Diffraction Final Project

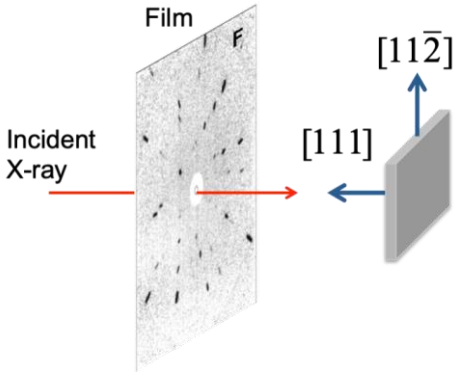
材料三

B06507002

林柏勳

1. Problem Description

A single-crystalline silicon sample is analyzed using the Laue Back-reflection method. A tungsten X-ray tube with wavelength (λ) between 0.6 Å and 1.2 Å is used. The silicon crystal is placed on the goniometer with its [111] axis toward the incident X-ray beam and its [112] axis is pointing upward.



Hint: Firstly, find out all the possible reflections of Si – the reciprocal lattice points of Si enclosed between the Ewald spheres of λ = 0.6 Å and λ = 1.2 Å, respectively. Then you calculate the projection of these diffraction to see whether or not they appear on the film.

2. Solution

Step 0:

Si is of diamond structure. Its lattice can be described as the convolution of cubic lattice $L(\bar{r})$ with FCC lattice points $\{(0,0,0),\left(\frac{1}{2},\frac{1}{2},0\right),\left(0,\frac{1}{2},\frac{1}{2}\right),(\frac{1}{2},0,\frac{1}{2})\}$ and then convolution with 2 motifs $\rho_b(\bar{r}) = \{(0,0,0),\left(\frac{1}{4},\frac{1}{4},\frac{1}{4}\right)\}$.

$$\rho(\bar{r}) = \rho_b(\bar{r}) * L_{fcc}(\bar{r}) * L_{cubic}(\bar{r}) \ (1)$$

The amplitude in the reciprocal space is proportional to $F\big(\rho(\bar{r})\big)$, which is equal to

$$A(\bar{k}) = f_e F\big(\rho(\bar{r})\big) = f_e F\big(\rho_b(\bar{r})\big) F\left(L_{fcc}(\bar{r})\right) F\big(L_{cubic}(\bar{r})\big) = f_e F_b F_{fcc} F\big(L_{cubic}(\bar{r})\big) \ (2)$$

where the reciprocal lattice of FCC is

$$F\left(L_{fcc}(\bar{r})\right) = F_{fcc} = 1 + e^{i\pi(h+k)} + e^{i\pi(k+l)} + e^{i\pi(h+l)}$$

, which has been deduced in class. Besides, the Fourier transform of the basis is in the below

$$F_b = F\big(\rho_b(\bar{r})\big) = f_{Si}\left(e^{2\pi i(k_x\times 0 + k_y\times 0 + k_z\times 0)} + e^{2\pi i\left(\frac{1}{4}k_x + \frac{1}{4}k_y + \frac{1}{4}k_z\right)}\right) = f_{si}(1 + e^{\pi i\left(\frac{1}{2}k_x + \frac{1}{2}k_y + \frac{1}{2}k_z\right)}) \ (3)$$

So, the diffraction condition is the section of reciprocal FCC lattice $F\left(L_{fcc}(\bar{r})\right) F\big(L_{cubic}(\bar{r})\big)$ and the Fourier transform of motifs F_b.

In the reciprocal space of FCC lattice, all reciprocal lattice points expect all odd and all even (hkl) has

the contribution to the diffraction amplitude. Each contribution is equal to 4.
Start at all odd and all even (hkl), if we discuss the case in detail in Eqn.3, then

$$\begin{cases} Case\ 1\text{: } (hkl)\ all\ odd\ and\ h+k+l=1+4m, & F_b=4f_{Si}(1+i) \\ Case\ 2\text{: } (hkl)\ all\ odd\ and\ h+k+l=3+4m, & F_b=4f_{Si}(1-i) \\ Case3\text{: } (hkl)\ all\ even\ and\ h+k+l=4m, & F_b=8f_{Si} \\ Case\ 4\text{: } (hkl)\ all\ even\ and\ h+k+l=2+4m, & F_b=0 \end{cases} \quad (4)$$

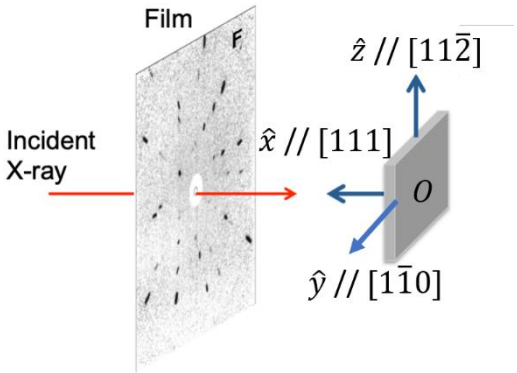
$$where\ m \in \boldsymbol{Z}$$

In order to have nonzero intensity in the reciprocal space of Si, (hkl) not only have to be all odd or all even, but also have to be in Case 1-3.

Step 1:

Back to the problem, if we define $\hat{x}=\frac{1}{\sqrt{3}}(1,1,1)$ and $\hat{z}=\frac{1}{\sqrt{6}}(1,1,-2)$

By using outer product $\hat{y}=\hat{z}\times\hat{x}=\frac{1}{\sqrt{2}}(1,-1,0)$.



In real space $\overrightarrow{a_1}=a\hat{x},\ \overrightarrow{a_2}=a\hat{y},\ \overrightarrow{a_3}=a\hat{z}.$

In the reciprocal space, $\overrightarrow{b_1}=\frac{1}{a}\hat{x},\ \overrightarrow{b_2}=\frac{1}{a}\hat{y},\ \overrightarrow{b_3}=\frac{1}{a}\hat{z}.$

The incident X-ray vector can be expressed as $\overrightarrow{k_0}=-\frac{1}{\lambda}\hat{x}$, pointing toward the origin O. Also, the radius of the Ewald sphere is $\frac{1}{\lambda}$. For a (hkl) plane, if we want to express its reciprocal lattice vector. $\overrightarrow{r^*_{hkl}}=b_1\hat{x}+b_2\hat{y}+b_3\hat{z}$,

$$\mathrm{b}_1=\overrightarrow{r^*_{hkl}}\cdot\hat{x}=\frac{1}{a}(h,k,l)\cdot\frac{1}{\sqrt{3}}(1,1,1)$$

$$\mathrm{b}_2=\overrightarrow{r^*_{hkl}}\cdot\hat{y}=\frac{1}{a}(h,k,l)\cdot\frac{1}{\sqrt{2}}(1,-1,0)$$

$$\mathrm{b}_3=\overrightarrow{r^*_{hkl}}\cdot\hat{z}=\frac{1}{a}(h,k,l)\cdot\frac{1}{\sqrt{6}}(1,1,-2)$$

or $\mathbf{T}\begin{bmatrix}b_1\\b_2\\b_3\end{bmatrix}=\frac{1}{a}\begin{bmatrix}h\\k\\l\end{bmatrix}\rightarrow\begin{bmatrix}b_1\\b_2\\b_3\end{bmatrix}=\frac{1}{a}\boldsymbol{T}^{-1}\begin{bmatrix}h\\k\\l\end{bmatrix}$ where **T** is defined as below:

$$\mathbf{T} = \begin{bmatrix} \frac{1}{\sqrt{3}} & \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & \frac{-1}{\sqrt{2}} & \frac{1}{\sqrt{6}} \\ \frac{1}{\sqrt{3}} & 0 & \frac{-2}{\sqrt{6}} \end{bmatrix}$$

The diffraction condition will be $\overrightarrow{k_0} + \overrightarrow{r^*_{hkl}} = \overrightarrow{k_1}$

$$\left|\overrightarrow{k_0} + \overrightarrow{r^*_{hkl}}\right| = \left|\overrightarrow{k_1}\right| = \frac{1}{\lambda}$$

$$\left|\overrightarrow{r^*_{hkl}} - \frac{1}{\lambda}\hat{x}\right| = \frac{1}{\lambda}$$

$$\left(b_1 - \frac{1}{\lambda}\right)^2 + b_2^2 + b_3^2 = \left(\frac{1}{\lambda}\right)^2$$

$$\lambda = \frac{2b_1}{b_1^2 + b_2^2 + b_3^2}$$

By constraint

$$0.6\ \text{Å} \leq \lambda \leq 1.2\ \text{Å}$$

we can select valid diffraction (hkl) planes.

(a) Simulate the Laue diffraction pattern. Assume that a film of 10 cm × 10 cm is placed 9 cm away from the sample for recording the diffraction signals.

Step 2:

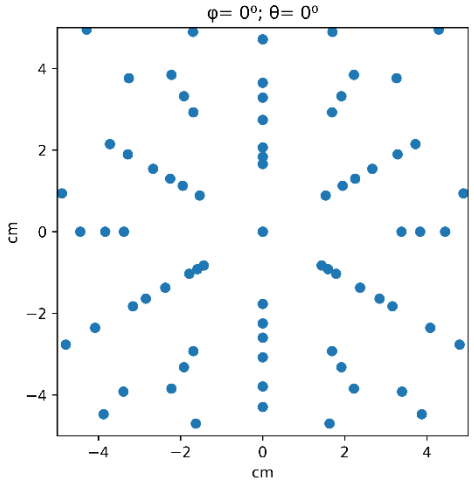
In order to make the projection vector

$$(b_1 - \frac{1}{\lambda})\hat{x} + b_2\hat{y} + b_3\hat{z})\frac{9}{b_1 - \frac{1}{\lambda}} = 9\,\hat{x} + \frac{9b_2}{b_1 - \frac{1}{\lambda}}\hat{y} + \frac{9b_3}{b_1 - \frac{1}{\lambda}}\hat{z}$$

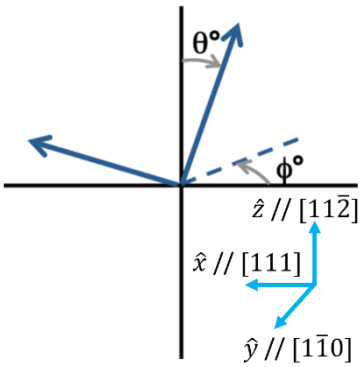
pointing to the film, (hkl) plane must satisfy

1. $b_1 - \frac{1}{\lambda} > 0$, for back-scattering diffraction
2. $\left|\frac{9b_2}{b_1 - \frac{1}{\lambda}}\right| \leq 5\ and\ \left|\frac{9b_3}{b_1 - \frac{1}{\lambda}}\right| \leq 5$, to lie on the 10× 10 film

The result is in the below:



(b) If the [111] axis of the crystal is off the incident X-ray direction by 5° rotation about the vertical axis (ϕ in the figure on the right) and 3° tilt (θ in the figure). Simulate the Laue diffraction pattern.



The operation can be deemed as: we first rotate the crystal through $\hat{z}$ counterclockwise by degree $\phi \equiv \mathbf{R}_{\hat{z}}(\boldsymbol{\phi})$ and then a rotate the crystal through $\hat{y}$ clockwise by degree $\theta \equiv \mathbf{R}_{\hat{y}}(-\boldsymbol{\theta})$. Note that $\mathbf{R}_{\hat{z}}, \mathbf{R}_{\hat{y}}$ are rotation matrices through $\hat{z}$ and $\hat{y}$.

As a result, all planes in the reciprocal space should be multiply $\mathbf{R}_{\hat{z}}(\boldsymbol{\phi})$ then by $\mathbf{R}_{\hat{y}}(-\boldsymbol{\theta})$.

The rotation matrix of 3D can be shown as below:

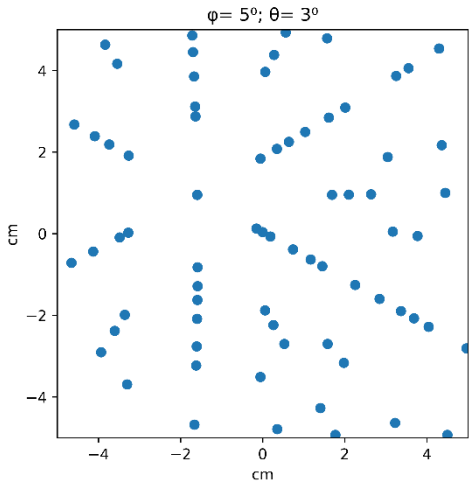
$$\mathbf{R}_{\hat{z}}(\boldsymbol{\phi}) = \begin{bmatrix} cos(\phi) & -sin(\phi) & 0 \\ sin(\phi) & cos(\phi) & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

$$\mathbf{R}_{\hat{y}}(-\boldsymbol{\theta}) = \begin{bmatrix} cos(\theta) & 0 & -sin(\theta) \\ 0 & 1 & 0 \\ sin(\theta) & 0 & cos(\theta) \end{bmatrix}$$

The matrix form of rotation operation can be shown as below:

$$\begin{bmatrix} b_1' \\ b_2' \\ b_3' \end{bmatrix} = \mathbf{R}_{\hat{y}}(-\boldsymbol{\theta})\mathbf{R}_{\hat{z}}(\boldsymbol{\phi}) \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} = \frac{1}{a}\mathbf{R}_{\hat{y}}(-\boldsymbol{\theta})\mathbf{R}_{\hat{z}}(\boldsymbol{\phi})\boldsymbol{T}^{-1} \begin{bmatrix} h \\ k \\ l \end{bmatrix} \quad (5)$$

In this problem, we first transform every (hkl) in the reciprocal space by using Eqn. 5, and then repeat Step 2. Finally, we can calculate the valid diffraction points on the film.



3. Discussion

1. How to choose the range of (hkl) points in this problem?

In this problem, if we visualize all Ewald spheres between λ = 0.6 Å and λ = 1.2 Å, we can see that all Ewald spheres will be enclosed in the Ewald sphere of λ = 0.6 Å. To give diffraction, the (hkl) points must lie in this Ewald sphere.

As a result, we can write a formula of the constraint of (hkl):

$$0 \leq b_1' \leq \frac{2}{\lambda} \quad (6)$$

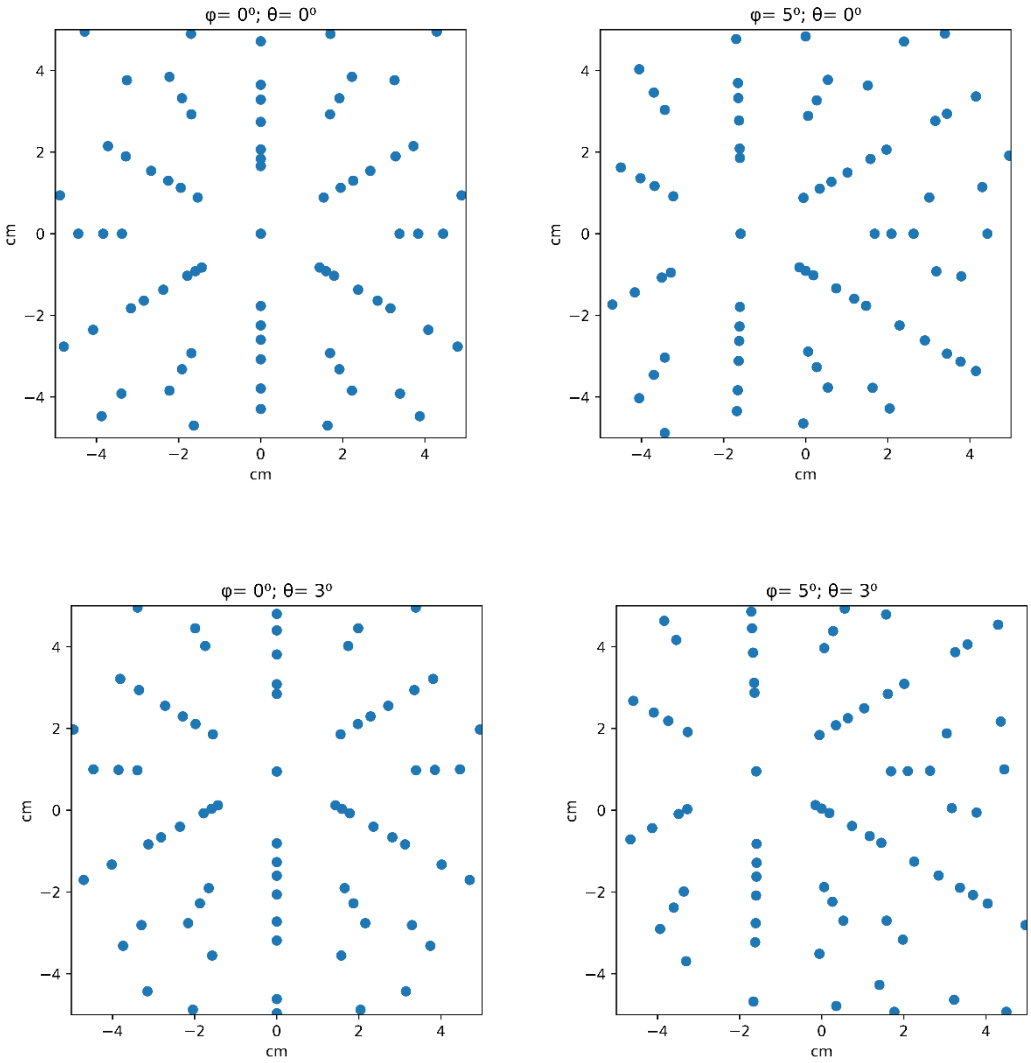
$$-\frac{1}{\lambda} \leq b_2' \leq \frac{1}{\lambda} \quad (7)$$

$$-\frac{1}{\lambda} \leq b_3' \leq \frac{1}{\lambda} \quad (8)$$

$$\frac{1}{\lambda}\begin{bmatrix}0\\-1\\-1\end{bmatrix} \leq \frac{1}{a}\mathbf{R}_{\hat{y}}(-\boldsymbol{\theta})\mathbf{R}_{\hat{z}}(\boldsymbol{\phi})\boldsymbol{T}^{-1}\begin{bmatrix}h\\k\\l\end{bmatrix} = \begin{bmatrix}b_1'\\b_2'\\b_3'\end{bmatrix} \leq \frac{1}{\lambda}\begin{bmatrix}2\\1\\1\end{bmatrix} \quad (9)$$

By solving the system of inequalities of Eqn. (6)~(8), we can get a valid range for the input (hkl) points. This process can speed up our calculation by dropping out invalid (hkl) points.

2. How does θ and ϕ affect the output pattern?



From the above 4 figures, we can see that as increasing ϕ results in the left-shift of the center and increasing θ results in the up-shift of the center.

4. Code

The code can be executed in the command line by the below command:

(a) python diffraction_final.py 0 0

(b) python diffraction_final.py 5 3

where the code diffraction_final.py is in below and after execution the diffraction pattern (.png) will be dumped in the same folder.

```
1. import numpy as np
2. import pandas as pd
3. import numpy.linalg as lin
4. from numpy import sqrt,cos,sin,pi
5. import matplotlib.pyplot as plt
```

```
6. import sys
7. phi,theta=sys.argv[1:]
8. phi=int(phi)/180*pi
9. theta=int(theta)/180*pi
10.  a=5.431
11.  lmin=0.6
12.  lmax=1.2
13.  def neg(a):
14.      b=-1*np.copy(a)
15.      b[-1]=-a[-2]
16.      b[-2]=-a[-1]
17.      return b
18.  def ero2d(a):
19.      a=np.copy(a).astype("float")
20.      if a[0,0]*a[1,0]>0:
21.          a[1]=a[1]+neg(a[0])/a[0,0]*a[1,0]
22.      else:
23.          a[1]=a[1]-a[0]/a[0,0]*a[1,0]
24.      if a[1,1]>0:
25.          a[1]=a[1]/a[1,1]
26.      else:
27.          a[1]=-neg(a[1])/a[1,1]
28.      if a[0,1]>0:
29.          a[0]=a[0]+neg(a[1])*a[0,1]
30.      else:
31.          a[0]=a[0]-a[1]*a[0,1]
32.      if a[0,0]>0:
33.          a[0]=a[0]/a[0,0]
34.      else:
35.          a[0]=-neg(a[0])/a[0,0]
36.      return a
37.  def ero3d(a):
38.      a=np.copy(a).astype("float")
39.      if a[0,0]*a[1,0]>0:
40.          a[1]=a[1]+neg(a[0])/a[0,0]*a[1,0]
41.      else:
42.          a[1]=a[1]-a[0]/a[0,0]*a[1,0]
43.
```

```
    if a[0,0]*a[2,0]>0:
        a[2]=a[2]+neg(a[0])/a[0,0]*a[2,0]
    else:
        a[2]=a[2]-a[0]/a[0,0]*a[2,0]
    a[1:,1:]=ero2d(a[1:,1:])
    if a[0,1]>0:
        a[0]=a[0]+neg(a[1])*a[0,1]
    else:
        a[0]=a[0]-a[1]*a[0,1]

    if a[0,2]>0:
        a[0]=a[0]+neg(a[2])*a[0,2]
    else:
        a[0]=a[0]-a[2]*a[0,2]

    if a[0,0]>0:
        a[0]=a[0]/a[0,0]
    else:
        a[0]=-neg(a[0])/a[0,0]
    return a[:,3:]


def selection_rule(h,k,l):
    if not (h or k or l):
        return 0
    elif h%2 and k%2 and l%2:
        return 1
    elif not (h%2 or k%2 or l%2):
        if (h+k+l)%4==0:
            return 1
        else:
            return 0
    else:
        return 0
C=np.array([[1/sqrt(3),1/sqrt(2),1/sqrt(6)],
   [1/sqrt(3),-1/sqrt(2),1/sqrt(6)],
   [1/sqrt(3),0,-2/sqrt(6)]])

```

```
M1=np.array([[cos(theta),0,-sin(theta)],
             [0,1,0],
             [sin(theta),0,cos(theta)]])
M2=np.array([[cos(phi),-sin(phi),0],
            [sin(phi),cos(phi),0],
            [0,0,1]])
mat=np.matmul(np.matmul(M1,M2),lin.inv(C))/a
min_max=np.array([[0,-1,-1],[2,1,1]]).T/lmin
constraint=np.concatenate([mat,min_max],axis=1)
con=ero3d(constraint)
reciprocal=[]
for i in range(int(con[0,0]),int(con[0,1])+1):
    for j in range(int(con[1,0]),int(con[1,1])+1):
        for k in range(int(con[2,0]),int(con[2,1])+1):
            if selection_rule(i,j,k):
                reciprocal.append([i,j,k])
reciprocal=np.array(reciprocal)
C3=np.matmul(np.matmul(M1,M2),np.matmul(lin.inv(C),reciprocal.T)).T/a
l=2*C3[:,0]/np.sum(C3**2,axis=1)
x=[]
y=[]
ind=[]
d=9
for i,ele in enumerate(l):
    if ele>=lmin and ele<=lmax:
        k1=C3[i]-np.array([1/ele,0,0])
        if k1[0]>0:
            project=k1/k1[0]*d
            if abs(project)[1]<=5 and abs(project)[2]<=5:
                x.append(-project[1])
                y.append(project[2])
                ind.append(reciprocal[i])

plt.figure(figsize=(5,5))
plt.xlabel("cm")
plt.ylabel("cm")
plt.title("φ= %s⁰; θ= %s⁰"%(sys.argv[1],sys.argv[2]))
plt.plot(x,y,".")
```

```
120. plt.savefig("phi_%s_theta_%s.png"%(sys.argv[1],sys.argv[2]),dpi=600)
```