最大似然估计

Maximum Likehood Evaluation
learning note For reading translation

我真的不懂忧郁

Delft University of Technology

最大似然估计

Maximum Likehood Evaluation learning note For reading translation

by

我真的不懂忧郁

Student Name	Student Number
First Surname	1234567

Instructor:	I. Surname
Teaching Assistant:	I. Surname
Project Duration:	Month, Year - Month, Year
Faculty:	Faculty of Aerospace Engineering, Delft

Preface

A preface...

我真的不懂忧郁
Delft, June 2024

Summary

A summary...

目录

Nomenclature

If a nomenclature is required, a simple template can be found below for convenience. Feel free to use, adapt or completely remove.

Abbreviations

Abbreviation	Definition
ISA	International Standard Atmosphere
...	

Symbols

Symbol	Definition	Unit
V	Velocity	[m/s]
...		
ρ	Density	[kg/m^3]
...		

Chapter 1

背景

1.1. 极大似然估计 (maximum likelihood estimation)

给定概率分布 D，已知其概率密度函数（连续分布）或者概率质量函数（离散分布）为 f_D，以及一个分布参数 θ，我们可以从这个分布中抽一个具有 n 个值的采样 $X_1, X_2, \cdots, X_n$，利用 f_D 计算出其似然函数

$$L(\theta|x_1, \cdots, x_n) = f_\theta(x_1, \cdots, x_n) \tag{1.1}$$

若 D 是离散分布，f_θ 即是在参数为 θ 时观测到这一采样的概率；若其是连续分布，f_θ 则为 $X_1, X_2, \cdots, X_n$ 联合分布的概率密度函数在观测值处的取值。

从数学上来说，我们可以在 θ 的所有可能取值中寻找一个值使得似然函数取到最大值。这个使可能性最大的 $\widehat{\theta}$ 值即称为 θ 的最大似然估计。由定义，最大似然估计是样本的函数。

相对熵

最大似然估计可以从相对熵推导而来。相对熵衡量了使用一个给定分布 Q 来近似另一个分布 P 时的信息损失，对于离散随机变量

$$D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)} \tag{1.2}$$

其中 P 是真实分布，Q 是近似分布。在最大似然估计的情景下，假设分布拥有一系列参数 θ，我们希望通过样本得到参数的估计值 $\hat{\theta}$，我们可以利用相对熵来评估估计的好坏

$$D_{KL}(p_\theta(x)||p_{\hat{\theta}}(x)) = \sum_{x \in E} p_\theta(x) log \frac{p_\theta(x)}{p_{\hat{\theta}}(x)} \tag{1.3}$$

根据数学期望的定义，上式可以改写

$$D_{KL}(p_\theta(x)||p_{\hat{\theta}}(x)) = \mathbb{E}_\theta[log(\frac{p_\theta(x)}{p_{\hat{\theta}}(x)})] = \mathbb{E}_\theta[log\ p_\theta(x)] - \mathbb{E}_\theta[log\ p_{\hat{\theta}}(x)] \tag{1.4}$$

KL 值越大，参数估计越坏，因此，需要通过改变估计参数 $\hat{\theta}$ 的值来获得最小的值，所对应的参数极为最佳估计参数

$$\hat{\theta}_{best} = arg \min_{\hat{\theta}} D_{KL}(p_\theta(x)||p_{\hat{\theta}}(x)) \tag{1.5}$$

假设有 n 个样本，根据**大数定律**，

$$\mathbb{E}_\theta[log\ p_{\hat{\theta}(x)}] \rightsquigarrow \frac{1}{n}\sum_{i=1}^{n} log\ p_{\hat{\theta}}(x) \tag{1.6}$$

因此我们可以通过下式去估计

$$D_{KL}(p_\theta(x)||p_{\hat{\theta}}(x)) = \mathbb{E}_\theta[log\ p_\theta(x)] - \frac{1}{n}\sum_{i=1}^{n} log\ p_{\hat{\theta}}(x_i) \tag{1.7}$$

对于一个已知的分布，其参数 θ 是确定的。因此，$\mathbb{E}_\theta[log\ p_{\hat{\theta}(x)}]$ 为常数。因此，我们可以通过最小化 KL 值获得最佳估计参数：

$$\hat{\theta} = arg \min_{\theta} \mathbb{E}_\theta[log\ p_{\hat{\theta}(x)}] - \frac{1}{n}\sum_{i=1}^{n} log\ p_{\hat{\theta}}(x_i) \tag{1.8}$$

只要求和项最大，那么整体就最小，这个优化问题等价于

$$\begin{aligned} & arg \max_\theta \frac{1}{n}\sum_{i=1}^{n} log\ p_{\hat{\theta}}(x_i) \\ & \Rightarrow arg \max_\theta log[\prod_{i=1}^{n} p_{\hat{\theta}}(x_i)] \\ & \Rightarrow arg \max_\theta \prod_{i=1}^{n} p_{\hat{\theta}}(x_i) \end{aligned} \tag{1.9}$$

因此，要得到最佳参数估计值，只需要最大化 $\prod_{i=1}^{n} p_{\hat{\theta}}(x_i)$，这就是最大似然函数。

1.2. 频率派 vs. 贝叶斯派

频率派和贝叶斯派的区别是是否允许先验估计。

Frequentist	Bayesian
频率论方法通过大量独立实验将概率解释为统计均值（大数定律）	贝叶斯方法则将概率解释为信念度（degree of belief）（不需要大量的实验）
频率学派把未知参数看作普通变量（固定值），把样本看作随机变量	贝叶斯学派把一切变量看作随机变量
频率论仅仅利用抽样数据	贝叶斯论善于利用过去的知识和抽样数据

图 1.1: 频率派和贝叶斯派

References

[1] I. Surname, I. Surname, and I. Surname. “The Title of the Article”. In: *The Title of the Journal* 1.2 (2000), pp. 123–456.

Chapter A

Source Code Example

Adding source code to your report/thesis is supported with the package listings. *An example can be found below. Files can be added using* \lstinputlisting[language=<language>]{<filename>}.

```python
"""
ISA Calculator: import the function, specify the height and it will return a
list in the following format: [Temperature,Density,Pressure,Speed of Sound].
Note that there is no check to see if the maximum altitude is reached.
"""

import math
g0 = 9.80665
R = 287.0
layer1 = [0, 288.15, 101325.0]
alt = [0,11000,20000,32000,47000,51000,71000,86000]
a = [-.0065,0,.0010,.0028,0,-.0028,-.0020]

def atmosphere(h):
    for i in range(0,len(alt)-1):
        if h >= alt[i]:
            layer0 = layer1[:]
            layer1[0] = min(h,alt[i+1])
            if a[i] != 0:
                layer1[1] = layer0[1] + a[i]*(layer1[0]-layer0[0])
                layer1[2] = layer0[2] * (layer1[1]/layer0[1])**(-g0/(a[i]*R))
            else:
                layer1[2] = layer0[2]*math.exp((-g0/(R*layer1[1]))*(layer1[0]-layer0[0]))
    return [layer1[1],layer1[2]/(R*layer1[1]),layer1[2],math.sqrt(1.4*R*layer1[1])]
```

Chapter B

Task Division Example

If a task division is required, a simple template can be found below for convenience. Feel free to use, adapt or completely remove.

表 **B.1:** Distribution of the workload

	Task	Student Name(s)
	Summary	
Chapter 1	Introduction	
Chapter 2		
Chapter 3		
Chapter *		
Chapter *	Conclusion	
	Editors	
	CAD and Figures	
	Document Design and Layout	