

第一性原理计算与VASP使用

朱文光

中国科学技术大学

2014.7

什么是“第一性原理计算” (First-principles calculations)

原则上指直接从薛定谔方程出发，“不含有任何经验参数”的计算方法，以此被认为是可以比较准确计算材料电子结构的方法。

目前在凝聚态物理和材料科学的“第一性原理计算”一般指以“密度泛函理论”（Density Functional Theory (DFT)）为基础的计算方法。

相近术语：*Ab initio* calculation (从头计算)

Density functional theory calculation

主要涉及的内容

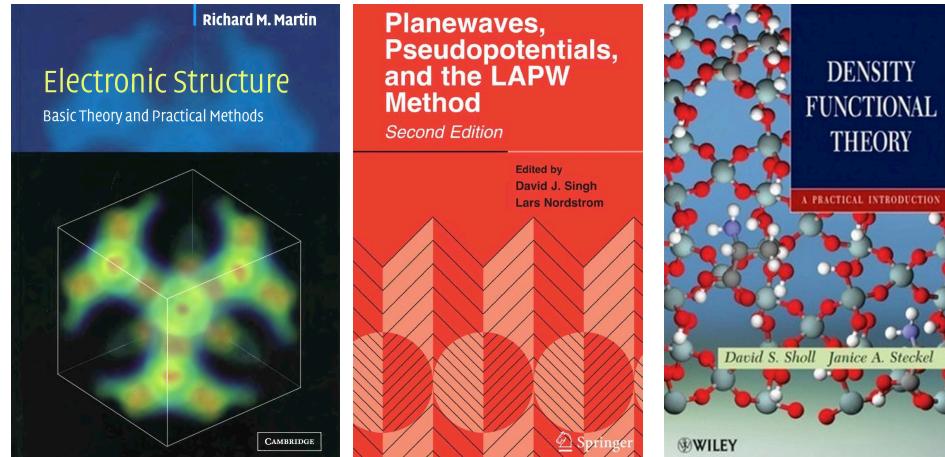
- 密度泛函理论 (Density Functional Theory (DFT)) 的基本概念
- VASP采用的计算方法的简单介绍
 - 方法的适用性和局限性
- VASP计算的一般用法
 - 模型的构建
 - 常用参数的设定
 - 计算结果的解读
 - 常用的辅助工具
 - 常见问题

不涉及的内容

- 理论公式的推导
- 源程序的解读

可参考：

任兴国老师的第一性原理计算的课程



本讲义的主要宗旨：

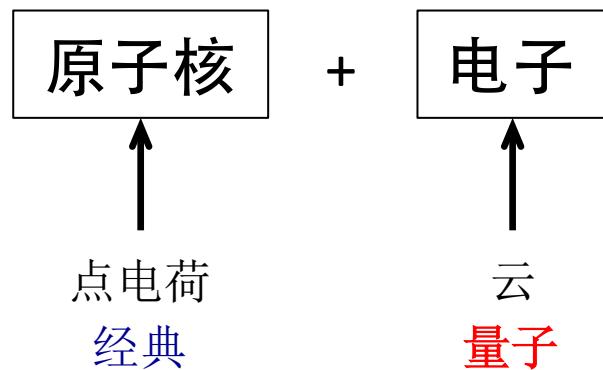
- 基本的物理图像
- 以实用为目的
- 尽量讲书本上没有的东西

学习要点：

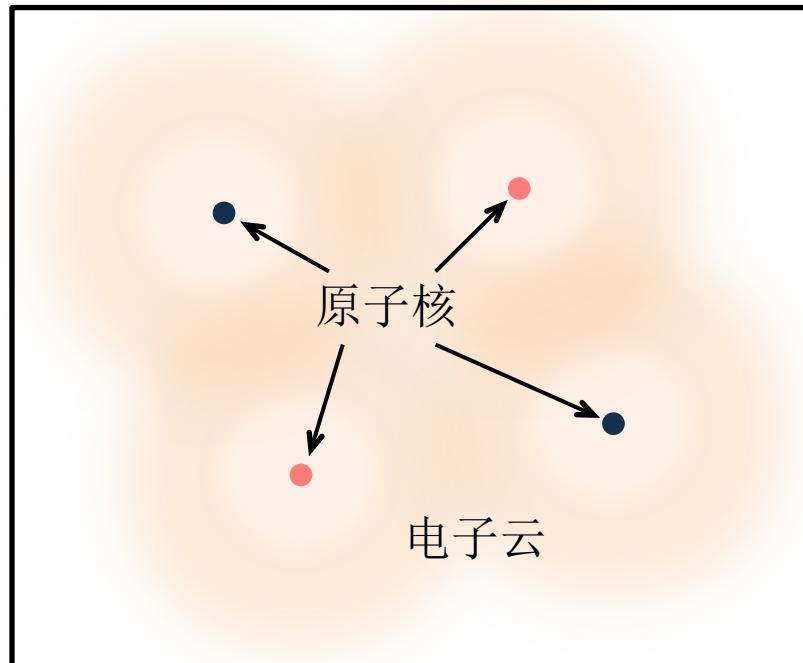
- 怎样得到想要的精度？(输入)
- 哪些参数影响计算速度？(输入)
- 怎样分析计算结果？(输出)

“第一性原理计算” 所处理的物理问题是什么？

系统的尺度：原子尺度



Challenge: 计算电子能级（能带）



绝热近似 (Adiabatic (Born-Oppenheimer) approximation) :

因为原子核质量远大于电子质量，电子的弛豫时间远小于原子核，原子核和电子可以分开处理。

(原子核+电子) 系统的复杂性

系统哈密顿量:

i,j : 电子; I,J : 原子核

$$\hat{H} = -\frac{\hbar^2}{2} \sum_I \frac{\nabla_I^2}{M_I} + \frac{1}{2} \sum_{I \neq J} \frac{Z_I Z_J e^2}{|R_I - R_J|} - \frac{\hbar^2}{2m_e} \sum_i \nabla_i^2 + \frac{1}{2} \sum_{i \neq j} \frac{e^2}{|r_i - r_j|} - \sum_{i,I} \frac{Z_I e^2}{|r_i - R_I|}$$

The diagram illustrates the decomposition of the total Hamiltonian \hat{H} into three main components: atomic nuclei, electrons, and nuclear-electron interactions. The first two terms on the left represent the atomic nuclei, while the last three terms on the right represent the electrons. The middle term, representing the interaction between different atomic nuclei, is also labeled as such.

通过绝热近似，考虑在原子核固定的情况下与电子相关的哈密顿量：

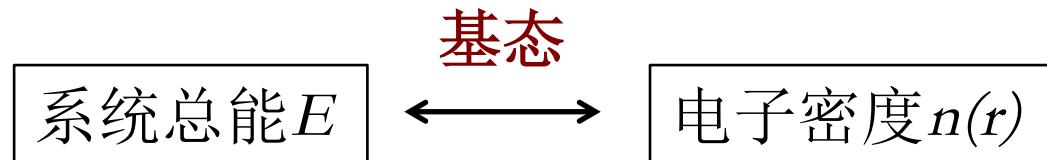
$$\hat{H} = -\frac{\hbar^2}{2m_e} \sum_i \nabla_i^2 + \frac{1}{2} \sum_{i \neq j} \frac{e^2}{|r_i - r_j|} - \sum_{i,I} \frac{Z_I e^2}{|r_i - R_I|}$$

The diagram shows the simplified Hamiltonian for the electron system under the Born-Oppenheimer approximation. It highlights the "complex" part (electron kinetic energy and electron-electron interactions) and the "external potential" part (electron-nuclear interactions). The electron-nuclear interaction term is underlined, indicating it is the primary focus of the approximation.

N个电子的波函数: $\Psi(r_1(x_1, y_1, z_1), r_2(x_2, y_2, z_2), \dots, r_N(x_N, y_N, z_N))$

密度泛函理论 (density functional theory)

在一个多电子相互作用的体系中，系统的总能是电子密度的泛函，并且使系统达到基态的电子密度是唯一的和真实的。



密度泛函理论把一个具有相互作用的多电子体系转化为一个没有相互作用的独立电子问题。

Kohn-Sham Hamiltonian

$$\hat{H} = -\frac{\hbar^2}{2m_e} \sum_i \nabla_i^2 + \int \frac{n(r')}{|r - r'|} dr' + V_{ext}(r) + V_{xc}[n(r)]$$

Hatree能, 库伦相互作用能 外势, 包括原子核对电子的势和各种外加势场 交换关联泛函

交换关联泛函 (Exchange - correlation functional)

原则上如果可以找到一个准确的交换关联泛函，得出的解就是严格精确的，但实际计算中需要做近似。

- LDA (Local Density Approximation)

$$V_{xc}^{LDA}(n)$$

- GGA (Generalized - Gradient Approximation)

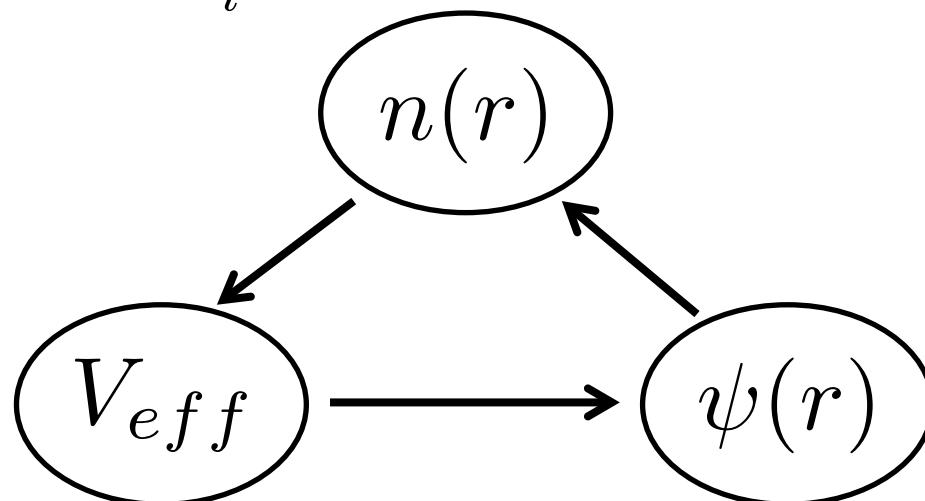
$$V_{xc}^{GGA}(n, \nabla n)$$

popular forms: PW86, PW91, PBE(最常用), revPBE, PBEsol, ...

- 更高阶的方法： Hybrid functional, Meta-GGA, Hyper-GGA, RPA, ...

Kohn-Sham方程的自洽解

$$\left[\begin{array}{l} \left[-\frac{1}{2} \nabla^2 + V_{eff}(r) \right] \psi_i(r) = \epsilon_i \psi_i(r) \\ \\ V_{eff}(r) = V_{ext}(r) + V_{Hartree}(n) + V_{xc}(n) \\ \\ n(r) = \sum_i | \psi_i(r) |^2 \end{array} \right.$$



密度泛函理论计算的准确性

- 对于电子处于基态的物理性质是比较准确的，如晶格常数，结合能，力学性质，原子振动…
 - 晶格常数：LDA偏小，GGA偏大
 - 结合能：LDA偏大，GGA偏小
- 对于与电子激发态相关的物理性质不准确，如半导体的能隙（低估~50%）。
 - 更高阶处理激发态问题的方法：
Hybrid functional, GW, TDDFT…
- 强关联体系常常处理不好
- Van der Waals相互作用被低估
 - 很多改进的方法可以使用

Kohn-Sham方程的数值解法

- 实空间格点
 - GWPAW, MIKA, PARSEC, Octopus(TDDFT)
- 平面波赝势
 - VASP, Quantum Espresso, Abinit, Materials Studio
- 原子轨道基
 - Siesta, FHI-aims, ADF
- 高斯函数基
 - Gaussian
- LAWP
 - Wien2K, ELK, EXCITING
- LMTO

For a more complete list, see <http://www.psi-k.org/codes.shtml#fpol>

Kohn-Sham方程的数值解法

波函数用平面波展开：

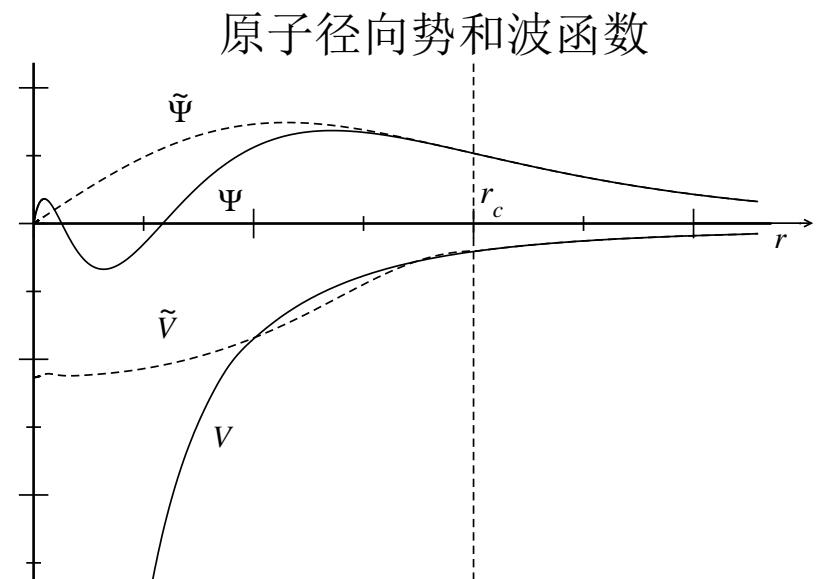
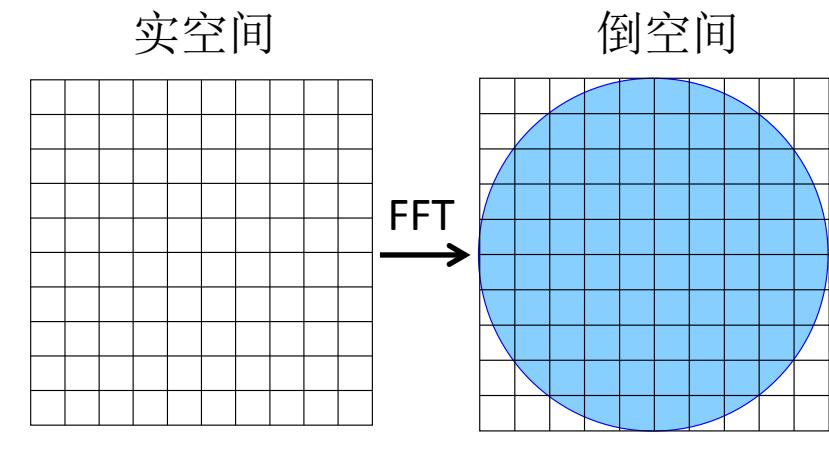
$$\psi_{n\mathbf{k}}(\mathbf{r}) = \frac{1}{\Omega^{1/2}} \sum_{\mathbf{G}} C_{\mathbf{G}n\mathbf{k}} e^{i(\mathbf{G}+\mathbf{k})\mathbf{r}}$$

平面波截断能：

$$\frac{1}{2}|\mathbf{G} + \mathbf{k}|^2 < E_{\text{cutoff}}$$

赝势：

- 用比较平滑的势取代真实的势，为减少平面波的数目。
- 同时要保证电子的化学性质不变。



VASP

(Vienna Ab initio Simulation Package)

- 基于密度泛函理论
- 周期性边界条件，倒空间解
- 平面波+超软赝势或PAW势
- LDA, GGA, Hybrid Functional, Van de Waals corrections
- Nudge Elastic Band (NEB)方法计算能量势垒
- 分子动力学模拟 (molecular dynamics)
- GW, density functional perturbation theory ...

VASP website: <http://www.vasp.at>

VASP Wiki: http://cms.mpi.univie.ac.at/wiki/index.php/Main_page

学习资料： VASP Manual, VASP Workshop Lectures (ppts and hands-ons)

<http://www.vasp.at/index.php/documentation>

The VASP Manual

The [online VASP manual](#) is a lengthy document generated with latex2html. If you're one of those die-hards that like real paper you might prefer to get a [pdf copy](#) of the manual.

Recently, we started a [wiki](#), that in future will replace the online manual completely.

► [The VASP Manual](#)

► [The VASP wiki](#)

► [The VASP forum](#)

VASP Workshop Lectures

In 2003 we organized a VASP workshop. The lecture notes and examples from the handson sessions are highly recommended as place to start if you are a beginner and might also be useful if you are not.

[Introduction to Computational Materials](#)

[Introduction to DFT](#)

[Pseudopotentials I](#)

[Hands-on Session 1, Example Files](#)

[Pseudopotentials II](#)

[Sampling the Brillouin zone](#)

[Hands-on Session 2, Example Files](#)

[Ionic relaxation methods](#)

[Electronic relaxation methods](#)

[Computational Platforms](#)

[Hands-on Session 3, Example Files](#)

[Accuracy and Validation of results](#)

[Pseudopotential Data Base](#)

[DFT in depth](#)

[Unpaired electrons in DFT](#)

[Hands-on Session 4, Example Files](#)

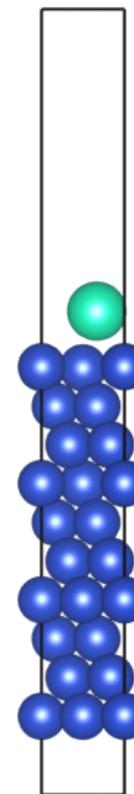
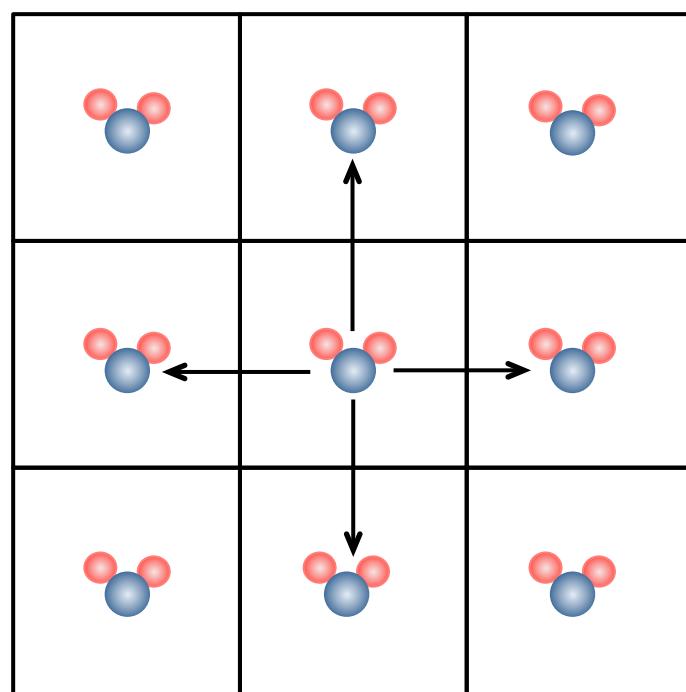
More recently the CSC-IT Center for Science Ltd. in Espoo, Finland, organized two VASP workshops (2006/2009). The lecture notes of those might interest you as well.

[Basics, PAW, DFT, Hybrid functionals](#)

[Linear response, Dielectric properties, GW](#)

周期性边界条件下如何计算非周期体系？

表面计算Slab模型



间距要在 10\AA 以上

VASP输入文件

- POSCAR: 定义Supercell及每个原子的坐标
- INCAR: 设置主要的计算控制参数
- KPOINTS: 定义倒空间（K空间）取点
- POTCAR: 各元素的赝势或PAW势

POTCAR

- 所有元素的赝势和PAW势已由VASP开发组产生好，不需要自己产生。
- VASP提供Ultrasoft Pseudopotential和PAW势的LDA, GGA-PW91, GGA-PBE。通常使用PAW-PBE。
- 使用时用cat命令将不同元素的POTCAR文件组合起来。
例如： cat Mo/POTCAR S/POTCAR >POTCAR
- X_sv, X_pv, X_dv：表示*s, p, d*电子也作为价电子处理，在精度要求比较高的系统中可以使用。

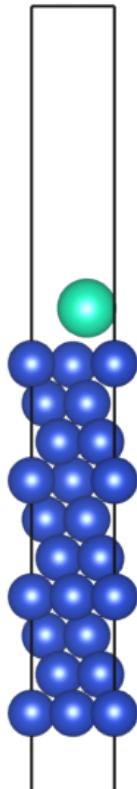
POSCAR

Title ←	MoS2
比例因子，一般取成一个晶格常数 ←	3.19030
Supercell的三个基失 [←	0.8660254037844386 -0.5000000000000000 0.0000000000000000 0.0000000000000000 1.0000000000000000 0.0000000000000000 0.0000000000000000 0.0000000000000000 6.0000000000000000
元素名 ←	Mo S
各元素的原子个数 ←	1 2
原子是否运动 ←	Selective dynamics
坐标格式 ←	Direct
原子坐标 [←	0.0000000000000000 0.0000000000000000 0.5000000000000000 F F F 0.333333333333357 0.666666666666643 0.4182374246372618 F F T 0.333333333333357 0.666666666666643 0.5817625753627382 F F T

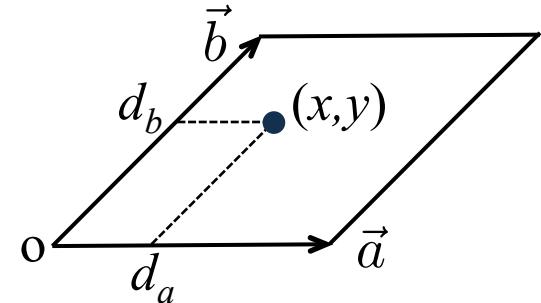
设置各个原子沿各基失方向是否允许移动

POSCAR

Direct的定义：原子在Supercell的基失坐标系下的坐标。



$$(x, y, z) = (\vec{a}, \vec{b}, \vec{c}) \begin{pmatrix} d_a \\ d_b \\ d_c \end{pmatrix} \alpha_0$$



三点建议：

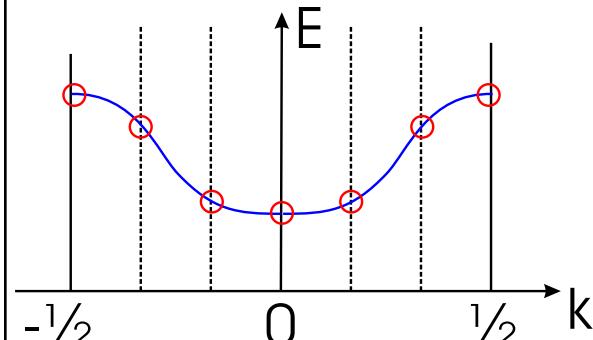
- 第二行的比例因子 α_0 不要设成1，而最好设成系统中某种材料的晶格常数。
- 比较关注的原子尽量放在前面（避免出错）。
- 文件的格式和数值的有效数字与VASP的输出文件CONTCAR保持一致。

KPOINTS

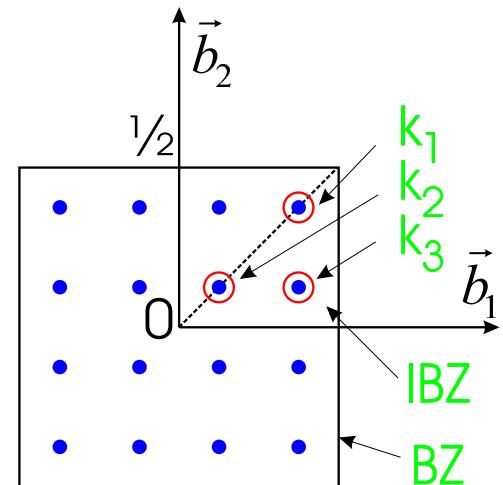
```

Automatic mesh      ! title
0      ! number of k-points = 0 ->automatic generation scheme
Gamma   ! generate a Gamma centered grid
4 4 4   ! subdivisions N_1, N_2 and N_3 along K vectors
0.0.0.  ! optional shift of the mesh (s_1, s_2, s_3)

```

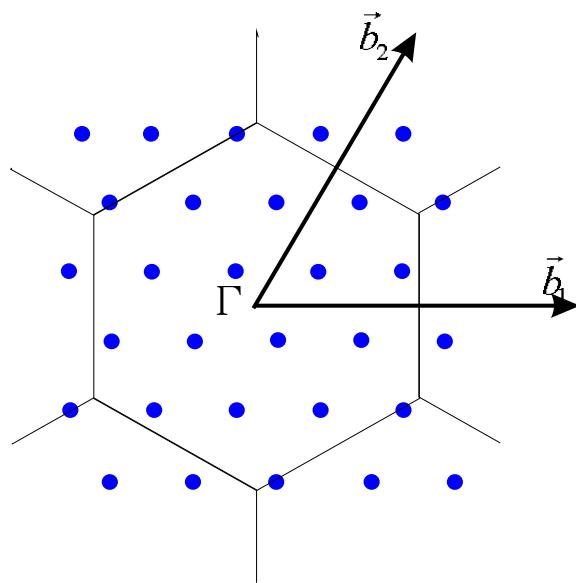


- 计算新系统先测试K点的收敛性
- K点越密越准确。
- 各个倒格矢方向取的各点数与倒格矢长度成正比，和正格矢长度成反比。
- 金属一般要比半导体和绝缘体取得密。
- 某个方向有真空则只在次方向上只取1。

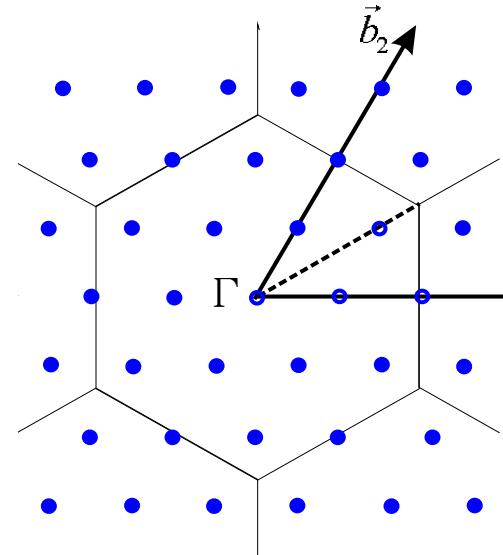


KPOINTS特例

三角晶格或者六角晶格



问题：偶数点阵不包含 Γ 点
将不具有系统的对称性！



解决方案：取奇数点阵，
或者偶数点阵包含 Γ 点。

INCAR

ENCUT: 平面波截断能。POTCAR会提供缺省值，一般不用修改。

EDIFF: 电子迭代收敛条件。一般1E-4。

EDIFFG: 离子收敛条件。-0.01到-0.03左右。符号表示原子受力。

ALGO: 电子迭代算法。一般用Normal比较保险。

IBRION: 离子演化算法。一般用2比较保险。

ENAUG: 设为ENCUT的3到4倍。

LREAL: 对小系统设F；对大系统设A，同时设ROPT=2E-4，有几种元素写几个。

NGX, NGY, NGZ: 一般不用设。只有在优化晶格或需要提高原子受力精度时。

在OUTCAR中找建议值：

WARNING: aliasing errors must be expected set NGX to 30 to avoid them

WARNING: aliasing errors must be expected set NGY to 30 to avoid them

WARNING: aliasing errors must be expected set NGZ to 234 to avoid them

ISMEAR, SIGMA: 结构弛豫对金属设1，半导体设0；静态计算设-5。细节参看手册。

NPAR: 控制并行。设为接近sqrt(Ncore)并且可以被Ncore整除的整数。

测试练习：

测试一下计算参数对计算速度及收敛性的影响

ENCUT, KPOINTS, (NGX, NGY, NGZ),

LREAL, (ISMEAR, SIGMA), ALGO,

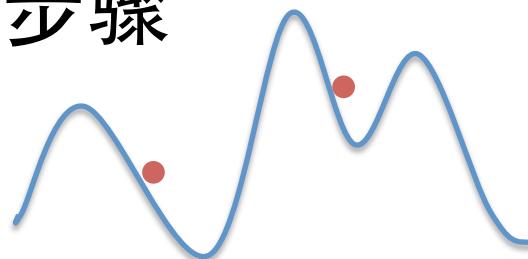
IBRION, NPAR

VASP计算一般步骤

构建结构
POSCAR



尽可能的构建接近于
最终稳定结构的构形。



根据研究的体系决
定是否要优化晶格



一般情况不用优化晶格。

晶格固定，弛豫
原子(relaxation)



已可得到可靠的系统的总能。

对弛豫后的结果计算
准确的电子结构



在需要得到态密度和能带的
情况下需要此步。

利用上一步产生的
CHGCAR计算能带图



KPOINTS为画能带的一系列
高对称线，通过非自洽计算
得到能带，其总能再无意义。

自开发小工具

lf: 列出当前目录下所有输入文件的重要参数和计算结果中的离子步

ldir [directories]: 列出当前目录下子目录的总能和磁矩

cin: 列出当前目录下所有子目录中INCAR的一些重要参数

ck: 列出当前目录下所有子目录中KPOINTS中取的格点数

cpt: 列出当前目录下所有子目录中POTCAR中一些标识参数

cps: 列出当前目录下所有子目录中POSCAR的前12行

clean: 清除当前目录下所有VASP输出文件，保留4个输入文件

cleancc: 清除当前目录下所有VASP输出文件，保留4个输入文件及CHGCAR

cf: 找出当前目录下所有子目录中没有收敛计算并打印最后一个离子步

cr: 打印当前目录下所有子目录中最后一个离子步

del [filename]: 删除当前目录及所有及目录下的所有指定文件

delv: 删除当前目录及所有及目录下的所有VASP产生的5个不常用文件，
vasprun.xml, WAVECAR, XDATCAR, PCDAT, CHG。

Linux常用命令

文件目录操作: ls, cd, rm, mkdir, cp, mv, chmod

显示文件: cat, more, less, head, tail

查找: grep, find

文件压缩: tar, gzip, gunzip

文件传输: scp, rsync (--max-size)

其它: bc, sed, awk

Linux基本概念: Shell, 管道 (pipeline) , 重定向 (redirection)

Linux环境变量: \$PATH, \$HOME, \$PS1, \$LD_LIBRARY_PATH

Linux环境设置文件 (alias, \$PATH): .bashrc, .bash_profile (隐藏文件)

Reference: <http://linuxcommand.org>

Vi编辑器

基本命令： search ‘vi quick reference’ in google.

© 1995, 1997, 1998 Phillip Farrell

VI Quick Reference -- Basic Commands

10/7/98

Notations and Conventions

Commands you type are in fixed font.
Italics = substitute desired value.
<CR> means press RETURN key.
'X' means press CONTROL and X keys together. Boxed commands switch to insert mode; press ESC key to end new text.

Beginning Your Edit Session

vi file <CR> edit or create file
vi -r <CR> show rescued files
vi -r file <CR> recover rescued file

Ending Your Edit Session

:q! <CR> quit and discard changes
:wq <CR> or ZZ quit and save changes
:wq new<CR> save as new and quit

Controlling Your Screen Display

^R Eliminate @ lines
^L Repaint screen after interruption
:set w=1 <CR> Auto word wrap at x chars before line end
:set nu <CR> Show line numbers on screen (not added to file)
:set nonu <CR> Stop showing line numbers on screen
(put set commands into .exrc file for automatic settings each time you start vi)

Moving the Cursor

h one position left
k one line up
j one line down
l (letter "ell") one position right
0 (zero) beginning of current line
\$ end of current line
w forward one word
W ... including punctuation
b backward one word
B ... including punctuation
e forward to end of current word
E ... including punctuation

- up one line, 1st non-blank char
+ or <CR>down line, 1st non-blank char
H beginning of first screen line
M beginning of middle screen line
L beginning of last screen line

Paging Through Text

^F forward one screen
^B backward one screen
^D scroll down half screen
^U scroll up half screen
nG move screen to line number n
G move screen to last line

Searching Through Text

/pattern forward search for pattern (regular expression syntax)
?pattern backward search for pattern (regular expression syntax)
n repeat search for next occurrence
N repeat search, reverse direction

Creating Text

Press ESC key to end new text. To enter a control character as text, precede it with ^V

a append text after cursor
A append text at end of current line
i insert text before cursor
I insert text before first non-blank character on current line
o open new line after current line
O open new line before current line
:r file <CR> insert contents of file after current line

Modifying Text -- Simple Changes

(* = can be preceded with a repeat count, e.g., 5x or 16dd. Count starts at cursor.)
x delete character at cursor *
dd delete line with cursor *
dw delete current word *
D delete to end of line.
rc change character at cursor to c

cw replace current word with new text *
cc replace entire current line *
C replace line from cursor to end
J join current line with next
~ change case of current character
u undo last text change
• repeat last text-change command (could be at new location)

Modifying Text -- Operators

(Can precede with repeat count. Double to affect whole lines, e.g., 5yy. Follow with one of the cursor movement or searching commands to select affected text, e.g., dw or c5w or y/Geology<CR>).
d delete
c change (replace with new text)
y yank (copy) to buffer
! filter selected text through command typed on status line, then replace with command output
<< shift line(s) left one tab position
>> shift line(s) right one tab position

Moving Text Around

(Use these commands to insert a copy of text from buffer. Deleted, replaced (old), or yanked text goes to unnamed buffer by default. Or use named buffer a thru z by prefixing command with " operator and buffer name, e.g., "a5yy and then "ap)
p copy buffer text after cursor or line
"ap ... using named buffer a (or b,c, etc)
P copy buffer text before cursor or line
"aP ... using named buffer a (or b,c, etc)
xp transpose characters

Global Text Substitution

:n,m s/old/new/g<CR>
Change all occurrences of regular expression old to text new on all lines n thru m. Can use symbolic line numbers ." (current line) or \$" (last line).

Vi编辑器

技巧 (Tips) : http://vim.wikia.com/wiki/Vim_Tips_Wiki

功能扩展: <http://www.vim.org/scripts/index.php>

把Vi变成一个编程开发环境 (IDE) :

<http://blog.csdn.net/wooin/article/details/1858917>

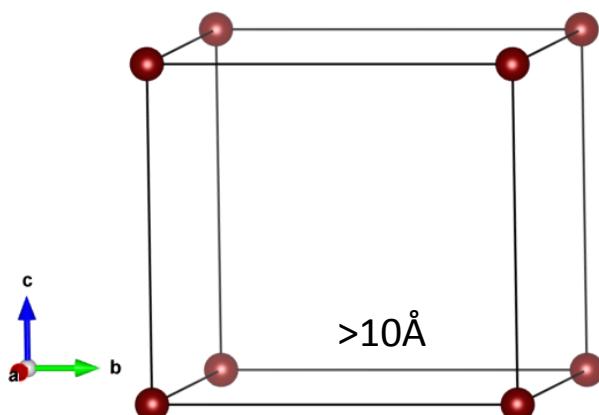
ctags: 与vi配合使用的一个
编程辅助工具

The screenshot shows a Vim-based IDE interface with several numbered callouts highlighting specific features:

- 1**: A red box highlights the status bar at the top right, which displays the URL <http://blog.csdn.net/wooin>.
- 2**: A red box highlights the left sidebar, which contains a file tree and a buffer list.
- 3**: A red box highlights the bottom-left corner of the code editor area.
- 4**: A red box highlights the bottom-right corner of the code editor area.
- 5**: A red box highlights the status bar at the bottom right, which displays the URL <http://blog.csdn.net/wooin>.
- 6**: A red box highlights the bottom center of the code editor area.

The code editor displays the file `main.c` with various C code snippets and syntax highlighting. The status bar at the bottom shows the file name and line numbers.

Example 1: a Single Atom



POSCAR:

atom in a box

```
1.000000000000000  
14.000000000000000 0.000000000000000 0.000000000000000  
0.000000000000000 13.000000000000000 0.000000000000000  
0.000000000000000 0.000000000000000 12.000000000000000
```

N

1

Selective dynamics

Direct

```
0.000000000000000 0.000000000000000 0.000000000000000 F F F
```

KPOINTS:

Gamma-point only

0

G

1 1 1

0 0 0

INCAR:

```
Spin-polarized  
ISPIN = 2  
MAGMOM = 3
```

EIGENVAL:

	spin-up	spin-down
1	-19.854828	-15.284319
2	-8.255876	-4.101525
3	-8.255614	-4.101398
4	-8.255561	-4.101392
5	-0.357830	-0.030598
6	0.407719	0.715501
7	0.554759	0.764094
8	0.651160	0.878340

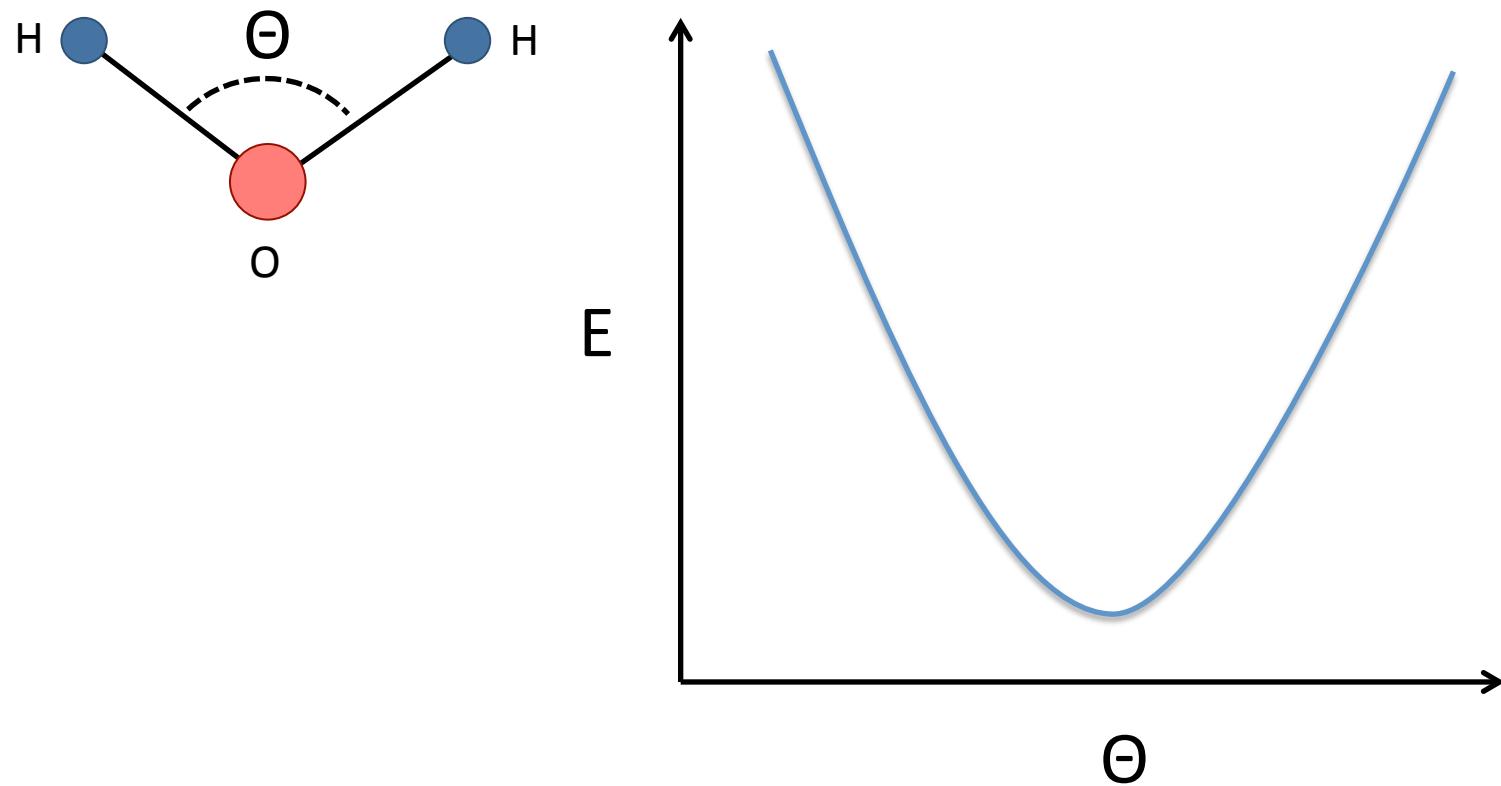
— s
} p

OSZICAR:

1 F= -.31150200E+01 E0= -.31150200E+01 d E=-.297179-207 mag= 3.0000

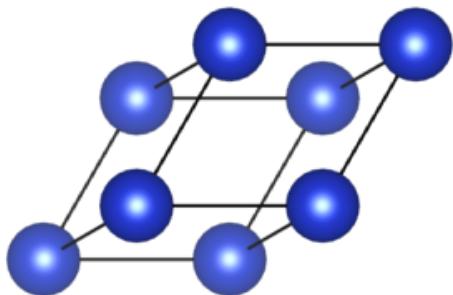
总磁矩

思考练习: 计算水分子能量和夹角的关系



Example 2: Lattice Optimization

Cu fcc bulk



KPOINTS:

21x21x21

方法1：

VASP自动优化晶格

参见：

Hands-on Session 2

方法2（推荐）：

计算一系列晶格常数值的系统总能，再通过拟合找到能量最低的最优晶格常数。

POSCAR:

Cu fcc 1x1x1

3.63

0.5000000000000000 0.5000000000000000 0.0000000000000000
0.0000000000000000 0.5000000000000000 0.5000000000000000
0.5000000000000000 0.0000000000000000 0.5000000000000000

Cu

1

Selective dynamics

Direct

0.0000000000000000 0.0000000000000000 0.0000000000000000 F F F

INCAR:

Electronic Relaxation

PREC = Normal

ENCUT = 280.00

EN AUG = 1000.00

ISMEAR = 1

SIGMA = 0.01

LREAL = False

ROPT = 2E-4 2E-4 2E-4 2E-4

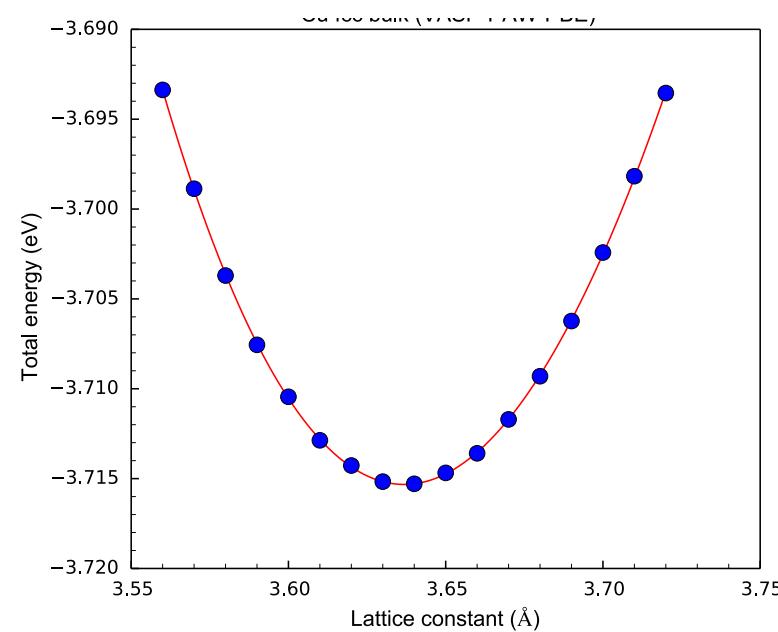
ALGO = Normal

NELM = 200

NELMIN = 3

EDIFF = 1E-06

NSW = 0



Example 2: Lattice Optimization

pbs.sh

```
#!/bin/sh -f

#PBS -N example2
#PBS -l nodes=1:ppn=4
#PBS -q ib
#PBS -e error

nprocs=`wc -l < $PBS_NODEFILE`
cd $PBS_O_WORKDIR

rm -f SUMMARY
for a in 3.56 3.58 3.60 3.62 3.64 3.66 3.68 3.70 3.72
do
    echo "a = $a"
    mkdir $a
    cp INCAR $a
    cp KPOINTS $a
    cp POTCAR $a
    sed s/LATTICE/$a/ POSCAR.0 > $a/POSCAR
    cd $a
    /opt/intel/impi/4.1.0.024/intel64/bin/mpirun -genv I_MPI_DEVICE rdma -np $nprocs /
opt/bin/vasp.5.2.12 > stdout
    E=`tail -1 OSZICAR | awk '{print $5}'`
    cd ..
    echo $a $E >>SUMMARY
done
```

这句话是将POSCAR.0中的LATTICE字符串改为循环变量\$a的值。因此使用时要将POSCAR改为POSCAR.0，并将其中要替换的数值改为LATTICE。Sed也可以替换指定的某一行，例见example 3。

Example 2: Lattice Optimization

```
[mu01:~/vasp-examples/2a-Cu_fcc_bulk_lattice]$ ls  
3.56 3.60 3.64 3.68 3.72 error      fit.dat KPOINTS POSCAR.0 SUMMARY  
3.58 3.62 3.66 3.70 AAID example2.o3750 INCAR  pbs.sh POTCAR
```

SUMMARY:

```
3.56 -.36944340E+01  
3.58 -.37048224E+01  
3.60 -.37116333E+01  
3.62 -.37155414E+01  
3.64 -.37153701E+01  
3.66 -.37137322E+01  
3.68 -.37094963E+01  
3.70 -.37026543E+01  
3.72 -.36937950E+01
```

用四次方多项式拟合得到最优晶格常数：

```
$ poly SUMMARY
```

```
Coefficients are:
```

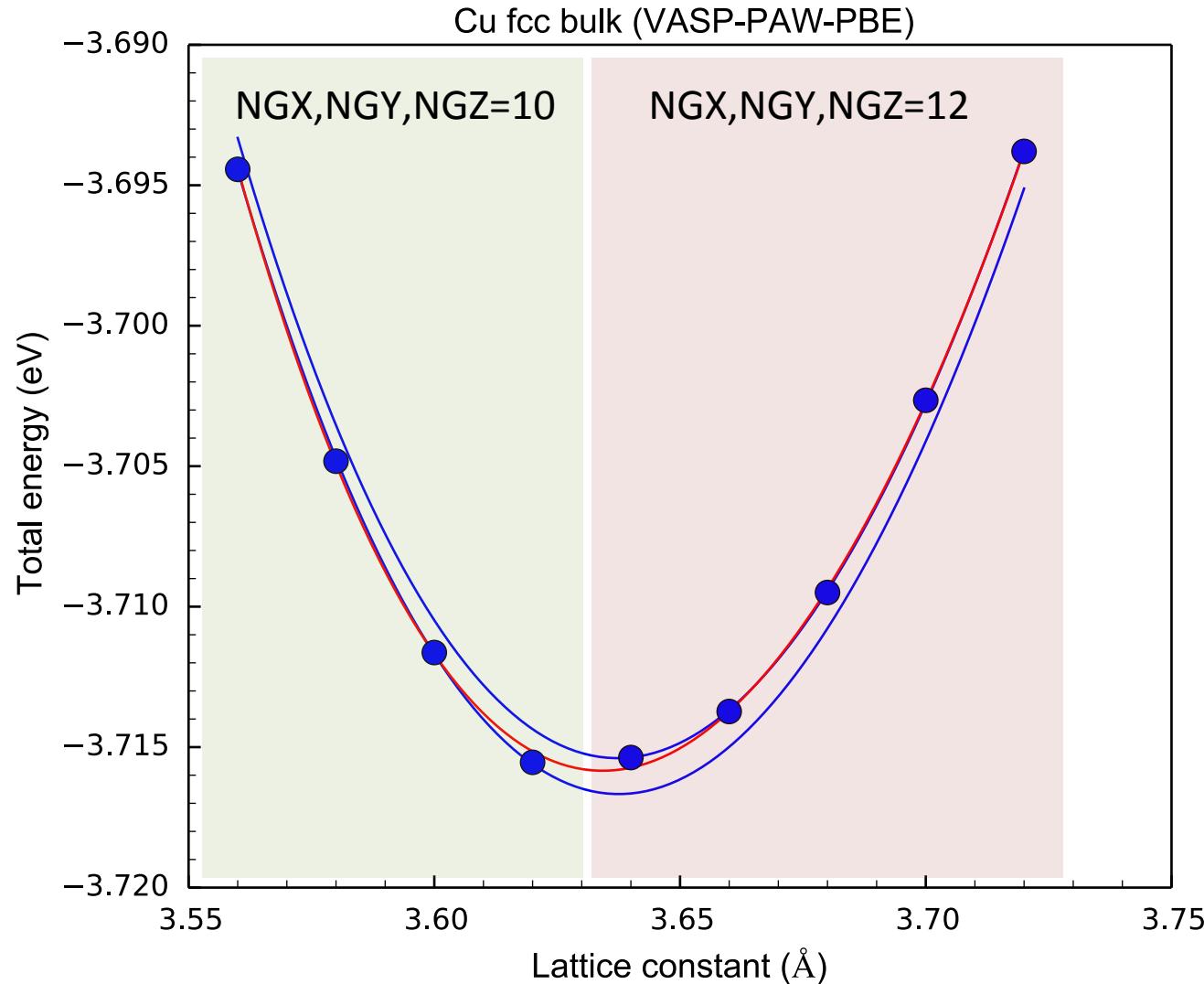
0	4803.666348
1	-5189.415604
2	2103.183293
3	-379.317309
4	25.688556

```
Standard deviation: 0.000215
```

Y Minimum is -3.715841 at [3.63424](#)

Example 2: Lattice Optimization

plot-lattice.py: 将SUMMARY和拟合的数据fit.dat画图



Example 2: Lattice Optimization

优化晶格需要设置NGX, NGY, NGZ!

OUTCAR:

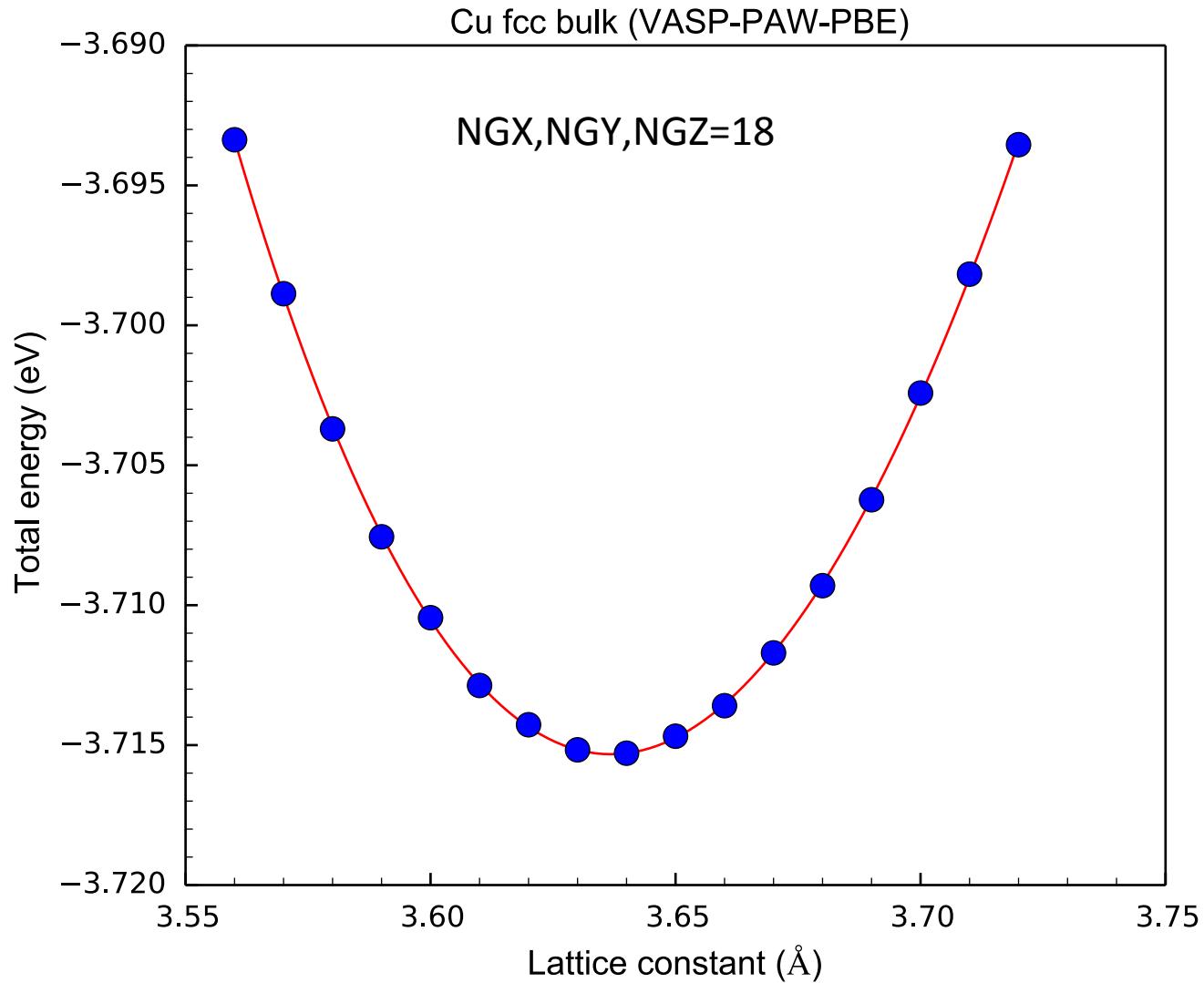
WARNING: aliasing errors must be expected set NGX to 18 to avoid them

WARNING: aliasing errors must be expected set NGY to 16 to avoid them

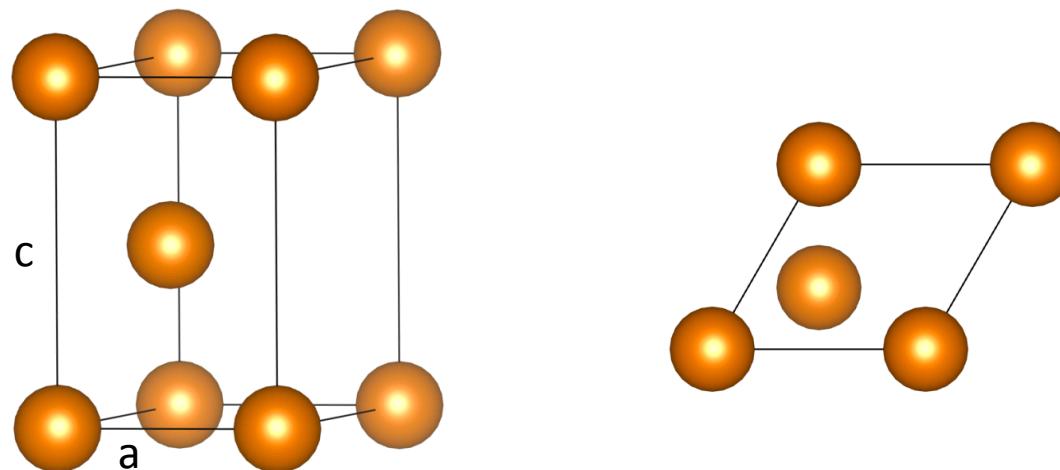
WARNING: aliasing errors must be expected set NGZ to 14 to avoid them

NGX, NGY, NGZ正比于ENCUT及基失的长度

Example 2: Lattice Optimization



思考练习: Mg hcp 结构的晶格常数



Mg

3.21

1.000000000000000	0.000000000000000	0.000000000000000
0.500000000000000	0.8660254037844386	0.000000000000000
0.000000000000000	0.000000000000000	1.620000000000000

Mg

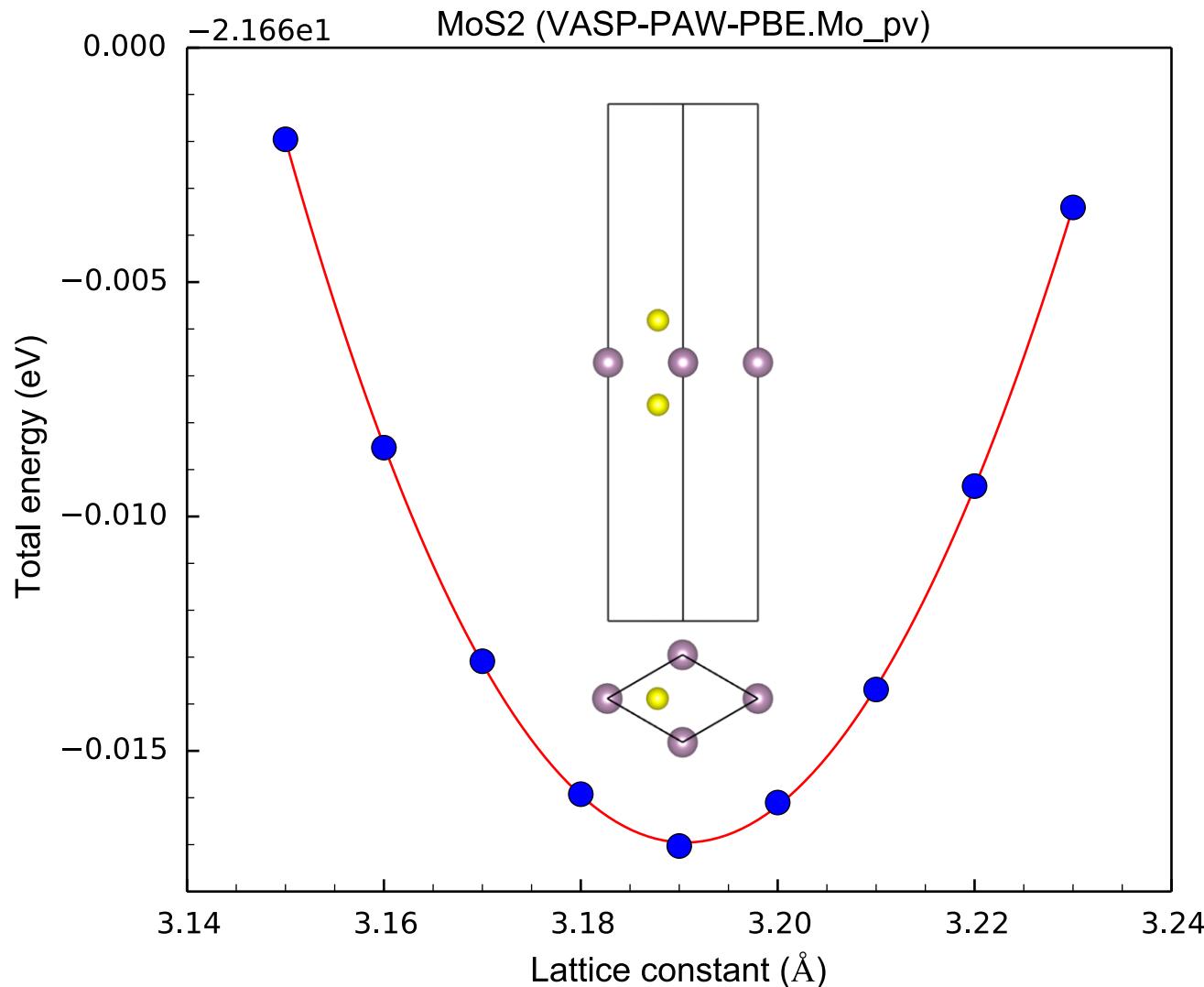
2

Selective dynamics

Direct

0.000000000000000	0.000000000000000	0.000000000000000	F	F	F
0.333333333333333	0.333333333333333	0.500000000000000	F	F	F

Example 3: MoS₂ Lattice Optimization



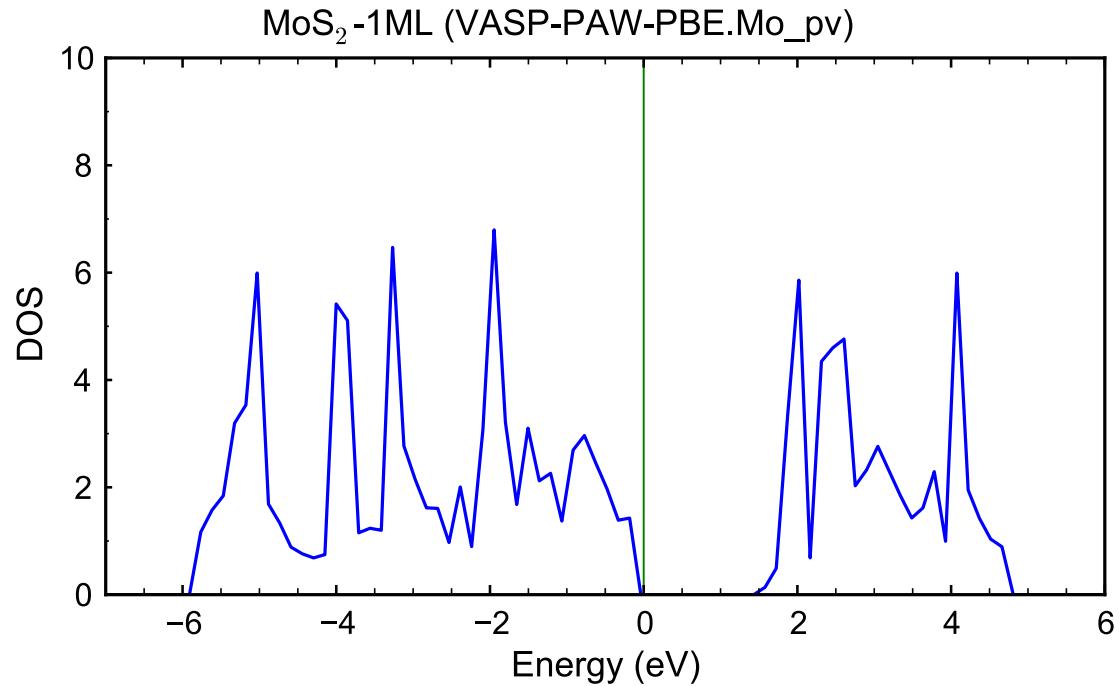
Example 5: MoS₂ Density of States

态密度输出到DOSCAR文件。得到DOSCAR不一定要另外做一步静态计算，结构弛豫计算结束后也会产生最后一步对应的态密度，但一般由于K点较少，得到的曲线很不光滑。更好的做法是再做一步静态计算得到CHGCAR，然后用密度更高的K点做一步非自洽计算(ICHARG=11)。

ISMEAR = -5 (0,1亦可)

LORBIT = 11 对每个原子
每个轨道进行投影，产生
PROCAR和DOSCAR中的局
域态密度。

画图工具: plot-dos.py



测试练习: 考察NEDOS, KPOINTS, (ISMEAR, SIGMA)对DOS曲线光滑度的影响

Example 6a: MoS₂ Band Structure

步骤：

1. 结构弛豫后做一步静态计算，设置：

LCHARG = T : 产生CHGCAR

NSW = 0 : 不弛豫

ISMEAR = -5

2. 通过非自治计算得到能带图，设置：

LCHARG = F

ICHARG = 11 : 非自治计算

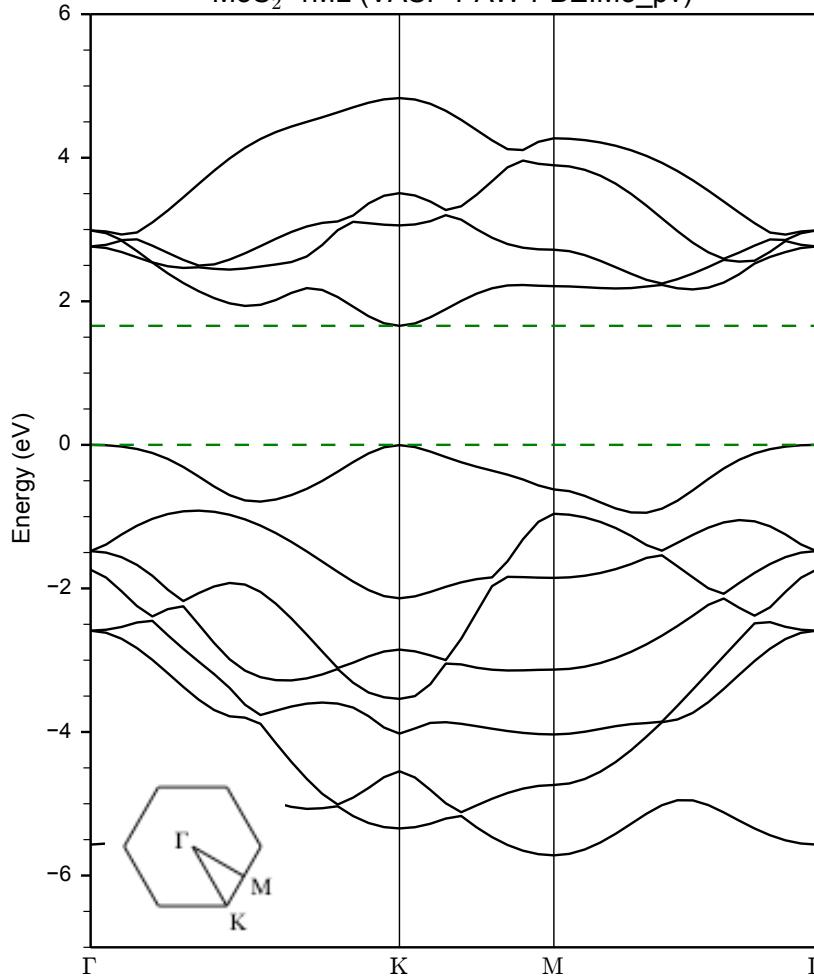
ISMEAR = 0 or 1

SIGMA = 0.01

同时需要上一步产生的CHGCAR，及沿着所画K空间路径的KPOINTS (K点产生工具:kpt-gen.py). 最后对EIGENVAL画图。

画图工具：plot-band.py

MoS₂-1ML (VASP-PAW-PBE.Mo_pv)



Example 6b: Projected Band Structure

计算步骤与一般计算能带相同。只是在INCAR中增加LORBIT = 11, 从而产生每个本征波函数对每个原子的每个轨道上的投影，相应的输出文件为PROCAR。而后利用画图工具plot-band-proj.py画出投影的能带结构。

plot-band-proj.py中的设置：

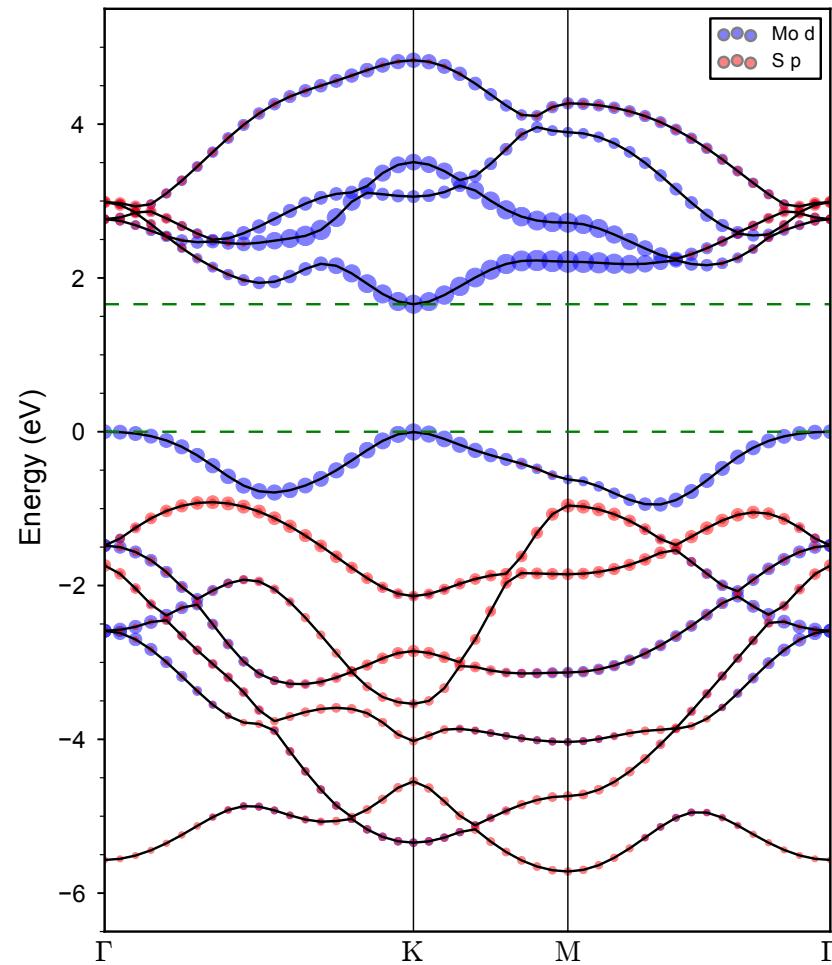
通过调用plot_projection子程序设置想要画的原子和轨道。例子中41, 42行：

```
41: plot_projection(proj, ev, [1],  
[5,6,7,8,9], eshift, 'b')  
42: plot_projection(proj, ev, [2,3],  
[2,3,4], eshift, 'r')
```

第一个[]中给出原子列表，第二个[]中给出轨道列表(具体轨道顺序可查看PROCAR文件)。所画的是[]中所有值的和。

画图工具：plot-band-proj.py

MoS₂-1ML (VASP-PAW-PBE.Mo_pv)



Example 7: Work Function

步骤:

1. 通过设置:

LVTOT = T

LVHAR = T

产生空间各点电势的文件LOCOTP。不需要额外的静态计算。

2. 利用glocpotc.py对LOCOTP中的数据在ab平面求平均, 输出如下:

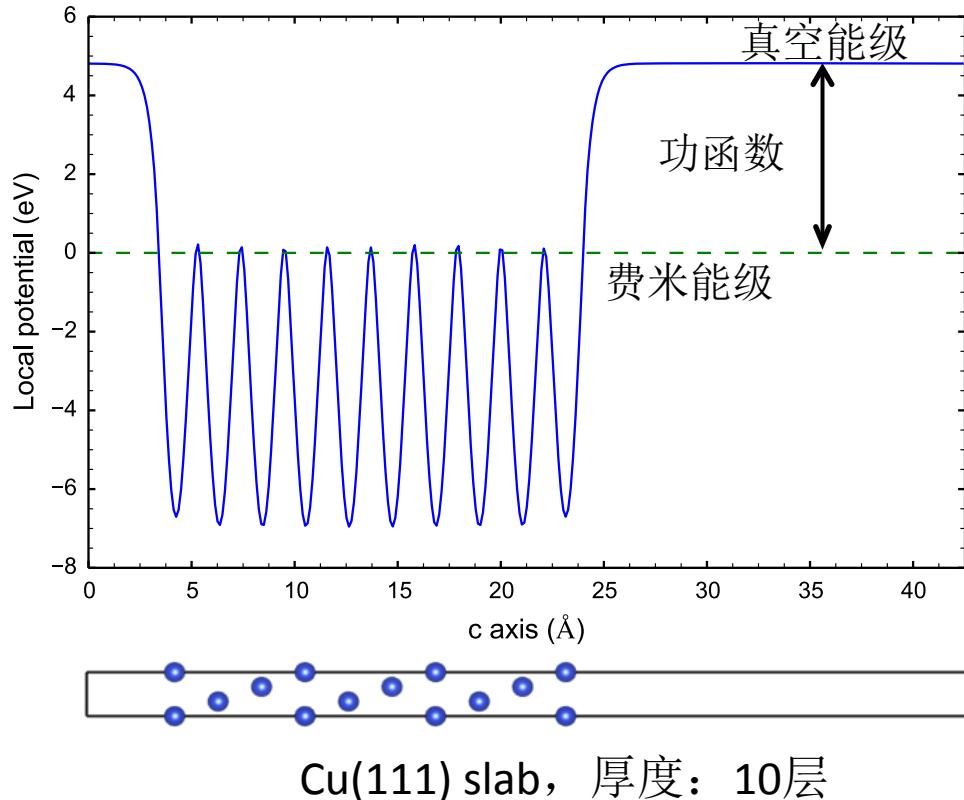
0	0.0000	4.218428	0.000007
1	0.0028	4.218442	-0.000471
2	0.0056	4.217487	-0.000563
3	0.0083	4.217316	-0.000783

依次为沿c方向格点; c方向位置(单位 \AA); ab面平均电势; 电势沿c方向1次倒数

如画图, 需将结果存入locpotc.dat文件(glocpotc.py > locpotc.dat)。

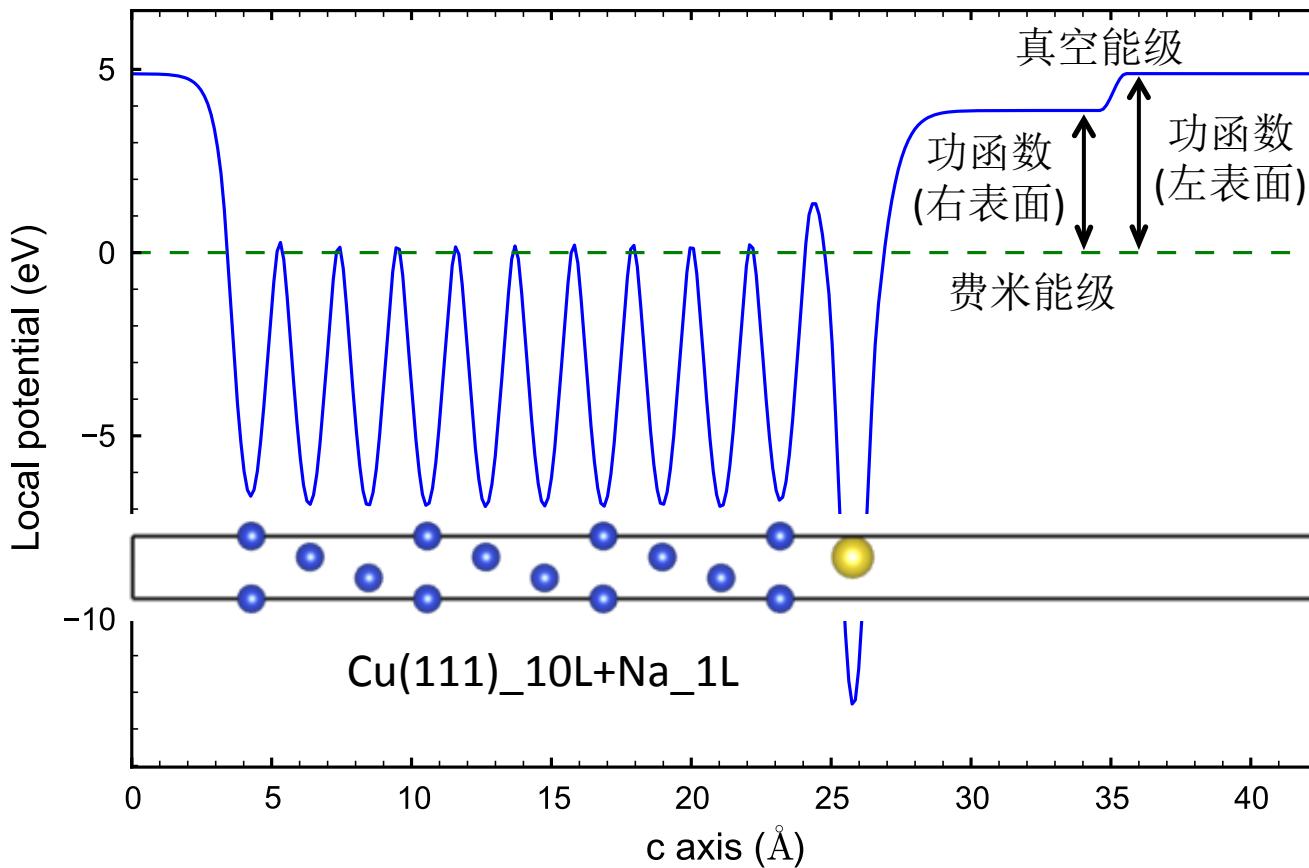
3. 利用plot_locpotc.py画图。

另注: 往往LOCOTP会很大, 可以先用glocpotc.py得到比较小的locpotc.dat文件, 然后将其传回本地画图。



Cu(111) slab, 厚度: 10层

Example 8: Work Function (asymmetric slab)



对于不对称的Slab，两个表面的功函数会不同，其本身也会具有一个电偶极矩，计算时需要加入dipole correction(LDIPOL = T; IDIPOL = 3)。从计算的电势分布图可以看到在真空区间电势有一个突变，其对应于计算中自动加入的一个电偶极矩层，两边的电势分别为于两个表面所对应的真空能级。

自开发小工具

gef: 得到当前目录下计算的费米能级

lef [directories]: 列出当前目录下各子目录的费米能级

gbandedges: 得到当前目录下计算的能带边界的信息

lbandedges: 列出当前目录下各子目录的能带边界的信息

例: -0.9197 0.8224 -0.1993 -1.0218 D1an-ZnSe2v.5
费米能级 能隙 导带底 价带顶 目录

ediff [dir1] [dir2]: 得到两个目录(dir1和dir2)所计算的总能的差

neighbors.py [POSCAR]: 计算POSCAR中每个原子与紧邻原子的距离

zspacing.py [POSCAR]: 计算POSCAR中在c方向原子层间的距离

例: MoS₂

```
Length of z axis: 19.14372
S 3: 0.333333 0.666667 0.581763
Mo 1: 0.000000 0.000000 0.500000
      z spacing: 0.081763 1.56524 层间距, 分别为以
      z spacing: 0.081763 1.56524 相对坐标和Å为单位
```

```
S 2: 0.333333 0.666667 0.418237
      z spacing: 0.836475 16.01324 最后一行即为
原子坐标, 从上到下排列      z spacing: 0.836475 16.01324 真空层厚度
```

简单画图程序：plot.py

用途：根据数据文件绘制简单的x-y曲线图。

用法：plot.py data_file [y] [symbol]

第一个参数为数据文件的文件名，为必需参数。数据需以列的方式存储，中间以空格分开，至少两列，可以多列。第一列定义为x，其它列为y。文件中可含有非数据的行，程序将会对此自动忽略。

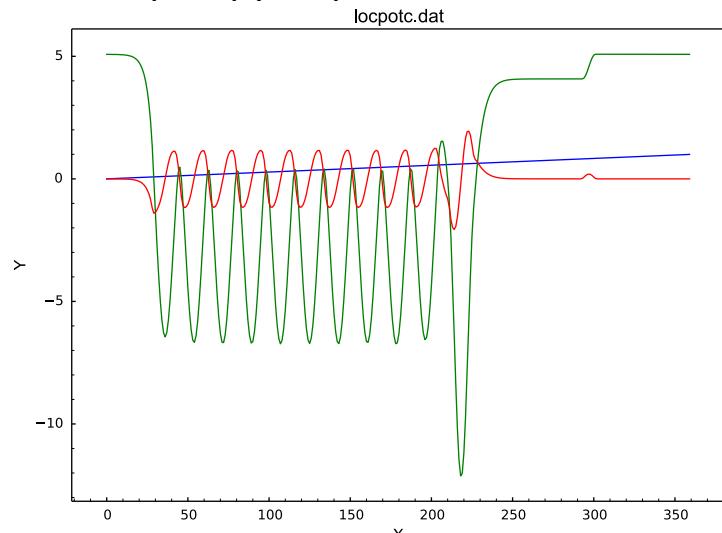
第二个参数y可指定对某一列y作图。此参数为可选参数。如空缺，程序将画出所有的曲线。

第三个参数symbol指定绘制曲线的类型，为可选。允许的设置为：

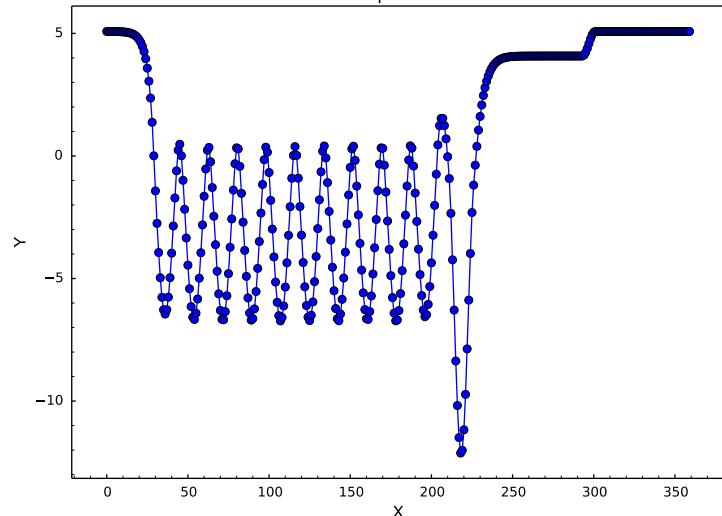
-：线；o：点；o-：点加线。默认为线。

输出：plot.pdf

例：plot.py locpotc.dat



例：plot.py locpotc.dat 2 o-
locpotc.dat



K点产生程序：kpt-gen.py

用途：产生用于能带计算中沿一系列布里渊区高对称线的K点，且间距一致。

输入文件：kpt-gen.in

例：`bands_num_points = 17` 第一条线取的各点数

```
begin kpoint_path      每条线的两端的坐标，定义为相对于倒格矢
    M 0.50000000 0.00000000 0.00000000  G 0.00000000 0.00000000 0.00000000
    G 0.00000000 0.00000000 0.00000000  K 0.33333333 0.33333333 0.00000000
    K 0.33333333 0.33333333 0.00000000  M 0.50000000 0.00000000 0.00000000
end kpoint_path
```

```
begin reciprocal_lattice_vectors      三个倒格矢，可在OUTCAR中找到
    0.116526227 0.000000000 0.000000000
    0.058263113 0.100914673 0.000000000
    0.000000000 0.000000000 0.030898682
end reciprocal_lattice_vectors
```

运行完产生kpt文件，与KPOINTS格式相同。同时屏幕输出给每高对称点及其位置，如：`['M', 'G', 'K', 'M']` 可方便用于设置plot-band.py。
`[1, 18, 38, 48]`

结构构造工具：cut

用途：从一个大的块体中切出一个任意的平行六面体单元。主要用于构造表面计算的slab模型。此程序适用于任何晶格结构，同时既可以产生平整的表面，也可以产生带有台阶的表面。

主要流程：

1. 准备一个POSCAR格式的单包结构文件。用read_poscar子程序读入。
2. 用stacking子程序堆建成一个大的块体。可通过产生的bulk.xyz查看。
3. 指定表面的方向(direction)，及新supercell原点的原子(atom_0)
4. 查看bulk.xyz，其中标注为蓝色的为原点原子，红色的为与原点原子同在指定平面及平面垂直线上的原子。从中找出abc三个基失顶点的原子，分别赋给atom_a, atom_b, atom_c。
5. 对切出的结构(POSCAR)利用rescale子程序留出适当的真空层。
6. 可利用shift子程序整体移动原子在supercell中的位置。
7. 利用sort_column_all子程序对原子从上到下排序。
8. 利用set_selective_flags子程序设置底下要固定的层数。

另注：目前通过顶点原子确定基失只是确定新的supercell的一种常用方式。此程序确定supercell与cut过程分开，因此如有需要也可自己发展新的确定supercell的方式。一旦supercell确定，可以利用同样地cut3d子程序进行切割。

结构构造工具：combine

用途: 将几个不同的结构组合到一个supercell中。

主要流程:

1. 计算可将不同结构组合在一起的方式，及各自的周期。
2. 准备所要组合结构的单包文件(POSCAR格式)。用read_poscar子程序分别读入。
3. 用stacking子程序分别将每种结构堆建成所要的周期。可分别输出到POSCAR查看。
4. 估算出组合后新的supercell所需的c轴长度，利用rescale子程序将各结构的c轴调整到所需的长度。
5. 可利用shift子程序分别对不同的结构整体移动原子在supercell中的位置。
6. 可利用sort_column_all子程序对原子从上到下排序。
7. 利用write_poscar将不同结构的原子组合到一个supercell中，输出到POSCAR。