**实训项目指导书**

项目名称： 差分隐私数据合成基础实验

申报年度：

责任教师：

团队组成：

# 项目介绍

本项目需要学生首先学习经典的差分隐私数据合成算法MWEM和Python语言的基础知识。此后，学生需要在提供的Python代码框架的基础上，根据对算法各部分的描述完善已有代码，完成对MWEM算法的实现。项目要求学生最终提交的算法实现可运行于Ubuntu系统中，能利用给定的实验数据集生成符合算法要求的合成数据。同时，学有余力的同学在此基础上可对原始算法进行一定的优化。

通过对本项目的学习，学生可理解差分隐私的基本理念，并对数据合成的基础流程有一定的了解。

# 基础知识

## 差分隐私

差分隐私（Differential Privacy）最早于2006年由Dwork等人提出。其通过对原始数据添加随机噪音，确保数据集在输出查询结果时受单条记录改变的影响始终低于某个阈值，从而使攻击者无法根据查询结果的变化判断单条记录的更改或增删，同时整体的数据特征仍然可得到保留。在数据脱敏领域中，与在其出现之前较为常用的K-匿名和L-多样化方法相比，差分隐私方法可有效抵御差分攻击，对个体隐私的保护能力更强。

具体来说，设原始数据集为，我们对其上的任意一条记录进行修改或删除，或是增加一条新记录后，就得到了其相邻数据集。此时，对于一个任意随机化算法，若其满足：

我们称该算法是满足差分隐私的。即，该算法作用于任何相邻数据集时，得到特定输出的概率是相近的。

其中，是差分隐私的隐私预算。一般来说，隐私预算越小，对数据的隐私保护程度越大，但是数据的可用性越差；隐私预算越大，数据的可用性越好，但是隐私保护能力越差。通过对隐私预算的设计，可实现差分隐私算法在隐私保护程度和数据可用性之间的权衡。

## 差分隐私数据合成

现有的差分隐私数据发布方案按照查询的操作方式可分为两种，即交互式差分隐私和非交互式差分隐私。

其中，交互式差分隐私模型是由用户向数据管理者提出查询操作请求，数据管理者再根据查询请求对数据集进行操作，并对结果进行扰动后反馈给用户。该模型能够保证查询者无法知晓数据全貌，从而保护数据集中的给个体隐私。

基于该模型的方案目前已大量应用于诸如个性化广告推荐、社交网络趋势分析、基于用户兴趣的商品推荐等场景中，在银行、车企、零售等行业中提供保护隐私的用户洞察服务。一般来说，交互式模型方案多与数据库引擎间集合紧密，以引擎中间件的形式处理查询请求，返回满足差分隐私的查询结果。由于该模型并未对数据集本身进行处理，其对数据集的动态更新支持较好，适合于各类数据更新频繁且分析者对数据新鲜度敏感的场景。

然而，虽然交互式模型目前已有较为成熟的应用，且能够有效抵御差分攻击，但由于每次查询均需要消耗一定量的隐私预算(Budget)，当隐私预算消耗完毕后，就无法再继续查询，其仅能支持有限次查询。此外，交互式模型即使多次进行同一个查询，其查询结果由于噪音扰动也可能是不同的，不具有一致性，用于数据分析时的效果欠佳。

针对交互式差分隐私模型的查询支持次数有限及查询不具有一致性的问题，目前有许多方案使用非交互式差分隐私模型来支持查询。在该模型下，数据管理者对原始数据集进行处理，并发布一个原始数据的“合成净化”版本，即合成数据集，因此该类方案也称为差分隐私的数据合成方案。查询用户可对该合成数据集自行进行所需的查询操作。

该模型方案更适合于需要进行大量查询的场景和数据集处理与后续存储分离的场景。当数据使用方有离线查询的需求时，非交互式模型生成的合成数据集可支持与原始数据集所支持的所有查询。因此，非交互式模型非常适合于敏感领域大数据分析(如医疗，金融，人口信息等)、数据分析外包、多方共用数据分析等场景下的保护隐私的数据分析服务，其发布后无需数据发布者进一步参与的特性使其能适应多种类型的系统。

由于该方法的加噪过程完成于“合成净化”阶段，对合成后的数据进行查询满足差分隐私的后处理鲁棒性。其在查询时不再消耗隐私预算，故其可支持无限次查询，同时对于相同查询的响应结果具有一致性。因此，数据合成方案是更适合于在海量数据分析过程中保护个体数据隐私的方案。

## 指数机制

如前所述，差异隐私是对随机计算的一种约束，即计算不应透露输入中存在的个别记录的细节。一个满足差分隐私的机制应在任何两个足够接近的数据集上表现得几乎相同。

那么，设想一个场景：数据集的记录来源于一个抽象域，其可被描述为一个来源于的函数，它被描述为一个从到自然数的函数，那么即可表示在数据集中的频率（出现的次数）。我们利用来描述表示频率差异的绝对值之和（要增加或删除多少条记录才能将A变为B）。

指数机制(Exponential Mechanism)是一个差分隐私化的机制，其可被用于在一组离散的备选项组成的集合中选择最佳匹配项，而该“最佳”是由一个与每个备选项和基础的秘密数据相关的函数所定义的，并非一个归一化的指标。

形式化地，对于一组备选项组成的集合，我们需要设定一个质量打分函数,其中关于和的函数可被解释为数据集B的结果r的质量。为了保证差分隐私，该函数还必须满足以下的稳定属性：对于每个结果来说，的最大值为.此时，指数机制从分布中选择，其中满足：

在上式中，符号意为该式的左值和右值成正比。因此，根据上式我们可以看出，该机制选择结果的概率是以指数方式偏向其质量分数的。总地来说，指数机制需要的时间与可能的结果数量成线性关系,对于每个均产生一个.

## MWEM-基于乘法权重的指数机制算法

MWEM(Multiplicative Weights Exponential Mechanism)是一种简单高效的数据合成发布算法，其迭代地执行指数机制-乘法权重更新，不断地优化一个对真实数据分布的估计，以期在合入噪音的同时，尽可能地使合成数据关于查询集合的结果与原始数据一致。

具体来说，MWEM首先生成一个对给定的真实数据集的估计模型(approximation)，此后进行一个多轮迭代过程。在每轮迭代中，首先使用选择机制选出使用近似数据模型对一给定查询集进行响应时误差最大的查询，然后利用真实数据集对该查询的响应来改进这个估计，以让此估计模型能够更好地反映查询的真实答案，从而迭代地提升由该估计模型产生的私有数据集在响应给定查询集时的准确性。

MWEM机制的每次迭代均可分为三步，即选择-测量-乘法权重更新。选择过程使用指数机制来选定本轮所用估计模型对给定查询集响应最差的查询，然后使用原始数据集响应该查询，并利用测量机制为该响应添加拉普拉斯噪音。最后，利用乘法权重机制来进行更新。

乘法权重方法(Multiplicative Weight)在计算机科学的许多领域都有应用。MWEM机制中使用的是Hardt和Rothblum提出的方法，该方法反复改进一个数据集的近似分布，以期能够更好地反映真实分布的特征。乘法权重的基础逻辑是，若我们发现一个查询在真实数据上响应比它在近似数据上的响应大得多，我们应该提高对错误率有正向贡献的记录的近似权重，降低对错误率有负向贡献的记录的权重。如果真实数据的响应远小于近似数据的响应，应进行相反的操作。

接下来，我们将形式化描述MWEM使用的乘法权重算法。若和是两个位于记录域上的分布，其中是一个用于估计关于查询的真实分布的合成分布。此时，乘法权重更新规则会倾向于通过以下方式更新对每条记录的权重:

其中，正比符号表明了近似值在进行缩放后应进行重新规范化。Hardt和Rothblum的研究表明，每次应用这一规则时，和之间的相对熵(relative entropy)都会以的形式加法级别减少。因此，只要我们能继续找到结果不一致的查询，我们就能继续对估计分布进行改进。

## Python语言

Python是一种高级、通用、解释型、面向对象的编程语言。它由Guido van Rossum在1989年创造，最初是为了在荷兰国家数学和计算机科学研究所编写一个新的脚本语言而设计。作为一种解释型语言，其代码不需要预先编译，可以在Windows、Mac、Linux等任意安装了Python解释器的操作系统上实现跨平台运行，且大多数情况下无需对代码所要运行的平台进行针对性地改变，特别适合于需要跨平台或是代码大量分发的场景。

Python语言在数据科学和机器学习领域拥有广泛的应用。基于活跃的开发社区，目前Python拥有大量用于数据科学领域的函数库，例如用于数据分析的pandas库、用于科学计算的numpy库、用于机器学习的scikit-learn库等。借助其拥有的大量函数库和开发框架，开发者可以快速地构建各种应用程序，且代码可读性强，易于维护。

# 开发环境

本部分将对项目开展所需的开发环境、开发工具、环境搭建方法等进行介绍。

## 工具或源码介绍

### 开发环境

系统：Ubuntu 18.04.6

虚拟机：VMWare Workstation Pro 17，VMWare Player 17或VirtualBox 7

语言解释器：Python 3.6.9及以上

### MWEM-Easy代码框架

本项目提供的代码框架MWEM-Easy是参考MWEM原作者提供的Julia版本源代码的一个Python语言MWEM机制实现。其基于常用的Python库numpy和matplotlib完整设计了MWEM的机制的流程，即初始化阶段、最差查询选择阶段、乘法权重更新阶段，并提供了误差分析工具和直方图绘制工具。

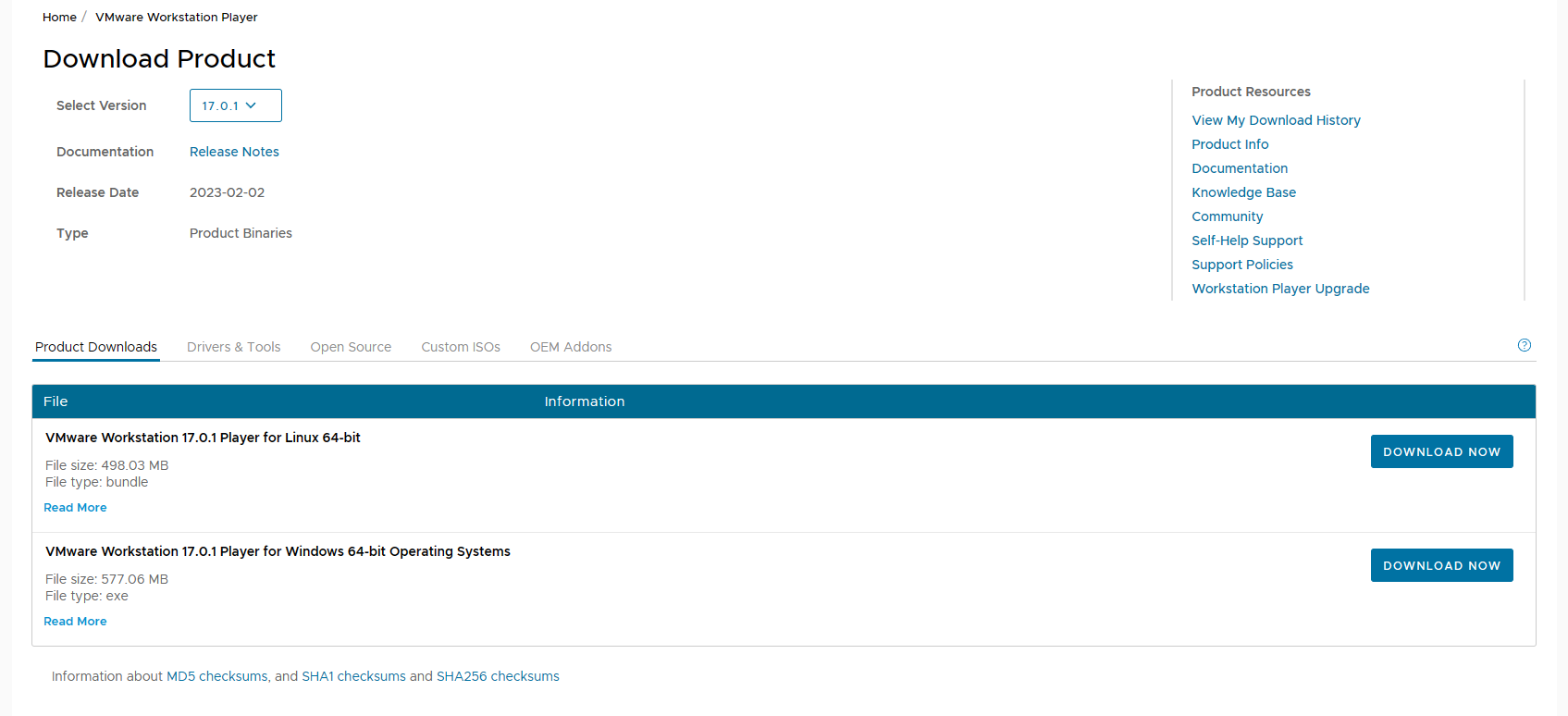
## 开发环境搭建

### 虚拟机和虚拟系统初始化

1.获得Ubuntu 18.04.6镜像。建议点击[这里](https://mirrors.tuna.tsinghua.edu.cn/ubuntu-releases/18.04/ubuntu-18.04.6-desktop-amd64.iso)从TUNA镜像源下载以获得最好的下载速度。使用浏览器直接下载或复制连接后使用任意下载工具下载，若已有18.04任意版本镜像则可直接使用。

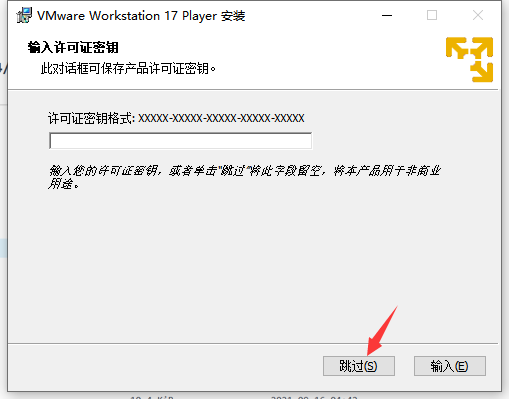


2. 点击[这里](https://customerconnect.vmware.com/en/downloads/details?downloadGroup=WKST-PLAYER-1701&productId=1377&rPId=100675)获得并安装虚拟机软件。图中以windows下安装VMWare Player为例，根据自己实体机系统选择合适的host版本下载即可。若已安装有VMWare Workstation Pro或VirtualBox，可直接使用。



3.安装VMWare Player。在安装完成提示输入许可证时，点击“许可证”，然后点击“跳过”完成非商业授权许可。





4.打开VMWare Player，选择创建新虚拟机

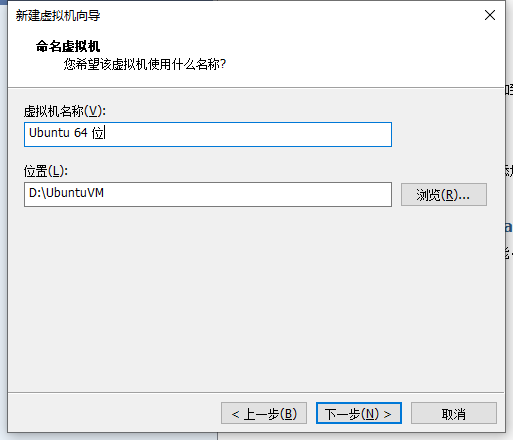


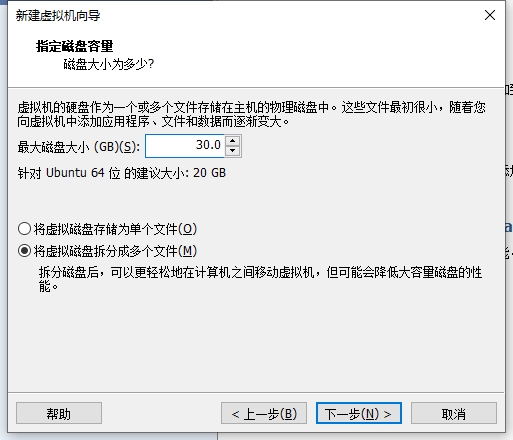
5. 选择安装来源为刚下载好的镜像，点击下一步，提供简易安装信息，完成后再点击下一步。

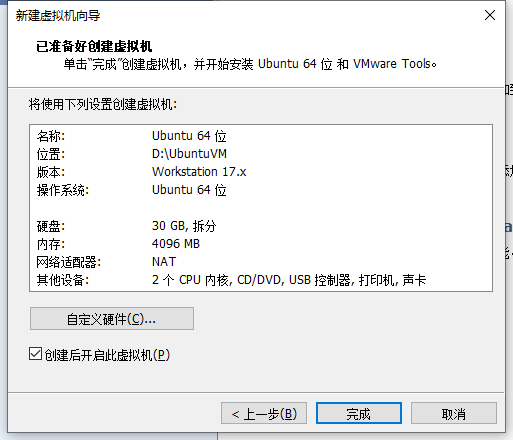




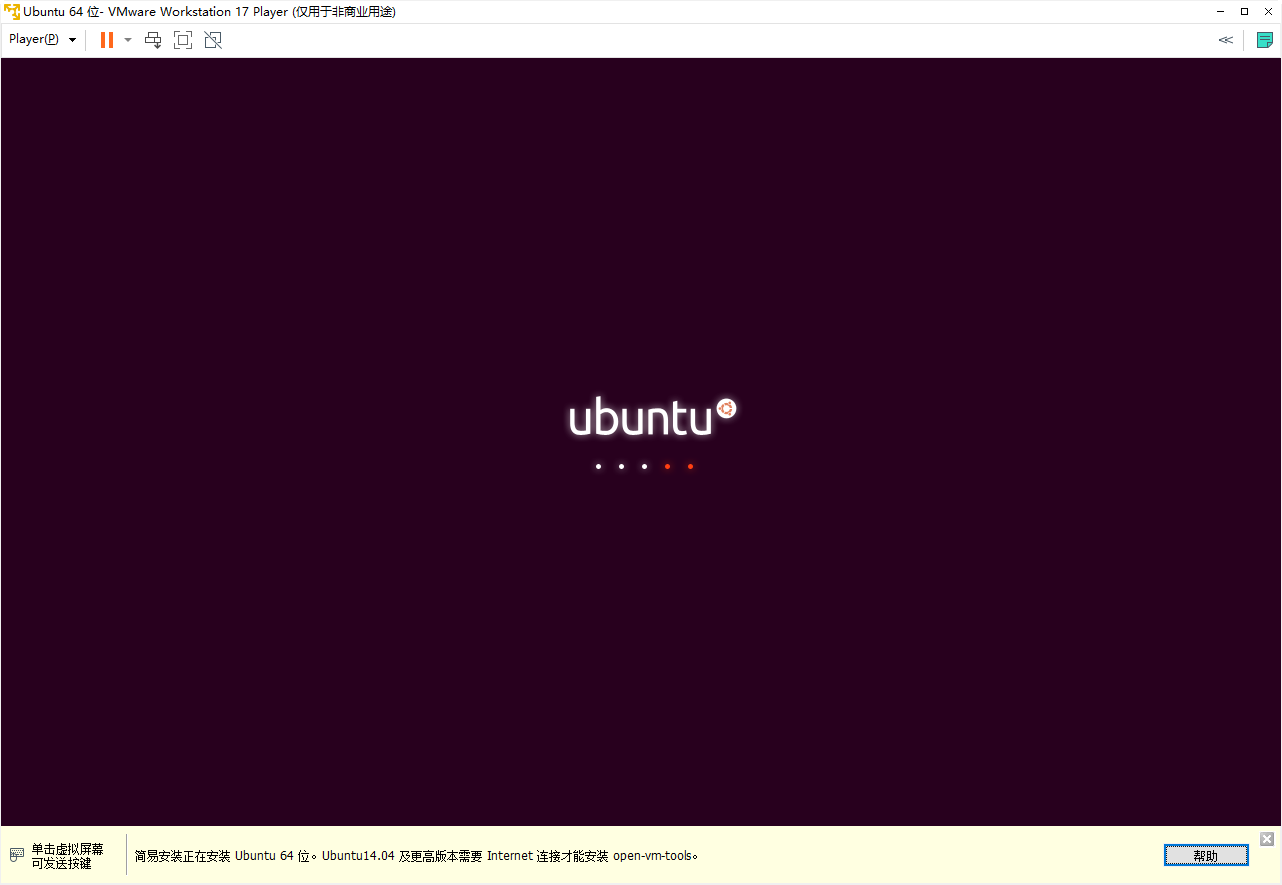
6. 选择合适的虚拟机位置，点击下一步对磁盘选型。VMWare Player默认分配30GB磁盘和4GB内存，以及2个CPU核心，一般足够实验使用。如果需要调整，可点击高级设置。



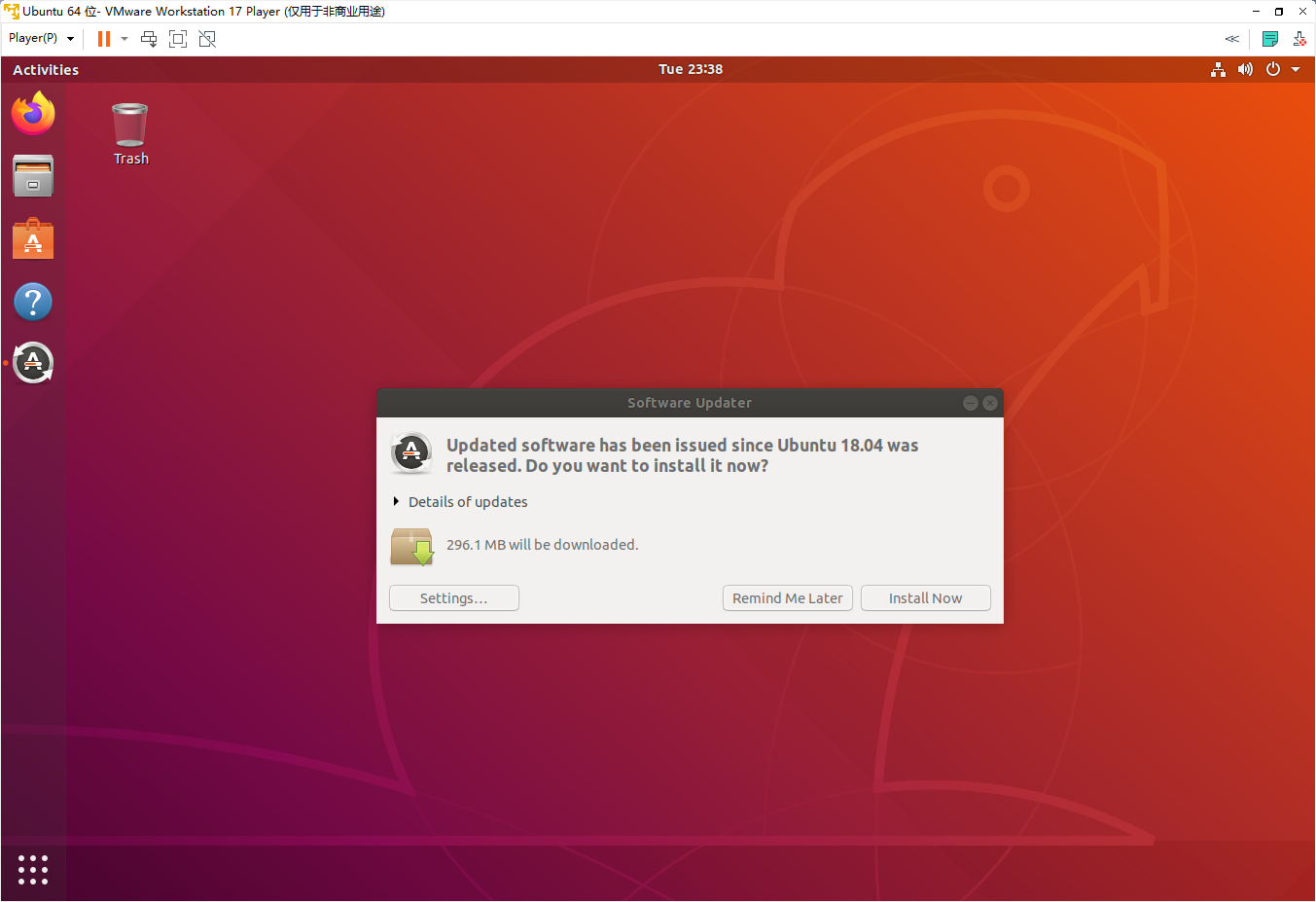




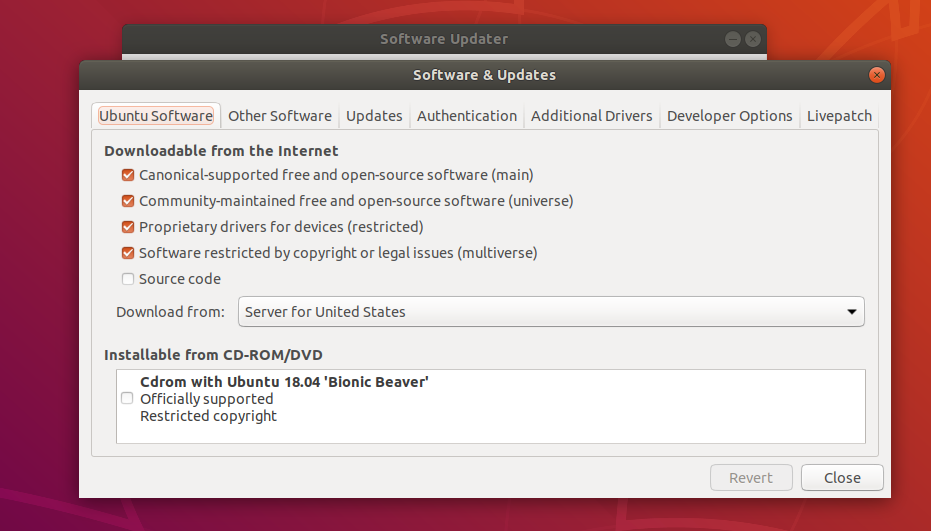
7. VM Player会自动进入安装进程，等待安装完成即可。

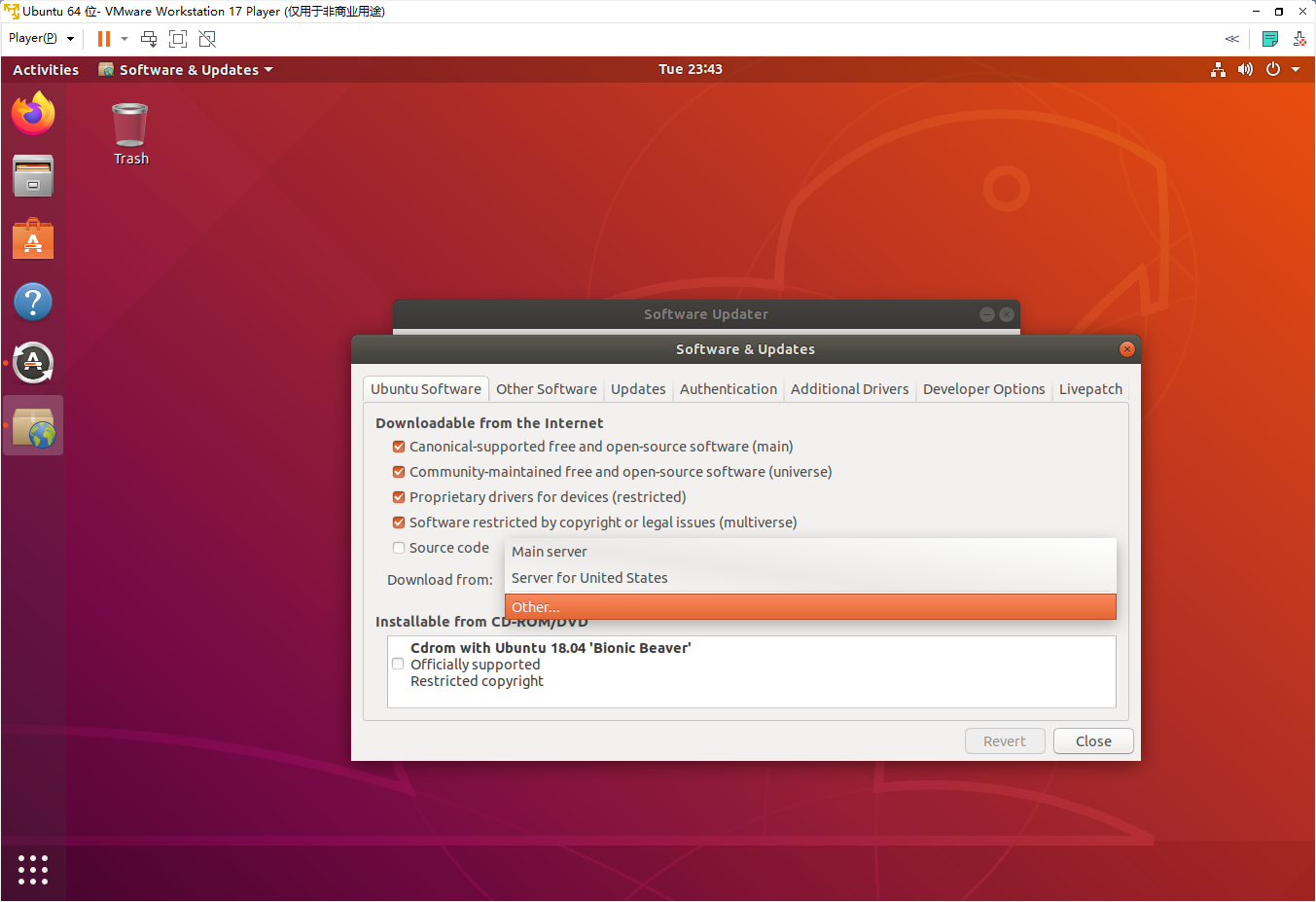


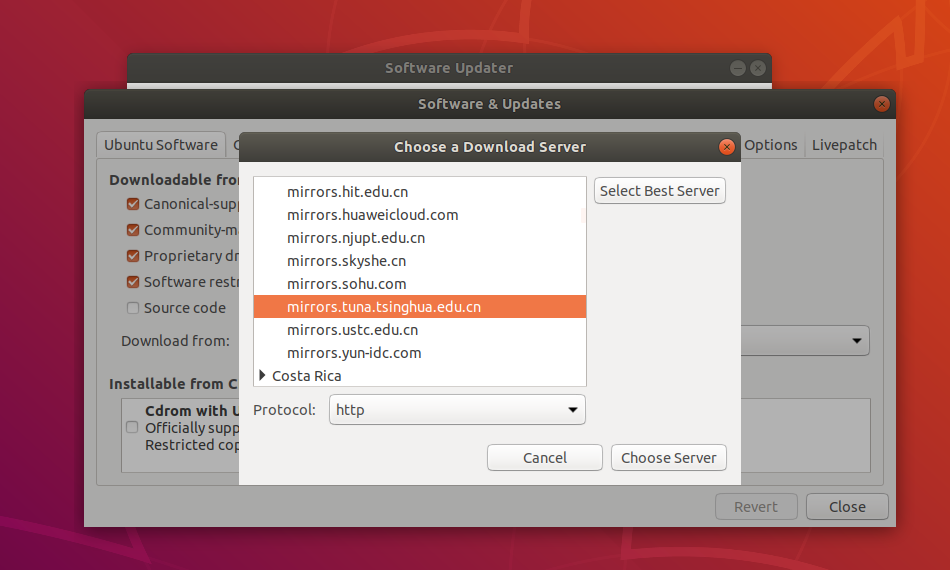
8. 安装完毕后使用预设的用户名和密码登陆到虚拟机中。此时可能会提示更新软件。在更新前，我们先设置软件源为国内软件源以加快下载速度，此时先点击“Settings”。



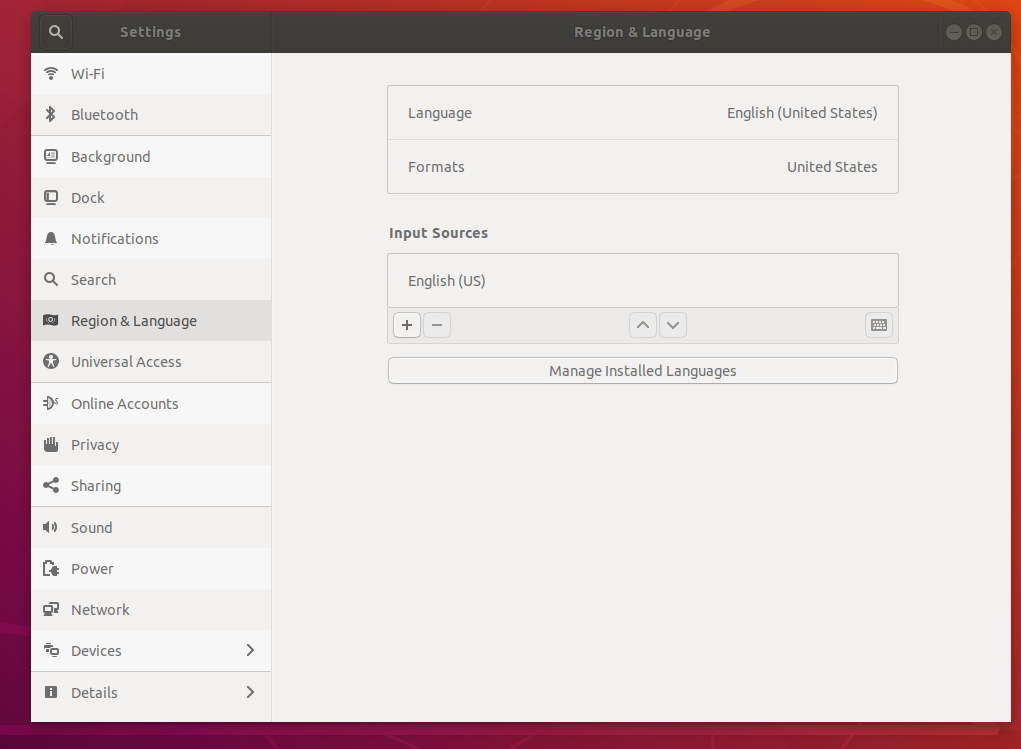
9. 点击Ubuntu Software选项卡，将Download From项目的下拉选单切换为“Other”，并在弹出的窗口中找到位于China的软件源服务器。此处我们选择较近的清华大学软件源(mirrors.tuna.tsinghua.edu.cn)，并点击“Choose Server”，输入密码完成更换。在关闭窗口时会提示需要reload，点击reload即可。

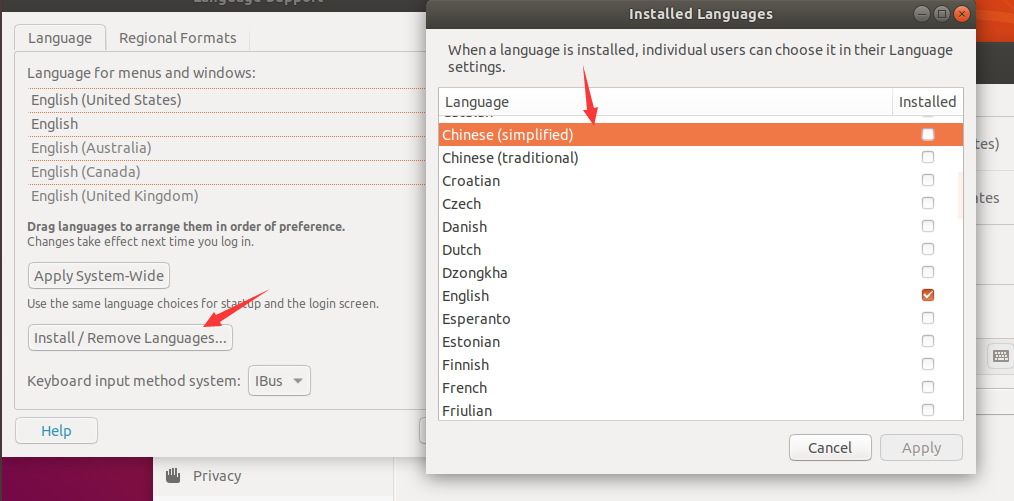






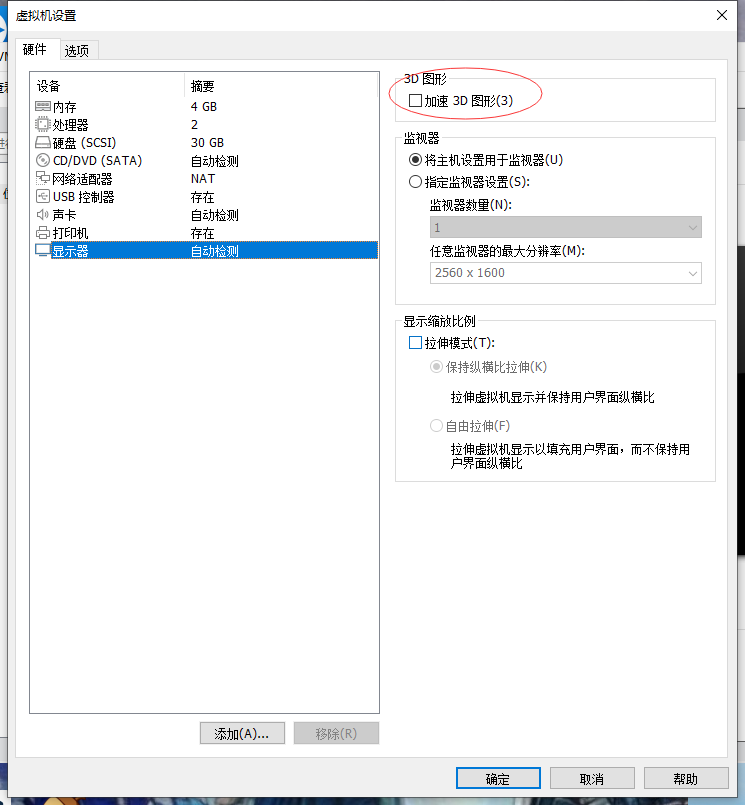
10. 等待源重载完毕后，此时可以点击install now进行软件更新了。接下来的步骤将继续以英语环境进行演示，如有需要可利用左下角Applications-Settings-Region & Language选项卡，使用Manage installed languages来加入中文包，然后切换到中文。





11. 额外提示：如果虚拟机系统呈现输入时卡顿的情况，可以尝试关闭虚拟机后，在VMWare Player首页点击“编辑此虚拟机”，切换至“显示”子项，取消勾选“加速3D图形”，一般即可解决问题。



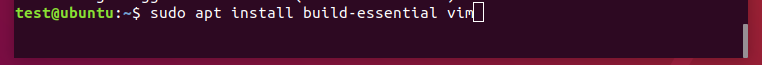


### 开发环境搭建

1. 在桌面上右键呼出菜单，点击Open Terminal，在打开的终端环境中输入

sudo apt install build-essential vim

并回车，输入密码后安装基本开发环境和vim文本处理工具。



2. 安装完毕后，使用指令

python3 --version

检查系统集成的Python版本。一般而言，18.04.6版本的ubuntu是集成有Python 3.6.9版本的，足够满足实验需求。如果没有的话，可使用指令

sudo apt install python3.10

来安装最新的Python3.10版本。



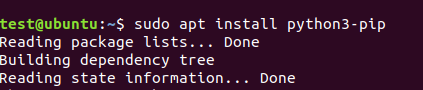
3. 系统集成的Python一般是不自带包管理工具pip的，需要手动安装。输入指令

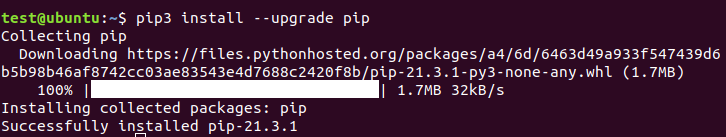
sudo apt install python3-pip

来完成管理工具安装，完成后输入指令

pip3 install --upgrade pip

来更新该工具。





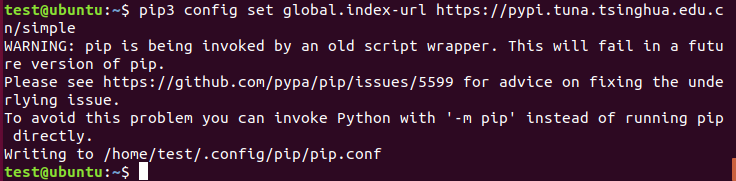
注：如果迟迟没有反应，应该是因为网络出现了问题，请使用以下指令作为替代：

pip3 install pip install -i https://pypi.tuna.tsinghua.edu.cn/simple --upgrade pip

4. 对pip工具进行换源。输入指令：

pip3 config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple

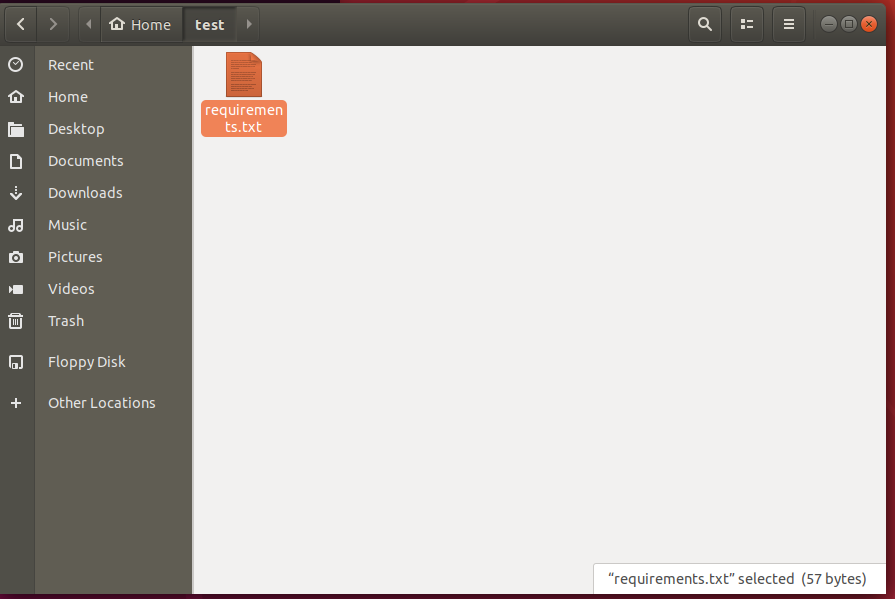
完成换源。

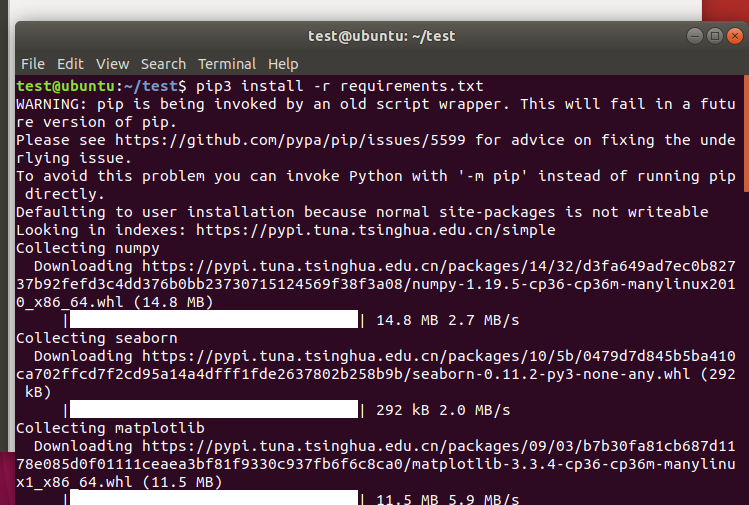


5. 作为脚本语言，Python程序的运行依赖于各式各样的扩展库，这些扩展库可为我们提供诸如科学计算、结构化数据处理、直方图绘制等高级功能。在本次实验的源码包中，我们已为各位同学整理了样例代码可能用到的依赖库，放置于requirements.txt之中。如要使用，将该文件复制入虚拟机中，在其所在的文件夹内右键，点击open in terminal在该文件夹内呼出终端。使用指令

pip3 install -r requirements.txt

即可安装该文件内提及的所有依赖库。在最终提交本项目文件时时，也请整理出属于你的requirements.txt。





注意：由于我们使用了ubuntu的apt根据来安装pip，各位同学之后在网上看到需要使用pip来安装的依赖库，**都必须将pip指令换为pip3指令来运行**，否则会提示找不到pip。

到此，虚拟机、虚拟系统和python环境就配置完成了。

# 项目实施

本项目的主要实施过程分为Python基础知识学习、差分隐私基础知识学习、MWEM方案学习和代码补全复现四个阶段。

## Python基础知识学习

在本阶段，学生可通过各类教程网站对Python语言进行整体了解，并达到以下目标：

·了解Python语言的基本逻辑，对Python运行的基本流程有所了解，能够使用pip工具解决相关依赖问题

·能够使用for、while等循环体实现基本循环结构

·能够灵活运用dict，list和tuple等Python常用数据结构

·能够掌握基本的IO功能，如file类和print函数，实现对文件的读取和写入功能

具体教程可参考如下网站：

[1] 菜鸟教程：[Python3 教程 | 菜鸟教程 (runoob.com)](https://www.runoob.com/python3/python3-tutorial.html)

[2] 廖雪峰的Python教程：[Python教程 - 廖雪峰的官方网站 (liaoxuefeng.com)](https://www.liaoxuefeng.com/wiki/1016959663602400)

## 差分隐私基础知识学习

在本阶段，学生应通过各类教程、网站、论文对差分隐私机制进行整体了解，并达到以下目标：

·了解差分隐私的基本定义和差分隐私的隐私观，知晓差分隐私保护的内容和不能保护的内容

·了解拉普拉斯机制和指数机制的原理，理解其推导过程

·了解隐私预算对差分隐私机制输出的影响

具体可参考以下文献和网站：

[1] Dwork C, Roth A. The algorithmic foundations of differential privacy[J]. Foundations and Trends® in Theoretical Computer Science, 2014, 9(3–4): 211-407. 差分隐私的经典书籍《差分隐私算法基础》，完成该部分目标至少应阅读前40页内容。

[2]《差分隐私算法基础》的部分中文翻译内容：<https://zhuanlan.zhihu.com/p/522721297> 为满足本部分目标，应至少学习至本专栏的第九节“(9)差分隐私算法基础——基础实现与组合定理4：指数机制”

## MWEM方案和MW机制基础

在本阶段，学生应通过学习MWEM方案和其参考的MW机制原始方案达到以下目标：

·了解合成数据的基础内容和通过MWEM方案进行数据合成的优势

·学习MWEM方案的基本构造，理解MWEM方案的运行逻辑和实现流程

·学习原始MW机制的乘法权重相关内容，理解MW机制的内在逻辑

具体可参考如下内容：

[1] Hardt M, Ligett K, McSherry F. A simple and practical algorithm for differentially private data release[J]. Advances in neural information processing systems, 2012, 25. MWEM方案的具体论文

[2] Hardt M, Rothblum G N. A multiplicative weights mechanism for privacy-preserving data analysis[C]//2010 IEEE 51st annual symposium on foundations of computer science. IEEE, 2010: 61-70. MWEM方案中使用的具体MW机制，请结合该机制和原始MWEM方案一起来思考如何补全MWEM-Easy的代码。

[3] 原作者提供的代码，以Julia语言编写，可用于参考实现：

<https://github.com/giladroyz/MWEM-project>

## MWEM-Easy算法框架的补全和复现

### 基本要求

本次实验的实验内容是完成对MWEM算法的一个简单版本的复现。具体来说，本实验需要：

1. 分步实现完整的MWEM机制，包含查询发起，误差测量，最差查询选中，乘法权重更新等，并能迭代地对已有观测进行优化。

2. 该机制能运行于一个一维的数据集上，对使用该数据集进行统计后分桶(bin)形成的直方图桶数据进行数据合成处理。

3. 合成后的数据与原始数据在响应同一组查询时误差尽可能小的同时，隐藏数据本身的具体特征。

4. 使用matplotlib库，为原始数据和合成数据生成直方图，以图的形式展示两者的差异。

### 提高项目

在4.4.1的基础上，完成二维MWEM算法的代码补全和复现，并生成合适的图像进行分析展示。

### 文件构成

本次项目所提供的文件主要包括两个文件夹和三个代码文件。其中，Dataset文件夹中存放有本次项目使用的、从childMentalHealth数据集中剥离出的两列数据AGE和DIFTHELP，规模为1013185条。本次项目的基本要求是能对其中AGE数据使用MWEM算法进行处理并产生合成数据。该数据集内的DIFTHELP用于学有余力的同学设计二维MWEM算法实现时使用。此外，该文件夹内还提供了一个规模为3519条的小型数据集（没有1M后缀的）用于测试。

Results文件夹初始为空，用于存放所生成的直方图的图片。对于设计了二维MWEM算法的同学，还应提供二维算法的相关统计图像。

三个代码文件中，当只完成项目基本要求时，只需要使用mwem.py和errortools.py。其中，mwem.py为主代码文件，其包含了MWEM算法复现的整体流程，各部分函数的运行逻辑和函数的输入输出已经写好，需要同学们根据运行逻辑和步骤对个别函数进行完善，最终得到结果。errortools.py文件为辅助文件，包含了算法误差评估的相关函数和查询实际执行函数。最后，mwem\_2D.py文件是二维MWEM算法的函数框架，供学有余力的同学挑战和提高。

### 函数参数表及主要功能

此处只列出参数表和大致功能描述，具体算法流程请参考框架内的注释。

1. \_\_init\_\_.py用于为同文件夹内的代码文件之间提供互相引用，请保持留空。

2. 在errortools.py中:

**Evaluate (query, data)**

query:指定的查询

data:运行查询的数据

Evaluate函数用于执行查询。其基本逻辑为：对于传入的查询(x,y)，其累加传入的数据中，值在x和y区间(含x和y)的查询。例如，对于一组有五个桶的数据[2,2,3,4,4]，查询(2,3)会累加位于第二位和第三位的数据，即其结果为2+3=5.

**maxError (real, synthetic, Q)**

**meanSqErr (real, synthetic, Q)**

**minError (real, synthetic, Q)**

**meanError(real, synthetic, Q)**

real:真实数据

synthetic:合成数据

Q:查询集合

上述四个函数分别用于计算传入的两数据集间的最大差异，均方差，平均差异和最小差异。

3. 在MWEM.py中:

**ExpM (B, A, Q, eps)**

B:原始数据

A:合成数据

Q:查询集

T:迭代次数

eps:隐私预算

函数使用指数机制来选定一个查询。其首先测量对原始数据集和合成数据运行同一个查询时其之间的误差作为打分函数，根据各查询得分的多少计算出每个查询被选中的概率，之后基于这些概率随机抽取一个查询。

**Laplace(sigma)**

sigma:隐私预算。

利用numpy的随机数模块来生成拉普拉斯随机数作为噪音。本函数直接给出，无需完善。

**MultiplicativeWeights(A, Q, measurements, repetitions)**

A:合成数据

Q:查询集

measurements:已观测过的查询

repetitions:重复次数

本函数实现了乘法权重更新。其维护一个循环体，每次循环开始时，其先对measurements进行洗牌，然后进行乘法权重的更新流程。具体作用请参考代码文件内随附的注释，下同。

**transformForPlotting(Histo, B)**

Histo: matplotlib.plot方法所需的格式

B:输入的数据集

转换查询为生成直方图所用的Plot。

**queryToBinary(qi,length)**

qi:输入的查询

length:数据集属性长度

转换查询为二进制形式，用于MW机制使用

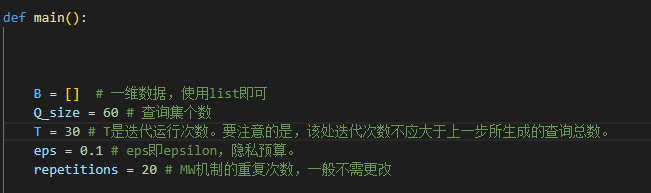
**histoDraw(real,syn,B)**

用于直方图绘制。

### MWEM.py实际运行结果参考

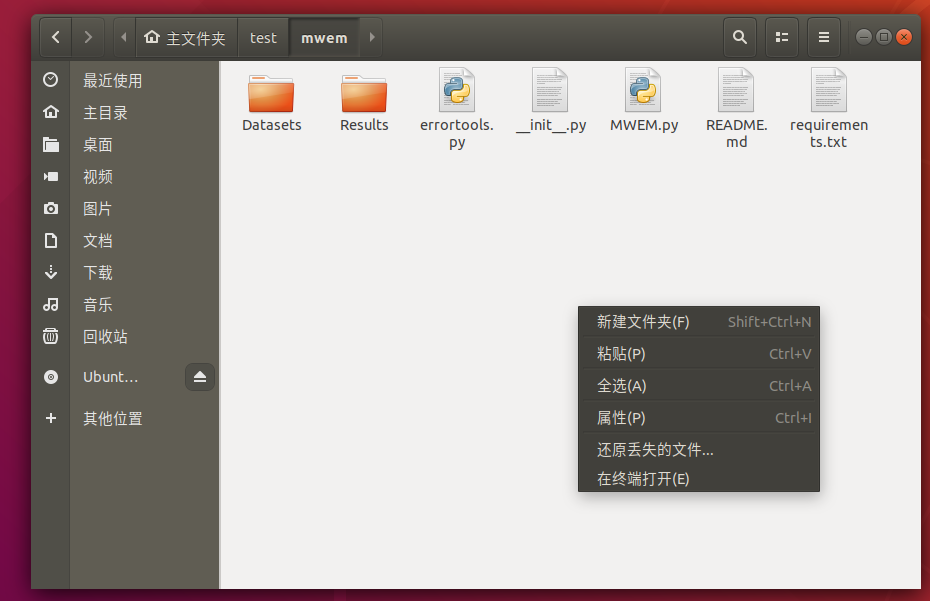
本部分将展示补全后的代码的实际运行结果和各部分函数应达到的目标。注：本部分使用隐私预算eps=0.1，轮次T=30。

1.在main函数中，正确设置各项实验参数并保存代码文件



2．在代码文件所在文件夹右键，选择“在终端打开(Open in terminal)”,输入以下代码来启动运行：

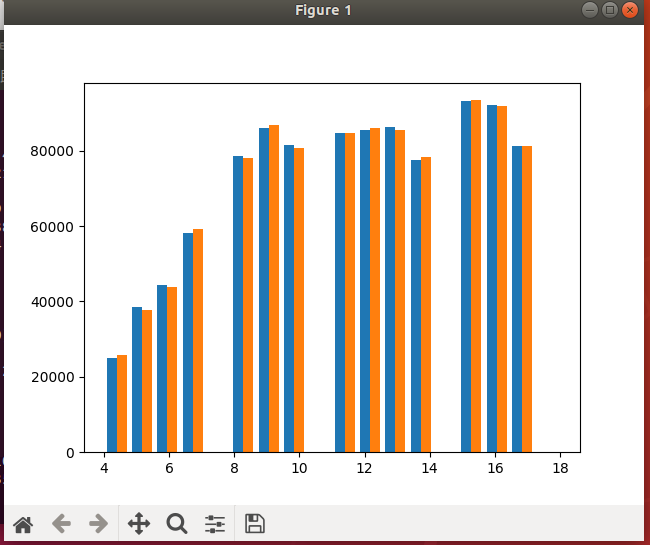
python3 ./MWEM.py



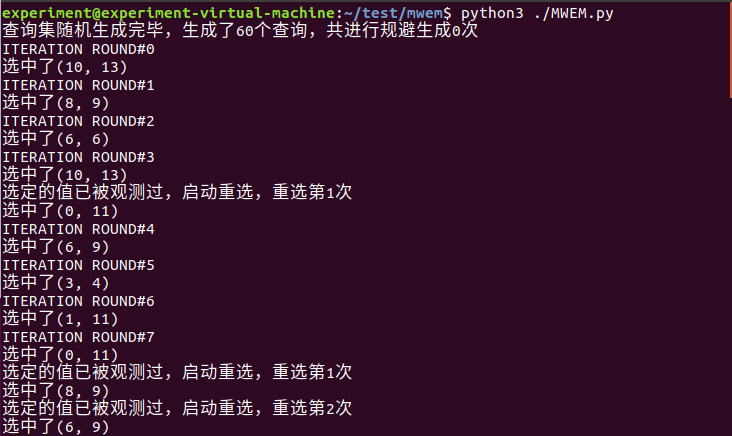


3. 你将可以依次看到以下内容：

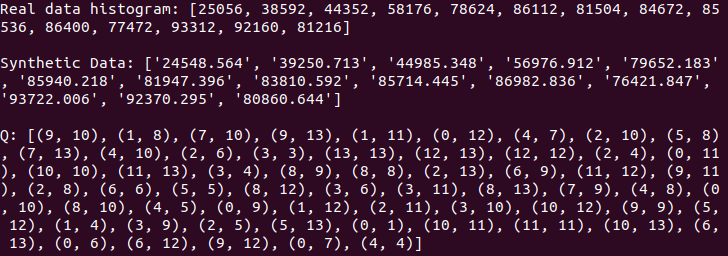
首先，会弹出自动生成的原始数据集和合成数据集的直方图：



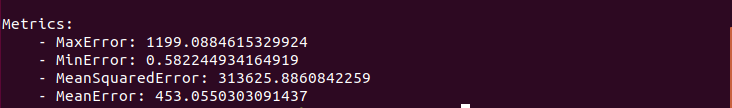
关闭弹出的图片，观察代码输出。输出的第一部分应是循环的轮次以及每轮选中的查询，当选中查询已被观测时还应输出重选的提示：



此后是利用原始数据生成的直方图桶详情和加噪合成后的样子，以及输出本次算法随机生成的查询：

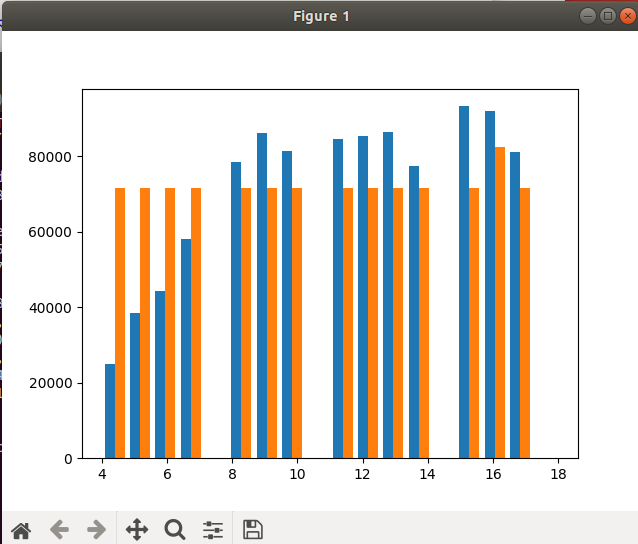


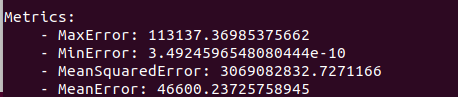
最后，是利用四种误差算法计算出的误差：



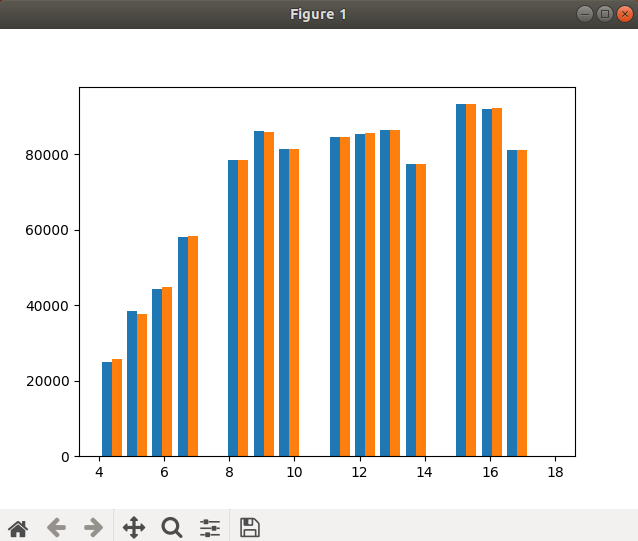
4.调整参数，将应能观察到以下变化：

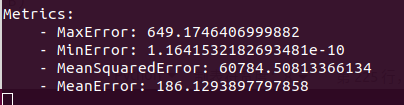
①当保持其他参数不变，调低循环轮数时，原始数据和合成数据的差异增大（从图上和统计数据都能得到此结论）：





②当保持其他参数不变，增大隐私预算时，原始数据和合成数据的差异减小（从图上和统计数据都能得到此结论。与3中的结果相比，由于30轮次循环已经较为充分，所以图上可能不太明显，但是统计数据上会非常明显）：

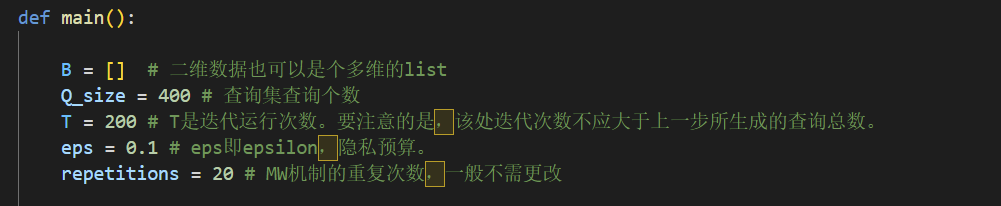




### MWEM\_2D.py实际运行结果参考

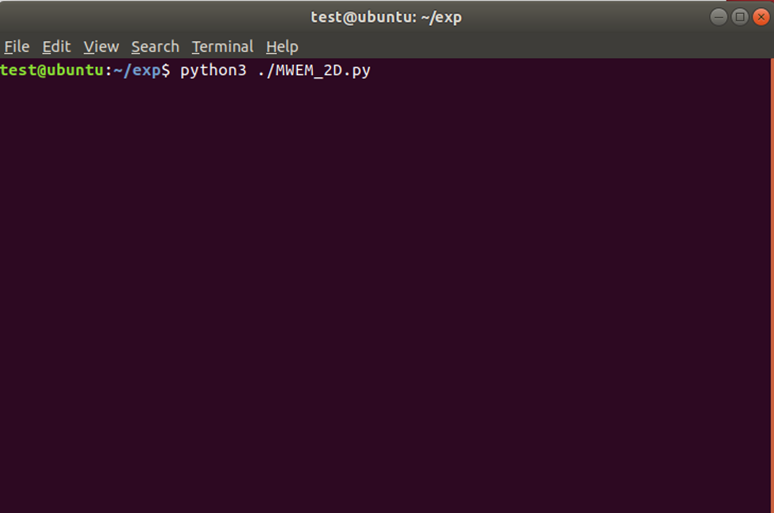
本部分将展示补全后的代码的实际运行结果和各部分函数应达到的目标。注：本部分使用隐私预算eps=3，查询集个数400，轮次T=200。在运行二维查询时，轮次小于100时效果非常不好。

1.在main函数中，正确设置各项实验参数并保存代码文件



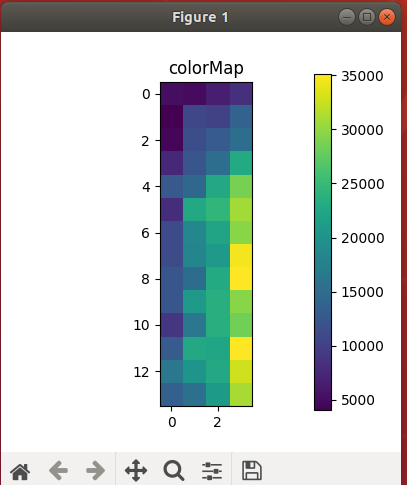
在代码文件所在文件夹右键，选择“在终端打开(Open in terminal)”,输入以下代码来启动运行：

python3 ./MWEM\_2D.py

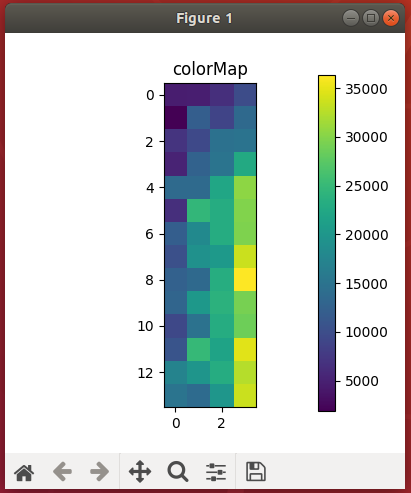


3. 你将可以依次看到以下内容：

首先，会弹出自动生成的原始数据集的数据点染色图，越深代表该点数据越多：



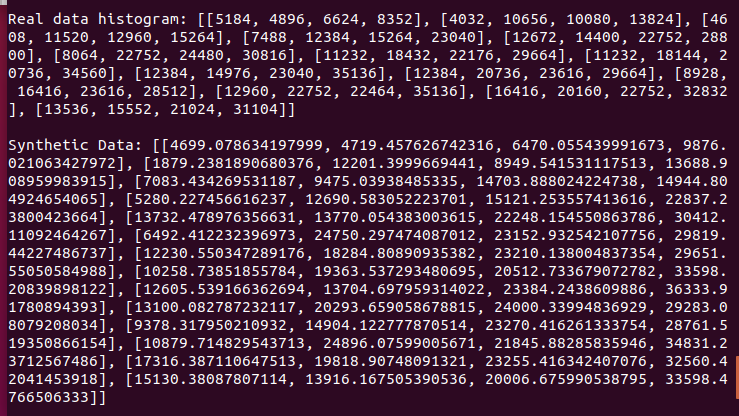
之后，会弹出自动生成的合成数据集的数据点染色图，越深代表该点数据越多：



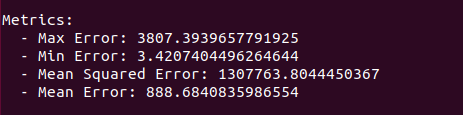
关闭弹出的图片，观察代码输出。考虑到轮次较多，本部分代码输出的第一部分应是循环轮次，不包含其他内容：



此后是利用原始数据生成的直方图桶详情和加噪合成后的样子：

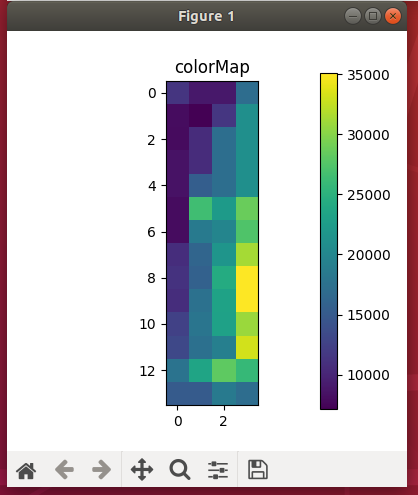
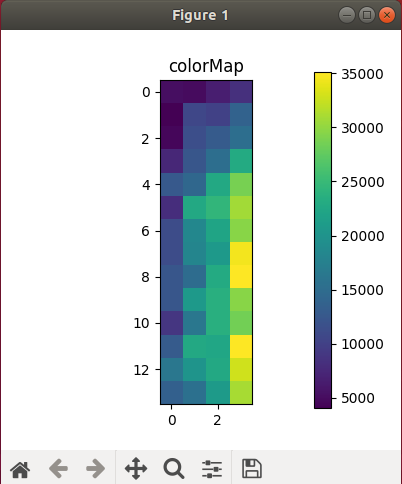


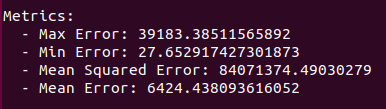
最后，是利用四种误差算法计算出的误差：



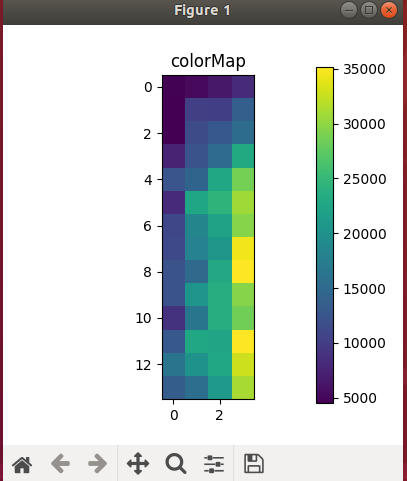
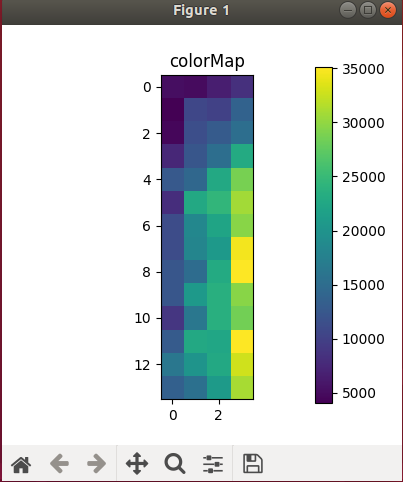
4.调整参数，将应能观察到以下变化：

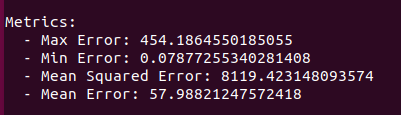
①当保持其他参数不变，调低循环轮数时，原始数据和合成数据的差异增大（从图上和统计数据都能得到此结论）：





②当保持其他参数不变，增大隐私预算时，原始数据和合成数据的差异减小：





## 项目提交要求

请以压缩包形式提交，压缩包内应包含如下文件：

Datasets文件夹：与项目下发时一致。

Results文件夹：应包含至少一次实验结果，请手动重命名所有实验结果文件为“文件名\_eps0.1\_T30.png”，如“result\_histo\_eps0.1\_T30.png”，以免后续评分时新生成的结果覆盖了提交的结果。

MWEM.py：提供完整补全、可运行的代码。

errortools.py：提供完整补全、可运行的代码。

（可选）MWEM\_2D.py：提供完整补全、可运行的代码。当提交此代码时，Results文件夹里应有相应的结果。

requirements.txt：可与原来一致，也可加入自己多引用的库，但要求使用后**必须**能完整安装提交代码所需的依赖库，否则，按代码无法运行处理。

实验报告：按照模板撰写的完整实验报告，以docx形式或pdf形式提交。