针对 VASP 的材料计算教程
为材料计算而生

Jiaqi Z.[1]

2024 年 8 月 12 日

目录

前言

“为材料计算而生”，是抱着多大的觉悟说出这种话啊。这只是一本书，有办法背负其他人的人生吗？

真是满脑子只想着自己呢……[1]

俗话说得好，好记性不如烂笔头，这句话在任何时候都显得格外贴切。尤其是在科研领域（特别是材料科学这样规范和流程化的学科），记录的重要性更是不言而喻。随着计算任务的不断增加，我们掌握的计算方法和参数也日益繁多。将这些知识、操作和方法记录下来，不仅能够帮助我们避免遗忘，还能在遇到类似问题时快速查找，而不必在海量的网络搜索结果中苦苦寻觅。

正是基于这样的考虑，我们结合自己科研团队在材料计算方面遇到的一些实际问题，整理编写了这本书。我们编写这本书的目的有两个：一是为了方便自己，在面对类似问题时能够迅速回忆起解决方案；二是为了通过集体的智慧，汇聚大家的方法和思路，以便在遇到新问题时能够迅速找到答案。

这本书的诞生，既是为了服务于材料计算，也是因材料计算而生。既然如此，我们为何不称它为“为材料计算而生”呢？

我们衷心希望这本书能够惠及更多的人，无论是我们团队的新成员，还是其他团队的老师或同学，都能从中获得帮助。

同时，我们也清楚地认识到自己的能力和知识是有限的，书中的内容难免会有疏漏或错误。我们诚挚地希望读者在使用过程中能够提出宝贵的意见和建议，或者分享你们的经验，共同促进我们的成长和进步。

最后，再次感谢您阅读并使用这本书。

Jiaqi Z.

2024 年 8 月青岛

[1] 以上内容改编自动漫《BanG Dream! It’s MyGO!!!!!》中丰川祥子的台词

如何联系作者

可通过以下任意一种方式联系:

- GitHub 的 Issue, 这是最直接的方式[2];
- email, 请发送邮件至 zhangjq00@sdust.edu.cn 或 zhangjq_sd@163.com

如何使用这本书

在使用时，请按照如下方法：

1. 根据研究问题，寻找合适的章节；如果没有，可以在 GitHub 上提交 Issue 或者贡献 Pull Request;
2. 在每一节开始，会介绍本节的内容和知识点，查看是否与你的研究问题符合；如果不符合，返回第 1 步重新查找新的章节;
3. 阅读这一节内容，并试着针对自己的问题进行操作（或简单检查自己操作是否正确）。如果报错或出现异常结果，进行第 4 步；如果成功，进行第 5 步;
4. 在该节后面的“错误处理”部分，会介绍如何处理报错或异常结果，并给出解决方案。请查找是否有你需要的解决方案，并尝试解决。如果已经解决，进入第 5 步；否则重新查找新的解决方案；若所有解决方案都无法解决，请提交 Issue 或者贡献 Pull Request;
5. 放下教程，继续你的研究；或者阅读这一章其他内容，了解其他相关内容。

上述步骤可能（也一定）会重复许多次

关于本笔记的版权使用说明

- 本书可免费用于学习, 科研等非商业活动;

[2]GitHub 仓库地址：https://github.com/JackyZhang00/Computational-Materials-Tutorial

- 可以以非商业目的进行传播, 但在传播过程中必须保证内容的完整性 (截止到最新发布时, 包括但不限于仓库内 Latex 源码, pdf 文件等. 下同), 需保证作者信息完整, 不得进行修改;
- 本书不可用于任何商业用途 (如确有需要, 需联系作者);
- 除在 GitHub 仓库以 pull request 形式进行编辑修改外, 不允许进行修改并公开传播私自修改版本 (以 GitHub 仓库版本为标准版本);
- 本书著作权归作者 (Jiaqi Z.) 所有, 其他进行创作的人员也可获得著作权, 其他著作权所有者不得违反上述版权说明;
- 如因违反上述说明传播而造成不良影响, 与作者和其他创作者无关, 特此声明;
- 以上说明解释权归 Jiaqi Z. 所有, 且如有后续更新, 以 GitHub 仓库最新版说明为准.

创作者名单

感谢以下人员参与贡献了内容：
Jiaqi Z.

0.1 关于如何编写模板（教程）

本节作者：Jiaqi Z.

在本节，你将要学到：

- 一些基本的 LaTeX 语法
- 如何输入公式
- 如何插入代码
- 如何插入代码块

0.1.1 文章结构

请在编写正文内容时，以“节”（section）为单位创建 tex 文档，同时为方便引用，请在每个小节的后面按照`\label{sec:节标题}`的格式创建标签。

若需要添加小节，使用`\subsection{小节标题}`命令，同时类似于上方关于节标题标签的创建规范，以`\label{subsec:节标题-小节标题}`的格式创建标签，方便他人引用。

对于更小一级的小节（`\subsubsection{}`），对标签不作规范。事实上，我们不建议在引用时涉及到这一层级。通常涉及到小节即可。

********** 以下内容为注意 **********

注意：若你需要修改某一节（或小节）的标题，编译后需要确认是否与他人的标签产生冲突（这通常出现在他人提前按照原有格式引用后发生了修改，从而导致无法指向正确标签）。因此，你需要检查编译后文件是正确的，至少要求通过编译，在必要时需要修改他人代码当中的引用标签与新标签一致。

********** 以上内容为注意 **********

对于正文内容，请使用正常的 LaTeX 语法。例如，当你希望对某一段文字进行强调时，请使用`\emph{}`语句。例如，这是一句强调的话在代码中体现为`\emph{这是一句强调的话}`。你并不需要关注具体的强调格式–LaTeX 会按照统一的格式进行编排。

当你希望分段时，使用空行即可。

对于条目,请在必要的时候使用itemize环境（没有顺序列表）或enumerate环境（有顺序列表）。在环境内使用\item进行编号。环境之间可以嵌套。例如：

- 列表 1
- 列表 2
    1. 列表 2-1
    2. 列表 2-2
- 列表 3

在代码中体现为：

```
\begin{itemize}
   \item 列表1
   \item 列表2
   \begin{enumerate}
      \item 列表2-1
      \item 列表2-2
   \end{enumerate}
   \item 列表3
\end{itemize}
```

0.1.2 一些特殊的环境

为统一教程格式，当你希望添加一段让读者注意的文字时，请使用环境attention例如，下面的语句

```
\begin{attention}
   当你写注意语句时，不需要在前面加任何符号。
\end{attention}
```

在编译后的结果为：

********** 以下内容为注意 **********

注意：当你写注意语句时，不需要在前面加任何符号。

********** 以上内容为注意 **********

类似地，对于一些补充性质的内容，可以使用extend环境，例如：

```
\begin{extend}
    这是一段补充的内容，同样不需要在前面加任何符号。
\end{extend}
```

编译后的结果为：

【扩展内容】：这是一段补充的内容，同样不需要在前面加任何符号。

0.1.3 数学公式

当你希望添加数学公式时，请使用equation环境。同时，在使用\label语句进行标签注明时，请如同代码所示那样，添加“节标题”避免冲突且方便引用。

```
\begin{equation}
    \label{eqn:数学公式-1}
    a^2+b^2=c^2
\end{equation}
```

$$a^2 + b^2 = c^2 \tag{1}$$

在引用公式时，请使用如\ref{eqn:数学公式-1}方式进行交叉引用。请勿直接在正文内写编号以免出现引用错误。

********** 以下内容为注意 **********

注意：在引用公式时，所引用的公式尽量保持在本节内。同时，为避免他人引用，请在编写完成后尽量不要修改相关标签。

为避免公式删除导致的错误，如确实需要引用其他章节的公式，一个较合理的做法是将其他章节的公式在使用时拷贝至当前章节，同时另起标签名。之后的引用限制在当前章节内。

********** 以上内容为注意 **********

如需要添加多行公式，请使用gather环境或align环境。例如，

$$a^2 + b^2 = c^2 \tag{2}$$

$$\int_a^b f(x)\,\mathrm{d}x = F(b) - F(a) \tag{3}$$

0.1.4 图片

当添加图片前，请首先在相关文件夹内创建名为fig的文件夹，在插入图片时如正常 LaTeX 代码插入即可。即使用下面的代码方式

```
\begin{figure}
    \centering
    \includegraphics[width=1\linewidth]{图片路径}
    \caption{图片标题}
    \label{fig:图片-标签}
\end{figure}
```

********** 以下内容为注意 **********

注意：在引用图片时，务必使用\ref{fig:图片-标签}进行引用。例如，图 *1*。不要使用“如上图”、“如下图”的表述形式，以免图片位置发生移动造成指代不明。

********** 以上内容为注意 **********

0.1.5 代码

代码使用lstlisting环境。

```
SYSTEM = 0
ISMEAR = 0
SIGMA = 0.05
NSW = 1
```

如果希望在代码左侧添加行号以示说明,请在引用环境的右侧添加numbers=left设置，即采用下面的代码：

图 1: 图片标题

```
\begin{lstlisting}[numbers=left]
    ..
```

得到效果如下所示

```
1 SYSTEM = O
2 ISMEAR = 0
3 SIGMA = 0.05
4 NSW = 1
```

********** 以下内容为注意 **********

注意:若你希望在代码块中添加代码名称(例如文件名),可以使用caption选项进行说明，例如，下面的代码:

```
\begin{lstlisting}[caption=简单的INCAR文件]
    ..
```

实际编译后结果为

Listing 1: 简单的 INCAR 文件

```
SYSTEM = O
ISMEAR = 0
SIGMA = 0.05
NSW = 1
```

********** 以上内容为注意 **********

0.1.6 添加索引

建议在编写过程中，为文中的程序关键字创建索引。为方便使用，可以使用命令\keyword{}。例如，在 VASP 讲解 SIGMA 函数时，可以使用下面的代码

```
\keyword{SIGMA}
```

从而在表述关键字的同时创建索引。同时,也可以使用\keywordin{}{}的形式创建带有所属关系的索引，例如

```
\keywordin{INCAR}{SIGMA}
```

便是创建了属于 INCAR 条目下的 SIGMA 索引。

********** 以下内容为注意 **********

注意：在编写时，鼓励使用索引方便他人根据关键字直接检索。在创建索引时，请提前确认现有的索引表是否有现成的索引。例如，\keyword{SIGMA}和\keywordin{INCAR}{SIGMA}会得到不同的结果。

同时，在编写过程中已经将输出和索引集成在一个命令中，你不需要特地再编写一个输出的命令如 *SIGMA**keyword{SIGMA}*，只需要使用 *keyword{SIGMA}* 便可在输出 *SIGMA* 的同时创建索引。

********** 以上内容为注意 **********

对于一些特殊的内容，可能不希望给出索引（通常是命令或者文件名等），它们既不属于关键字，也不属于简单的英文单词。为了将它们区分，使用\code{}命令进行编写。例如，\code{cd \$OLDPWD}的输出结果为 cd $OLDPWD

********** 以下内容为注意 **********

注意：在一些特殊的情况下（例如上面的例子），可能会包含 LaTeX 本身的特殊符号（如 $ 本身作为公式符号）。在输入时，应当使用反斜杠 \ 作为转义字符。

********** 以上内容为注意 **********

Part I

Linux 基础

Chapter 1

Linux 命令行操作

Contents

1.1 认识 Linux 目录

本节作者：Jiaqi Z.

在本节，你将要学到：

- Linux 命令格式
- 如何在 Linux 当中表示目录
- 绝对路径和相对路径
- 如何快速表示当前目录和上一级目录

1.1.1 命令格式

与 Windows 使用可视化界面不同，Linux 大多时候使用命令行（shell）进行操作。因此，在 Linux 的学习过程中，一个最重要的任务，就是掌握一些常见的 Linux 命令。对于大多数科研课题组而言，Linux 系统都是在远程云端服务器上，因此在本地往往只需要一个终端程序即可连接到服务器。一些常见的终端软件包括 Xshell、MobaXterm、甚至 VS Code[1]等。

********** 以下内容为注意 **********

注意：如果你熟悉其他操作系统，可能听闻过类似于 *Windows Server*，或者 *Linux* 的 *Ubuntu* 这样的操作系统。明明也可以使用可视化界面，为什么在科研过程中从来不会用到它们呢？（更严谨地说，在远程服务器上）。实际上，当使用可视化界面进行远程连接时，所产生的网络资源消耗是巨大的，通常需要更大的带宽，而使用命令行就可以提高数据传输效率。此外，更重要的一点是，使用命令行可以很容易实现批量处理，这在后续的章节会介绍到。

********** 以上内容为注意 **********

[1]对于 VS Code 而言，可能需要扩展插件（例如 Remote-SSH）的支持

在 Linux 当中，输入命令通常采用的格式是命令 [-选项] [参数]，其中中括号表示这个部分是可选的，即可以没有的。例如，当我们希望列出当前目录下所有文件时，可以使用 ls直接输出，也可以使用 ls -l 以列表格式输出。

********** 以下内容为注意 **********

注意：在后面可能会看到选项有多个的情况，此时为了简化，可以将选项合并在一起。例如，*ls -l -a* 可以简化为 *ls -la*。

命令与选项、参数之间是以空格进行分割，且这个空格不能省略。

*********** 以上内容为注意 **********

1.1.2 目录表示方法

在 Linux 当中，所有目录都是以根目录/为起点，任何目录都是根目录的子目录。根目录下存在一些固定的目录（这些目录通常有特定的含义），例如，在根目录下有一个叫做 bin 的目录（通常写作/bin），它存放的都是二进制文件，也就是系统可以执行的程序文件。

********** 以下内容为注意 **********

注意：在 *Linux* 当中，任何命令实际上都是可执行程序。你可以在*/bin* 目录下看到后面所学的所有 *Linux* 终端命令。

*********** 以上内容为注意 **********

另一个比较重要的位置是家目录/home，它存放的是用户个人文件。在这一目录下，你可以看到系统所注册的所有用户名。但是，这些文件夹大多数是无法查看的[2]。对于用户自己的家目录，通常也可以表示为~。通常来说，当你使用终端等连接登录时，默认的所在目录就是家目录~

1.1.3 绝对路径和相对路径

任何目录在操作时都具有两种表示方式，绝对路径和相对路径。正如1.1.2所介绍的那样，任何目录都是从根目录开始的。因此在描述一个目录时，

[2]这涉及到 Linux 操作权限的问题，通常来说，权限分为三组，即所有者权限、所属组权限和其他用户权限。对于/home 目录下而言，所有目录都是对所有者（即这个用户本身）提供全部权限，而其他人无法访问、修改。

可以从根目录（即/）开始。例如，若你在你的家目录下有一个叫做 vasp 的目录，那么它的绝对路径就是/home/< 你的用户名 >/vasp。

随着层级逐渐增多，这种表示方法也会越来越复杂，因此，在表示一个目录时，默认也可以从当前所在目录开始算起（即相对路径）。例如，若你刚刚进入终端，此时所在目录就是~ 目录，即/home/< 你的用户名 > 下，此时若想表示 vasp，则只需要使用 vasp 即可。

********** 以下内容为注意 **********

注意：在这种情况下，你可以将目录 *vasp* 理解为 < 当前所在目录 >/*vasp*，即等价于/*home*/< 你的用户名 >/*vasp*。

千万不要写成/*vasp*，它表示根目录下的 *vasp* 目录。如果你希望特别强调当前目录，可以使用符号.（一个点）表示“当前目录”，即可以写成./*vasp*

********** 以上内容为注意 **********

然而，在这种情况下，回到当前目录的上一级目录是麻烦的，即在目前所学范围内，只能使用绝对路径。好在 Linux 提供了一个命令：..（两个点）表示上一级目录。因此，如果你当前处在目录/home/< 你的用户名 >/vasp 当中，则.. 表示/home/< 你的用户名 >

同理，../.. 表示父目录的父目录，在上面的例子中即为/home 目录。

********** 以下内容为注意 **********

注意：在终端当中，..（两个点）表示父目录（即上一级目录），而一个点.表示当前目录。

这些符号（指令）在后续关于目录操作中都是可以使用的。

********** 以上内容为注意 **********

看到这里，可以思考下面的问题：如果在你的家目录下有两个目录 python 和 vasp，此时你在/home/< 你的用户名 >/vasp 目录下，如何可以快速表示 python 目录呢（不能使用绝对路径）？答案在下面

答案：../python 即可表示/home/< 你的用户名 >/python 目录

1.2 目录操作

本节作者：Jiaqi Z.

在本节，你将要学到：

- 如何显示当前目录下所有文件
- 如何创建目录
- 如何切换至其他目录

1.2.1 显示目录文件 `ls`

在这一节以及下一节，我们将讨论如何对目录和文件做基本的操作。无论是哪一种，一个最基本的前提是*知道当前目录有哪些文件和目录*，从而才能进行后续操作（例如编辑、删除、移动、进入目录等）

在 Linux 当中，列出一个目录下所有文件使用的是 `ls` 命令。在没有任何参数与选项的前提下，它输出的结果就是*当前所在目录下的所有文件和目录*。以 1.1.3一节最后的例子为例，家目录下有 `vasp` 和 `python` 两个目录，当在家目录下执行 `ls` 命令时，结果如下：

```
$ ls
vasp python
```

同时，`ls` 支持在后面添加一个参数表示要输出的目录。例如，在这一例子下，若在家目录当中执行命令 `ls vasp`，将会输出 `vasp` 目录下的所有文件和目录。利用`..` 表示上一级目录的用法，若当前处在`~/vasp` 目录下，使用 `ls ..` 便可得到上一级目录（即家目录）下的所有文件和目录。

`ls -l`

下面介绍两个常见的 `ls` 选项，首先是`-l` 选项，它表示*以列表形式输出结果*。例如，还是上面的例子，使用这一命令的结果为：

```
$ ls -l
```

```
total 0
drwxrwxr-x 2 zjq zjq 6 Aug 12 16:35 python
drwxrwxr-x 2 zjq zjq 6 Aug 12 16:35 vasp
```

【扩展内容】：每一个文件的输出结果可以分为 *9* 个部分，分别是：权限、文件硬链接数或目录子目录数、拥有者用户名、拥有者所在组、文件大小、文件修改月份、日期、时间、文件名。

关于权限，可以将其分成四部分：第一部分（一个字符）表示文件类型（这里的 *d* 表示目录），第二部分（三个字符）表示拥有者权限（*rwx* 表示可读可写可执行），第三部分（三个字符）表示组用户权限，第四部分（三个字符）表示其他用户权限（*r-x*）表示可读，可执行但不可编辑。

对于文件硬链接数和目标子目录数，对于初始创建的文件而言，通常为 *1*，而对于目录而言，默认为 *2*（因为有两个子目录 . 和 ..）

有时，也可以使用 ll代替指令 ls -l，其二者是完全等价的。

ls -a

-a 选项表示列出所有文件，包括隐藏文件。例如，在7vasp 目录下，使用 ls -a 命令，结果为：

```
$ ls -a
.  ..extend
```

【扩展内容】：正如前面所介绍的那样，任何一个空目录都会默认有两个隐藏目录–自身和它的上一级目录。而这也解释了 *1.1.3*一节所介绍的 . 和 .. 的本质，它们实际上就是任何当前目录下的两个子目录。

********** 以下内容为注意 **********

注意：前面所介绍的*-l* 选项和*-a* 选项是可以合并使用的，此时可以将两个选项之间以空格分割，如 *ls -l -a*，或者将两个选项写在一起 *ls -la*

当选项写在一起时，选项的排列顺序不重要。

与最开始介绍 *ls* 后面加参数表示目录一样，带有选项的 *ls* 同样可以在后面添加参数，例如，*ls -a vasp* 表示列出当前目录下的 *vasp* 子目录下的所有文件和目录（包括隐藏文件）

********** 以上内容为注意 **********

1.2.2 关于隐藏文件

【扩展内容】：隐藏文件是指在文件名前面加上 . 的，例如 *.bashrc*。

隐藏文件在 *Linux* 当中的常见用途有：

- 配置文件
- 临时文件
- 缓存文件
- 等

总而言之，隐藏文件是为了防止误操作而存在的。（这可能与一些人认为的“隐藏文件是避免别人看到”不同）事实上，哪怕在 *Windows* 操作系统中，隐藏文件也是存在且方便查看的[3]

1.2.3 创建目录 `mkdir`

如果所有操作都在家目录下进行，那文件管理就太复杂了。试想一下，在科研里面算了好几年的结果，全部“一股脑”堆在一起，既难找，也容易忘记当时是做了什么。因此，一个好的目录管理至关重要。而前提，就是知道如何创建目录。

在 Linux 当中，创建目录的方法是使用 `mkdir` 命令。与前面介绍的 `ls`，以及后面要介绍的 `cd` 不同的是，`mkdir` 必须带有一个参数，表示创建的目录路径。对于刚开始接触的初学者，一个最简单的命令格式是：`mkdir < 目录名 >`，其中表示在当前目录下创建一个名为 `< 目录名 >` 的目录。例如，希望在当前目录下创建一个名为 `ML` 的目录，则可以使用命令 `mkdir ML`。

正如前面所介绍的路径，`mkdir` 后面的路径也可以是绝对路径或相对路径。无论是哪种形式，其含义是一样的，即在你所描述的路径下创建目录。利用这种方法，你可以在更远的层级关系下创建目录。例如，在`~/vasp/lattice/Fe` 目录下创建`~/python/ML/plot` 目录。

[3]在 Windows 操作系统中，可以通过右键-属性-隐藏的方式将文件或文件夹设置为隐藏；相对地，对 Windows10 操作系统而言，可以通过文件夹菜单栏的“查看”-“隐藏的项目”找到那些隐藏文件。只不过在 Linux 当中，隐藏文件使用前面加点 . 的方式设置，但无论如何，隐藏文件永远不是不让别人看见的方法，如果想达成这一目的，正确的方法是设置权限。

********** 以下内容为注意 **********

注意：你所写的路径名，应当是你所要创建的目录。这句话似乎有点绕，举个例子，如果你希望在*/home/zjq/vasp* 下创建一个名为 *lattice* 的目录，则你需要运行的命令是 *mkdir /home/zjq/vasp/lattice*。注意到，后面的路径实际上就是你要创建的目录。

********** 以上内容为注意 **********

1.2.4 切换目录 cd

在 Linux 当中，切换目录使用的命令是 cd，通常来说，后面需要配合一个参数，表示要切换到哪里。例如，使用命令 cd /home 则是将当前目录切换到/home 目录下。配合以..，可以使用 cd .. 切换到上一级目录。

思考：如果使用 cd .，会得到什么结果？

答案：

特殊的，对于家目录而言，除了可以使用 cd~ 外，Linux 也支持直接使用 cd，不添加任何参数实现这一功能，即二者是等价的。

1.2.5 错误处理

-bash: cd: < 目录名 >: Not a directory

cd 后面的参数必须是目录，不能是文件。如果参数是文件，则会报该错误。

如果不知道哪个是目录，哪个是文件，可以使用 ls -l 查看第一个字符（文件类型），如果第一个字符是 d，则表示目录，如果是-，则表示文件[4]。例如，

```
$ ls -l
total 4
-rw-rw-r-- 1 zjq zjq 4 Aug 12 17:12 INCAR
drwxrwxr-x 2 zjq zjq 6 Aug 12 16:35 python
drwxrwxr-x 2 zjq zjq 6 Aug 12 16:35 vasp
```

[4]在一些比较新的终端程序中，可能会将文件和目录以不同颜色区分。例如，在 MobaXterm 当中，默认情况下文件是白色，目录是蓝色。当然，这些颜色设置都是可以通过 Settings-Terminal-Default color settings 设置颜色主题，这里所说的这一例子为“Dark background / Light text”主题

表示 INCAR 是文件,而 vasp 和 python 是目录。如果执行了 cd INCAR，则会报错。

-bash: cd: < 目录名 >: Permission denied

这表明你尝试进入一个你没有权限的目录。例如，在/home 目录下，有 ljk 和 zjq 两个目录，分别表示两个用户。如果执行 ls -l 则会发现：

```
$ ls -l
total 32
drwx------ 13 ljk  ljk   4096 Aug 5 17:34 ljk
drwx------ 75 zjq  zjq   4096 Aug 12 17:12 zjq
```

很显然，每个目录只有目录拥有者自己可以访问。例如，作为用户 zjq，当尝试执行 cd ljk 时，则会报错。

【扩展内容】：这种情况有一个特例：*root* 用户。对于 *root* 用户而言，可以进入任何目录。但通常来说，*root* 用户是由服务器管理者所持有的，作为一般用户而言，不需要也不应该进入没有权限的目录，或者执行没有权限的操作[5]。

mkdir: cannot create directory < 目录名 >: No such file or directory

虽然我们说可以用绝对路径或相对路径在更远的层级关系下创建目录。但这一操作的前提是，这个目录的上一级目录需要存在。例如，当你执行 mkdir vasp/lattice/Fe 时，首先需要确保目录 vasp 和 vasp/lattice 存在，才会创建 vasp/lattice/Fe。如果你要创建的目录其上一级目录不存在，则会报错。

一个很自然的解决方法是：一层一层创建。这种方法虽然麻烦，但可以确保目录是清晰的[6]。

[5] “删库跑路”这件事对于一般用户来说是不可能的

[6] 如果你确实想要一个快捷的方法，可以使用选项-p。这一选项可以在遇到没有的目录时自动为你创建。例如，上面的例子也可以直接使用 mkdir -p vasp/lattice/Fe，但这一操作需要保证输入内容是正确

mkdir: missing operand

很显然，你在使用 `mkdir` 时没有给任何参数。正如 1.2.3所说的那样，在调用 `mkdir` 时必须提供一个参数表示要创建的目录路径。

1.3 文件操作

本节作者：Jiaqi Z.

在本节，你将要学到：

- 如何移动文件（目录），如何给文件（目录）重命名
- 如何删除文件（目录）
- 如何复制文件（目录）

这一节，我们专注于文件的相关操作。类似于 Windows 的基本操作，Linux 的文件操作也无外乎就是移动、删除、复制。同时，这一节的许多命令对于文件和目录都是适用的，但可能会有一个注意事项，这往往会出错。

1.3.1 移动文件 `mv`

在 Linux 当中，移动文件使用的命令是 `mv`。其基本用法是 `mv < 源文件路径 > < 目标文件路径 >`。例如，我们在 `vasp` 目录下，希望将里面的 `OUTCAR` 移动至上一级目录，可以使用 `mv OUTCAR ..`。类似地，对于更复杂的文件移动，只不过在描述路径时稍微复杂一点，其他的步骤没有什么不同。

如果你足够敏感，也许会发现一点问题：为什么前面的命令，*OUTCAR* 是文件，而 `..` 是目录？两个难道不应该统一吗？

对于这个问题，可以分两个部分讨论：如果前面是文件，后面也是文件，例如 `mv OUTCAR ../OUTCAR`，这个命令与前面的命令效果是完全等价的。但是，有趣的地方在于，如果你试着执行 `mv OUTCAR ../INCAR` 的话，你会发现，Linux 将 `OUTCAR` 移动到 `..` 的同时，还将其改名为 `INCAR`。

的。一旦有内容输错，则极有可能造成目录结构混乱。

进一步想一下，如果我们现在直接写成 mv OUTCAR INCAR 的话，可以将其看作把当前目录下的 OUTCAR 移动至当前目录，同时改名为 INCAR，总的效果就是，文件被重命名为 INCAR。

********** 以下内容为注意 **********

注意：正如你所见到的那样，*Linux* 没有单独的重命名文件命令，而是通过 *mv* 命令来完成。

*********** 以上内容为注意 ***********

进一步，如果前后两个参数都是目录会发生什么呢？很简单，就是将前面的目录移动至后面的目录，从效果上看，近似于将第一个参数的目录看作文件。

********** 以下内容为注意 **********

注意：与文件移动类似的操作，如重命名，对目录的移动同样成立。

*********** 以上内容为注意 ***********

1.3.2 如何删除文件 rm

相比于移动文件需要两个参数，删除文件的命令 rm 只需要一个参数即可，也许你也能猜到这个参数的含义，即 rm < 删除的文件路径 >。例如，要删除当前目录下的 INCAR 文件，只需要执行 rm INCAR 即可。同样的，你也可以使用更复杂的绝对路径或相对路径，例如，删除上一级目录下的 OUTCAR 文件，可以使用 rm ../OUTCAR。

【扩展内容】：与 *Windows* 不同，*Linux* 删除文件通常是直接删除，而不是放在所谓的回收站内。因此，在删除文件时务必小心。

在有些版本的 *Linux*(例如 *Ubuntu*)当中，删除的文件被移动至 */home/<用户名>/.local/share/Trash/files* 当中，这个目录起到的临时的回收站功能，但你不应该寄希望于这个功能，而是仔细检查删除文件的正确性，并做好合适的备份。

对于删除目录而言，情况有点特殊，需要使用 rm -r命令删除一个目录，此时后面所接参数为目录的路径，例如，删除当前目录下的 vasp 目录，则可以使用 rm -r vasp。

********** 以下内容为注意 **********

注意：*-r* 选项通常表示递归，例如，在 *rm -r* 当中，表示递归删除，从而达到删除一个目录的效果。在删除目录时会连同里面的所有内容都删除掉，因此需要特别小心。

如果担心删除错误的文件，可以在选项中使用*-i*。*rm -i*表示在删除时询问是否删除。

对于空目录而言，*Linux* 还提供了一个命令 *rmdir*，其用法为 *rmdir* < 目录路径名 >，可以删除一个空目录。

********** 以上内容为注意 **********

1.3.3 如何复制文件 cp

复制文件的命令为 cp，其用法与移动文件 mv 几乎完全一样，无非就是将移动改为复制。简单来说，语法就是 cp < 源文件路径 > < 目标文件路径 >，类似于 1.3.1当中所介绍的重命名方法，使用 cp 命令同样可以做到复制的同时重命名。例如，cp vasp/OUTCAR ../INCAR 表示将 vasp 目录下的 OUTCAR 文件复制到上一层目录，并重命名为 INCAR

如果想要复制一个目录，也需要使用选项 cp -r。例如，cp -r vasp/ python/表示将 vasp 目录复制到当前目录并重命名为 python。

********** 以下内容为注意 **********

注意：我们在上面的命令当中使用 *vasp/*和 *python/*表示两个目录。其中使用了符号/作为结尾，这个符号通常强调该路径是个目录。对于 *Linux* 本身而言，有没有这个符号并没有区别。例如，*cp -r vasp python* 也可以表示上面的操作。我们这么写只是为了强调这两个路径是目录而不是文件。

********** 以上内容为注意 **********

1.3.4 一次性处理多个文件

前面介绍的 rm，cp，mv，以及在 1.2一节所介绍的 mkdir，都是可以针对多个文件同时操作的。以 rm 为例，如果想同时删除多个文件，只需要在后面添加多个参数即可，其中参数之间以空格分割。例如，rm INCAR KPOINTS 表示删除当前目录下的 INCAR 文件和 KPOINTS 文件。对于 mkdir

创建多个目录而言，也是一样的用法，例如，使用 `mkdir vasp ML` 表示在当前目录下创建 `vasp` 目录和 `ML` 目录。

对于 `cp` 和 `mv` 而言，情况稍有不同。它们自身就需要两个参数，第一个是源路径，第二个是目标路径。如果有多个文件需要处理，Linux 默认最后一个路径为目标路径，前面的所有参数都是源路径。例如，`cp INCAR KPOINTS POSCAR POTCAR ..` 表示将 `INCAR`，`KPOINTS`，`POSCAR` 和 `POTCAR` 复制到上一级目录中。

********** 以下内容为注意 **********

注意：对于 *cp* 和 *mv* 而言，若一次性移动多个文件，则最后一个参数必须是目录。这就意味着不能进行重命名操作。

********** 以上内容为注意 **********

1.3.5 错误处理

rmdir: failed to remove < 路径名 >: Directory not empty

使用 `rmdir` 命令时，只能用来删除空目录。当要删除的目录不是空目录时，执行该命令则会报错。使用 `rm -r <` 路径名 `>` 往往是删除非空目录的常见方法。

cp: -r not specified; omitting directory < 路径名 >

当使用 `cp` 复制目录时，需要添加`-r` 选项。如果没有添加这一选项则会报错。

cp: target < 路径名 > is not a directory

这通常出现在尝试使用 `cp` 复制多个文件时，最后的参数必须是目录。如果此时不是目录，则会报错。

rm: cannot remove < 路径名 >: Is a directory

类似于 `cp` 复制目录，使用 `rm` 删除目录时，也需要添加`-r` 选项。特别地，对于一次性删除多个文件，如果在删除文件的同时也存在把目录删除的情况，也需要添加这一选项。

Part II

VASP 计算

Chapter 2

电子性质

Contents

2.1 能带

Part III

Python 与机器学习

索引