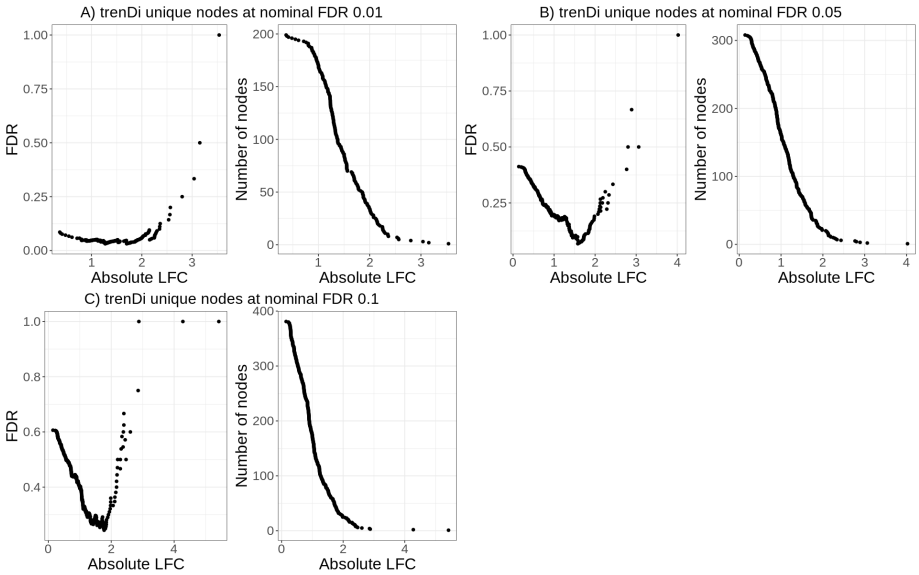

Figure 6: Examination of error metrics for the unique nodes obtained for `trenDi` when doing `trenDi` vs `treeclimbR` analysis at the different nominal FDR thresholds for the `BrSimNorm` dataset. We vary the magnitude of log fold change (LFC) and plot the empirical FDR and the total number of nodes that are left after filtering the unique nodes based on LFC.

3.3 Analyzing the MouseMuscle dataset

We next analyzed the `MouseMuscle` dataset which consists of 6 samples each from EDL and MAST muscle tissues (Section 2.4). The first two dimensions of the PCA using the top 1000 variable genes for the dataset are shown in Figure S10. The total number of differentially expressed genes (DEGs), differentially expressed transcripts (DETs) at the different nominal FDRs and the total number of transcripts that are covered by the candidate nodes/groups output by the different methods at the 0.01 nominal FDR are shown in Table S3, S4, Figure S11. We observe that `trenDi`-reported nodes cover more transcripts compared to `treeclimbR`-reported nodes. Depending on the resolution at which a feature is observed, we find many genes that are called significant when differential testing is carried out on the genes, but the underlying transcripts are not called as significant when the testing is performed on the transcripts (Table S5). Similarly, when differential testing is carried out on the transcripts, some of the significant transcripts map to non-significant genes (Table S6). Some of the candidate nodes output by `trenDi` neither map to a significant gene nor contain a single underlying transcript that is significant (Table S7). We also find many candidate nodes in `trenDi` that map to significant genes, but the genes do not contain a single transcript that is called significant during transcript-level differential analysis (Table 4). We provide some examples for these genes in (Figure 7, 8, S12, S13, S14, S15 and S16) obtained for 0.01 nominal FDR. For example, we can see that for the gene `PRRG1`, the transcripts highlighted in red (which underlie the selected candidate node) constitute a very similar set of exons, but those exons vary in length. This is reflected in the tree structure in Figure 7B, where the transcripts `ENSMUST000000114024.8` and `ENSMUST000000177904.7` are aggregated first, followed by transcript `ENSMUST000000114025.7`. Thus, in Figure 7C we see that the transcript `ENSMUST000000177904.7` has a high inferential relative variance (`infRV`). This also affects the quality of the signal, which gets stronger at the candidate node level constituting the aggregated transcripts (Figure 7D).

Table 4: Number of `trenDi` nodes that map to a significant gene but the gene doesn't consist of any significant transcript for the `MouseMuscle` dataset.

Nominal FDR	0.01	0.05	0.10
Number of `trenDi` nodes	309	428	465

3.4 Analyzing the ChimpBrain dataset

We also assessed the performance of `trenDi` on the `ChimpBrain` dataset, which contains 5 samples each from both `cerebellum` and `medial dorsal nucleus of thalamus` (Section 2.4). The number of genes and transcripts that are differentially expressed between the conditions at the different FDR thresholds is provided in Table S8 and S9 with the total number of

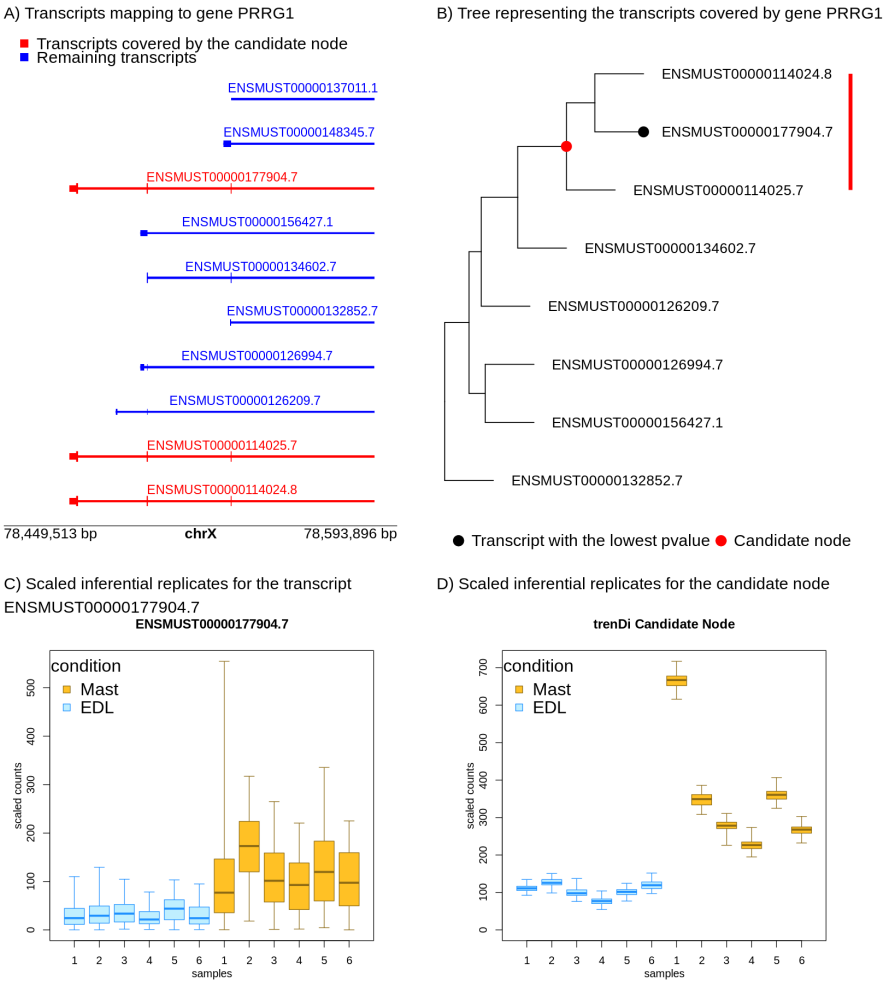

Figure 7: Examining the transcript profile for the gene PRRG1 in the `MouseMuscle` dataset. A) Transcripts in a pileup style. B) Tree representing the transcripts covered by the gene PRRG1, with the red node representing the transcripts covered by the `trenDi` candidate node. C) Inferential replicates for the transcript ENSMUST00000177904.7, which had the lowest p-value among all the transcripts in the tree. D) Inferential replicates for the `trenDi` candidate node.

features across the different methods shown in Figure S17. We again observe that the `trenDi`-reported nodes cover the largest number of transcripts. As observed for the `MouseMuscle` dataset, we have many examples for both cases where significantly called transcripts do not necessarily map to significantly called genes and vice versa for significant transcripts (Table S10, S11). Similarly, we find some of the nodes output by `trenDi` that map to non-significant

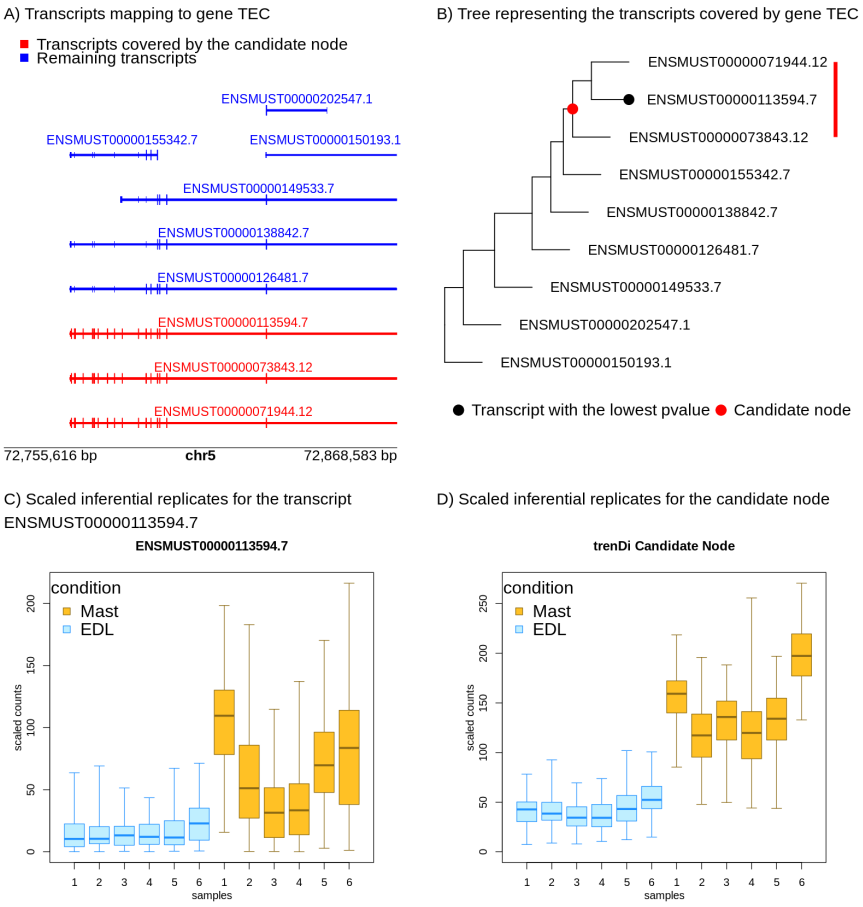

Figure 8: Examining the transcript profile for the gene TEC in the `MouseMuscle` dataset. A) Transcripts in a pileup style. B) Tree representing the transcripts covered by the gene TEC, with the red node representing the transcripts covered by the `trenDi` candidate node. C) Inferential replicates for the transcript ENSMUST00000113594.7, which had the lowest p-value among all the transcripts in the tree. D) Inferential replicates for the `trenDi` candidate node.

genes and also don't contain a single transcript that is called significant (Table S12).

On the other hand, there also exist many other `trenDi` nodes that map to those significant genes whose underlying transcripts were not called significant (Table 5). We report some of these cases in Figure 9, 10, S18, S19, S20, S21, S22 and S23 for the 0.01 nominal FDR. For example, for gene EYA4 in Figure 9,

we observe that the transcripts ENSPTRT00000034383 and ENSPTRT00000104666 differ by only 1 exon, and the candidate group that aggregates both these transcripts has lower uncertainty and shows an improved signal, compared to the transcript ENSPTRT00000104666 which has high uncertainty.

Table 5: Number of trenDi nodes that map to a significant gene but the gene doesn't consist of any significant transcript for the ChimpBrain dataset.

Nominal FDR	0.01	0.05	0.10
Number of trenDi nodes	407	194	213

3.5 Comparison of running time

We compared the runtime of treeclimbR and trenDi on the BrSimNorm and the MouseMuscle datasets. For treeclimbR, two functions, namely getCand and evalCand, have to be called to obtain candidates. We executed the getCand function once and then ran the evalCand function at the different nominal FDR thresholds. trenDi is parallelizable and supports multiple cores. In our comparison, as shown in Table 6, we presented the run time for getCand computed once and the runtime for evalCand on each nominal FDR threshold. Additionally, we showcased the time taken by trenDi when running on a single core and on four cores. Notably, the getCand function takes an order of magnitude more time compared to all the other functions. Furthermore, for trenDi, we observed an increase in runtime as we raised the FDR threshold. The increase may be attributed to the exploration of more branches, as the threshold becomes less stringent. The parallelized version exhibits a similar pattern. We also observe the benefits of parallelization, achieving a more than threefold increase in speed using four cores compared to using a single core.

Table 6: Time taken in (hh:mm:ss) by treeclimbR and trenDi on the BrSimNorm and MouseMuscle dataset.

Method		BrSimNorm	MouseMuscle
treeclimbR	getCand	37:06:5.11	15:05:40
	evalCand($\alpha = 0.01$)	0:00:50.63	0:00:28.93
	evalCand($\alpha = 0.05$)	0:00:50.25	0:00:28.52
	evalCand($\alpha = 0.1$)	0:00:50.20	0:00:28.91
trenDi (cores = 1)	$\alpha = 0.01$	0:07:15.66	0:06:49.19
	$\alpha = 0.05$	0:08:49.20	0:12:13.8
	$\alpha = 0.1$	0:09:59.81	0:15:23.29
trenDi (cores = 4)	$\alpha = 0.01$	0:01:58.23	0:02:21.52
	$\alpha = 0.05$	0:02:24.65	0:03:54.66
	$\alpha = 0.1$	0:02:46.01	0:04:43.61

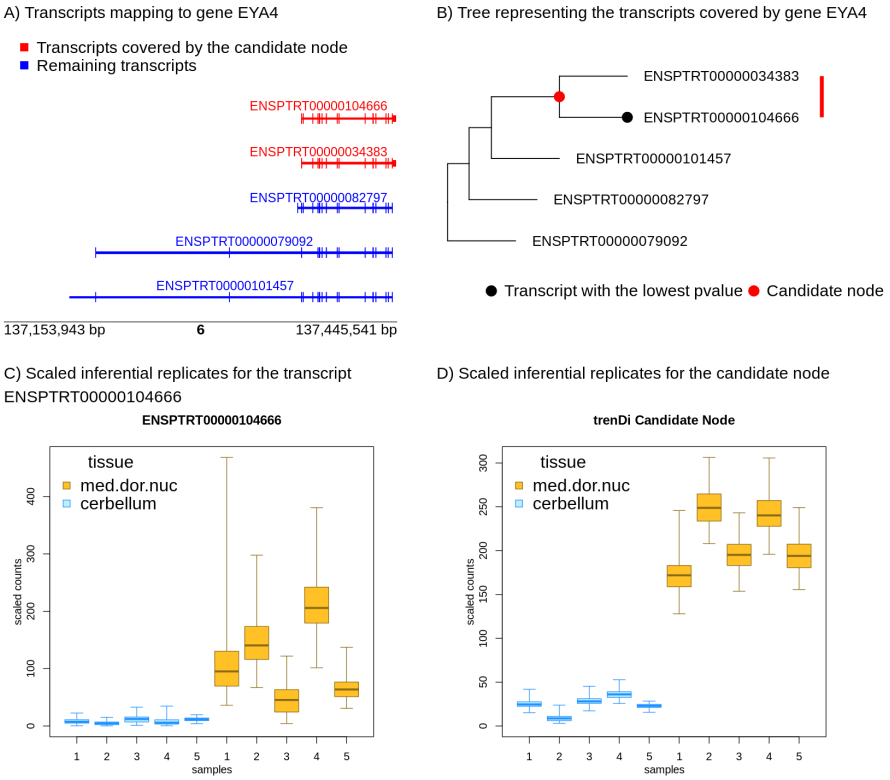

Figure 9: Examining the transcript profile for the gene EYA4 in the `ChimpBrain` dataset. A) Transcripts in a pileup style. B) Tree representing the transcripts covered by the gene EYA4, with the red node representing the transcripts covered by the `trenDi` candidate node. C) Inferential replicates for the transcript ENSMUST00000104666, which had the lowest p-value among all the transcripts in the tree. D) Inferential replicates for the `trenDi` candidate node.

4 Discussion

The high level of uncertainty that is observed in the abundances of some transcripts during RNA-Seq quantification poses a challenge for downstream analyses, particularly for tasks such as differential testing. Various differential testing methods have been proposed that incorporate uncertainty into testing, and these

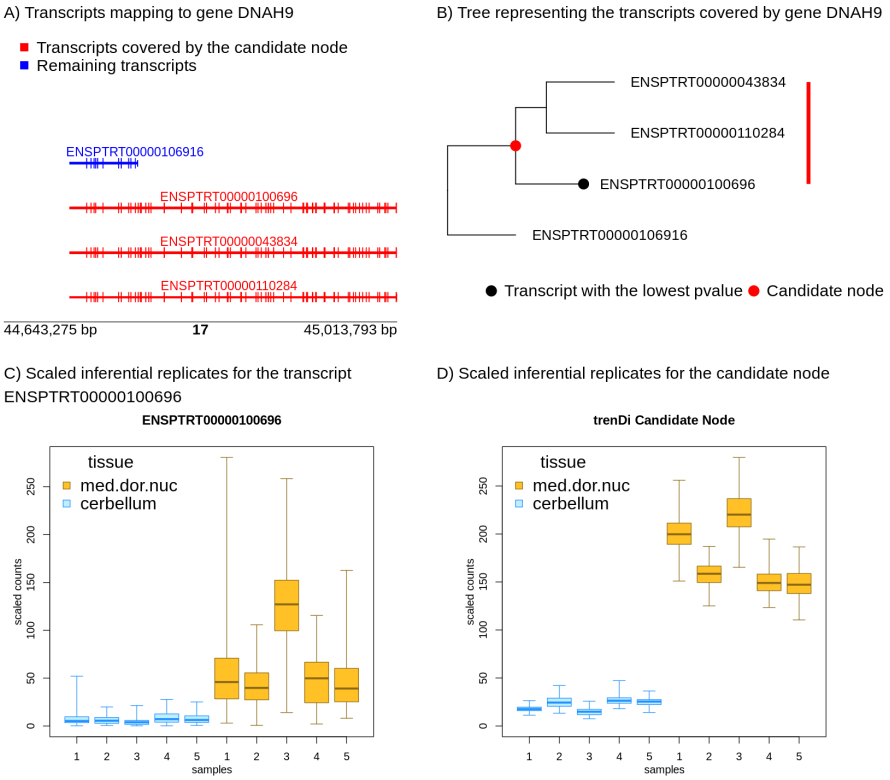

Figure 10: Examining the transcript profile for the gene DNAH9 in the `ChimpBrain` dataset. A) Transcripts in a pileup style. B) Tree representing the transcripts covered by the gene DNAH9, with the red node representing the transcripts covered by the `trenDi` candidate node. C) Inferential replicates for the transcript ENSMUST00000100696, which had the lowest p-value among all the transcripts in the tree. D) Inferential replicates for the `trenDi` candidate node.

methods have been applied either to transcripts or genes. This approach can occasionally result in overlooking the signal for the features that are difficult to quantify. Our method `trenDi`, makes use of the tree structure computed by `TreeTerminus` [Singh et al., 2023] to produce the candidate nodes that are significant between conditions. `trenDi` provides a data-driven procedure for finding signal at a resolution between the gene and transcript level for RNA-

Seq differential analysis, where a candidate inner node captures the signal for a set of underlying transcripts that have high uncertainty. While many tree-based differential testing procedures have been proposed, our method `trenDi` is among the few tailored toward RNA-Seq data, which requires specific considerations. Our method requires p-values for all the nodes of the tree and thus is flexible in the choice of the differential testing method used, though we have used `Swish` [Zhu et al., 2019] in this study.

We assessed and evaluated `trenDi` on both the simulated and experimental datasets. In the variations of the simulated datasets without true signal, we observed that the number of false positives generated by `trenDi` was comparable to those generated by doing testing only on the transcripts, despite considerably more hypotheses being tested in the former case. This suggests that `trenDi` controls the FDR well under the null hypothesis. Similarly, on both variations of simulated data that included true signal, we observe an increased sensitivity for `trenDi` while controlling the FDR at or very near the nominal level. The enhanced sensitivity is not only evident quantitatively, through the metrics we report, but is also reflected in the discovery of more transcript branches on the tree when conducting pairwise comparisons with other methods. Interestingly, `treeclimbR` [Huang et al., 2021], which has been designed to control FDR at the leaf level, shows a very high sensitivity but also outputs a large number of false positives. Similarly, on the experimental datasets, we observe many significant genes that do not contain a single underlying transcript that is deemed significant when doing transcript-level differential analysis, but which contribute to nodes reported by `trenDi`. This is a reflection on the ability of `trenDi` to find signal that would have been lost when doing transcript-level analysis. Shared exons are commonly observed among these transcripts that belong to the candidate nodes, increasing our confidence in the utility of the approach. We also observed a clear reduction in uncertainty for the aggregated transcript groups compared to the individual transcripts.

There exist several areas for improvement of our method, and also opportunities for developing new and related methods. The difference in the p-value distribution observed for the inner and leaf nodes in the null simulation when the p-values for all nodes are calculated together suggests the need for a differential testing method that takes the hierarchy of features into account. It seems that, at least in the different simulations that we have created, we are controlling the FDR. However, since we are using the same dataset to create the tree and do differential testing, this technically leads to double-dipping [Kriegeskorte et al., 2009], which has the possibility to lead to an exaggeration of false positives. This could be one of the reasons why we observe an inflated FDR for `treeclimbR`.

To mitigate double-dipping, a Poisson/negative binomial sampling of the original data into independent partitions of the same dimensions was introduced in an approach called `count splitting`[Neufeld et al., 2022, 2023]. The first partition can be used for latent-variable estimation and the second partition for inference. Incorporating `count splitting` into `trenDi` will be a promising future direction. The challenge, however, will be how to do this without negatively

affecting the power, since `trenDi` already appears empirically well-calibrated.

There might also be another source of false positives. When doing a multiple-hypothesis correction on a set of hypotheses, the error rate is controlled over the entire set and not on a subset. Using our filtering strategy, we are outputting a subset of hypotheses that have p-value below a certain threshold. `Focused BH` [Katsevich et al., 2023] provides a way of selecting a p-value threshold by running the filter for the different p-value thresholds and finding the largest threshold at which the computed conservative estimate of the false discovery proportion for the set is less than the desired error rate. The limitation of this approach is that, apart from being conservative, it also puts constraints on the properties of the filters. Further, `Focused BH` would be computationally slow in its native form, specifically if filters have to be computed over hundreds or thousands of p-value thresholds and each filter takes non-trivial time to run. Adapting this approach for `trenDi` can also be explored in the future.

Finally, it would be intriguing to explore the development of a `TreeTerminus` and `trenDi` like approach in the context of single-cell RNA-seq analysis, where there are quite distinct, but related, opportunities for data-driven aggregation to increase the power and specificity of differential analysis.

5 Availability of data and materials

`trenDi` is available as an `R` package and can be obtained from https://github.com/NPSDC/trenDi. The scripts to reproduce the results in this manuscript are provided in the GitHub repository https://github.com/NPSDC/trenDi-paper-analysis. The scripts to obtain and generate the datasets that have been analyzed in this manuscript are provided in https://github.com/NPSDC/trenDi-paper-data.

6 Contributions

N.P.S, M.I.L and R.P conceived the trenDi method. N.P.S wrote the software. N.P.S, E.Y.W, M.I.L and R.P conceived the experiments, and N.P.S carried out the experiments. N.P.S, M.I.L and R.P interpreted the results. N.P.S, J.F carried out the simulations. N.P.S, M.I.L and R.P wrote and revised the manuscript. M.I.L and R.P secured funding.

7 Funding

This work is supported by the National Institute of Health under grant award numbers R01HG009937 to RP and ML, and the National Science Foundation under grant award numbers CCF-1750472, and CNS-1763680 to RP. The funders had no role in the design of the method, data analysis, decision to publish, or preparation of the manuscript.

8 Declarations of interests

RP is a cofounder of Ocean Genomics, Inc.

References

S. Anders. Huber w ((2010)) differential expression analysis for sequence count data. *Genome Biol*, 11(10):R106, 2010.

S. Andrews, F. Krueger, A. Segonds-Pichon, L. Biggins, C. Krueger, and S. Wingett. FastQC. Babraham Institute, Jan. 2012.

P. L. Baldoni, Y. Chen, S. Hediyeh-zadeh, Y. Liao, X. Dong, M. E. Ritchie, W. Shi, and G. K. Smyth. Dividing out quantification uncertainty allows efficient assessment of differential transcript expression. *bioRxiv*, pages 2023–04, 2023.

Y. Benjamini and Y. Hochberg. Controlling the false discovery rate: a practical and powerful approach to multiple testing. *Journal of the Royal statistical society: series B (Methodological)*, 57(1):289–300, 1995.

A. Bichat, C. Ambroise, and M. Mariadassou. Hierarchical correction of p-values via an ultrametric tree running ornstein-uhlenbeck process. *Computational Statistics*, pages 1–19, 2022.

J. Bien, X. Yan, L. Simpson, and C. L. Müller. Tree-aggregated predictive modeling of microbiome data. *Scientific Reports*, 11(1):14505, 2021.

M. Bogomolov, C. B. Peterson, Y. Benjamini, and C. Sabatti. Hypotheses on a tree: new error rates and testing strategies. *Biometrika*, 108(3):575–590, 2021.

N. L. Bray, H. Pimentel, P. Melsted, and L. Pachter. Near-optimal probabilistic rna-seq quantification. *Nature biotechnology*, 34(5):525–527, 2016.

F. Cunningham, J. E. Allen, J. Allen, J. Alvarez-Jarreta, M. R. Amode, I. M. Armean, O. Austine-Orimoloye, A. G. Azov, I. Barnes, R. Bennett, et al. Ensembl 2022. *Nucleic acids research*, 50(D1):D988–D995, 2022.

P. Ewels, M. Magnusson, S. Lundin, and M. Käller. Multiqc: summarize analysis results for multiple tools and samples in a single report. *Bioinformatics*, 32 (19):3047–3048, 2016.

A. Frankish, M. Diekhans, A.-M. Ferreira, R. Johnson, I. Jungreis, J. Loveland, J. M. Mudge, C. Sisu, J. Wright, J. Armstrong, et al. Gencode reference annotation for the human and mouse genomes. *Nucleic acids research*, 47 (D1):D766–D773, 2019.

A. C. Frazee, A. E. Jaffe, B. Langmead, and J. T. Leek. Polyester: simulating rna-seq datasets with differential transcript expression. *Bioinformatics*, 31 (17):2778–2784, 2015.

J. J. Goeman and A. Solari. The sequential rejection principle of familywise error control. *The Annals of Statistics*, pages 3782–3810, 2010.

R. Huang, C. Soneson, P.-L. Germain, T. S. Schmidt, C. V. Mering, and M. D. Robinson. treeclimbr pinpoints the data-dependent resolution of hierarchical hypotheses. *Genome biology*, 22(1):1–21, 2021.

E. Katsevich, C. Sabatti, and M. Bogomolov. Filtering the rejection set while preserving false discovery rate control. *Journal of the American Statistical Association*, 118(541):165–176, 2023.

N. Kriegeskorte, W. K. Simmons, P. S. Bellgowan, and C. I. Baker. Circular analysis in systems neuroscience: the dangers of double dipping. *Nature neuroscience*, 12(5):535–540, 2009.

T. Lappalainen, M. Sammeth, M. R. Friedländer, P. Ac't Hoen, J. Monlong, M. A. Rivas, M. Gonzalez-Porta, N. Kurbatova, T. Griebel, P. G. Ferreira, et al. Transcriptome and genome sequencing uncovers functional variation in humans. *Nature*, 501(7468):506–511, 2013.

J. T. Leek. Svaseq: removing batch effects and other unwanted noise from sequencing data. *Nucleic acids research*, 42(21):e161–e161, 2014.

B. Li and C. N. Dewey. Rsem: accurate transcript quantification from rna-seq data with or without a reference genome. *BMC bioinformatics*, 12(1):1–16, 2011.

Y. Li, Y.-J. Hu, and G. A. Satten. A bottom-up approach to testing hypotheses that have a branching tree dependence structure, with error rate control. *Journal of the American Statistical Association*, 117(538):664–677, 2022.

M. I. Love, C. Soneson, and R. Patro. Swimming downstream: statistical analysis of differential transcript usage following salmon quantification. *F1000Research*, 7, 2018.

G. Lynch and W. Guo. On procedures controlling the fdr for testing hierarchically ordered hypotheses. *arXiv preprint arXiv:1612.04467*, 2016.

I. Mandric, Y. Tiagueu, T. Shcheglova, S. A. Seesi, A. Zelikovsky, and I. Mandoiu. Fast bootstrapping-based estimation of confidence intervals of expression levels and differential expression from rna-seq data. *Bioinformatics*, 33 (20):3302–3304, Oct. 2017. URL https://doi.org/10.1093/bioinformatics/btx365.

N. Meinshausen. Hierarchical testing of variable importance. *Biometrika*, 95 (2):265–278, 2008.

J. C. Miecznikowski and J. Wang. Error control in tree structured hypothesis testing. *Wiley Interdisciplinary Reviews: Computational Statistics*, 15(4): e1603, 2023.

F. Mölder, K. P. Jablonski, B. Letcher, M. B. Hall, C. H. Tomkins-Tinch, V. Sochat, J. Forster, S. Lee, S. O. Twardziok, A. Kanitz, et al. Sustainable data analysis with snakemake. *F1000Research*, 10, 2021.

A. Neufeld, L. L. Gao, J. Popp, A. Battle, and D. Witten. Inference after latent variable estimation for single-cell rna sequencing data. *arXiv preprint arXiv:2207.00554*, 2022.

A. Neufeld, J. Popp, L. L. Gao, A. Battle, and D. Witten. Negative binomial count splitting for single-cell rna sequencing data. *arXiv preprint arXiv:2307.12985*, 2023.

J. Ostner, S. Carcy, and C. L. Müller. tasccoda: Bayesian tree-aggregated analysis of compositional amplicon and single-cell data. *Frontiers in genetics*, 12:766405, 2021.

R. Patro, G. Duggal, M. I. Love, R. A. Irizarry, and C. Kingsford. Salmon provides fast and bias-aware quantification of transcript expression. *Nature methods*, 14(4):417–419, 2017.

H. Pimentel, N. L. Bray, S. Puente, P. Melsted, and L. Pachter. Differential analysis of rna-seq incorporating quantification uncertainty. *Nature methods*, 14(7):687–690, 2017.

A. R. Quinlan and I. M. Hall. Bedtools: a flexible suite of utilities for comparing genomic features. *Bioinformatics*, 26(6):841–842, 2010.

M. E. Ritchie, B. Phipson, D. Wu, Y. Hu, C. W. Law, W. Shi, and G. K. Smyth. limma powers differential expression analyses for rna-sequencing and microarray studies. *Nucleic acids research*, 43(7):e47–e47, 2015.

H. Sarkar, A. Srivastava, H. C. Bravo, M. I. Love, and R. Patro. Terminus enables the discovery of data-driven, robust transcript groups from rna-seq data. *Bioinformatics*, 36(Supplement_1):i102–i110, 2020.

S. A. Seesi, Y. Tiagueu, A. Zelikovsky, and I. Mandoiu. Bootstrap-based differential gene expression analysis for rna-seq data with and without replicates. *BMC Genomics*, 15(Suppl 8):S2, 2014. URL http://www.biomedcentral.com/1471-2164/15/S8/S2/.

N. P. Singh. beaveR - R package for processing the output of TreeTerminus. https://github.com/NPSDC/beaveR, 2023.

N. P. Singh, M. I. Love, and R. Patro. Treeterminus—creating transcript trees using inferential replicate counts. *iScience*, 26(6), 2023.

C. Soneson, M. I. Love, and M. D. Robinson. Differential analyses for rna-seq: transcript-level estimates improve gene-level inferences. *F1000Research*, 4, 2015.

A. M. Sousa, Y. Zhu, M. A. Raghanti, R. R. Kitchen, M. Onorati, A. T. Tebbenkamp, B. Stutz, K. A. Meyer, M. Li, Y. I. Kawasawa, et al. Molecular and cellular reorganization of neural circuits in the human lineage. *Science*, 358(6366):1027–1032, 2017.

E. E. Terry, X. Zhang, C. Hoffmann, L. D. Hughes, S. A. Lewis, J. Li, M. J. Wallace, L. A. Riley, C. M. Douglas, M. A. Gutierrez-Monreal, et al. Transcriptional profiling reveals extraordinary diversity among skeletal muscle tissues. *Elife*, 7:e34613, 2018.

E. Turro, S.-Y. Su, Â. Gonçalves, L. J. Coin, S. Richardson, and A. Lewin. Haplotype and isoform specific expression estimation using multi-mapping rna-seq reads. *Genome biology*, 12(2):1–15, 2011.

E. Turro, W. J. Astle, and S. Tavare. Flexible analysis of rna-seq data using mixed effects models. *Bioinformatics*, 30(2):180–188, 2014.

J. Xiao, H. Cao, and J. Chen. False discovery rate control incorporating phylogenetic tree increases detection power in microbiome-wide multiple testing. *Bioinformatics*, 33(18):2873–2881, 2017.

D. Yekutieli. Hierarchical false discovery rate–controlling methodology. *Journal of the American Statistical Association*, 103(481):309–316, 2008.

A. Zhu, A. Srivastava, J. G. Ibrahim, R. Patro, and M. I. Love. Nonparametric expression analysis using inferential replicate counts. *Nucleic Acids Research*, 47(18):e105–e105, 2019.