LCLS Analysis Software Note
April 29, 2016
Mikhail Dubrovin

X-ray Diffraction on Crystal for Dummies (Software Developers)

1 Quantum mechanical definitions

Einstein's **relativistic energy-momentum equation** between particle energy, E, momentum, p, and rest invariant mass, m, is

$$E^2 = p^2c^2 + m^2c^4,$$

where c is a speed of light in vacuum. In case of photon $m = 0$ and this equation is reduced to

$$E = pc. \tag{1}$$

Photon energy E associated with *time structure* of the electromagnetic wave and is expressed in terms of its oscillation priod τ, or frequency $\nu = 1/\tau$, or cyclic frequency $\omega = 2\pi\nu$ as

$$E = \frac{h}{\tau} \equiv h\nu \equiv \hbar\omega, \tag{2}$$

using Planck constant h or its reduced value $\hbar = h/2\pi$.

Photon momentum p associated with *space structure* of the electromagnetic wave and is expressed in terms of its oscillation wavelength, $\lambda = c\tau$, or wave number (vector), $k = 2\pi/\lambda$, or its reduced value, $\bar{k} \equiv k/2\pi = 1/\lambda$,

$$p = \frac{h}{\lambda} \equiv h\bar{k} \equiv \hbar k, \tag{3}$$

By definition E and p fulfill Eqn.1,

$$E = \frac{h}{\tau} = \frac{hc}{\tau c} = \frac{hc}{\lambda} = pc. \tag{4}$$

2 Wavelength for photon energy

PDG 2014 lists values for reduced Planck constant and speed of light in vacuum

$$\hbar = 6.58211928(15) \times 10^{-16} \text{ eV s},$$

$$c = 299792458 \text{ m/s} \quad \text{(exact)}.$$

Speed of light in vacuum is an exact value, because of the meter definition; *"The meter is the length of the path traveled by light in vacuum during time interval of 1/299792458 of a second."*

Eqn.4 can be used to express wavelength through the photon energy,

$$\lambda[\text{nm}] = \frac{hc}{E} = \frac{2\pi\hbar[\text{eV s}] \cdot c[\text{m/s}] \cdot 10^9[\text{nm/m}]}{E[\text{eV}]} = \frac{1239.8493[\text{eV nm}]}{E[\text{eV}]}.$$

Note, that 1m=10^9nm=$10^{10}\mathring{A}$, so 1nm=$10\mathring{A}$.

3 Wave vector

There may be two equivalent definitions for the wave vector, associated with photon momentum, $\vec{p}$, pointing in the direction of photon propagation, "physics" and "crystallographer's"

$$\vec{p} = \hbar\vec{k} \equiv h\vec{k},$$

Magnitude of the wave vector is a wave number

$$k = |\vec{k}| = \frac{2\pi}{\lambda}, \qquad \hbar \equiv \frac{k}{2\pi} = \frac{1}{\lambda}.$$

4 Scattering vector

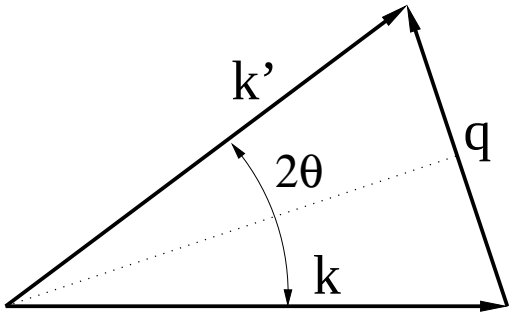

Figure 1: Scaterring vector.

Scattering vector $\vec{q}$ is a difference between final and initial wave vectors in the process of photon scattering

$$\vec{q} = \vec{k'} - \vec{k}, \quad \text{where} \quad |\vec{k'}| = |\vec{k}|.$$

If angle between vectors $\vec{k'}$ and $\vec{k}$ is 2θ, as shown in Fig. 1, the value of scattering vector is

$$q = 2|k| \sin\theta, \tag{5}$$

that can be expressed in terms of "physics" and "crystallographer's" units as

$$q_{physics} = \frac{4\pi}{\lambda} \sin\theta, \qquad q_{cryst} = \frac{2}{\lambda} \sin\theta. \tag{6}$$

5 Bragg's law

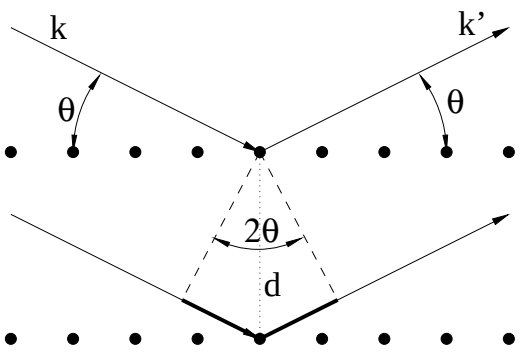

Figure 2: Bragg's reflection.

Constructive interference is observed when the light path difference shown by the bold line in Fig. 2 is folded in number of wavelength, $n\lambda$,

$$2d \sin\theta = n\lambda,$$

then inverse distance between crystal nodes, $1/d$, is

$$\frac{n}{d} = \frac{2}{\lambda} \sin\theta = q_{cryst} \equiv \frac{q_{physics}}{2\pi}.$$

6 Reciprocal space units

From Sections 4 and 5 it is clear, that we have to use consistent units for reciprocal space and q

- "crystallographer's" units $\propto \frac{1}{d}$, $\hbar = \frac{1}{\lambda}$, $q_{cryst} = 2\hbar \sin\theta = \frac{2}{\lambda}\sin\theta$
- "physics" units $\propto \frac{2\pi}{d}$, $k = \frac{2\pi}{\lambda}$, $q_{physics} = 2k \sin\theta = \frac{4\pi}{\lambda}\sin\theta$

7 Bravias lattice

Bravias lattice defines node position $\vec{R}$ through the set of primiteve vectors, for exampe in 3-d space $\vec{a}_1$, $\vec{a}_2$, $\vec{a}_3$, and associated Muller indices (h, k, l) using equation

$$\vec{R} = h\vec{a}_1 + k\vec{a}_2 + l\vec{a}_3. \tag{7}$$

Note, h is an index, not a Planck constant here...

8 Lattice primitive vectors in 3-d space

Lattice primitive vectors can be defined in 3-d and in reciprocal space... In 3-d space we use lattice crystal cell edge length and angles between crystal axes a, b, c, α, β, γ to define lattice primitive vectors $\vec{a}_1$, $\vec{a}_2$, $\vec{a}_3$ oriented along crystal cell edges/axes. Essential code which defines lattice primitive vectors in 3-d space:

```
alp,  bet,  gam  = math.radians(alpha), math.radians(beta), math.radians(gamma)
calp, cbet, cgam = math.cos(alp), math.cos(bet), math.cos(gam)
salp, sbet, sgam = math.sin(alp), math.sin(bet), math.sin(gam)

cx = -c*cbet
cy =  c*calp*sgam
cz = math.sqrt(c*c - cx*cx - cy*cy)

a1 = (a, 0, 0)
a2 = (-b*cgam, b*sgam, 0)
a3 = (cx, cy, cz)
```

9 Lattice primitive vectors in reciprocal space

Lattice primitive vectors in reciprocal space can be expressed in terms of "crystallographer's" definition $\propto 1/d$,

$$\vec{b}_1 = \frac{[\vec{a}_2 \times \vec{a}_3]}{\vec{a}_1 \cdot [\vec{a}_2 \times \vec{a}_3]}, \quad \vec{b}_2 = \frac{[\vec{a}_3 \times \vec{a}_1]}{\vec{a}_2 \cdot [\vec{a}_3 \times \vec{a}_1]}, \quad \vec{b}_3 = \frac{[\vec{a}_1 \times \vec{a}_2]}{\vec{a}_3 \cdot [\vec{a}_1 \times \vec{a}_2]}, \tag{8}$$

or in terms of "physics" definition $\propto 2\pi/d$, the same Eqn.8 but with factor 2π.

10 Ewald's sphere

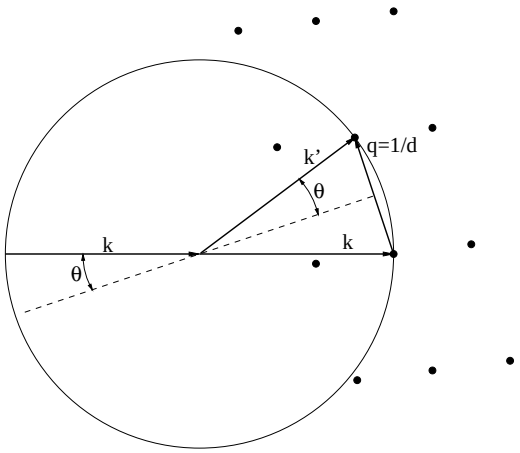

Figure 3: Bragg's reflection and Ewald's sphere.

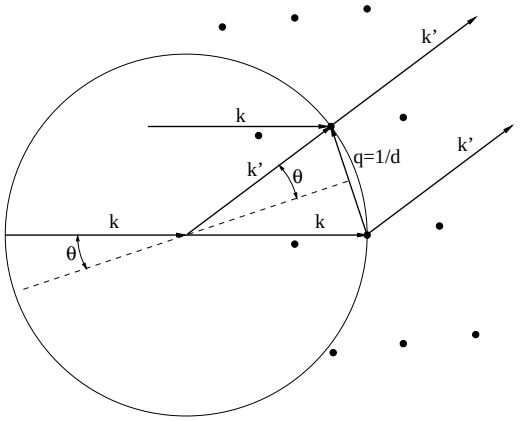

Figure 4: Bragg's reflection and Ewald's sphere for complete dummies.

If the photon energy is conserving in the process of scattering (elastic scattering) as considered in Sec. 4 the length of the scattered wave vector stays the same

$$|\vec{k'}| = |\vec{k}|.$$

All possible tip-points of vector $\vec{k'}$ belong to sphere of radius $|\vec{k}|$ which is called Ewald's sphere. Combining this observation with Bragg's law from Sec. 5 we come to conclusion that interference is observed for lattice nodes in reciprocal space on Ewald's sphere.

11 Look-up table for indexing

Assuming 2- or 3-d model of lattice we may evaluate lattice primitive vectors like in Sec. 8 and convert them in reciprocal spase using Eqn. 8. Then cycling over Muller indices h, k, l in Eqn. 7 we may generate Bravias lattice of any size in reciproal space.

Then we may put the reciprocal lattice origin on the Ewald's sphere, like shown in Fig. 3. For each 3-d lattice orientation we may find which nodes are on the Ewald's sphere within toleranse/resolution parameter. This algorithm is implemented in `cxif5315/make-index-ta` It generates look-up table with content like

```
# file name: ./v02-lut-cxif5315-r0169-2016-02-03T15:10:48.txt

# photon energy = 6003.1936 eV
# wavelength = 2.0653 A
# wave number/Evald radius k = 1/lambda = 0.484187 1/A
# sigma_ql = 0.001453 1/A (approximately = k * <pixel size>/
# sigma_qt = 0.000484 1/A (approximately = k * <pixel size>/<sample-to-detector distance> = k*100um/100mm)
# 3*sigma_ql = 0.004358 1/A

# 3*sigma_qt = 0.001453 1/A

# Triclinic crystal cell parameters:
#   a = 18.55 A
#   b = 27.19 A
#   c = 4.86 A
#   alpha = 90.00 deg
#   beta  = 90.00 deg
#   gamma = 78.47 deg

# 3-d space primitive vectors:
#   a1 = ( 18.550000,   0.000000,   0.000000)
#   a2 = ( -5.435624,  26.645524,   0.000000)
#   a3 = (  0.000000,   0.000000,   4.860100)

# reciprocal space primitive vectors:
#   b1 = (  0.053908,   0.010997,  -0.000000)
#   b2 = (  0.000000,   0.037530,   0.000000)
#   b3 = (  0.000000,  -0.000000,   0.205757)

# ------------------------------------------------------------------------------------

...

# beta 0.00  omega 14.00 degree
# index    beta   omega   h   k   l   dr[1/A]  R(h,k,l)   qv[1/A]   qh[1/A]  P(omega)
    29     0.00   14.00  -3  -6   0  0.001372  0.304642  0.000000 -0.304208  0.640317
    29     0.00   14.00  -2  -5   0  0.003480  0.235743  0.000000 -0.234875  0.056681
    29     0.00   14.00  -1  -3   0 -0.003649  0.134832  0.000000 -0.135294  0.042615
    29     0.00   14.00   0   6   0 -0.002119  0.225179  0.000000  0.225663  0.345084

# beta 0.00  omega 14.50 degree
# index    beta   omega   h   k   l   dr[1/A]  R(h,k,l)   qv[1/A]   qh[1/A]  P(omega)
    30     0.00   14.50  -3  -6   0  0.003889  0.304642  0.000000 -0.303401  0.027767
    30     0.00   14.50  -1  -3   0 -0.002481  0.134832  0.000000 -0.135156  0.232658
    30     0.00   14.50   0   6   0 -0.004036  0.225179  0.000000  0.226087  0.021077
```

12 Scattering vector in 3-d space

In applications we need to evaluate scattering vector components from known initial wave vector, location of the Interaction Point (IP), and assuming that scattered photon crosses certain point P in 3-d space. To this end, it is convenient to use coordinate system with origin in IP, one axis oriented along the initial wave vector (longitudinal direction), and point P in 3-d space defined through the longitudinal and transverse (tangent) components $P(P_\parallel, P_\perp)$. Using this definition we may find distance from IP to P, longitudinal and tangent components of the scattering vector

$$L = \sqrt{P_\parallel^2 + P_\perp^2}, \tag{9}$$

$$q_\parallel = |k|\Big(\frac{P_\parallel}{L} - 1\Big), \tag{10}$$

$$q_\perp = |k|\frac{P_\perp}{L}. \tag{11}$$

13 Scattering vector from detector data

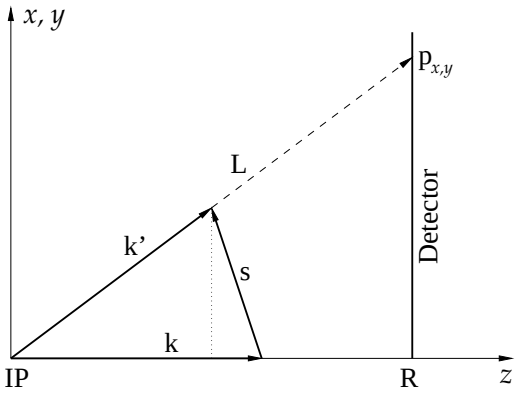

Figure 5: Scaterring vector definition from point on image.

$$\vec{s} = \vec{k'} - \vec{k}, \quad \text{where} \quad |\vec{k'}| = |\vec{k}|,$$

$$L = \sqrt{R^2 + p_x^2 + p_y^2},$$

$$s_{x,y} = |k|\frac{p_{x,y}}{L},$$

$$s_z = |k|\Big(\frac{R}{L} - 1\Big).$$

Python method evaluates normalized scattering vector components $\vec{s}/|k|$ for the 3-d space point (detector pixel)

```
from pyimgalgos.FiberAngles recipnorm
(qh_to_xy - its variation with pretty bad name...)
```

14 Fraser's transformation

Fraser's transformation converts image pixel coordinates $p_{x,y}$ to the 2-d presentation of scattering vector s with horizontal and vertical components

$$(s_H, s_V) = f \cdot \Big(\frac{s_x}{|s_x|}\sqrt{s_x^2 + s_z^2}, s_y\Big) \tag{12}$$

where f is a scale factor to convert scattering vector components to image.

15 Rotation for angle β

We define β as a fiber rotation angle around horizontal axis, parallel to the axis x in the image.

$$D(\beta) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\beta & \sin\beta \\ 0 & -\sin\beta & \cos\beta \end{pmatrix} \tag{13}$$

For points on equator second (y) component should be zero after rotation,

$$\left(1 - \frac{R}{L}\right)\sin\beta = \frac{p_y}{L}\cos\beta \tag{14}$$

$$\tan\beta = \frac{p_y}{L - R},$$

or in scale of R

$$\tan\beta = \frac{y}{\sqrt{1 + x^2 + y^2} - 1}, \quad \text{where} \quad x = p_x/R, \quad y = p_y/R. \tag{15}$$

16 Evaluation of angle φ

Any point from equatorial region should have the same $\tan\beta$ in Eqn.15. Then, for two points (x_1, y_1) and (x_2, y_2) rotated by angle φ

$$\frac{x_1\sin\varphi + y_1\cos\varphi}{d_1} = \frac{x_2\sin\varphi + y_2\cos\varphi}{d_2}, \quad \text{where} \quad d_i = \sqrt{1 + x_i^2 + y_i^2} - 1$$

that brings us to

$$\tan\varphi = \frac{y_2 d_1 - y_1 d_2}{x_1 d_2 - x_2 d_1} \tag{16}$$

Python method:

```
from pyimgalgos.FiberAngles calc_phi
```

17 Evaluation of angle β

When angle φ is known, then for each point from equatorial region angle beta can be defined using Eqn.15,

$$\tan\beta = \frac{x\sin\varphi + y\cos\varphi}{\sqrt{1 + x^2 + y^2} - 1} \tag{17}$$

Python method:

```
from pyimgalgos.FiberAngles calc_beta
```

18 Evaluation of $y(x, \varphi, \beta)$ for $l = 0$ lattice nodes

Eqn.14 transformed to Eqn.17 can be used to evaluate dependence $y(x, \varphi, \beta)$. Applying notations

$$t = \tan\beta, \quad s = \frac{\sin\varphi}{t}, \quad \text{and} \quad c = \frac{\cos\varphi}{t} \tag{18}$$

it can be re-written as

$$\sqrt{1 + x^2 + y^2} = xs + yc + 1 \tag{19}$$

and further reduced to the form of quadratic equation

$$y^2 + 2By + C = 0, \tag{20}$$

which has two solutions

$$y = -B \pm \sqrt{B^2 - C}, \tag{21}$$

where

$$B = \frac{c(xs + 1)}{c^2 - 1} \quad \text{and} \quad C = \frac{x^2(s^2 - 1) + 2xs}{c^2 - 1}. \tag{22}$$

Correct sign of the root in Eqn.21 can be choosen from requitement that $y(x = 0) = 0$. In this case $C = 0$ and this requirement with Eqn.21 gives

$$y = -B \pm |B| = 0,$$

meaning that sign of the root and parameter B should be the same. Eqn.21 can be used to fit positions of peaks in equatorial region and get angles φ and β from fit parameters. In case of $\sin\beta = 0$ or $t = 0$ Eqn.17 gives

$$y = -x\tan\varphi. \tag{23}$$

Python method which parametrizes this solution and standard fitting method can be imported as

```
from pyimgalgos.FiberAngles import funcy_l0
from scipy.optimize import curve_fit
```

19 Evaluation of $y(x, \varphi, \beta)$ for $l = 1$ lattice nodes

Peaks in Arc region are associated with X-ray scattering on $l = 1$ lattice nodes. In this case Eqn.14 needs to be changed. Peak vertical positions for $l = 1$ lattice nodes on image after rotations φ and β should be located along the line separated by distance D from equatorial plane

$$(const.)\frac{D}{R} = \frac{p_y}{L}\cos\beta + \left(\frac{R}{L} - 1\right)\sin\beta, \tag{24}$$

where we use normalized value D/R, the same way like we do for p_x and p_y later. Multiplying Eqn.24 by $L/(R\sin\beta)$ and re-grouping terms we get

$$\frac{L}{R}\left(1 + \frac{D}{R\sin\beta}\right) = \frac{p_y}{R\tan\beta} + 1. \tag{25}$$

Substituting p_y with their rotated value $p_x\sin\varphi + p_y\cos\varphi$, (like in Eqn.17), using notations

$$x = \frac{p_x}{R},\ \ y = \frac{p_y}{R},\ \ t = \tan\beta,\ \ s = \frac{g\sin\varphi}{t},\ \ c = \frac{g\cos\varphi}{t},\ \text{ and }\ g = \left(1 + \frac{D}{R\sin\beta}\right)^{-1}, \tag{26}$$

we get equation

$$\frac{L}{R} \equiv \sqrt{1 + x^2 + y^2} = xs + yc + g, \tag{27}$$

which can be reduced to the form of quadratic Eqn.20 with solution Eqn.21, where

$$B = \frac{c(xs + g)}{c^2 - 1}\ \text{ and }\ C = \frac{x^2(s^2 - 1) + 2xsg + g^2 - 1}{c^2 - 1}. \tag{28}$$

In case of $\sin\beta = 0$ the last term of Eqn.24 disappears,

$$\frac{D}{R} = \frac{p_y}{L}\cos\beta,\ \ \text{or}\ \ \frac{L}{R} = \frac{p_x\sin\varphi + p_y\cos\varphi}{R}\frac{R}{D}\cos\beta \tag{29}$$

which brings us to Eqn. 27 with changed notations

$$s|_{\beta=0} = \frac{R}{D}\sin\varphi\cos\beta,\ \ c|_{\beta=0} = \frac{R}{D}\cos\varphi\cos\beta,\ \ \text{and}\ \ g|_{\beta=0} = 0. \tag{30}$$

Python method which parametrizes this solution and standard fitting method can be imported as

```
from pyimgalgos.FiberAngles import funcy_l1_v1
from scipy.optimize import curve_fit
```

20 References

1. Package pyimgalgos
2. Package cxif5315