

Lecture 3: Simulation

Tianjun Ke

Renmin University of China

Introduction to simulation, May 2023

Table of Contents

- 1 引入
- 2 背景知识
- 3 simulation 的使用场景
- 4 simulation \rightarrow insight!
- 5 如何做 simulation ?
- 6 总结

Table of Contents

1 引入

2 背景知识

3 simulation 的使用场景

4 simulation \rightarrow insight!

5 如何做 simulation ?

6 总结

什么是 simulation: 输入 + 模型 + 输出

1. Q: 输入? \implies A: 一组数据, 通常由概率分布 (Lecture 1) 产生
2. Q: 模型? \implies A: ML, DL, ... (比如 Lecture 2 中的 kernel ridge regression)
3. Q: 输出? \implies A: 用于评估模型的某些变量 (正确率, 损失值, 概率分布, ...)

simulation 有什么用：近似 + 测试 + 验证

1. 输入太复杂了（图片，文字，...）
2. 模型太复杂了（积分没有解析解）
3. 输出太复杂了（损失函数图像紊乱）

总的来说, simulation 是一种建模和分析复杂系统的强大工具。它使我们能够研究在真实世界中难以或不可能研究的现象,并在受到可控且可复现的环境中测试假设和理论。通过模拟不同的情境和条件,我们可以深入了解模型的各种行为。

Table of Contents

1 引入

2 背景知识

3 simulation 的使用场景

4 simulation \rightarrow insight!

5 如何做 simulation ?

6 总结

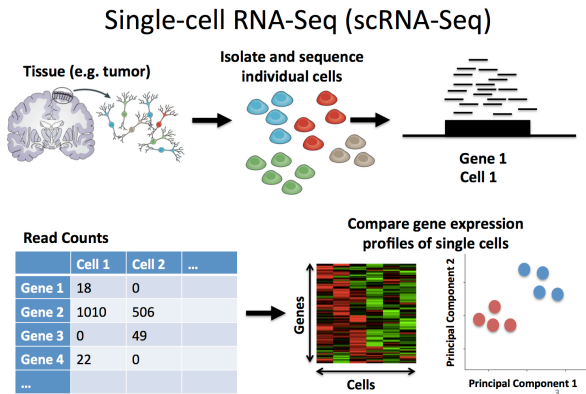
因为本节课以介绍为主，所以不需要额外的背景知识，Lecture 1 中介绍的基础统计知识已经足够。

Table of Contents

- 1 引入
- 2 背景知识
- 3 simulation 的使用场景**
- 4 simulation \rightarrow insight!
- 5 如何做 simulation ?
- 6 总结

复杂的输入：Single-cell RNA sequencing

单细胞 RNA 测序 (Single-cell RNA sequencing) 是一种提供单个细胞的序列信息的新时代测序技术。



图：单细胞 RNA 测序

复杂的输入：Single-cell RNA sequencing

对单细胞 RNA 测序数据的一个重要的任务：细胞分类。一种相对准确的方法是使用标记基因（marker gene）来区分一个组织样本中的各种细胞类型。

然而，标记基因有以下这些问题：

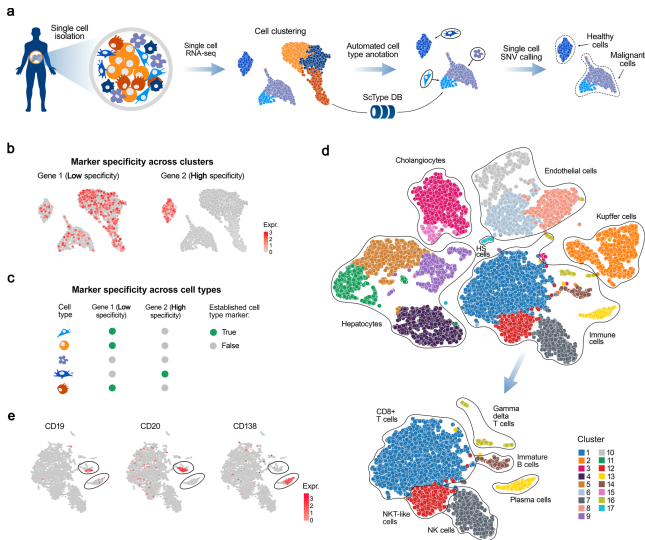
- ① 在多种细胞中表达
- ② 对应多种细胞

例如，CD44（一种基因）在多种免疫细胞群体中都有表达。

另外，单细胞 RNA 测序数据有这些问题：

- ① 数据缺失
- ② 数据噪声

复杂的输入：Single-cell RNA sequencing



图：标记基因

复杂的输入：Single-cell RNA sequencing

假设我们现在提出了一个概率模型去建模标记基因在单细胞 RNA 测序数据中的分布，我们需要验证它的合理性（比如是否真的可以在假设成立的前提下，成功建模单细胞 RNA 测序数据）。这时候复杂的真实数据无法满足我们对可控环境的要求。

复杂的输入：Single-cell RNA sequencing

假设我们现在提出了一个概率模型去建模标记基因在单细胞 RNA 测序数据中的分布，我们需要验证它的合理性（比如是否真的可以在假设成立的前提下，成功建模单细胞 RNA 测序数据）。这时候复杂的真实数据无法满足我们对可控环境的要求。

此时我们就需要借助 simulation 工具！

SymSim: single cell RNA-Seq data simulator

SymSim is an R package made to simulate single cell RNA-seq data. It can be used to generate a single population of cells with similar statistical properties to real data, or to generate multiple discrete or continuous populations of cells, where users can input a tree to represent relationships between multiple populations. SymSim has the following applications:

1. Benchmark clustering methods;
2. Benchmark methods for differentially expressed genes;
3. Benchmark trajectory inference methods;
4. Test the effects of different confounding factors on the performance of each computational method;
5. Estimate how many cells we need to sequence in order to detect a rare population under various realistic scenarios.

图: SymSim 模拟器

通过 simulation，可以在一定程度上验证模型的有效性。

Table 1: The distribution of marker genes using KRATOS and ACE. Here “C”, “D”, and “N” stand for the number of causal genes, dependent genes, and noise genes, respectively, among the top-20 identified marker genes. We see that KRATOS can extract comparable causal genes as ACE, but the noise genes chosen by KRATOS is far lower than ACE (for the most part, 0, vs ~ 5 for SOTA).

Clust ID	C_{Kratos}	D_{Kratos}	N_{Kratos}	C_{ACE}	D_{ACE}	N_{ACE}
1	8	12	0	9	7	4
2	8	11	1	8	6	6
3	10	10	0	10	5	5
4	10	10	0	8	7	5
5	9	11	0	9	5	6

复杂的模型：SpAM

稀疏加和模型 (SpAM) 是一类非参的统计模型，我们在 Lecture 2 中曾举过例子。下面是一篇关于为 SpAM 构建置信区间的统计文章。对于这种传统的统计方法而言，simulation 是必须的。

Let $\mathbf{X} = (X_1, \dots, X_d)^T$ be a d -dimensional random vector in \mathcal{X}^d . Without the loss of generality, in this paper, we assume $\mathcal{X} = [0, 1]$. The sparse additive model (SpAM) is of the form given in (1.2), with only a small number of additive components nonzero. Let $\mathcal{S} \subseteq [d]$ be of size $s = |\mathcal{S}| \ll d$. Then the model in (1.2) can be written as

$$Y = \mu + \sum_{j \in \mathcal{S}} f_j(X_j) + \varepsilon \quad (2.1)$$

此时，simulation 就需要有一个实验设置。

Example 5.1. We consider the sparse additive model $Y_i = \sum_{j=1}^4 f_j(X_{ij}) + \varepsilon_i$, where

$$f_1(t) = 6(0.1 \sin(2\pi t) + 0.2 \cos(2\pi t) + 0.3(\sin(2\pi t))^2 + 0.4(\cos(2\pi t))^3 + 0.5(\sin(2\pi t))^3),$$
$$f_2(t) = 3(2t - 1)^2, \quad f_3(t) = 5t, \quad f_4(t) = 4 \sin(2\pi t)/(2 - \sin(2\pi t)).$$

复杂的模型：SpAM

通过 simulation，我们可以验证方法的有效性。

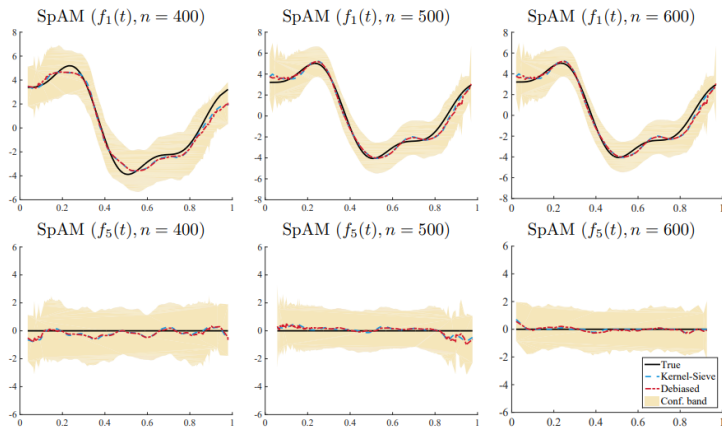


Figure 2: Kernel-sieve hybrid estimators for the $d = 600$ dimensional SpAM model $Y = \sum_{j=1}^d f_j(X_j) + \varepsilon$, for $n = 400, 500, 600$ and the noise $\varepsilon \sim N(0, 1.5^2)$. The confidence bands at significant level 95% cover $f_1(t)$ on the first row and $f_5(t) = 0$ on the second row.

复杂的输出：紊乱的损失

这一个部分我们来看一个实例（同学遇到的一个损失函数图像）。在这种情况下，从一种简单的数据开始 simulation 是非常好的 debug 的方式。

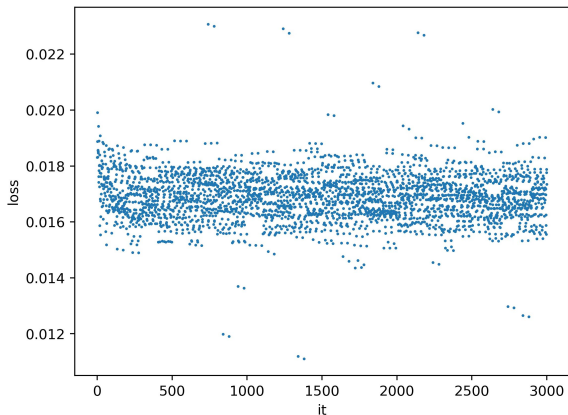
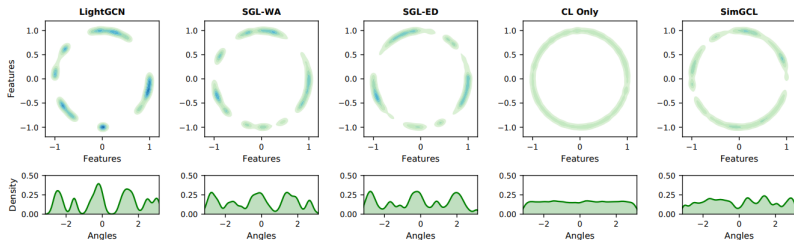


Table of Contents

- 1 引入
- 2 背景知识
- 3 simulation 的使用场景
- 4 simulation \rightarrow insight!**
- 5 如何做 simulation ?
- 6 总结

有时候，在一些简单的设置下看模型可以发现一些新的 insight



(a) Distribution of item representations learned from the dataset of Yelp2018.

Table of Contents

- 1 引入
- 2 背景知识
- 3 simulation 的使用场景
- 4 simulation \rightarrow insight!
- 5 如何做 simulation ?**
- 6 总结

如何做 simulation ?

我们这里只是简单地讲一些比较广泛的方法论。simulation 主要可以分为两个大部头：模型实现 + 实验设置

- ① 模型实现：如何把公式变成代码？
- ② 实验设置：怎么做实验设置？


通常来说，我们在这一步已经有写好的公式了，只需要将公式转化为代码即可。在这里面，最大的两个问题在于理解公式与矩阵化。

Algorithm 1 Training

```
1: repeat  
2:    $\mathbf{x}_0 \sim q(\mathbf{x}_0)$   
3:    $t \sim \text{Uniform}(\{1, \dots, T\})$   
4:    $\boldsymbol{\epsilon} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$   
5:   Take gradient descent step on  
      $\nabla_{\theta} \|\boldsymbol{\epsilon} - \boldsymbol{\epsilon}_{\theta}(\sqrt{\bar{\alpha}_t}\mathbf{x}_0 + \sqrt{1 - \bar{\alpha}_t}\boldsymbol{\epsilon}, t)\|^2$   
6: until converged
```

Algorithm 2 Sampling

```
1:  $\mathbf{x}_T \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$   
2: for  $t = T, \dots, 1$  do  
3:    $\mathbf{z} \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$  if  $t > 1$ , else  $\mathbf{z} = \mathbf{0}$   
4:    $\mathbf{x}_{t-1} = \frac{1}{\sqrt{\alpha_t}} \left( \mathbf{x}_t - \frac{1 - \alpha_t}{\sqrt{1 - \alpha_t}} \boldsymbol{\epsilon}_{\theta}(\mathbf{x}_t, t) \right) + \sigma_t \mathbf{z}$   
5: end for  
6: return  $\mathbf{x}_0$ 
```

 $\bar{\alpha}_t$ 是什么？ $\bar{\alpha}_t$ 要梯度吗？现实中怎么去实现 $\boldsymbol{\epsilon}(\cdot, \cdot)$ 比较好？

矩阵化有时候也是一个头疼的问题，比如下面这个例子：

$$\text{Estimator } \hat{f}_{mt} = \sum_{i=1}^n \alpha_{mti} k_m(\mathbf{x}_{ti}, \cdot).$$

我们现在已经得到了这个预测函数的参数 α ，预测的时候，对于一个新的点 \mathbf{x} ，我们想要计算 $\hat{f}_t(\mathbf{x}) = \sum_{i=1}^n \sum_{m=1}^M \alpha_{mti} k_m(\mathbf{x}_{ti}, \mathbf{x})$ 来做预测，怎么高速地进行预测？

- k 怎么矩阵化？有没有包支持矩阵化？
- 这个预测式怎么写成矩阵的形式？
- ...

实验设置

实验设置通常会包括以下的元素，这也是需要在 paper 中汇报的东西：

- 数据分布
- 各个参数 (e.g., 样本数)
- 超参数选择 (网格搜索, 训练集验证集划分, ...)
- 比较的指标 (AUC, acc, MSE, ...)
- 比较的模型 (baseline, sota, ...)
- ...

但核心在于**多试**，毕竟我们有没有免费的午餐定理，即没有一种机器学习算法是适用于所有情况的。换句话说，所有模型都有其特定的适用情况。

Table of Contents

- 1 引入
- 2 背景知识
- 3 simulation 的使用场景
- 4 simulation \rightarrow insight!
- 5 如何做 simulation ?
- 6 总结**

总结

我们今天介绍了以下内容：

- 什么是 simulation: 输入 + 模型 + 输出
- simulation 有什么用: 近似 + 测试 + 验证
- simulation 的使用场景
- 宏观角度下, 怎么做 simulation

1. $\hat{f}_t(\mathbf{x}) = \sum_{i=1}^n \sum_{m=1}^M \alpha_{mti} k_m(\mathbf{x}_{ti}, \mathbf{x})$, 其中 $\mathbf{x}, \mathbf{x}_{ti} \in \mathbb{R}^p$, $\alpha_{mti} \in \mathbb{R}$, k_m 对应第 m 个核函数。已知有一个包能够实现高速地计算单个核函数的核矩阵 $k_m(\mathbf{X}, \mathbf{X}') \in \mathbb{R}^{n_1 \times n_2}$, 其中 $\mathbf{X} = [\mathbf{x}_{t1}, \dots, \mathbf{x}_{tn_1}]^\top \in \mathbb{R}^{n_1 \times p}$, $\mathbf{X}' = [\mathbf{x}'_{t1}, \dots, \mathbf{x}'_{tn_2}]^\top \in \mathbb{R}^{n_2 \times p}$, 也就是说可以获得 n_1 个样本与 n_2 个样本之间的核矩阵。已知 M 个核函数无法并行计算 (即对于 $\sum_{m=1}^M$ 这一项加和只能用循环实现)。请自行定义参数的矩阵形式, 给出 `pytorch` 下矩阵实现预测函数的伪代码。