ORIGINAL PAPER

scMaSigPro: Differential Expression Analysis along Single-Cell Trajectories

Priyansh Srivastava1,2, Marta Benegas Coll1, Stefan Götz1, María José Nueda2,3 and Ana Conesa4∗

1Biobam Bioinformatics S.L., Valencia, Spain, 2Department of Computer Science, University of Valencia, Valencia, Spain, 3Mathematics Department, University of Alicante, Alicante, Spain and 4Institute for Integrative Systems Biology (I³SysBio), Consejo Superior de Investigaciones Científicas (CSIC), Valencia, Spain

*Corresponding author: ana.conesa@csic.es

FOR PUBLISHER ONLY Received on 02 March 2024; revised on Date April 2024; accepted on 28 June 2024

Abstract

Motivation: Understanding the dynamics of gene expression across different cellular states is crucial for discerning the mechanisms underneath cellular differentiation. Genes that exhibit variation in mean expression as a function of Pseudotime and between branching trajectories are expected to govern cell fate decisions. We introduce scMaSigPro, a method for the identification of differential gene expression patterns along Pseudotime and branching paths simultaneously.

Results: We assessed the performance of scMaSigPro using synthetic and public datasets. Our evaluation shows that scMaSigPro outperforms existing methods in controlling the False Positive Rate and is computationally efficient.

Availability and Implementation: scMaSigPro is available as a free R package (version 4.0 or higher) under the GPL(≥ 2) license on GitHub at ‘github.com/BioBam/scMaSigPro’ and archived with version 0.03 on Zenodo at ‘zenodo.org/records/12568922’

Key words: Pseudotime, Time-Series Differential Expression, Differential Genes, Trajectory Branches, Software

Introduction

Single-cell RNA-seq (scRNA-Seq) datasets, particularly those featuring cell differentiation and transition experiments, greatly benefit from Trajectory Inference (TI) analysis. TI method reveals the dynamics of cellular processes by assigning Pseudotime to individual cells (Trapnell et al. 2014). Pseudotime allocation facilitates the ordering of cells along the continuum of cellular processes and delineates the course of cell lineages as specific paths or branches within a scRNA-Seq dataset (Deconinck et al. 2021). By associating Pseudotime to gene expression values, transcriptional profiles across cell states can be delineated. Genes that show significant variation of their mean expression in Pseudotime along a specific branch of a trajectory are expected to play crucial roles in cell fate decisions. Identifying these genes is key to understanding the mechanisms of cell differentiation (Song and Li 2021).

Several methods, including tradeSeq, PseudotimeDE, and Monocle3, have been developed to detect Differentially Expressed (DE) genes along the inferred Pseudotime (Van Den Berge et al. 2020, Song and Li 2021, Cao et al. 2019). However, these methods face challenges. Some are limited to detecting linear patterns, while others struggle with modelling gene expression along branching paths and Pseudotime simultaneously (Song and Li 2021). Additionally, Generalized Additive Models (GAMs), commonly employed by these methods, are prone to overfitting and pose interpretability challenges due to their black-box nature. Moreover, as the dimensionality of the dataset increases, the computational performance of GAMs can significantly decline. Previous studies have highlighted that zero-inflated (ZI) versions of GAMs and Generalized Linear Models (GLMs) often result in a power loss, leading to high False Negative Rates (FNR) (Silverman et al. 2020).

In response to these challenges, we introduce scMaSigPro, an adaptation of maSigPro (Conesa et al. 2006, Nueda, Tarazona, and Conesa 2014), a method initially developed for serial analysis of transcriptomics data, to the analysis of scRNA-seq trajectory data. scMaSigPro is specifically designed to detect genes that change their expression in Pseudotime and between branching paths, thereby identifying genes critical to cell-fate decisions.

Methods

maSigPro utilizes Polynomial Generalized Linear models (Poly-GLM) to model gene expression over time across one or multiple experimental groups, identifying genes that exhibit differential expression profiles over time. However, the direct application of maSigPro to scRNA-seq data is challenging due to the higher levels of intrinsic noise from stochastic gene expression, sparsity resulting from technical dropouts, and the inherently high dimensionality of the data.
To tailor the maSigPro algorithm for scRNA-seq trajectory analysis, three major modifications were made: 1) A binning strategy was introduced to create pseudo-bulk or binned \textit{Pseudotime} points to compensate for the low signal-to-noise ratio of the scRNA-seq counts. 2) An equalizing method was employed to manage the uneven distribution of cells along a branching path. 3) The Poly-GLM model was enhanced by incorporating size factors to adjust for variations in bin sizes. The majority of \textit{maSigPro} functions have been modified to accommodate the new ‘scMaSigPro’ S4 Class, enabling parallel computation during model fitting. Additionally, new functions have been introduced to the package to facilitate the implementation of these adaptations (Figure 1, Supplementary Figure 2).

**Binning Along Pseudotime**

\textit{Pseudotime} values serve as indicators of the cells’ positions within a dynamic biological process. Therefore, grouping cells based on their \textit{Pseudotime} values effectively clusters cells that are in similar states into the same category or bin (Supplementary Figure 2).

In the \textit{scMaSigPro} workflow, the original \textit{Pseudotime} values, which are continuous, are discretized into \textit{Binned Pseudotime} values, which is the ordinal index of the bins capturing cells with similar states.

The binning along the \textit{Pseudotime} is carried out using the ‘\texttt{sc.squeeze()}’ function, and the determination of the bins (size and numbers) is guided by one of the histogram binning strategies outlined in the supplementary material, section 1.1. For clarity, we demonstrate how \textit{scMaSigPro} conducts binning using the ‘Sturges’ Rule.

Let \( N \) be the number of inferred \textit{Pseudotime} values, collected in a vector \( T_{\text{pseudotime}} = (t_1, t_2, t_3, t_4, ..., t_N) \).

The number of bins is \( B_{\text{number}} \) is calculated by \( B_{\text{number}} = \lfloor \log_2(N) + 1 \rfloor \) \( k \), where \( k \in [0.3, \infty) \) is a drop-factor, and \( \lfloor \log_2(N) + 1 \rfloor \) is the ‘Sturges’ formula.

The size of each bin, \( B_{\text{size}} \), is then defined by:

\[
B_{\text{size}} = \max(T_{\text{pseudotime}}) - \min(T_{\text{pseudotime}}) / B_{\text{number}}
\]

After the determination of the \( B_{\text{size}} \), \textit{scMaSigPro} partitions the entire range of the original \textit{Pseudotime} values and assigns an ordinal index to the bins. The bins are structured as left-closed and right-open intervals, except for the last bin, which is closed on both ends.

Finally, \textit{scMaSigPro} aggregates the cell counts of each gene within each bin, resulting in a pseudo-bulk count per bin. This process enhances the signal-to-noise ratio. Binning is carried out separately for each branching path to maintain the distinctions in cell fate and to create expression profiles compatible with \textit{maSigPro} analysis.

**Equalising Heterogeneous Cell Distributions**

Cell states can transition at different velocities or be captured variably in the scRNA-seq dataset, leading to an uneven distribution of cells along the differentiation trajectory. Consequently, the number of cells in each bin may substantially differ, affecting the uniformity of counts aggregated across \textit{Binned Pseudotime Points} and challenging the assumptions of homoscedasticity.

To address this issue, \textit{scMaSigPro} incorporates a bin equalization step, enabled by default with the \texttt{split_bin} parameter set to TRUE. \textit{scMaSigPro} calculates the mean (\( \mu \)) and standard deviation (\( \sigma \)) of the sizes of all bins per path to establish maximum (\( \mu + \sigma \)) and minimum (\( \mu - \sigma \)) allowable bin sizes. Bins larger than the maximum size are divided, and those smaller than the minimum size can be optionally removed, ensuring uniform bin sizes. The process is repeated until all bins conform to the established size criteria.

**Polynomial Generalized Linear Model**

After obtaining the pseudo-bulk counts for each \textit{Binned Pseudotime Point} along every branching path, \textit{scMaSigPro} establishes the polynomial model by assigning dummy variables to each branch, following the approach of the original \textit{maSigPro} method (Conesa et al. 2006) for the Generalized Linear Model (GLM) described in Nueda, Tarazona, and Conesa 2014.

For clarity, let us consider two developmental trajectories stemming from common progenitor cells (\textit{ProgC1},\textit{A}), which diverge into \textit{ProgC1} + \textit{CellA} and \textit{ProgC1} + \textit{CellB}. Each cell’s trajectory is indicated by a binary variable \( j \), where \( j = 0 \) corresponds to the path leading to \textit{CellA} and \( j = 1 \) to \textit{CellB}. After applying the binning strategy discussed in the previous section over the inferred \textit{Pseudotime} of each branching path, the \textit{binned Pseudotime} will be given by \( I \in \{1, ..., B_{\text{number}}\} \) per branching path. The variable \( Y_{ij} \), representing the observed counts of gene \( i \) in bin \( j \), is modelled as follows:

\[
g(\mu_{ij}) = \beta_0 + \beta_1 t_i + \beta_2 t_i^2 + \beta_3 z_j + \beta_4 t_i z_j + \beta_5 t_i^2 z_j + \text{Offset}_{i, j}
\]

Here, \( g(.) \) is the ‘link function’ that characterizes the GLM (McCullagh and Nelder 2019), effectively transforming the linear predictor into the appropriate scale for the response variable, which is the pseudo-bulk count in this model. The term \( \mu_{ij} = E(Y_{ij}) \) represents the expected mean of the counts, conditioned on the \textit{binned Pseudotime} \( t_i \) and the binary variable \( z_j \).

The default model assumes a Negative Binomial distribution, \( Y_{ij} \sim NB(\mu_{ij}, \theta) \), and sets the ‘link function’ to the logarithmic
function, \( g(\mu_{ij}) = \log(\mu_{ij}) \), although this can be changed by the user (e.g. to ‘identity’) for other data distributions. The logarithmic link function ensures that the expected counts \( (\mu_{ij}) \) are positive, and it is particularly appropriate for the overdispersed count data typically found in scRNA-Seq experiments. The parameter \( \theta \) represents the dispersion parameter of the Negative Binomial distribution, addressing the overdispersion commonly seen in this type of data.

Finally, unlike \texttt{maSigPro} that relies on normalized expression values, \texttt{scMaSigPro} incorporates an offset value to the Poly-GLM model to adjust for disparities in total read counts per bin. This offset is calculated as the logarithm of size factors for each bin (Supplementary Material, Section 1.3).

Workflow

\texttt{scMaSigPro} implements a two-step strategy similar to the \texttt{maSigPro} method, which consists of first identifying genes with dynamic profiles and then applying stepwise regression to select the best model for each gene (Conesa et al. 2006, Nueda, Tarazona, and Conesa 2014).

Initially, \texttt{scMaSigPro} fits the full polynomial model for each gene, as outlined in equation-1, using the \texttt{sc.p.vector()} function. This model incorporates all polynomial terms to capture the full range of potential gene expression dynamics. Subsequently, the adequacy of each gene’s full model is evaluated against a simpler intercept-only model via hypothesis testing based on the log-likelihood ratio statistic as it is explained in Nueda, Tarazona, and Conesa 2014. Only genes whose full models meet the significance threshold (default is 0.05) proceed to the next stage.

In the next stage, stepwise regression is employed through the \texttt{sc.t.fit()} function to refine the models that passed the initial significance filter. This process iteratively eliminates non-significant polynomial terms and returns a model that best fits the expression profile of each gene.

In this second stage, the goodness of fit, \( R^2 \), of each optimized gene model is computed. This parameter is used to select genes with well-defined expression trends. In Linear Models, \( R^2 \) is defined from the residual sum of squares, while in Generalized Linear Models, the goodness of fit is evaluated as the percentage of deviance explained by the model (Nueda, Tarazona, and Conesa 2014).

Lastly, the list of significant genes is selected based on whether they have a relatively high \( R^2 \) associated with its model (Nueda, Tarazona, and Conesa 2014). The selected genes can be subject to clustering, effectively revealing trends in \textit{Pseudotime}.

Evaluation with Synthetic Data

We used Splatter (Zappia, Phipson, and Oshlack 2017) to generate synthetic datasets, including 2,000 genes and 3,000 cells, featuring a bifurcation topology (Supplementary Figure 3).

Splatter mimics differential expression by simulating fold changes between the start and the end of the cell trajectory. Let the base expression of gene \( g_i \) be denoted as \( \alpha_i \) in branching path \( path_i \), and the expression of \( g_i \) at the end of \( path_i \) be \( \gamma_i \). The effective change in expression, denoted as \( \delta_i \), across the simulated \textit{Pseudotime} can be obtained by calculating the difference between \( \alpha_i \) and \( \gamma_i \). The expression for \( \delta_i \) is given by:

\[
\delta_i = \gamma_i - \alpha_i
\]

If the value of \( \delta_i \) is not equal to 0, then gene \( g_i \) is called to be differentially expressed in \textit{Pseudotime} along pathway \( path_i \) (Figure 2A-C). If the value is 0, we consider the gene not differentially expressed (Figure 2D).

To closely replicate real scRNA-seq data, simulation parameters were estimated from the raw counts reported in Setty et al. 2019. Specifically, we estimated the inherent characteristics of the real dataset such as library size, dropouts and biological coefficient of variation (BCV). Finally, we used Splatter’s \texttt{splatSimulate(method = ‘paths’)} function to simulate a differentiation process. This base simulation displayed approximately 38% sparsity, with each branching path comprising around 1,500 cells evenly distributed. A total of 550 genes were simulated to be DE along the trajectory (Supplementary Material, Section 2).

To assess the influence of dropout events, four synthetic datasets with increasing sparsity levels were generated, ranging from 60% to approximately 90% in 10% increments (Supplementary Figure 4A-D). We adjusted the \texttt{dropout.shape} parameter in the \texttt{splatSimulate()} function to increase sparsity levels in combination with the base 38% sparsity learned from the data. To evaluate scenarios where certain cell states are captured more frequently than others, resulting in uneven distribution of cells across the \textit{Pseudotime} (or skewed cell distributions), we created four more datasets (Supplementary Figure 5A-D). In Splatter, the degree of skewness is regulated by the \texttt{path.skew} parameter (default at 0.5), with settings near 0 producing paths where cells predominantly accumulate at the end (indicating an abundance of differentiated cells), and settings close to 1 biasing cell accumulation towards the start (indicating a predominance of stem cells). Two datasets were simulated with strong left skewness \( \texttt{path.skew} = 0.9 \) & 1 and two with strong right skewness \( \texttt{path.skew} = 0 \) & 0.1.

In our final set of simulations, we generated four datasets to represent variable developmental times, which can result in branches of differing lengths (Supplementary Figure 6A-D). Specifically, a quick transition of a cell to its end state is characterized by a shorter \textit{Pseudotime} range, while a more gradual transition results in longer \textit{Pseudotime} ranges. To model such scenarios, we adjusted the \texttt{path.mSteps} parameter, which specifies the number of steps between the start and endpoints of a path.

For all simulated datasets, full models (i.e. all polynomial terms with equation-1) were initially computed using a cubic polynomial degree to capture potential non-linear expression trends along \textit{Pseudotime} with \texttt{sc.p.vector()} function. Subsequently, stepwise regression was applied to select significant full models (\texttt{p-value <= 0.05}) using \texttt{sc.t.fit()} function. The \( R^2 \) of the final significant models containing only significant terms were obtained.

We generated Operating Characteristic (ROC) curves by varying the \( R^2 \) threshold from 0.1 to 0.95 in increments of 0.05. Genes were classified as DE if the model’s \( R^2 \) exceeded the threshold and not DE otherwise. Performance metrics such as True Positive Rate (TPR), False Positive Rate (FPR), Accuracy, and F1 Score were calculated based on these classifications.

Comparison with \texttt{tradeSeq}

We compared the performance of \texttt{scMaSigPro} with \texttt{tradeSeq} (Van Den Berge et al. 2020), a negative binomial GAM framework, that fits cubic splines to each branching path on the same K cubic basis functions. In addition, \texttt{tradeSeq} includes covariates that represent each cell and offsets in the model.
(Supplementary Material, Section 3). Apart from the differences between scMaSigPro and tradeSeq fitted models, they differ in the gene selection criteria. While tradeSeq prioritizes genes based exclusively on their p-values, scMaSigPro also incorporates a measure of the quality of the model, the percentage of deviance explained by the GLM, i.e. $R^2$.

To compare and evaluate the difference between the tradeSeq and scMaSigPro, we simulated another dataset with 3000 cells and 2000 genes (Supplementary Figure 7). We simulated all the stated complexities in the previous section, i.e., sparsity, skewness and different branch lengths, in the same dataset. Globally, 30% of the genes were simulated to be DE.

In accordance with the guidelines outlined in Van Den Berge et al. 2020, we first normalized the raw simulated counts using the ‘FQ normalization script’ available from the tradeSeq package. Subsequently, we used the ‘evaluket()’ function from tradeSeq to determine the optimal number of knots, ranging from 3 to 15 (Supplementary Figure 8). This analysis identified 5 as the optimal number, which we then applied in the ‘fitGAM()’ function, maintaining all other parameters at their default settings. After completing the model fitting, we employed the ‘patternTest()’ and ‘diffEndTest()’ function to generate lists of DE genes together with their corresponding p-values.

The comparative analysis was conducted using the iCOBRA framework (Soneson and Robinson 2016, Supplementary Material, Section 3.2). To make scMaSigPro results compatible with the iCOBRA package, non-significant genes (p-value $\geq 0.05$ after ‘sc.p.vector()’ function) and gene with low $R^2$ ($R^2 \leq 0.6$ after ‘sc.t.fit()’ function) were assigned a p-value of 1. This allowed the evaluation of gene rankings and filtered genes with the iCOBRA framework.

**Computational Runtime**

Taking advantage of the parallel processing capabilities offered by both packages, we assessed their computational runtime using 1 and 8 CPU cores on a laptop with Core i7-9700K clocked at 2.90GHz with 8 CPU cores (4 + 4 virtual). We used R’s ‘microbenchmark’ package to evaluate the wall clock time taken by individual functions. For tradeSeq, ‘fitGAM()’ and ‘patternTest()’ where monitored, while ‘sc.p.vector()’ and ‘sc.t.fit()’ were evaluated in scMaSigPro. For any of the comparisons, we did not consider the additional time that might be used to estimate the optimal number of knots with the ‘evaluket()’ function.

We first evaluated a sub-sample of 1500 cells and 1000 genes from the simulated dataset, which was used for comparison. Next, we evaluated a bigger simulated dataset of 6000 cells, each branch having 3000 cells ten times. Unfortunately, we could not compare tradeSeq with more than 6000 cells with 8 CPU cores on a laptop with 32 GB memory. Performing such computation with tradeSeq produces large memory overheads, making the R session crash. However, scMaSigpro is scalable and can efficiently use memory. Thus, to evaluate further, we simulate 4 datasets with an increasing number of cells, up to 10,000 cells per branching path. In this evaluation, we considered the ‘sc.p.vector()’ and ‘sc.t.fit()’ functions individually and evaluated them ten times.

**Public Data**

We examined the CD34+ enriched stem/progenitor cells (HSPCs) dataset from Setty et al. 2019, which includes differentiating HSPCs from three healthy donors. The raw FastQ files were downloaded from the European Nucleotide Archive (ENA), having project accession as PRJEB37166. Cell Ranger v7 was used to align the reads to the genome (Zheng et al. 2017). Specifically, we utilized the ‘GRCh38.primary_assembly’ from GENCODE, along with the annotation GTF ‘gencode.v43.annotation.filtered.gtf’. The Cell Ranger ‘count’ was run independently for each donor to generate three digital feature barcode matrices.

We used Seurat (version 5.0.1) for quality control and filtering (Y Hao, Stuart, et al. 2023) (Supplementary Table 3). As per methods in Setty et al. 2019, we also removed the cell cycle effects from each donor. Specifically, we used the reference ‘Human Bone marrow’ from Azimuth References, which consisted of 297,627 bone marrow cells from 39 donors and three different studies performed with 10x Genomics v2 chemistry to match with Setty et al. 2019 (Y Hao, S Hao, et al. 2021).

Next, we sub-sampled each dataset to exclude cell types typically not originating from the bone marrow, such as stromal cells. Using Monocle3, trajectories were inferred with HSCs designated as the starting cells. Specifically, the trajectory
graph was learned using default parameters, except for the ‘close loop’, ‘use partition’, and ‘prune graph’, which were set to FALSE (Supplementary Figure 15). For each of the donors, a distinct set of branching paths was analyzed. The results for Donor-1, showcasing Prog Megakaryocytes (ProgMk) and Early Erythrocytes (EarlyE) originating from Erythro-Myeloid Progenitors (EMPs/MEPs), are presented here. Results for Donor-2/3 are detailed in the Supplementary Material, Section 4.

The inferred Pseudotime and the definitive assignment of each cell to its corresponding branch served as the input for scMaSigPro. We applied cubic poly-GLM to accurately model the nonlinear gene expression patterns observed as MEPs differentiate into ProgMk or EarlyE. Significant genes expected to mediate differentiation towards ProgMk were selected by scMaSigPro at an $R^2$ of 0.7. The resulting gene list was used for clustering and a Gene Ontology (GO) enrichment analysis, using all detected genes as background.

**Results**

**Synthetic Data**

Synthetic data was used to assess the performance of scMaSigPro across various scenarios with different levels of sparsity, cell distribution skewness, and branch lengths.

**Zero-Inflation:** scMaSigPro demonstrated a high True Positive Rate (TPR) close to 90% and a False Positive Rate (FPR) below 5% when Zero-Inflation (ZI) levels were set to 60% and 70% (Figure 2B) at an $R^2$ range of 0.45 to 0.6. A minor reduction in TPR was noted as ZI increased to 80%, while extreme ZI levels significantly affected the detection of true positives (Supplementary Figure 4E-H). Nonetheless, the FPR consistently stayed below the 0.05 threshold in all cases. These findings underscore the robustness of scMaSigPro to varying degrees of sparsity and its effectiveness in minimizing false discoveries under realistic data scenarios. For datasets with more than 80% ZI, data imputation is advisable prior to scMaSigPro analysis.

**Skewness:** scMaSigPro effectively controlled FPR in datasets with high skewness while maintaining the TPR around 80% (Figure 2B). The direction of skewness influenced the FPR (Supplementary Figure 5E-L), which significantly dropped upon activating the split_bins option, which splits disproportionately large bins into smaller ones to improve performance (Figure 2B).

**Unequal Lengths of Branching Paths:** In cases of branching paths were of similar length, scMaSigPro showed optimal performance with the TPR approaching 90% and FPR below 5% (Figure 2B). In scenarios where the lengths of branching paths significantly differ (i.e. > two folds), the TPR decreased to 50% at an $R^2$ threshold of 0.6 (Supplementary Figure 6E-H), while the FPR was maintained between 5-10%. Given these findings, subsampling the dataset to ensure more uniformity in trajectory lengths can potentially enhance the scMaSigPro accuracy, mitigating the elevated FPR.

**Comparison with tradeSeq**

The ‘diffEndTest()’ function from tradeSeq, which identifies genes that show differential patterns towards the end of Pseudotime and between the branching paths, performed comparably to scMaSigPro, while the ‘patternTest()’ function from tradeSeq that detects differential patterns across and between the branching path Pseudotime resulted in a higher number of False Positives (FPs) (Figure 2F). In our comparative analysis, scMaSigPro demonstrated better performance over tradeSeq in controlling the FPR; however, the TPR was only marginally better (Figure 2F).

We investigated the characteristics of False negative (FN) genes, i.e., those incorrectly identified as not DE by each method. We observed that scMaSigPro struggled to identify genes with low fold changes, particularly when a gene is DE in only one of the branching paths and exhibits a stable expression pattern in the other (Supplementary Figure 9,10). Such patterns were usually detected by the tradeSeq’s ‘diffEndTest()’ and ‘patternTest()’ functions.

**Computational Runtime**

When utilizing a single CPU, scMaSigPro proved to be four times faster, resulting in a lower carbon footprint (30.53 mg CO2e) compared to tradeSeq (122.14 mg CO2e) (Lannelongue, Grealey, and Inouye 2021, Supplementary Figure 11, Supplementary Table 1). This performance advantage increases with the use of 8 CPUs, where scMaSigPro is five times faster than tradeSeq. Also, the performance was consistent on the larger dataset of 6000 cells (Supplementary Figure 12).

Evaluation of the individual function of scMaSigPro, with datasets up to 10,000 cells per branch, shows a linear increase in the runtime with an increasing number of cells (Supplementary Figure 13). The ‘sc.p.vector()’ function was found to be considerably faster than the ‘sc.t.fit()’ as a result of global model fitting. Although ‘sc.t.fit()’ function consumes more time due to stepwise regression, it greatly benefits from more cores, which allow faster evaluations, simultaneously keeping the memory overhead minimal, which is typically challenging in R.

**Public Data**

The cell types were annotated using Azimuth References (Bone Marrow) based on the cell-specific markers listed in the Supplementary Material Table 2. The top 6000 Hypervariable genes were used for the analysis, including UMAP visualization (Supplementary Figure 14, Supplementary Table 3). Next, after pre-processing the data with Seurat, we transferred the UMAP embedding to Monocle3 for trajectory inference and selected different branching paths per donor for scMaSigPro input (Supplementary Figure 15).

The dataset from Donor-1 (Age 35, Male) contained 195 cells and 6000 genes. scMaSigPro divided Erythrocyte-Lineage into six bins and ProgMk-Lineage into three bins, averaging 19 and 27 cells per bin, respectively (Supplementary Figure 16). Using a cubic polynomial model, Erythrocyte-Lineage as a reference and a $R^2$ threshold of 0.7, scMaSigPro identified 300 features exhibiting significant and consistent non-flat expression trends (Supplementary Table 4). Among these, 278 genes showed significant change in expression between ProgMk-Lineage and Erythrocyte-Lineage with Pseudotime and, 22 showed significant change in expression with Pseudotime, but without significant differences between the lineages. Subsequently, clustering analysis of 278 genes using the ‘sc.cluster.trend()’ function resulted in six clusters (Supplementary Figure 17).

Genes in Cluster 2 (159 Genes) and Cluster 3 (81 Genes) showed an increasing profile in ProgMk-Lineage (Supplementary Figure 3B). This gene list of 240 genes was enriched in terms such as ‘response to wound healing’, ‘blood coagulation’, and ‘platelet activation’, pointing towards the differentiation process of the HSPCs to Megakaryocyte Lineage resulting in
platelets (Supplementary Figure 18). Manual inspection of the gene list confirmed the presence of known markers such as LTBP1, ARHGAP6, GP9, SPX, SELP, RBPMS2 and WFDC1 (Supplementary Figure 19). Glycoprotein IX (GP9) is critical in platelet adhesion and aggregation, particularly under high-shear stress conditions. Further results for additional donors are outlined in the supplementary materials.

Discussion

scMaSigPro unlocks the utilization of poly-GLM models for differential trajectory analysis of scRNA-seq data by applying a Pseudotime binning approach that preserves the original cell order along the Pseudotime while reducing data sparsity and heterogeneity. Previous studies have shown that pseudo-bulk methods overperform approaches that modelled cells individually (Murphy and Skene 2022). Our benchmarks confirm that Pseudotime binning significantly enhances the sensitivity and yields a consistently high TPR over a wide range of realistic data scenarios. For datasets with extreme ZI (over 85%), we recommend adjusting the drop factor (\(k\)) to increase compression, yielding fewer but more balanced bins per path.

The `sc.squeeze()` function in scMaSigPro provides various adjustable parameters that can be tailored according to the range of the Pseudotime values. For example, Rice’s rule is suitable when Pseudotime values span a broad range. Alternatively, Doane’s Binning formula is preferred for non-normally distributed Pseudotime values as it accounts for skewness. The drop_tails parameter is useful when the user wants to focus only on the comparable Pseudotime segments of the bifurcating trajectories. Activating this option will eliminate any additional bins that extend beyond the Pseudotime bin of the shortest path. The prune_bin parameter removes bins not meeting the minimum size criteria (\(\mu - \sigma\)).

Parameters in the `sc.p.vector()` and `sc.t.fit()` functions offer flexibility for adapting to the characteristics of the single-cell data. For instance, in our analysis of the HSPC public dataset, cell-cycle effects were removed, and counts were transformed into continuous values. In such cases, the link function can be set to be a Gaussian distribution. Similarly, if counts are already normalised, users may opt to disable the incorporation of offsets in the GLM. Collectively, the functions `sc.squeeze()` (new to scMaSigPro), `sc.p.vector()` and `sc.t.fit()` (adapted from maSigPro), provides comprehensive control over the model fitting process. scMaSigPro is therefore suited for diverse topologies and cell state compositions typical of cell differentiation scRNA-seq experiments. Compared to methods based on GAMs, scMaSigPro improves precision and significantly reduces computational time, representing a sustainable alternative to the processing of large-scale datasets.

In conclusion, scMaSigPro efficiently identifies DE genes along branching paths and the Pseudotime, controlling the FPR effectively. Its compatibility with existing R packages, such as Monocle3, enhances its integration in scRNA-seq analysis workflows. Future developments may include an extension to spatial transcriptomics and changes in cell communication during cellular dynamic processes.

Author Contributions

AC conceived the project. PS performed analyses. PS and MBC wrote the R-package. AC, SG and MJN supervised the project.

Acknowledgement

This work received funding from the European Union Framework Programme for Research and Innovation Horizon 2020 under the Marie Sklodowska-Curie Grant Agreement No 953407.

References


Conesa, Ana et al. (May 1, 2006) “maSigPro: a method to identify significantly differential expression profiles in time-course microarray experiments”. In: Bioinformatics 22.9, pp. 1096–1102. ISSN: 1367-4811, 1367-4803.

Deconinck, Louise et al. (Sept. 2021) “Recent advances in trajectory inference from single-cell omics data”. In: Current Opinion in Systems Biology 27, p. 100344. ISSN: 24523100.


Nueda, María José, Sonia Tarazona, and Ana Conesa (Sept. 15, 2014) “Next maSigPro: updating maSigPro bioconductor package for RNA-seq time series”. In: Bioinformatics 30.18, pp. 2598–2602. issn: 1367-4811, 1367-4803.


Silverman, Justin D. et al. (2020) “Naught all zeros in sequence count data are the same”. In: Computational and Structural Biotechnology Journal 18, pp. 2789–2798. issn: 20010370.


Song, Dongyuan and Jingyi Jessica Li (Apr. 29, 2021) “PseudotimeDE: inference of differential gene expression along cell pseudotime with well-calibrated p-values from single-cell RNA sequencing data”. In: Genome Biology 22.1, p. 124. issn: 1477-760X


The workflow of *scMaSigPro* begins with the creation of an S4 object, followed by binning, model fitting, and visualization. Functions in yellow are adapted from *maSigPro*, while functions in pink are newly implemented in *scMaSigPro*.

803x144mm (300 x 300 DPI)
(A-D) Simulated ground truth expression of genes and corresponding patterns. (ABC) Differentially Expressed Genes in Pseudotime. (A) A simulated gene showing similar expression change in Pseudotime in both the branching paths. (B) A simulated gene showing dissimilar expression change in Pseudotime in both the branching paths. (C) A simulated gene showing expression changes only in one path in Pseudotime. (D) No difference along the Pseudotime or among the branching paths is considered Not Differentially Expressed. (E) ROC curves for three synthetic datasets show the performance at varying R² values. The triangles represent values at R² of 0.6. (F) Performance comparison between scMaSigPro and tradeSeq, the former exhibits tighter FPR control with a slight increase in TPR. Genes that do not meet the scMaSigPro selection criteria (p-value ≤ 0.05 and R² ≥ 0.6) are assigned a p-value of 1 during the iCOBRA evaluation. This assignment leads to a flattened ROC curve when the false positive rate exceeds 0.05.
Analysis of genes with differential trajectories in the HSPC dataset. (A) Clustering of genes identified as differential by scMaSigPro for Donor-1. Clusters 2 and 3 (240 genes) showed an upward trend in the Megakaryocyte Lineage. (B) scMaSigPro models of GP9 expression reveal increased expression as MEPs differentiate into ProgMk and stable expression within the Erythrocyte Lineage.

1033x387mm (236 x 236 DPI)