Tutorial VASP — Aula 2

Bulk modulus, estrutura de bandas e DOS do silício

Material de aula — Ilum / CNPEM

Visão geral da aula

Na aula anterior vocês rodaram um SCF de ponto único do silício bulk e aprenderam a anatomia dos arquivos de entrada e saída. Agora vamos extrair propriedades físicas mensuráveis comparáveis com o experimento. A aula tem três objetivos principais:

1. Otimizar o parâmetro de rede do silício e, a partir da curva E(V), extrair o módulo de bulk B_0 via ajuste à equação de estado de Birch-Murnaghan.
2. Calcular a estrutura de bandas eletrônicas do silício ao longo de um caminho de alta simetria L-Γ-X-U|K-Γ na zona de Brillouin.
3. Calcular a densidade de estados (DOS) total e a densidade de estados projetada nos orbitais s e p (PDOS), entendendo a contribuição de cada caráter orbital às bandas de valência e condução.

Esses três cálculos formam o repertório básico que vocês usarão em qualquer estudo de sólidos: equação de estado dá acesso a propriedades termodinâmicas, bandas mostram a natureza eletrônica (metal, semicondutor, isolante) e o DOS conecta a estrutura eletrônica com espectroscopia e ligação química.

1. Organização do trabalho

Como vamos rodar muitos cálculos relacionados, organizem desde o início. Criem a seguinte estrutura no Heisenberg:

```
cd $SCRATCH/vasp_tutorial
mkdir si_bulk_modulus si_bandas si_dos
ls -la
```

Os três diretórios serão usados em sequência. Vamos compartilhar arquivos entre eles, então mantenham tudo no mesmo diretório-pai.

2. Parte I — Otimização e módulo de bulk

2.1. Teoria mínima

O módulo de bulk B_0 mede a resistência de um material à compressão volumétrica isotrópica. Por definição:

$$B = V\left(\frac{\partial^2 E}{\partial V^2}\right)_{T,S} = -V\left(\frac{\partial P}{\partial V}\right)_T$$

Para extrair B_0 de um cálculo DFT, calculamos a energia total para uma série de volumes diferentes em torno do equilíbrio e ajustamos a curva E(V) a uma equação de estado. A mais usada é a de Birch-Murnaghan de terceira ordem:

$$P(V) = \frac{3B_0}{2}\left[\left(\frac{V_0}{V}\right)^{7/3} - \left(\frac{V_0}{V}\right)^{5/3}\right]\left\{1 + \frac{3}{4}\left(B_0' - 4\right)\left[\left(\frac{V_0}{V}\right)^{2/3} - 1\right]\right\}.$$

Quatro parâmetros saem do ajuste: E_0 (energia de equilíbrio), V_0 (volume de equilíbrio), B_0 (módulo de bulk) e B_0' (sua derivada com a pressão). Para o silício, valores experimentais são $B_0 \approx 97.9$ GPa e $B_0' \approx 4.1$; PBE costuma subestimar B_0 em ~10%.

Dica: Existem duas formas equivalentes de fazer essa otimização: (a) manualmente, rodando vários SCFs em volumes diferentes e ajustando a curva E(V); (b) automaticamente, deixando o VASP relaxar a célula com ISIF = 3. Vamos fazer a abordagem manual primeiro, porque é mais didática e permite ver a curva. Ela também é mais confiável: a abordagem ISIF = 3 com ondas planas tem o problema das "tensões de Pulay" (Pulay stress) quando o volume muda em uma base de tamanho fixo.

2.2. Procedimento: varredura E(V)

A ideia é simples: rodar o mesmo SCF para 7-9 valores diferentes do parâmetro de rede em torno do experimental (5.43 Å), e plotar E versus V.

Passo 1: preparar os arquivos base

No diretório si_bulk_modulus, criem os arquivos comuns a todos os cálculos:

```
cd si_bulk_modulus

# INCAR (comum a todos os volumes)
cat > INCAR << 'EOF'
SYSTEM = Si - varredura E(V)
ISTART = 0
ICHARG = 2
ENCUT  = 400
PREC   = Accurate
EDIFF  = 1E-07
NELM   = 60
ALGO   = Normal
ISMEAR = 0
SIGMA  = 0.05
LREAL  = .FALSE.
NSW    = 0
IBRION = -1
LWAVE  = .FALSE.
LCHARG = .FALSE.
EOF

# KPOINTS (mesma malha para todos)
cat > KPOINTS << 'EOF'
K-points
0
Gamma
12 12 12
0 0 0
EOF

# POTCAR
module load vasp/6.2.0
cp $VASP_PP_PATH/potpaw_PBE.54/Si/POTCAR ./POTCAR
```

⚠️ **Atenção:** Aumentamos a malha de pontos-k de 8×8×8 para 12×12×12 nesta varredura porque queremos extrair derivadas da energia (que é o que B_0 representa). Quando se faz uma derivada numérica, qualquer ruído por convergência insuficiente é amplificado. Use sempre malhas mais densas e EDIFF mais apertado quando o que vocês querem é uma propriedade derivada.

Passo 2: gerar POSCARs em volumes diferentes

Em vez de criar 9 arquivos POSCAR manualmente, vamos usar um loop em bash. Cada cálculo terá seu próprio subdiretório. Para isso crie um arquivo script.bash com o seguinte conteúdo:

```
for a in 5.30 5.34 5.38 5.41 5.43 5.45 5.48 5.52 5.56
do
    mkdir a_$a
    cp INCAR KPOINTS POTCAR a_$a/
```

```
    # POSCAR com o fator de escala a
    cat > a_$a/POSCAR << EOF
Si diamante - a = $a Angstrom
   $a
     0.5  0.5  0.0
     0.0  0.5  0.5
     0.5  0.0  0.5
   Si
     2
Direct
  0.0  0.0  0.0
  0.25 0.25 0.25
EOF
done
ls -la
```

Executem o script com:

bash script.bash

Vocês terão 9 subdiretórios, cada um com seus quatro arquivos prontos.

Passo 3: script SLURM e submissão em lote

Criem um script SLURM no diretório-pai e submetam um job para cada volume. Vamos usar um script SLURM padrão similar ao da aula anterior, mas adaptado para ser submetido em loop:

```
cat > run_one.sh << 'EOF'
#!/bin/bash
#SBATCH -n 8
#SBATCH --ntasks-per-node=8
#SBATCH -p cpu
#SBATCH --mem=16000mb
#SBATCH -J Si_EV

module load vasp/6.2.0

hostname
export OMP_NUM_THREADS=1
mpirun -n 8 vasp_std > vasp.out
EOF
```

Crie um script para submeter todos os jobs:

```
# Submeter um job para cada volume
for a in 5.30 5.34 5.38 5.41 5.43 5.45 5.48 5.52 5.56
do
    cd a_$a
    sbatch ../run_one.sh
    cd ..
done
```

Acompanhe seus jobs com “squeue -u $USER”

Passo 4: extrair E(V) e ajustar Birch-Murnaghan

Quando todos os jobs terminarem (deve levar poucos minutos), coletem as energias e volumes:

```
echo "# a(Ang)   V(Ang^3)   E(eV)" > EV.dat
for a in 5.30 5.34 5.38 5.41 5.43 5.45 5.48 5.52 5.56
do
    E=$(grep 'free  energy   TOTEN' a_$a/OUTCAR | tail -1 | awk '{print $5}')
    V=$(grep 'volume of cell' a_$a/OUTCAR | tail -1 | awk '{print $5}')
```

```
    echo "$a $V $E" >> EV.dat
done
cat EV.dat
```

Agora façam o ajuste em Python. Criem o script fit_BM.py no mesmo diretório:

```
import numpy as np
from scipy.optimize import curve_fit
import matplotlib.pyplot as plt

def birch_murnaghan(V, E0, V0, B0, B0p):
    """Birch-Murnaghan EOS de 3a ordem. B0 em eV/A^3."""
    eta = (V0/V)**(2.0/3.0)
    return E0 + (9.0*V0*B0/16.0) * ((eta - 1)**3 * B0p
            + (eta - 1)**2 * (6.0 - 4.0*eta))

# Carregar dados
data = np.loadtxt('EV.dat')
a, V, E = data[:,0], data[:,1], data[:,2]

# Chutes iniciais razoaveis
p0 = [E.min(), V[E.argmin()], 0.6, 4.0]   # 0.6 eV/A^3 ~ 96 GPa
popt, pcov = curve_fit(birch_murnaghan, V, E, p0=p0)
E0, V0, B0, B0p = popt

# Converter B0 de eV/A^3 para GPa
B0_GPa = B0 * 160.21766208
a0 = (V0 * 4)**(1.0/3.0)   # a^3 = 4*V_prim, pois 4 atomos na conv. fcc

print(f"E0 = {E0:.4f} eV")
print(f"V0 = {V0:.4f} A^3")
print(f"a0 = {a0:.4f} A (parametro de rede da celula convencional)")
print(f"B0 = {B0_GPa:.2f} GPa")
print(f"B0' = {B0p:.3f}")

# Plotar
Vfit = np.linspace(V.min(), V.max(), 200)
Efit = birch_murnaghan(Vfit, *popt)
plt.figure(figsize=(7,5))
plt.plot(V, E, 'o', label='VASP')
plt.plot(Vfit, Efit, '-', label=f'BM3 (B0={B0_GPa:.1f} GPa)')
plt.xlabel('Volume (A^3)')
plt.ylabel('Energia (eV)')
plt.legend(); plt.grid(alpha=0.3)
plt.tight_layout()
plt.savefig('EV_fit.png', dpi=150)
print("Plot salvo em EV_fit.png")
```

Rodem com:

```
python3 fit_BM.py
```

Resultado esperado para o silício com PBE: $V_0 \approx 40.5$ Å³ (ou seja, $a_0 \approx 5.47$ Å), $B_0 \approx 88$-92 GPa, $B_0' \approx 4.2$. Comparem com os valores experimentais ($a_0 = 5.431$ Å, $B_0 = 97.9$ GPa) e discutam o desvio.

Dica: PBE superestima sistematicamente os parâmetros de rede de sólidos (em geral 1-2 %) e, consequentemente, subestima B_0 em 5-10 %. Isso é uma propriedade conhecida do funcional, não um erro de cálculo. Funcionais como PBEsol ou meta-GGAs (SCAN) corrigem em parte esse problema.

3. Parte II — Estrutura de bandas e DOS

O cálculo da estrutura de bandas e do DOS segue uma sequência de dois passos que vocês precisam internalizar bem, porque ela vai aparecer em quase todos os estudos de estado sólido:

4. Cálculo SCF de alta qualidade para obter uma densidade de carga (CHGCAR) bem convergida em uma malha densa e regular de pontos-k.
5. Cálculo não-autoconsistente (ICHARG = 11), que lê o CHGCAR fixo e calcula apenas as energias e funções de onda em uma nova distribuição de pontos-k — sejam eles um caminho na zona de Brillouin (para bandas) ou uma malha ainda mais densa (para DOS preciso).

A razão para essa separação é discutida em detalhe nos slides teóricos. Em resumo: para bandas, os pontos-k não formam uma malha regular e um SCF direto sobre eles daria uma densidade fisicamente incorreta. Para DOS, queremos congelar a densidade obtida com uma malha confiável e apenas refinar a amostragem das energias.

3.1. O cálculo SCF base

Esta etapa será compartilhada por bandas e DOS. Façamos no diretório si_bandas (depois copiaremos o CHGCAR para si_dos):

```
cd ../si_bandas

# Use o parametro de rede otimizado da Parte I (ajuste para o valor que voces obtiveram)
cat > POSCAR << 'EOF'
Si diamante - a otimizado
   5.4680
     0.5  0.5  0.0
     0.0  0.5  0.5
     0.5  0.0  0.5
   Si
     2
Direct
  0.0  0.0  0.0
  0.25 0.25 0.25
EOF

cat > INCAR << 'EOF'
SYSTEM = Si - SCF para gerar CHGCAR
ISTART = 0
ICHARG = 2
ENCUT  = 400
PREC   = Accurate
EDIFF  = 1E-07
NELM   = 100
ALGO   = Normal

! Tetraedros para SCF preciso em semicondutor
ISMEAR = -5

LREAL  = .FALSE.
NSW    = 0
IBRION = -1

! IMPORTANTE: salvar CHGCAR para usar nos proximos passos
LWAVE  = .FALSE.
LCHARG = .TRUE.
EOF

cat > KPOINTS << 'EOF'
K-points SCF
0
Gamma
12 12 12
0 0 0
EOF

module load vasp/6.2.0
```

```
cp $VASP_PP_PATH/potpaw_PBE.54/Si/POTCAR ./POTCAR

# Copiar o script SLURM da aula anterior
cp ../si_bulk_modulus/run_one.sh ./
sbatch run_one.sh
```

⚠️ **Atenção:** Usamos ISMEAR = -5 (tetraedros) aqui porque é o método mais preciso para semicondutores e isolantes em malhas regulares. Ele exige no mínimo 4 pontos-k irredutíveis — para um sistema cúbico com malha 12×12×12, sobra muita margem.

Quando terminar, confiram a convergência e a existência do CHGCAR:

```
tail -5 OSZICAR    # ver energia final
ls -lh CHGCAR      # deve existir e ter alguns MB
```

3.2. Estrutura de bandas

Passo 1: o caminho na zona de Brillouin

A estrutura de bandas é uma "fatia" do dispersão E(k) ao longo de um caminho passando por pontos de alta simetria. Para a estrutura diamante (espaço FCC), o caminho convencional é L → Γ → X → U|K → Γ. Os pontos de alta simetria têm coordenadas fracionárias bem definidas na zona de Brillouin:

- **L** = (½, ½, ½)
- **Γ** = (0, 0, 0)
- **X** = (0, ½, ½)
- **U** = (¼, ⅝, ⅝)
- **K** = (⅜, ¾, ⅜)

U e K estão conectados por uma operação de simetria do grupo do diamante, por isso o caminho "salta" de U para K sem custar uma aresta no gráfico — escrevemos U|K para denotar essa identificação.

Passo 2: criar o diretório de bandas com CHGCAR

```
mkdir bands && cd bands
cp ../CHGCAR .
cp ../POSCAR .
cp ../POTCAR .
cp ../run_one.sh .

cat > INCAR << 'EOF'
SYSTEM = Si - estrutura de bandas
ICHARG = 11           ! NAO-SCF: ler CHGCAR fixo
ISTART = 0
ENCUT  = 400
PREC   = Accurate
LREAL  = .FALSE.

! Para bandas, ISMEAR = 0 (Gauss); tetraedros nao se aplicam
! a caminhos nao-regulares
ISMEAR = 0
SIGMA  = 0.05

NSW    = 0
IBRION = -1
LWAVE  = .FALSE.
LCHARG = .FALSE.

! Numero de bandas: por padrao VASP escolhe automaticamente,
! mas para estrutura de bandas e bom ter algumas a mais
NBANDS = 16
```

```
EOF

cat > KPOINTS << 'EOF'
Caminho L-G-X-U|K-G para Si diamante
40
Line-mode
Reciprocal
  0.50000  0.50000  0.50000   ! L
  0.00000  0.00000  0.00000   ! Gamma

  0.00000  0.00000  0.00000   ! Gamma
  0.00000  0.50000  0.50000   ! X

  0.00000  0.50000  0.50000   ! X
  0.25000  0.62500  0.62500   ! U

  0.37500  0.75000  0.37500   ! K
  0.00000  0.00000  0.00000   ! Gamma
EOF

sbatch run_one.sh
```

O formato do KPOINTS para bandas é diferente do que vocês viram antes. As linhas significam:

- **Linha 1:** comentário.
- **Linha 2:** número de pontos por segmento (40 aqui — bom para gráficos lisos).
- **Linha 3:** "Line-mode" — modo de geração de pontos ao longo de linhas.
- **Linha 4:** "Reciprocal" — coordenadas dos pontos em base recíproca (fracionárias).
- **Pares de linhas seguintes:** início e fim de cada segmento do caminho. Linha em branco separa segmentos.

⚠️ **Atenção:** Note a interrupção entre X-U e K-Γ. O salto U→K não tem segmento explícito porque eles estão conectados por simetria (operação do grupo espacial Fd-3m do diamante). Na ferramenta de visualização, o segmento U|K aparece como uma única descontinuidade vertical no gráfico.

Passo 3: visualizar as bandas

Rodem o script abaixo no diretório *bands*, após terminar o cálculo:

```
from pymatgen.io.vasp.outputs import Vasprun, BSVasprun
from pymatgen.electronic_structure.plotter import BSPlotter
import matplotlib
matplotlib.use("Agg")    # backend sem GUI (para HPC)

# BSVasprun é otimizado: lê só os dados de banda do vasprun.xml
run = BSVasprun("vasprun.xml", parse_projected_eigen=False)
bs = run.get_band_structure(line_mode=True)

plotter = BSPlotter(bs)
plotter.save_plot("Si_bandas.png")
print(f"Gap = {bs.get_band_gap()['energy']:.3f} eV")
print(f"Direto? {bs.get_band_gap()['direct']}")
print("Bandas salvas em Si_bandas.png")
```

Este script irá gerar um arquivo com as bandas de energia.

Dica: O silício é um semicondutor de gap indireto. No gráfico de bandas, vocês devem ver o máximo da banda de valência em Γ e o mínimo da banda de condução próximo a X (a cerca de 85% do caminho Γ-X). O gap calculado com PBE estará em torno de 0.6 eV — bem menor que o experimental de 1.17 eV. Esse é o famoso *gap problem* do DFT-GGA, do qual falaremos nas próximas aulas.

3.3. Densidade de estados (DOS) e PDOS

Para o DOS, voltamos ao diretório si_dos e fazemos um cálculo similar ao de bandas, mas com uma malha regular e densa de pontos-k em vez de um caminho:

```
cd ../../si_dos

cp ../si_bandas/CHGCAR .
cp ../si_bandas/POSCAR .
cp ../si_bandas/POTCAR .
cp ../si_bandas/run_one.sh .

cat > INCAR << 'EOF'
SYSTEM = Si - DOS e PDOS
ICHARG = 11           ! NAO-SCF: ler CHGCAR fixo
ISTART = 0
ENCUT  = 400
PREC   = Accurate
LREAL  = .FALSE.

! Tetraedros para DOS preciso
ISMEAR = -5

! Projecao em orbitais para PDOS
LORBIT = 11           ! 11 = projeta em s, px, py, pz, dxy, ...
                      ! Usa raios de Wigner-Seitz padrao do POTCAR

! Resolucao do DOS
NEDOS  = 601          ! 601 pontos de energia
EMIN   = -15          ! eV, absoluto (nao relativo a Ef)
EMAX   =  15          ! eV

NSW    = 0
IBRION = -1
LWAVE  = .FALSE.
LCHARG = .FALSE.
EOF

cat > KPOINTS << 'EOF'
K-points densos para DOS
0
Gamma
20 20 20
0 0 0
EOF

sbatch run_one.sh
```

⚠️ **Atenção:** Para o DOS, usamos uma malha bem mais densa (20×20×20) do que para o SCF. Isso é o lado bom de usar ICHARG = 11: como a densidade está congelada, podemos mudar a amostragem livremente sem que isso afete a autoconsistência. O custo é só do cálculo dos autovalores, que é barato.

Visualizando o DOS e PDOS

```
from pymatgen.io.vasp.outputs import Vasprun
from pymatgen.electronic_structure.plotter import DosPlotter
from pymatgen.electronic_structure.core import OrbitalType
import matplotlib
matplotlib.use("Agg")

run = Vasprun("vasprun.xml", parse_dos=True)
cdos = run.complete_dos

# DOS total
plotter_total = DosPlotter()
```

```
plotter_total.add_dos("Total", cdos)
plotter_total.save_plot("Si_DOS_total.png", xlim=[-13, 8])

# PDOS por orbital (s e p somados sobre todos os atomos de Si)
spd = cdos.get_spd_dos()
plotter_pdos = DosPlotter()
plotter_pdos.add_dos("s", spd[OrbitalType.s])
plotter_pdos.add_dos("p", spd[OrbitalType.p])
plotter_pdos.save_plot("Si_PDOS.png", xlim=[-13, 8])

print(f"E-Fermi = {cdos.efermi:.3f} eV")
print("DOS e PDOS salvos.")
```

Para entender o que vocês vão ver:

- Banda de valência (energia < 0, ou seja, abaixo do nível de Fermi): tem três sub-bandas. A mais baixa é majoritariamente caráter s, as duas mais altas são majoritariamente caráter p. Isso reflete a hibridização sp^3 do silício.
- Banda de condução (E > gap): mistura forte de s e p, com s dominando na parte mais baixa em alguns intervalos de energia.
- Gap: para PBE, vocês verão um gap de cerca de 0.6 eV separando as duas regiões.

Quem quiser explorar mais esses gráficos, pode tentar usar a biblioteca sumo: https://github.com/SMTG-Bham/sumo.

4. Análise física dos resultados

Depois de gerar os três conjuntos de gráficos (E(V) com ajuste BM, estrutura de bandas, DOS+PDOS), discutam as seguintes questões. Elas serão pedidas no relatório:

6. O parâmetro de rede otimizado pelo PBE é maior ou menor que o experimental? Por quanto? Como esse desvio se compara com a precisão experimental?
7. O módulo de bulk calculado está dentro de 10% do valor experimental? Compare seu valor com B_0 experimental (97.9 GPa) e discuta o desvio à luz da aproximação GGA-PBE.
8. A estrutura de bandas mostra um gap direto ou indireto? Onde estão o topo da banda de valência e o fundo da banda de condução? Como o gap calculado se compara com o experimental (1.17 eV)? Por que o DFT-PBE subestima sistematicamente o gap?
9. No PDOS, identifiquem a janela de energia onde a banda de valência tem caráter quase puramente s e onde ela é quase puramente p. Como esses intervalos correspondem às sub-bandas que vocês veem no gráfico de bandas?
10. Calculem o número total de elétrons de valência por célula primitiva integrando o DOS total até o nível de Fermi (na prática, isso aparece direto como “number of electrons” no OUTCAR). Confiram que dá 8 (cada Si contribui com 4 elétrons de valência, e há 2 átomos por célula).

5. Apêndice: dicas operacionais

5.1. Sobre os tetraedros (ISMEAR = -5)

ISMEAR = -5 é o método mais preciso para semicondutores e isolantes em malhas regulares, mas tem três limitações importantes:

- Exige no mínimo 4 pontos-k irredutíveis. Falha em sistemas pequenos demais ou com simetria muito alta.

- Não dá forças e tensões confiáveis. NÃO usar em otimizações de geometria — sempre voltar para ISMEAR = 0 ou 1 nesses casos.
- Falha em caminhos lineares de pontos-k. Para bandas, sempre usar ISMEAR = 0.

5.2. Sobre o LORBIT

LORBIT = 10 projeta nos canais l (s, p, d, f como um único bloco cada). LORBIT = 11 projeta nos canais lm individuais (s, px, py, pz, dxy, etc.). Para o silício, LORBIT = 10 já basta — vocês querem s e p totais, não as projeções m-resolvidas. Mas LORBIT = 11 também funciona; só gera um DOSCAR maior.

5.3. Convergência do nível de Fermi

Em todos os cálculos, o VASP imprime o nível de Fermi no OUTCAR. Para semicondutores, ele cai no meio do gap por convenção, e seu valor exato depende ligeiramente da malha de k. Sempre confiram:

```
grep 'E-fermi' OUTCAR
```

5.4. Quando algo der errado

Erros comuns nesta aula:

- **“BRMIX: very serious problems”:** problema de convergência do mixing. Tentem ALGO = All ou reduzir AMIX = 0.1.
- **Bandas com energias absurdas (~+50 eV):** esqueceram de copiar o CHGCAR ou ICHARG não está em 11.
- **“tetrahedron method fails for NKPT < 4”:** malha pequena demais para ISMEAR = -5. Aumentem ou troquem para ISMEAR = 0.
- **Curva E(V) com ruído visível:** EDIFF não estava apertado o suficiente, ou malha de k pouco densa. Apertem ambos.

6. O que vocês aprenderam

Ao fim desta aula, vocês saberão fazer três classes de cálculo que cobrem boa parte do que se publica em estrutura eletrônica de sólidos:

- Equação de estado e propriedades termodinâmicas via varredura controlada de volume.
- Estrutura de bandas em caminhos de alta simetria — a forma canônica de visualizar a estrutura eletrônica de cristais.
- DOS e PDOS — a ponte entre estrutura eletrônica e ligação química, e a base para interpretar espectroscopias.

Na próxima aula avançaremos para sistemas com mais de um elemento (compostos binários como GaAs ou TiO_2), onde o PDOS revelará a transferência de carga e a natureza da ligação química. Daí em diante o caminho leva naturalmente para defeitos, superfícies e propriedades magnéticas.