# efficiency

November 18, 2025

[4]:
```
import matplotlib.pyplot as plt
import numpy as np
from numpy import pi
from scipy.optimize import curve_fit
from scipy.constants import *
from engineering_notation import EngNumber

import BBO

%matplotlib inline
# %matplotlib widget
```

$$P = \frac{NE}{t} = \frac{hc}{\lambda}\frac{N}{t}$$

$$\eta = \frac{P_{out}}{P_{in}}$$

Z doświadczeń dane pokazuję że wydajność wynosi:

[5]:
```
P0 = 2.5e-3
l = 370e-9
N0dot = P0 / (h * c / (l))
Ndot = 100e3
print(EngNumber(Ndot / N0dot))
```

```
21.48p
```

## 0.1 Obliczanie wydajności

### 0.1.1 W oparciu o "Handbook of Nonlinear Optical Crystals" V.M. Dimitrieva

Dla odwrotnego SHG ($\omega_2 \to \omega_1 + \omega_1$) (tab. 2.19):

$$\eta^{-1} = \frac{2\pi^2}{\varepsilon_0 c}\frac{d_{eff}^2}{n_1^2 n_2^2}\frac{L^2}{\lambda_2^2}\frac{P}{A}$$

Gdzie $d_{eff}$ jest efektywną podatnością elektryczną, która najogólniej wynosi (eq. 2.46):

$$P_i = \varepsilon_0 \chi_{ij} E_j + 2 d_{ik} E_k^2$$

W związku symetriami Kleinmana (eq. 2.48) (które działają przy **braku** dyspersji) $d_{31} = d_{15}$, wiedząc że BBO jest kryształem klasy 3 (3.1.18), to z (tab. 2.3)

$$d_{eff,BBO} = d_{15} \sin\theta - d_{22} \cos\theta \sin 3\varphi$$

gdzie $\theta$ jest kątem do osi kryształu. Dane z (3.1.18):

$$d_{22} = (2.22 \pm 0.09) \times 10^{-12} \text{ m/V}$$

$$d_{31} = (0.16 \pm 0.08) \times 10^{-12} \text{ m/V}$$

### Ale Z (tab. 4.16) dane jest że $d_{eff,BBO} = 0.3 \times d_{eff,KDP}$, a z 3.1.1 dla KDP ($KH_2PO_4$):

$$d_{eff,KDP} \approx d_{36,KDP}(\lambda = 0.63\mu \text{ m}) = 7.1 \times 10^{-13} \text{ m/V}$$

[6]:
```
def efficiency(deff, n1, n2, L, l2, P, A):
    return (
        2 * pi**2 / epsilon_0 * deff / (n1 * n2) ** 2 * L**2 / (l2**2) * P / A
    ) ** -1


l = 0.370
L = 1e-3
A = (0.3e-3) ** 2  # na oko

deffBBO = 2.22e-12  # m/V
print(EngNumber(efficiency(deffBBO, BBO.no(2 * l), BBO.ne(l), L, l * 1e-6, P0,␣
 ↪A)))

deffKDP = 7e-13  # m/V
deffBBO = 0.3 * deffKDP
print(EngNumber(efficiency(deffBBO, BBO.no(2 * l), BBO.ne(l), L, l * 1e-6, P0,␣
 ↪A)))

deffBBO = 0.16e-12  # m/V
print(EngNumber(efficiency(deffBBO, BBO.no(2 * l), BBO.ne(l), L, l * 1e-6, P0,␣
 ↪A)))
```

```
6.83p
72.17p
94.72p
```

Z **grubsza** można oszacować:

$$\eta = (6 \div 95) \times 10^{-12}$$