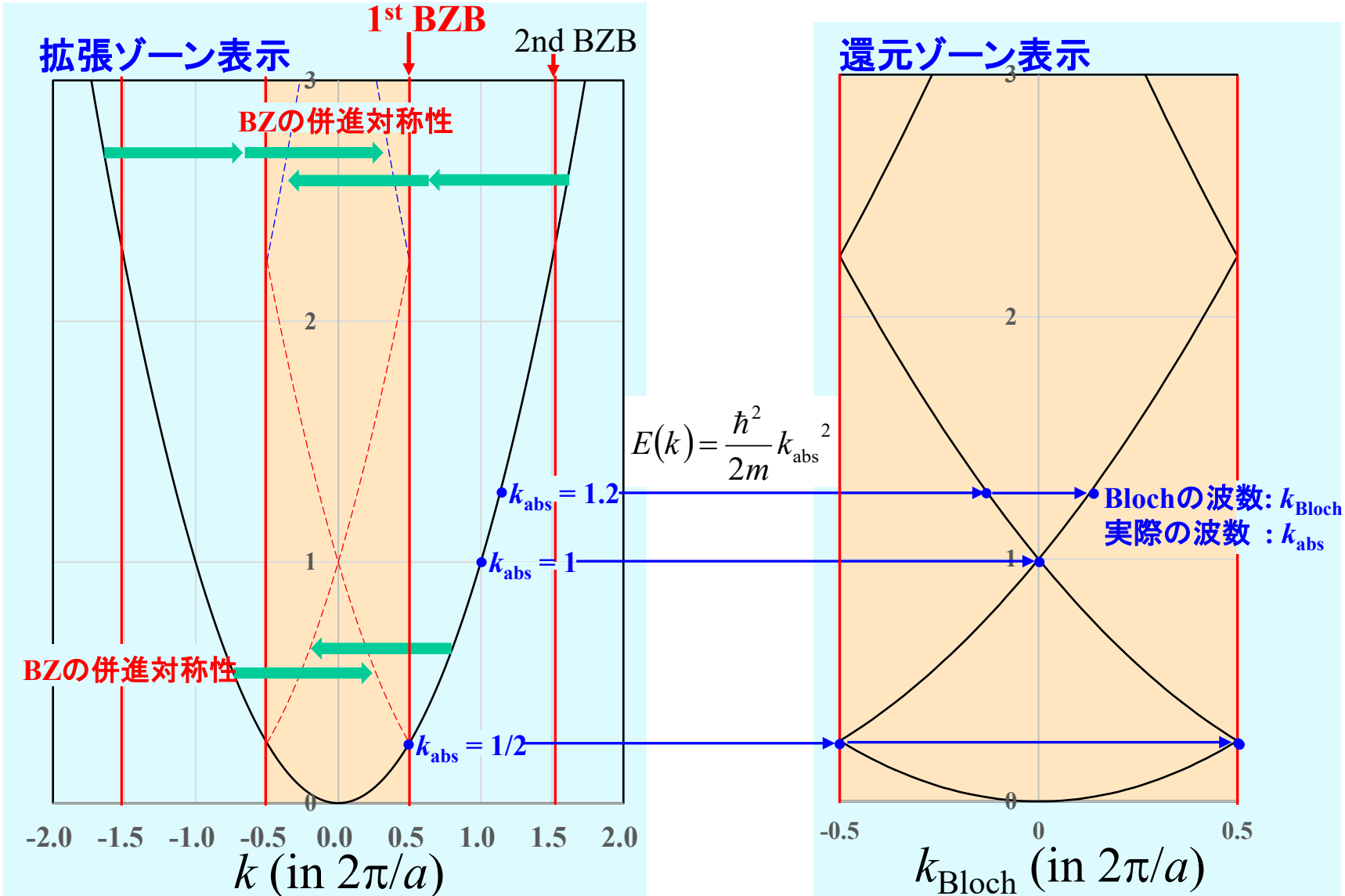
自由電子 (空格子) バンド

$$\Psi_k(x) = C\exp\left[i(k+G_h)\right] = C\exp\left[i(k+ha^*)\right] \qquad h = \cdots, -2, -1, 0, 1, 2, \cdots$$



プログラム: 自由電子バンド

free_electron_band.py

Si の Bravais格子のk点に対してプロット

a = 5.4064 Å

$m^* = 1.0m_e$

W: (1/2 0 1)

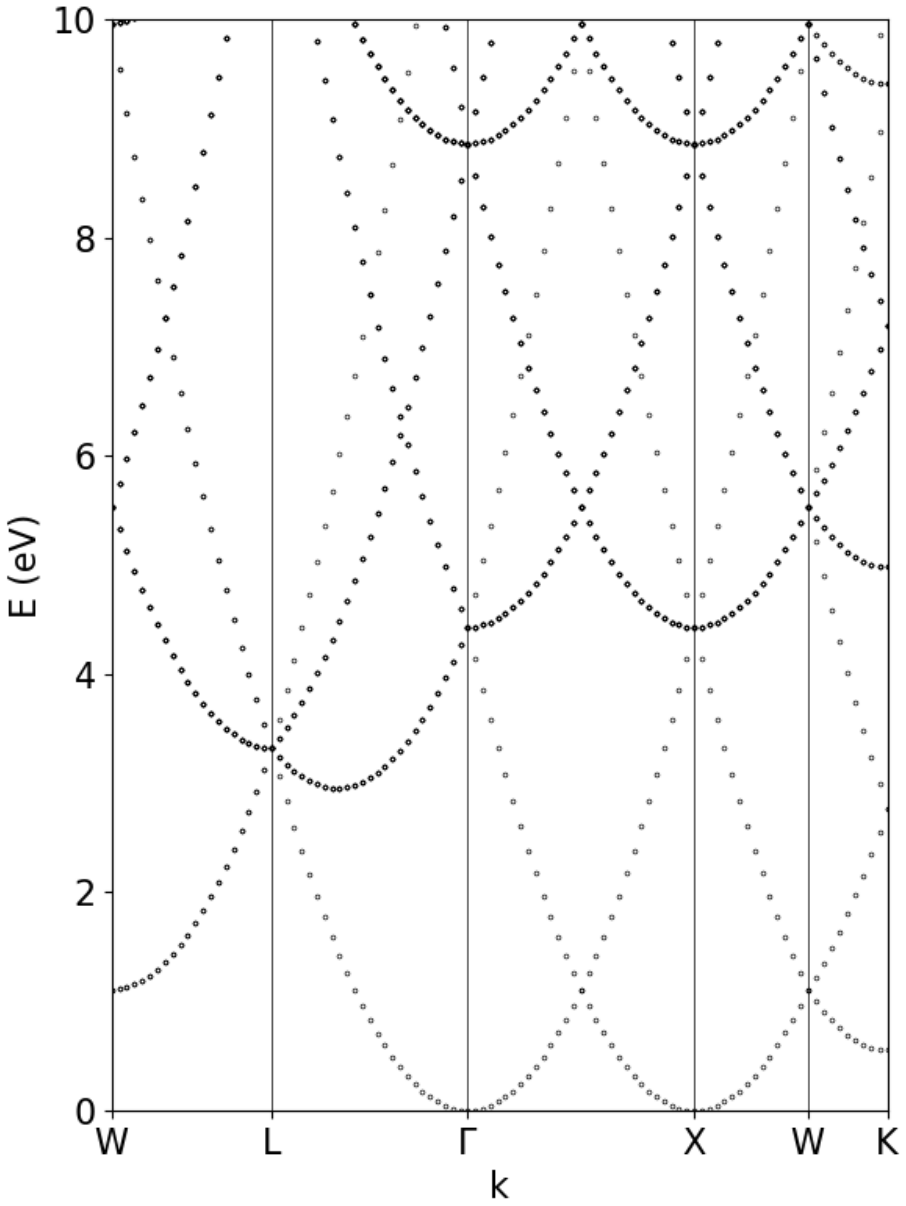
L: (1/2 1/2 1/2)

Γ: (0 0 0)

X: (0 0 1)

W: (1/2 0 1)

K: (3/4 0 3/4)



プログラム: 自由電子バンド

free_electron_band.py

```
from pprint import pprint #リストを整形して出力

a  = 5.4064            # angstrom, lattice parameter
#逆格子のmetricsを計算
rg = np.zeros([3, 3])
rg[0][0] = 2.0 * pi / a
rg[1][1] = rg[0][0]
rg[2][2] = rg[0][0]

# バンド構造をプロットするk点の軌跡:
# [kx, ky, kz, k点名称]
klist = [
      [0.5,  0.0, 1.0, "W"]
    , [0.5,  0.5, 0.5,  "L"]
    , [0.0,  0.0, 0.0,  "$¥Gamma$"]
    , [0.0,  0.0, 1.0,  "X"]
    , [0.5,  0.0, 1.0,  "W"]
    , [0.75, 0.0, 0.75, "K"]
    ]
# プロットするバンド構造E(k)のk点数の概数
nk = 101

# Ehkl(k)を計算するhkl範囲
hrange = [-3, 3]
krange = [-3, 3]
lrange = [-3, 3]

# プロットするエネルギー範囲
Erange = [0.0, 10.0]    # eV

# 逆格子のmetrixから、2点のk点間の距離を計算
def cal_kdistance(rg, k0, k1):
    dkx = k1[0] - k0[0]
    dky = k1[1] - k0[1]
    dkz = k1[2] - k0[2]
    r2  = rg[0][0] * dkx*dkx + rg[1][1] * dky*dky +
rg[2][2] * dkz*dkz
    r2 += 2.0 * (rg[0][1] * dkx*dky + rg[1][2] *
dky*dkz + rg[2][0] * dky*dkx)

    return sqrt(r2)
```

プログラム: 自由電子バンド

free_electron_band.py

```
# k点を与えて自由電子のエネルギーを計算
def cal_E(k, Ghkl):
    global rg

    kabs2  = rg[0][0] * (k[0] + Ghkl[0])**2
    kabs2 += rg[1][1] * (k[1] + Ghkl[1])**2
    kabs2 += rg[2][2] * (k[2] + Ghkl[2])**2

    return KE * kabs2  # in eV

# プロットするk点リスト klistとk点数の概数 nk から、
# なるべくk点間隔が等間隔になるように、
# 計算するk点などをリストアップする
# バンド構造プロットに必要なリストも返す
def get_cal_klist(klist, nk):
   …
```

```
# k点のリストとhkl範囲を与え、Ehkl(k)を計算
def get_cal_Elist(xkvec, hrange, krange, lrange):
    yE = []
    for i in range(len(xkvec)):
        kx = xkvec[i][0]
        ky = xkvec[i][1]
        kz = xkvec[i][2]
        Elist = []
        for ih in range(hrange[0], hrange[1]+1):
            for ik in range(krange[0], krange[1]+1):
                for il in range(lrange[0], lrange[1]+1):
                    E = cal_E([kx, ky, kz], [ih, ik, il])
                    Elist.append(E)

        yE.append(Elist)

    return yE
```

プログラム: 自由電子バンド

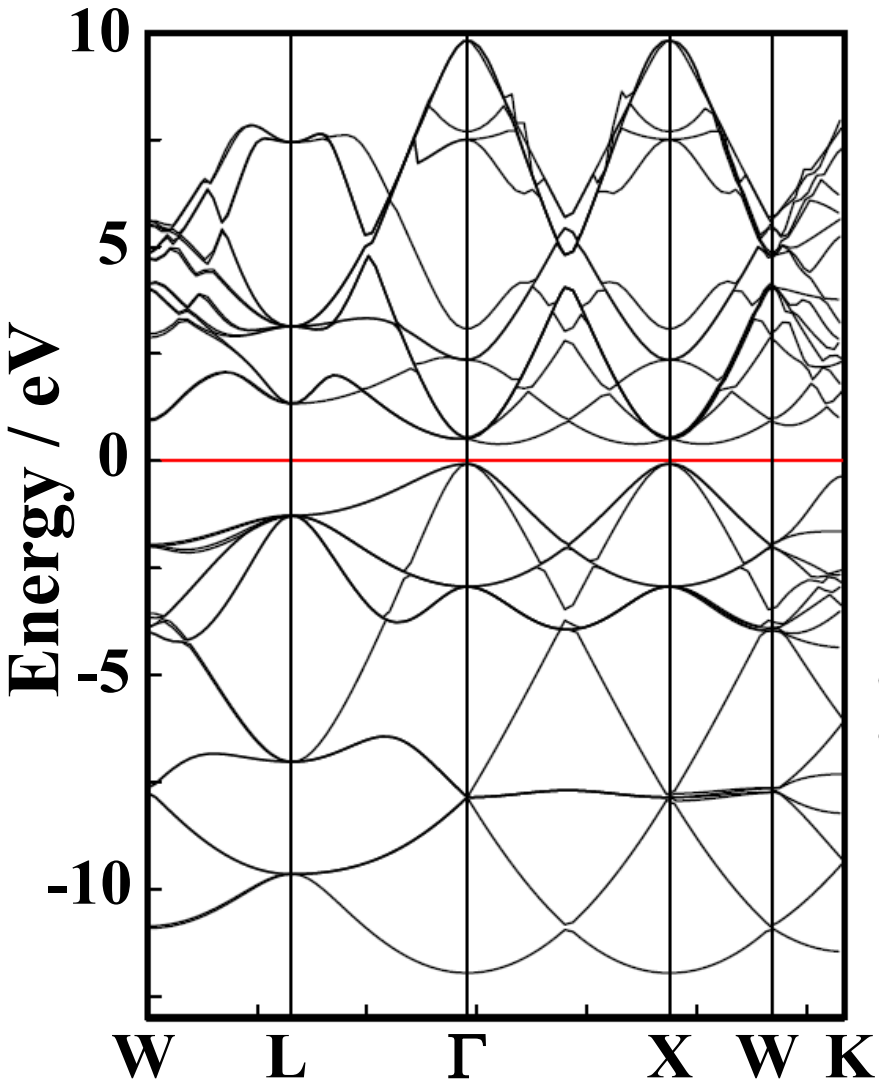
free_electron_band.py

```
# バンド構造をプロット
def plot_band(axis, xk, yE, Erange, ktotallist,
ktotal_namelist):
# 表示範囲は決め打ち
    axis.set_xlim([min(xk), max(xk)])
    axis.set_ylim(Erange)

# バンド構造をプロット
    axis.plot(xk, yE, linestyle = 'none',
              marker = 'o', markerfacecolor = 'none',
markeredgecolor = 'black',
              markeredgewidth = 0.5, markersize = 2.0)

# Γ点、BZ境界の縦線を引く
    for i in range(1, len(ktotallist)):
        axis.plot([ktotallist[i-1], ktotallist[i-1]], Erange,
                  linestyle = '-', color = 'black', linewidth =
0.5)
```

```
# k軸の目盛りにk点の名称を表示する
# グラフ枠が一つであれば plt.xtics()で設定できる
# axisに対しては、.setpでattributeを直接書き換える
必要があるらしい
    plt.setp(axis, xticks = ktotallist, xticklabels =
ktotal_namelist)
    axis.set_xlabel("k", fontsize = fontsize)
    axis.set_ylabel("E (eV)", fontsize = fontsize)
    axis.tick_params(labelsize = fontsize)
```

(拡がった)バンド構造は自由電子として理解できる: Siの例

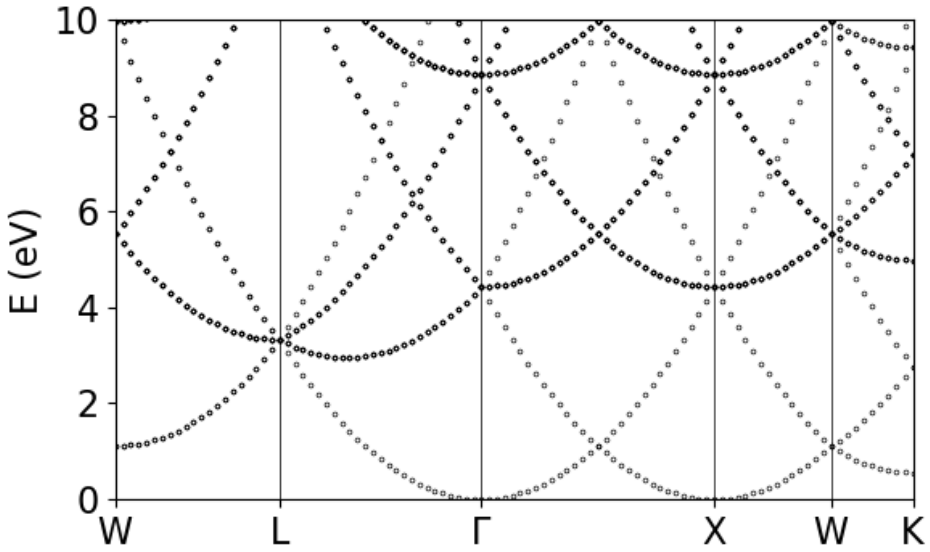
VASP, PBE96, Conventional cell



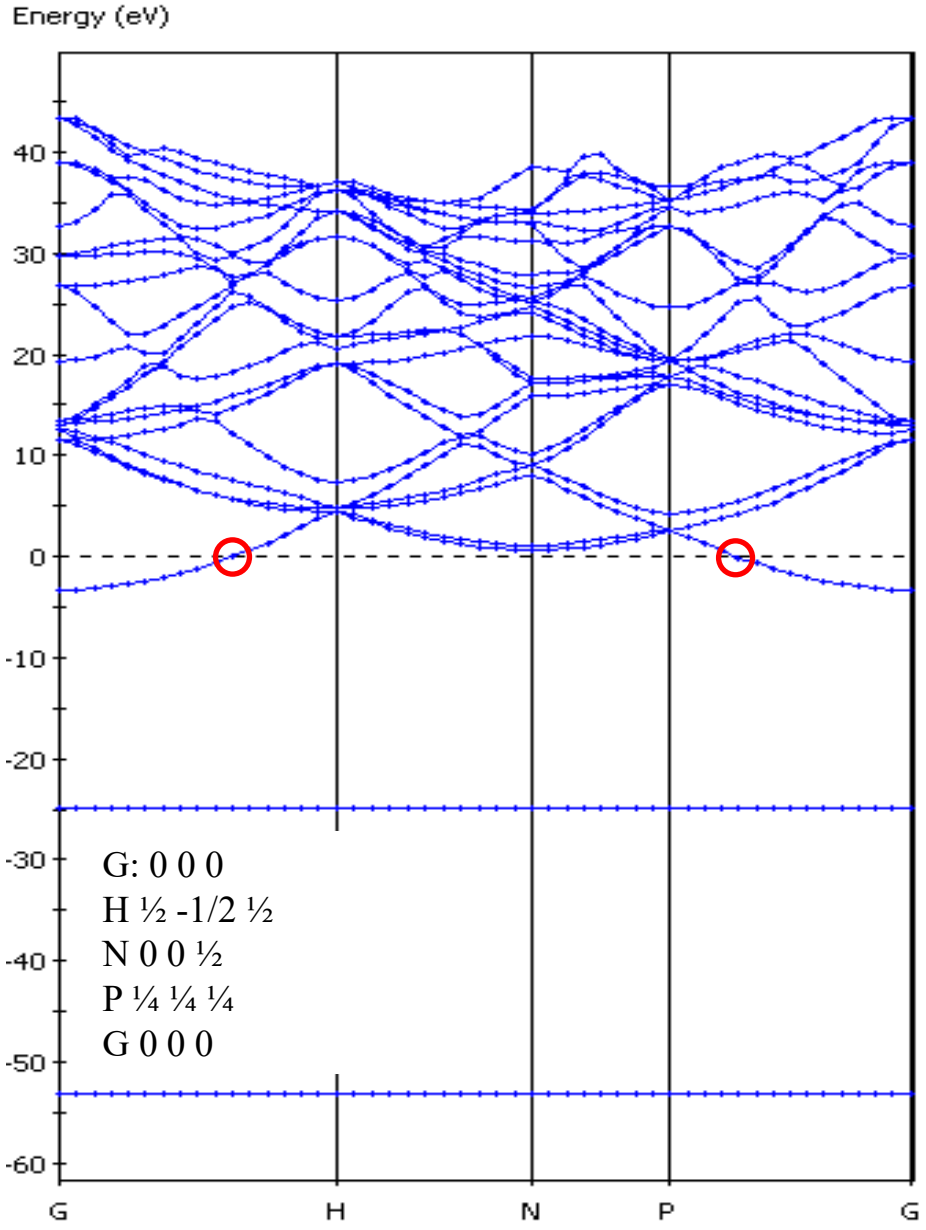
自由電子モデル

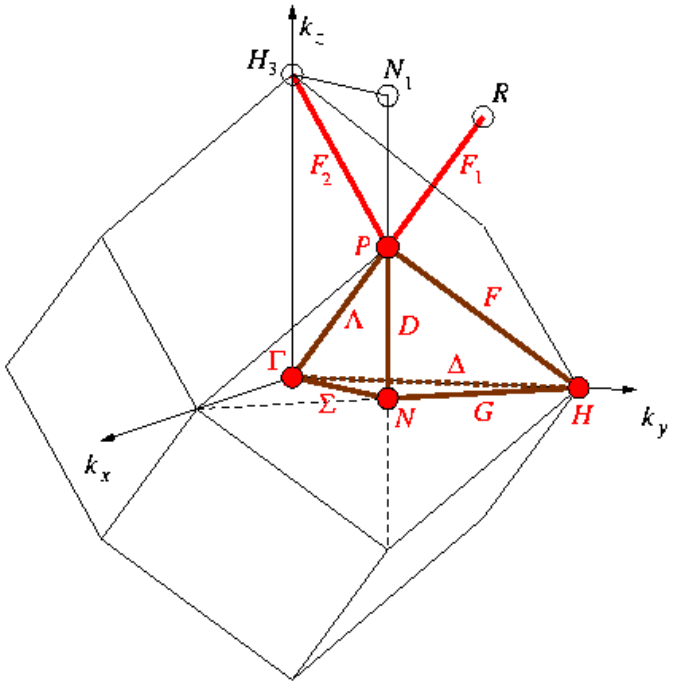
$$E_{free}(\mathbf{k}) = \frac{\hbar^2}{2m}\mathbf{K}^2 = \frac{\hbar^2}{2m}(\mathbf{k}+\mathbf{G}_{\mathbf{hkl}})^2$$

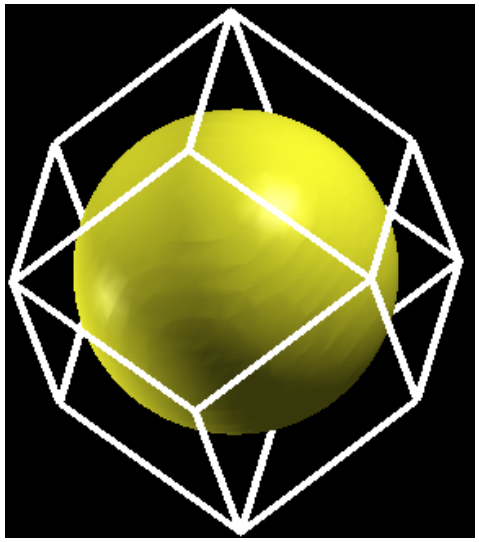
$$\mathbf{G}_{\mathbf{hkl}} = h\mathbf{a}^* + k\mathbf{b}^* + l\mathbf{c}^*$$



Na (BCC) のフェルミ面







平面波法

一次結合の基底関数として平面波を使う

$$\phi_{\mathbf{k}}(\mathbf{r}) = \exp(i\mathbf{k}\cdot\mathbf{r})\sum C_{hkl}u_{hkl}(\mathbf{r}) \qquad u_{hkl}(\mathbf{r}) = \exp[i\mathbf{G}_{\mathbf{hkl}}\cdot\mathbf{r}]$$

波数 $G_{\mathbf{hkl}}$ の平面波は格子周期の関数の完全基底系:
すべての hkl について和を取れば、完全に正しい解になる
=> **実際の計算では $|G_{\mathbf{hkl}}| < G_{\mathbf{max}}$ の範囲で近似**する

$$\begin{vmatrix} H_{11}-ES_{11} & H_{12}-ES_{12} & \cdots & H_{1n}-ES_{1n} \\ H_{21}-ES_{21} & H_{22}-ES_{ss} & & H_{2n}-ES_{2n} \\ \vdots & & \ddots & \vdots \\ H_{n1}-ES_{n1} & H_{n2}-ES_{n2} & \cdots & H_{nn}-ES_{nn} \end{vmatrix} = 0$$

$$\left\langle u_{h'k'l'}\middle|H\middle|u_{hkl}\right\rangle = \int e^{-i(\mathbf{k}+\mathbf{G}_{\mathbf{h'k'l'}})\cdot\mathbf{r}}\left[-\frac{\hbar^2}{2m}\nabla^2 + V(\mathbf{r})\right]e^{i(\mathbf{k}+\mathbf{G}_{\mathbf{hkl}})\cdot\mathbf{r}}d\mathbf{r}$$

$$= \delta_{hkl,h'k',l'}\frac{\hbar^2}{2m}k^2 + V^*(\mathbf{G}_{\mathbf{hkl}}-\mathbf{G}_{\mathbf{h'k'l'}})$$

実際の計算のほとんどがポテンシャルのフーリエ変換
=> GPUで高速化が容易

プログラム: 一次元平面波法

http://conf.msl.titech.ac.jp/jsap-crystal/

平面波基底による一次元バンド計算　pw1d.py

Usage: python pw1d.py

python pw1d.py (ft a na pottype bwidth bpot)

python pw1d.py (band a na pottype bwidth bpot nG kmin kmax nk)

python pw1d.py (wf a na pottype bwidth bpot nG kw iLevel xwmin xwmax nxw)

pottype: rect|gauss

実行例: python pw1d.py ft 5.4064 64 rect 0.5 10.0

ポテンシャルのフーリエ変換を表示。

格子定数5.4064Å、単位格子を $2^6 = 64$分割 (FFTのためnaは2^n)

矩形ポテンシャル 0.5 Å幅、10.0 eV高さ

実行例: python pw1d.py band 5.4064 64 rect 0.5 10.0 3 -0.5 0.5 21

バンド構造を計算。構造、分割数、ポテンシャルは上と同じ

バンド構造を 逆空間内部座標 [-½ ½] (第一ブリルアンゾーン) で21分割して表示

実行例: python pw1d.py wf 5.4064 64 rect 0.5 10.0 3 0.0 0 0.0 16.2192 101

結晶波動関数を表示。構造、分割数、ポテンシャルは上と同じ

波数ベクトルはΓ点に近い3点を用いる。

k = 0.0 (Γ点), 固有解の 0 番目の準位の波動関数を、

0.0 ~ 16.2192 オングストロームの範囲で101分割して表示

(注意: 固有解はエネルギー順にソートしていないので、
コンソール出力のEnergy levels:で準位の番号を確認)

Energy levels:

0　0.624459 eV

1　6.39666 eV

2　6.08362 eV

プログラム: 一次元平面波法

pw1d.py

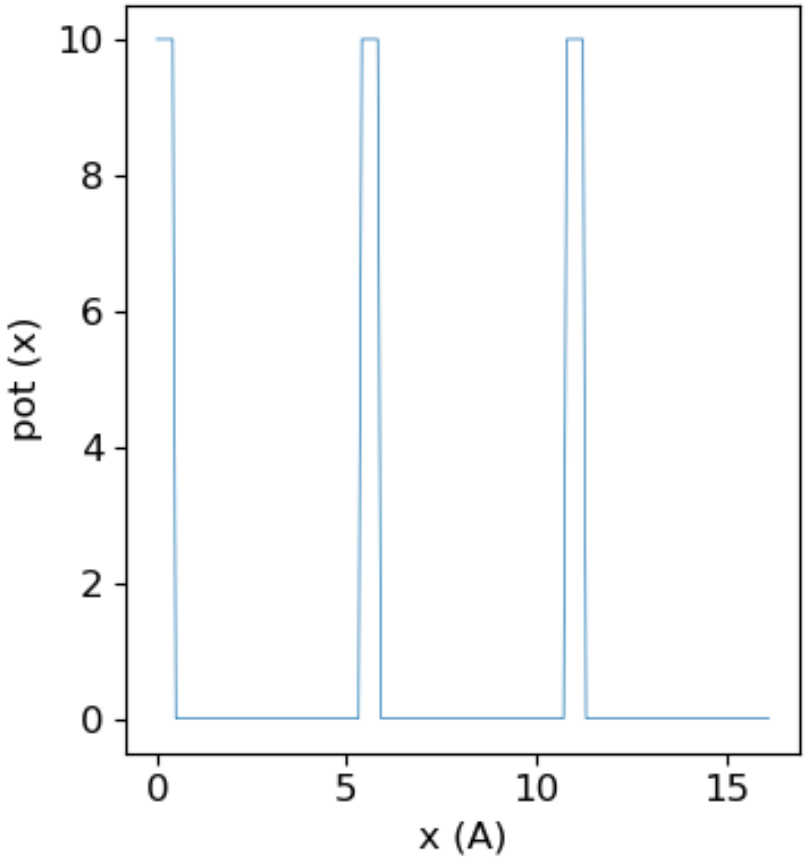
Si の格子定数　a = 5.4064 Å　　$m^* = 1.0m_e$

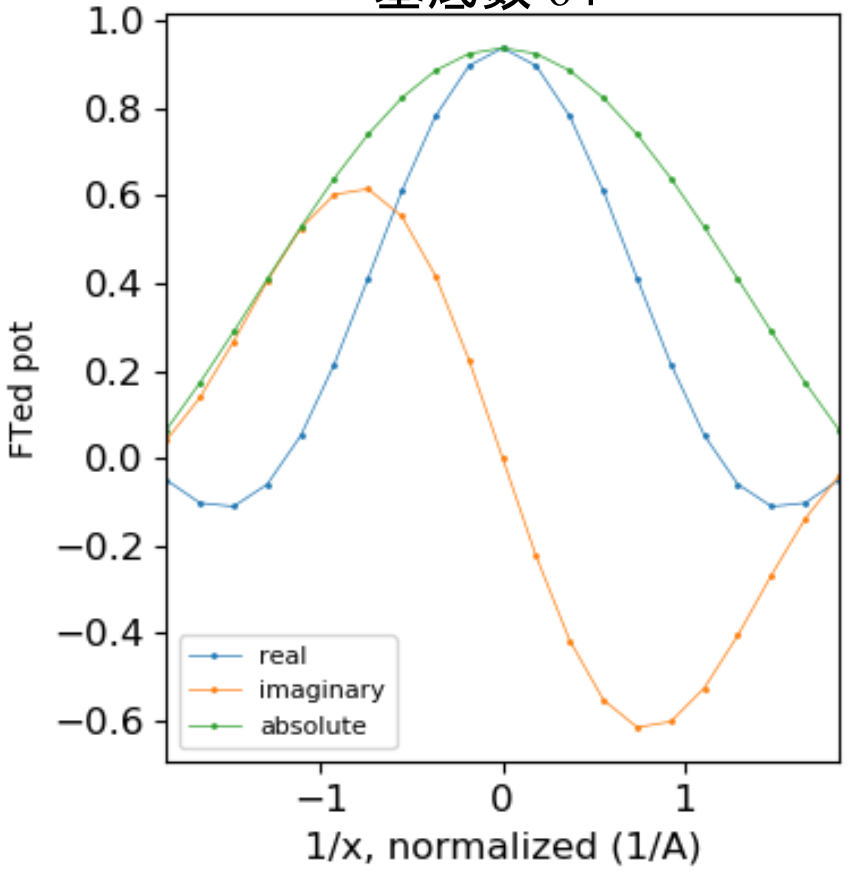
ポテンシャル $V(x)$:　障壁幅　　0.5 Å　　障壁高さ 10.0 eV

python pw1d.py ft 5.4064 64 rect 0.5 10.0 9 -0.5 0.5 21

ポテンシャルの
フーリエ変換係数

基底数 64





プログラム: 一次元平面波法

pw1d.py

Si の格子定数　a = 5.4064 Å　　$m^* = 1.0m_e$

障壁幅　0.5 Å　障壁高さ 10.0 eV

python pw1d.py band 5.4064 64 rect 0.5 10.0 9 -0.5 0.5 21

python pw1d.py wf 5.4064 64 rect 0.5 10.0 9 0.0 2 0.0 17 501

k = 0.0, 1番目の準位 (E=0.608eV)の波動関数

(注意: 準位はエネルギー順にソートしていないので、
コンソール出力のEnergy levels:で準位の番号を確認)

Energy levels:

0	83.3381 eV
1	47.3631 eV
2	0.608457 eV
3	6.91001 eV
4	22.477 eV
5	83.3212 eV
6	47.2599 eV
7	20.7386 eV
8	5.18836 eV