Inference of Cellular Developmental Time

transition-speed: default transition: fade autosize: false

"From Multivariate to Longitudinal Data"April 11, 2017

Caleb Lareau bit.ly/LareauBST245

Overview

- Motivation
- Single cell RNA-Seq
- Model Dataset
- EDA
- Methods of estimating developmental time
- PCA
- Probablistic PCA; Bayesian PCA
- Gaussian Process Latent Variable Modeling

Trajectories...

==

~3 million cells ~ 20 billion cells ~ 50 trillion cells

- What are the key points in development for disease?

==

Questions from developmental biology

- What happens in a cell such that it becomes a brain, toe, or a heart? - When do these decisions get made? "Who" makes them? - How do the developmental trajectories of disease (leukemia / schizophrenia) differ from healthy individuals? - Can we identify important transition points and the genetic signature underlying them?

Waddington Landscape

Some cancers regain stemness programs

Stergachis *et al.*, Cell 2013

Stem cell-likeness in AML

Corces *et al.* Nature Genetics, 2016

How do we characterize single cells?

==

Proserpio and Mahata, Immunology 2015

==

==

==

==

==

==

==

==

==

EDA

class: small-code “‘{r, eval = FALSE} > dim(deng)

[1] 17585 255

    sum(deng == 0) / prod(dim(deng))

[1] 0.5019552

    head(sample(colnames(deng)))

[1] “earlyblast” “16cell” “4cell” “midblast” “lateblast” “16cell”

head(sample(rownames(deng)))

[1] “Gm7073” “Mir697” “Uqcrc2” “Ap2m1” “Slc10a3” “Ccr1”

“‘

==

Linearly increasing

Dropoff

Linear Decreasing

Varying, no clear effect

V-shaped

Transition on/off

Sigmoidal with dropout

Overall picture

∀ gene g, fit OLS Regression with known timepoint t per cell–

$$\log_2(g+1) \sim \beta_0 + \beta_1 t$$

Permuted

∀ gene g, fit regression with permuted timepoint t^* per cell

$$\log_2(g+1) \sim \beta_0 + \beta_1 t^*$$

Permuted

$\forall$ gene g, fit regression with permuted **factor** timepoint t^{**}

$$\log_2(g+1) \sim \beta_0 + \beta_1 t^{**}$$

Statement of problem

Given a matrix $\mathbf{Y}$ of D genes (features) by n samples, determine a latent vector $\mathbf{P}$ with dimension 1 x n that reflects the developmental trajectory of the n cells from the variance in D genes. - D can be thought of has a higher dimension space, and we want to infer d ($d < D$) latent variables in the gene data. - One of the d latent variables ideally reflects developmental ordering.

Perfect latent variable

class: small-code

"'{r, eval = FALSE} > cor(runif(length(time)) %>% sort(), as.numeric(time) %>% sort())^2

[1] 0.8799669 "'

PCA

Computing PCA

$\mathbf{Y}$ is a D x n matrix. Compute the covariance matrix–

$$\mathbf{\Sigma} = E(\mathbf{Y}\mathbf{Y}^{\mathbf{T}}) - \mu\mu^{\mathbf{T}}$$

where $\mu = E(\mathbf{Y})$

Then compute the spectral decomposition of $\mathbf{\Sigma}$

$$\mathbf{\Sigma} a_j = \lambda_j a_j$$

for $j \in (1, ..., D)$ Then a_j represent the eigenvectors of the data matrix $\mathbf{Y}$. Note the ordering of j is meaningful–

$$\lambda_1 \geq \lambda_2 \geq ... \geq \lambda_D \geq 0$$

Computing PCA

class: small-code “‘{r, eval = FALSE} > irlba::prcomp_irlba()

prcomp()

princomp()

“‘

PCA

class: small-code

“‘{r, eval = FALSE} > cor(pca$rotation[,1], as.numeric(time))ˆ2

[1] 0.5933009 “‘

Correlation with all PCs

Pause…

PCA by itself isn’t satisfactory… Ideas for improvements?

Improving on PCA

1) Quantifiying uncertainty 2) Non-linear latent variable inference

Uncertainty

From Buenrostro *et al.* bioRxiv 2017

Mannifold / Non-linear dimension learning

- Next several images taken from slides via Guy Wolf (Yale)
  - These slides can be found here

==

==

==

==

Tackling 1 and building to 2

==

==

Slide from Neil Lawrence

Factor Analysis

Y is a D x n matrix. **x** is a d x n matrix ($d < D$). Want to relate **x** and **Y** and assume that the relationship is linear–

$$\mathbf{Y} = \mathbf{W}\mathbf{x} + \epsilon$$

where W is a D x d matrix.

By convention (after locating the matrix),

$$\mathbf{x} \sim \mathcal{N}(0, \mathbf{I})$$

where **I** is the identity matrix of dimension d x d.

Factor Analysis II

Specify the error,

$$\epsilon \sim \mathcal{N}(0, \mathbf{\Psi})$$

where we assume $\mathbf{\Psi}$ to be a diagonal matrix D x D. - Assuming the diagonal structure of $\mathbf{\Psi}$ means that all observertional correlations is due to the latent variables. - In other words, Y_i for $i \in (1, ..., n)$ are conditionally independent given **x**

Factor Analysis III

Then Y_i for $i \in (1, ..., n)$,

$$Y_i | x_i \sim \mathcal{N}(0, \mathbf{\Psi})$$

Then we can integrate out the latent variables–

$$p(Y_i | \mathbf{W}, \mathbf{\Psi}) = \int p(Y_i | x_i, \mathbf{W}, \mathbf{\Psi}) p(x_i) dx_i$$

Under this specification, we can write the MVN distribution for our observations–

$$\mathbf{Y} \sim \mathcal{N}(0, \mathbf{C}), \mathbf{C} = \mathbf{W}\mathbf{W^T} + \mathbf{\Psi}$$

Computing PCA via Factor Analysis

Young-Whittle Factor Analysis (1950s) ψ_i element of the diagonal of Ψ; add constraint $\psi_i = \sigma^2$ Assume σ^2 known, MLE yields same W as PCA

$$\mathcal{L} = \tfrac{-N}{2}\{D \log(2\pi) + \log|\mathbf{C}| + \text{tr}(\mathbf{C}^{-1}\mathbf{\Sigma})\}$$

$$\mathbf{C} = \mathbf{W}\mathbf{W^T} + \mathbf{\Psi}$$

$$\mathbf{x} = \mathbf{C^{-1}}\mathbf{W^T}\mathbf{Y}$$

Probablistic PCA (Tipping and Bishop)

- Assuming σ^2 may not be reasonable; want to estimate it from the data (keep in likelihood)
- Can estimate $\mathbf{W}, \sigma^2$ using EM & with a prior over $\mathbf{x}$

For $\mathbf{W}_{\text{MLE}}$,

$$\sigma^2_{MLE} = \frac{1}{D - d} \sum_{j=d+1}^{D} \lambda_j$$

- Similarly, we can integrate over $\mathbf{W}$ given a prior, yielding

$$\mathbf{x} \sim \mathcal{N}(\mathbf{C^{-1}}\mathbf{W^T}\mathbf{Y}, \sigma^2 C^{-1})$$

Bayesian PCA

Bayesian PCA

- α_i represents the inverse of the variance
- for large α_i, contribution of latent factor i is low.
- Solved by EM / similar algorithm

Bayesian / Probablistic PCA in R

class: small-code "‘{r, eval = FALSE} library(pcaMethods)

pca()

resPPCA <- pca(data, method="ppca", center=FALSE, nPcs=5) resBPCA <- pca(data, method="bpca", center=FALSE, nPcs=5)

"‘

==

Non-linear

$$\mathbf{Y} \sim \mathcal{N}(0, \mathbf{C}), \mathbf{C} = \mathbf{W}\mathbf{W}^{\mathbf{T}} + \mathbf{\Psi}$$
$$\mathbf{C}(\mathbf{W_i}, \mathbf{W_j}) = \mathbf{W_i^T}\mathbf{W_j} + \sigma^2\delta_{ij}$$
$$\mathbf{C}(\mathbf{W_i}, \mathbf{W_j}) = \theta_{rbf} \exp\left(\frac{-\gamma}{2}(\mathbf{W_i} - \mathbf{W_j})^T(\mathbf{W_i} - \mathbf{W_j})\right) + ...$$

- (2) being a special case (linear, iid) of (3)
- Computationally more challenging, but there are fast algorithms out there

GPLVM

GPLVM

class: small-code "‘{r, eval = FALSE} library(pseudogp)

fit <- fitPseudotime(data, smoothing_alpha = 30, smoothing_beta = 6, iter = 1000, chains = 1)

posteriorBoxplot(fit) "‘

GPLVM – Noteable application

GPLVM

class: small-code

“‘{r, eval = FALSE} > cor(gplvm_means, as.numeric(time))ˆ2

[1] 0.783522 “‘

GPLVM versus PCA 1

class: small-code

“‘{r, eval = FALSE}

    cor(pca$rotation[,1], as.numeric(time))ˆ2

[1] 0.5933009

    cor(gplvm_means, as.numeric(time))ˆ2

[1] 0.783522 “‘

GPLVM versus PCA 2

Wrapping up…

- GPLVM provide both a probablistic and non-linear latent variable inference structure - All other published algorithms provide point estimates, difficult to ascertain uncertainty - Under this framework, Bayesian priors are allowed, which have been useful in published studies - Tools and methods useful in many data analyses contexts (including machine learning) - Motivated here by single cell analysis

Detailed look at early embryo neurogenesis?

==

Thanks!