Differential abundance testing on single-cell data: leveraging heterogeneity and enhancing scalability

Master's thesis

Alice Kluzer
alkluzer@ethz.ch

Department of Biosystems Science and Engineering
ETH Zürich
Cancer Research UK Institute
University of Cambridge

Supervisors:
Dr. Michael Morgan
Prof. Dr. Barbara Treutlein
Prof. Dr. John Marioni

May 9, 2022

Abstract

Technological advances in microfluidics and decreases in sequencing costs have massively expanded the scale of single cell experiments, with large cell atlases and population-wide cohort studies becoming increasingly common. Such vast, complex datasets present a unique set of challenges, including an increase in the volume of data being generated, hierarchical experimental designs with clusters of non-independent observations and multiple sources of heterogeneity. Consequently, there is a growing need for computational tools to analyse single-cell RNA sequencing (scRNA-seq) data that are computationally efficient and statistically suitable. Here we present Milo+, a statistical framework that performs very fast differential abundance testing on a k-nearest neighbour graph by approximating distances with graph-based metrics, and leverages heterogeneity by incorporating a negative binomial generalised linear mixed model to account for additional sources of variability. Using simulations and scRNA-seq data, we show Milo+ scales to datasets of over a million cells and accommodates complex experimental designs by incorporating random effects.

Introduction

The recent emergence and development of single-cell high-throughput sequencing now offers the possibility to profile single cells at an unprecedented scale, highlighting cellular heterogeneity that was masked with bulk sequencing. Since the first single cell RNA-sequencing (scRNA-seq) experiment in 2009, characterising just 8 cells [1], technological advances in microfluidics and combinatorial indexing have massively increased sequencing throughput and decreased costs [2, 3]. Less than a decade later, a 1.3 million brain cell dataset was published by 10X genomics [4]. With the creation of several human and animal atlases [5–7], as well as population-wide cohort studies [8], the size and complexity of single cell experiments continues to expand. To take advantage of these large volumes of data, efficient computational tools are needed.

An emerging challenge in scRNA-seq is identifying cellular subpopulations whose abundance differs between different experimental states, such as healthy vs diseased individuals, control vs drug treatment or across different timepoints of a developmental process [9]. Common computational workflows for differential abundance testing rely heavily on clustering cells, usually in an unsupervised manner, prior to measuring the proportion of cells from each condition in individual clusters [10–13]. Clustering may however be inadequate in situations where no biologically meaningful discrete groups of cells are present in the data, for example when considering continuous trajectories in a developmental setting [14].

Alternative methods have been proposed to solve the challenge of performing differential abundance analysis without clustering. For example, Zhao et al.

recently published DA-seq, which computes a local differential abundance measure for each cell based on its k-nearest neighbours (K-NNs), but is limited to pairwise comparisons [9]. Burkhardt et al. proposed MELD, a method that estimates the likelihood of observing each cell in each experimental condition using a K-NN graph. However, MELD does not provide any statistical measures of confidence and does not incorporate experimental design in its likelihood estimation [15]. Both methods are thus limited in their use for datasets with more complex designs, such as continuous variables and confounding factors. An alternative approach for differential abundance testing that addresses these challenges was recently published [16]. Milo is a statistical framework that performs abundance testing by translating cells onto a K-NN graph and assigning them to overlapping neighbourhoods. Milo avoids the need for discrete clustering and leverages the flexibility of generalized linear models [16].

However, the Milo framework is not designed to cope with increasingly large-scale datasets of millions of cells due to its reliance on time- and memory-expensive distance calculations in reduced dimensional space. Moreover, scRNA-seq studies comprising large numbers of cells over many individuals present new statistical problems that need to be addressed. Two main challenges include the presence of highly structured, hierarchical data with clusters of non-independent observations (e.g. repeated measures within individuals), and additional sources of heterogeneity, for example genetic relatedness between donors. [17].

One of the standard statistical approaches to handling such data is a generalized linear mixed model (GLMM). GLMMs extend traditional linear models to include both fixed and random effects, as well as non-normal distributions, such as count data [18]. Common examples of random effects are blocks in experiments replicated across sites or time, and variation among individuals. Ignoring random effects can lead to pseudoreplication, as observations within groups are assumed to be independent, but are in fact correlated. Treating a random effect as fixed requires fitting a coefficient to each level of the variable and uses up many degrees of freedom, as well as limiting the scope of inference [19]. Using GLMMs where appropriate can therefore improve the accuracy of parameter estimation, provide informative estimates of variance and allow results to be extrapolated to unmeasured groups [17].

Here, we propose Milo+, a model extension that addresses these challenges by (i) moving away from distance calculations in reduced dimensional space to a graph-based approach, enabling the analysis of potentially millions of cells and (ii) incorporating a negative binomial generalised linear mixed model (NB-GLMM) to model additional sources of variability, thus increasing statistical power in large datasets. Overall, Milo+ enhances scalability and leverages heterogeneity in complex scRNA-seq datasets.

Results

Improving speed and scalability with a graph-based implementation

Milo models the difference in abundance of cells among experimental conditions using graph neighborhoods. The original Milo workflow proceeds via a 5 step process: 1) A k-nearest neighbour (K-NN) graph is built, representing the relationships between cells in high-dimensional space; 2) Representative neighborhoods are defined with a refined sampling scheme that controls the overall number of neighborhoods whilst providing a high coverage of the K-NN graph. Neighborhoods are allowed to overlap, thus avoiding discrete clustering; 3) For each neighborhood, the number of cells in each condition is counted; 4) Cell counts are modelled with a negative binomial generalised linear model (NB-GLM) to determine differentially abundant neighborhoods; 5) The false discovery rate (FDR) is corrected with a spatial FDR algorithm, which weights the P-values by the reciprocal of the distance in principal component (PC) space to the kth nearest neighbor of each index cell [16]. The original implementation relies heavily on calculating Euclidean distances in reduced dimensional space with a runtime complexity of $O(n^2)$, which is a bottleneck for analyses involving many hundred thousands of cells. In Milo+, we therefore propose to approximate PC distances with graph-based metrics, thus alleviating the computational burden of performing distance calculations (Fig. 1a).

The first development in Milo+ is in the refined sampling scheme used to define neighborhoods. A neighborhood is defined as all the cells that are connected to an index cell by an edge in the K-NN graph. To select the neighborhood index cells, a refined sampling scheme was developed in the original Milo publication. The refined sampling scheme defines a more representative subset of cells by calculating the mean PC position of all the K-NNs of a randomly sampled index cell and defining the new refined index as the cell closest to this mean position [16]. Here, we use a purely graph-based approach to approximate PC distances. Starting with an induced subgraph of each randomly sampled index cell and its K-NNs, the cell with the highest number of triangles is selected as the refined index cell. Triangles are used as a measure of graph connectivity, and are defined as a set of three vertices where each possible edge that could exist between them is present [20]. The graph-based sampling scheme in Milo+ performs very similarly to the original Milo implementation; in both cases, fewer, yet larger and more representative neighborhoods are selected (Fig. 1b), as initial random samplings converge to the same refined index cell (Supplementary Fig. 1a).

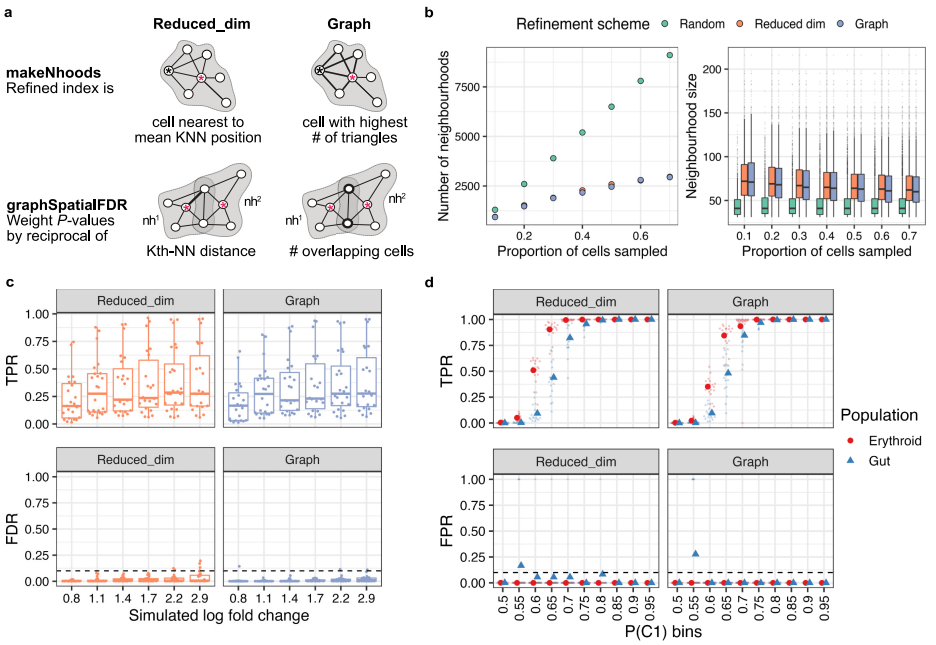

Fig. 1 Graph-based Milo+ is a good approximation of the original Milo implementation. **a**, Schematic of the differences between the Milo reduced-dimensional space workflow and the Milo+ graph-based workflow. Triangles are used as a local measure of connectivity in the graph implementation, and are defined as a set of three nodes where all possible edges exist between them. Black asterisks denote the initial randomly sampled index cells; red asterisks denote the refined index cells. Bold lines are used to highlight the differences between the two implementations. **b**, Sampling with either a reduced-dimension-based or graph-based refinement scheme leads to fewer (left panel) and larger (right panel) neighborhoods compared to random sampling. Boxplots show the median and interquartile ranges. Whiskers extend to the largest value no further than 1.5x the interquatile range. Outlier data beyond the whiskers are plotted individually. **c**, True positive rate (TPR) and false discovery rate (FDR) of the reduced-dimension and graph-based workflows when applied to the mouse gastrulation dataset with increasing simulated fold changes of cell populations. Boxplots show the median and interquartile ranges for eight different populations with 3 replicates each. All data points are plotted. **d**, TPR and FPR of the reduced-dimension and graph-based workflows when applied to cells of different bins of P(C1), the cell probabilities used to define differentially abundant regions. Bin sizes are 0.05 (x axis label denotes lower bin value). The plotted data shows 36 simulations across two populations. The filled in data points are the mean bin value.

The second advance in Milo+ is in the spatial FDR procedure used to account for multiple hypothesis testing. In Milo, per-neighborhood P-values are weighted by the reciprocal of the kth nearest neighbor distance. This approach, however, relies on calculating Euclidean distances within neighborhoods, which is computationally intensive. We adapt this procedure and weight each P-value by the reciprocal of the number of overlapping cells between neighborhoods (Fig. 1a, Supplementary Fig. 1b). To demonstrate the equivalent performance of our new approximation and the original approach, we create a ground-truth dataset with simulated regions of differential abundance between two conditions (C1 and C2) based on a real single cell atlas of mouse gastrulation (64,018 cells) using

the same approach introduced in the original Milo publication [16]. Briefly, this involves generating per-cell probabilities (P(C1)) and using these to assign cells to differentially abundant regions based on a threshold value. We vary both the differentially abundant cell population and the simulated fold change. Both the reduced-dimension and the graph implementation detect simulated differential abundance with high sensitivity and low FDR, with very similar results (Fig. 1c). The sensitivity limitations of both methods are also very similar, and vary depending on cell population (Fig. 1d), as described in the original Milo paper [16].

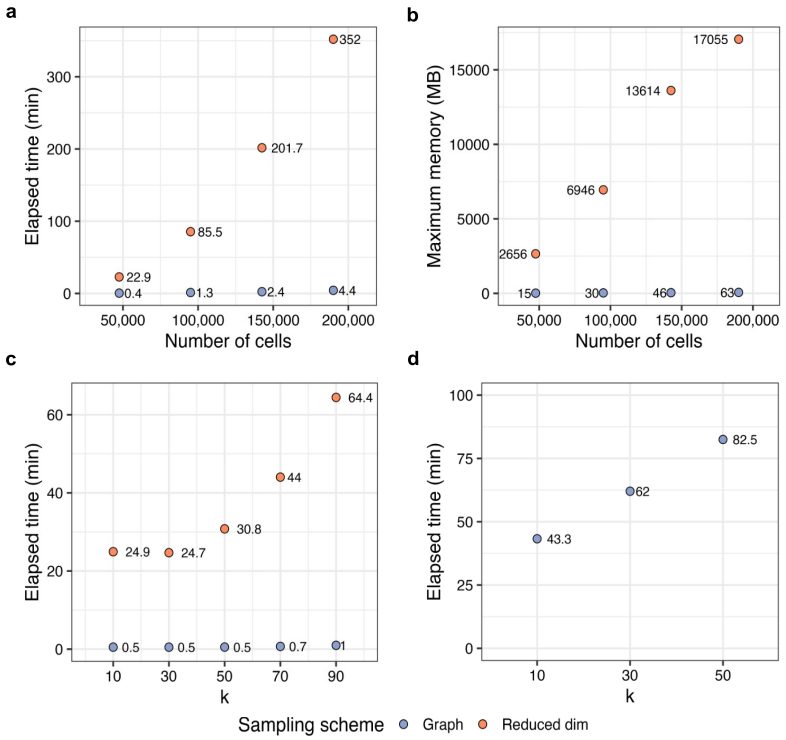

Fig. 2 Milo+ is faster and more scalable than the original Milo implementation. **a**, Run time (y-axis) of the Milo reduced-dimension workflow compared to the Milo+ graph-based workflow. Each point represents a downsampled subset of a simulated dataset. Labels show the elapsed system time in minutes. **b**, Maximum memory (y-axis) required to run the Milo reduced-dimension and the Milo+ graph-based workflows. Each point represents a downsampled subset of a simulated dataset. Labels show the maximum memory usage in megabytes (MB). **c**, Run time (y-axis) of the Milo reduced-dimension workflow compared to the Milo+ graph-based workflow for different values of k on a 50,000 cell simulated dataset. Labels show the elapsed system time in minutes. **d**, Run time (y-axis) of the Milo graph-based workflow on a $\sim$ million (982,538) cell dataset. Labels show the elapsed system time in minutes.

Next, we tested the scalability and speed of the Milo+ graph-based implementation compared to the original Milo workflow. For this, we ran both Milo and

Milo+ on a simulated dataset of 200,000 cells, subsetted at different proportions (25 - 100 %). We measured the amount of time and memory required to execute the two workflows, from the initial K-NN graph building to differential abundance testing. Notably, Milo+ is two orders of magnitude faster than the original Milo implementation, running large analyses with 200,000 cells in under 5 minutes (Fig. 2a). Milo+ also requires significantly less memory, which can easily be accommodated by any common desktop computer (Fig. 2b). Additionally, we measured the amount of time required to run both implementations for different values of k, using a 50,000 cell subset of the dataset (Fig. 2c). The elapsed time increases exponentially with increasing values of k due to the $O(n^2)$ scaling of distance calculations, where the number of nodes, n, is defined by k, as distances are computed between k nearest neighbors in a neighborhood. Again, Milo+ offers considerable speed improvements.

Moreover, since we foresee the scale of scRNA-seq experiments will increase significantly in the future, we also tested the scalability of the Milo+ workflow on a million cell dataset. We ran Milo+ on 982,538 cells (Fig. 2d) over three different values of k. This analysis demonstrates that Milo+ outperforms Milo in terms of speed and memory by orders of magnitude and is able to analyse extremely large datasets within a reasonable timeframe, with an analysis of one million cells taking about an hour.

In summary, we show that Milo+ generates very similar results to the original Milo workflow, whilst offering significant speed and memory improvements.

Modeling random effects with a negative binomial generalised linear mixed model (NB-GLMM)

In addition to speed and scalability issues, large scRNA-seq experiments across many individuals present the challenge of needing to incorporate additional sources of variability into differential abundance testing, such as genetic relationships between individuals, correlated observations or repeated measures. One possibility is to model these variables with random effects using linear mixed models (LMMs), which are useful when the assumptions of independence and constant variance are violated. Generalized linear mixed models (GLMMs) extend LMMs to accommodate dependent variables that are not normally distributed, such as count data that may follow a negative binomial distribution [21]. In datasets where such sources of variability exist, ignoring them may at best limit the scope of the study, and at worst confound downstream analyses, which may lead to erroneous conclusions [19]. To address these issues, we propose to leverage the flexibility of GLMMs and add these as an extension of the Milo workflow.

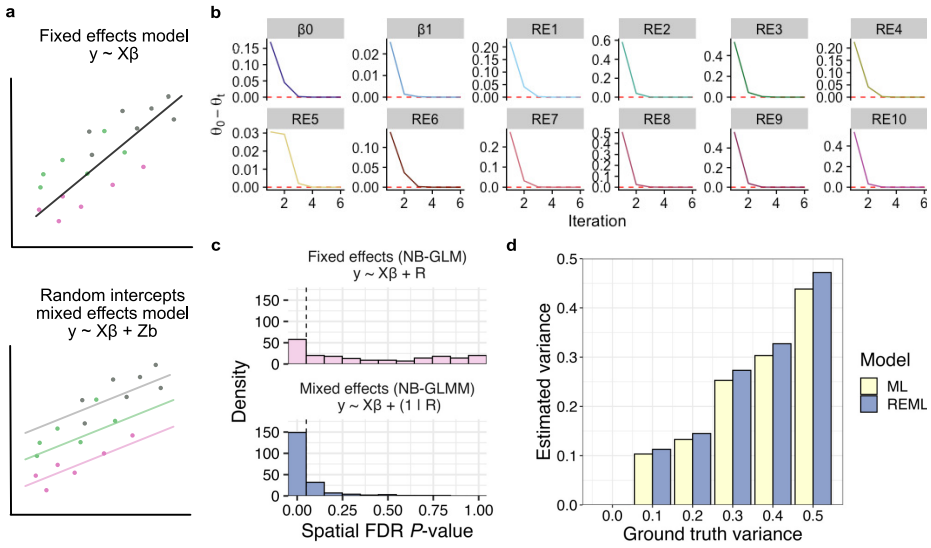

Fig. 3 Building a random intercepts NB-GLMM with pseudo-likelihood approximation. a, Schematic of a random intercepts mixed effects model compared to a fixed effects model. Data points are colored by an arbitrarily assigned experimental variable. The mixed effects model allows each level of the random effect to be modeled with a different intercept, whilst slopes remain constant. **b**, Absolute difference between parameter values (y-axis) at each iteration of estimation. Each panel represents a fixed or random effect (RE) estimate. Dashed red line denotes the convergence limit at 1e-5. Simulation parameter values: N=1000, β_0=2, β_1=0.25, σ^2=0.05, ϕ=2, random effect levels = 10, REML=TRUE. **c**, *P*-value histograms after differential abundance testing and spatial FDR correction on simulated data where a variable, R, was modeled as a fixed effect using a NB-GLM (top panel) or as a random effect using a NB-GLMM (bottom panel). The x axis was divided into 11 bins. The vertical dashed line shows the threshold at *P*-value = 0.05. The formula (1|R) denotes a random intercepts model for each level of R. Simulation parameters values: N=500, parameters sampled for 200 neighborhoods from uniform distributions with min and max values β_0=[1, 2], β_1=[0.15, 0.25], σ^2=[0.05, 0.1], ϕ=[2, 2.5], random effect levels = 10, REML=TRUE. **d**, Estimated random effect variance (y-axis) compared to the ground truth variance, for maximum likelihood (ML) and restricted maximum likelihood (REML) versions of the NB-GLMM. Simulation parameter values: N=1000, β_0=2, β_1=0.05, ϕ=0.5, random effect levels = 15.

In Milo+, we implement a random intercepts NB-GLMM, which allows differences between groups to be modeled as random effects (Fig. 3a). Modeling only fixed effects masks structure in the data and may violate underlying statistical assumptions, such as homogeneity of variance [19]. Instead, random intercepts GLMMs allow the intercepts to vary for each level of a random effect, whilst the slope remains constant. GLMMs are mathematically defined as:

$$g(E[y|b]) = X\beta + Zb$$
$$b \sim N(0, G)$$

where the expectation of y, conditioned on the random effects b, is related to the linear predictor $X\beta + Zb$ via the link function $g()$. Here, X and Z are

the fixed and random effect design matrices, respectively, and β and b are the fixed and random effect coefficients, respectively. G is the variance-covariance matrix of the random effects. GLMMs attempt to estimate both random and fixed effects. However, unlike LMMs, GLMM likelihood functions do not have simple closed-form solutions, and must therefore be solved with either numerical optimisation, which is slow, or with an approximation method, which is faster but less accurate [21]. As we require fast computation over hundreds of neighborhoods and are less interested in exact estimates, here we implement a NB-GLMM using a pseudo-likelihood approximation approach. Briefly, a modified dependent variable is created, the pseudo-variable y^*, which follows a Gaussian distribution and therefore allows us to use closed-form LMM estimating equations (see Methods) [22]. Our NB-GLMM implementation performs a doubly-iterative estimation of β and b, and the variance of the random effects, σ^2. Over iterations, the estimates converge to a final value, as shown by running our NB-GLMM on a simulated dataset with a ten-level random effect (Fig. 3b).

To illustrate the power of our NB-GLMM, we develop a simulation framework that allows us to define parameter values for β, b, σ^2 and the negative binomial dispersion parameter, ϕ (see Methods for details). The simulation framework is leveraged to create multiple neighborhood cell counts, by randomly sampling from a uniform distribution of parameter values. We then compare our NB-GLMM results to the original NB-GLM implementation (Fig. 3c). Modeling the data with a NB-GLMM results in a clear shift towards more significant P-values, as expected, since the addition of random effects generally increases the statistical power of analyses and reduces the standard errors of estimates [17].

The pseudo-likelihood approach models random effects by computing maximum likelihood (ML) estimates of b, β and σ^2. Standard ML estimation, however, suffers from downward bias in variance estimation, as it is assumes that the fixed effect estimates are exact [19]. Restricted maximum likelihood (REML) estimation was developed to average over some of the uncertainty inherent in fixed effect parameter estimation [21]. We therefore implement REML estimation to obtain a less biased estimate of the random effect variance. As shown, REML estimates of variance are closer to the true values (Fig. 3d). Overall, this analysis shows that our random intercepts NB-GLMM implemented with a pseudo-likelihood approach is superior to a NB-GLM in the presence of confounding effects.

Milo+ NB-GLMM implementation performs equally well as existing methods whilst providing additional features

To benchmark the performance of our NB-GLMM implementation, we ran a series of comparisons against existing R software packages for GLMM analysis, namely glmmTMB (glmmTMB), glmm-PQL (MASS) and glmer (lme4) [23–25]. Although these methods are able to solve NB-GLMMs accurately, they lack the ability to define custom covariance structures for the random effects, which are

required, for example, to model genetic correlations (Fig. 4a). Our implementation using actual derivatives and a pseudo-likelihood approximation is very flexible and can be recoded to allow user-defined covariance structures. One drawback of our current approach is that it currently only fits random intercept models, and thus cannot be used to model random slopes. However, these are only required in very complicated experimental designs and are rarely used. We test our NB-GLMM implementation against existing methods using simulated datasets with known parameter values, as described in the previous section. All the methods achieve very similar results for both estimated parameter values and their standard errors (Fig. 4b).

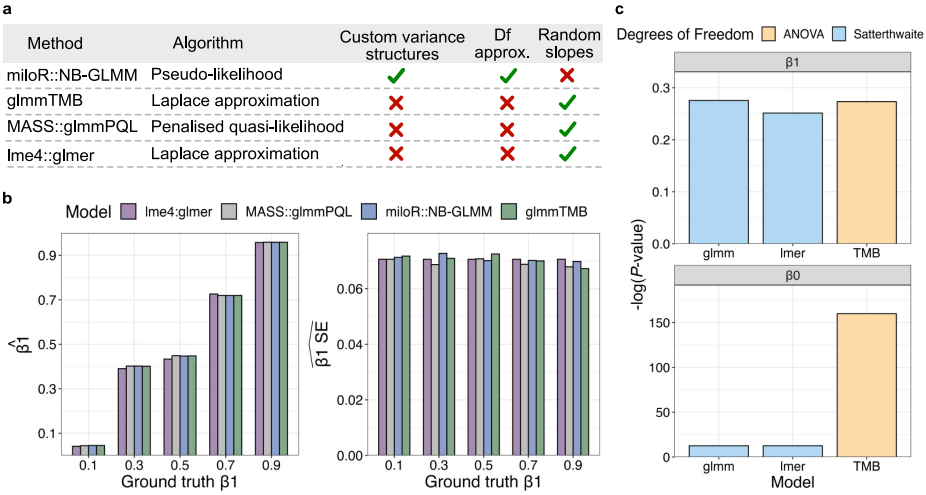

Fig. 4 Milo+ NB-GLMM performs very similarly to existing GLMM software packages and includes additional functionalities a, A table outlining the characteristics of other GLMM R software packages compared to the one implemented in Milo+. **b**, Estimated β_1 values (y-axis - left panel) and estimated β_1 standard errors (SE) (y-axis - right panel) compared to the ground truth β_1 for Milo+ NB-GLMM (Milo::NB-GLMM), lme4:glmer, MASS:glmmPQL and glmmTMB:glmmTMB. All models are run on the same simulated data. Simulation parameter values: N=500, β_0=2, σ^2=0.05, ϕ=2, random effect levels = 10, REML = TRUE. **c**, *P*-values plotted on a -log10 scale (y-axis) produced by three different models, NB-GLMM, lmer and glmmTMB. The panels show the *P*-values for β_1 (top) and β_0 (bottom). Bar plots are colored by the underlying method used for degree of freedom calculations. Simulation parameter values: N=500, β_0=2, β_1=0.1, σ^2=0.05, ϕ=2, random effect levels = 10, REML=TRUE.

After estimating parameter values, we test our fixed effect estimates against a null hypothesis using a Wald *t*-test. In GLMM inference, it is often not obvious what the appropriate degrees of freedom are, as there is inherent uncertainly in counting parameter values for a variable with multiple levels, as is the case for random effects [26]. Two main methods have been developed to approximate degrees of freedom, namely the Satterthwaite and Kenward-Roger approximations [27, 28]. *P*-values calculated with approximated degrees of freedom are less anti-conservative than *P*-values produced with likelihood ratio tests, which are

especially unreliable at small sample sizes [26]. As both methods are reported to perform similarly, and Kenward-Roger is more mathematically complex, here we implement the Satterthwaite degrees of freedom approximation as part of our NB-GLMM. None of the above GLMM R packages provide any method for approximating degrees of freedom, so in order to benchmark our implementation we extract the normally distributed pseudo-variable, y^*, and use it as an input for LMM analysis with lmer, which also implements Satterthwaite [24]. As expected, P-values computed without Satterthwaite degrees of freedom are very small (Fig. 4b). Our NB-GLMM model yields very similar results to lmer, suggesting the Satterthwaite approximation is implemented correctly.

Overall, we developed a NB-GLMM solver that performs on par with existing R packages, whilst including more accurate inference with the Satterthwaite degrees of freedom approximation and a flexible framework that allows custom covariance structures.

Discussion

As single cell datasets increase in size and complexity, there is a growing need for computational tools that scale to millions of cells and are able to correctly model complex design structures. In this study, we introduced Milo+, an extension to the Milo framework for cluster-independent differential abundance testing. Milo+ provides two major advances: first, a graph-based implementation ensures scalability to very large datasets; second, the development of a NB-GLMM expands Milo+ to a broader suite of experimental designs and accounts for inevitable donor-donor heterogeneity.

Our graph-based approach implemented in Milo+ overcomes the scaling limitations of the previous version of the workflow, resulting in speed and memory improvements of several orders of magnitude. The new framework can be run on a million cells in just over an hour with memory requirements that can easily be accommodated on any common desktop computer. An additional strength of our approach is its interoperability with multi-modal integration methods, such as Seurat's weighted-nearest neighbor (WNN) or batch balanced k nearest neighbors (BBKNN) [29, 30]. Such integration methods result in the construction of graphs derived from multiple modalities, and therefore cannot be defined by a single reduced dimensional space. As the graph-based implementation relies only on graph metrics, it expands the interoperability to a larger number of established workflows.

The addition of a NB-GLMM framework allows Milo+ to incorporate more complex experimental designs, such as replicated experimental blocks and repeated measures. As shown, modeling cell counts with a NB-GLMM results in significant increases in power, particularly in large, complex datasets with

a lot of variability. Other GLMM solvers have been proposed to model negative binomial count data [23–25]. Although most of these are well-developed and include additional functionalities, they only offer a limited set of predefined covariance structures, for example heterogeneous Toeplitz or diagonal. The Milo+ implementation is very interoperable and offers the possibility to add custom, user-defined covariance structures, such as genetic relationships between individuals, which is an essential feature for applications in quantitative genetics. Future work should explore this functionality further, perhaps by modeling genetic associations between individuals in a real, large-scale scRNA-seq dataset. Our NB-GLMM also includes degrees of freedom approximation using the Satterthwaite method, which results in fewer false positives compared to other methods [26].

Although the Milo+ NB-GLMM framework offers several advantages over existing methods, it also has its limitations. Firstly, our NB-GLMM was built for the Milo package and as such offers a relatively small, targeted set of functionalities. Specifically, other software packages for GLMM analysis can be applied to a greater range of exponential families, such as logistic and binomial distributions. However, in the context of scRNA-seq data, our NB-GLMM is sufficient, and can also be applied to data that follows a Poisson distribution. Secondly, the speed of our implementation may be a bottleneck when applied to very large sample sizes with multiple random effects. This highlights a) the importance of speeding up other aspects of the Milo+ pipeline and b) an inherent limitation of GLMMs. In certain situations (eg. where only one random effect is modeled), we might see further speed improvements using FaST-LMM-like approaches [31]. Finally, the current version only runs a random-intercepts model, as these are the most common, and as such cannot be used to model random slopes.

In summary, in this study we introduced Milo+, a statistical framework for differential abundance testing aimed at large-scale experiments with complex designs. We applied Milo+ to a variety of simulated and real datasets and showed that it is able to analyse a million cells in roughly an hour, as well as model neighborhoods with a random intercepts GLMM to increase power and incorporate heterogeneity.

Methods

Milo+

Milo+ performs differential abundance testing between conditions by modeling counts of cells in overlapping neighborhoods with a NB-GLMM. Additionally, it offers significant speed improvements by performing many steps with graph-based approximations, rather than distance calculations. A full description of the original Milo algorithm can be found in [16]. The following sections describe the advances implemented in Milo+.

(A) Graph-based definition of cell neighborhoods

We implement a new algorithm to define a representative subset of cell neighborhoods on a K-NN graph. Previously, for each randomly sampled cell c_i, we considered its K nearest neighbours and computed their mean position in PC space, $\bar{x}_j$. We then searched for the cell c_j with the smallest Euclidean distance to $\bar{x}_j$. In the graph-based implementation, we instead create an induced subgraph of cell c_i and its immediate neighbours. We then select the cell c_j within the induced subgraph with the highest number of triangles, using the `count_triangles` function from the igraph package. A triangle is a set of 3 vertices where each possible edge that could exist between them is present [20].

(B) Testing for differential abundance in neighborhoods using a NB-GLMM

To test for differential abundance between biological conditions, Milo previously fitted a NB-GLM to the counts for each neighborhood. In the current implementation, we instead extend Milo to the analysis of neighborhood counts using a NB-GLMM (see Section D for details). GLMMs are an extension of classic linear models that accommodate non-normal data and random effects. We consider a neighborhood with cell counts y and an additional source of variability that we model with a random effect b. We therefore assume that:

$$y|b \sim NB(\mu|b, \phi)$$

$$log(\mu|b) = X\beta + Zb$$

where ϕ is the negative binomial dispersion parameter, X is the fixed effects matrix containing the explanatory variables, β is a vector of the fixed effects regression coefficients, Z is the random effects design matrix and b is the vector of random effects. β and b are estimated by fitting the NB-GLMM to the count data of each neighborhood. Dispersion estimation is performed using the conditional maximum likelihood method from edgeR [32]. A t-test is used to compute a P-value for each neighborhood, where the degrees of freedom are approximated with the Satterthwaite method (see section D).

(C) Controlling the neighborhood FDR in graph space

To control the FDR, Milo was designed to account for the overlap between neighborhoods by using the spatial FDR method. Briefly, a weighted version of the Benjamini-Hochberg method is applied, where P-values are weighted by the reciprocal of the neighborhood connectivity. In the original Milo implementation, the neighborhood connectivity for each neighborhood is the Euclidean distance in PC space to the kth nearest neighbor of the index cell. In the current graph-based implementation, we instead use the number of overlapping cells between any given neighborhood and all others as the measure of neighborhood connectivity

for that neighborhood. This eliminates the need to compute any distances, thus greatly reducing runtime.

(D) Building a NB-GLMM

To test for differential abundance, we fit a NB-GLMM to the counts for each neighborhood. The following section describes how the NB-GLMM solver was developed.

The negative binomial likelihood function $L(\theta;\ y)$ for $\theta = [p, \phi]$ is

$$L(p, \phi;\ y) = \binom{y + \phi - 1}{y}(1-p)^{\phi}(p)^{y} \tag{1}$$

We want to find the parameterisation of this function in exponential family form so that we can frame it as a GLMM. An exponential family distribution has the form:

$$f(y \mid \theta) = exp\left[\frac{t(y)\theta - b(\theta)}{a(\gamma)} + c(y, \gamma)\right] \tag{2}$$

where $t(y)$ is the sufficient statistic, θ is the canonical parameter, $b(\theta)$ is the cumulant generating function in the canonical parameter form, c(y, γ) is a rescaled non-negative function of y, and $a(\gamma)$ is a scale parameter for the exponential family distribution [33]. For the negative binomial distribution we can define the likelihood function $L(p, \phi;\ y)$ in exponential family form if we treat ϕ, the negative binomial dispersion parameter, as a known value:

$$L(p, \phi;\ y) = exp\Bigg[\underbrace{ylog[p]}_{t(y)\theta} + \overbrace{\phi log[1-p]}^{b(\theta)} + \underbrace{log[\frac{\Gamma(y)}{\Gamma(y)}]}_{c(y,\gamma)}\Bigg] \tag{3}$$

Therefore, for a negative binomial distribution, $t(y) = y$, $\theta = log(p)$, $a(\gamma) = 1$ and $b(\theta) = -\phi log(1-p)$. The essential idea of maximum likelihood estimation is to determine the model parameter values that maximise this likelihood.

There are several components that we need to construct a GLMM, namely:

- A linear predictor: $\eta = X\beta + Zb$
- A sampling distribution for the observations conditional on the random effects: $E(y|b) = \mu|b$ and $Var(y|b) = V^{1/2}AV^{1/2}$, where $V^{1/2} = diag[\sqrt{\frac{\partial^2 b(\theta)}{\partial \theta^2}}]$, $A = diag[\frac{1}{a(\gamma)}]$ and $y|b$ has a distribution that belongs to the exponential family
- A sampling distribution for the random effects: $b \sim N(0, G)$, where G is the variance-covariance matrix of the random effects, $\begin{bmatrix}\sigma_0^2 & \sigma_{01} \\ \sigma_{01} & \sigma_1^2\end{bmatrix}$.

- A link function to connect the mean value to the linear predictor: $\eta = g(\mu|b)$, or alternatively, an inverse link function: $h(\eta) = X\beta + Zb$

To determine the value of $V^{1/2}$, we need to find the variance, $\frac{\partial^2 b(\theta)}{\partial \theta^2}$, of the negative binomial distribution expressed in terms of μ (see Appendix B for derivation).

Estimation

Once all the components have been derived, we can begin to build our NB-GLMM. In our case we have:

- A linear predictor: $\eta = X\beta + Zb$
- A sampling distribution: $V^{1/2} = diag(\sqrt{\frac{\mu^2}{\phi} + \mu})$ and $a(\gamma) = 1$, therefore $Var(y|b) = V = diag(\frac{\mu^2}{\phi} + \mu)$ and $y|b$ has a negative binomial distribution with fixed ϕ.
- A sampling distribution for the random effects: $b \sim N(0, G)$. As we are only interested in estimating the variance of the intercepts (random intercepts model), $G = \left[\sigma_0^2\right]$, where σ_0^2, simplified to σ^2, is the variance of the random intercepts.
- The log link function for a negative binomial distribution: $\eta = log(\mu|b)$; and an inverse link function: $e^{\eta} = X\beta + Zb$

Generally, the quasi likelihood of the observations conditional on the random effects for a GLMM is $ql(y|b)$ and the likelihood of the random effects is l(b). Therefore, the joint quasi-likelihood is $l(b) + ql(y|b)$ and the marginal quasi-likelihood is the integral:

$$ql(y) = \int_b \int \Big[ql(y|b) + l(b)\Big] db$$

which is often computationally infeasible to solve directly, as one must integrate over all levels of each random effect, b [21]. We therefore need to use a form of approximation. Here, we use the pseudo-likelihood method, described by Wolfinger and O'Connell [22]. This method transforms the data into a pseudo-variable, y*, which is approximately Gaussian, thus allowing us to use well-established mixed model equations. The pseudo-likelihood method applied to our negative binomial model is outlined below.

We begin by defining the first order derivative of the inverse link function $h(\eta)$ evaluated at $\tilde{\eta}$ as:

$$\frac{\partial h(\tilde{\eta})}{\partial \tilde{\eta}} = h'(\tilde{\eta})$$

$$\tilde{D} = diag[h'(\tilde{\eta})] = diag[e^{\tilde{\eta}}]$$

The first order Taylor series expansion of the inverse link function is:

$$h(n) \approx h(\tilde{\eta}) + \tilde{D}(\eta - \tilde{\eta})$$

Rearranging terms yields:

$$X\beta + Zb \approx X\tilde{\beta} + Z\tilde{b} + \tilde{D}^{-1}(h(\eta) - h(\tilde{\eta}))$$

The GLMM pseudo-variable can now be defined as:

$$y^* = \tilde{\eta} + \tilde{D}^{-1}[y - h(\tilde{\eta})] \tag{4}$$

where $\tilde{\eta} = X\tilde{\beta} + Z\tilde{b}$ and the tilde refers to the current estimate of the parameter. It has been shown that:

$$E(y^*|b) = X\tilde{\beta} + Z\tilde{b} + \tilde{D}^{-1}[h(\eta) - h(\tilde{\eta})]$$

$$Var(y^*|b) = \tilde{D}^{-1}V\tilde{D}^{-1} \tag{5}$$

With this approach, we assume that $y^*|b$ has a normal distribution. We can now look at y* as the response variable, and use the linear mixed model framework to estimate the parameters of our NB-GLMM. The NB-GLMM pseudo-likelihood estimating equations are as follows:

$$\begin{bmatrix} X^TW^{-1}X & X^TW^{-1}Z \\ Z^TW^{-1}X & Z^TW^{-1}Z + G^{-1} \end{bmatrix} \begin{bmatrix} \beta \\ b \end{bmatrix} = \begin{bmatrix} X^TW^{-1}y^* \\ Z^TW^{-1}y^* \end{bmatrix} \tag{6}$$

where $W = D^{-1}VD^{-1}$. Importantly, the pseudo-likelihood method allows us to define the marginal pseudo-variance as:

$$Var(y^*) = V^*(\sigma) = ZGZ^T + D^{-1}VD^{-1}$$

The pseudo-log-likelihood function is therefore:

$$pl = -\frac{n}{2}log(2\pi) - \frac{1}{2}log(|V^*(\sigma)|) - \frac{1}{2}(y^* - X\beta)^T[V^*(\sigma)]^{-1}(y^* - X\beta)$$

with y^* replacing y and $V^*(\sigma)$ replacing $V(\sigma)$ in the classic LMM log-likelihood. The restricted pseudo-log-likelihood function is:

$$pl_R = -\frac{n-p}{2}log(2\pi) - \frac{1}{2}log(|V^*(\sigma)|) - \frac{1}{2}log(|X^T[V^*(\sigma)]^{-1}X|) - \frac{1}{2}(r^*)^T[V^*(\sigma)]^{-1}r^*$$

It follows that the ML and REML score vectors $[s_i(\sigma)]$ and information matrix $[I_{ij}(\sigma)]$ terms for LMMs can be used for GLMM covariance component estimation, by replacing y with y^* and $V(\sigma)$ with $V^*(\sigma)$. REML maximizes the likelihood after accounting for the model's fixed effects, thus removing fixed

effects from the estimation of σ^2 [21]. For ML, the score and information matrices are:

$$s_i(\sigma) = -(\frac{1}{2})tr\left[V^*(\sigma)\left(\frac{\partial V^*(\sigma)}{\partial \sigma_i}\right)\right] + \frac{1}{2}(y^* - X\beta)^T V^*(\sigma)\left(\frac{\partial V^*(\sigma)}{\partial \sigma_i}\right)V^*(\sigma)(y - X\beta)$$

$$I_{ij}(\sigma) = (\frac{1}{2})tr\left[V^*(\sigma)^{-1}\left(\frac{\partial V^*(\sigma)}{\partial \sigma_i}\right)V^*(\sigma)^{-1}\left(\frac{\partial V^*(\sigma)}{\partial \sigma_j}\right)\right] \tag{7}$$

where in our case σ_i and σ_j refer to σ^2 and the partial derivatives are:

$$V^*(\sigma) = ZGZ^T + D^{-1}VD^{-1}$$

$$\frac{\partial V^*(\sigma)}{\partial \sigma} = ZZ^T$$

The score and information matrices are implementing in the Fisher scoring algorithm to iteratively update our variance-covariance estimate:

$$\sigma^2 \approx \tilde{\sigma}^2 + [I(\tilde{\sigma})]^{-1}s(\tilde{\sigma}) \tag{8}$$

where I is the information matrix defined above. $\tilde{\sigma}^2$ denotes the value of the variance component from the previous iteration. For REML, the following score vector and information matrices should be used:

$$s_i(\sigma) = -(\frac{1}{2})tr\left[P\left(\frac{\partial V^*(\sigma)}{\partial \sigma_i}\right)\right] + \frac{1}{2}(y^* - X\beta)^T V^*(\sigma)\left(\frac{\partial V^*(\sigma)}{\partial \sigma_i}\right)V^*(\sigma)(y - X\beta)$$

$$I_{ij}(\sigma) = (\frac{1}{2})tr\left[P\left(\frac{\partial V^*(\sigma)}{\partial \sigma_i}\right)P\left(\frac{\partial V^*(\sigma)}{\partial \sigma_j}\right)\right] \tag{9}$$

where $P = [V^*(\sigma)]^{-1} - [V^*(\sigma)]^{-1}X(X^T[V^*(\sigma)]^{-1}X)^{-1}X^T[V^*(\sigma)]^{-1}$.

Estimation for the GLMM proceeds iteratively: the process starts with initial estimates of β, b and σ^2. These are plugged into the GLMM estimating equations (Equation 6) to compute updated solutions for β and b. Next, the current estimates of β and b are used to calculate values for the score and information matrix, and to update σ^2 with the Fisher scoring algorithm. Estimation continues until convergence (default convergence threshold: 1e-5).

Fixed effect inference

Standard errors are calculated by taking the square root of the variance-covariance matrix of the fixed effect estimates. The t-value is calculated with the following equation:

$$t = \frac{\hat{\psi} - \psi_0}{SE(\psi)} \sim t_v \tag{10}$$

where $\hat{\psi}$ is the fixed effect estimate, ψ_0 is its hypothesized value under the null, $SE(\psi)$ is the standard error of the fixed effect estimate and v denotes the degrees of freedom, which are calculated using the Satterthwaite approximation:

$$v \sim 2\left[\frac{SE^2}{g^T V_A g}\right] \tag{11}$$

where $g = \frac{\partial SE}{\partial \sigma}$, $V_{A,ij} = 2 \times tr\left[P\left(\frac{\partial V(\sigma)}{\partial \sigma_i}\right)P\left(\frac{\partial V(\sigma)}{\partial \sigma_j}\right)\right]^{-1}$ and P is defined as in Equation 9. Finally, the P-value is calculated using a t-test with v degrees of freedom.

Benchmarking graph-based functionalities of Milo+

To evaluate the graph-based version of Milo+, we performed a series of benchmarks against the original Milo implementation.

For benchmarking with simulations (Supplementary Fig. 1b), we load the simulated trajectory data from the MiloR package and follow the steps described in the basic Milo vignette [16]. We extract the P-values obtained when running `testNhoods` with the two spatial FDR weighting schemes, k-distance and graph-overlap.

For benchmarking using real single-cell datasets (Fig. 1b, Supplementary Fig. 1a), we download the mouse gastrulation atlas raw count data from Pijuan-Sala et al. via the R package MouseGastrulationData [34]. We subset 4 samples each at developmental stages E7 and E7.5 (14679 cells). The dataset is already pre-processed, so we use the batch-corrected PCA matrix for Milo and Milo+ K-NN graph construction. For the ground-truth dataset (Fig. 1c,d) we use a different subset of samples from the mouse gastrulation atlas: developmental stages E7.75-E8.5 (64,018 cells). To assign cells to two experimental conditions we implement a previously developed algorithm that generates condition probabilities (P(C1)) and assigns simulated condition labels correspondingly [16].

We benchmark the performance of reduced dimension-based Milo against graph-based Milo+ by setting the respective parameters in the functions `makeNhoods` and `testNhoods`. We evaluate the performance of both methods by quantifying the TPR and FDR, as in the original publication [16].

Scalability analysis

We assess the scalability of Milo+ by performing a series of comparisons with Milo on a range of dataset sizes (Fig. 2). We use a simulated single-cell dataset of 194,000 cells to measure the time and memory taken to execute the two complete Milo workflows, from graph building to differential abundance testing.

The simulated dataset is generated with the dyntoy package in R. We downsample the dataset at specific proportions (100%, 75%, 50% and 25%) and measure the elapsed time and maximum memory consumption with the base R functions `sys.time` and `gc`, respectively. Timings are reported in minutes, whilst memory consumption is reported in megabytes (MB). Additionally, we subset the dataset to 50,000 cells and measure the elapsed time over five values of k (10, 30, 50, 70 and 90) to investigate the effect of increasing k.

To characterise Milo+, we also run the graph-based workflow on a million (982,538) cell dataset from cellxgene [35] over three values of k (10, 30, 50). The cellxgene dataset contains a series of healthy and diseased human lung 10x scRNA-seq datasets that have been integrated by mapping to Azimuth references and jointly annotated. The dataset is already pre-proccessed and the pca.corrected slot is used for K-NN graph construction.

All analyses were run on a single node of the European Bioinformatics (EBI) high-performance computing cluster. All jobs were run with one CPU core and 50GB requested memory.

Benchmarking Milo+ NB-GLMM

To define a ground truth dataset for the NB-GLMM benchmarking, we developed a simulation framework that allows us to define parameter values for the fixed effects, β, the variance of the random effects, σ^2, the number of random effect levels, l, the negative binomial dispersion parameter, ϕ, and the sample size, N.

The simulation creates an X matrix by randomly sampling from a binary vector, for categorical variables, or from a normal distribution for continuous variables. Similarly, a Z matrix is created by randomly sampling with replacement from a range of user-defined integers. The random effect values, b, are sampled from a normal distribution with mean 0 and variance σ^2. The dependent variable, y, is then calculated using the following formula:

$$y = exp(X\beta + Zb) + e_i \tag{12}$$

where e_i is random error sampled from the distribution $N(0, 0.001)$. y is then used as a mean value parameter for the `rnbinom` function from the stats R package in order to simulate negative binomial count data.

To simulate neighborhood count data, we followed the same approach as above, but allowed the βs, σ^2, and dispersion, ϕ, to vary between neighborhoods by sampling from a uniform distribution of specified values (see figure legends for details). Additionally, the number of nhoods (200) and the sample size (500) was set. To compare the original GLM implementation to the new NB-GLMM, we model Z as a fixed effect using a NB-GLM ($\sim$ X + Z) and as a random

effect using a NB-GLMM ($\sim$ X + (1 | Z)).

We benchmark the performance of the `runGLMM` function in Milo+ against the `glmmTMB` (glmmTMB), `glmm-PQL` (MASS) and `glmer` (lme4) R functions (the respective packages are given in brackets). All methods were run with the same parameters, namely: a negative binomial response variable, log link function, REML = TRUE, and one fixed effect and one random effect. Additionally, to evaluate the Satterthwaite approximation for the degrees of freedom we used lmer [24], as this method is not implemented in glmmTMB. Since lmer only runs a LMM model, we used the y^* pseudo-variable from our model as an input for lmer.

Code Availability

Milo+ is available as an open-source R package at https://github.com/MarioniLab/miloR (devel branch). The code used to perform all analyses and generate the figures can be found at https://github.com/MarioniLab/milo_analysis_2022.

References

[1] Tang, F., et al.: mrna-seq whole-transcriptome analysis of a single cell. Nat Methods **6**, 377–382 (2009). 10.1038/nmeth.1315

[2] Lähnemann, D., et al.: Eleven grand challenges in single-cell data science. Genome Biol **21**(31) (2020). 10.1186/s13059-020-1926-6

[3] Linnarsson, S., Teichmann, S.A.: Single-cell genomics: coming of age. Genome Biol **17**(97) (2016). 10.1186/s13059-016-0960-x

[4] Application note: Transcriptional profiling of 1.3 million brain cells with the chromium single cell gene expression solution. 10x Genomics (2020). https://pages.10xgenomics.com/rs/446-PBO-704/images/10x_LIT015_Chromium_Million-Brain-Cells_Application-Note_Letter_digital.pdf

[5] Shaum, N., et al: Single-cell transcriptomic characterization of 20 organs and tissues from individual mice creates a tabula muris. Nature **562**(7727), 367–372 (2018). 10.1038/s41586-018-0590-4

[6] A., R., et al.: The human cell atlas. elife **6**(e27041) (2017). 10.7554/eLife.27041

[7] Davie, K., et al.: A single-cell transcriptome atlas of the aging drosophila brain. Cell **174**(4), 982–998 (2018). 10.1016/j.cell.2018.05.057

[8] Stephenson, E., et al.: Single-cell multi-omics analysis of the immune response in covid-19. Nature medicine **27**(5), 904–916 (2021). 10.1038/s41591-021-01329-2

[9] Zhao, J., et al.: Detection of differentially abundant cell subpopulations in scrna-seq data. Proc. Natl Acad. Sci. **118**(22) (2021). 10.1073/pnas.2100293118

[10] Kiselev, V., et al.: Sc3: consensus clustering of single-cell rna-seq data. Nature Methods **14**, 483–486 (2017). 10.1038/nmeth.4236

[11] Ramachandran, P., et al.: Resolving the fibrotic niche of human liver cirrhosis at single-cell level. Nature **575**, 512–518 (2019). 10.1038/s41586-019-1631-3

[12] Liao, M., et al.: Single-cell landscape of bronchoalveolar immune cells in patients with covid-19. Nature medicine **26**, 842–844 (2020). 10.1038/s41591-020-0901-9

[13] Pijuan-Sala, B., et al.: A single-cell molecular map of mouse gastrulation and early organogenesis. Nature **566**, 490–495 (2019). 10.1038/s41586-019-0933-9

[14] Kiselev, V.Y., Andrews, T.S., Hemberg, M.: Challenges in unsupervised clustering of single-cell rna-seq data. Nat Rev Genet **20**, 273–282 (2019). 10.1038/s41576-018-0088-9

[15] Burkhardt, D.B., et al.: Quantifying the effect of experimental perturbations at single-cell resolution. Nat Biotechnol **39**, 619–629 (2021). 10.1038/s41587-020-00803-5

[16] Dann, E., et al.: Differential abundance testing on single-cell data using k-nearest neighbor graphs. Nat Biotechnol **40**, 245–253 (2020). 10.1038/s41587-021-01033-z

[17] Harrison, X.J., et al.: A brief introduction to mixed effects modelling and multi-model inference in ecology. PeerJ **6**(e4794) (2018). 10.7717/peerj.4794

[18] Xie, L., Madden, L.V.: Hpglimmix: A high-performance sas macro for glmm estimation. Journal of Statistical Software **58**(8), 1–25 (2014). 10.18637/jss.v058.i08

[19] Bolker, B., et al.: Generalized linear mixed models: A practical guide for ecology and evolution. Trends in Ecology and Evolution **24**, 127–35 (2009). 10.1016/j.tree.2008.10.008

[20] Lagraa, S., Seba, H.: An efficient exact algorithm for triangle listing in large graphs. Data Mining and Knowledge Discovery, Springer **30**(5) (2016). 10.1007/s10618-016-0451-4

[21] Stroup, W.W.: Generalized Linear Mixed Models: Modern Concepts, Methods and Applications. CRC Press, Taylor I& Francis Group, Boca Raton (2013)

[22] Wolfinger, R., O' connell, M.: Generalized linear mixed models a pseudo-likelihood approach. Journal of Statistical Computation and Simulation **48**(3-4), 233–243 (1993). 10.1080/00949659308811554

[23] Brooks, M.E., et al.: glmmtmb balances speed and flexibility among packages for zero-inflated generalized linear mixed modeling. The R Journal **9**(2), 378–400 (2017). 10.1038/s41587-020-00803-5

[24] Bates, D., et al.: Fitting linear mixed-effects models using lme4. Journal of Statistical Software **67**(1), 1–48 (2015). 10.18637/jss.v067.i01

[25] Venables, W.N., Ripley, B.D.: Modern Applied Statistics with S, 4th edn. Springer, New York (2002)

[26] Luke, S.G.: Evaluating significance in linear mixed-effects models in r. Behav

Res **49**, 1494–1502 (2017). 10.3758/s13428-016-0809-y

[27] Satterthwaite, F.E.: Synthesis of variance. Psychometrika **6**(5), 309–316 (1941)

[28] Kenward, M.G., Roger, J.H.: Small sample inference for fixed effects from restricted maximum likelihood. Biometrics, 983–997 (1997)

[29] Hao, Y., et al.: Integrated analysis of multimodal single-cell data. Cell **184**(13), 3573–3587 (2021). 10.1016/j.cell.2021.04.048

[30] Polański, K., et al.: Bbknn: fast batch alignment of single cell transcriptomes. Bioinformatics **36**(3), 964–965 (2020). 10.1093/bioinformatics/btz625

[31] Lippert, C., et al.: Fast linear mixed models for genome-wide association studies. Nat Methods **8**, 833–835 (2011). 10.1038/nmeth.1681

[32] Robinson, M.D., McCarthy, D.J., Smyth, G.K.: edger: a bioconductor package for differential expression analysis of digital gene expression data. Data Mining and Knowledge Discovery, Springer **26**(1), 139–140 (2010). 10.1093/bioinformatics/btp616

[33] Nelder, J.A., Wedderburn, R.W.M.: Generalized linear models. Journal of the Royal Statistical Society. **135**(3), 370–384 (1972). 10.2307/2344614

[34] Pijuan-Sala, B., et al.: A single-cell molecular map of mouse gastrulation and early organogenesis. Nature **566**, 490–495 (2019). 10.1038/s41586-019-0933-9

[35] Satija, R.: Azimuth meta-analysis of human scrna-seq datasets. cellxgene (2022). https://cellxgene.cziscience.com/collections/2f75d249-1bec-459b-bf2b-b86221097ced

Appendix A

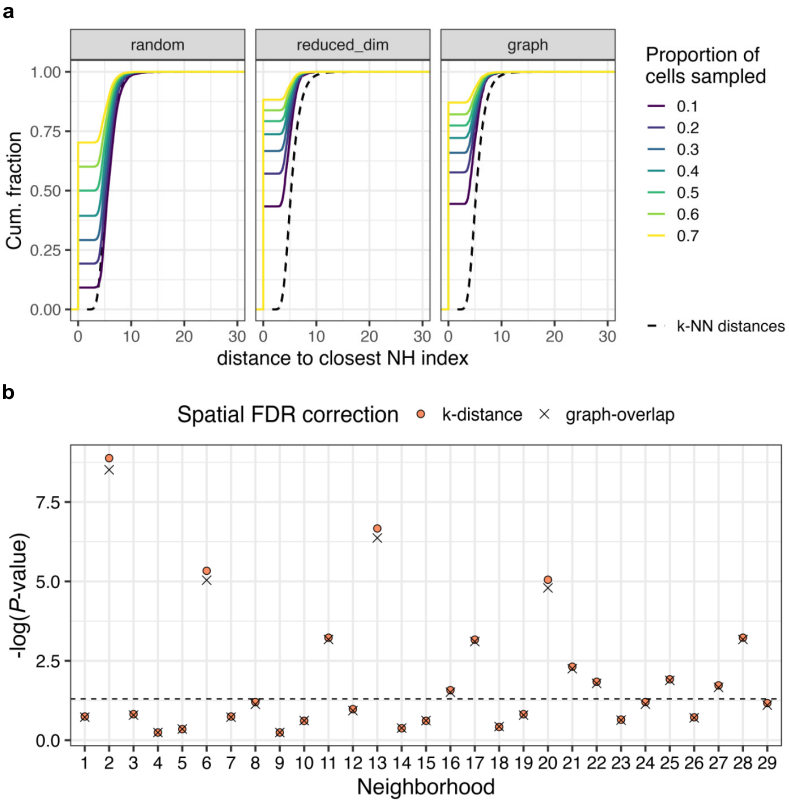

Fig. 1 supplementary: Graph-based Milo+ performs very similarly to the original reduced dimension-based implementation. **a**, Sampling with either the reduced-dimension-based (middle panel) or the graph-based refinement (right panel) generates robust neighbourhoods across initializations. For each index cell we calculate the distance in PC space to the closest index in a sampling with different initialization (across 3 different initializations). The cumulative distribution (y-axis) of distances to the closest index is shown. NH: neighbourhood; Cum: cumulative. **b**, P-values plotted on a -log10 scale (y-axis) for 29 neighborhoods (x-axis) produced by the original reduced-dimension Milo workflow and the graph-based workflow. Dashed line shows threshold at P-value = 0.05. The analysis was conduced on the simulated trajectory datasets available in the MiloR package.

Appendix B

The mean and variance of the negative binomial distribution can be calculated using the following properties of the exponential family: $\frac{db(\theta)}{d\theta} = E[Y] = \mu$ and $Var[Y] = \gamma\frac{d^2b(\theta)}{d\theta^2}$.

Using the previously determined expression for $b(\theta)$ (see Methods Section D) and setting $e^\theta = p$,

$$b(\theta) = -\phi log[1 - e^\theta]$$

Let $u = 1 - e^\theta$ and $v = \phi logu$. Using the chain rule:

$$E[Y] = \mu = \frac{db(\theta)}{d\theta} = (\frac{du}{d\theta})(\frac{dv}{du}) = [-e^\theta \cdot \frac{-\phi}{1 - e^\theta}] = \frac{\phi e^\theta}{1 - e^\theta}$$

Substituting $e^\theta = p$:

$$E[Y] = \mu = \frac{db(\theta)}{d\theta} = \frac{\phi p}{1 - p} \tag{1}$$

Now that we know the mean, we can find the variance as follows:

$$Var[Y] = \gamma\frac{d^2b(\theta)}{d\theta^2} = \gamma\frac{d(db(\theta)/d\theta)}{d\theta} = \gamma\frac{d\mu}{d\theta} = \gamma\frac{d\frac{\phi p}{1-p}}{\theta} = \gamma\frac{d\frac{\phi e^\theta}{1-e^\theta}}{d\theta}$$

Using the quotient rule:

$$\frac{df}{dt} = \frac{v\frac{du}{dt} - u\frac{dv}{dt}}{v^2}$$

Let $u = \phi e^\theta$ and $v = 1 - e^\theta$, with $\frac{du}{d\theta} = \phi e^\theta$ and $\frac{dv}{d\theta} = -e^\theta$:

$$\begin{aligned} Var[Y] = \gamma\frac{d\frac{\phi e^\theta}{1-e^\theta}}{d\theta} &= \gamma\frac{(1 - e^\theta) \cdot \phi e^\theta - (\phi e^\theta \cdot -e^\theta)}{(1 - e^\theta)^2} \\ &= \gamma\frac{\phi e^\theta - \phi(e^\theta)^2 + \phi(e^\theta)^2}{(1 - e^\theta)^2} = \gamma\frac{\phi e^\theta}{(1 - e^\theta)^2} \end{aligned}$$

Substituting $e^\theta = p$:

$$Var[Y] = \gamma\frac{d\frac{\phi e^\theta}{1-e^\theta}}{d\theta} = \gamma\frac{\phi p}{(1 - p)^2} \tag{2}$$

where, as determined previously, $a(\gamma) = 1$. Finally, to build the NB-GLM we require the variance term expressed in terms of μ. First, rearrange to get an expression for p in terms of μ:

$$\mu = \frac{\phi p}{1-p}$$
$$p = \frac{\mu}{\mu+\phi}$$

Then, plug $p = \frac{\mu}{\mu+\phi}$ back in to the equation for the variance and rearrange:

$$Var[Y] = \frac{\phi(\frac{\mu}{\mu+\phi})}{(1-\frac{\mu}{\mu+\phi})^2} = \frac{\frac{\phi\mu}{\mu+\phi}}{\frac{\phi^2}{(\mu+\phi)^2}} = \frac{\phi\mu}{\mu+\phi}\frac{(\mu+\phi)^2}{\phi^2} = \frac{\mu^2+\phi\mu}{\phi} = \frac{\mu^2}{\phi} + \mu \quad (3)$$