实验 C10 基于 Linux 和 Python 的粒子实验模拟数据分析*

XXX†

（1. 中山大学物理学院，广东广州 510275）

摘 要： Linux 操作系统是一款高度定制性的操作系统，具有开源、免费、支持多用户、多线程、多 CPU 等特色，在超算、云计算、数据中心、企业 WEB 服务器、硬件虚拟化、人工智能等领域广泛应用。目前网络应用的服务器，大部分采用 Linux 操作系统。Python 是一种解释型、面向对象、动态的高级程序设计语言。经过众多开源社区人员的开发和建设，它已成为多数平台上写脚本和快速开发应用程序的编程语言。Python 因跨平台、开源、简洁、易读、有大量科学计算拓展库的特点，十分适合工程技术、科研人员进行数据处理、图表制作、开发科学计算应用程序等工作。将 Linux 和 Python 的配合使用，为物理学研究提供了一个有力的工具。在粒子物理实验中需要使用统计分析的方法为特定的事件进行分类，以判断该事件更可能是哪种。本实验将了解 Linux 系统的基本架构，熟悉基础代码操作指令，在 Linux 系统中运行 C 语言等。在 Jupyter Notebook 交互式编程中熟悉 Python 的 Numpy 和 Pandas 库，熟悉 Markdown 写作形式，并运行基础 Python 程序。最后在 Python 环境中模拟两个存在相互影响的事件，然后通过使用极大似然法对该事件进行分离，并评估模型的置信区间。

关键词：Linux，Python，极大似然法，置信区间

Experiment C10: Data Analysis of Particle Experiment Simulation Based on Linux and Python*

XXX[1]

1. School of Physics, Sun Yat-sen University, Guangzhou 510275, China

Abstract： Linux operating system is a highly customized operating system. It is open source, free, which also supports multi-user, multi-thread, multi-CPU and other features. It is widely used in super computing, cloud computing, data center, enterprise WEB server, hardware virtualization, artificial intelligence and other fields. Most current network application servers are using the Linux operating system. Python is an object - oriented, dynamic, high-level programming language. Through the development and construction of many people in the open source community, Python has gained more popularity for scripting and rapid application on digital platforms. Python is a cross-platform language with open source, and has a lot of applications in science. It is very suitable for engineering and carrying out data processing, chart making and scientific project development calculation and so on for scientific research personnel. The combination of Linux and Python provides a powerful tool for physics research. Statistical analysis is used in quantum physics experiments to classify specific events in determining which result is more likely. This experiment helped me understand the basic construction of Linux system, familiar with the basic code operation instructions, run C programs in Linux system. What's more, I was then familiar with Python's Numpy and Pandas in Jupyter Notebook, including Markdown writing form, together with running basic Python programs. Finally, two events with mutual influence were simulated in Python environment. After that the event was separated by maximum likelihood method and estimated the confidence interval of the model.

Key words： Linux, Python, maximum likelihood method, confidence interval

1 引 言

本实验首先让同学们对 Linux 系统及数据处理方法有一定了解，了解 Linux 操作系统的历史，理解与 windows 操作系统的差别，学习 Linux 系统终端 (terminal) 的常用命令和系统环境变量，掌握 Linux 系统的基本基本使用方法，学会使用开源编译器 gcc，创建 Makefile，辅助编译多文件交叉引用，为未来使用该系统开展工作打下基础。在 Python 语言方面学习 Linux 系统下 Python 语言的基本编程方法及 Numpy、Pandas、SciPy 库文件的应用方法并利用 Python 进行物理实验数据拟合。最后用极大似然估计法进行数据分析，学习置信度和置信区间的过程中熟练使用 Python 的 Numpy,Scipy,Matplotlib 等库。

2 介 绍

2.1 Linux 系统

Linux 系统将所有操作权都交给用户，向用户开源包括系统内核在内的所有细节，是一款高度定制性的操作系统。Linux 具有开源、免费、跨平台、多用户、多任务、设备独立性强、丰富的网络功能、可靠的安全性、稳定性和良好的可移植性等特点。当前 80% 以上的服务器使用该系统，如超算中心、云计算中心、数据中心、企业 WEB 服务器、硬件虚拟化、人工智能等，手机上使用的 android 系统也是使用的 Linux[1]。

因 Linux 的开放性和完全可控性，使得开发人员可以看到程序是如何运行的，编译器会给出程序运行报错的提示，根据错误提示，开发人员更容易找到开发过程中遇到的问题的解决方法。Linux 系统在给予用户完全控制权的同时也隐含了极大的风险，为了避免用户误操作指令使操作系统崩溃，Linux 发行版一般提供管理员和普通用户，通过权限设置不同来有效避免误操作。

现今主流的 Linux 桌面环境有 KDE，gnome，Xfce，LXDE 等，除此之外还有小众的 Ambient，EDE，IRIX Interactive Desktop，Mezzo，Sugar，CDE 等。

Linux 操作系统诞生、发展和成长过程始终依赖着五个重要支柱：UNIX 操作系统、MINIX 操作系统、GNU 计划、POSIX 标准和 Internet 网络 [2]。

Linux 系统的基本架构包括硬件（Hardware）、内核（Kernel）、用户界面（Shell）、应用程序（Utilities）等四大部分 [3]。

2.2 Python

2.2.1 Python 简介

Python 是一种解释型、面向对象、动态的高级程序设计语言。经过众多开源社区人员的开发和建设，它已成为多数平台上写脚本和快速开发应用程序的编程语言。Python 语言简洁易读，有丰富的开源标准库，具有良好的跨平台特性，同时易于扩展，可以使用 C、C++ 扩展新的功能和数据类型。

Python 有许多专用的科学计算扩展包，如 Numpy、SciPy 和 Matplotlib 等，可以提供快速数组处理、数值运算、常用科学计算应用和数据可视化。而近年来人工智能的二次兴起，擅长处理机器学习（Machine Learning）的 Scikit-learn，高级数据结构分析库 Pandas，以 TensorFlow、Keras 和 Pytorch 为代表的机器学习库更是成为了深度学习领域高频使用的技术选择。

---

*由中山大学物理学院陆佑堂提供器材和指导。

†作者简介：XXX，xxxxxxxx；E-mail: xxxx@mail2.sysu.edu.cn

2.2.2 交互式编程与文本式编程

Python 既可以在 IPython, Jupyter Notebook，Jupyter Lab 等场景下使用交互式编程，也可以利用 PyCharm、Spyder、Sublime Text 等 IDE（集成开发环境）或文本编辑工具进行文本式编程。

交互式编程的特点是文本介绍和程序相结合，通过分块单元（Cell）分割运行不同的程序模块和文本模块，程序的运行以单元为准，可以实现“即见即所得”的编程方式，为构建程序提供良好的交互性，分块化编程也降低程序调试的难度。此外，支持 Markdown 的文本模块穿插在程序模块间，增强程序的可读性，将以程序为核心的编程模式转变为互动交流和展示结果为核心的文学编程交互模式。因此，交互式编程非常合适在数据分析领域，报告、学习等场景下使用。另一方面，交互式编程存在局限性，因其单元分割的特点，和编程环境的限制，不利于大规模程序的开发调试和程序的扩展和重用，且不能编译成静态或动态链接库。

文本式编程是传统的编程模式，它通过文本编辑器或 IDE 将程序语句写到一个文件中（一般存为.py 格式），然后通过已经安装好的 Python 编译器对 py 文件进行编译运行。运行方式和 C,C++ 等传统程序方式相同，其特点是，以程序为核心，注释为辅助，是程序项目实施的主要载体，实施代码重用，程序拓展、形成算法库的基础。编写程序，实现功能，对程序进行封装常以文本编程模式优先；交流、展示和学习则可以考虑使用交互式编程。

2.3 粒子物理实验数据处理

2.3.1 粒子物理实验中的统计问题

粒子加速器实验会提供一次事件的多种信息，比如欧洲粒子加速器 LHC 中的 Bolometer 实验，仅仅提供某次事件发生的总能量，但中微子闪烁探测器 SNO+，则会提供该事件发生的位置、能量以及可能的方向等信息。粒子实验过程中，并不会针对某一个单一事件的信息进行采集，比如在无中微子双 β 衰变能量范围内的数据，有可能存在太阳中微子、辐射衰变这两类事件所产生的信号，需要通过数据分析的方法区分开来。另外，受限于探测器的能量分辨率，实际测量中还需要提高信噪比来区分各种事件，使得问题更加复杂。

在实践中，我们利用不同类型事件具有不同的统计分布规律这一特性来对数据进行分类，进而分析得到有用的信息，例如，探测器获取的数据中，无中微子双 衰变的信号发生在分离能范围，太阳中微子事件会更倾向于均匀分布，而放射性衰变产生的本底事件则更容易发生在高放射性同位素所处区域的附近等。利用这些特性，可以为每类事件定义一个概率密度函数（probability density function 简称 PDF），然后使用极大似然法对实验数据进行分类。本实验中，我们将以能量信号分类为例，用随机数产生两组已知特征且存在交叠区域的事件 A、B，再通过极大似然估计法将另外产生的待测信号分离出来。

2.3.2 极大似然估计（Maximum Likelihood Analysis）

极大似然估计法是一种基于频率的参数估计方法。假设某个统计对象，总体分布满足的概率密度函数为 $f(X;\theta)$，其中 X 为变量，θ 为参数。如果从总体分布中抽出样本 $(x_1, x_2, ...x_n)$，可以得到这些样本的联合分布概率为：

$$L(x_1, x_2, ...x_n; \theta) = f(x_1; \theta)f(x_2; \theta)...f(x_n; \theta) = \prod_{i=1}^{n} f(x_i; \theta) \tag{1}$$

若 θ 已知，则 L 是以 $(x_1, x_2, ...x_n)$ 为变量的概率密度函数。实践中，往往是已知 x_i，而参数 θ 待求，此时，L 称为似然函数 (Likelihood)。若 $\theta = \theta^*$ 时 L 最大，则表示参数为 θ^* 的概率分布最可能符合当前的抽样结果。因此把

$$\theta^* = argmaxL(x_1, x_2, ...x_n; \theta) \tag{2}$$

称为 θ 的极大似然估计值。

如前所述的例子中，假设 A、B 两类事件均满足正态分布，

$$f(x;\mu,\sigma)=\frac{1}{\sigma\sqrt{2\pi}}exp^{-\frac{(x-\mu)^2}{2\sigma^2}} \tag{3}$$

其中 A 事件的能量分布满足 $\mu=10MeV$，$\sigma=2MeV$；B 事件的能量分布满足 $\mu=15MeV$，$\sigma=3MeV$。这样，两类事件存在比较宽的交叠区域。如果测得一组实验数据同时包含 A、B 两类事件，我们将利用 A、B 模型产生的结果为参考，对实验数据进行分析。

首先将 0-25MeV 划分为 40 个区间，统计落入到各个区间内的粒子数。能量信号满足抽样分布特征，在事件数较少时，随机落入第 i 个区间的概率更接近泊松分布

$$\pi_i=f(k_i;\lambda_i)=\frac{\lambda_i^{k_i}e^{-\lambda_i}}{k_i!} \tag{4}$$

其中，k_i 是第 i 个分区的实测值，λ_i 是该分区的期望事件数。由于第 i 个分区中发生的事件可能来自于 A 或 B，因此，该区间的期望值与 A、B 模型的已知分布的关系为

$$\lambda_i=\sum_j \mu_j H_{ij} \tag{5}$$

下标 j 表示不同的事件 (j = A, B)，H_{ij} 是第 j 类事件的归一化因子，可由 A、B 的已知事件获得，μ_j 是第 j 类事件在第 i 个分区中的期望值。

根据似然函数的定义，可得似然函数为

$$L(k;\lambda)=\prod_i^n f(k_i;\lambda_i)=\prod_i^n \frac{\lambda_i^{k_i}e^{-\lambda_i}}{k_i!} \tag{6}$$

对似然函数 L 求满足极大条件下的 λ 值，可获得 AB 的分布情况。由于似然函数 L 中存在连乘、阶乘以及幂函数等函数，直接求解通常都会非常复杂。对似然函数 L 求负对数可得

$$-lnL=\sum_i \lambda_i-k_i ln(\lambda_i)+ln(k_i!) \tag{7}$$

因此, 求解 L 的最大值可等价为求-lnL 的极小值。由于 $ln(k_i)$ 不含参数 λ_i，在求函数梯度时相当于常数，因此式（7）可简化为

$$-lnL=\sum_i \lambda_i-k_i ln(\lambda_i) \tag{8}$$

2.3.3 置信区间

似然函数表征依赖于特定参数（如 θ，λ）的概率，受抽样影响，获得的极值必然存在统计误差。为了评估极大似然法获得的参数的置信区间，本实验将利用泊松分布函数和卡方分布函数 χ^2 相似的特点，结合 Wilks 定理进行分析。在大样本情况下，似然函数的分布与 χ^2 分布相似，满足

$$\chi^2=-2ln\frac{L(k;\mu_1;\mu^*_{i\neq 1})}{L(k;\mu^*)} \tag{9}$$

其中 L 是某个参数（如 μ_1）取极大似然时，其他参数对应的分布，L 则为所有参数的极大似然分布。似然函数采用负对数形式，可得

$$\chi^2=2(l\ -l)=2\Delta l \tag{10}$$

其中，$l=-lnL$。在一个自由度的情形下，χ^2 的置信度水平为 1σ 时 $\chi^2=1$，与 $\Delta l=1/2$ 等价。换而言之，可以通过固定某个参数为极大似然参数（如 $\mu_A=\mu_A^*$），然后扫描改变其他参数（如 μ_B），分别从 μ_B 的取值上界和下界扫描，得到满足 $\Delta l=1/2$ 的参数值 μ_{lo} 和 μ_{hi}，则 $\mu_B=\mu_B^*$ 也是极大似然参数。置信水平为 1σ 时，置信区间为 $[\mu_{lo},\mu_{hi}]$。

3 实验仪器

安装了 Linux 操作系统（本实验使用 *ubuntu_64*）、Python3 及 Numpy、Matplotlib、Scipy 库、Jupyter Notebook 和 LibreOffice 软件的计算机。

4 实验成果

4.1 Linux 的使用

4.1.1 Linux 系统命令行操作和桌面环境使用

在以下操作中，我们初步了解了 Linux 系统

1. 查看系统相关信息
2. 文件夹和文件的基本操作
3. 环境变量的查看和设置
4. 设置命令的别名或者简化命令
5. 图形桌面环境的使用

4.1.2 GCC 编译器配合 Makefile 制作可执行文件

编译 C 语言文件及交叉编译分别见下图：

```
(base) manager@PC4:~/doc/3.1$ gcc -c test
gcc: warning: test: linker input file unused because linking not done
(base) manager@PC4:~/doc/3.1$ gcc -c print.c print.h
print.c: In function ‘printHello’:
print.c:3:1: warning: implicit declaration of function ‘printf’ [-Wimplicit-func
tion-declaration]
 printf("Hello World!\n");
 ^
print.c:3:1: warning: incompatible implicit declaration of built-in function ‘pr
```

```
(base) manager@PC4:~/doc/3.1$ gcc -c main.c print.h
(base) manager@PC4:~/doc/3.1$ gcc -o test main.o print.o
(base) manager@PC4:~/doc/3.1$ make
Makefile:2: *** missing separator.  Stop.
(base) manager@PC4:~/doc/3.1$ ./test
Hello World!
```

4.1.3 标准开源软件的安装

安装代码如下：

```
$ cp ~/Downloads/gsl-2.5.tar.gz ~/Exercise/
$ cd ~/Exercise
$ tar -xzvf gsl-2.5.tar.gz #tar 为 tape archive 缩写，用来建立、还原备份文件的命令。$ cd gsl-2.5
$ ./configure --prefix=/YOUR/INSTALL/DIRECTORY
$ make
$ make install
$ make check
```

4.1.4 环境变量的设置及常用函数库的调用（以快速傅里叶变换为例）

首先编写程序 *fftw_test.cpp*，将程序保存到自己目录下，其次设置环境变量并编译程序，最后准备数据并执行程序。

执行结果如下：

```
(base) manager@PC5:~/Documents/10.1/Exercise$ g++ -o test.x fftw_test.cpp -L$LD_LIBRARY_PATH -I$INCLUDE -lfftw3
(base) manager@PC5:~/Documents/10.1/Exercise$ ./test.x
reading
There are 93 data
initializing 1D FFT...
FFT initialized
 Reading data
0 1.46356 0
1 1.49957 0
2 1.4463 0
3 1.3085 0
4 1.09847 0
5 0.83499 0
6 0.54158 0
7 0.24446 0
8 -0.02984 0
9 -0.2568 0
10 -0.41617 0
11 -0.49369 0
12 -0.48245 0
13 -0.38345 0
14 -0.20554 0
15 0.0354 0
16 0.31784 0
```

4.2 Linux 上 Python 的使用

4.2.1 Numpy 的使用

使用 numpy 产生不同数组，以四维数组为例：

```
array([[[[  1,   2,   3,   4,   5],
         [  6,   7,   8,   9,  10],
         [ 11,  12,  13,  14,  15],
         [ 16,  17,  18,  19,  20],
         [ 21,  22,  23,  24,  25]],

        [[ 26,  27,  28,  29,  30],
         [ 31,  32,  33,  34,  35],
         [ 36,  37,  38,  39,  40],
         [ 41,  42,  43,  44,  45],
         [ 46,  47,  48,  49,  50]],
```

```
12
13          [[ 51,  52,  53,  54,  55],
14           [ 56,  57,  58,  59,  60],
15           [ 61,  62,  63,  64,  65],
16           [ 66,  67,  68,  69,  70],
17           [ 71,  72,  73,  74,  75]]],
18
19
20         [[[ 76,  77,  78,  79,  80],
21           [ 81,  82,  83,  84,  85],
22           [ 86,  87,  88,  89,  90],
23           [ 91,  92,  93,  94,  95],
24           [ 96,  97,  98,  99, 100]],
25
26          [[101, 102, 103, 104, 105],
27           [106, 107, 108, 109, 110],
28           [111, 112, 113, 114, 115],
29           [116, 117, 118, 119, 120],
30           [121, 122, 123, 124, 125]],
31
32          [[126, 127, 128, 129, 130],
33           [131, 132, 133, 134, 135],
34           [136, 137, 138, 139, 140],
35           [141, 142, 143, 144, 145],
36           [146, 147, 148, 149, 150]]]])
```

多维度切片：

```
1   array([[[[  1,   2,   3,   4,   5],
2            [  6,   7,   8,   9,  10],
3            [ 11,  12,  13,  14,  15],
4            [ 16,  17,  18,  19,  20],
5            [ 21,  22,  23,  24,  25]],
6
7           [[ 26,  27,  28,  29,  30],
8            [ 31,  32,  33,  34,  35],
9            [ 36,  37,  38,  39,  40],
10           [ 41,  42,  43,  44,  45],
11           [ 46,  47,  48,  49,  50]],
12
13          [[ 51,  52,  53,  54,  55],
14           [ 56,  57,  58,  59,  60],
15           [ 61,  62,  63,  64,  65],
16           [ 66,  67,  68,  69,  70],
```

```
         [ 71,  72,  73,  74,  75]]],


       [[[ 76,  77,  78,  79,  80],
         [ 81,  82,  83,  84,  85],
         [ 86,  87,  88,  89,  90],
         [ 91,  92,  93,  94,  95],
         [ 96,  97,  98,  99, 100]],

        [[101, 102, 103, 104, 105],
         [106, 107, 108, 109, 110],
         [111, 112, 113, 114, 115],
         [116, 117, 118, 119, 120],
         [121, 122, 123, 124, 125]],

        [[126, 127, 128, 129, 130],
         [131, 132, 133, 134, 135],
         [136, 137, 138, 139, 140],
         [141, 142, 143, 144, 145],
         [146, 147, 148, 149, 150]]])
```

改变数组维度后组合：

```
   [[[ 11.  12.  13.  14.  15.]
   [ 36.  37.  38.  39.  40.]
   [ 61.  62.  63.  64.  65.]]

   [[ 86.  87.  88.  89.  90.]
   [111. 112. 113. 114. 115.]
   [136. 137. 138. 139. 140.]]]
   Origin dimension (2, 3, 5, 5)
   Average with axis=0: (3, 5, 5)
   Average with axis=1: (2, 5, 5)
   Average with axis=2: (2, 3, 5)
```

获得满足条件的数据的位置:

```
array([  2,   4,   6,   8,  10,  12,  14,  16,  18,  20,  22,  24,  26,
       28,  30,  32,  34,  36,  38,  40,  42,  44,  46,  48,  50,  52,
       54,  56,  58,  60,  62,  64,  66,  68,  70,  72,  74,  76,  78,
       80,  82,  84,  86,  88,  90,  92,  94,  96,  98, 100, 102, 104,
      106, 108, 110, 112, 114, 116, 118, 120, 122, 124, 126, 128, 130,
      132, 134, 136, 138, 140, 142, 144, 146, 148, 150])
```

理解广播功能对数据表的影响:

```
arr3:
  [[[0  1  2]]

  [[3  4  5]]

  [[6  7  8]]]
  arr2.shape= (3, 3, 3, 3)
  arr3.shape=(3, 1, 3)
  res.shape=(3, 3, 3, 3)
```

基于 numpy 的矩阵运算：

```
[[0  1  2]
[3  4  5]
[6  7  8]]
[[0.84031151  0.50897893  0.40527553]
[0.5581838   0.8295436   0.62273949]
[0.59079143  0.20958703  0.93200127]]
[[ 1.73976666   1.24871767   2.48674203]
[ 7.70762689   5.89304633   8.36679088]
[13.67548711  10.537375     14.24683974]]
[[ 1.73976666   1.24871767   2.48674203]
[ 7.70762689   5.89304633   8.36679088]
[13.67548711  10.537375     14.24683974]]
[[ 2.06454261  -1.2511201   -0.06178823]
[-0.48935601   1.74687173  -0.95442143]
[-1.19865868   0.40024556   1.32675604]]
12
```

4.2.2 pandas 使用练习

读取外部数据"C0.4data"，利用 pandas 库，学习表格型数据结构的概念和应用场景，对比 numpy 类数据的区别。

原始数据如下：

ID	Class	Name	Gendar	ClassSponsor	Math	English	Physics
2011A01	2011级A班	麦麦提	男	郭靖	88	87	90
2011A02	2011级A班	孙涵	男	郭靖	90	59	91
2011A03	2011级A班	汤子	女	郭靖	92	98	92
2011A04	2011级A班	王婧文	男	郭靖	84	71	90
2011A05	2011级A班	王瑞	女	郭靖	75	58	72
...	...	...	...	...	...	...	...
2013B07	2013级B班	梁芷若	男	周伯通	96	77	90
2013B08	2013级B班	廖军迎	男	周伯通	83	74	89
2013B09	2013级B班	廖志广	女	周伯通	91	51	87
2013B10	2013级B班	刘家灵	男	周伯通	93	67	87
2013B11	2013级B班	赵燕俐	男	周伯通	95	95	89

选取其中几列，展现如下：

ID	Class	ClassSponsor	Name	Physics
2011A10	2011级A班	郭靖	许阳刚	92
2011A03	2011级A班	郭靖	汤子	92
2011B01	2011级B班	郭靖	杨昭贤	92
2012B05	2012级B班	黄蓉	王庆	92
2013A01	2013级A班	周伯通	曹芳	91
...	...	...	...	...
2011A05	2011级A班	郭靖	王瑞	72
2012B11	2012级B班	黄蓉	肖庆鹏	70
2011A09	2011级A班	郭靖	谢烨	64
2011A11	2011级A班	郭靖	杨劼池	58
2011B05	2011级B班	郭靖	袁灵娟	49

选取其中几列，展现如下：

ID	Class	Name	Gendar	ClassSponsor	Math	English	Physics
2011A01	2011级A班	麦麦提	男	郭靖	88	87	90
2011A02	2011级A班	孙涵	男	郭靖	90	59	91
2011A03	2011级A班	汤子	女	郭靖	92	98	92
2011A04	2011级A班	王婧文	男	郭靖	84	71	90
2011A05	2011级A班	王瑞	女	郭靖	75	58	72
2011A06	2011级A班	魏志	男	郭靖	84	97	89
2011A07	2011级A班	文玺	男	郭靖	87	93	89
2011A08	2011级A班	吴茼茼	男	郭靖	92	78	87
2011A09	2011级A班	谢烨	男	郭靖	61	94	64
2011A10	2011级A班	许阳刚	男	郭靖	94	68	92
2011A11	2011级A班	杨劼池	女	郭靖	58	61	58
2011A12	2011级A班	杨燊	男	郭靖	86	77	87

条件筛选，以 2011 级数学少于 70 分为例：

ID	Class	Name	Gendar	ClassSponsor	Math	English	Physics
2011A09	2011级A班	谢烨	男	郭靖	61	94	64
2011A11	2011级A班	杨劼池	女	郭靖	58	61	58

计算班级平均分，展现如下：

Class	Math	English	Physics
2011级A班	82.583333	78.416667	83.416667
2011级B班	85.416667	70.333333	85.583333
2012级A班	86.500000	79.333333	87.833333
2012级B班	87.846154	78.538462	87.076923
2013级A班	88.100000	76.600000	88.100000
2013级B班	89.454545	77.727273	88.363636

提取行名关键字，数据重排：

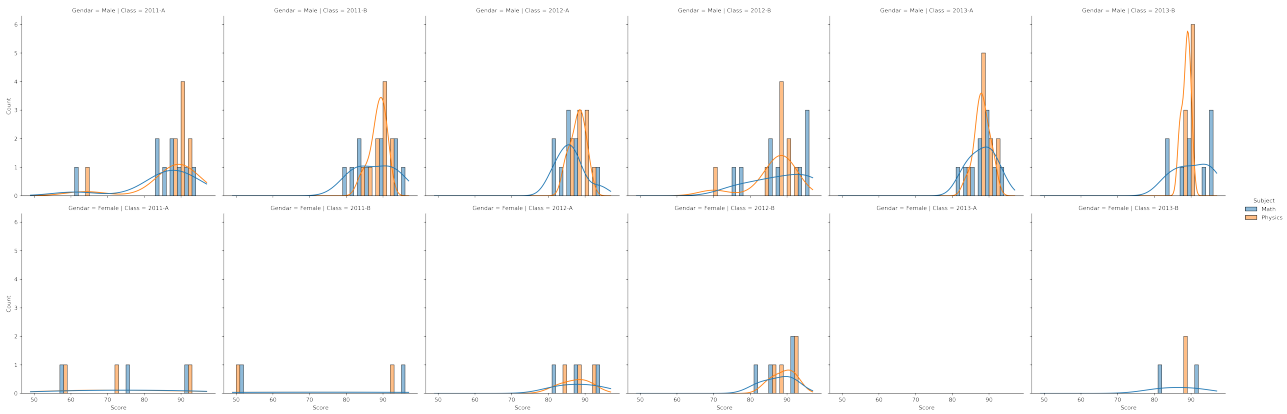
ID	Class	Name	Gendar	ClassSponsor	Math	English	Physics	Grade	Classes
2011A01	2011级A班	麦麦提	男	郭靖	88	87	90	2011	A
2011A02	2011级A班	孙涵	男	郭靖	90	59	91	2011	A
2011A03	2011级A班	汤子	女	郭靖	92	98	92	2011	A
2011A04	2011级A班	王婧文	男	郭靖	84	71	90	2011	A
2011A05	2011级A班	王瑞	女	郭靖	75	58	72	2011	A
...	...	...	...	...	...	...	...	...	...
2013B07	2013级B班	梁芷若	男	周伯通	96	77	90	2013	B
2013B08	2013级B班	廖军迎	男	周伯通	83	74	89	2013	B
2013B09	2013级B班	廖志广	女	周伯通	91	51	87	2013	B
2013B10	2013级B班	刘家灵	男	周伯通	93	67	87	2013	B
2013B11	2013级B班	赵燕俐	男	周伯通	95	95	89	2013	B

做各班的数据透视表，查看不同年级，不同班的物理、英语成绩平均值、人数，以及均方差。

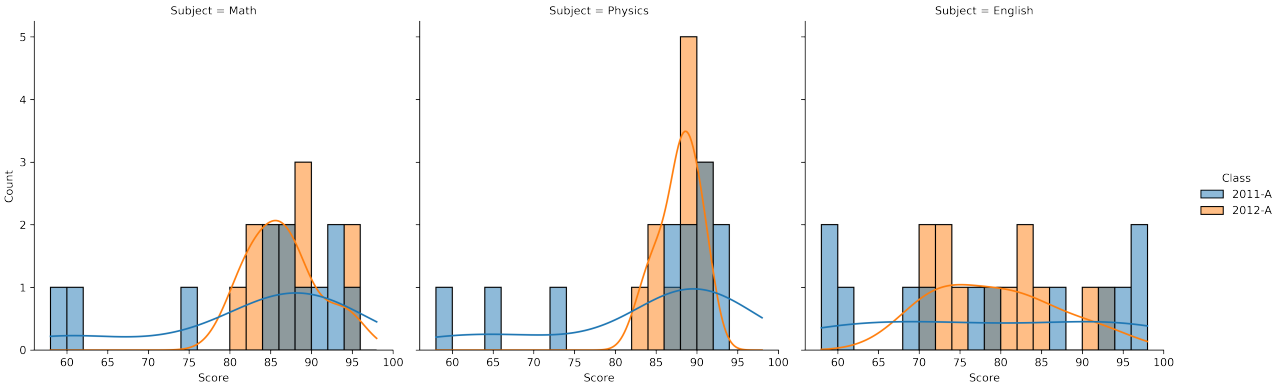
		English		Physics		
		mean	std	count	mean	std
Grade	Classes					
2011	A	78.416667	15.138502	12.0	83.416667	11.804917
	B	70.333333	16.400296	12.0	85.583333	11.735404
2012	A	79.333333	7.487363	12.0	87.833333	2.480225
	B	78.538462	11.990381	13.0	87.076923	5.604485
2013	A	76.600000	14.354248	10.0	88.100000	2.078995
	B	77.727273	15.172942	11.0	88.363636	1.206045

4.2.3 数据可视化

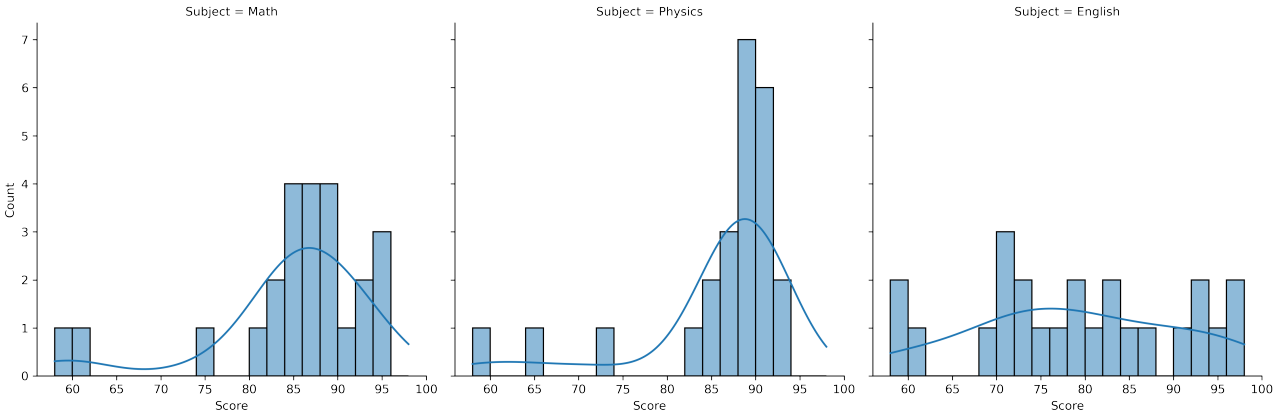
按不同班，画出男女生的物理、数学成绩分布



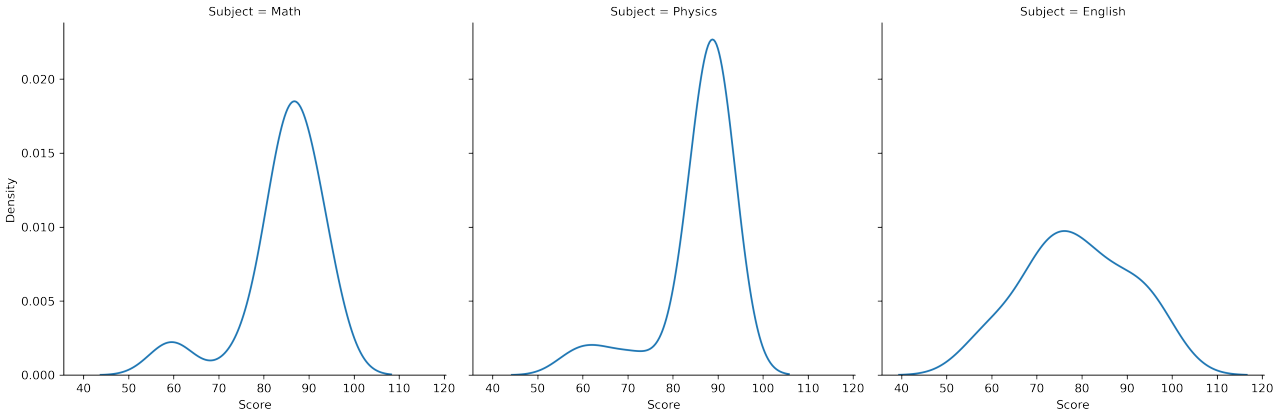
统计两个班的成绩分布情况



画出班级 2011 级和 2012 级 A 班各科成绩的分布图



分析密度函数



4.3 粒子实验模拟数据处理

4.3.1 产生实验用模拟信号

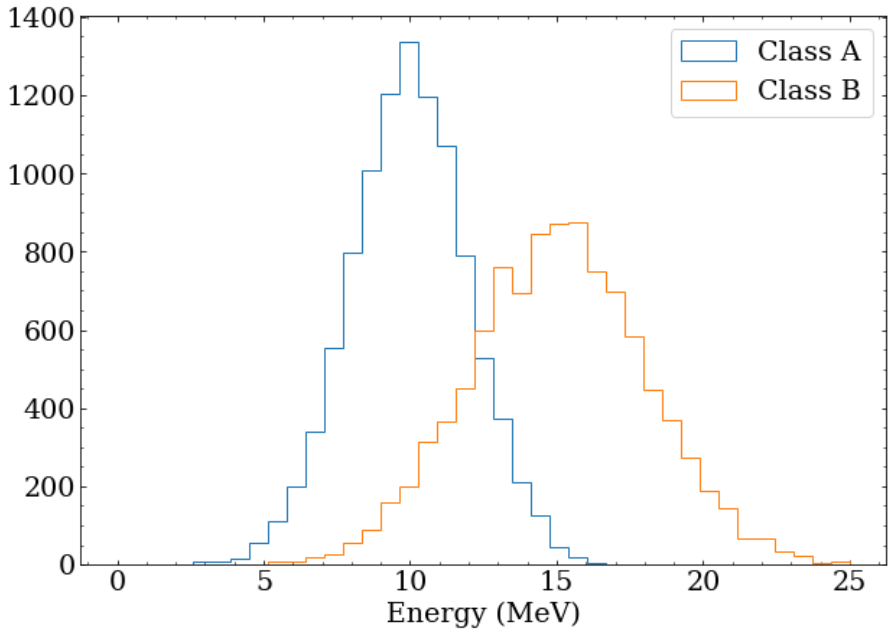
本实验采用粒子实验模拟数据进行分析。利用 Numpy 库中的 random.normal 函数产生两组满足正态分布的随机数，以模拟两类测量结果接近的粒子实验 A 事件和 B 事件。其中：

1. A 类事件平均能量是 10MeV, 半高宽为 2MeV

2. B 类事件平均能量是 15Mev, 半高宽为 3MeV

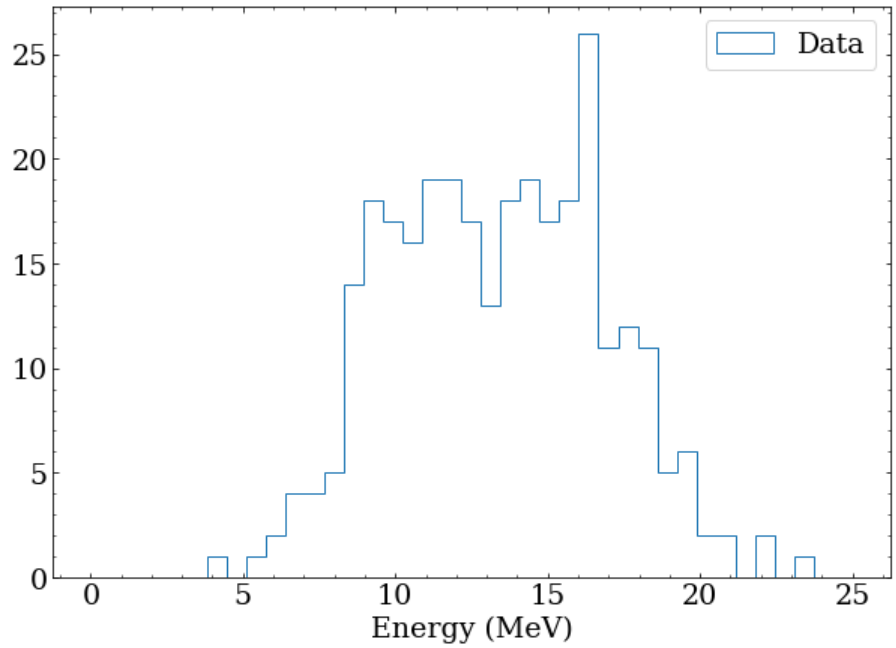
这样，A 类和 B 类事件之间会存在信号交叠。我们利用已知的 A、B 各 1 万个事件作为分析依据，对假想实验中测得一组由 100 个 A 类、200 个 B 类的赝事件组成的信号进行分类。A、B 的已知事件结果存在 *mc_class_a* 和 *mc_class_b* 中，而等待分析的赝事件则存在 data 中，用于后面的极大似然分析。此外，本实验将 0-25MeV 的能量划分为 40 个区间，后面数据分析都以此为基础。

分别产生 10000 个 A、B 赝事件，可以看出数据分布近似正态分布

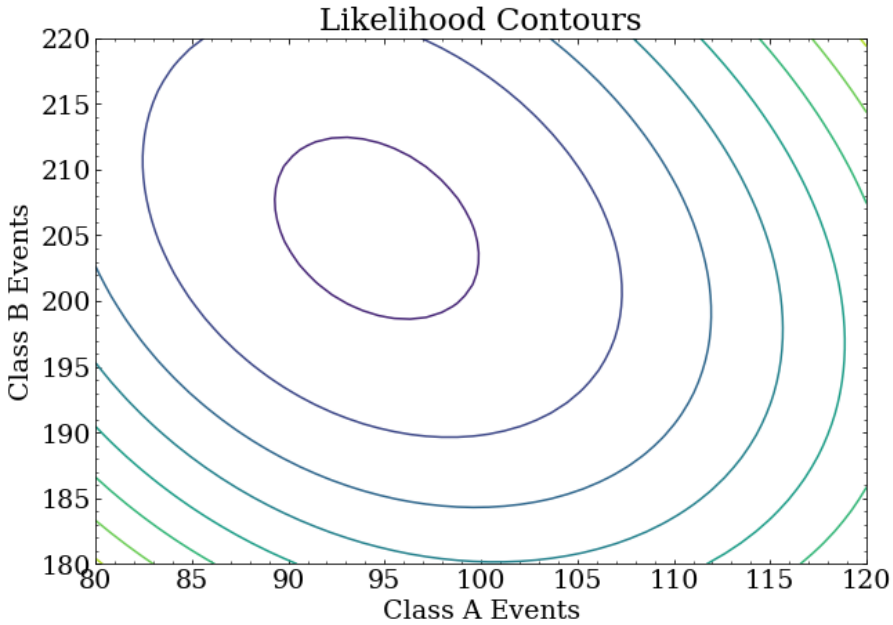


求解极大似然函数，首先需要构造似然函数。本实验中，我们利用泊松分布进行构造。在原理介绍中提到，泊松似然函数包含了指数，幂指数和阶乘等函数，在实际计算中，如果直接采用定义计算，哪怕数据样本量不大，也可能导致内存溢出，因此我们直接使用 Scipy 库提供的 scipy.stats.poisson 函数进行计算，该函数已经优化，可以很好地处理幂指数乘积的问题。

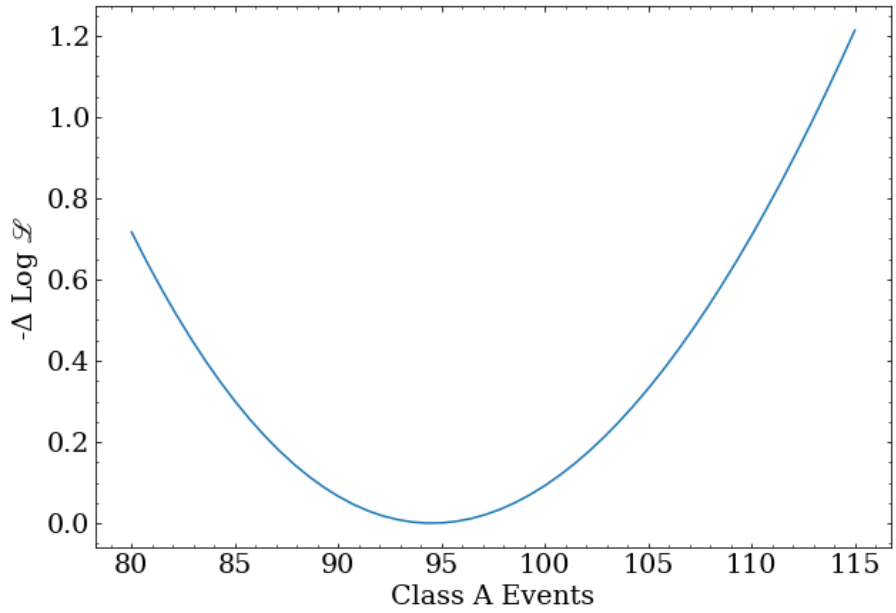
另一方面，似然函数多涉及连乘和 exp 计算，因此对似然函数进行对数化处理，既不影响最大值的位置，也可以提高求解的精度。在本参考案例中，将以　的负对数 (−ln) 取代　作为最优化极值的对象。下面代码中同时保留了直接法和对数法求似然（Likelihood）函数的内容，实验者可以通过修改代码的注释符选择不同的算法，以便对两种方法得到的结果进行对比。以模拟信号 A 为例，作出 100 个 A 事件的图：



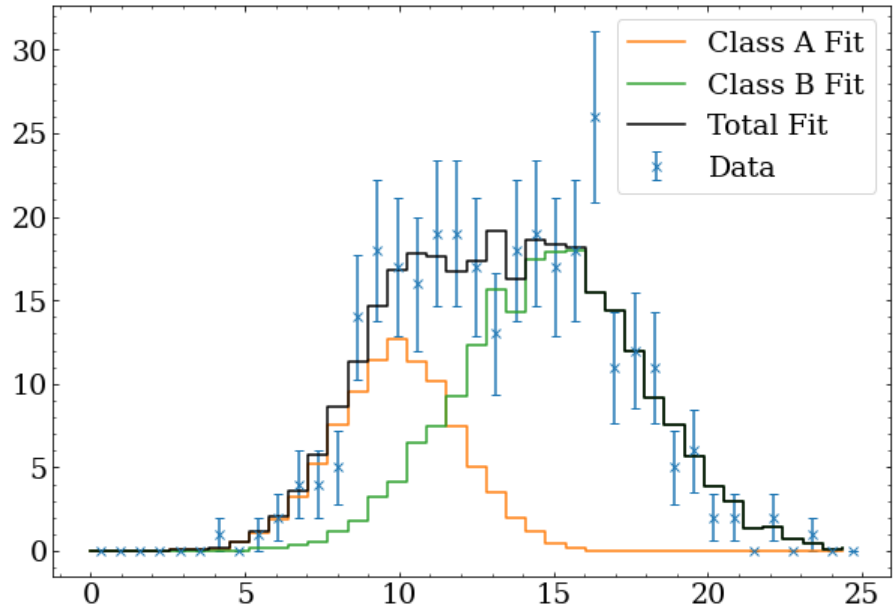
根据本实验的设定，抽样空间是 300，A,B 类事件分别发生了 100 次和 200 次。以 A,B 的待求参数为对象，则可以预期的求解值应该为 $(\mu A^*, \mu B^*) = (100,200)$，利用已经定义的似然函数，可以画出 $\mu A, \mu B$ 参数空间的等高图。



利用 Scipy 库中的最优化工具 scipy.optimize.minimize 函数求负对数似然函数的最小值所对应的参数，即对应似然函数的最大值，似然函数负对数的最小值。



结合前面代码，计算并讨论根据待测数据获得的预期 A 和 B 事件的最大可能分布情况如何，并将误差棒整合到分析结果中。



5 讨 论

5.1 Linux 的使用

Linux 中 terminal 的命令运行要格外注意当前所在的工作目录，在调用已有软件或程序时先写明 export PATH，以免报错。关于在 Linux 中下载软件，除了用 wget 命令获取压缩包之外，还可以安装 miniconda 管理软件。运用指令 conda install [文件名称] 安装指定软件。

5.2 Markdown 文件表格与 LaTeX 表格代码比较

```
#markdown表格
|Syntax|Description|
|------|-----------|
|header|title      |
|MRB   |ZYZ        |

#LaTeX表格
\begin{table}[htbp]
  \centering
    \caption{有机玻璃实验结果}
    \vspace{1em}
  \resizebox{\textwidth}{60mm}{ %第一个大括号为宽度，第二个大括号为高度（60mm）可随机设置，调整到适合该表格的大小为止
    \begin{tabular}{ccccccc}
    \toprule
    时间$\tau/s$ & 温差热电势$V_t/mV$ & 中心面热电势$V_c/mV$ & 温度差$\Delta t/^{\circ}C$ & 中心面温度$t_c^{\circ}C$ & $\Delta V/mV$ & $a\tau/R^2$ \\
    \midrule
    60    & 0.051 & 0.018 & 1.275 & 20.45 &       & 0.0576 \\
    120   & 0.109 & 0.023 & 2.725 & 20.575 & 0.005 & 0.1152 \\
    180   & 0.142 & 0.036 & 3.55  & 20.9  & 0.013 & 0.1728 \\
    240   & 0.162 & 0.053 & 4.05  & 21.325 & 0.017 & 0.2304 \\
    300   & 0.173 & 0.073 & 4.325 & 21.825 & 0.02  & 0.288 \\
    360   & 0.18  & 0.095 & 4.5   & 22.375 & 0.022 & 0.3456 \\
    420   & 0.185 & 0.118 & 4.625 & 22.95 & 0.023 & 0.4032 \\
    480   & 0.188 & 0.141 & 4.7   & 23.525 & 0.023 & 0.4608 \\
    \midrule
    540   & 0.191 & 0.165 & 4.775 & 24.125 & 0.024 & 0.5184 \\
    600   & 0.193 & 0.188 & 4.825 & 24.7  & 0.023 & 0.576 \\
    660   & 0.195 & 0.211 & 4.875 & 25.275 & 0.023 & 0.6336 \\
    720   & 0.196 & 0.235 & 4.9   & 25.875 & 0.024 & 0.6912 \\
    780   & 0.197 & 0.258 & 4.925 & 26.45 & 0.023 & 0.7488 \\
    840   & 0.198 & 0.282 & 4.95  & 27.05 & 0.024 & 0.8064 \\
    900   & 0.199 & 0.305 & 4.975 & 27.625 & 0.023 & 0.864 \\
    960   & 0.2   & 0.329 & 5     & 28.225 & 0.024 & 0.9216 \\
    1020  & 0.2   & 0.352 & 5     & 28.8  & 0.023 & 0.9792 \\
    1080  & 0.201 & 0.375 & 5.025 & 29.375 & 0.023 & 1.0368 \\
    1140  & 0.201 & 0.398 & 5.025 & 29.95 & 0.023 & 1.0944 \\
    1200  & 0.202 & 0.42  & 5.05  & 30.5  & 0.022 & 1.152 \\
    1260  & 0.203 & 0.443 & 5.075 & 31.075 & 0.023 & 1.2096 \\
    1320  & 0.203 & 0.466 & 5.075 & 31.65 & 0.023 & 1.2672 \\
    1380  & 0.203 & 0.488 & 5.075 & 32.2  & 0.022 & 1.3248 \\
    1440  & 0.204 & 0.51  & 5.1   & 32.75 & 0.022 & 1.3824 \\
```

```
    1500  & 0.204 & 0.532 & 5.1   & 33.3  & 0.022 & 1.44 \\
    \bottomrule
    \end{tabular}}%注意这里还有一个半括号
  \label{tab:data}%
  \end{table、
```

Markdown 表格以 | 间隔，而 LaTeX 表格以 & 间隔。除此之外，LaTeX 表格对表格名，行高、行宽和间距等都有更详细的设置。对于制表而言，LaTeX 更专业，但是难以上手；Markdown 容易上手，但是个性化定制有所欠缺。

5.3 似然函数拟合问题

在实验过程中，我发现对于 100 个随机 A 事件，200 个随机 B 事件，似然函数拟合求极值可以轻松完成，但当 A、B 事件增加到千甚至万的量级的时候，拟合时常难以收敛，导致置信区间无法计算。猜测一个是由于事件太多，难以拟合进一个单一函数，另外也可能本身选取拟合函数在大数据时不太合适。

6 结 论

本实验带我们熟悉了 Linux 的架构以及 Linux 语言中 Python 的使用。使用 Numpy 库生成了不同维度的数组并对其中的数进行替换，使用 pandas 库分析了不同班级学生成绩的数据。从数据中发现 2011 级有两位同学数学成绩低于 70 分，2013 级 B 班数学成绩均分最高，英语和物理班级均分最高的分别是 2012 级 A 班和 2013 级 B 班。相比英语，物理的各班标准差较小，说明各班同学在物理上差距较小。从成绩分布图看出理科科目学生成绩较集中，而英语成绩分布较广较平均，从 60 分到 100 分。

在模拟粒子信号实验中随机产生 100 个 A 事件，200 个 B 事件，A、B 事件满足正态分布。采用泊松概率函数求似然函数，A 事件数目和 B 事件数目的置信区间分别为 [93.34,119.34]、[88.38,110.46] 置信区间长度分别为 26.00 和 32.08.

参考文献

[1] Linux，https://en.wikipedia.org/wiki/Linux.

[2] 郭贵鑫，Linux 入门基础培训，国家超算中心培训讲义。

[3] 刘忆智等著. Linux 从入门到精通 (第 2 版附光盘). 北京: 清华大学出版社, 2014 年, ISBN:9787302312727。

[4] Linux 官网. https://www.linux.org/.

[5] [美]Jesse M. Kinder, Philip Nelson 著, 盖磊译. Python 物理建模初学者指南 [M]. 北京: 人民邮电出版社, 2017.

[6] 王国辉，李磊，冯春龙，编著. Python 从入门到项目实践 [M]. 长春: 吉林大学出版社, 2018.

[7] Python For Beginners[OL].https://www.python.org/about/gettingstarted/.

[8] Jupyter Notebook: Markdown+LaTeX[OL]. https://blog.csdn.net/z583636762/article/details/79167130.

[9] Markdown Cheatsheet[OL].https://github.com/adam-p/markdown-here/wiki/Markdown-Cheatsheet.

[10] Numpy 中文网. https://www.numpy.org.cn/.

[11] Pandas 中文网. https://www.pypandas.cn/.

[12] 宗序平主编. 概率论与数理统计 (第 4 版). 北京: 机械工业出版社,2019. ISBN：9787111615422.

[13] (美)John A. Rice 著，田金方译. 数理统计与数据分析 (原书第 3 版). 北京: 机械工业出版社,2011. ISBN：9787111336464.

[14] Numpy Reference, https://numpy.org/doc/stable/reference/?v=20220210114806.

[15] Scipy Reference, https://docs.scipy.org/doc/scipy/reference/.

[16] Matplotlib Reference, https://matplotlib.org/stable/users/index.html.

附录 A

【思考题】

1. 基于 GSL 提供的各类函数库，数值求解 π

π 的数学近似可由贝利-波尔温-普劳夫公式（BBP 公式）得出：

$$\pi = \sum_{n=0}^{\infty} \left[\frac{1}{16^n} \left(\frac{4}{8n+1} - \frac{2}{8n+4} - \frac{1}{8n+5} - \frac{1}{8n+6} \right) \right]$$

C 语言实现方法如下，循环累加得到 π 的数值近似。

```c
#include<stdio.h>
#include<math.h>
int main()
{
double float x=0;
int k,i;
scanf("%d",&k);

    for(i=0;i<=k;i++)
    {
        x=x+(4/(8*i+1)-2/(8*i+4)-1/(8*i+5)-1/(8*i+6))/(pow(16,i));
    }
    printf("%lf\n",x);
 }
```

2. 利用网络查找 fftw 库的函数使用方法

FFTW(The Fastest Fourier Transform in the West) 是一个计算一维或者多维 DFT(the discrete Fourier transform) 的 c 程序子库，支持任意输入长度，可以是实数或者复数数据。软件的基准是：工作在各种平台上且效果优于其他典型的 FFT 软件，与 vendor-tuned codes 也有得一拼 (可能是另一种 FFT 软件) 且 FFTW 是可移植的，不需修改就能在其他架构上直接使用，而 vendor-tuned codes 不行。

一、软件特点

1. 速度快
2. 支持一维和多维变换
3. 支持任意输入长度变换
4. 纯实数输入输出时能快速变换
5. 支持实数奇偶输入数据变换

6. 高效处理多个或者多步变换，如一次性处理多个需要变换的数据列，变换多维数据的某一维，变换多部分向量的某一部分

7. 并行变换，只对支持多线程的机器有效

8. 对使用 C 编译器的平台都是可移植的

9. 开源协议为：GPL

二、资源信息

汇总介绍：https://blog.csdn.net/kissgoodbye2012/article/details/98037800?utm_medium=distribute.pc_aggpage_search_result.none-task-blog-2~aggregatepage~first_rank_ecpm_v1~rank_v31_ecpm-5-98037800.pc_agg_new_rank&utm_term=FFTW%E5%87%BD%E6%95%B0%E5%BA%93&spm=1000.2123.3001.4430

说明文档：http://www.fftw.org/fftw3.pdf

问题和回答：http://www.fftw.org/faq/

和其他 FFT 算法的速度比较：http://www.fftw.org/speed/

和其他 FFT 算法的准确度比较：http://www.fftw.org/accuracy/

对作者的采访：https://www.rce-cast.com/Podcast/rce-73-fftw-fastest-fourier-transform-in-the-west.html

3. 利用 Numpy 练习中初始的数组 arr, 产生一个 6×5×5 的三维数组, 在实验报告中编写一小段程序，实现将 arr 中尾数是 3 的数字替换为所在行的元素的总和。

```
#新建数组
x=array(1:150, dim=c(6,5,5))
#计算尾数为3的数字个数
n=length(which(x%%10==3))
#记录尾数为3的数字坐标
y=which(x%%10==3,arr.ind = T)
a=c()

#循环赋值
for (i in 1:n) {
  a=c(a,sum(x[y[i,1],,y[i,3]]))
  x[y[i,1],y[i,2],y[i,3]]=a[i]
}
```

替换后的数组如下所示：

```
, , 1

     [,1] [,2] [,3] [,4] [,5]
[1,]    1    7   65   19   25
[2,]    2    8   14   20   26
[3,]   75    9   15   21   27
[4,]    4   10   16   22   28
```

```
[5,]    5   11   17   85   29
[6,]    6   12   18   24   30

, , 2

     [,1] [,2] [,3] [,4] [,5]
[1,]   31   37  215   49   55
[2,]   32   38   44   50   56
[3,]  225   39   45   51   57
[4,]   34   40   46   52   58
[5,]   35   41   47  235   59
[6,]   36   42   48   54   60

, , 3

     [,1] [,2] [,3] [,4] [,5]
[1,]   61   67  365   79   85
[2,]   62   68   74   80   86
[3,]  375   69   75   81   87
[4,]   64   70   76   82   88
[5,]   65   71   77  385   89
[6,]   66   72   78   84   90

, , 4

     [,1] [,2] [,3] [,4] [,5]
[1,]   91   97  515  109  115
[2,]   92   98  104  110  116
[3,]  525   99  105  111  117
[4,]   94  100  106  112  118
[5,]   95  101  107  535  119
[6,]   96  102  108  114  120

, , 5

     [,1] [,2] [,3] [,4] [,5]
[1,]  121  127  665  139  145
[2,]  122  128  134  140  146
[3,]  675  129  135  141  147
[4,]  124  130  136  142  148
[5,]  125  131  137  685  149
[6,]  126  132  138  144  150
```

4. 用 pandas 中的数据，利用统计学相关性函数，分析不同学科成绩之间的是否存在 Pearson 相关性？

```
#导入数据
library(readxl)
data=read_excel("Experiment Report/C10/C10.2data.xlsx")
View(data)

#相关系数计算
cor(data[,"Math"],data[,"English"])
          English
Math  0.01157862
cor(data[,"Math"],data[,"Physics"])
         Physics
Math  0.8709206
cor(data[,"Physics"],data[,"English"])
              English
Physics  0.08828891
```

可以看出，相关系数分别为 0.01157862，0.8709206，0.08828891，数学和物理成绩相关性最大，数学和英语成绩相关性最小。

5. 以下是一个随时间衰减的振荡信号数据 signals，请在报告中解释利用快速傅里叶变换 (fft) 实现分析的程序思路。

```
#导入Python库
from matplotlib import pyplot as plt
#在区间0至10平均取500个点
t=np.linspace(0,10,500,endpoint=False)
#时间衰减的震荡信号函数
signals=np.exp(-0.5*t)*np.sin(2*np.pi*t)
#signals=np.sin(2*np.pi*t)
#作图
plt.plot(t,signals)
plt.show()
```

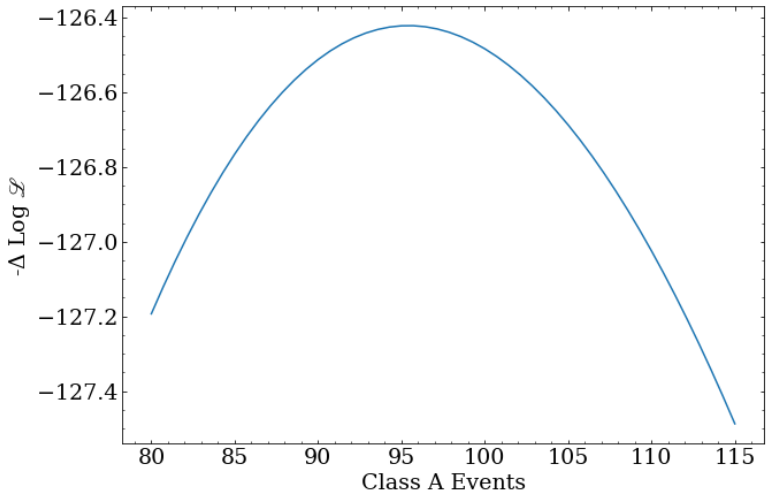
6. 什么是对数似然函数法？分别利用极大似然函数法和对数似然函数法分析数据，比较两种方法所得结果的异同。

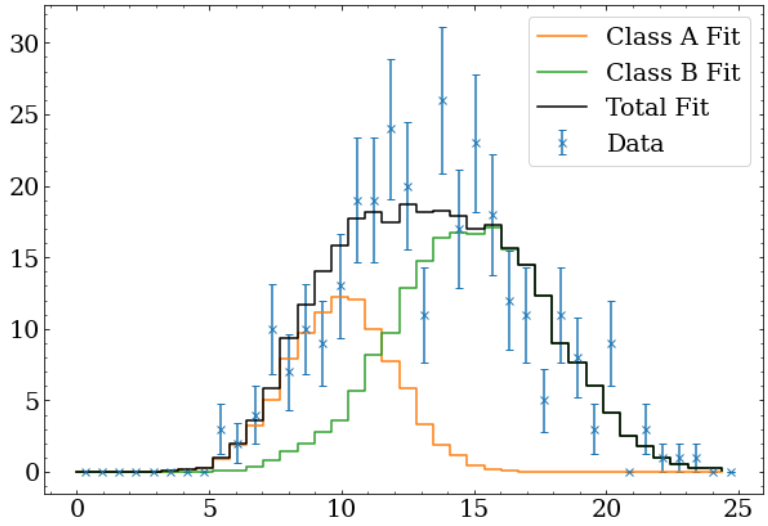
似然函数 L 中若存在连乘、阶乘以及幂函数等函数时，直接求道通常都会非常复杂。对似然函数 L 求负对数则可将函数变得线性，求解偏导消耗计算资源相对较小，所以借助对数似然函数。并且对数函数是单调增函数，所以极大值点仍相同。

极大似然函数法详见实验成果，对数似然函数法代码如下：

```
#求负对数似然函数的最小值所对应的参数
nll_result = opt.minimize(lambda x: -lfn(*x),x0=(50,50),method='Nelder-Mead')
#输出极值对应结果。
print(nll_result)
```

```
#分别定义 a 和 b 的 profile 函数，计算其中一个不变时，另一个的似然函数值
def profile_class_a(nev):
#b 不变时，导致的最小似然值的偏差
    return np.log(-opt.minimize(lambda x: -lfn(nev,x[0]),x0=(50,),method='Nelder
        -Mead').fun) + np.log(-nll_result.fun)
def profile_class_b(nev):
#a 不变时，导致的最小似然值的偏差
    return np.log(-opt.minimize(lambda x: -lfn(x[0],nev),x0=(50,),method='Nelder
        -Mead').fun) + np.log(-nll_result.fun)
#检查 profile 函数的输出结果，以 a 为例
x = np.linspace(80,115,50)
y = [profile_class_a(nev) for nev in x]
plt.plot(x,y)
plt.xlabel('Class A Events')
plt.ylabel('-$\Delta$ Log ${\scr L}$')
```





两种方法原理相同，在算法方面略有差异。总的来说，两种方法的差异一在计算资源的占用量不同，二在处理后似然函数求极大还是极小值的差别。

7. 调整 A、B 随机赝事件的比例，但仍采用对数似然函数法进行分析，探讨保持相同置信水平的情况下，置信度区间如何变化。

表 1: 实验结果

实验次数	A 事件数	B 事件数	A 置信区间	区间长度	B 置信区间	区间长度
1	50	250	$58.58^{+11.53}_{-10.72}$	22.25	$241.42^{+17.89}_{-17.13}$	35.02
2	100	200	$105.97^{+13.37}_{-12.63}$	26.00	$194.03^{+16.43}_{-15.65}$	32.08
3	150	150	$163.74^{+15.03}_{-14.34}$	29.37	$136.26^{+14.13}_{-13.30}$	27.43
4	200	100	$194.33^{+15.98}_{-15.31}$	31.29	$105.67^{+12.93}_{-12.06}$	24.99
5	250	50	$248.75^{+17.13}_{-16.46}$	33.59	$51.25^{+9.66}_{-8.72}$	18.38

从表中看出，当 A、B 事件总量一定时，事件数越多的事件类型区间长度越长，估算误差越大。就像表中随着 A 事件增多，A 的置信区间不断增长，B 不断减短。