Title	【課題研究報告書】Development of a Phase Retrieval Method in Coherent X-ray Diffraction Imaging for Nanoscale Structure Visualization
Author(s)	唐, 朝宇
Citation	
Issue Date	2024-03
Type	Thesis or Dissertation
Text version	none
URL	http://hdl.handle.net/10119/18910
Rights	
Description	Supervisor: DAM, Hieu Chi, 先端科学技術研究科, 修士(情報科学)

Master's Research Project Report

Development of a Phase Retrieval Method in Coherent X-ray Diffraction Imaging for Nanoscale Structure Visualization

TANG CHAOYU

Supervisor DAM, Hieu Chi

Graduate School of Advanced Science and Technology
Japan Advanced Institute of Science and Technology
(Information Science)

March, 2024

Abstract

The traditional optical microscopy system is a common tool for observing microscopic structures. However, in recent times, more advanced phase imaging techniques have emerged, such as ptychography and Coherent Diffraction Imaging (CDI). These techniques discard the lenses required by traditional optical microscopy systems and instead utilize computational methods for imaging microscopic objects. Therefore, they can be categorized into the field of computational imaging.

Using coherent X-rays to shot the specimen through aperture, it will generate Diffraction images on a detector receiver. X-rays possess excellent penetration capabilities, allowing them to effectively pass through specimen. The coherence of the X-rays ensures that the diffraction image contains the Fourier transform result of the specimen. Based on this principle, image reconstruction requires phase retrieval. In this paper, we employ a code-simulated optical diffraction system and data to practice phase imaging. We utilize traditional iterative methods for phase retrieval to reconstruct images. Due to the high cost of generating coherent light in real systems, this paper also explores the optimization of a classic iterative algorithm using data science methods under conditions of limited data.

Acknowledgement

Before coming to Japan, I used to just think about having fun. However, after arriving here with the goal of finding work and staying in Japan, it was probably the first time in my life that I dedicated a long period of intense effort towards something. I started learning Japanese from hiragana, and continued tirelessly day and night for a year. Eventually, I passed the Japanese company interview and found a decent job, but unfortunately, I encountered mismatch issues with the professor in my previous research lab. This led to conflicts that even affected my graduation, the job I had worked so hard to find was also at risk of being ruined, and despite reaching out to several other professors, no one was willing to take me in.

Fortunately, at the 1 day before the last deadline, Professor DAM kindly responded to my plea for help in my most difficult time. Upon joining the new research lab, everyone treated me well. Professor DAM assigned me an advanced research project related to physics area, which was one of the areas of greatest interest to me. Since coming here, I have learned a lot of mathematics and physics. While working on this new project, doctoral seniors Adam-san and Sinh-san in the lab provided me with detailed guidance and assistance, helping me progress rapidly.

Contents

List of Figures

Chapter 1
Introduction

1.1 overview

The well-known Fourier transform is a mathematical tool used to convert time-domain information into frequency domain. Its principle involves fitting actual time-domain data using a large number of different frequency triangular periodic functions. Despite being an abstract mathematical tool, it paradoxically corresponds to a practical mapping in reality. In optics, when light waves undergo diffraction through a small aperture, the detector receiving the diffracted light waves will display a distinctive pattern. This pattern represents the two-dimensional Fourier transform of the aperture. If, at this point, we place a small specimen we want to observe into this aperture, the Fourier transform image on the detector screen will simultaneously contain information about the specimen and the aperture. By performing the inverse Fourier transform on this Fourier transform image, we obtain the original image of the specimen, including the aperture. This enables effective observation of nano-scale specimen. Based on this principle, various microscopy techniques have been invented.

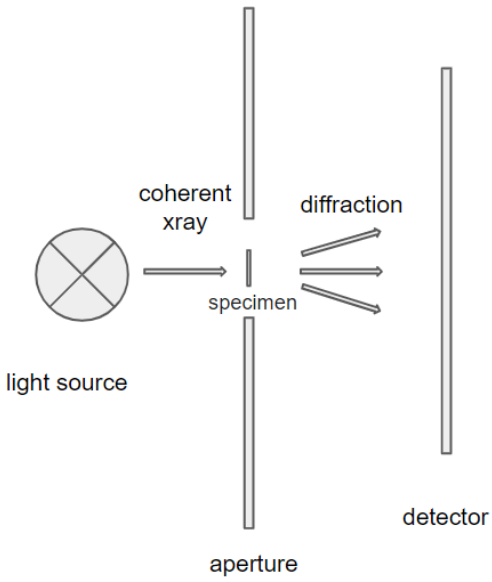


Figure 1.1: A schematic diagram of diffraction generation

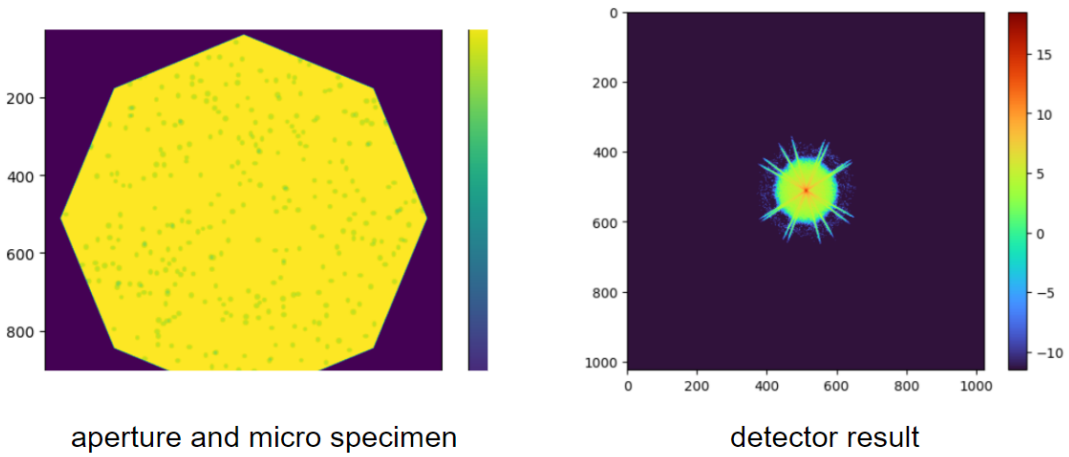


Figure 1.2: Left, aperture and molecular specimen sample,
Right, the diffraction result of the specimen and aperture

Now the question leads to a new dimension. If we possess the Fourier transform image of the original object, restoring its true appearance requires performing the Fourier inverse transform. The Fourier transform itself provides two types of outputs: amplitude and phase. Amplitude corresponds to the magnitudes of various frequencies in the Fourier transform, while phase corresponds to the initial displacement of the periodic functions at time 0 for each frequency in the Fourier transform.

When capturing the image on the detector, we only obtain the amplitude information. During the process of Fourier inverse transform, restoring the shape of the image is significantly influenced by the phase. Therefore, in the absence of phase information, reconstructing the original image becomes a challenging problem. At this point, we need to generate a phase to assist in performing the Fourier inverse transform and reconstructing the original image.

Given that the known information includes the original shape of the aperture, and the unknown information pertains to the structure of the specimen we want to observe, leveraging the known information about the aperture shape allows us to partially restore the unknown phase information in the correct direction.

Based on this principle, related microscopy techniques have been developed, such as ptychography and Coherent Diffraction Imaging (CDI). Unlike traditional optical microscopy techniques, these methods do not involve direct optical imaging but require computational approaches to reconstruct the original image. The computation in question involves using the Fourier inverse transform mentioned earlier to restore the true image from the diffraction pattern. They all face a common challenge known as phase retrieval.

1.2 Research objective

The objective of this study is to optimize existing phase retrieval methods. Traditional iterative methods for phase retrieval have their drawbacks, such as suboptimal performance, weak noise resistance, and excessive iteration requirements.

Meanwhile, emerging methods that incorporate deep learning for phase retrieval often demand a substantial amount of data and may perform poorly when data is scarce. Since this experiment requires the use of coherent X-ray to generate diffraction data, manufacturing coherent X-rays in reality often comes with high costs, implying that the cost of data acquisition would be exceptionally high. Therefore, the aim of this paper is to find a compromise, combining traditional phase retrieval iterative methods. This approach not only allows for the application of data science strategies to optimize traditional methods but also enables achieving satisfactory results with a reduced amount of data, thereby saving costs.

1.3 Research Outline

This report mainly includes several sections, including background knowledge, related work, experimental setup, experimental results, proposed methods, and their effectiveness.

Chapter 2
Preliminaries

2.1 why the light diffraction can be seen as the Fourier transformation

From the well-known double-slit experiment, we can infer that light exhibits two forms: particle and wave. In this context, we consider light in its wave form, which can be mathematically expressed as:

$$E * e^{2\pi ivt}$$

Where E represents intensity, ν is the frequency of the light wave, and t is time, indicating that light, as a wave, varies in amplitude over time.

Considering light as a wave, according to Huygens' principle, each point on the aperture acts as a point source, essentially being treated as a new light source. Thus, selecting a point dx within the aperture will emit a light wave to the detector point P. Analyzing the phase variation of this light beam as a wave, we focus on the relationship between phase change, wavelength λ, and propagation distance. The distance between dx and point P is denoted as r, resulting in r / λ complete waves, and the corresponding phase change is 2πr / λ. Therefore, the light wave from dx to point P can be expressed as:

$$dE = E_0 e^{2\pi ivt} e^{\frac{2\pi ir}{\lambda}}$$

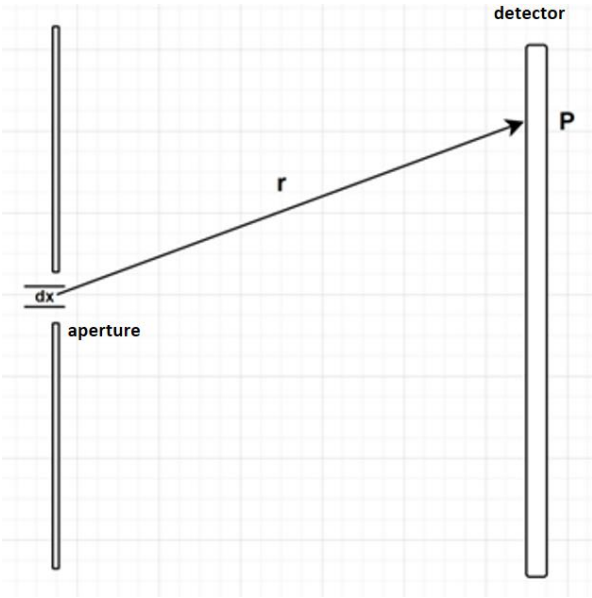


Figure 2.1: A point in aperture as new light source

The total light wave propagating to point P after passing through the entire aperture is the integral of all the light waves emitted by every point on the aperture. The result is expressed as:

$$E = \int_{aperture} E_0 e^{2\pi ivt} e^{\frac{2\pi ir}{\lambda}} \, dx$$

Within this integral, the only variable related to the integration term x is r, while the rest are constant terms. Therefore, we can move the constant terms outside the integral.

$$E = E_0 e^{2\pi ivt} \int_{aperture} e^{\frac{2\pi ir}{\lambda}} \, dx$$

if the distance to the detector is sufficiently large, $r \gg x$, an approximation can be derived as follows:

$$r = r_0 - x sin\theta$$

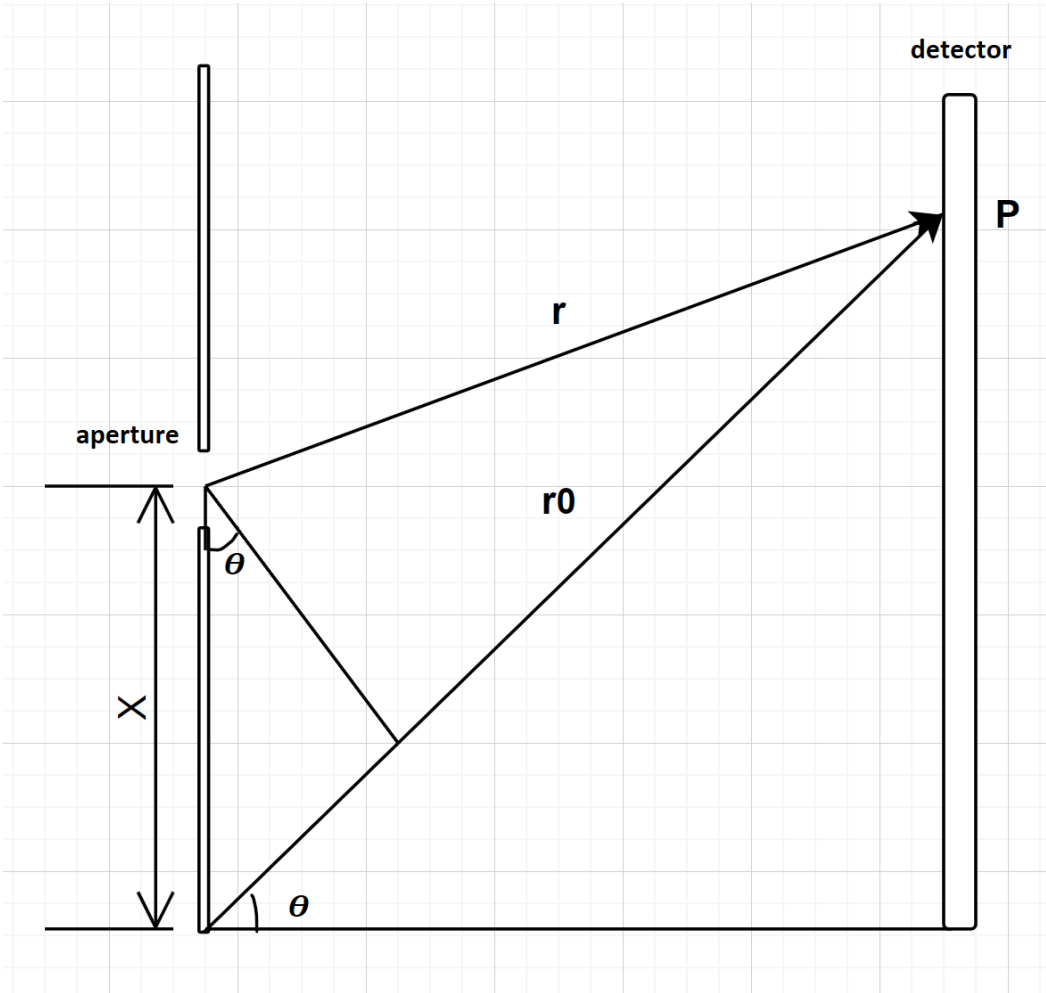


Figure 2.2: Scheme of calculation for light density in point P

Substituting the above approximation $r = r_0 - xsin\theta$ into the expression for the light wave, we obtain

$$E = E_0 e^{2\pi ivt} \int_{aperture} e^{\frac{2\pi i(r_0 - xsin\theta)}{\lambda}} dx$$

$$E = E_0 e^{2\pi ivt} e^{\frac{2\pi ir_0}{\lambda}} \int_{aperture} e^{\frac{2\pi i * -xsin\theta}{\lambda}} dx$$

At this point, outside the integral, there are only constants, and our focus is on the integral part. Therefore, we can express the expression as:

$$E \propto \int_{aperture} e^{\frac{-2\pi ixsin\theta}{\lambda}} dx$$

Here, we make $p = sin\theta / \lambda$, then

$$E \propto \int_{aperture} e^{-2\pi ipx}\, dx$$

For the small aperture on the diffraction screen, we can represent it using the aperture function A(x)

$$A(x)\begin{cases} = 1\ , & x \in aperture \\ = 0\ , & otherwise \end{cases}$$

Only the small aperture allows the passage of light waves, and the rest is non-transmissive. If there is a specimen in this region, there will be a numerical value between 0 and 1 representing the transmittance. Then, we can obtain

$$E \propto \int_{-\infty}^{\infty} A(x) e^{-2\pi ipx}\, dx$$

At this point, it can be observed that on the right side of the equation is precisely the Fourier transform of the function A(x)

$$E \propto \mathrm{F\,A(p)} \quad , \qquad p = x sin\theta / \lambda$$

We can get a conclusion that Intensity of the light is the amplitude of the Fourier Transform of aperture function.

2.2 Fourier transformation

Fourier Transform is a mathematical tool that emerged in the 19th century, designed to transform a function related to time or space into another set of functions composed of sine and cosine functions. The transformative idea is to convert a time-domain function into a linear combination of numerous periodic functions, thereby revealing the characteristics of that function in the frequency domain.

Periodic functions serve as mathematical representations of periodic motions in the objective world, such as the simple harmonic motion of an object hanging on a spring, the oscillation of a pendulum, or the electronic oscillation in a radio electronic oscillator. Most of these phenomena can be expressed in the form of periodic functions.

In a more understandable explanation, Fourier Transform is like a magical black technology in mathematics. Its mission is to transform a messy entity in time or space into a group of well-behaved frequencies. Imagine holding a pen that transforms time's canvas into a palette of frequencies. On this palette, you can use pigments of various frequencies to create one pattern after another.

The formula for Fourier Transform in complex form is as follows:

$$F(\omega) = \int_{-\infty}^{\infty} f(t) e^{-2\pi i \omega t} \, dx$$

The function F(ω) represents a complex function in the frequency domain, where ω is the frequency, and $e^{-2\pi i \omega t}$ is the complex exponential function.

The above formula represents the continuous Fourier transform. However, in real-world applications, the data we can collect is often discrete. Continuous and infinitely divisible data typically only exists in definitions. Therefore, in engineering practical applications, we need to use the Discrete Fourier Transform (DFT) instead of the continuous Fourier transform, as we cannot obtain continuous and infinitely divisible data.

The Discrete Fourier Transform (DFT) makes a series of assumptions to enable Fourier transformation on imperfect data. These assumptions include uniformly sampling data at equal time intervals, the discrete data have a complete cycle, and so on. Based on these assumptions, the Fourier transform, which needs to determine the distribution of each frequency, is transformed into a series of linear operations.

In contrast to the continuous Fourier transform, where data is infinite, the data in the case of DFT is finite and can be expressed as a matrix multiplication. The unknown components of each frequency, multiplied by the corresponding time, result in the known sampled data. Therefore, solving for the components of each frequency becomes a linear matrix operation, The practical operation of solving the Discrete Fourier Transform is just a massive matrix multiplication.

We use the Fast Fourier Transform (FFT) in our experiment, which is a specific form of Discrete Fourier Transform (DFT). Compared to the regular DFT, FFT accelerates the computation speed by leveraging the periodicity and symmetry of the signal. Through stepwise reduction of the problem size, FFT reduces the original computational complexity from $O(N^2)$ to $O(NlogN)$.

This divide-and-conquer strategy makes FFT faster in handling large-scale data compared to directly computing the DFT. In terms of specific operations, FFT optimizes the multiplication of the massive matrix involved in the Discrete Fourier Transform itself. The approach involves breaking down the large matrix into several smaller matrices (typically four). After the matrix is decomposed into smaller ones, a transformation is performed to convert it into a diagonal matrix multiplication, thereby saving a significant amount of time.

2.3 The importance of phase retrieval

As we discussed earlier, Fourier Transform generates two components: phase and amplitude. However, the image received on the detector represents the amplitude result of the Fourier Transform of the specimen and aperture. For two-dimensional Fourier Transform, the amplitude reflects the intensity of various frequencies in the two-dimensional image. In images, amplitude tends to determine brightness and color intensity, while phase plays a more significant role in defining the contours of the image. Here, we perform Fourier Transform on two different images to obtain their respective phase and amplitude. Afterward, we conduct Fourier Inverse Transform while keeping the amplitude unchanged. Instead, we replace the original phase with the phase from the other image. In this process, we observe that the restored result, especially in terms of phase, approximates the contours of the image associated with that particular phase.

This diagram illustrates how crucial a role the phase plays in the accurate restoration of the image.

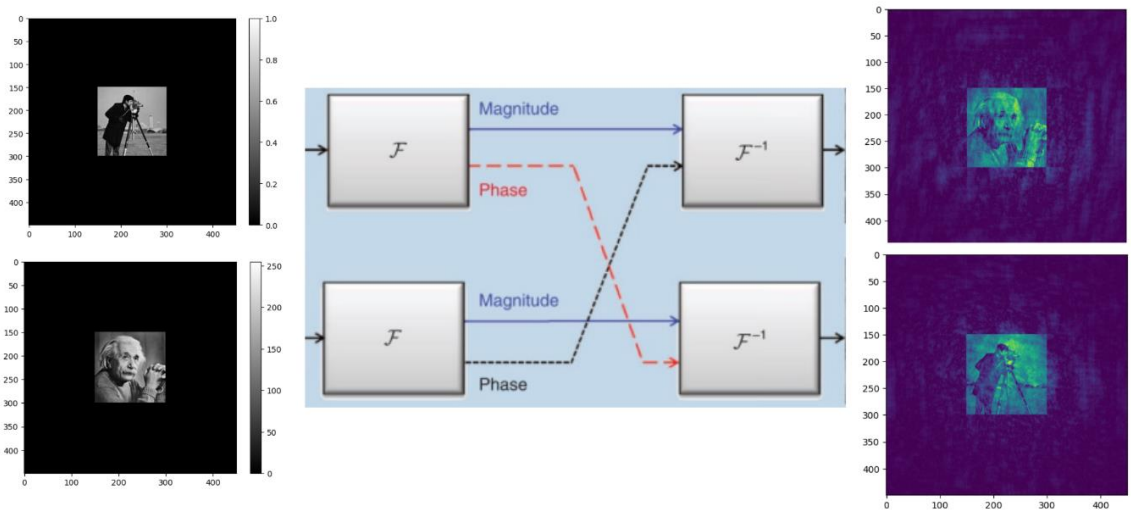


Figure 2.3: Experiment result of exchange phase in inverse Fourier transformation

Chapter 3

Related Works

3.1 Phase retrieval method: ER and HIO

Methods like ER and HIO leverage known information to progressively approach the correct phase during the iterative process. The known information typically involves the shape of our aperture since we have no prior knowledge of the structure of the observed specimen. However, the aperture itself is part of our observation system, a component in our system design. Therefore, the information about the aperture is known. In this context, we can initially generate a random phase and use this random phase for Fourier inverse transformation to restore the original image.

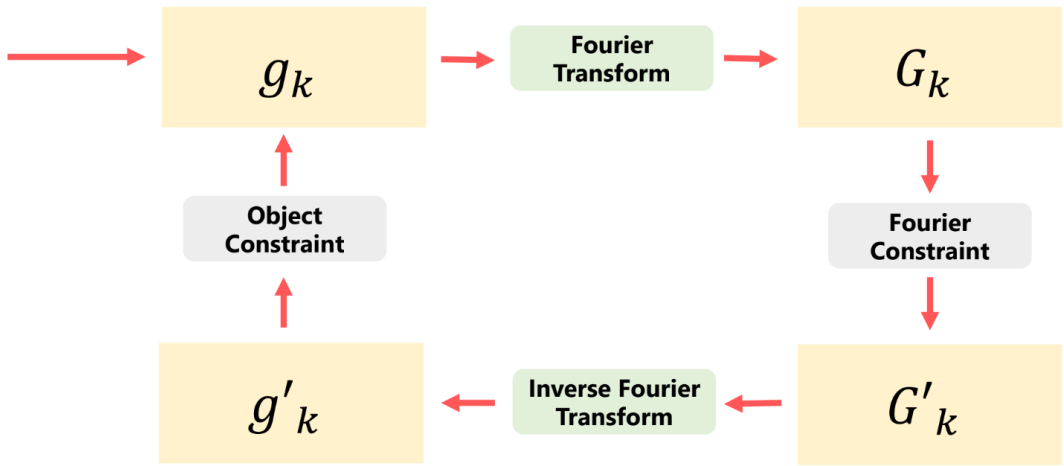


Figure 3.1: Diagram of the steps in the HIO and ER algorithms

Because the original image outside the aperture is non-transmissive (having intensity values of 0), using a random phase during the restoration process results in incorrect values outside the aperture. To address this, for the ER algorithm, known erroneous values are set to 0. In contrast, the HIO algorithm reduces erroneous values to a certain ratio (e.g., 0.5). This process yields a new restored image, which is then Fourier-transformed to obtain a new diffraction pattern. This iterative cycle is repeated several times. Through multiple iterations, we gradually recover an original image that approximates the correct image. This iterative approach effectively utilizes our known information about the original image's aperture to restore the phase.

The ER Algorithm

$$g_{k+1}(x) = \begin{cases} g'_k(x), & x \notin \gamma, \\ 0, & x \in \gamma. \end{cases}$$

The HIO Algorithm

$$g_{k+1}(x) = \begin{cases} g'_k(x), & x \notin \gamma, \\ g'_k(x) - \beta g'_k(x), & x \in \gamma \end{cases}$$

Figure 3.2: Difference of error handling strategy between HIO and ER algorithms

3.2 Phase retrieval method: EPIE

The EPIE algorithm, also based on iterations, differs slightly from the previous ER and HIO algorithms. This algorithm scans the diffraction pattern into partitioned blocks with some overlap between adjacent blocks. Each block corresponds to an independent sub-diffraction pattern.

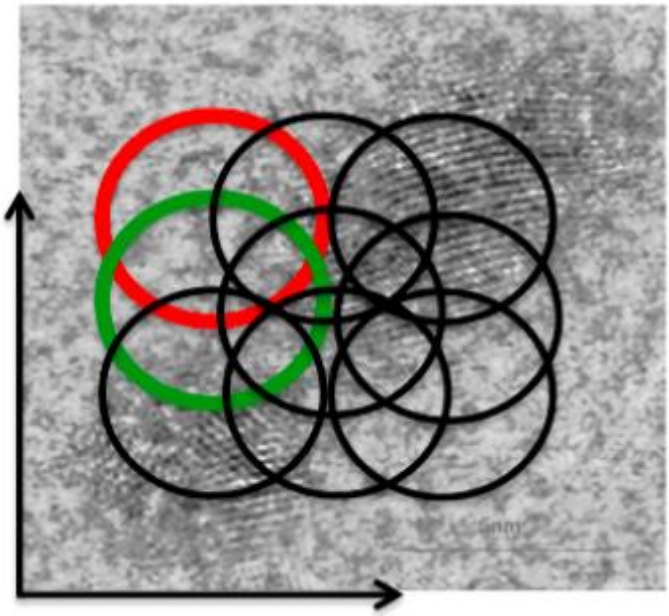


Figure 3.3: Partitioned blocks in EPIE Algorithm

Initially, we randomly initialize an original sample object and original probe as original input image. After Fourier transforming this original sample image, it generates a diffraction pattern. Typically, there is a difference between the diffraction pattern based on the randomly generated sample and the true sample.

In the EPIE algorithm, we iterate through each sub-diffraction pattern from the true sample and correct the randomly generated original sample image accordingly. Because each sub-diffraction pattern overlaps with others, they share some information. After correcting the original image for each individual sub-diffraction pattern, we obtain a new original image. This new original image generates adjacent sub-diffraction samples. Due to the overlap between adjacent sub-diffraction samples, this overlapping part helps

make the new sub-diffraction samples generated based on the assumed original image closer to the true sub-diffraction samples. Through multiple cycles of iteration, the assumed original image gradually approaches the true original image.

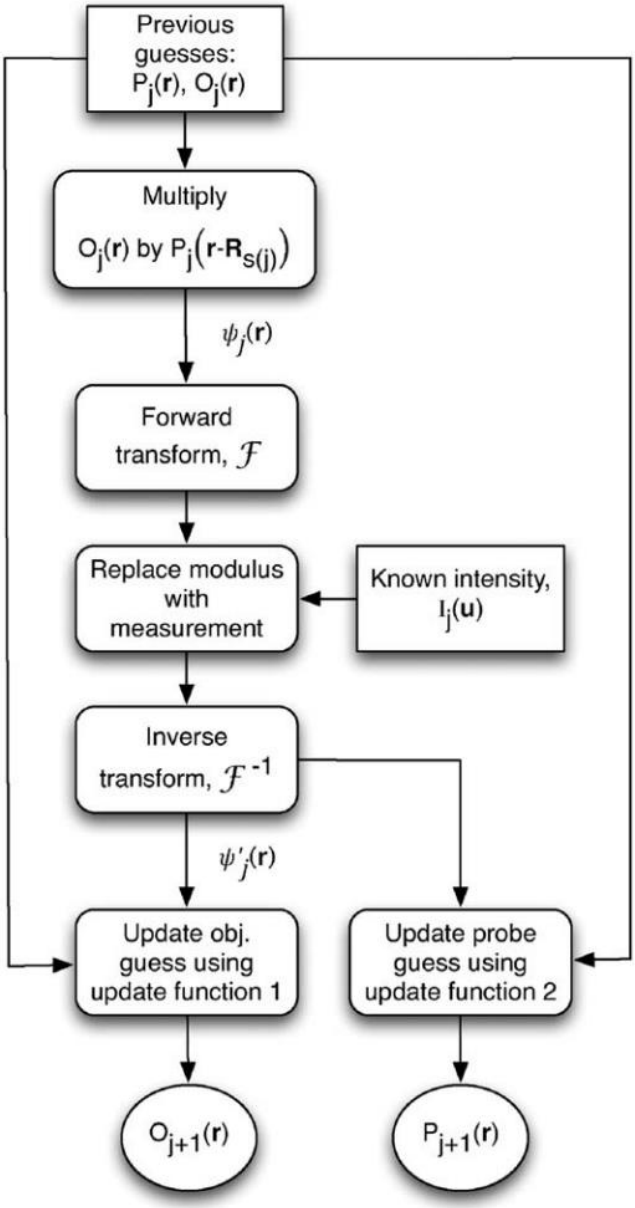


Figure 3.4: The flow of EPIE Algorithm

$$O_{j+1}(\mathbf{r}) = O_j(\mathbf{r}) + \alpha \frac{P_j^*(\mathbf{r} - \mathbf{R}_{s(j)})}{|P_j(\mathbf{r} - \mathbf{R}_{s(j)})|^2_{\max}} (\psi_j^{'}(\mathbf{r}) - \psi_j(\mathbf{r})).$$

$$P_{j+1}(\mathbf{r}) = P_j(\mathbf{r}) + \beta \frac{O_j^*(\mathbf{r} + \mathbf{R}_{s(j)})}{|O_j(\mathbf{r} + \mathbf{R}_{s(j)})|^2_{\max}} (\psi_j^{'}(\mathbf{r}) - \psi_j(\mathbf{r})).$$

Figure 3.5: Object and probe update method in EPIE

Chapter 4
Experimentation

4.1 experiment environment setting

Here, we utilized python code that build a simulation for generating coherent X-rays, molecular specimen, apertures, detectors, and also the light diffraction.

For coherent x-ray, we using code to set these different property

Like light energy, light wave length, light energy, light speed, plank constant.

Also setting the detector in here, like the detector resolution, distance between the specimen and the detector, each pixel size in detector.

```
class optical_condition:
    L:      float # distance from the sample plane to the detector plane
    p:      float # pixel size of the detector
    Npixel: int   # window size of the probe function, object functions, and diffraction patterns
    energy: float # x-ray energy of the incident beam
    h:      float # plank constant
    c:      float # velocity of the light
    lamb:   float # wavelength
    k:      float # wavenumber
    dx:     float # pixel resolution at the sample plane
    DR:     float # dynamic range of the detector, for example opt.DR = log10{1e7 (photons) / 1 (photon}
    Wnoise: bool  # flag to apply photon shot noise to each calculated diffraction pattern


opt         = optical_condition()

opt.L       = 3                                         # unit: m
opt.p       = 75e-6                                     # unit: m
opt.Npixel  = 1024                                      # unit: pixel
opt.energy  = 5000                                      # unit: eV (electronVolt)
opt.h       = 6.62607015e-34                            # unit: m^2 kg/s
opt.c       = 299792458                                 # unit: m/s
opt.lamb    = (opt.h * 6.242e18) * opt.c / opt.energy   # unit: m
opt.k       = 2 * np.pi / opt.lamb                      # unit: 1/m
opt.dx      = opt.L * opt.lamb / (opt.Npixel * opt.p)   # unit: m/pixel
opt.DR      = 7
opt.Wnoise  = True
```

Figure 4.1: optical setting code, include x-ray and detector

We also set our aperture in here, include the aperture size, aperture shape, the distance between our aperture and specimen.

```
class aperture_condition:
    sideNumber:         int     # number of the side
    sideLength:         float   # length of a side
    Lpropagation:       float   # distance from the aperture to the sample
    needPropagation:    bool    # flag whether propagation from the aperture to the sample is needed

aper = aperture_condition()

aper.sideNumber     = 3     # input: 3 --> Triangle
aper.sideLength     = 4e-6  # unit: m
aper.Lpropagation   = 0  # 5e-4     # unit: m, range: # 500-1000*10^-6

if aper.Lpropagation > 0:
    aper.needPropagation = True
else:
    aper.needPropagation = False
```

Figure 4.2: aperture setting code

The last is our specimen, we setting our specimen as golden particles, and then set the particle property, like it's light wave absorption rate, light wave phase shift rate when light pass the particle, diameter of the particle, particle move velocity, and the total number of particles.

```
class particle_condition:
    beta_Au:    float # parameter related to the absorption of the wavefield
    delta_Au:   float # parameter related to the phase shift of the wavefield
    diameter:   float # diameter of the gold nanoparticle
    ratio:      float # (Area occupied with gold nanoparticles) / (Area of the window, Npixel x Npixel)
    number:     int   # total number of the gold nanoparticles within the window
    velocity:   float # displacement of the gold nanoparticles (per frame)

par = particle_condition()

par.beta_Au     = 2.633e-5
par.delta_Au    = 1.2143e-4
par.diameter    = 300e-9 #300e-9 # unit: m
par.ratio       = 100e-3
par.number      = int(np.floor((opt.Npixel * opt.dx)**2 / (np.pi * (par.diameter / 2)**2) * par.ratio))
par.Nframe      = 11
par.velocity    = 1 * opt.dx    # unit: m/frame
```

Figure 4.3: specimen setting code

According to our simulation code, our data generation process is as follows:

- Coherent X-ray light is generated and illuminates the aperture and specimen.
- The detector records the corresponding light intensity and collects data.

The most crucial data we can obtain here includes the aperture, specimen, and the diffraction results collected by the detector. Since this simulation is based on code, we have information about the microstructure of the specimen. However, in a real observe system, the structure of the specimen is unknown and need to be observed. Therefore, the data of the specimen just serves as validation data to check whether the reconstructed image matches its original image.

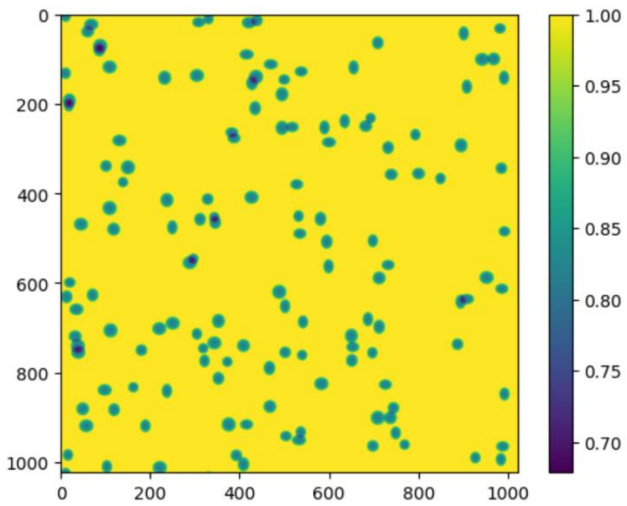


Figure 4.4: generating Au particles

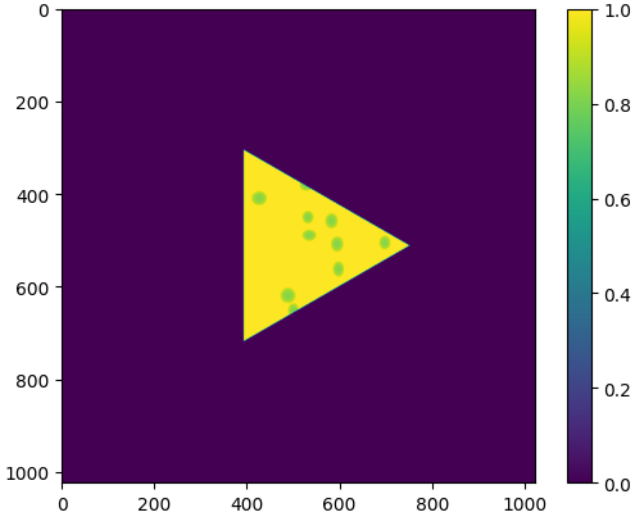


Figure 4.5: particles with triangle aperture

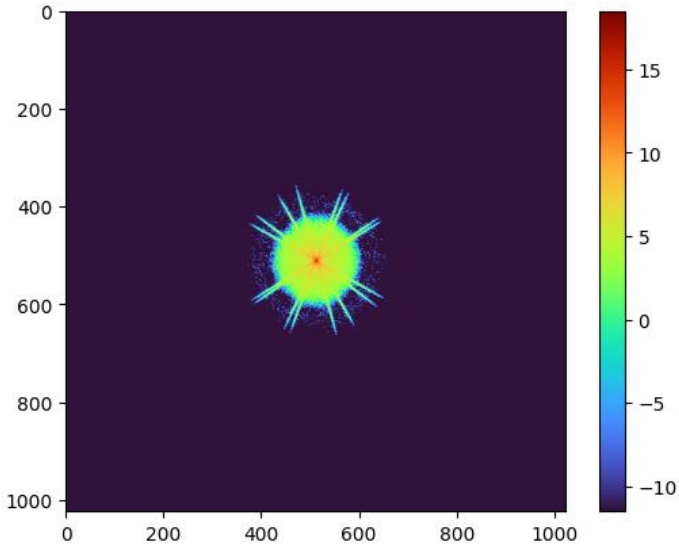


Figure 4.6: diffraction result in detector

4.2 phase retrieval in our experiment

Based on the previously mentioned code simulation of the diffraction system, we have simulated data, including the aperture, specimen, and diffraction detected by the detector. The diffraction result corresponds to the Fourier transform amplitude of the aperture and specimen. Now, we need to reconstruct the original specimen and aperture from the detector's diffraction image. The reconstruction involves retrieving the Fourier transform phase, which is not directly observable in the observation system. Therefore, we use phase retrieval methods to find the phase close to the original image. In this context, the aperture in the observation system is known, and we also have knowledge of the Fourier transform amplitude. Leveraging these two pieces of known data, we attempt to reconstruct the original specimen.

4.3 phase retrieval with HIO and ER algorithms

We will first implement the classic iterative phase retrieval algorithms, ER (Error Reduction) and HIO (Hybrid Input-Output), using code. Subsequently, we will observe the effectiveness of phase retrieval on our data.

As introduced in Section 3.1, both the ER and HIO algorithms utilize object constraint and Fourier constraint for iterative phase retrieval. The initialization and iteration processes in the code are as follows:

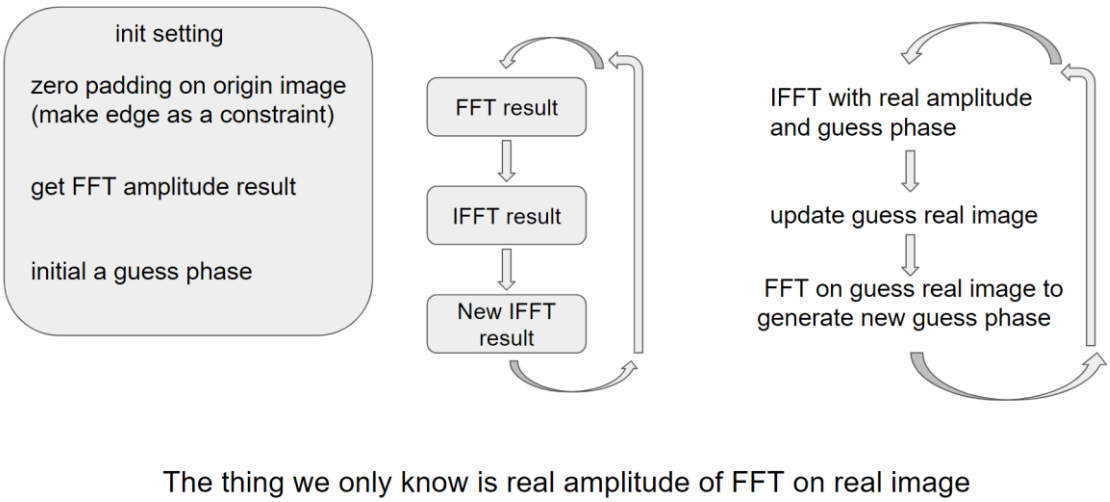
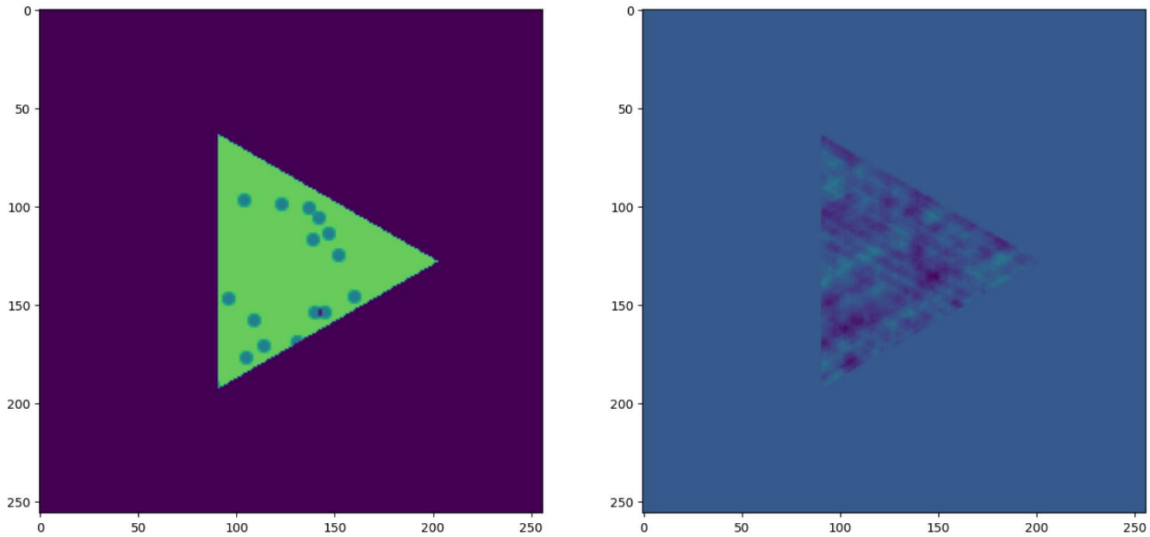


Figure 4.7: ER/HIO algorithm initialization and iteration

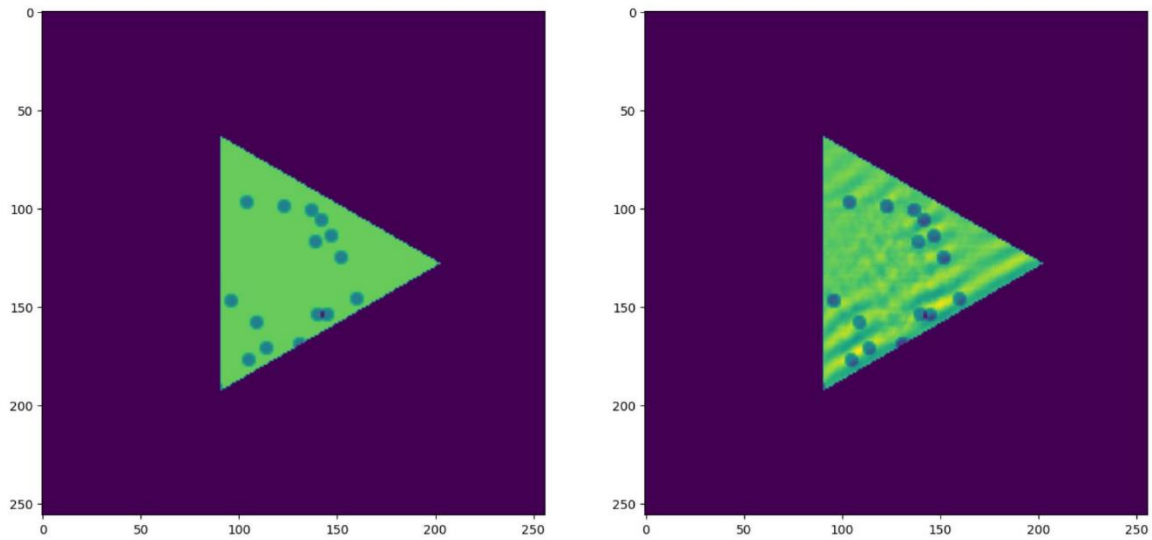
- FFT: Fast Fourier Transform
- IFFT: Inverse Fast Fourier Transform

Here, we use the HIO algorithm for iteration. The images below show the reconstruction results at the 10th and 1000th iterations. It can be observed that the reconstruction quality is relatively poor with a small number of iterations, and as the number of iterations increases, the reconstruction quality gets closer to the original image. However, further improvement tends to plateau.



Left: original image Right: reconstruction image

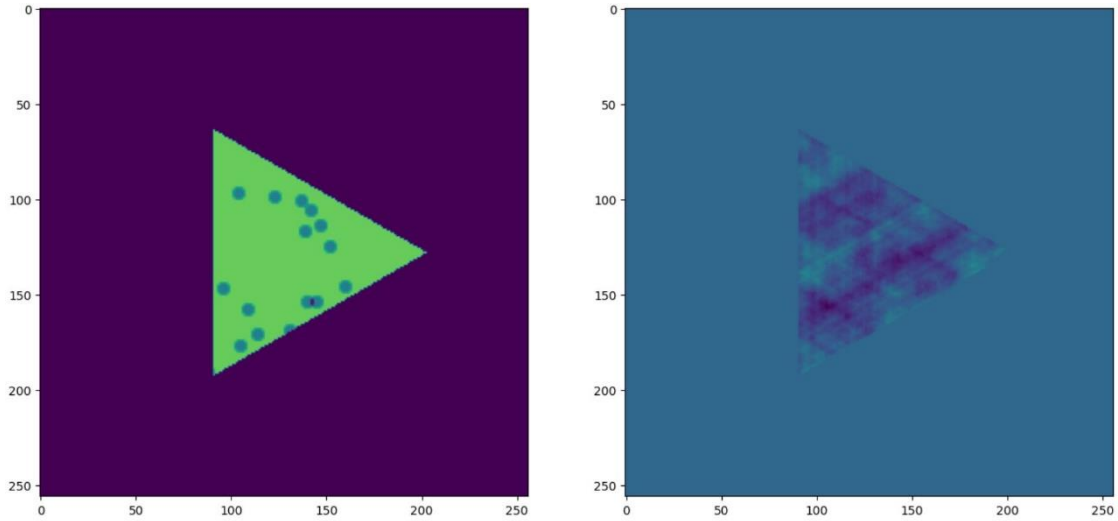
Figure 4.8: Phase Retrieval effect with **10** iterations by HIO algorithm



Left: original image Right: reconstruction image

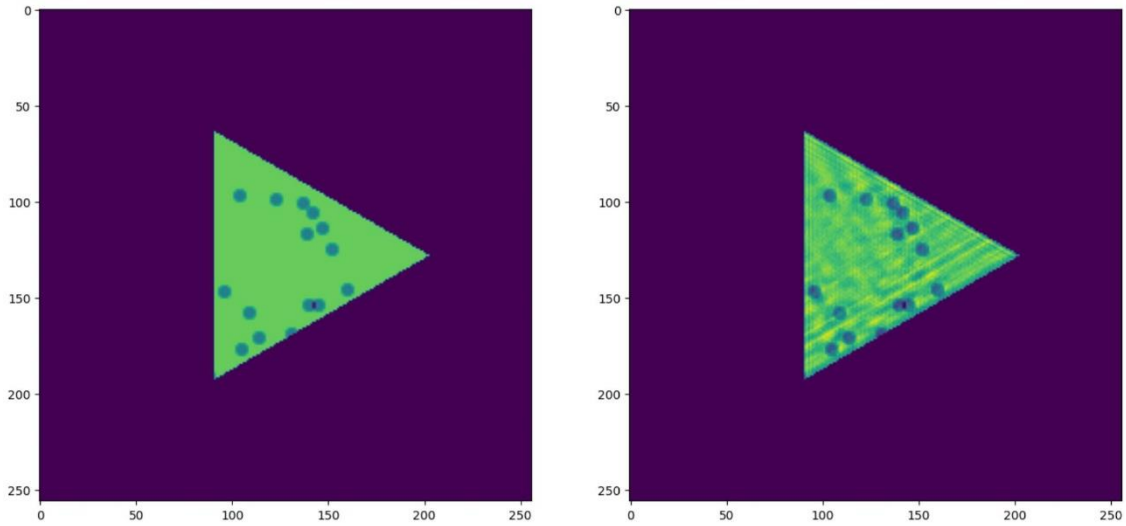
Figure 4.9: Phase Retrieval effect with **1000** iterations by HIO algorithm

We also use the ER algorithm for iteration, the difference between ER and HIO is just the error update strategy, ER set the error part to 0, and HIO reduce it's to a partition. The images below show the reconstruction results at the 10th and 2000th iterations, as the number of iterations increases, the reconstruction quality gets closer to the original image. However, further improvement tends to plateau.



Left: original image Right: reconstruction image

Figure 4.10: Phase Retrieval effect with **10** iterations by ER algorithm



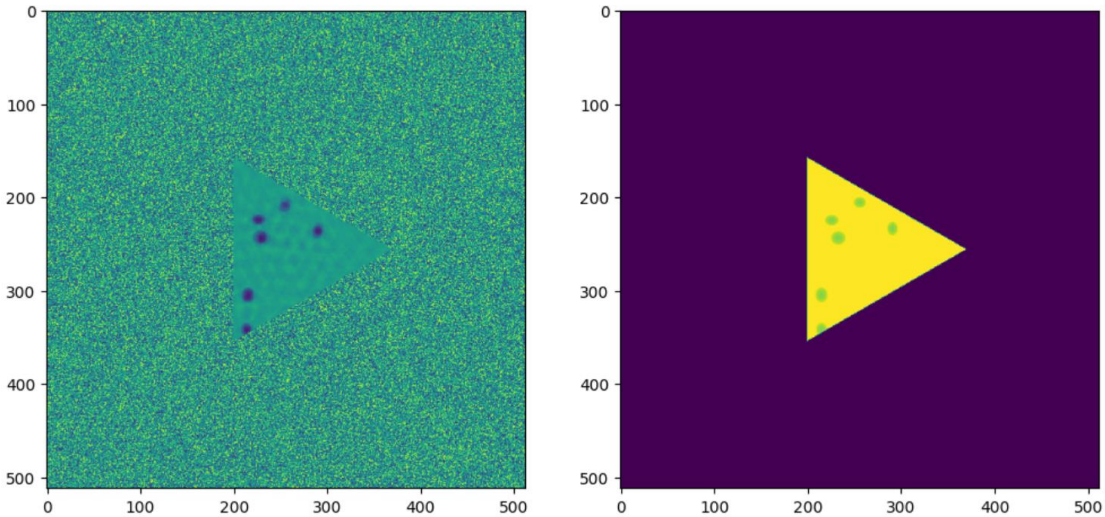
Left: original image Right: reconstruction image

Figure 4.11: Phase Retrieval effect with **2000** iterations by ER algorithm

4.4 phase retrieval with Epie algorithm

The Epie algorithm segments individual image data into overlapping blocks, utilizing a significant number of overlapping blocks to achieve oversampling. Without overlap and oversampling, the algorithm degrades into the HIO algorithm, providing decent results in a fewer number of iterations. Additionally, there is no need to pre-specify the shape of the aperture in Epie. During the update process, Epie dynamically generates an assumed aperture and updates it along with the object.

Bottom show the particles position reconstruction effect from diffraction.



Left: reconstruction image Right: original image

Figure 4.12: Phase Retrieval effect with 10 iterations by Epie algorithm

Compare to ER and HIO, Epie need much less iteration that can get an accepted result.

4.5 improving phase retrieval by combine EPIE algorithm with gradient decent approach

Here, we can consider using the gradient descent method to moderately improve the algorithm's performance. If we have a considerable amount of data, we may explore using deep learning for phase retrieval. Since the discrete Fourier transform itself involves a massive matrix multiplication, which is similar to neural network operations, we can use a neural network to some extent to fit the original Fourier transform calculation. However, generating coherent X-rays is challenging and comes with high data costs. In scenarios with insufficient data, the effectiveness of deep learning tends to be suboptimal. Therefore, we can consider combining the EPIE algorithm with simple data science strategies like gradient descent during the EPIE algorithm's iterative process to attempt to enhance its performance.

We define the loss function as the difference between the assumed sample and the diffraction image calculated from current generated object, the object is generated from the current stage of the EPIE algorithm. We use the Adam optimizer as the gradient descent method. Since the EPIE algorithm itself does not require many iterations, we perform gradient descent 30 times between each iteration interval of the EPIE algorithm. In other words, within 30 iterations of gradient descent, we perform 1 iteration of the EPIE algorithm. Each batch contains 8 diffraction data samples, using only 64 data samples.

Experiment result

```
iterations: 0    loss:  tf.Tensor(1372.2244, shape=(), dtype=float32)
iterations: 30    loss:  tf.Tensor(67.876045, shape=(), dtype=float32)
iterations: 60    loss:  tf.Tensor(67.605286, shape=(), dtype=float32)
iterations: 90    loss:  tf.Tensor(67.31621, shape=(), dtype=float32)
iterations: 120    loss:  tf.Tensor(67.01907, shape=(), dtype=float32)
iterations: 150    loss:  tf.Tensor(66.71775, shape=(), dtype=float32)
iterations: 180    loss:  tf.Tensor(66.41411, shape=(), dtype=float32)
iterations: 210    loss:  tf.Tensor(66.109215, shape=(), dtype=float32)
iterations: 240    loss:  tf.Tensor(65.80371, shape=(), dtype=float32)
iterations: 270    loss:  tf.Tensor(65.497986, shape=(), dtype=float32)
iterations: 300    loss:  tf.Tensor(65.19233, shape=(), dtype=float32)
```

Figure 4.13: Phase Retrieval loss with 300 iterations by only gradient decent

```
iterations: 0    loss:  tf.Tensor(1372.214, shape=(), dtype=float32)
iterations: 30    loss:  tf.Tensor(68.149895, shape=(), dtype=float32)
iterations: 60    loss:  tf.Tensor(11.2663555, shape=(), dtype=float32)
iterations: 90    loss:  tf.Tensor(7.6797743, shape=(), dtype=float32)
iterations: 120    loss:  tf.Tensor(6.6070204, shape=(), dtype=float32)
iterations: 150    loss:  tf.Tensor(6.108151, shape=(), dtype=float32)
iterations: 180    loss:  tf.Tensor(5.7521863, shape=(), dtype=float32)
iterations: 210    loss:  tf.Tensor(5.523768, shape=(), dtype=float32)
iterations: 240    loss:  tf.Tensor(5.361302, shape=(), dtype=float32)
iterations: 270    loss:  tf.Tensor(5.239236, shape=(), dtype=float32)
iterations: 300    loss:  tf.Tensor(5.15006, shape=(), dtype=float32)
```

Figure 4.14: Phase Retrieval loss with 300 iterations by only EPIE algorithm

```
iterations: 0    loss:  tf.Tensor(1372.221, shape=(), dtype=float32)
iterations: 30    loss:  tf.Tensor(65.96863, shape=(), dtype=float32)
iterations: 60    loss:  tf.Tensor(10.801302, shape=(), dtype=float32)
iterations: 90    loss:  tf.Tensor(7.4121976, shape=(), dtype=float32)
iterations: 120    loss:  tf.Tensor(6.330072, shape=(), dtype=float32)
iterations: 150    loss:  tf.Tensor(5.868695, shape=(), dtype=float32)
iterations: 180    loss:  tf.Tensor(5.5139236, shape=(), dtype=float32)
iterations: 210    loss:  tf.Tensor(5.298865, shape=(), dtype=float32)
iterations: 240    loss:  tf.Tensor(5.1233735, shape=(), dtype=float32)
iterations: 270    loss:  tf.Tensor(4.996434, shape=(), dtype=float32)
iterations: 300    loss:  tf.Tensor(4.8908696, shape=(), dtype=float32)
```

Figure 4.15: Phase Retrieval loss with 300 iterations by combine EPIE algorithm with gradient decent

Here, we can see that the EPIE algorithm, combined with the gradient descent strategy, converges the fastest, Within the same number of iterations, compared to the standalone gradient descent strategy and the standalone EPIE algorithm, its image restoration performance is the best. The standalone EPIE algorithm also outperforms the standalone gradient descent strategy in terms of image restoration effectiveness.

When we attempted to increase the data, we found that the convergence speed of the loss became faster.

```
iterations: 0    loss:  tf.Tensor(1332.0299, shape=(), dtype=float32)
iterations: 30    loss:  tf.Tensor(59.391056, shape=(), dtype=float32)
iterations: 60    loss:  tf.Tensor(8.758686, shape=(), dtype=float32)
iterations: 90    loss:  tf.Tensor(5.2599287, shape=(), dtype=float32)
iterations: 120    loss:  tf.Tensor(3.9665828, shape=(), dtype=float32)
iterations: 150    loss:  tf.Tensor(3.3090901, shape=(), dtype=float32)
iterations: 180    loss:  tf.Tensor(2.931524, shape=(), dtype=float32)
iterations: 210    loss:  tf.Tensor(2.7075634, shape=(), dtype=float32)
iterations: 240    loss:  tf.Tensor(2.5586162, shape=(), dtype=float32)
iterations: 270    loss:  tf.Tensor(2.4517176, shape=(), dtype=float32)
iterations: 300    loss:  tf.Tensor(2.3731465, shape=(), dtype=float32)
```

Figure 4.16: Phase Retrieval loss with 300 iterations by combine EPIE algorithm with gradient decent with increasing data

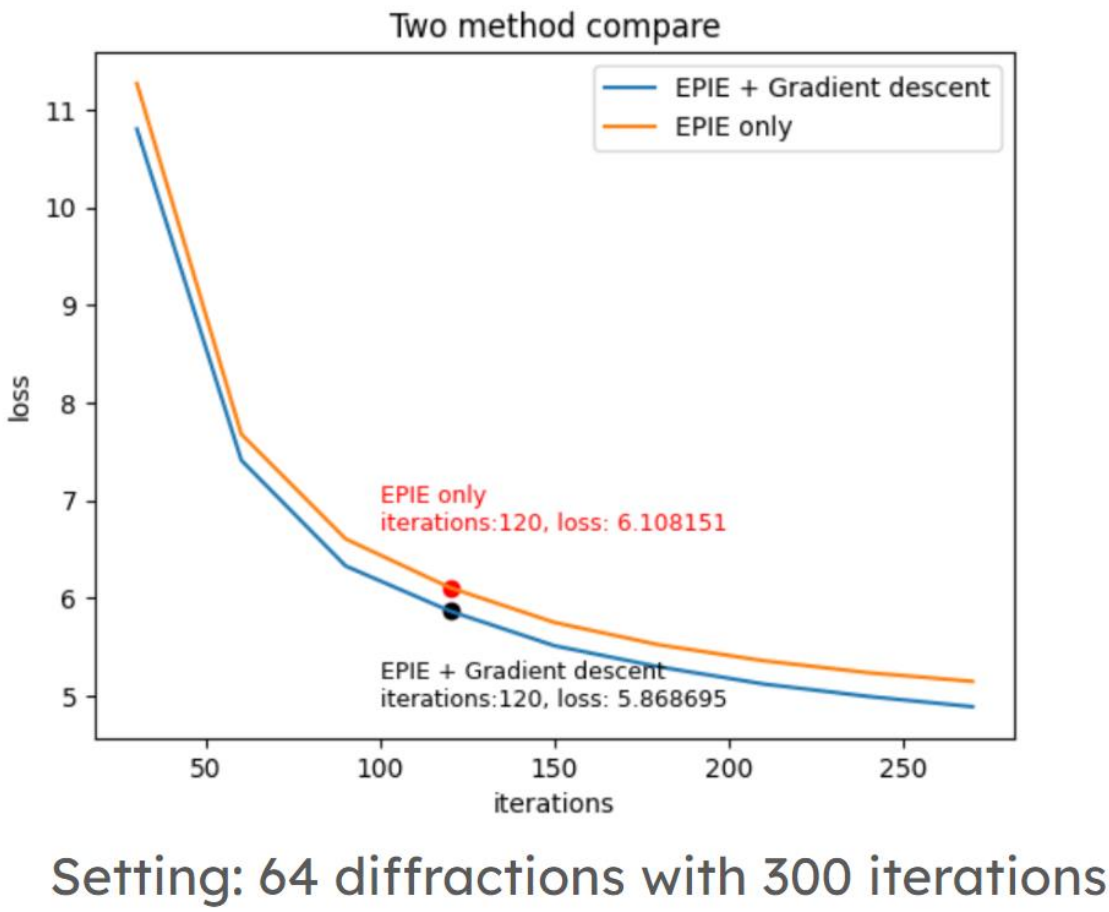
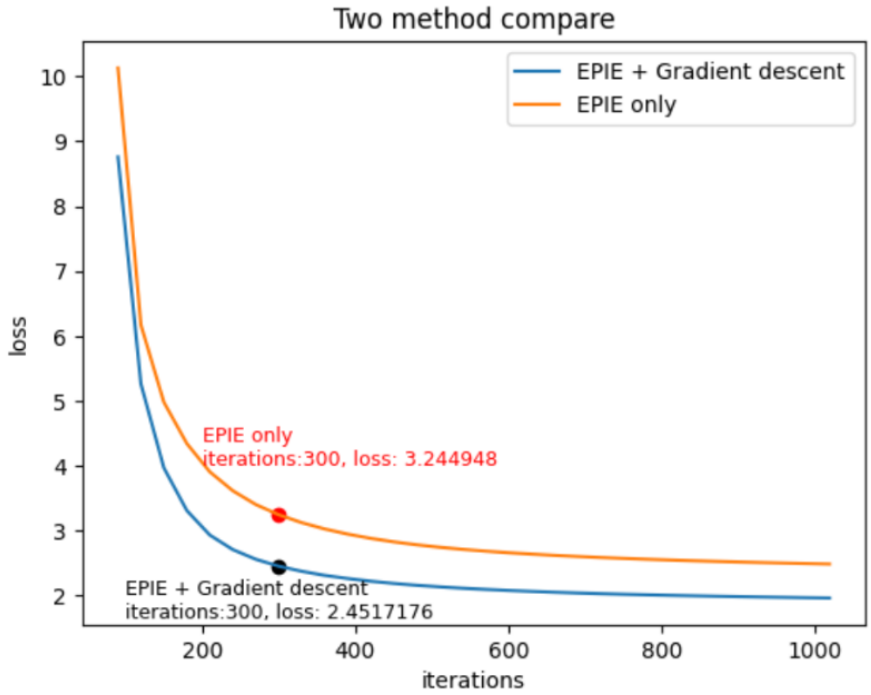


Figure 4.17: Phase Retrieval effect compare with 64 diffractions



Setting: 512 diffractions with 1000 iterations

Figure 4.18: Phase Retrieval effect compare with 512 diffractions

Chapter 5
Conclusion

From the experimental results, it can be observed that traditional phase retrieval iterative algorithms can achieve acceptable image reconstruction even in the absence of phase information. Moreover, even in cases with limited data, employing a data science approach can lead to modest improvements over traditional methods, and when we attempt to increase the data, we find that the improvement becomes even more pronounced. We believe that the potential of these methods will become even greater with larger datasets.

Bibliography

[1] J. Rodenburg and A. Maiden, “Ptychography,” Springer Handbooks, pp. 819–904, 2019

[2] M. Nakasako et al., “Methods and application of coherent X-ray diffraction imaging of noncrystalline

[3] J. R. Fienup, “Phase retrieval algorithms: a comparison,” Applied Optics, Vol. 21, Issue 15, pp. 2758-2769, vol. 21, no. 15, pp. 2758–2769, Aug. 1982

[4] Andrew M. Maiden, John M. Rodenburg,An improved ptychographical phase retrieval algorithm for diffractive imaging,Ultramicroscopy,Volume 109, Issue 10,2009,Pages 1256-1262,ISSN 0304-3991

[5] Manekar, Raunak, et al. "Deep learning initialized phase retrieval." NeurIPS 2020 Workshop on Deep Learning and Inverse Problems. 2020.

[6] T. Latychevskaia, "Iterative phase retrieval in coherent diffractive imaging: practical issues," Appl. Opt. 57, 7187-7197 (2018).

[7] Taylor, L. "The phase retrieval problem." IEEE Transactions on Antennas and Propagation 29.2 (1981): 386-391.

[8] Takazawa, Shuntaro, et al. "Demonstration of single-frame coherent X-ray diffraction imaging using triangular aperture: Towards dynamic nanoimaging of extended objects." Optics Express 29.10 (2021): 14394-14402.

[9] Kang, Jungmin, et al. "Single-frame coherent diffraction imaging of extended objects using triangular aperture." Optics Express 29.2 (2021): 1441-1453.