Algorithms for a high-quality low-cost reconstructional spectrometer

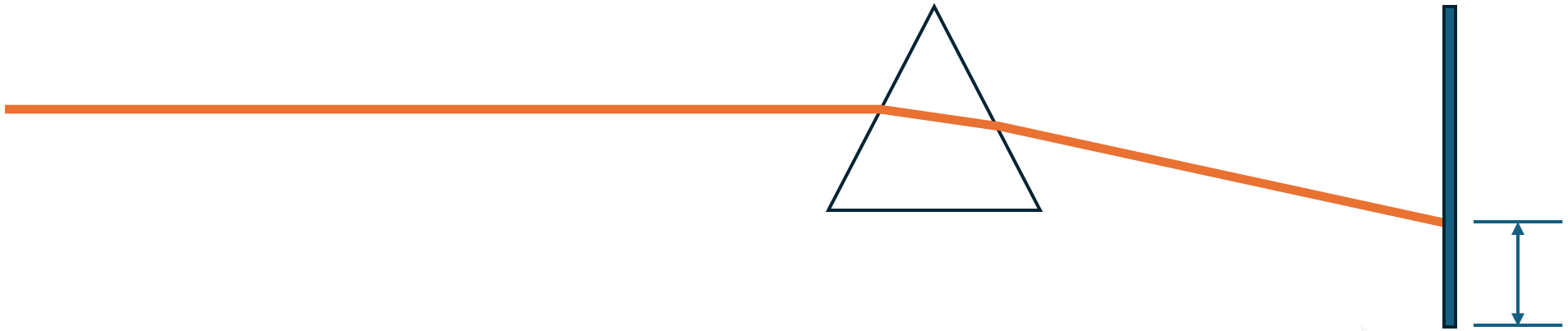
Yu Liu

Supervisor: Dr. Andrea Morandi

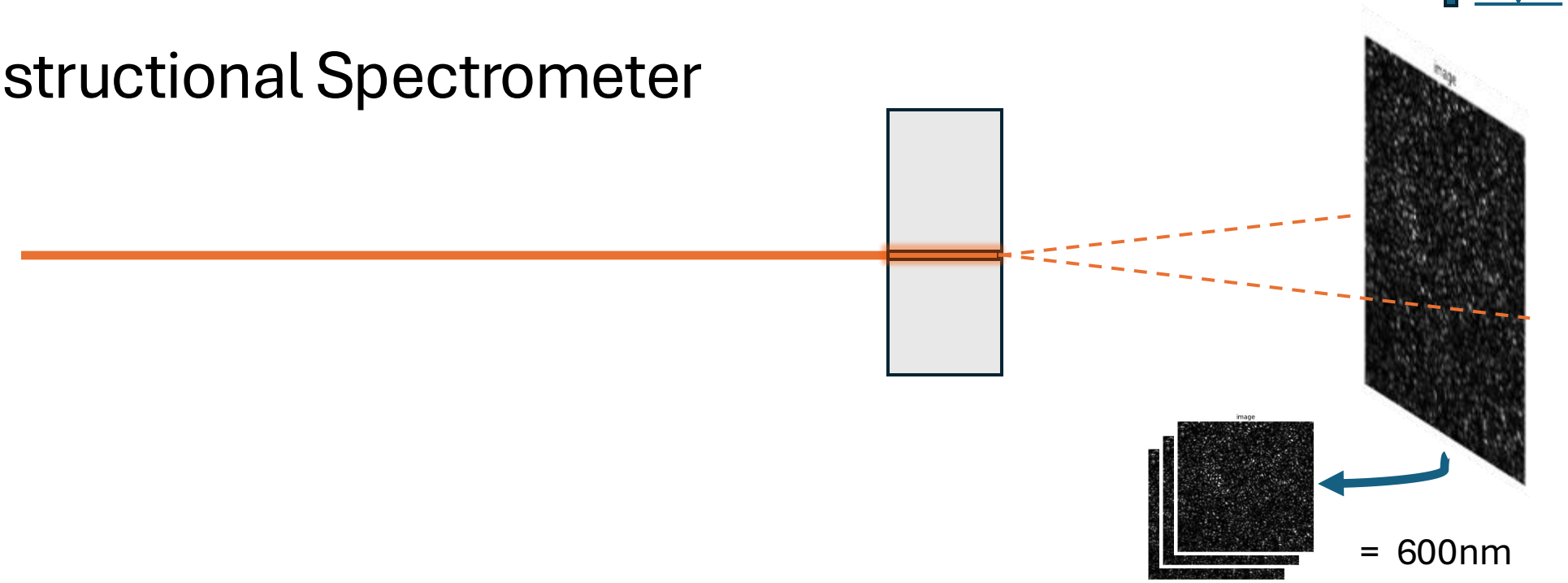
Outline

- **Introduction and goal of the project**
- **Reconstructional spectroscopy algorithms**
- **System setup**
- **Result**
  - **CW laser**
  - **Supercontinuum source**
- **Conclusion**
- **Outlook**

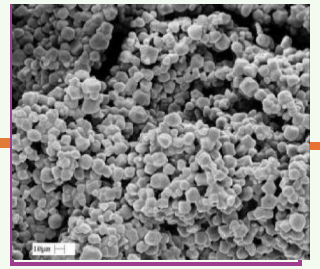
Spectrometer



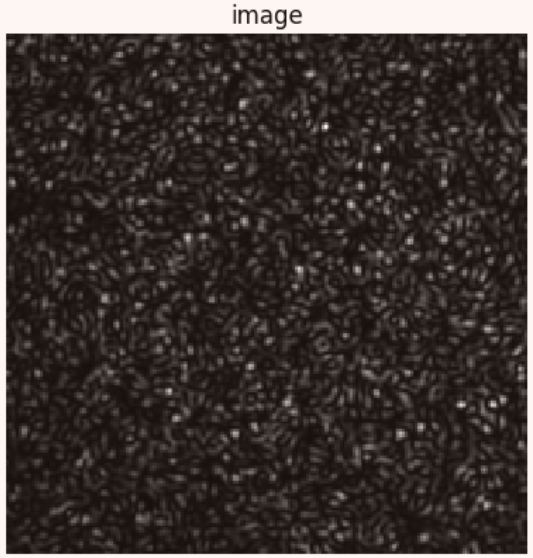
Reconstructional Spectrometer



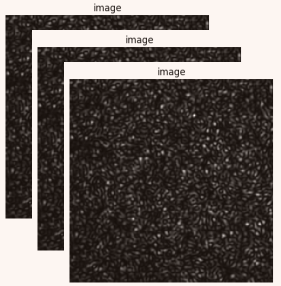
Reconstructional spectrometer



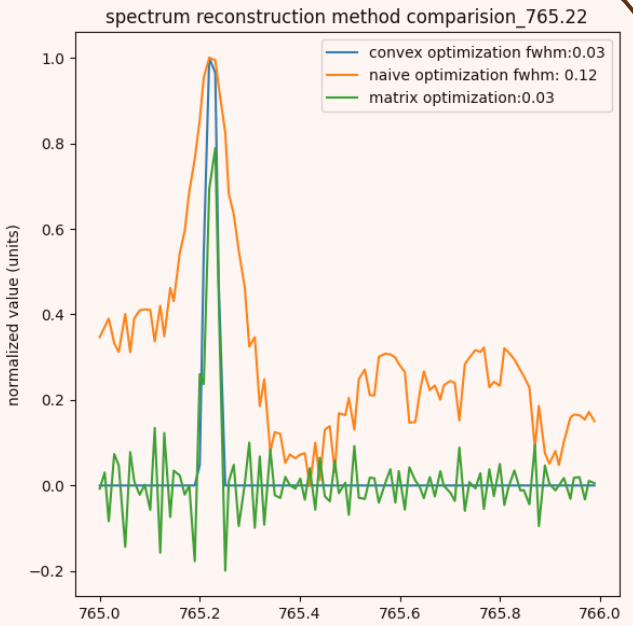
$LiNbO_2$



Speckle pattern



Reconstruction



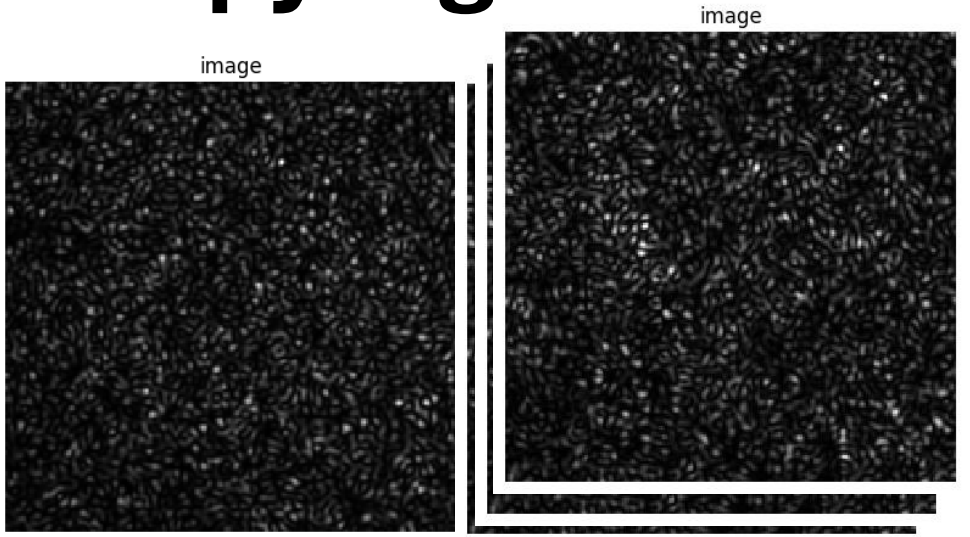
Spectrum

Goal of the project

- To find the best algorithm that improve most the reconstructed spectrum resolution

Reconstructional spectroscopy algorithms

- Correlation method
  - $x = \boldsymbol{B}^{\boldsymbol{T}} \cdot A$
- Matrix method
  - $\boldsymbol{B} \cdot x = A$
  - $x = \boldsymbol{B}^{-1} \cdot A$
  - $x = \boldsymbol{B}^{*}(\boldsymbol{B}\boldsymbol{B}^{*})^{-1} \cdot A$
- Nonlinear method
  - $\min_{x, x>0} \|Bx - A\|_2 + q\|x\|_2 + r\|\delta x\|_2$
- Neural network +Nonlinear
  - Autoencoder + nonlinear optimizer



A: to be tested

B : calibration

$$\begin{bmatrix} B_{00} & \cdots & B_{0S} \\ \vdots & \ddots & \vdots \\ B_{n0} & \cdots & B_{nS} \end{bmatrix} X = \begin{bmatrix} A_0 \\ \vdots \\ A_n \end{bmatrix}$$

Image x

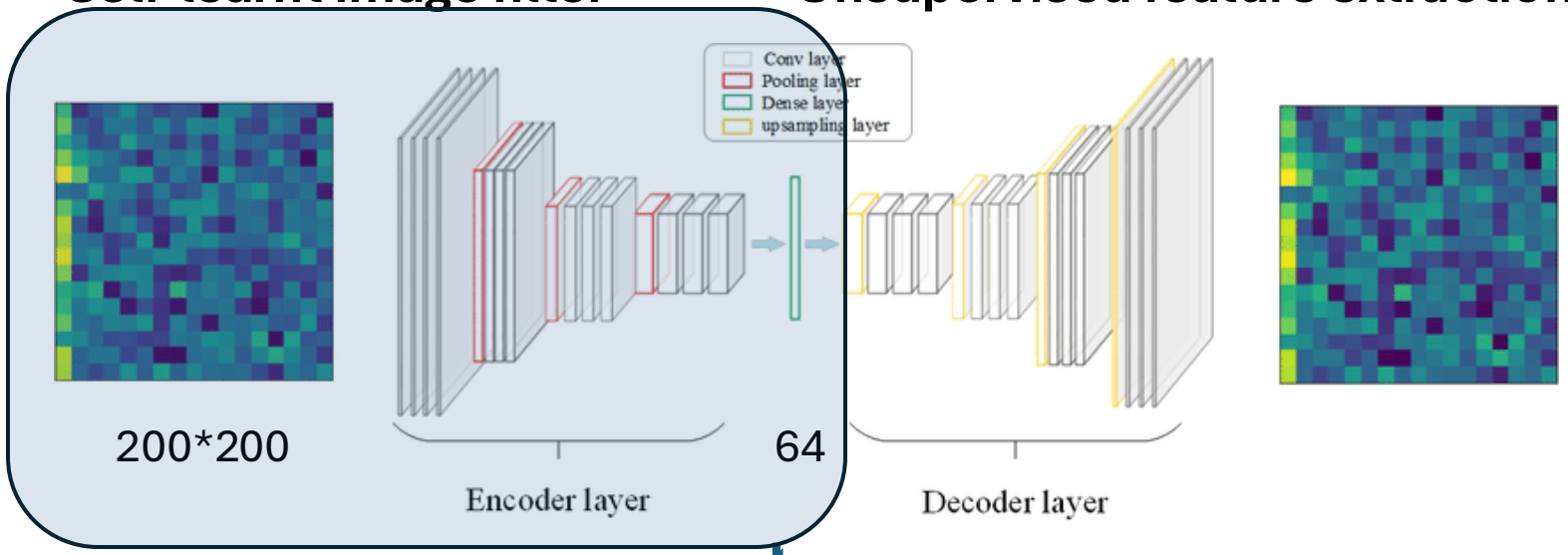
Calibration image pixel value

Spectrum

Detected image

Convolutional Autoencoder

Self-learnt image filter + **Unsupervised feature extraction** = **Efficient representation**

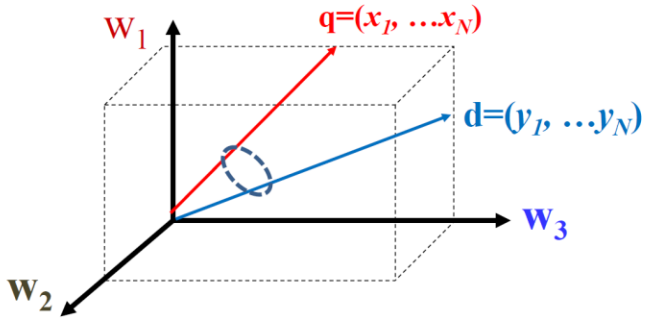


$$\min_{x,x>0} (Ax-B)+q\|x\|_2 + r\|\delta x\|_2$$

Compression ratio ~ 625

Similarity Instantiation: Dot Product

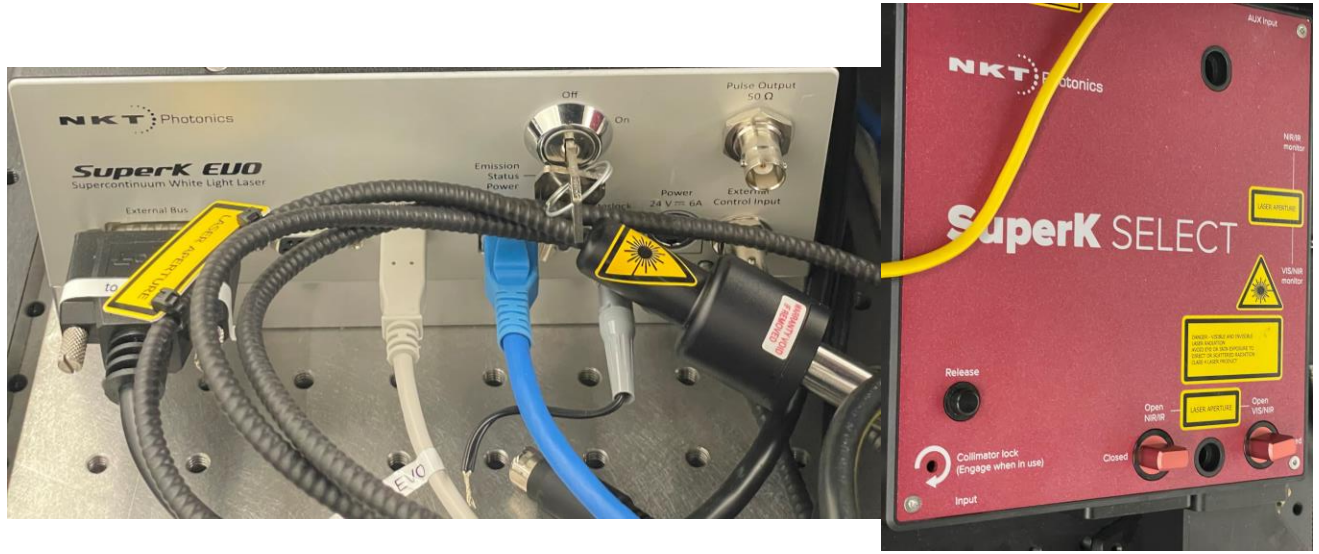
$$Sim(q,d)=q.d= x_1 y_1 + \ldots + x_N y_N = \sum_{i=1}^{N} x_i y_i$$

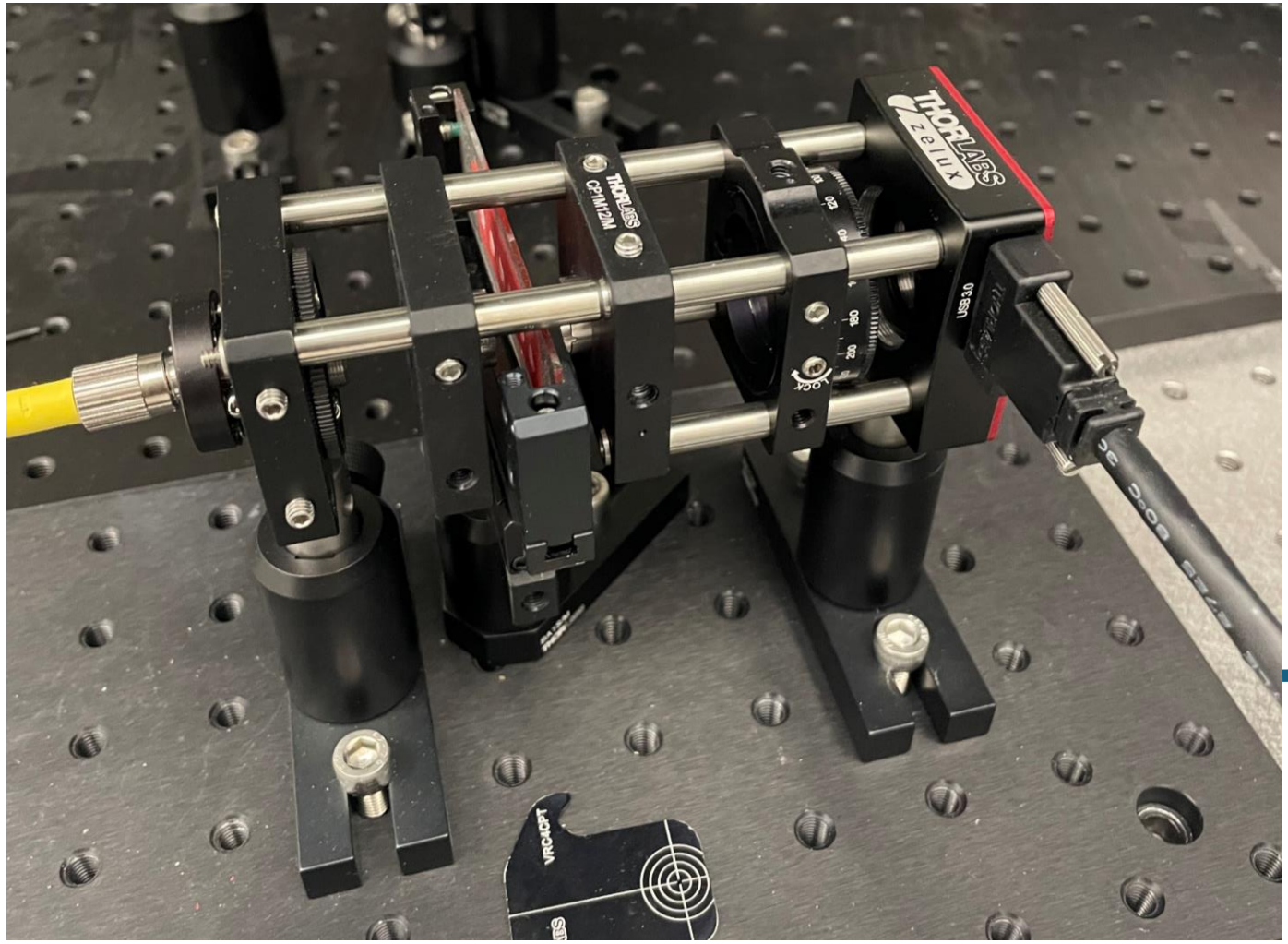


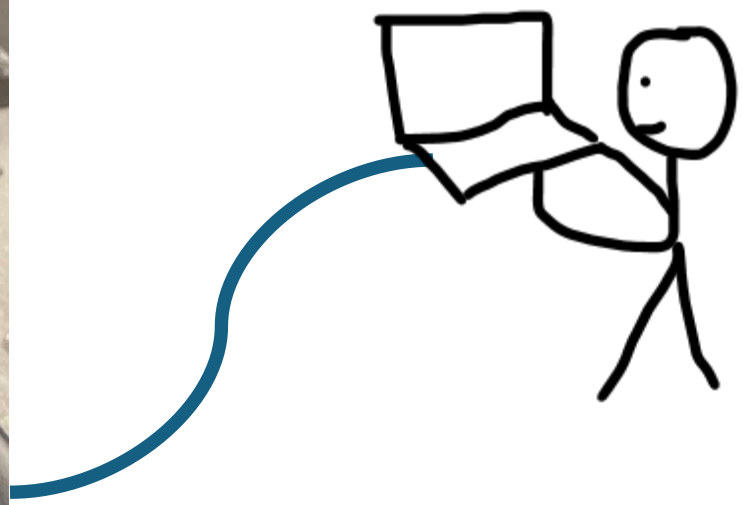
Experiment setup

- **CW laser**
  - **Wavelength range**
    - **765nm – 780nm**
  - **Minimum step**
    - **0.005nm**

- **Supercontinuum source + SuperK SELECT**
  - **540nm – 700nm**
  - **1nm (0.5nm-1.8nm)**




THORLABS
zelux
USB 3.0
THORLABS
CP1WT2M
VRC4CPT



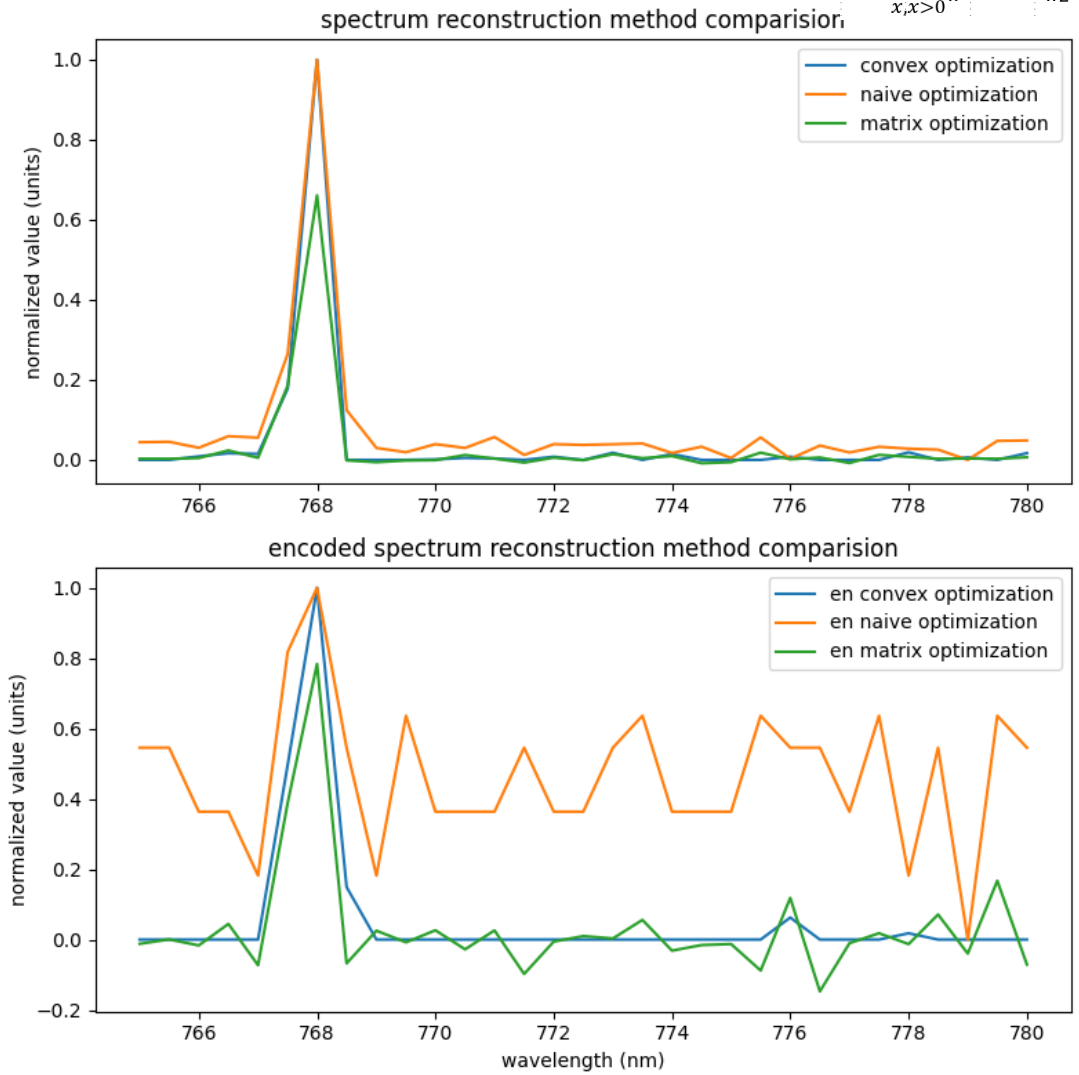
CW: Algorithm comparison

Correlation method
- $x = \boldsymbol{B}^{\boldsymbol{T}} \cdot A$

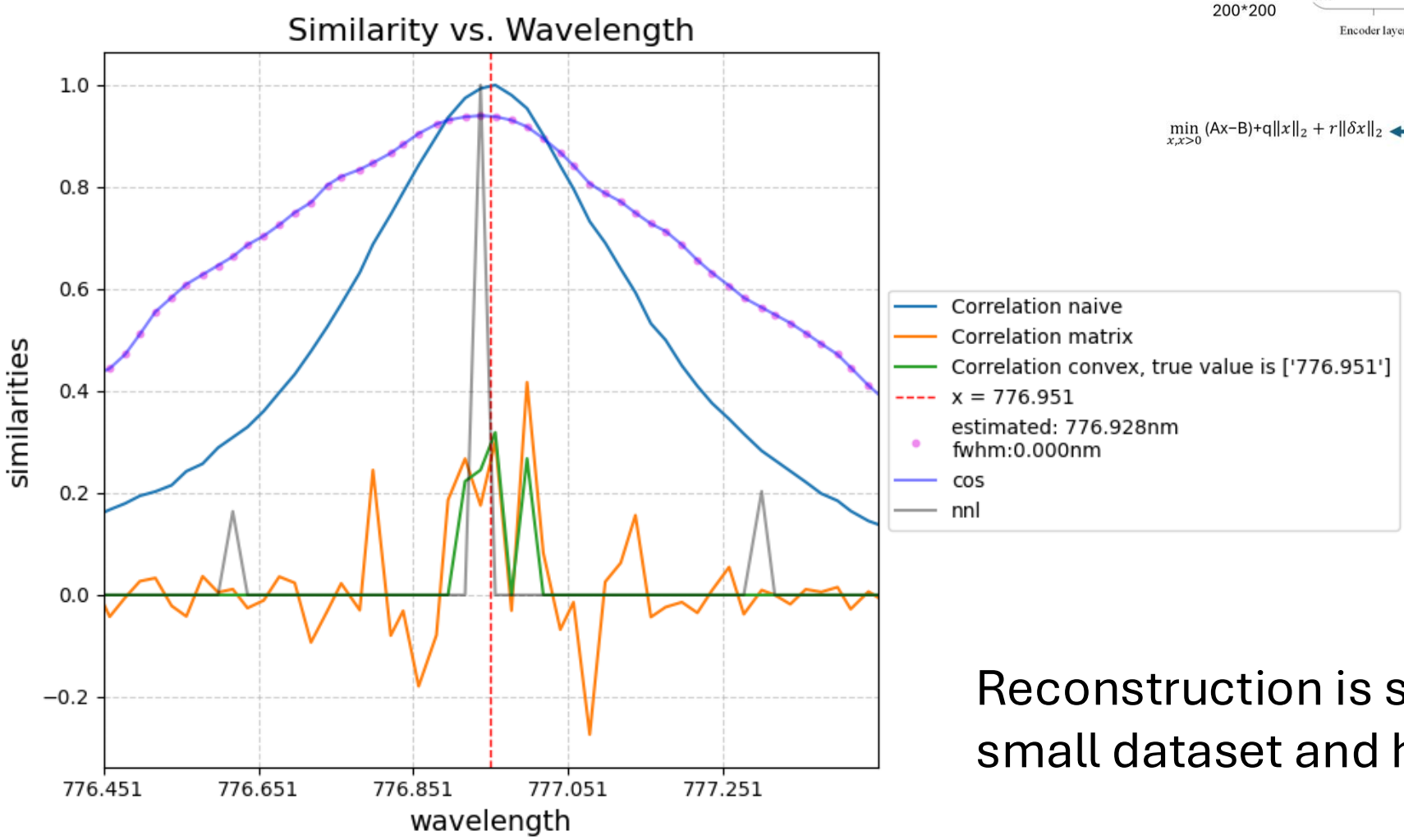
Matrix method
- $\boldsymbol{B} \cdot x = A$
- $x = \boldsymbol{B}^{-1} \cdot A$
- $x = \boldsymbol{B}^{*}(\boldsymbol{B}\boldsymbol{B}^{*})^{-1} \cdot A$

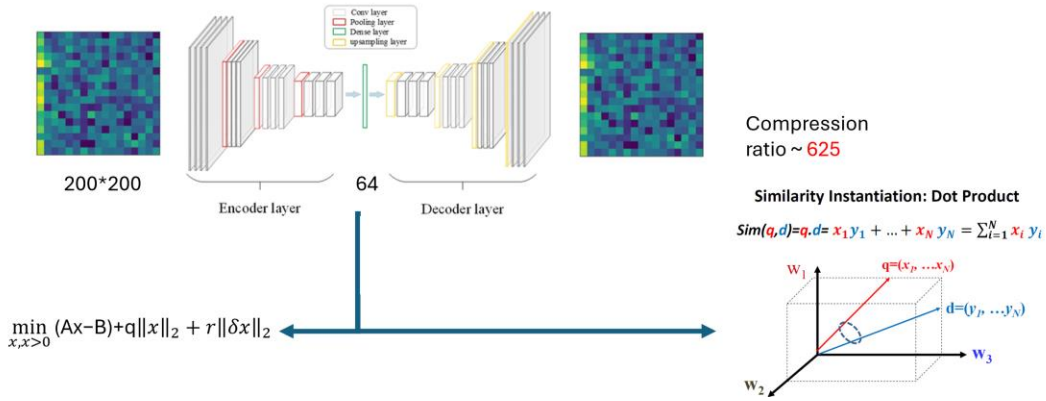
Nonlinear method
- $\min_{x, x>0} \|Bx - A\|_2 + q\|x\|_2 + r\|\delta x\|_2$

- Convex optimization is always the best method.



CW: Algorithm comparison





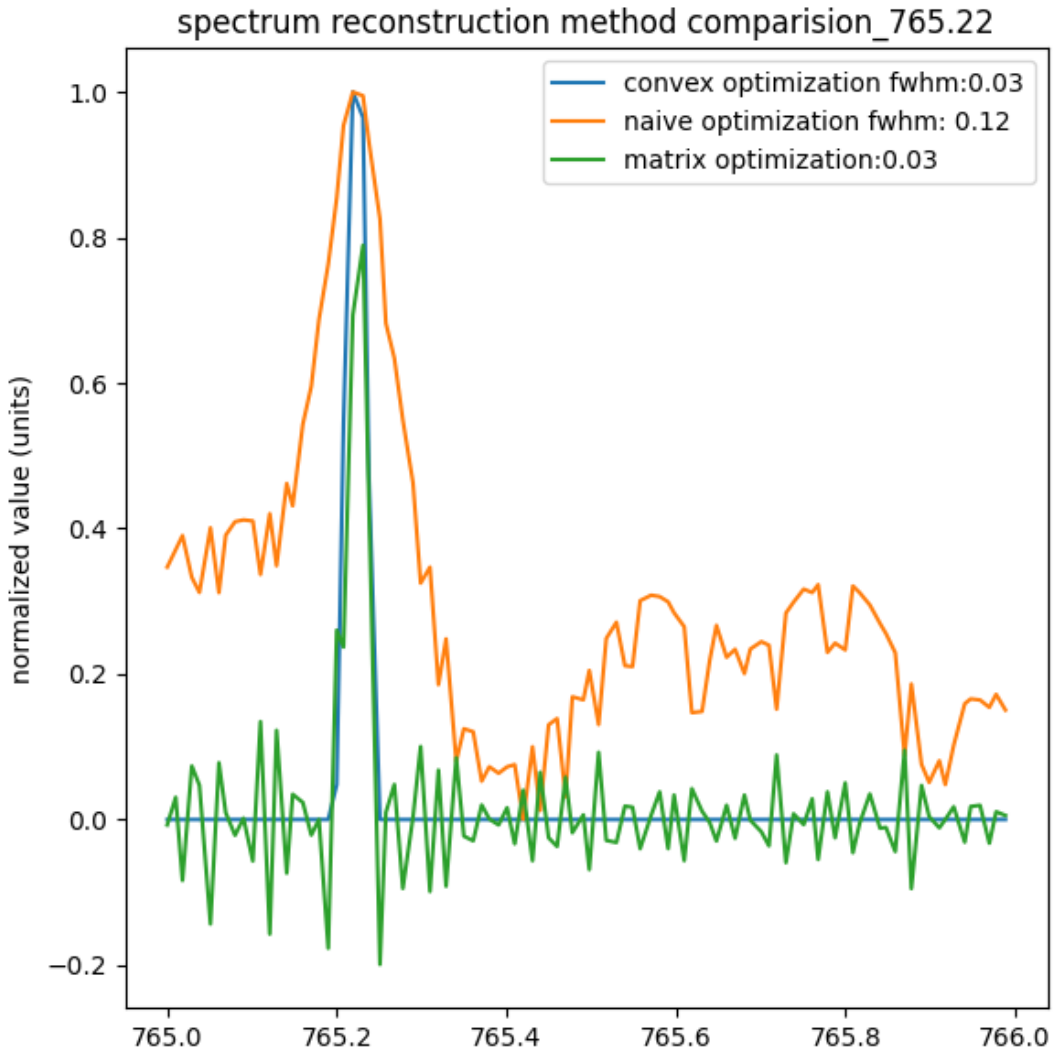
#data= 100

Compression
ratio ~ 625

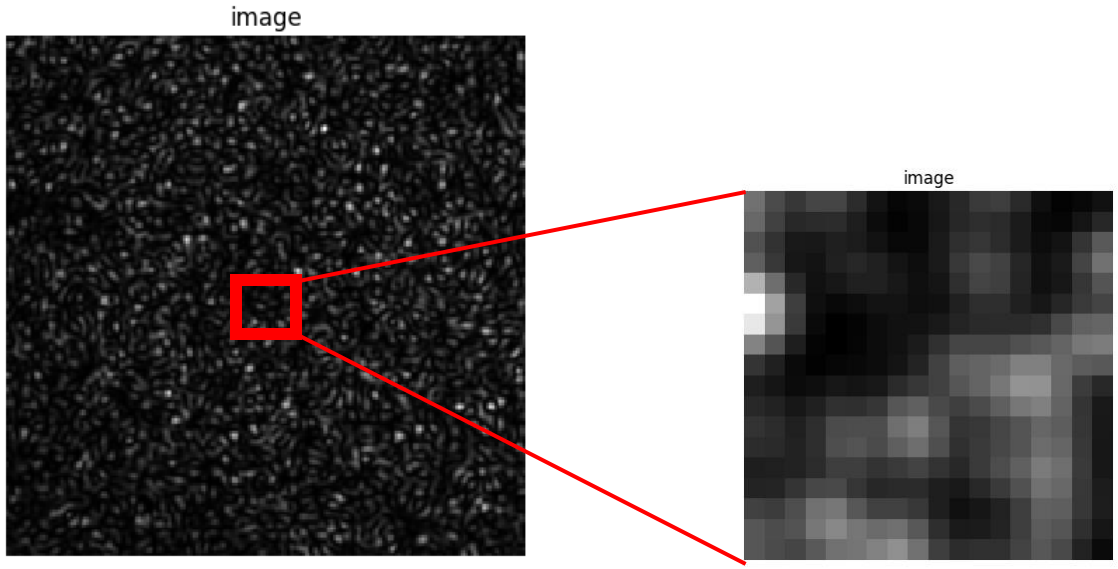
Reconstruction is still available with small dataset and high compression ratio

CW: How good is the spectrometer?

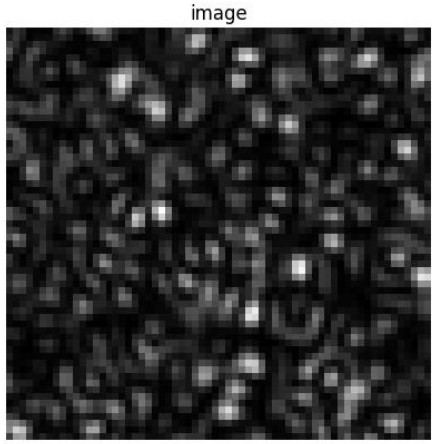
- Resolution
  - Full wavelength at half maximum (FWHM)
  - **0.03nm** for 0.01nm calibration separation



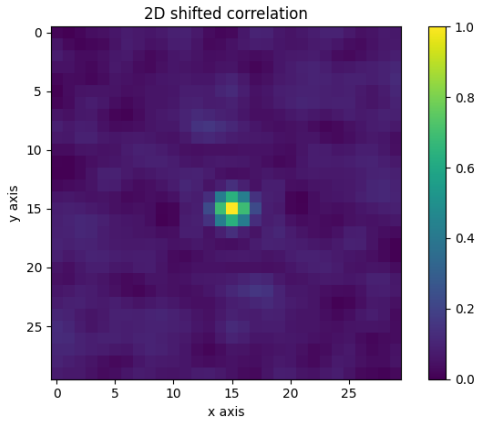
CW: How compact can it be?



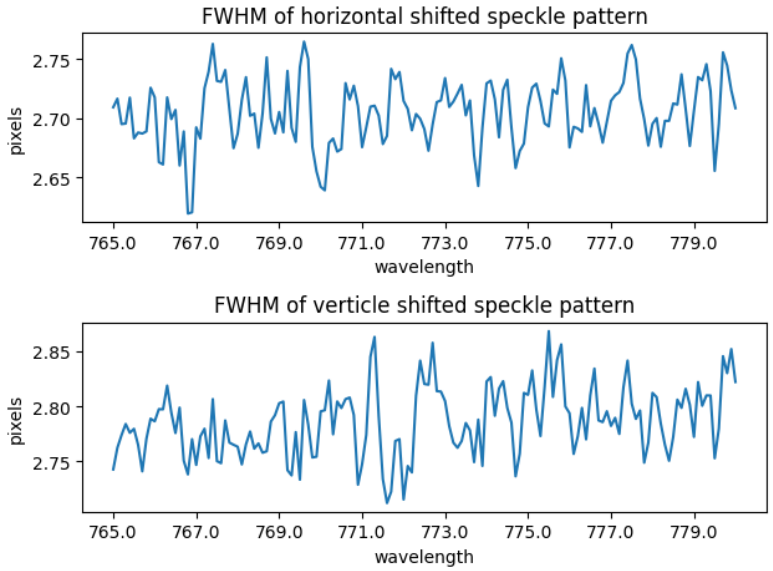
CW: how many pixels does one speckle have?



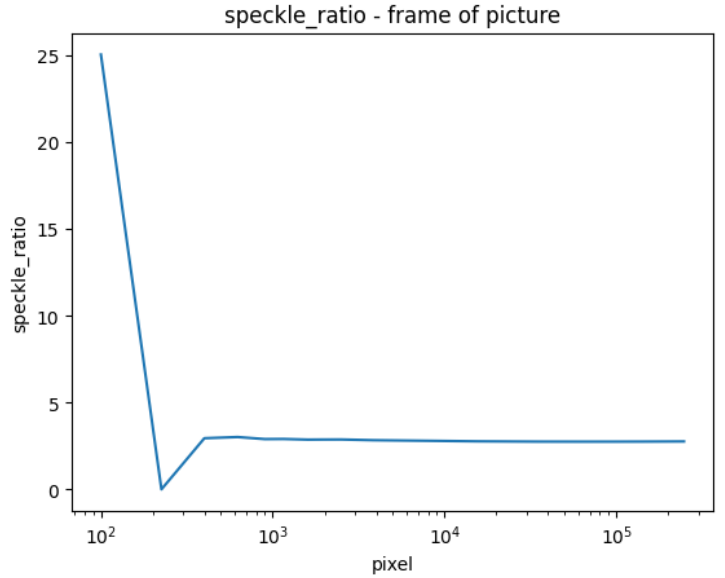
Self-correlation of shifted image



Scan through wavelength



Take average and scan through different size of image

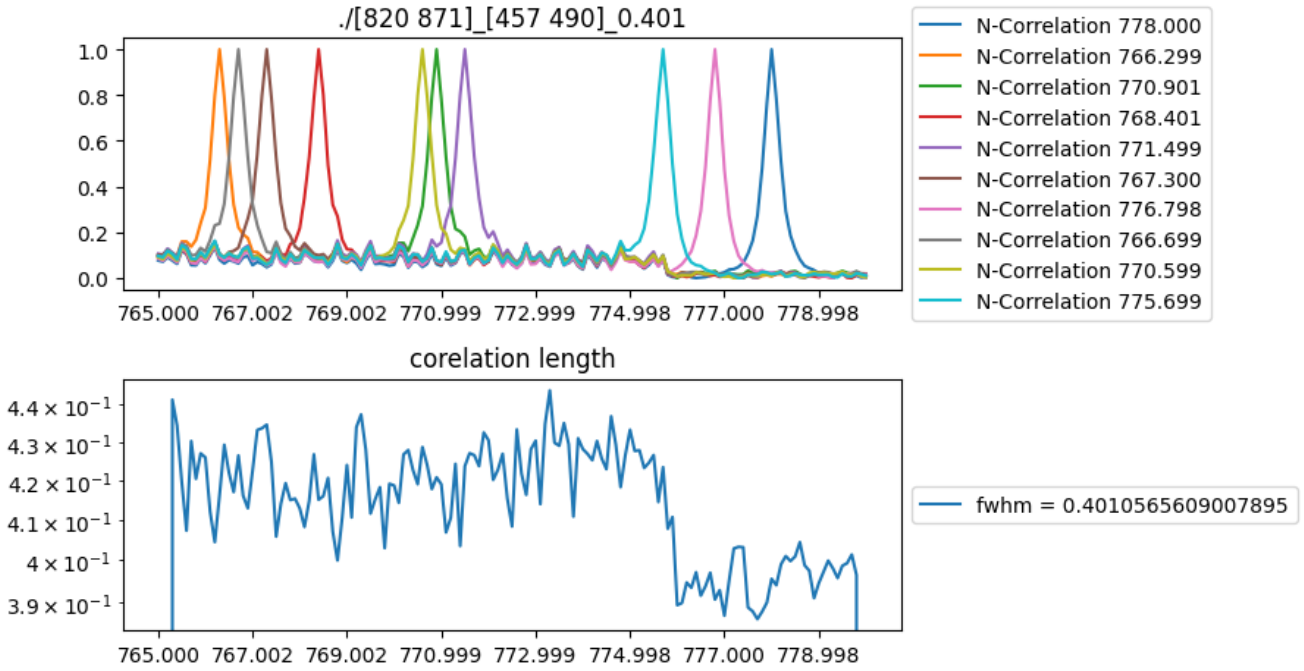


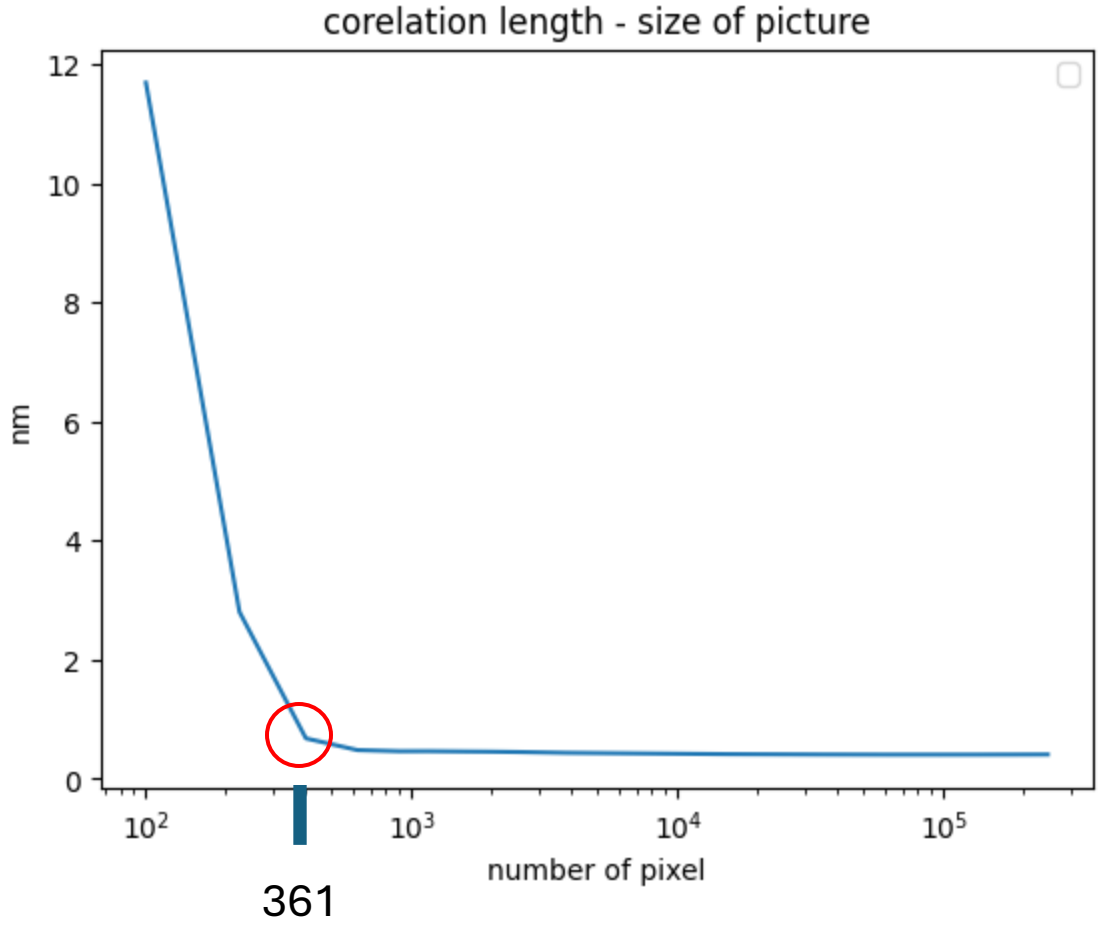
Self-correlation of shifted image

- Speckles take up 8 pixels

CW: How many speckles do I need?

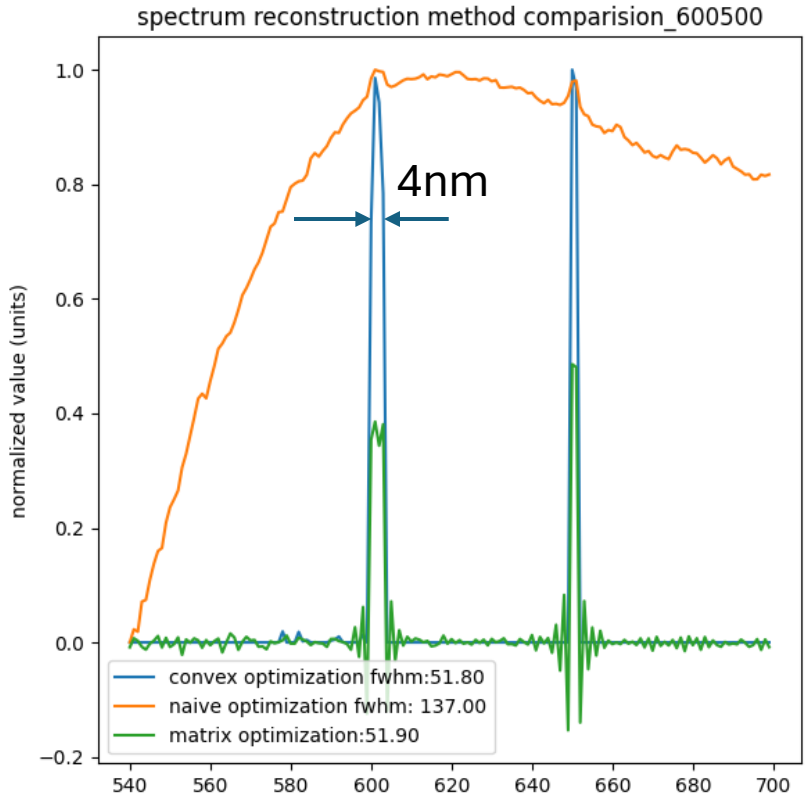
- 0.01 nm calibration dataset gives 0.4 nm resolution (correlation method)
- Correlation length increase at speckle num = 10



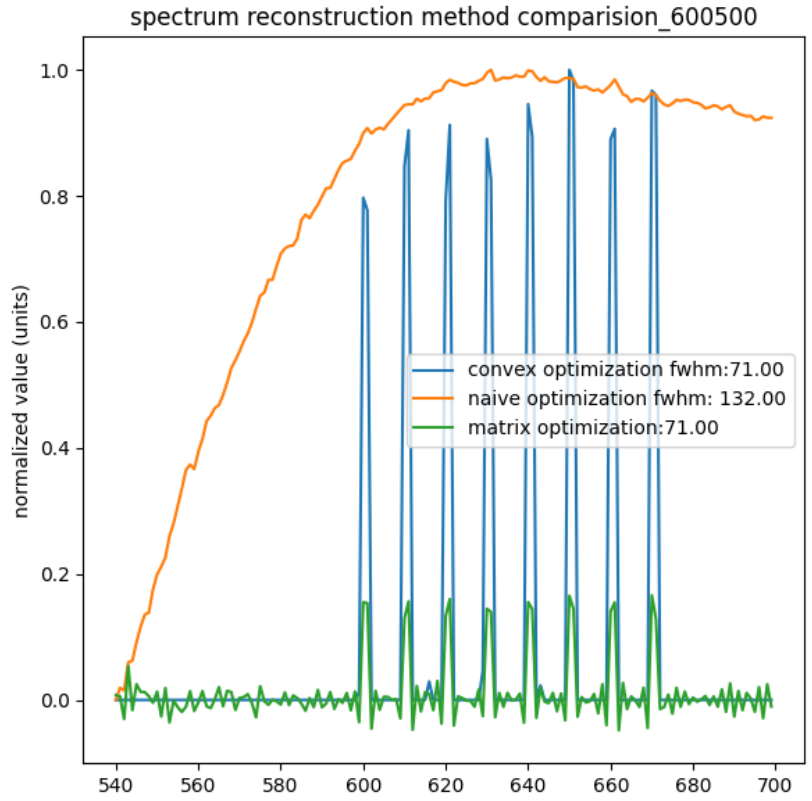


This means we only need 45 speckles, 361 pixels.

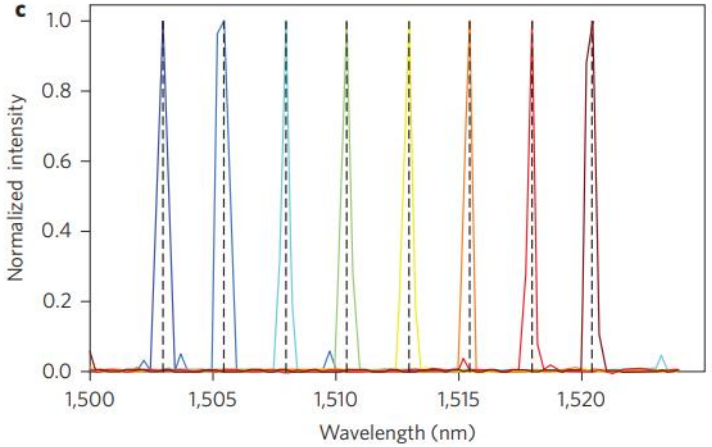
Sup: Spectrum of 8 different wavelength



Calibration from 540nm-700nm with 1nm resolution using super-con source



Test light contains 600.5,610.5,620.5 …670.5nm wavelength read by spectrometer

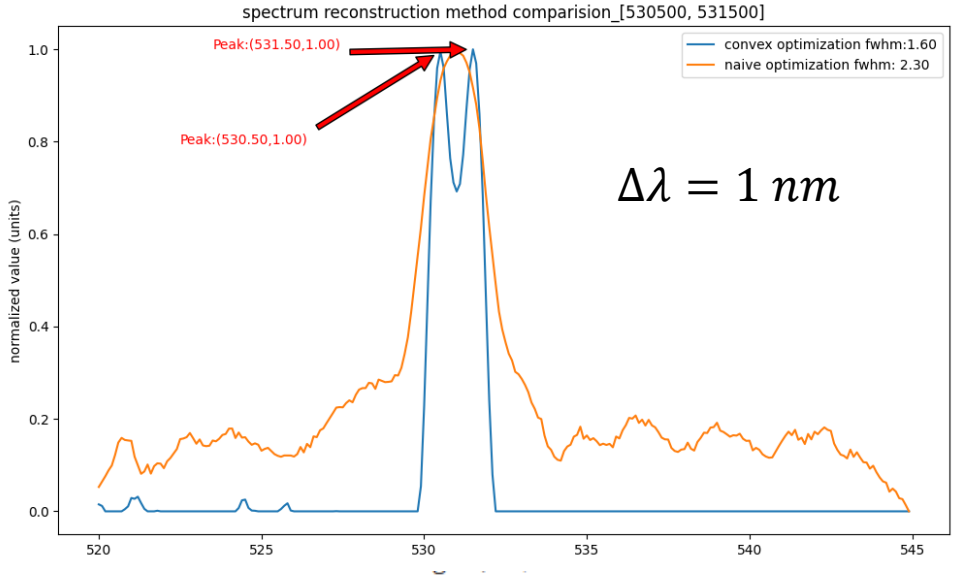


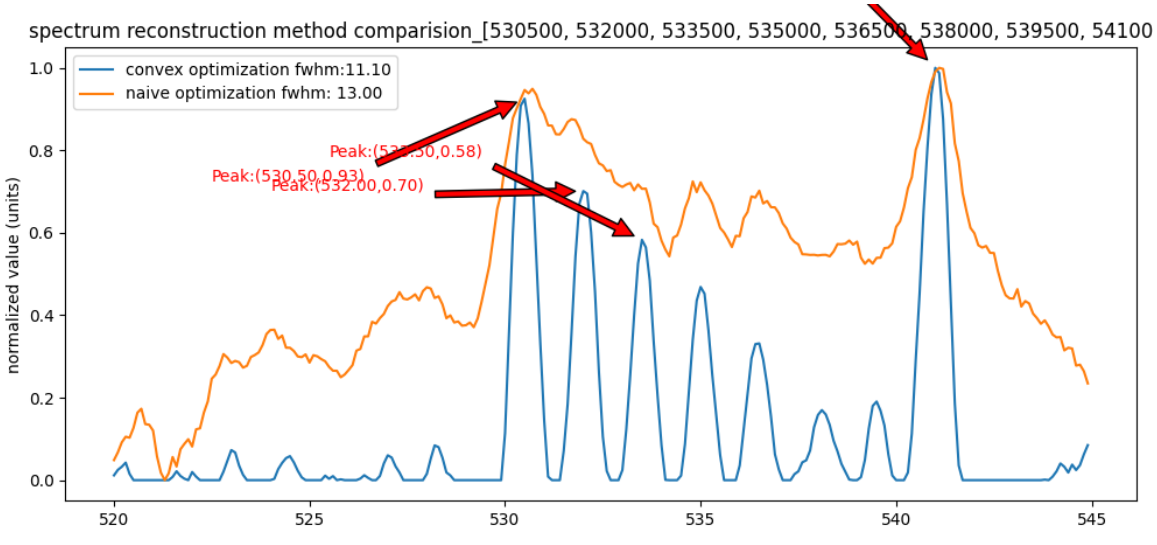
- Much larger bandwidth
- Comparable resolution, given better calibration

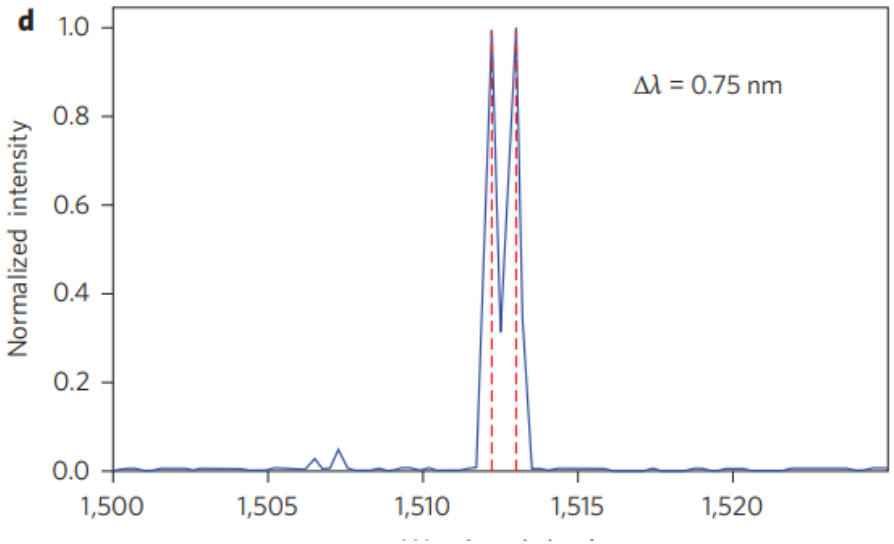
main idea: we can do better than the literature

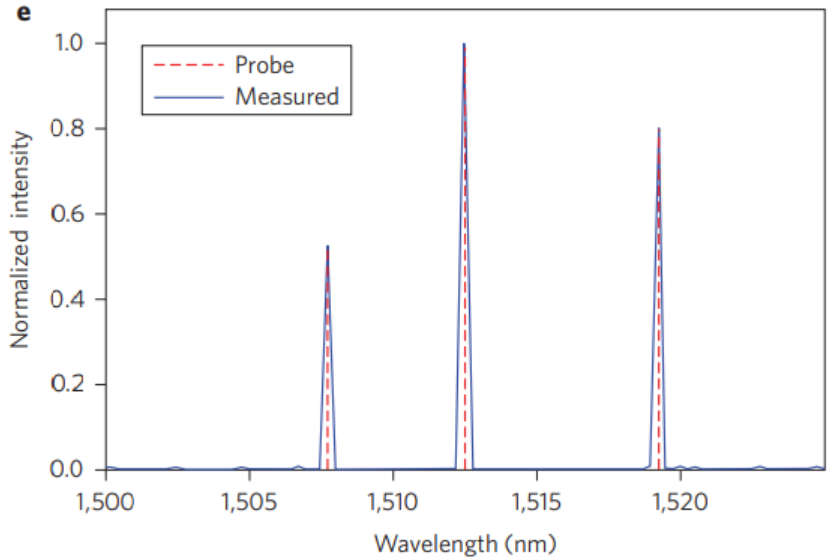
Redding, B., Liew, S., Sarma, R. *et al.* Compact spectrometer based on a disordered photonic chip. *Nature Photon* **7**, 746–751 (2013). https://doi.org/10.1038/nphoton.2013.190

Sup: Spectrum of 8 different wavelength



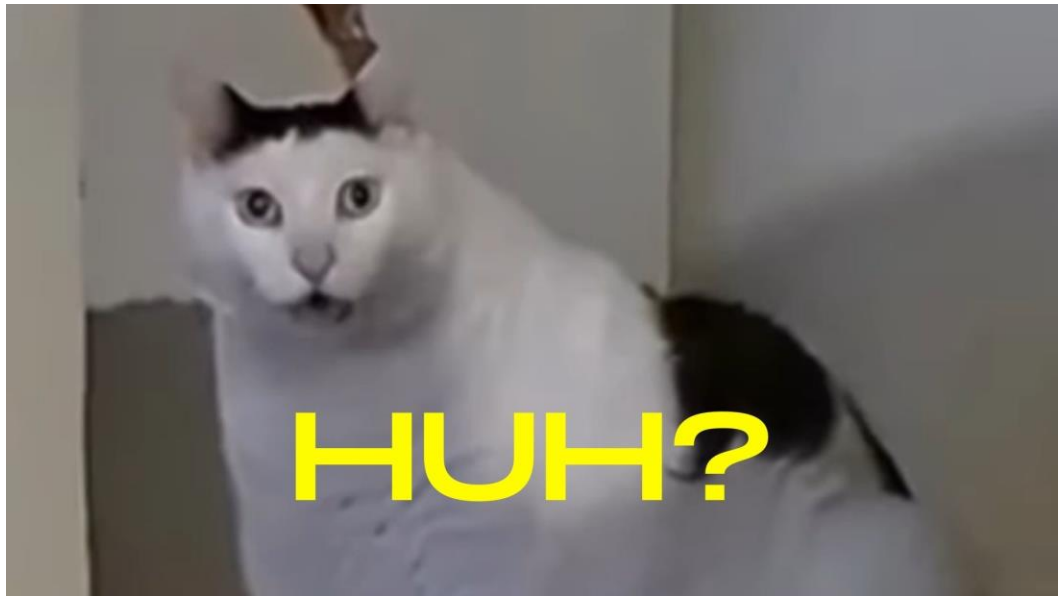


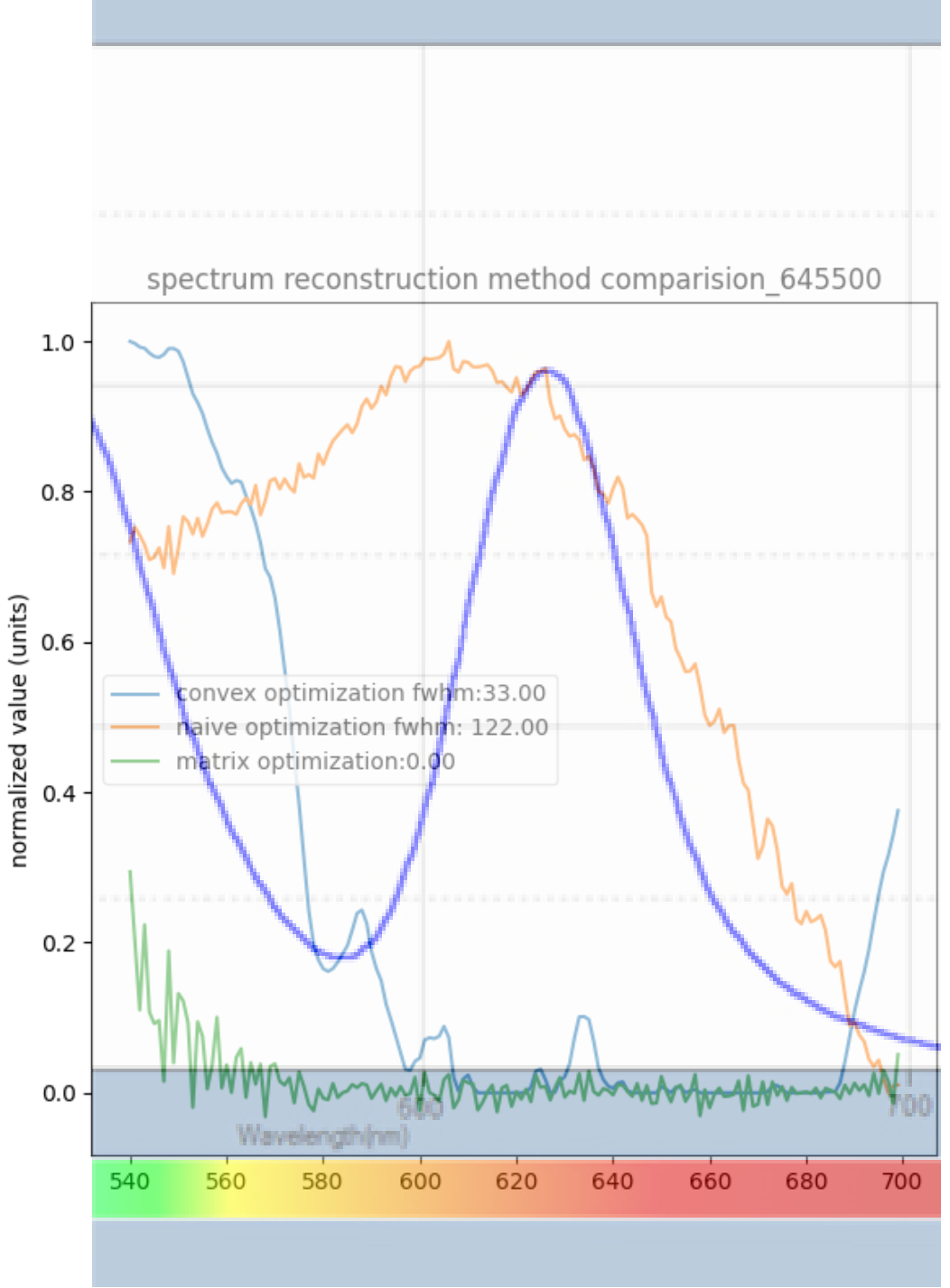




Sup: Let do something fun

- Measure flashlight of iphone 12 pro
  - Extremely Low coherence source
  - Bad coupling of flash light on fiber
  - Amplitude fluctuation



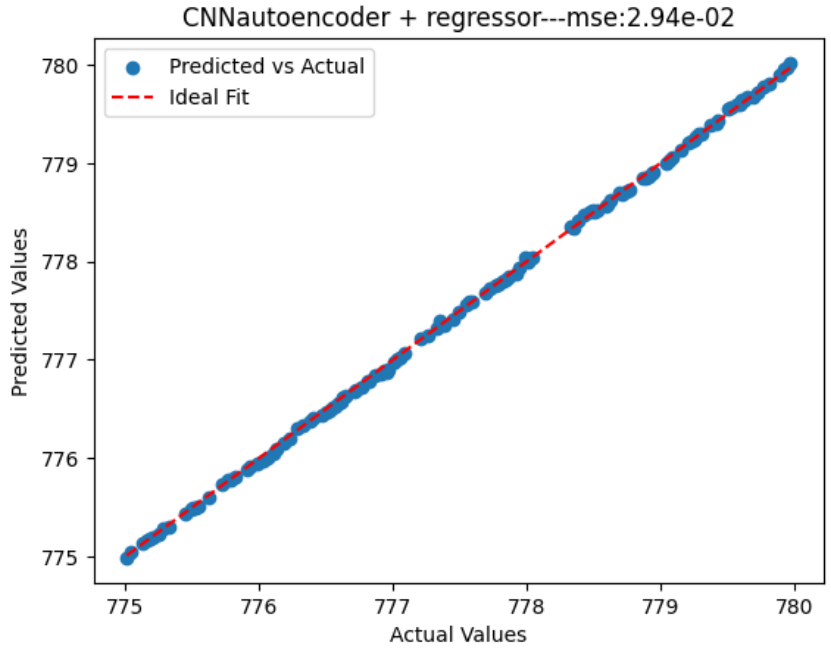


Conclusion

- Nonlinear optimization algorithm exceeds the performance of other algorithm
- Convolutional autoencoder works on compressing data
- Resolution can reach 0.03nm

Outlooks

- Improve resolution
  - Using deep learning to do classification
- Application
  - Measurement of glucose value of blood
    - Regression on the detected parameter
  - Polarization-wavelength sensor
    - Increase data set
  - Second harmonic
- Better define the resolution
  - Signal-to-noise ratio



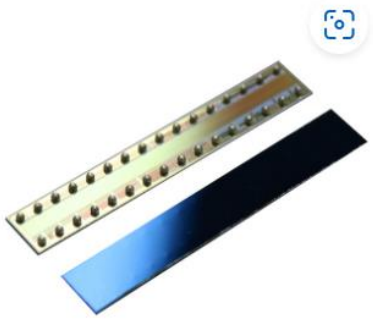
For NN we have those network implemented:

- Metric learning
- **Convolutional variational autoencoder + MLP**
- ResNet
- Transfer learning

Outlooks

Prototyping spectrometer

436 €



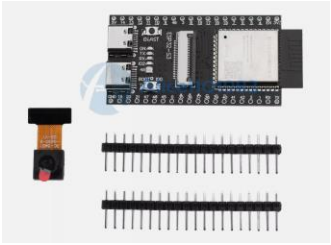
34 €

2 €

80 €

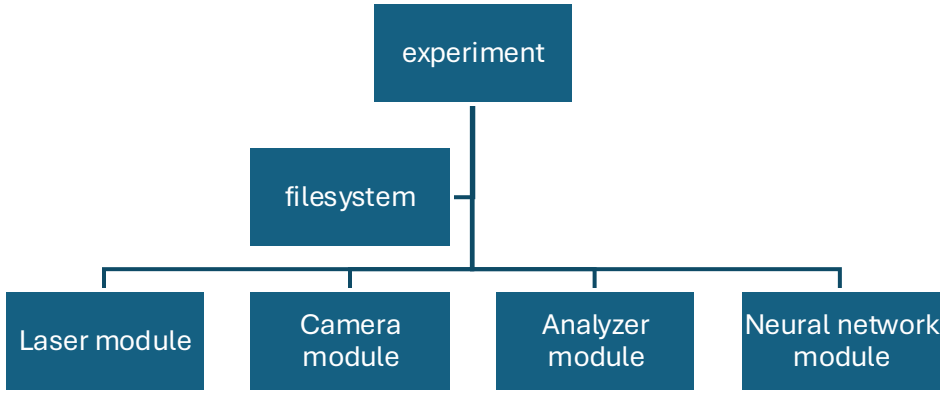
1 €

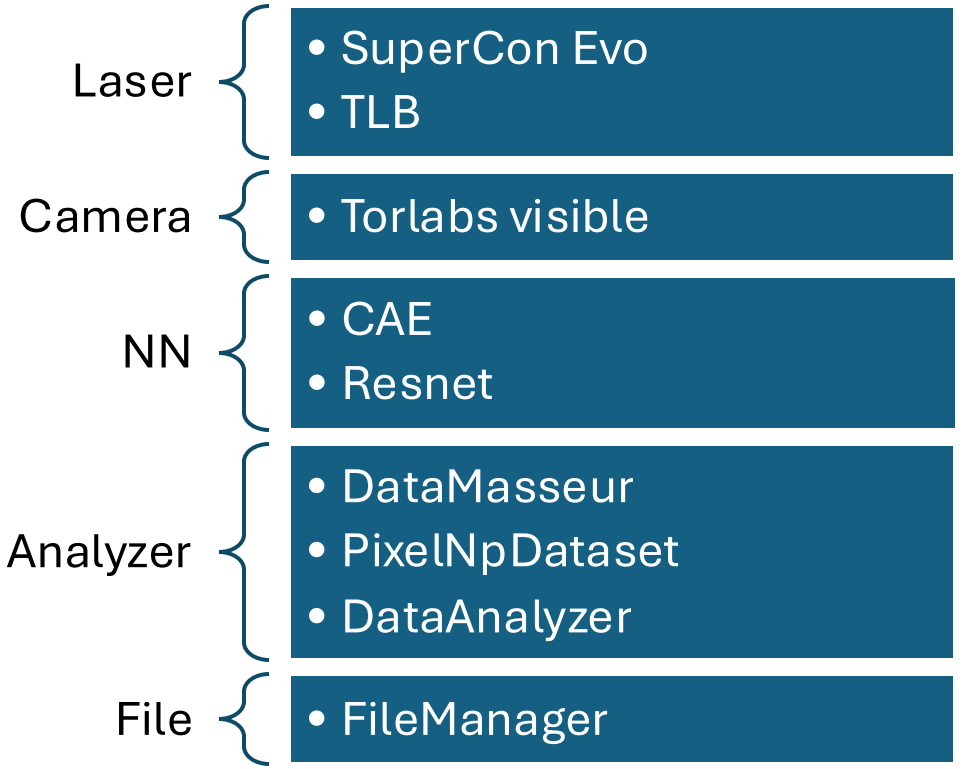
540 €



20 €

Outlooks

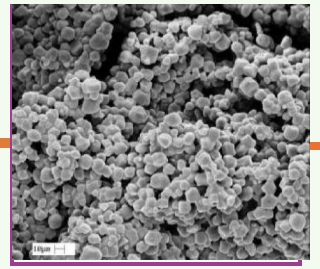




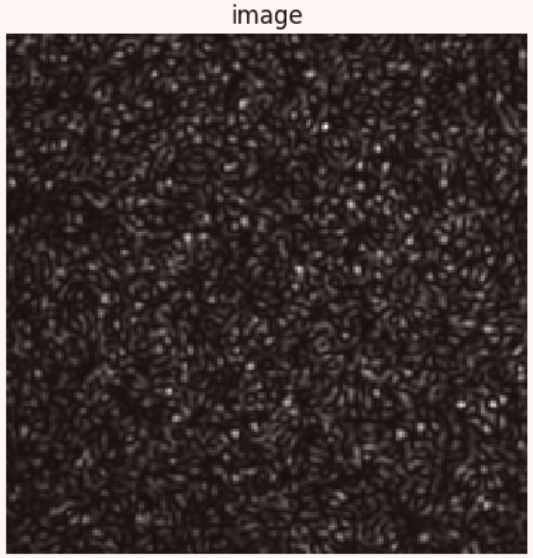
Improve the python program

- GUI

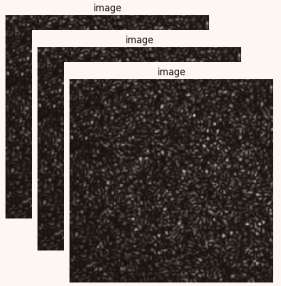
Reconstructional spectrometer



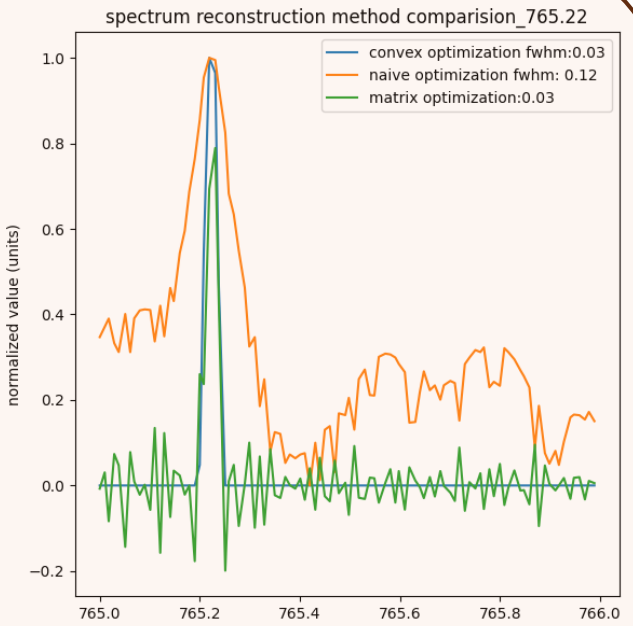
$LiNbO_2$



Speckle pattern



Reconstruction



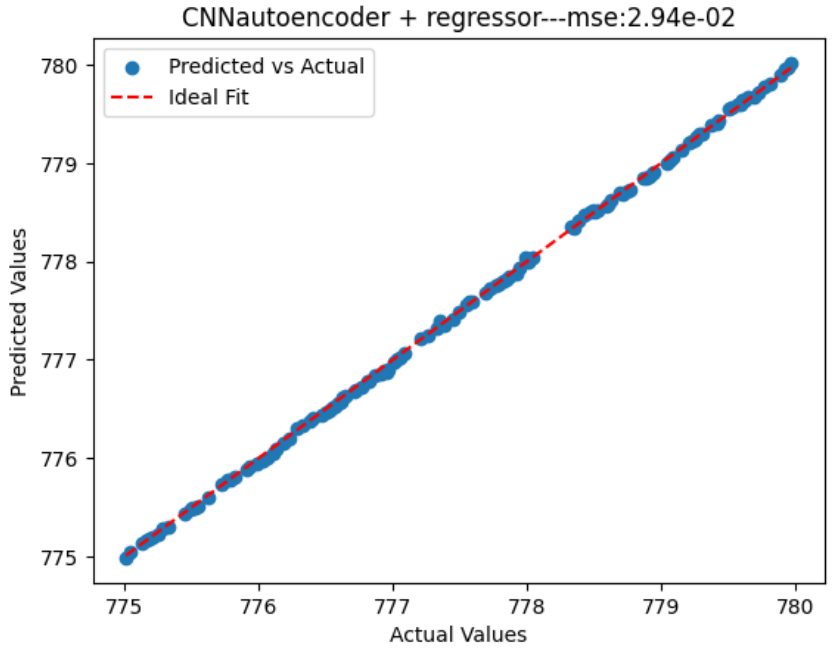
Spectrum

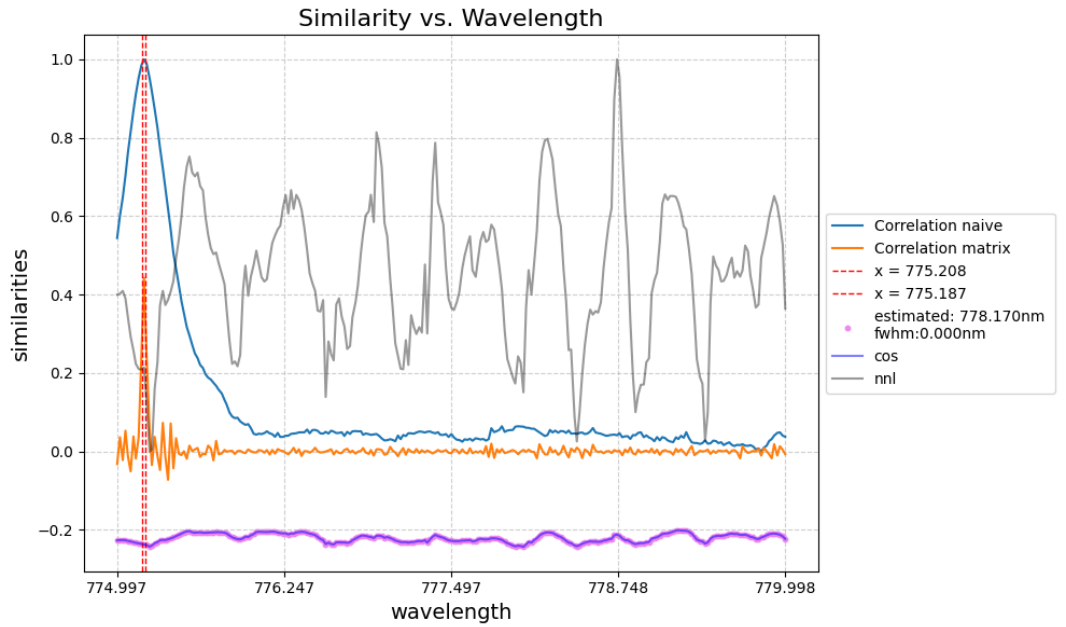
Supplementary

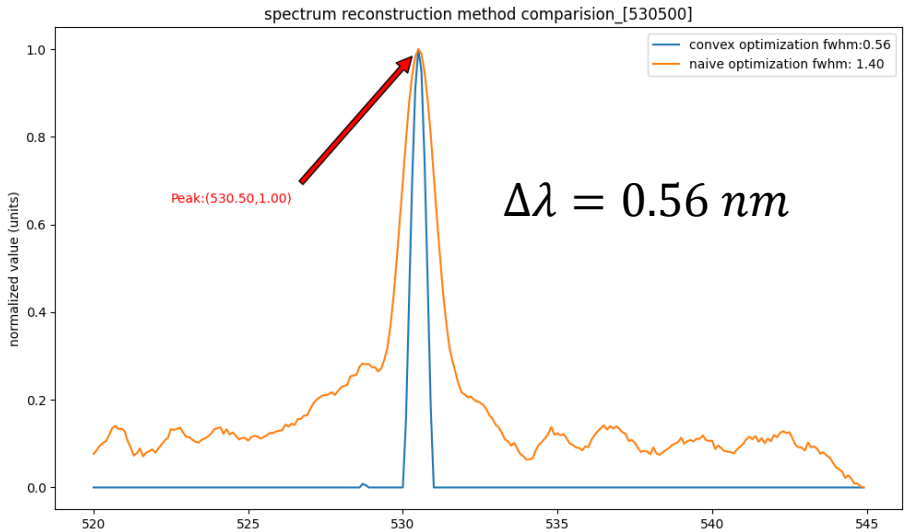
For NN we have those network implemented:

- Metric learning
- **Convolutional variational autoencoder**
- ResNet
- Transfer learning

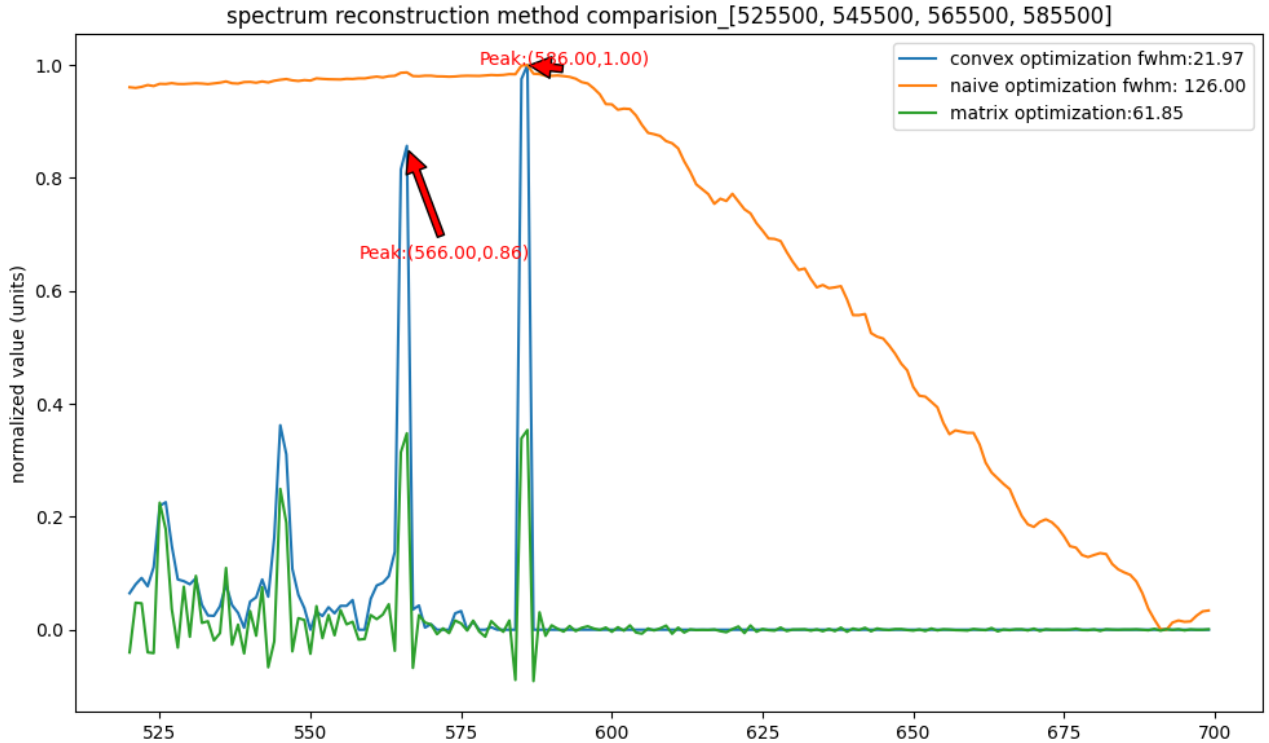
```
ConvAutoencoder(
  (encoder): Convencoder(
    (enc1): Conv2d(1, 16, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (enc2): Conv2d(16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (enc3): Conv2d(16, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (enc4): Conv2d(32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (enc5): Conv2d(32, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (lin1): Linear(in_features=288, out_features=64, bias=True)
    (dropout): Dropout(p=0.2, inplace=False)
  )
  (decoder): Convdecoder(
    (lin1): Linear(in_features=64, out_features=288, bias=True)
    (dec1): ConvTranspose2d(32, 32, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (dec2): Conv2d(32, 32, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (dec3): ConvTranspose2d(32, 16, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1))
    (dropout): Dropout(p=0.2, inplace=False)
    (dec4): Conv2d(16, 16, kernel_size=(3, 3), stride=(1, 1), padding=(1, 1))
    (dec5): ConvTranspose2d(16, 1, kernel_size=(3, 3), stride=(2, 2), padding=(1, 1), output_padding=(1, 1))
  )
)
```







$$\Delta\lambda = 0.56\ nm$$



Possible direction of neural network approach

- Information theory based compression algorithm
- Possibility to generalization:
  - Multi-label input
    - Temperature, material type,position..
    - Compress experiment setup into 64 dimensions space
  - Increase dataset
    - 10000 pictures
    - Variational autoencoder
- Note the position of calibrated point on sample

Experiment class overview

```python
class Super_con(BaseLaserController):
    def __init__(self,name,RF_PORT='COM3',SE
    def set_module_address_select(self,new_a
    def connect_laser(self):…
    def open_laser(self):…
    def close_laser(self):…
    def close_connection(self):…
    def change_laser_power(self,value):…
    def change_laser_mode(self,setup_num):…
    def enable_interlock(self):…
    def turn_on_RF(self):…
    def turn_off_RF(self):…
    def turn_laser_channel(self,wavelength,a
    def read_status(self):…
    @staticmethod
    def connect_port():…
```

```python
class Calibration(BaseExperiment):
    def __init__(self,name,file_module,laser_module,camera_module,analyzer):
        super().__init__(name,file_module)
        self.laser = laser_module
        self.camera = camera_module
        self.analyzer = analyzer
    def initialization_laser(self):
        self.laser.connect_laser()
        self.laser.open_laser()
        self.laser.change_laser_mode(0)
        self.laser.change_laser_power(900) # 50%
        self.laser.enable_interlock()
    def initialization_RF(self):
        self.laser.turn_on_RF()
    def clean_up(self):
        self.laser.turn_off_RF()
        self.laser.close_laser()
        self.laser.close_connection()
    def calibrate(self,range_low,range_high,step):
        self.initialization_laser()
        self.initialization_RF()      You, 4 days ago • python Experiment system added. IR Camera rewra…
        channel =0
        self.analyzer.data = {}
        cropped = {}
```