MUNI
SCI

Kvazikrystal

Anna Richterková (Vice President)
Csaba Morvay (Senior Executive Assistant)
Petr Pazourek (CEO)

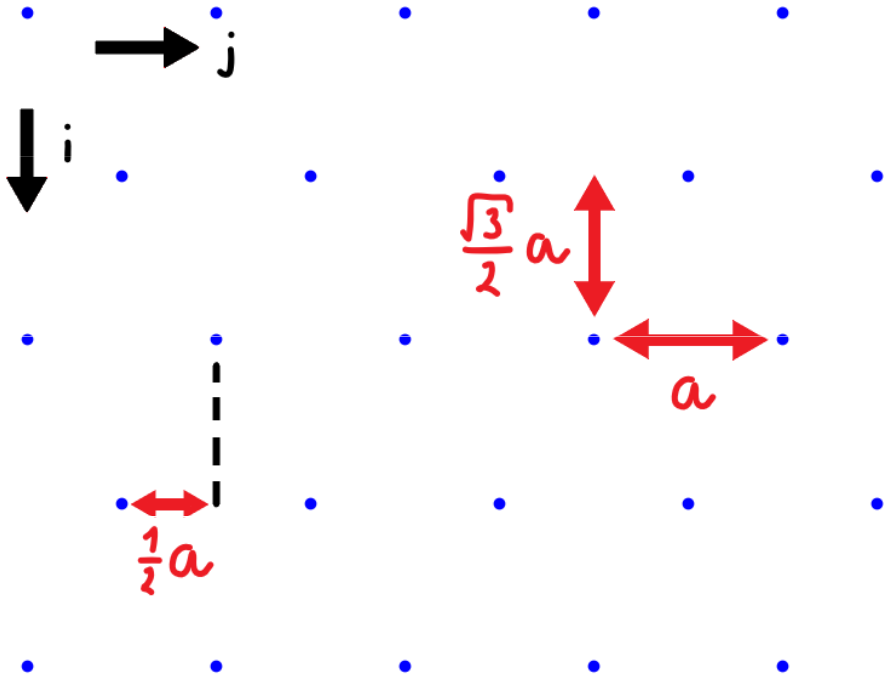
F6150 Pokročilé numerické metody
Jaro 2024

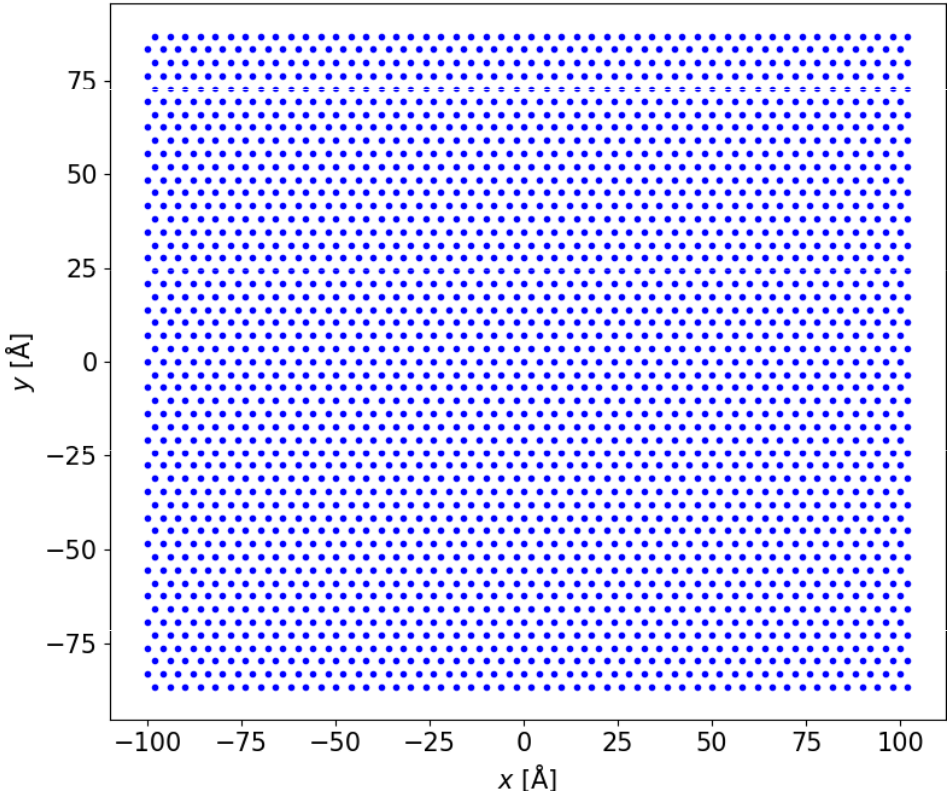
Zadání úlohy

- Zkonstruovat kvazikrystal a osadit ho vhodnými atomy – např. Gaussovými funkcemi
- Provést Fourierovu transformaci vzniklé hustoty
- Totéž provést pro hexagonální mřížku

Popis kódu

– Generování 2D hexagonální mřížky





MUNI SCI

Popis kódu

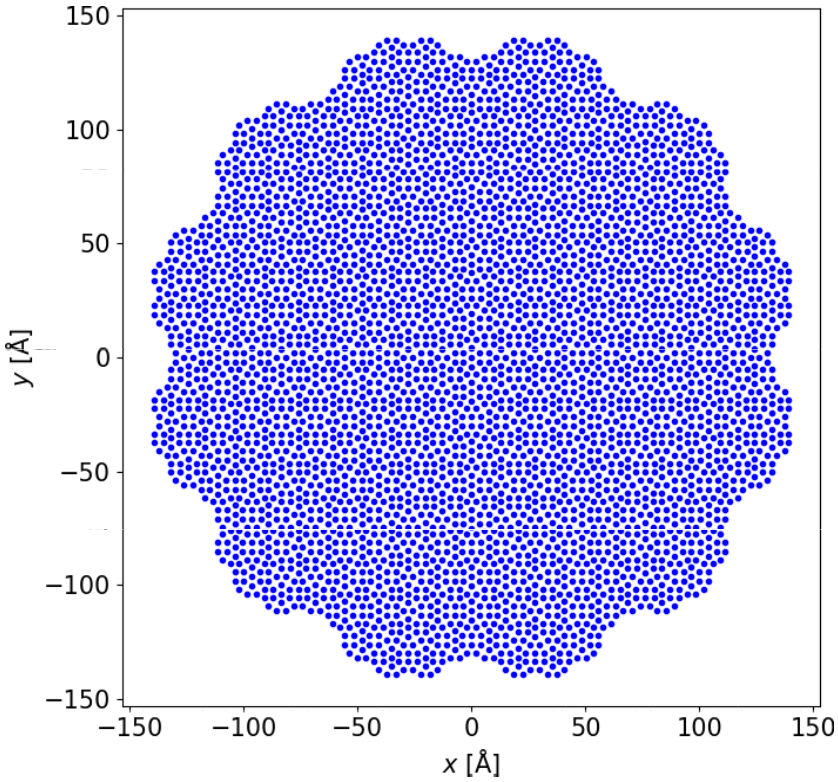
– Generování 2D hexagonální mřížky

```
#FUNKCE GENERUJÍCÍ 2D HEXAGONÁLNÍ MŘÍŽKU
#výstupem jsou 2 vektory: jeden udává postupně x-ové souřadnice bodů, druhý y-ové souřadnice bodů mřížky
@jit
def hexagonal_lattice(size, lattice_const):
    half_size = size // 2              #střed mřížky umístíme do počátku souřadnic
    x_coords, y_coords = [], []        #seznamy, do kterých budeme ukládat x-ové a y-ové souřadnice bodů mřížky

    for i in range(-half_size, half_size + 1):                    #i = index řádků
        for j in range(-half_size, half_size + 1):                #j = index sloupců
            x = j*lattice_const + (i % 2)*lattice_const/2         #x-ové souřadnice: j*lattice_const = poloha v horizontálním směru
                                                                  #(i % 2)*lattice_const/2 = posun každého druhého řádku o lattice_const/2
            y = i*np.sqrt(3)*lattice_const/2                      #y-ové souřadnice: i*np.sqrt(3)*lattice_const/2 = poloha ve vertikálním směru
            x_coords.append(x)                                    #uložení souřadnic do seznamů
            y_coords.append(y)

    return np.array(x_coords), np.array(y_coords)
```


MUNI SCI

Popis kódu

– Generování 2D kvazikrystalu



MUNI SCI

Popis kódu

– Generování 2D kvazikrystalu

```
#FUNKCE GENERUJÍCÍ 2D KVAZIKRYSTAL
#výstupem je seznam s body kvazikrystalu
@jit
def quasi_crystal(I, lattice_const):
    a = lattice_const * (2+np.sqrt(3))**I  #škálování mřížkové konstanty
    center = np.array([0.0, 0.0])          #výchozí bod, do kterého umístíme střed opakujícího se útvaru v první iteraci

    points_list = []                       #do tohoto listu budeme ukládat body kvazikrystalu
    points_list.append(center)

    for i in range(I):
        #škálování charakteristických rozměrů
        r_1 = a/(2+np.sqrt(3))**(i+1)
        r_2 = (np.sqrt(3/2)+1/np.sqrt(2)) * a/(2+np.sqrt(3))**(i+1)
        r_3 = r_2

        points_listTEMP = []    #vytvoření dočasného seznamu, bez něj by se nám to celé zacyklilo
        for m in points_list:
            #umisťování středu opakujícího se vzoru do správných poloh; polohy a jejich počet závisí na iteraci
            x_0 = m[0]
            y_0 = m[1]
            for n in range(5):
                #vytvoření opakujícího se vzoru v dané poloze
```


MUNI SCI

Popis kódu

– Generování 2D kvazikrystalu

```
        for n in range(5):
            #vytvoření opakujícího se vzoru v dané poloze
            theta_n = n*np.pi/3
            phi_1 = np.pi/6
            phi_2 = np.pi/12
            phi_3 = np.pi/4
            point_1 = np.array([ x_0 + r_1*np.cos(phi_1+theta_n) , y_0 + r_1*np.sin(phi_1+theta_n) ])
            point_2 = np.array([ x_0 + r_2*np.cos(phi_2+theta_n) , y_0 + r_2*np.sin(phi_2+theta_n) ])
            point_3 = np.array([ x_0 + r_3*np.cos(phi_3+theta_n) , y_0 + r_3*np.sin(phi_3+theta_n) ])

            #tyto body uložíme do dočasného seznamu
            points_listTEMP.append(point_1)
            points_listTEMP.append(point_2)
            points_listTEMP.append(point_3)

    #až na konci celé iterace smažeme duplicitní body pomocí "remove_duplicate()"
    #poté přesypeme obsah dočasného seznamu do trvalého seznamu, jinak, jak už bylo řečeno, by se kód zacyklil
    points_list.extend(remove_duplicate(points_listTEMP, threshold))
return points_list
```


MUNI SCI

Popis kódu

– Mazání duplicitních bodů 2D kvazikrystalu

```
#FUNKCE MAZAJÍCÍ DUPLICITNÍ BODY V KVAZIKRYSTALU
@jit
def remove_duplicate(arraylist, threshold):
    unique_list = []                            #seznam pro unikátní body
    #iterujeme přes všechny body a každý bod porovnáváme s ostatními body
    for i in range(len(arraylist)):
        duplicate = False
        #druhá iterace "j" je o 1 index posunutá dopředu
        #tímto nebudeme porovnávat každý bod sám se sebou a také nebudeme porovnávat body, které jsme už porovnaly v předchozích iteracích
        for j in range(i+1, len(arraylist)):
            if np.linalg.norm(arraylist[i] - arraylist[j]) < threshold:     #pokud je velikost rozdílu bodů menší než předem určená mezní hodnota,
                duplicate = True                                            #tak bod "i" je duplicitní (hodnota "duplicate" je True) a pokračujeme další hodnotou "j"
                break
        if not duplicate:                                                   #pokud bod "i" není duplicitní (hodnota "duplicate" je False),
            unique_list.append(arraylist[i])                                #tak ho přidáme do seznamu unikátních bodů
    return unique_list
```


MUNI SCI

Popis kódu

– Vkládání Gaussových funkcí do bodů (kvazi)krystalu

```
#DEFINOVÁNÍ GAUSSOVY FUNKCE
@jit
def gaussian(x, y, x0, y0, A, sigma):
    B = 2*sigma**2
    return A*np.exp(-((x-x0)**2 + (y-y0)**2) / B)
```

```
#vytvoření pole, ve kterém budeme počítat hodnoty Gaussovek reprezentující jednotlivé atomy tvořící mřížku
#pole je tvořené rovnoměrně vzdálenými body pokrývajícími celý prostor kvazikrystalu
x_minq, x_maxq = np.min(x_q), np.max(x_q)       #výpočet okrajových hodnot pole - bereme největší a nejmenší x-ové a y-ové souřadnice bodů kvazikrystalu
y_minq, y_maxq = np.min(y_q), np.max(y_q)
x_gridq = np.linspace(x_minq, x_maxq, N_q)
y_gridq = np.linspace(y_minq, y_maxq, N_q)
X_q, Y_q = np.meshgrid(x_gridq, y_gridq)        #vytvoření pole

#umístění Gaussovky do každého bodu kvazikrystalu a součet všech Gaussovek
I_q = np.zeros_like(X_q)        #vytvoření matice naplněné nulami, která má stejný tvar a typ jako "X_q"
for x0, y0 in points_list:
    I_q = I_q + gaussian(X_q, Y_q, x0, y0, A_q, sigma_q)

#přenásobení celého kvazikrystalu gaussovkou - potlačení artefaktů
I_q = I_q*gaussian(X_q, Y_q, 0, 0, A_q, biggauss_q)
```


MUNI SCI

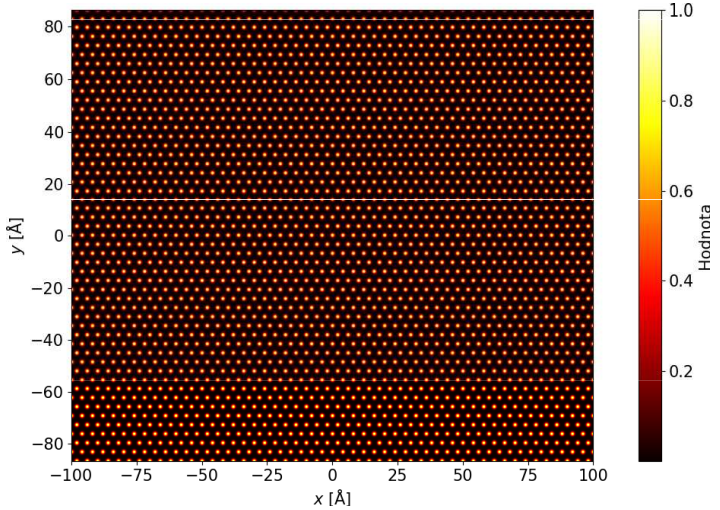
Popis kódu

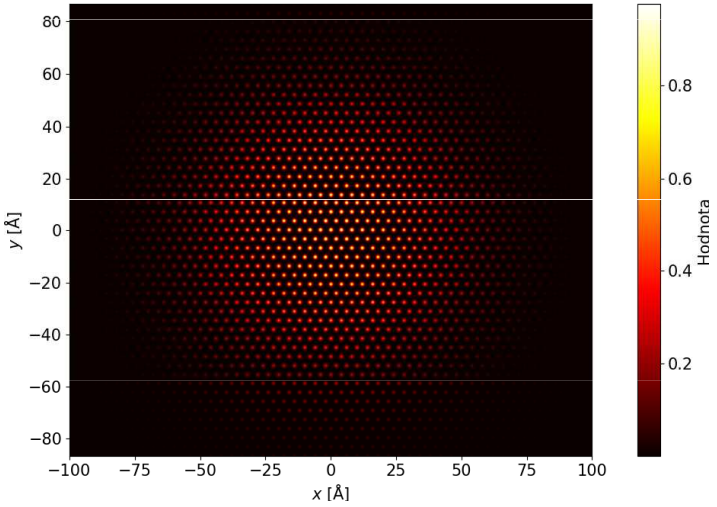
– Provedení Fourierovy transformace

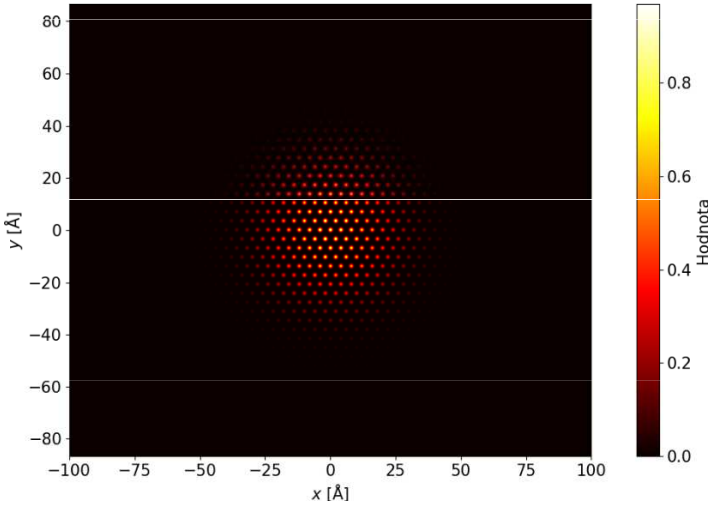
```
#PROVEDENÍ FOURIEROVY TRANSFORMACE
data_q = imageio.imread("test_q.png", mode="L")     #přečtení přímého obrazu kvazikrystalu
c_q = scipy.fft.fft2(data_q)                        #provedení 2D Fourierovy transformace
c_q = scipy.fft.fftshift(c_q)                       #prohození 1. a 3. kvadrantu, a 2. a 4. kvadrantu
```

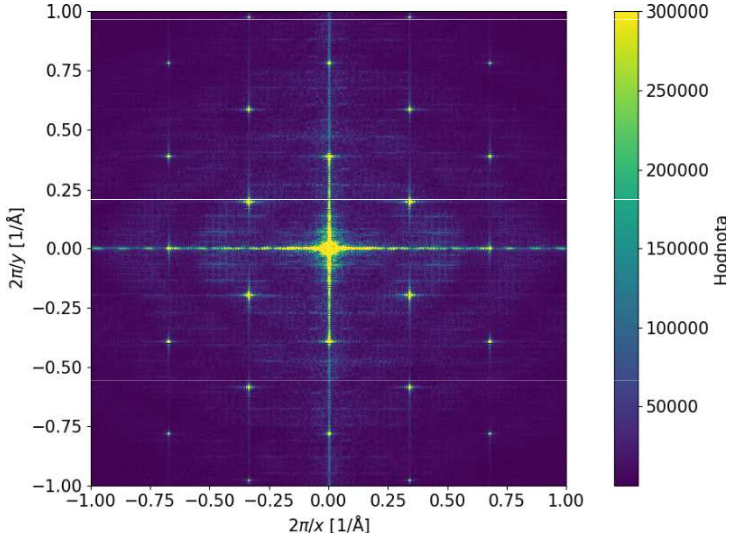

MUNI SCI

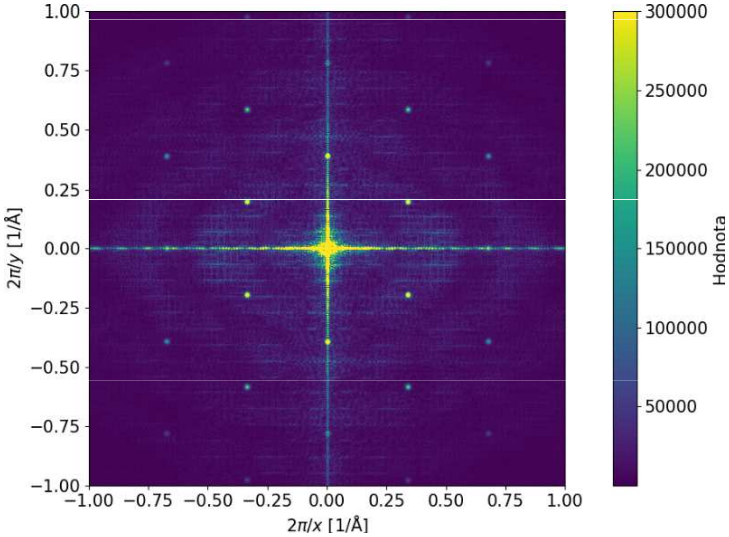
Výsledky – vliv přenásobení Gaussovkou: bez, 34, 17

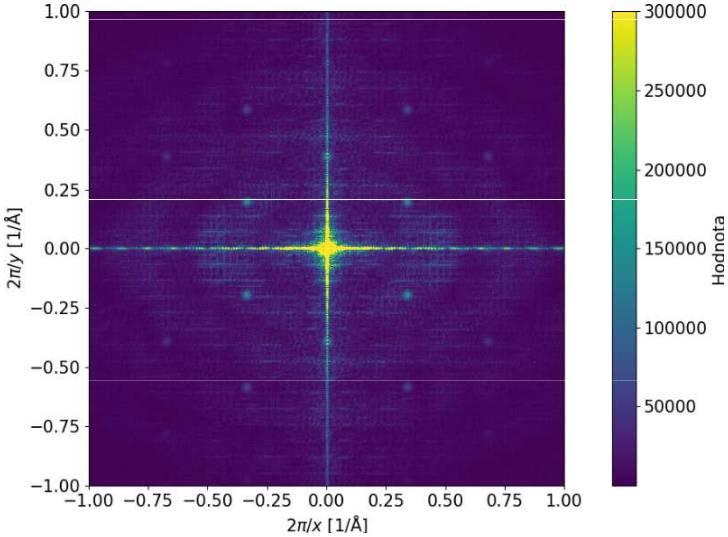




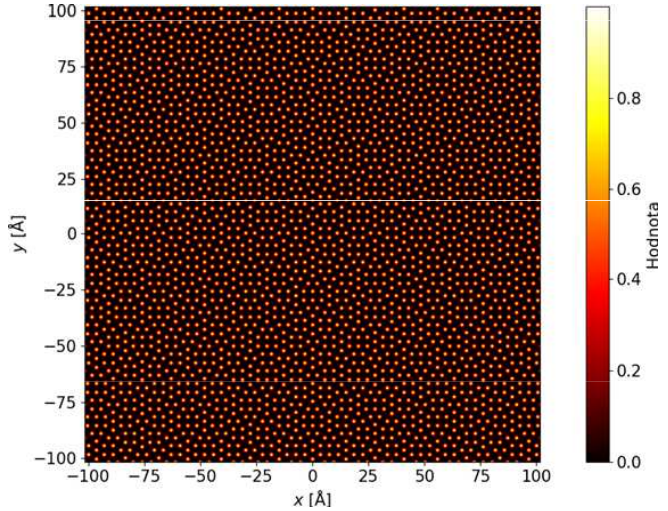


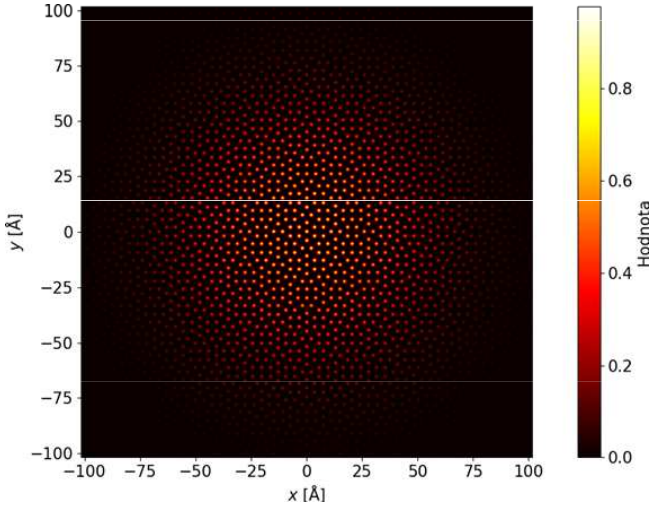


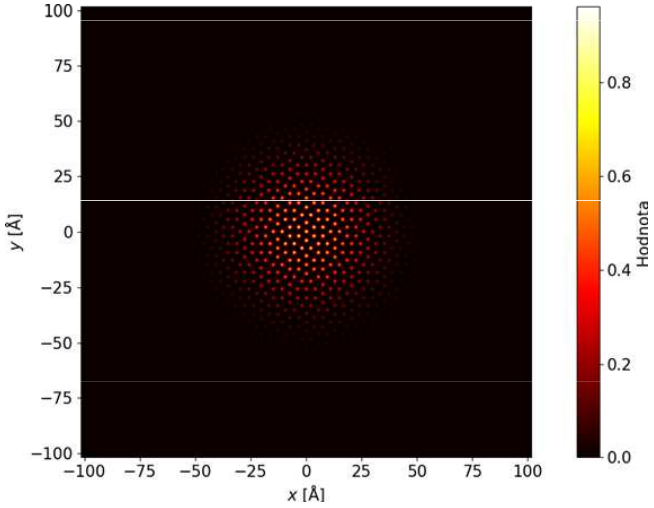


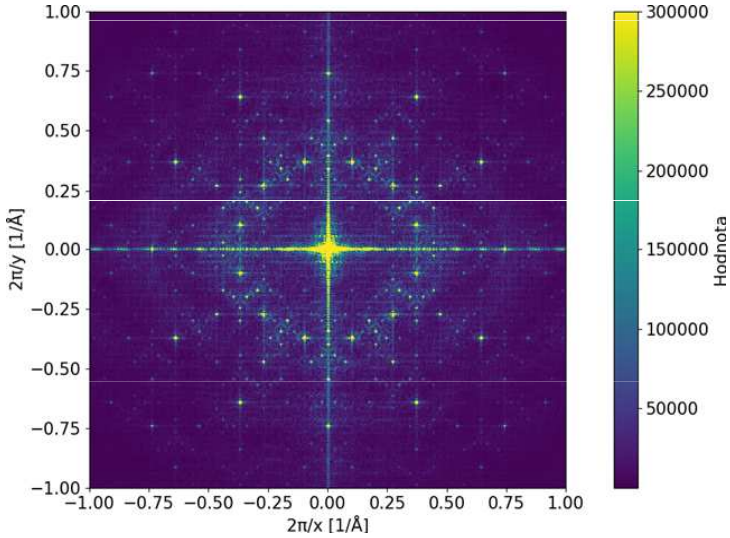


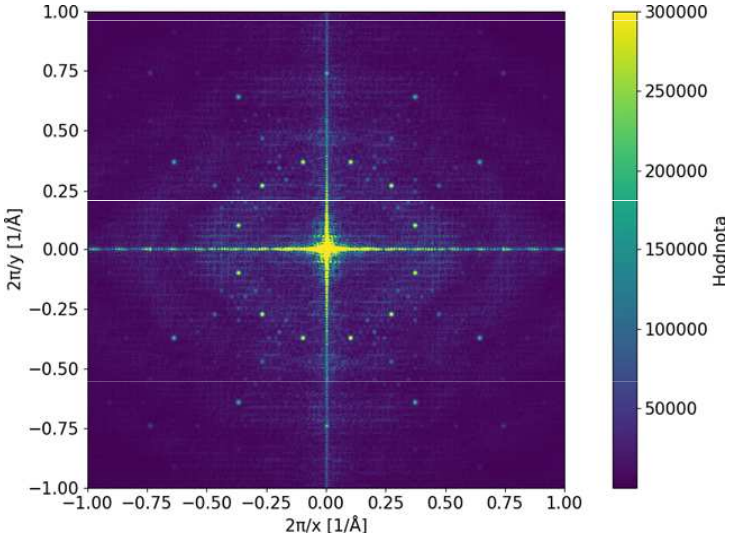
Výsledky – vliv přenásobení Gaussovkou: bez, 40, 20

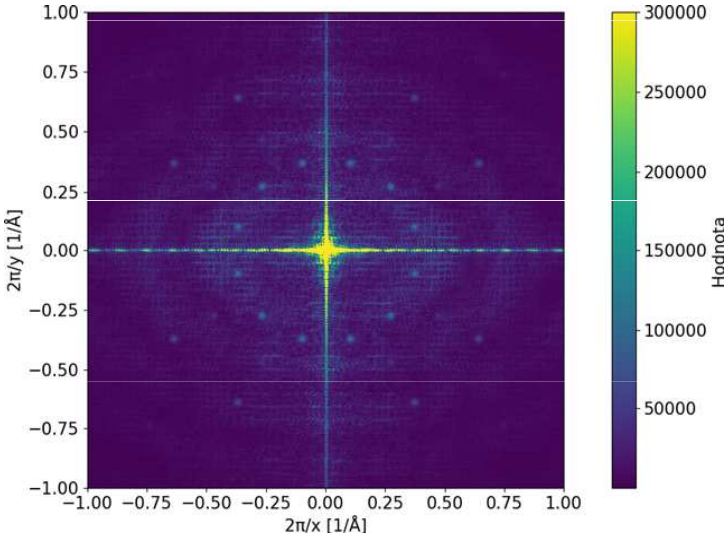




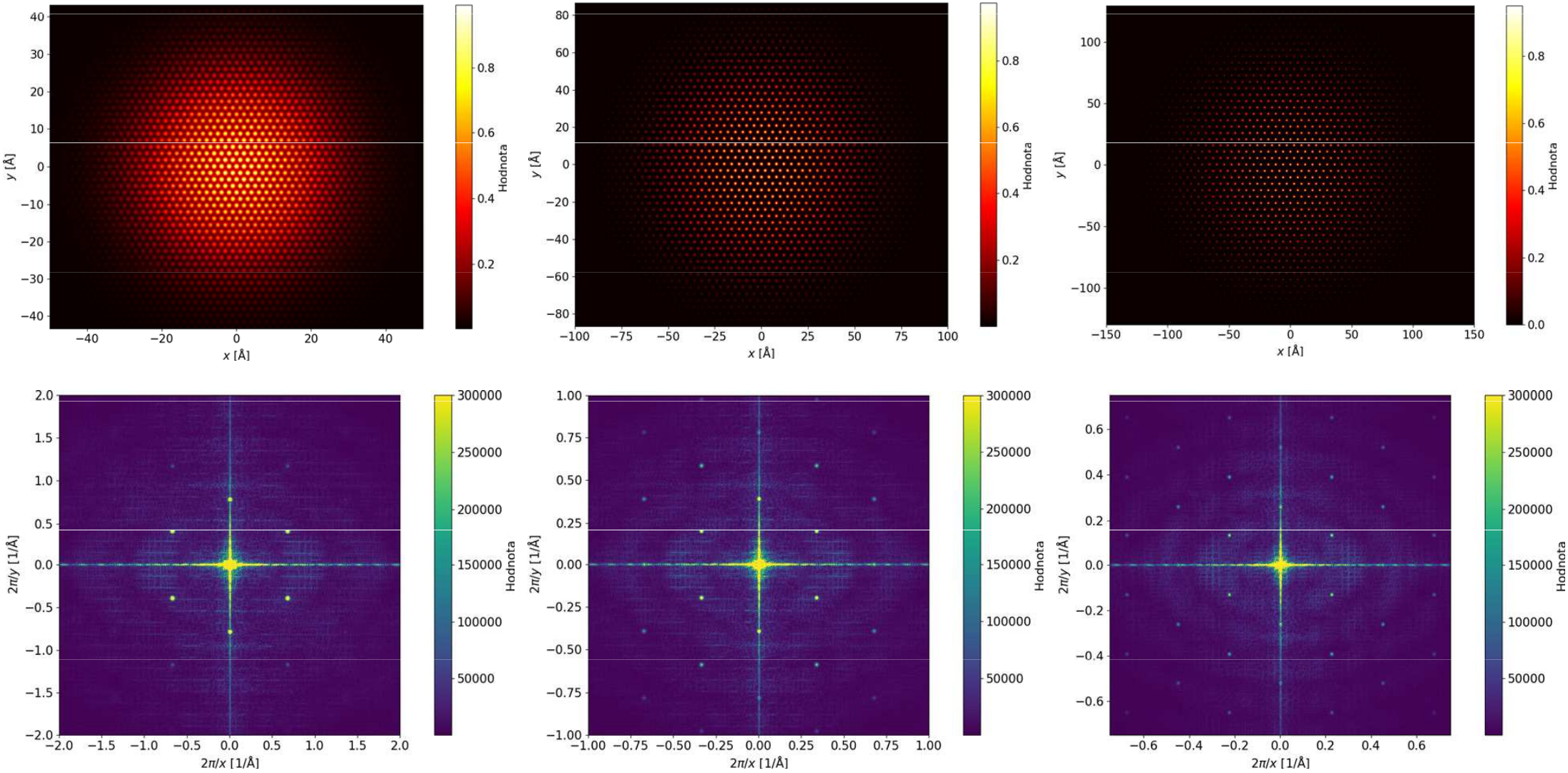




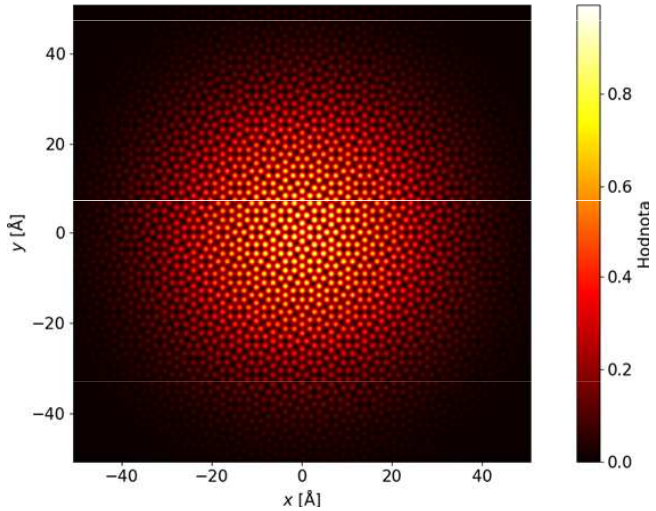


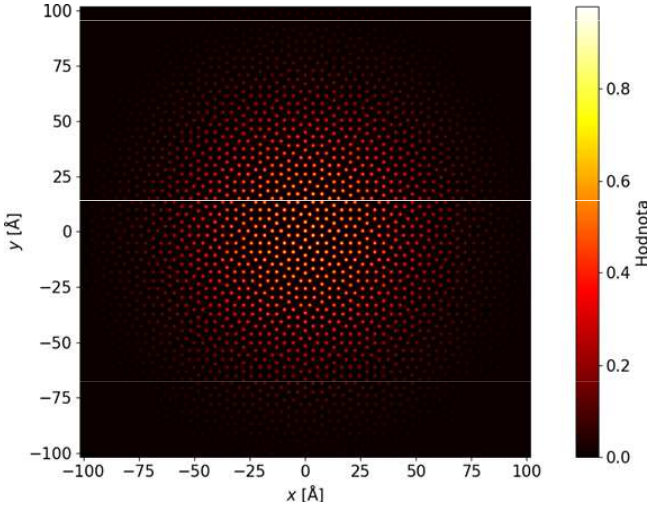


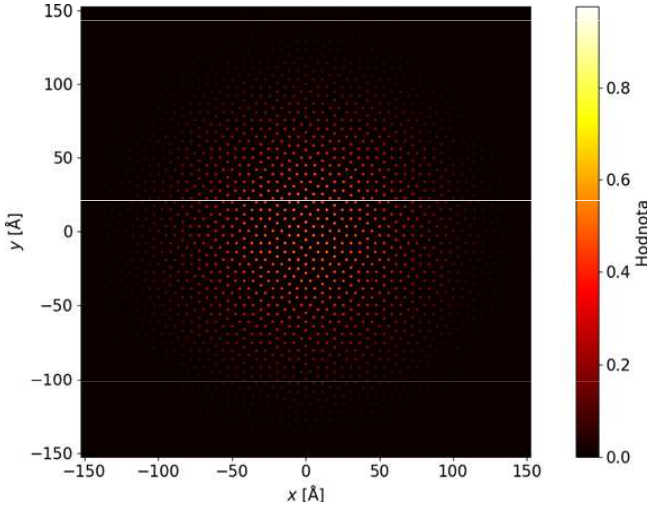
Výsledky – vliv velikosti mříž. konst.: a = 2.0, 4.0, 6.0

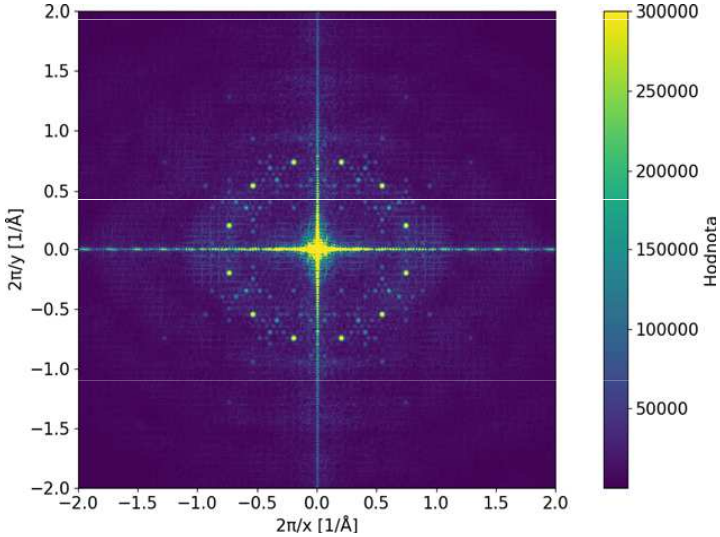


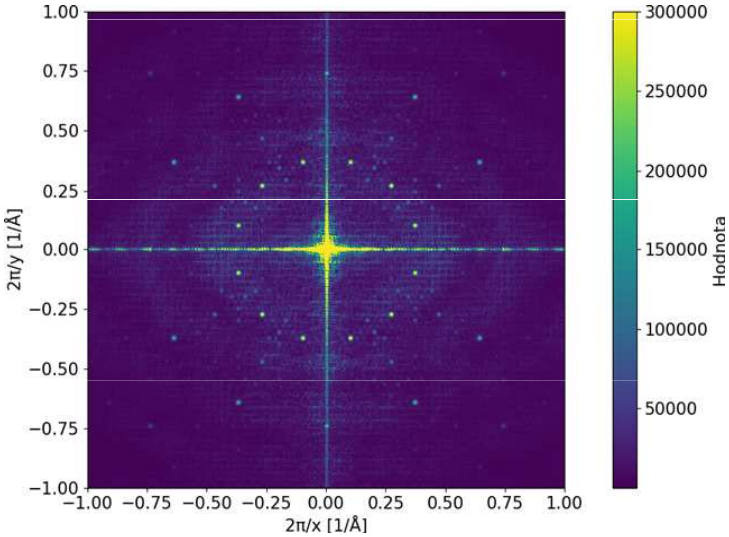
Výsledky – vliv velikosti mříž. konst.: a = 2.0, 4.0, 6.0

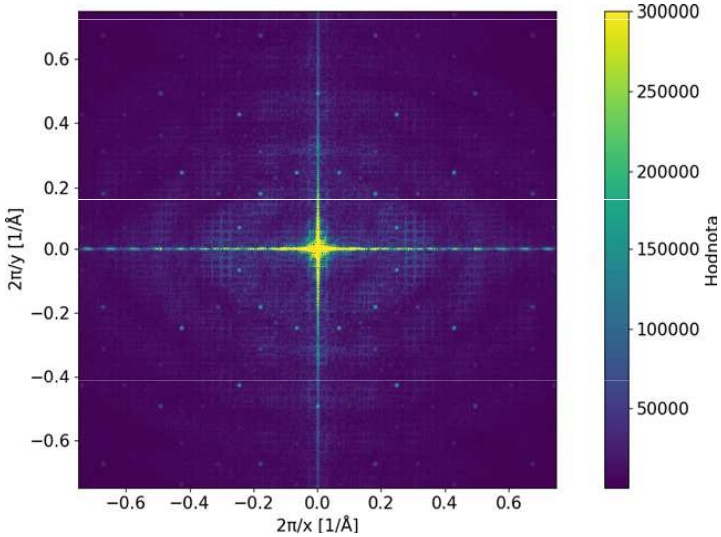




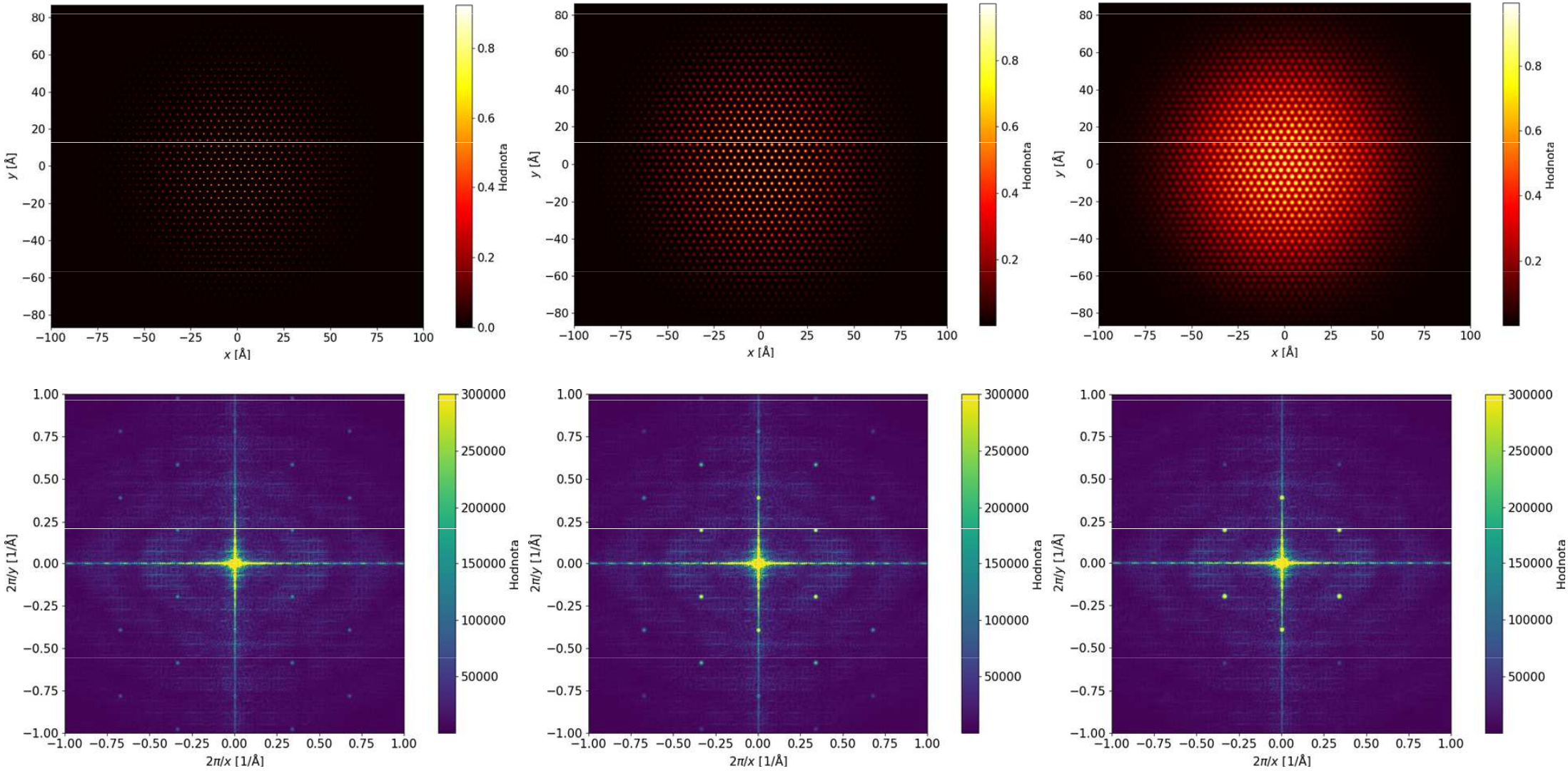




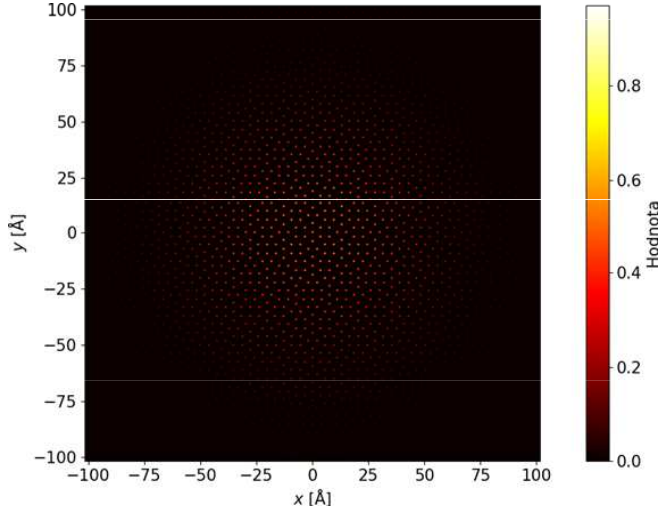


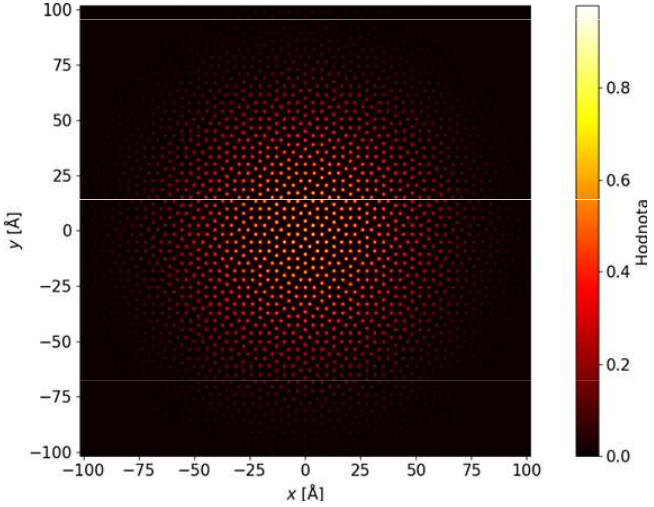


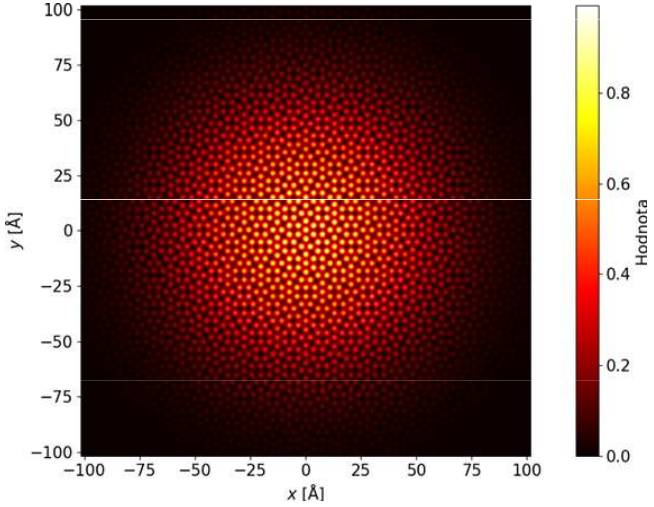
Výsledky – vliv velikosti atomů: σ = 0.25, 0.5, 1.0

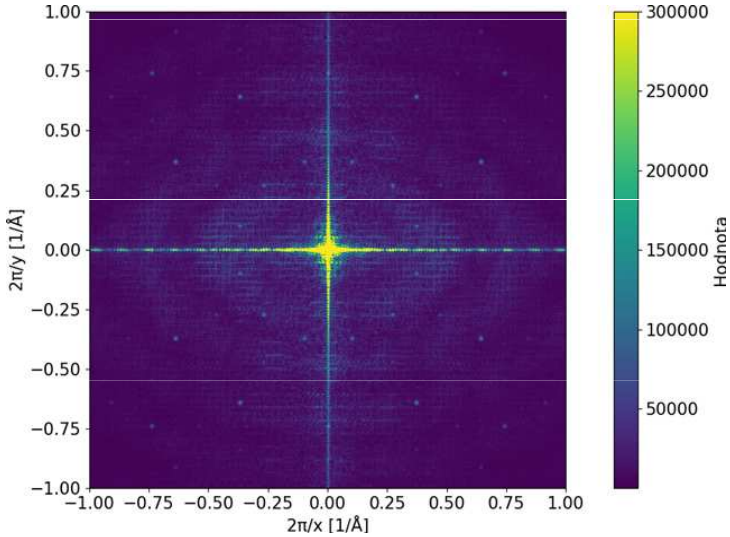


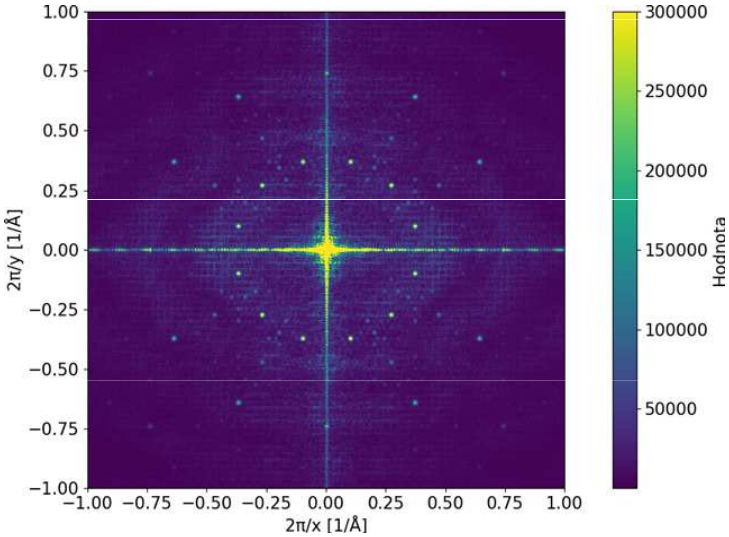
Výsledky – vliv velikosti atomů: σ = 0.25, 0.5, 1.0

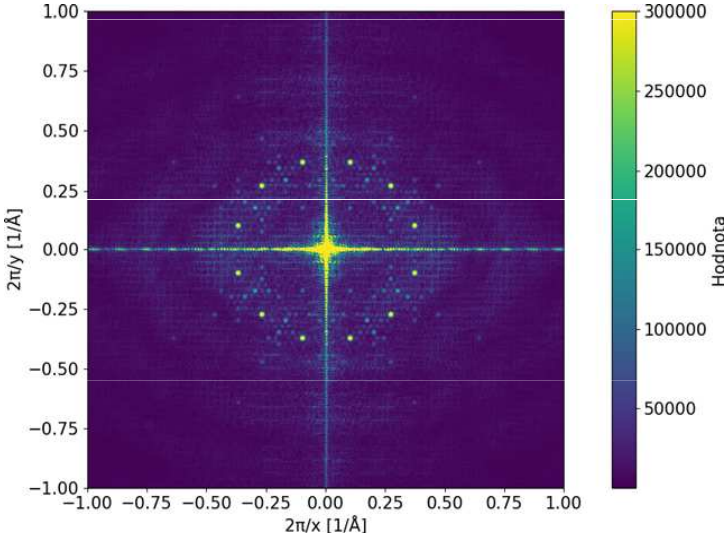




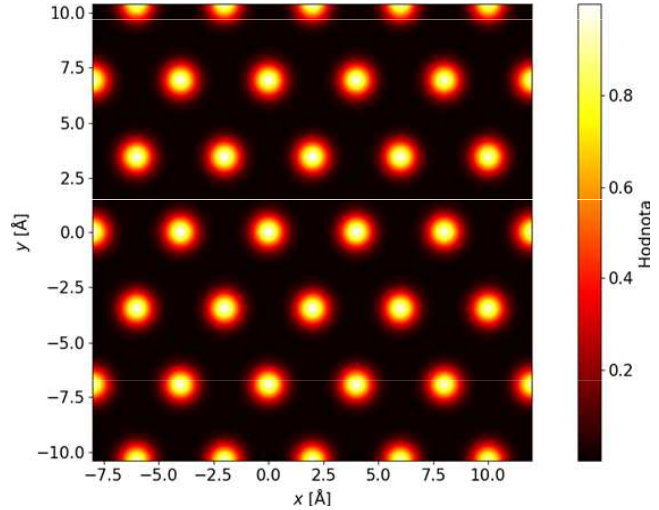


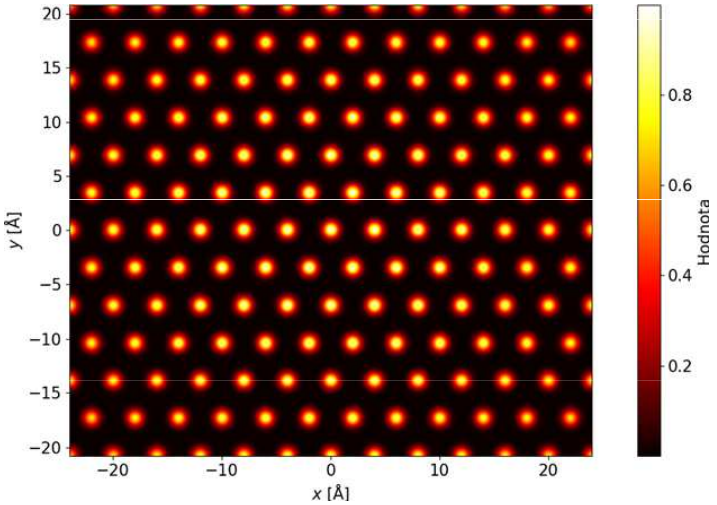


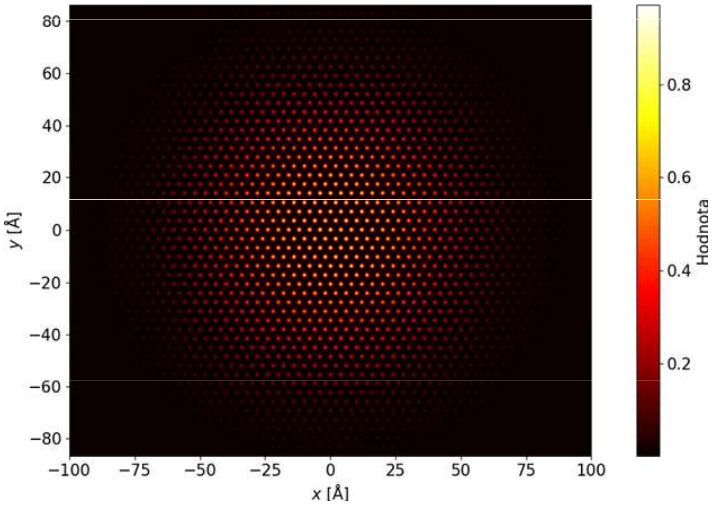


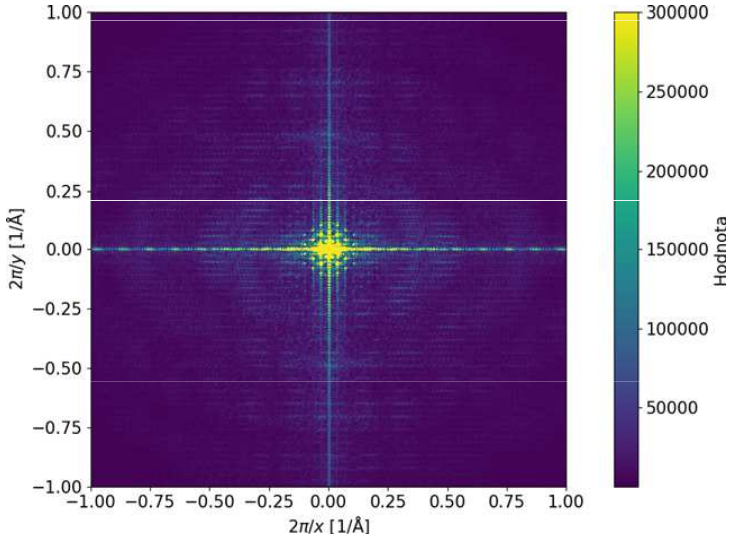


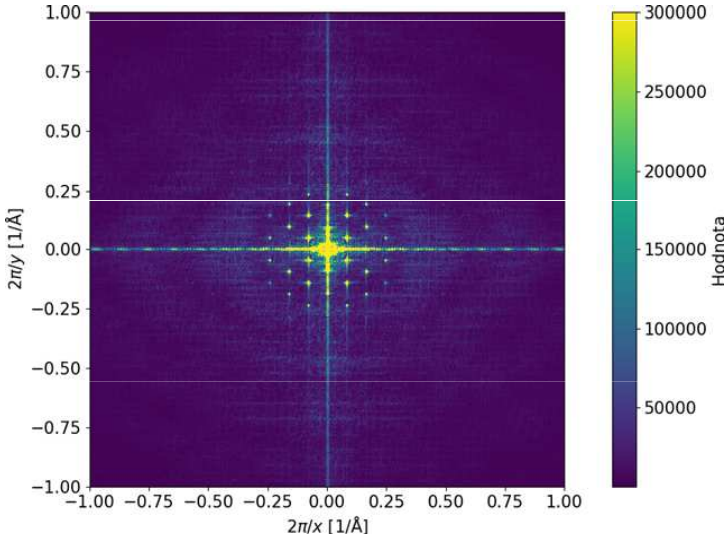
Výsledky – vliv počtu atomů: size = 6, 12, 50

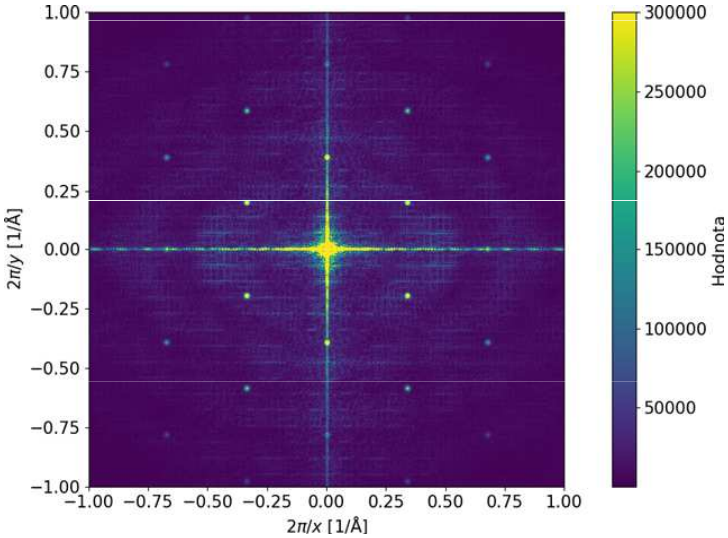




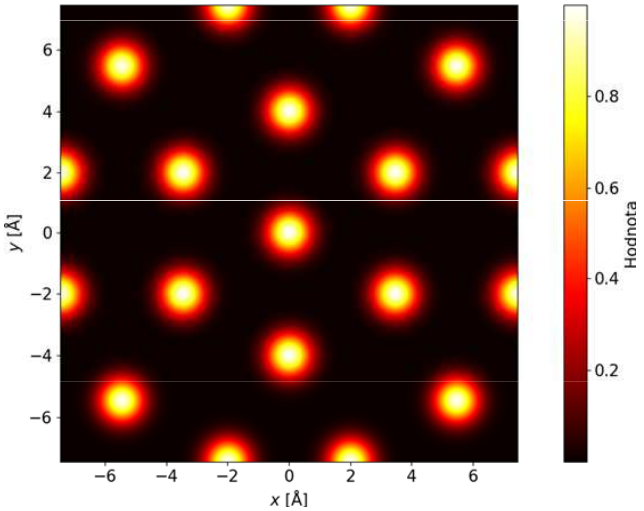


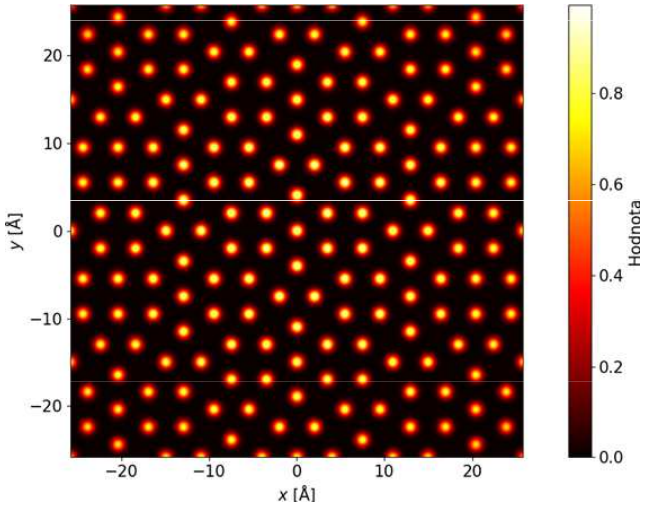


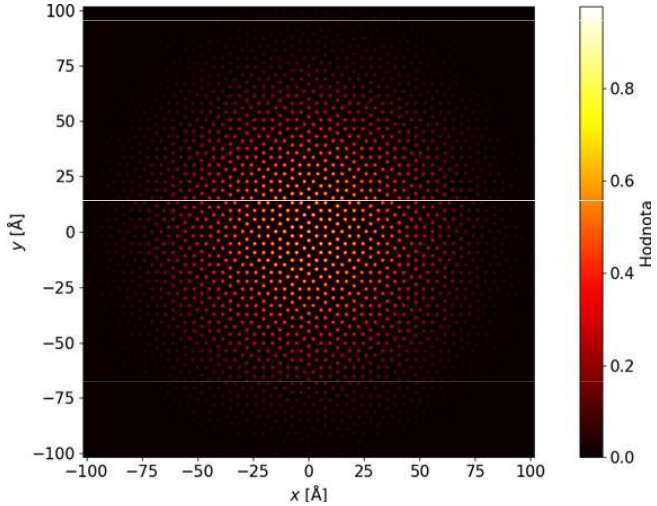


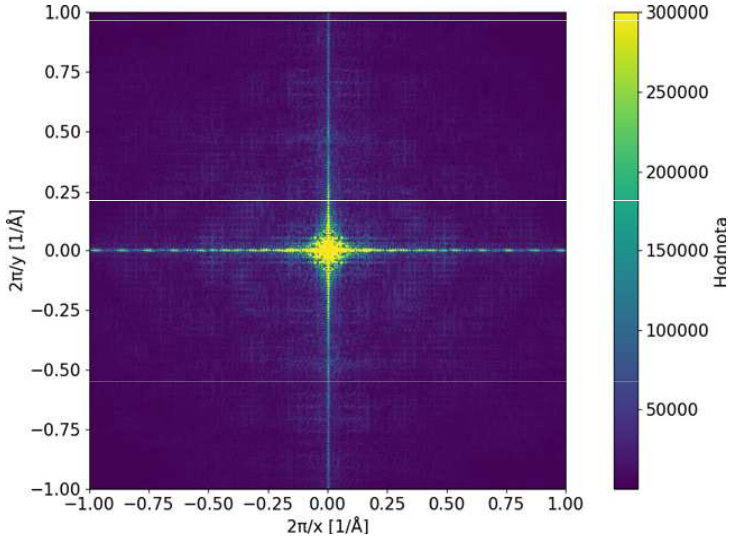


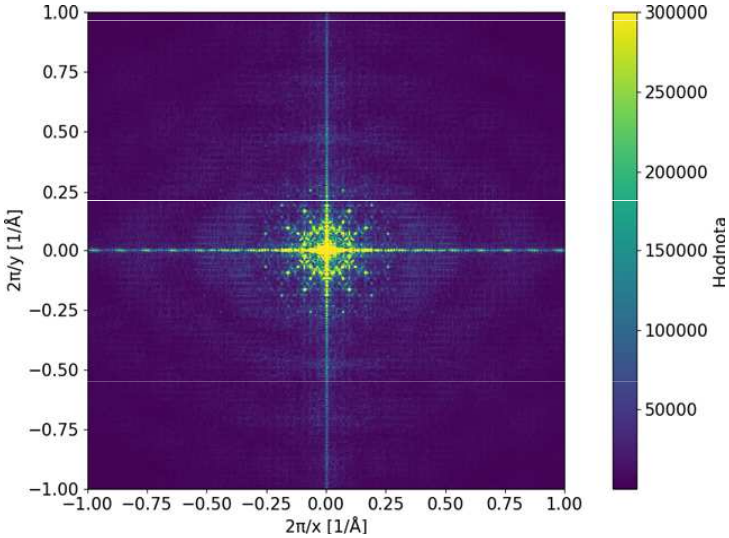
Výsledky – vliv počtu atomů: I = 1, 2, 3

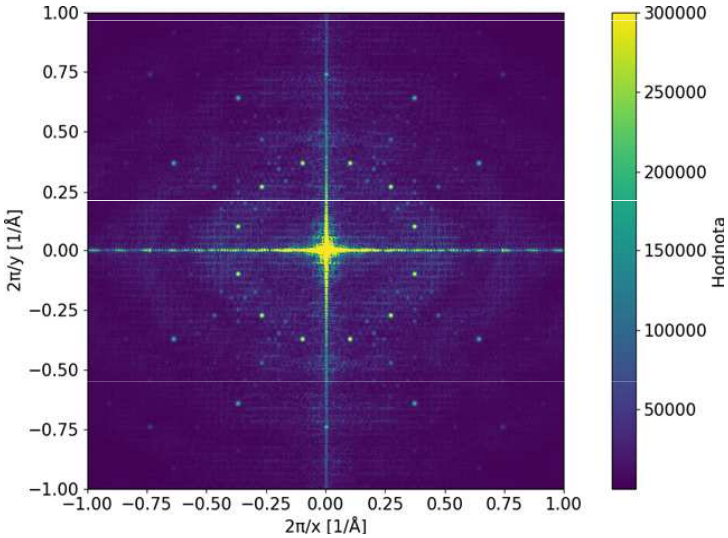












Tabulka parametrů

Parametr	h5	h2	h4	h1	h2	h3	h6	h2	h7	h8	h9	h2
size_hex	50	50	50	50	50	50	50	50	50	6	12	50
lattice_const_hex	4.0	4.0	4.0	2.0	4.0	6.0	4.0	4.0	4.0	4.0	4.0	4.0
A_hex	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
sigma_hex	0.5	0.5	0.5	0.5	0.5	0.5	0.25	0.5	1.0	0.5	0.5	0.5
N_hex	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
biggauss_hex	bez	34	17	17	34	51	34	34	34	34	34	34
reciprok	+-1	+-1	+-1	+-2	+-1	+-0.75	+-1	+-1	+-1	+-1	+-1	+-1
	Vliv přenásobení Gaussovkou			Vliv velikosti mřížkové konstanty			Vliv velikosti atomů			Vliv počtu atomů		
Parametr	q5	q2	q4	q1	q2	q3	q6	q2	q7	q8	q9	q2
I	3	3	3	3	3	3	3	3	3	1	2	3
lattice_const_q	4.0	4.0	4.0	2.0	4.0	6.0	4.0	4.0	4.0	4.0	4.0	4.0
threshold	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1
A_q	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0	1.0
sigma_q	0.5	0.5	0.5	0.5	0.5	0.5	0.25	0.5	1.0	0.5	0.5	0.5
N_q	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000	1000
biggauss_q	bez	40	20	20	40	60	40	40	40	40	40	40
reciprok	+-1	+-1	+-1	+-2	+-1	+-0.75	+-1	+-1	+-1	+-1	+-1	+-1

MUNI SCI

MASARYKOVA
UNIVERZITA