A review of pseudotime reconstruction algorithms

Guilherme de Sena Brandine

September 23, 2016

1 Motivation

Single cell resolution RNA sequencing is a technique with high potential to aid the understanding of biological processes which involve the continuous differentiation of cells towards one or multiple end states. It allows the profiling of the entire transcriptome of single cells, which, upon using the correct analysis techniques, can lead to the *de novo* discovery of new biological pathways and which genes play a role in such pathways. Examples of such processes are commonly found in stem cell differentiation, such as the formation of lymphocytes from hematopoietic stem cells in the bone marrow, differentiation of T cells from early lymphocytes through thymus hormone stimulations, early skeletal mycocyte formation from myoblasts, cell differentiations in early embryos, neurogenesis and so on.

From now on we assume there are N cells and D genes and the set $V = \{x_1 \dots x_N\} \subset \mathbb{R}^D$ represents the set of cells as points in the D-dimensional space. The output of every algorithm is a set $T = \{t_1 \dots t_N\}$, with $0 \leq t_i \leq 1 \; \forall \; 1 \leq i \leq n$ which assigns a *pseudotime* to each of the cells indicating their progression in a biological development. Optionally, the algorithm can also output a tree hierarchy that shows how cells have branched from each other during development.

A subsequent question is to find the differentially expressed genes in each pathway, but this is a somewhat simpler problem as it can be solved by modelling the gene's temporal expression as a generalized additive model and attempting to reject the null hypothesis that the expression remains constant throughout time. Hidden Markov Models can also be used for this purpose, so we won't be focusing on this problem here, but rather on the pseudotime problem.

In general, a good pseudotime ordering algorithm must have the following properties:

1) It correctly identifies the up/down regulation of known marker genes in the biological problem at hand

2) It is robust to the intrinsic technical and biological noise that is characteristic of scRNA-Seq data.

3) It finds a smooth curve with few "turns" (mathematically a small total integral of the absolute value of the curve's second derivative) which mathematically describe the underlying pathway and thus can be subsequently used to predict the pseudotime of new cells.

Most algorithms described below do a decent job in criteria (1) (for the datasets they work on) and some of them also in (2). However, all of them rely on graph theory, which generates piecewise linear curves that are further used as a backbone to project the data. These curves are conceptually flawed in two senses: First, they assume the biological trajectory is piecewise linear with sharp changes of direction (given by the edge intersections), which, from the biological perspective of strongly regulated differentiation steps, is highly unlikely. Second, they do not attempt to minimize the total squared error between the data points and the projected values, which does not guarantee the optimality of the resulting curve. Furthermore, most algorithms (except for DPT) work on a reduced dimension space to decrease runtime and improve robustness, which potentially yields loss of information of marker genes.

2 Algorithms

2.1 Monocle

Paper title: *The dynamics and regulators of cell fate decisions are revealed by pseudotemporal ordering of single cells* (data: GSE52529)

Biological background: Skeletal Myoblast (embryonic cells that form tendons) were sequenced on a Fluidigm microfluidics system. They go from Myoblast → Intermediate Myocyte → Mature Skeletal Myocyte. A small number of TFs regulate this process. Cells were captured with microfluidigm system. 4 million reads per library(ie, per cell). Cells were first grown in high-mitogen condition and induced differentiation by switching to low serum. Sequencing was done in single cells in different time points (0h, 24h, 48h and 72h), and all of them showed similar heterogeneity, hypothetically due to cells being in different stages of differentiation.

Algorithm: Monocle models the observations as $V = \{\rho(s_0) \dots \rho(s_n)\}$ where $\rho : \mathbb{R} \to \mathbb{R}^D, \rho(s) = f(s) + \delta(s)$ is the sum of a smooth function f that truly predicts the timeline and a noise function δ. Monocle doesn't actually work in D dimensions. Instead, it uses Independent Component Analysis (ICA) to reduce the cells to two dimensions, whose aim is to reduce the influence of genes that contribute, at most, only to noise in the cell.

Once in a lower dimension, it builds the complete graph (ie, all pairs of cells are connected by a node), where each edge is weighted by a distance function $d(x_i, x_j) = 1 - \frac{\rho_{x_i,x_j}}{2}$, where $\rho_{x,y}$ is the pearson correlation between cell x and cell y. Monocle then finds the minimum spanning tree of this graph and the longest path in it (recall you can easily find the longest path in a tree by doing a DFS from an arbitrary node, finding the farthest node in the path and doing a DFS starting from this node).

To find branching points, Monocle uses a PQ tree to keep track of possible new pathways. For this purpose, it finds the **indecisive backbone** of the longest path, that is, the longest continuous subset of vertices of the path for which both endpoints have degree 2 (that is, no branching). It adds the vertices of degree 2 as children of the Q node and the ones with degree greater than 2 as P nodes attached to the current Q node, then recursing on the new Q vertices created.

How pseudotime is calculated: Once the branching tree is obtained, Monocle defines the pseudotime of any node as the pseudotime of its parent + the edge weight between the parent and itself. Naturally, the pseudotime of the root is equal to 0.

2.2 Wanderlust

Paper title: *Single-Cell Trajectory Detection Uncovers Progression and Regulatory Coordination in Human B Cell Development*

Biological background: In order for B cells (lymphocytes, a type of white blood cell) to develop, they start as hematopoietic stem cells and go through the patwhay of lymphoid progenitor cell $\rightarrow$ pro-B cell $\rightarrow$ pre-B cell and finally an immature B cell. The paper aims to understand the timing of key events of this differentiation process, in particular the Immunoglobulin Heavy Chain (IgH) rearrangement, which is the mechanism of genetic recombination that diversifies the immunoglobulins and T cell receptors and creates the large set of antibodies in the body.

Algorithm: The method starts by building a k-nearest neighbors graph, where similarity is given by cosine distance, that is:

$$d(x_i, x_j) = 1 - \frac{x_i \cdot x_j}{||x_i|| \cdot ||x_j||}$$

Wanderlust then calculates a *shortest path distance* between pairs of cells in this k-NN graph. Based on a user-defined starting cell, subsequent cells are sorted by the shortest-path distance to this starting cell. To overcome short circuits (eg, cells that have small distance due to noise rather than for being developmentally close), Wanderlust creates an *l-k-NNG*, which is a k-NNG in which one iterates through the nodes and randomly keeps l out of its k nearest neighbors. Short circuits will only exist in 2l/k of these graphs, so upon taking the mean of several l-k-NNGs, short circuits can be overcome.

To increase robustness on the choice of initiator cell, Wanderlust also chooses random "waypoint" cells uniformly chosen along the graph and refines each cell's position based on the shortest-path distance from each waypoint. Distances are weighed so that waypoints closer to the target contribute more to the calculation.

How pseudotime is calculated: Wanderlust outputs trajectory scores as an average of all trajectory scores in the l-k-NNGs. Scores are calculated through an optimization process by recalculating distances from cells to waypoints until convergence

2.3 Wishbone

Paper title: *Wishbone identifies bifurcating developmental trajectories from single-cell data* (Data: GSE72857)

Biological background: Lymphoid progenitor cells are stimulated by thymus hormones (thymosins) to differentiate into T cells, which can be divided in either CD8+ cytotoxic or CD4+ helper T cells. The expressions of known genes such as CD44 and CD25 fluctuate along this process. Wishbone was developed to *de novo* identify where the lymphoid cells branch and find new

differentially expressed genes in the trajectory.

Algorithm: Wishbone first uses diffusion maps for dimensionality reduction. This space is used to construct a kNNG using Euclidean Distance in the embedded space. A user-defined initial cell s is used to calculate the shortest path to all other cells using Dijkstra. The trajectory $\tau_i^{(0)}$ is initialized as the ordering of cells based on the shortest path to s (τ_i is the pseudotime for cell i). Shortest paths are less reliable as cells go farther due to accumulation of noise, so like Wanderlust, Wishbone uses waypoint cells, except that instead of using a random cell as waypoint, it uses the median of its k nearest neighbors (median filter) of randomly chosen cells to overcome the fact that the cell can be a potential outlier. It then calculates the shortest paths from each waypoint to all cells to build a matrix $D \in \mathbb{R}^{nW \times N}$ where N is the number of cells and nW is the user-define number of waypoints, after which it constructs the perspective of each cell i with reference to the waypoint w using the initially guessed trajectory as reference. Formally, the perspective P_{wi} is given by:

$$P_{wi} = \begin{cases} \tau_w^{(0)} + D_{wi} \text{ if } \tau_i^{(0)} > \tau_w^{(0)} \\ \tau_w^{(0)} - D_{wi} \text{ otherwise} \end{cases}$$

Which puts all the waypoint orderings in a common ground. The new trajectory pseudotimes are now given by a weighted average of the above perspectives weighted by the distance from each cell to the waypoint to account for possibly distal waypoint cells. Formally, we have:

$$W_{wi} = \exp\left(\frac{-D_{wi}^2}{\sigma}\right) / \sum_{k=1}^{N} \exp\left(\frac{-D_{wk}^2}{\sigma}\right)$$

$$\tau_i^{(1)} = \sum_{w \in \text{waypoint set}} P_{wi} W_{wi}$$

Where σ is the standard deviation of the distance matrix. To identify branches, inconsistencies between waypoint trajectories are used. If i and t lie in the same trajectory, then a path from s to t will approximately be correlated to a path from s to i + a path from i to t. Correlation will be smaller if paths are different. Hence Wishbone uses the aforementioned perspectives to compute a *disagreement* matrix $Q \in \mathbb{R}^{nW \times nW}$ between waypoint cells. Namely:

$$Q_{ij} = |P_{ij} - \tau_j^{(0)}|$$

Which is a symmetric distance matrix, whence it can be decomposed by the spectral theorem and the value of the second eigenvector for each waypoint indicates whether a waypoint is in the trunk of the branck ($v_{2w} \approx 0$), or in the left branch ($v_{2w} > 0$) or right branch ($v_{2w} < 0$), which creates a wishbone-like structure when plotting the eigenvector value vs the trajectory τ. Trajectories τ are further calculated once again for each set (trunk+branch 1 and trunk + branch 2) and refined until convergence.

2.4 Waterfall

Paper title: *Single-Cell RNA-Seq with Waterfall Reveals Molecular Cascades underlying Adult Neurogenesis* (Data: GSE71485)

Biological background: Quiescent Neural Stem Cells (qNSCs) continuously generate new neurons by a complex regulation of quiescent/active state, cell cycle regulation and cell fate decision (that is, which type neuron the stem cell will become). The paper aims on elucidating the molecular dynamics during initial phases of mice adult neurogenesis in vivo.

Algorithm: I believe Waterfall is the simplest algorithm among these described here as it aims on reconstructing pseudotime based on very little prior information and it does not account for branching. It starts by doing unsupervised clustering by using pearson correlation as distance metric. It then uses PCA for dimensionality reduction and k-means to group cells into clusters (the way they decide k is not specified in the paper so I'm assuming it's given as a user input). It then connects the cluster centroids through a minimum spanning tree and projects all the points through perpendicular projection to the closest edge. The line is then straightened (also not specified how straightening is done in case the MST contains branches) and the pseudotimes are normalized by the resulting line length, ranging from 0 to 1.

2.5 Diffusion Pseudotime (DPT)

Paper title:*Diffusion pseudotime robustly reconstructs lineage branching* (Data: GSE61470, GSE65525, GSE72857)

Biological background: DPT was applied on single-cell qPCR data focusing on early blood development in mice as well as in the Klein Drop-Seq dataset that studies mouse embryonic stem cell differentiation upon leukemia inhibitory factor (LIF) withdrawal. Hematopoietic stem cells can become either red blood cells or endothelial-like cells (cells that lie in the interior surface of blood vessels).

Algorithm: DPT creates a transition matrix based on the k-nearest-neighbors graph of the cell. However, instead of Euclidean distance, the transition probabilities are given by Gaussian distributions. Assume we fix the number k of nearest neighbors. We define the Kernel of cell j as $\sigma_j = ||x_j - x_j^{(k)}||^2$, where $x_j^{(k)}$ is the k-th nearest neighbor of cell j. The (non-normalized) interference (eg, distance) of two cells is given by the Hellinger distance between their respective gaussian functions:

$$K(x_i, x_j) = \left(\frac{2\sigma_i\sigma_j}{\sigma_i^2 + \sigma_j^2}\right)^{1/2} \exp\left(-\frac{||x_i - x_j||^2}{2(\sigma_i^2 + \sigma_j^2)}\right)$$

Which can be properly normalized by the row sums, forming the transition probability matrix between all pairs of cells x_i and x_j, called T, whose elements T_{ij} are calculated as follows:

$$W(x_i, x_j) = \frac{K(x_i, x_j)}{\sum_{k=1}^N K(x_i, x_k) \sum_{k=1}^N K(x_k, x_j)}$$

$$T_{ij} = \frac{W(x_i, x_j)}{\sum_{k=1}^N W(x_i, x_k)}$$

Note that T is a stochastic matrix.

How pseudotime is calculated: Given the above TPM, we can calculate the probability that any cell x eventually reaches another cell by the following matrix:

$$M = \sum_{i=1}^{\infty} \tilde{T}^i = (I - \tilde{T})^{-1} - I$$

Where $\tilde{T} = T - \psi_0 \psi_0^T$ is the TPM by removing the eigenvector corresponding to the eigenvalue 1 (which by Perron-Frobenius is always the largest eigenvalue of the matrix), which removes the information from the stationary distribution of the Markov Chain.
Assume now we know a root cell x_0. The **diffusion pseudotime** of each subsequent cell x_j is given by:

$$DPT(x_0, x_j) = ||M(x_0, \cdot) - M(x_j, \cdot)||$$

That is, the euclidean distance between the rows corresponding to both cells in the matrix M. The ordering of the cells is then given by sorting the pseudotimes with respect to the root cell. DPT also finds branching by looking at the correlation between the path starting at the root cell and the path starting at the farthest cell from the root. Upon branching both paths go from anticorrelated to correlated.

Side note: Since there's no explicit curve finding in this method, we couldn't use DPT as a training set to infer the position of new cells in the timeline, for example, as we would have to recalculate the stochastic matrix.