MICROSCOPY OF GOLD MICROCRYSTALS BY COHERENT X-RAY DIFFRACTIVE IMAGING

BY

GARTH JONATHAN WILLIAMS

B.S., University of Akron, 1998
M.S., University of Illinois, 2000

DISSERTATION

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Physics
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2005

Urbana, Illinois

Acknowledgments

I am tremendously grateful to all those who have made this dissertation possible.

In particular, I am indebted to my advisor Ian K. Robinson, whose patience and deep knowledge of his field made this project feasible. The experiment described herein would have been enormously cumbersome without the use of the Advanced Photon Source and so I must thank the staff of UNICAT for their tireless efforts in maximizing the productivity of that facility for their users.

I also wish to thank Ivan Vartanyants for his many contributions to this work. I truly enjoyed our many conversations on the methods discussed herein, as well as those on their extension and outlook, and I thank him for sharing his experience and expertise.

I have also enjoyed conversations, interrogations, and debates with with my fellow graduate students M. Pfeifer, J. Pitney, S. Boutet, T. Angelini, W. Zhang, Y. Da, M. Liang, and V. Crecea. Discussions with R. Harder about the reliability and extension of these methods also proved helpful.

Lastly, I gratefully acknowledge the patience and understanding displayed by my family as I pursued this goal.

This research was supported by the NSF under grants DMR98-76610 and DMR03-08660. The UNICAT facility at the Advanced Photon Source (APS) is supported by the University of Illinois at Urbana-Champaign, Materials Research Laboratory (U.S. DOE contract DEFG02-91ER45439, the State of Illinois-IBHE-HECA, and the NSF), the Oak Ridge National Laboratory (U.S. DOE under contract with UT-Battelle LLC), and the National Institute of Standards and Technology (U.S. Department of Commerce).

Table of Contents

List of Tables

List of Figures

Chapter 1

Introduction

Many important problems in Physics require a 3D map of a system to answer completely. In materials science, these problem range from the effect of granular structure on the failure of semiconductor devices through electromigration to imaging the strain map around a single defect in a crystal. Further, questions of basic import, for example, the absolute free energy of crystal's face, can be answered by imaging the 3D density of the crystal and comparing the equilibrium crystal shape at different temperatures. Similarly, applications exist in novel materials, surface chemistry, and biological systems. Unfortunately, acquiring such a map is extremely difficult.

X-rays are a logical choice as a probe particle for a technique of this kind. They have typical wavelength of $\sim$1 Å, very close to atomic spacings in matter, promising high resolution imaging. X-rays are highly penetrating, which is an important requirement for a truly 3D technique. Further, the interaction of X-rays with matter is very small for inorganic materials so they may be regarded non-interacting probe particles, which is not always a good approximation for electrons.

Imaging with X-rays is therefore a very tantalizing prospect. Unfortunately, X-rays also have indices of refraction very slightly less than unity in most materials [98, 7], making the manufacture of focusing optics very difficult, although amazing progress is being made for parabolic refractive lenses[82], phase zone plates[100], and Kirkpatrick-Baez mirrors[41, 44]. It is with this in mind that we ask whether the lens of the X-ray instrument might be replaced by a computation, yielding a lensless X-ray microscope.

The remainder of this chapter provides background material required for understanding the results and methods in subsequent chapters. We begin with a short discussion of the inverse problem inherent in the technique and the variety of topics in the physical and biological systems in which it appears. Next, the interaction of X-rays with matter will be discussed,

culminating in the result which links the 3D density map to the diffracted X-ray intensity. Of pivotal importance is the algorithm used to solve the "phase problem" we are presented with, so a short literature review is provided.

Chapter 2 is entirely devoted to describing the algorithms and their variants which will be used and tested in later chapters. Chapter 3 provides a detailed description of a 3D coherent X-ray diffraction (CXD) experiment conducted on a Au sample at the Advanced Photon Source (APS). This chapter contains a discussion of the methodology used to recover the 3D density map of the sample and presents the resulting image of a small Au crystal with 50 nm resolution.

It is important to understand the effect of a noisy signal on the outcome of the experiment described in Chapter 3, so Chapter 4 presents a simulation of a CXD signal contaminated by different kinds of noise present in a typical CXD experiment. The resilience and behavior of the algorithms on data corrupted by various levels of background and alien scattering is examined for a test image. Additionally, the effect of varying the parameters of the algorithms and mixing them is explored to fascinating result.

Finally, Chapter 5 summarizes the important results of the earlier chapters and provides directions for future work.

1.1 Inverse Problems

Many topics in physics lead to the need to solve an inverse problem[51], that is, the question: Given a measurement at some point, what were the details leading to this result? For example, one could study the direct problem of deriving the intensity that would be measured at some point, $I(P,T)$, given a particular incident wave, $E(\mathbf{r},t)$, and some function describing the real-space density, $\rho(\mathbf{r},t)$, from which the photons scatter. However, the inverse problem–"given $I(P,T)$, find $\rho(\mathbf{r})$"–is often the more relevant one. Problems of this nature arise in diverse fields in the physical and biological sciences: geophysical problems[49]–including water table studies[62, 90], seismology[70, 50], and imaging and prospecting[63, 30]–sonar[30] and radar[14] which are subsets of the more general inverse scattering problem[22] that can also be applied to the study of echolocation in bats[21]–the ubiquitous computerized tomography (CT) scan[47], the analysis of the early difficulties with the optics of the Hubble Space Telescope[35], imaging objects through a turbulent atmosphere[31], the prospect of using X-rays to image quantum dot arrays[96] and the successful imaging of a Carbon nanotube by electron diffraction[101].

Within the X-ray community, such methods have also found a home in solving the phase problem[59]. That is, the scattered amplitude at a point is a complex quantity, $A(P) = |A(P)|e^{i\phi(P)}$, of which only the intensity, *i.e.*, $A^*(P)A(P) = |A(P)|^2$, is known. This measurement obviously results in the loss of all phase information in the scattered wavefield at P. Methods have been developed for solution of the crystallographic phase problem in certain situations[79, 80, 20] and methods exist for microscopy via X-ray holography[88]. This thesis seeks instead to explore the phase problem in the context of a coherent X-ray diffractive imaging experiment.

Several methods have been proposed to solve the general phase problem. These range from methods for finding the zeroes of the complex amplitude[66, 67] to measuring the phase difference introduced into a Bragg diffracted wave by domains within the sample[69]. In the present thesis, we use the properties of a coherent X-ray beam, our sample, and the sampling of the diffracted intensity to make possible the recovery of the diffracted phases through application of an iterative algorithm.

Illuminating a small object with a coherent X-ray beam results in a far-field pattern, as will be shown shortly, which, when sampled in 2D, is related to the Fourier transform of the projection of the density of the object onto a plane perpendicular to the exiting diffracted wavevector. This simple relationship would enable an elementary back-transform to image the object if the lost phase information could be recovered. This is the essence of coherent X-ray diffractive imaging. Sayre, *et al.* was the first to propose this under the name non-crystallographic phase retrieval[81]. Shortly thereafter, several additional papers were presented by describing the fundamental concepts and application to non-crystalline samples. In 2001, the successful recovery[77] of the projection of a 2D single crystal of Au was reported by Robinson, *et al.*. One year later, Miao *et. al*, reported the recovery of a pseudo-3D test object, which consisted of a patterned deposition of Ni on either side of a SiN window[57]. In 2003, Williams, *et al.* published the 3D density map of a single crystal of Au[99]. Many other examples of variations of the basic technique have been reported during this time period, see for example Refs. [42, 87, 55, 91].

In general, an ill-posed inverse problem of this sort will have many solutions, but in practice the application of prior knowledge can reduce the number of solutions and in some cases–for example, CT[10]–uniqueness can be proved. Similarly it has been proposed that in CXD imaging experiments knowledge of the curvature of the wave front at the sample can be combined with the scattering data to render a unique solution[68]. We, however, seek

a general method for retrieving phases from a generic CXD measurement.

1.2 Coherent X-ray Diffraction

The particular situation of interest is the case of diffraction from a finite crystal illuminated with coherent radiation. The first topic is diffraction, tacitly assuming coherent illumination. With the expected result in hand, the effects of the partial coherence of the illumination will be discussed.

In order to describe the expected diffracted amplitude, we must first describe the scatterer. We assume a crystal is coherently illuminated by plane waves of wavelength $\sim$1Å. We take a crystal to be composed of unit cells arranged on a Bravais lattice. This lattice is composed of an infinite number of points regularly spaced so that the lattice appears identical when viewed from any lattice point[8]. The Bravais lattice can be described as an infinite sum of delta functions

$$\rho_L(\mathbf{r}) = \sum_n \delta(\mathbf{r} - \mathbf{R_n}), \tag{1.1}$$

where $\mathbf{R_n}$ is a lattice vector, *i.e.*, a vector from the origin to a lattice point. An important property of such a lattice is that its Fourier transform is also a Bravais lattice. The Bravais lattice described in [1.1] will be referred to as the real-space lattice and its Fourier transform is the reciprocal-space lattice. A lattice vector $\mathbf{R_n}$ can be written as a linear combination of vectors that span the space:

$$\mathbf{R_n} = n_1\mathbf{a} + n_2\mathbf{b} + n_3\mathbf{c}, \tag{1.2}$$

where the vectors $\mathbf{a}$, $\mathbf{b}$, $\mathbf{c}$, are called primitive lattice vectors and the scalar prefactors are integers. There will then be a reciprocal-space lattice vector,

$$\mathbf{H_n} = h\mathbf{a}^* + k\mathbf{b}^* + l\mathbf{c}^*, \tag{1.3}$$

with the primitive reciprocal lattice vectors determined by the vector product of the primitive real lattice vectors, for example:

$$\mathbf{a}^* = 2\pi\frac{\mathbf{b}\times\mathbf{c}}{\mathbf{a}\cdot(\mathbf{b}\times\mathbf{c})}. \tag{1.4}$$

A plane is easily defined by the vector perpendicular to it, so if [1.3] is the shortest vector pointing in a particular direction, a plane described by the coordinates (h,k,l) is called the plane with Miller indices h,k,l. It is usual to abbreviate the coordinates to eliminate the commas and substitute

$\bar{h_0}$ in place of $-h_0$, for example, the vector with coordinates $(1, -1, 1)$ is said to point in the $(1\bar{1}1)$ direction. On the topic of notation, it should also be pointed out that the shorthand $\{hkl\}$ will be used to express the equivalent directions. This equivalence will depend on the Bravais lattice.

In general, a crystal is composed of the smallest unit of regular structure, a unit cell, on each lattice point. Such unit cells may contain single atoms or large numbers of different atoms arranged arbitrarily. Therefore, we think of a crystal as possessing a unit cell, composed of some arrangement of atoms, convolved with a Bravais lattice. This formalism creates an infinite crystal, but a finite crystal of any shape may be expressed as the product of the unit cell convolved with the lattice and a shape function, which is zero valued outside the volume of the desired crystal.

1.2.1 Diffraction

The fundamental scatterer of X-ray radiation is the electron and we follow the typical method to derive the scattering[98, 7]. Classically, we can regard this scattering as an acceleration of the electron leading to radiation[45] as the electron oscillates in the incident X-ray beam. If we consider a collection of electrons to be a continuous charge density, $\rho(\mathbf{r})$, the scattered radiation due to the distribution is the superposition of the radiated fields from each volume element $\rho(\mathbf{r})\mathrm{d}\mathbf{r}$ at the point of observation, P. To simplify the problem, we make the assumption that the incident wave is a plane wave and that each photon in the incident wave interacts with the charge distribution only once. This is often called the kinematical limit, as opposed to the dynamical limit where Maxwell's equations must be solved within and without the distribution. We will also neglect Compton scattering and consider only the elastically scattered photons. In the kinematical limit, the Born approximation holds and the amplitude of incident wave is assumed to be constant; therefore, the quantity of interest is the phase difference between waves scattered from different elements.

As the incident wave interacts with the charge distribution elements at $\mathbf{r}$ and the origin, a path length difference arises in the scattered radiation, which results in the retardation of the phase in one contribution to the total scattered radiation relative to the another. The amount of the phase shift due to the path length difference is $2\pi(\mathbf{k_f} - \mathbf{k_i}) \cdot \mathbf{r}$, where $\mathbf{k_i}$ is the incident wave vector and $\mathbf{k_f}$ is the scattered wavevector, which has the usual relationship to the wavelength, λ, $|k| = 2\pi/\lambda$. It is usual to define a variable $\mathbf{q} = \mathbf{k_f} - \mathbf{k_i}$, sometimes called the momentum transfer. Therefore, to find

the total scattering, a sum over all volume elements is performed:

$$A(\mathbf{q}) = \frac{E_0 q^2}{mc^2 R} e^{i2\pi(\nu t - R/\lambda)} \int \rho(\mathbf{r}) e^{i\mathbf{q}\cdot\mathbf{r}} \mathrm{d}\mathbf{r} = \frac{E_0 q^2}{mc^2 R} e^{i2\pi(\nu t - R/\lambda)} f_0, \qquad (1.5)$$

where E_0 is the amplitude of the incident wave, q is the electron charge, m is the electron mass, c is the speed of light, R is the distance from the origin to point P, and ν and λ are the frequency and wavelength of the incident beam. For brevity, we define $C = \frac{E_0 q^2}{mc^2 R} e^{i2\pi(\nu t - R/\lambda)}$. If the charge distribution is assumed to be an atom, f_0 is called the atomic form factor. We recognize [1.5] as the familiar Fourier transform defined by:

$$f(\mathbf{r}) = \frac{1}{(2\pi)^3} \int \mathrm{d}\mathbf{q} F(\mathbf{q}) e^{-i\mathbf{r}\cdot\mathbf{q}} \qquad (1.6)$$

and its inverse:

$$F(\mathbf{q}) = \int \mathrm{d}\mathbf{r} f(\mathbf{r}) e^{i\mathbf{r}\cdot\mathbf{q}}. \qquad (1.7)$$

Similarly, the form factor of a unit cell is given by

$$F_n(\mathbf{q}) = \sum_{m=1}^{M} \int \rho_{nm}(\mathbf{r}) e^{i\mathbf{q}\cdot\mathbf{r}} \mathrm{d}\mathbf{r}, \qquad (1.8)$$

where the sum is over all M atoms in a unit cell and the integral is over the charge distribution attributed to atom m in unit cell n.

The following approach for converting the the scattering factors from a sum over lattice points to an integral over all space was originally due to Von Laue[95]. We seek to rewrite [1.5] for the case of a finite crystal, so it is important to recognize that in this case the real-space density should be regarded as the product of the finite shape function of the crystal, $s(\mathbf{r})$, with the convolution, denoted by $\otimes$, of the density of a unit cell, $\rho_u(\mathbf{r})$, with the real-space lattice [1.1]. In the present case, we assume that all unit cells are identical so that the $F_n(\mathbf{q})$ may be replaced by $F(\mathbf{q})$, the Fourier transform

of $\rho_u(\mathbf{r})$. The diffracted amplitude from [1.5] is then:

$$\begin{aligned} A(\mathbf{q}) &= C \int s(\mathbf{r})[\rho_L(\mathbf{r}) \otimes \rho_u(\mathbf{r})] e^{i\mathbf{q}\cdot\mathbf{r}} \mathrm{d}\mathbf{r} \\ &= C(S(\mathbf{q}) \otimes [R_L(\mathbf{q})F(\mathbf{q})]) \\ &= CF(\mathbf{q})[S(\mathbf{q}) \otimes R_L(\mathbf{q})] \\ &= \frac{C}{V} F(\mathbf{q})[S(\mathbf{q}) \otimes \sum_n \delta(\mathbf{q} - \mathbf{h_n})] \\ &= \frac{C}{V} F(\mathbf{q}) \sum_n S(\mathbf{q} - \mathbf{h_n}) \end{aligned} \tag{1.9}$$

where V is the volume of a unit cell, $R_L(\mathbf{q})$ is the reciprocal-space lattice, $S(\mathbf{q})$ is the Fourier transform of $s(\mathbf{r})$, and $\mathbf{h_n} = 2\pi\mathbf{H_n}$–with $\mathbf{H_n}$ denoting a reciprocal lattice vector. $F(\mathbf{q})$ has been pulled out of the convolution, which is justified as it is a very slowly varying function of $\mathbf{q}$.

To calculate intensity of the diffracted wave, *i.e.*, the squared modulus of [1.9], we make one further simplification: the amplitude is localized near the Bragg point. This allows us to neglect cross terms in $A^*(\mathbf{q})A(\mathbf{q})$–arising from sum over the reciprocal lattice points–can be neglected giving:

$$\begin{aligned} I(\mathbf{q}) &= A^*(\mathbf{q})A(\mathbf{q}) \\ &= \left|\frac{C}{V}\right|^2 |F(\mathbf{q})|^2 \sum_n |S(\mathbf{q} - \mathbf{h_n})|^2, \end{aligned} \tag{1.10}$$

where $A^*(\mathbf{q})$ is the complex conjugate of $A(\mathbf{q})$. In a CXD experiment, typically only a small section of reciprocal space is measured, so we define $\mathbf{Q} = \mathbf{q} - \mathbf{h_n}$ and write the intensity near a Bragg peak(a reciprocal lattice point) as:

$$\begin{aligned} I(\mathbf{Q}) &= A^*(\mathbf{Q})A(\mathbf{Q}) \\ &= \left|\frac{C}{V}\right|^2 |F(\mathbf{h_n})|^2 |S(\mathbf{Q})|^2 \\ &= \left|\frac{C}{V}\right|^2 |F(\mathbf{h_n})|^2 \left| \int s(\mathbf{r}) e^{i\mathbf{Q}\cdot\mathbf{r}} \mathrm{d}\mathbf{r} \right|^2. \end{aligned} \tag{1.11}$$

This result clearly shows that a diffraction pattern measured from a small coherently illuminated crystal in the vicinity of a Bragg peak will contain the Fourier transform of its shape. Since this quantity is squared, we can make no direct measurement of the phase of the complex amplitude, so recovering the shape function is an inverse problem.

According to [1.9], the diffracted intensity is periodic about every Bragg point, $\mathbf{h_n}$. If $s(\mathbf{r})$ is purely real and neglecting the scaling factor $|F(\mathbf{h_n})|$,

the intensity distribution about every Bragg point will be exactly the same. Further, since the intensity depends only on the Fourier transform of the shape, the pattern will be locally symmetric about each Bragg point, since the Fourier transform of a real object is centrosymmetric.

The shape function, $s(\mathbf{r})$, can be a complex function, $s(\mathbf{r}) = |s(\mathbf{r})|e^{i\phi}$. Typically, the phase angle ϕ contains information about the strain within the crystal. Since strain can be represented as a displacement, $\mathbf{u}(\mathbf{r})$, from the real-space lattice its contribution to the diffraction is a phase factor based on the path length difference $\mathbf{u}(\mathbf{r})\cdot\mathbf{q}$, *i.e.*, $s(\mathbf{r}) = |s(\mathbf{r})|e^{i\mathbf{u}(\mathbf{r})\cdot\mathbf{q}}$. Therefore, strain within a crystal will be immediately obvious from the lack of symmetry in its diffraction pattern. The problem of recovering the strain map of a diffracting density is discussed in Ref. [76].

If more than one particle is coherently illuminated, the resulting intensity will be the square of the coherent superposition of the Fourier transforms of the shapes. If, on the other hand, the two are not coherently illuminated, the intensities due to each particle will add. The latter case is the usual powder diffraction experiment.

In practice, a CXD pattern may be collected by means of a 2D CCD detector; therefore, it is desirable to understand the physical meaning of such a 2D slice through the intensity distribution. In order to measure the diffraction, the detector plane is oriented perpendicular to the scattering vector $\mathbf{k_f}$. We choose our axes such that $\mathbf{x}$ and $\mathbf{y}$ lie in the plane of the detector and $\mathbf{z}$ is in the direction of $\mathbf{k_f}$. Then, the distribution collected in the 2D plane of the detector located at Q_z^0 along the z-axis is:

$$\begin{aligned} I(Q_x, Q_y, Q_z^0) &= \left|\frac{C}{V}\right|^2 |F(\mathbf{h_n})|^2 |S(\mathbf{Q})|^2 \\ &= \left|\frac{C}{V}\right|^2 |F(\mathbf{h_n})|^2 \left| \int \mathrm{d}x\, \mathrm{d}y\, s_z(x, y, Q_z^0) e^{iQ_x x + iQ_y y} \right|^2 \end{aligned} \tag{1.12}$$

where

$$s_z(x, y, Q_z^0) = \int \mathrm{d}z\, s(x, y, z) e^{iQ_z^0 z} \tag{1.13}$$

Therefore, a 2D slice through the CXD pattern is related to the Fourier transform of the projection of the real-space density onto the x-y plane.

1.2.2 Coherence length

In this section, we will briefly treat the problem of partial coherence from a classical optics perspective[15]. A complete version of this theory has been extended to X-ray radiation in Ref. [85] and this particular application has

is addressed in detail in Ref. [95]. We will begin by connecting the previous section to classical optics and show that the effect of partially coherent illumination is that the resulting diffracted intensity is the product of the real-space autocorrelation function and a function describing the coherence of the incident beam. For this derivation, it is normally assumed that the incoming radiation is quasimonochromatic, *i.e.*, there is some mean wavelength, $\bar{\lambda}$, and the deviation from the mean is very small compared to the mean, $\Delta\lambda/\bar{\lambda} \ll 1$. This carries over into the other quantities of interest, particularly, $|\bar{\mathbf{k}}| = 2\pi/\bar{\lambda}$, which leads to a slight alteration in our previous definition of $\mathbf{q}$: $\mathbf{q} = \bar{\mathbf{k}}_\mathbf{f} - \bar{\mathbf{k}}_\mathbf{i}$. Two further assumptions are typically made. First, that the wavefield is stationary, in other words, that averages over time are not sensitive to the origin. Second, that the ensemble of waves is ergodic, so that the time average of a typical member of the ensemble is the same as the ensemble average.

Huygens-Fresnel diffraction theory can be used to derive the diffracted fields of the previous section rigorously arriving at the scattered amplitude:

$$A(\mathbf{q}) = \int \mathrm{d}\mathbf{r}\rho(\mathbf{r})A_{in}(\mathbf{r}, t - \tau_r)e^{i\mathbf{q}\cdot\mathbf{r}} \tag{1.14}$$

where $A_{in}(\mathbf{r}, t - \tau_r)$ is the incident wave, and τ_r is the propagation delay between $\mathbf{r}$ and some point on the detector plane, $\tau_r = l_d/c$. For simplicity, we consider only one polarization of the incident wave. This leads to a periodic amplitude[95] analogous to [1.9]:

$$A(\mathbf{q}) = \frac{C}{V}F(\mathbf{q})\sum_n A_n(\mathbf{q} - \mathbf{h}_\mathbf{n}, t) \tag{1.15}$$

with the definition

$$A_n(\mathbf{q}, t) = \int \mathrm{d}\mathbf{r}s(\mathbf{r})A_{in}(\mathbf{r}, t - \tau_r)e^{i\mathbf{q}\cdot\mathbf{r}} \tag{1.16}$$

where $s(\mathbf{r})$ is assumed to be a purely real shape function for simplicity.

Now, the expected intensity near the jth Bragg point, in analogy to [1.11] is:

$$\begin{aligned} I(\mathbf{Q}) &= \langle A(\mathbf{Q}, t), A^*(\mathbf{Q}, t)\rangle \\ &= \left|\frac{C}{V}\right|^2 |F(\mathbf{h}_\mathbf{j})|^2 \, |A_j(\mathbf{Q}, t)|^2 \\ &= \left|\frac{C}{V}\right|^2 |F(\mathbf{h}_\mathbf{j})|^2 \int\int \mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}' s(\mathbf{r})s(\mathbf{r}')\Gamma_{in}(\mathbf{r}, \mathbf{r}', \Delta\tau)e^{i\mathbf{Q}\cdot(\mathbf{r}-\mathbf{r}')} \end{aligned} \tag{1.17}$$

where $\Delta\tau = (l_r - l_{r'})/c$ is a time delay and the mutual coherence function is defined by

$$\Gamma_{in} = \sqrt{I(\mathbf{r})}\sqrt{I(\mathbf{r'})}\gamma_{in}(\mathbf{r},\mathbf{r'})W(\tau) \tag{1.18}$$

where $I(\mathbf{r}) = \langle A_{in}(\mathbf{r},t)A^*_{in}(\mathbf{r},t)\rangle$ is the incoming intensity at $\mathbf{r}$, γ_{in} is the normalized mutual coherence function–called the complex degree of coherence– and $W(\tau)$ is the time autocorrelation function. In the case of uniform illumination, where $I(\mathbf{r}) = I(\mathbf{r'})$, we have a very simple connection to the visibility–$\mathcal{V}$, the difference in intensity between the minima and the maxima of the fringes in the diffraction: $\mathcal{V} = W(\tau)\gamma_{in}(\mathbf{r},\mathbf{r'})$. So, a perfectly coherent source would result in $\mathcal{V} = 1$, while a perfectly incoherent source would give $\mathcal{V} = 0$.

In the paraxial approximation, γ_{in} can be expressed as an integral over the source, denoted by d$\mathbf{s}$:

$$\gamma_{in}(\mathbf{r},\mathbf{r'}) = \frac{e^{i\frac{k}{2L}(r^2-r'^2)}\int \mathrm{d}\mathbf{s} I(\mathbf{s})e^{i(k/L)(\mathbf{r}-\mathbf{r'})\cdot\mathbf{s}}}{\int \mathrm{d}\mathbf{s} I(\mathbf{s})} \tag{1.19}$$

where L is the source to sample distance. [1.19] is the van Cittert-Zernike Theorem, which relates the intensity distribution of the extended source to the mutual coherence function. In the far field limit, the intensity at the sample is approximately constant over small regions, *i.e.*, $I(\mathbf{r}) \simeq I(\mathbf{r'})$ and can be written as:

$$I(\mathbf{r}) = I_0 = \left(\frac{\bar{\lambda}}{L}\right)^2 \int \mathrm{d}\mathbf{s} I(\mathbf{s}). \tag{1.20}$$

A synchrotron source is typically modeled as having a Gaussian intensity distribution:

$$I(s_x, s_y) = \frac{I_0}{2\pi\sigma_x\sigma_y} e^{-\frac{1}{2}\left(s_x^2/\sigma_x^2 + s_y^2/\sigma_y^2\right)} \tag{1.21}$$

introducing σ_x and σ_y as the Gaussian half widths in the x and y directions. Since the mutual coherence function is simply a normalized Fourier transform multiplied by some phase function and recognizing that in the far field limit $\bar{k}(r^2 - r'^2)/L$ approaches zero, [1.19] tells us that the mutual intensity function is the normalized Fourier transform of the source, [1.21]:

$$\gamma_{in}(\mathbf{r}_\perp - \mathbf{r'}_\perp) = e^{-\frac{(x-x')^2}{2\eta_x^2} - \frac{(y-y')^2}{2\eta_y^2}} \tag{1.22}$$

$\mathbf{r}_\perp$ has been introduced to indicate that [1.22] is relevant only for the directions orthogonal to the propagation direction. From this Gaussian distribu-

tion, the transverse correlation lengths are then:

$$\eta_\perp = \frac{L}{\bar{k}\sigma_\perp}. \tag{1.23}$$

The coherence length in the third direction depends on $W(\tau)$, the time autocorrelation function. For synchrotron sources, whose power spectral density is assumed to be Lorentzian [95, 85], this function is an exponential:

$$W(\tau) = W_0 e^{-\tau/\tau_\parallel} \tag{1.24}$$

and the correlation length is defined $\eta_\parallel = c\tau_\parallel$. This correlation length is determined by the bandwidth of incident radiation[95]:

$$\eta_\parallel = \frac{2\bar{\lambda}}{\pi}\frac{\bar{\lambda}}{\Delta\lambda} \tag{1.25}$$

which is typically determined by the monochromator in an experiment. We can also write $W(\tau)$ as a function of the real-space variable $\mathbf{r}$:

$$W(\Delta\tau) = W_0 e^{-c\Delta\tau/c\tau_\parallel} = W_0 e^{-|l_r - l_{r'}|/\eta_\parallel} = W_0 e^{-|r_\parallel - r'_\parallel|/\eta_\parallel} = W(|r_\parallel - r'_\parallel|) \tag{1.26}$$

Now, putting all this back into [1.17], the intensity near the jth Bragg point is

$$\begin{aligned} I(\mathbf{Q}) &= \langle A(\mathbf{Q},t), A^*(\mathbf{Q},t)\rangle \\ &= \left|\frac{C}{V}\right|^2 |F(\mathbf{h_j})|^2 \, |A_j(\mathbf{Q},t)|^2 \\ &= \left|\frac{CF(\mathbf{h_j})}{V}\right|^2 \times \\ &\qquad \int\int \mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}' s(\mathbf{r}) s(\mathbf{r}') \gamma_{in}(\mathbf{r}_\perp - \mathbf{r}'_\perp) W(|\mathbf{r}_\parallel - \mathbf{r}'_\parallel|) e^{i\mathbf{Q}\cdot(\mathbf{r}-\mathbf{r}')} \\ &= \left|\frac{CF(\mathbf{h_j})}{V}\right|^2 \int \mathrm{d}\mathbf{r} \phi(\mathbf{r}) \gamma_{in}(\mathbf{r}_\perp) W(|\mathbf{r}_\parallel|) e^{i\mathbf{Q}\cdot(\mathbf{r})} \end{aligned} \tag{1.27}$$

where the last step is accomplished through a change of variable and the definition:

$$\phi(\mathbf{r}) = \int \mathrm{d}\mathbf{r}' s(\mathbf{r}') s(\mathbf{r}' + \mathbf{r}), \tag{1.28}$$

recognized as the autocorrelation of the shape function.

For a perfectly coherent source, $\gamma_{in}(\mathbf{r}_\perp) = 1$ and $W(|\mathbf{r}_\parallel|) = 1$, and the

coherent intensity from [1.27] is:

$$I^{coh}(\mathbf{Q}) = \left|\frac{CF(\mathbf{h_j})}{V}\right|^2 \int \mathrm{d}\mathbf{r}\phi(\mathbf{r})e^{i\mathbf{Q}\cdot(\mathbf{r})}, \tag{1.29}$$

the Fourier transform of the real-space autocorrelation function. This is precisely the expected result in the coherent limit. Now, [1.27] can rewritten using the Convolution Theorem to give:

$$I(\mathbf{Q}) = \int \mathrm{d}\mathbf{Q} I^{coh}(\mathbf{Q})\Gamma(\mathbf{Q}-\mathbf{Q}'), \tag{1.30}$$

where $\Gamma(\mathbf{Q})$ is the Fourier transform of $\gamma_{in}(\mathbf{r}_\perp)W(|\mathbf{r}_\parallel|)$. Thus, the diffracted intensity is a convolution of expected intensity from a completely coherent measurement and the Fourier transform of the mutual coherence function. This means that the coherently diffracted intensity might be recovered if it could be deconvolved from the mutual intensity of the beam.

In a CXD experiment, the quantity measured is a 2D slice through the 3D intensity distribution:

$$I(Q_x, Q_y, Q_z^0) = \left|\frac{CF(\mathbf{h_j})}{V}\right|^2 \int \mathrm{d}\mathbf{r}\phi^z(\mathbf{r})\gamma_{in}(x,y)e^{i(Q_x x + Q_y y)}, \tag{1.31}$$

with the definition:

$$\phi^z(x,y) = \int \mathrm{d}z\phi(\mathbf{r})e^{iQ_z^0 z}e^{-|z|/\eta_\parallel} \tag{1.32}$$

using [1.24] and [1.26]. So, if $\eta_\parallel \gg D$, the size of the crystal, the entire crystal density adds coherently to form the projection of the autocorrelation, *i.e.*, $e^{-|z|/\eta_\parallel} \sim 1$. On the the other hand, if $\eta_\parallel \geq D$, only a portion of the crystal along z will add coherently. Therefore, if a partial coherence effect is present, it will have arrived in the reconstruction through a convolution, [1.30], with the coherent diffraction from the real-space object, which means our reconstruction will be based on the Fourier transform of the product of γ_{in} with the autocorrelation rather than than the autocorrelation alone, [1.32].

1.3 Phase Retrieval from Intensity Measurement

Iterative phase retrieval works because, in the general case, there are very few sets of phases[17, 11, 78, 9] that can be matched to the measured amplitudes when the real-space function has compact support. Successful recon-

struction depends on many things including the sampling of the data and the real-space constraints derived from *a priori* knowledge of the sample.

In this section, the first topic derives the requisite data sampling rate and its interpretation in real-space. Having established how finely we must sample the intensity, a literature review regarding the uniqueness of phase sets consistent with the measured intensity is presented.

1.3.1 Oversampling

In general, the support of a function $f(x, y)$ is the region of real-space where $f(x, y)$ is non-zero; however, for generality, we redefine the support as the smallest rectangular region containing all the non-zero real-space density, as in [61]. Then, $f(x, y)$ has support of total area $a \times b$:

$$S[f(x, y)] = (-a/2, a/2) \times (-b/2, b/2). \tag{1.33}$$

From the convolution theorem–or equivalently, assuming perfect coherence and dropping prefactors in [1.27]–the measured intensity is the autocorrelation of $f(x, y)$:

$$\phi(x, y) = \int\int \mathrm{d}x'\mathrm{d}y' f(x', y') f(x - x', y - y'). \tag{1.34}$$

Thus, the quantity measured has real-space support:

$$S[\phi(x, y)] = (-a, a) \times (-b, b), \tag{1.35}$$

which is twice the support in each dimension required for $f(x, y)$, from [1.33], the quantity we seek to recover by phasing the modulus of the amplitude, *i.e.*, the square root of the measured intensity. Formally, we need to be able to analytically continue the intensity $|F(m, n)|$ onto the complex plane, which requires us to know the function continuously. To satisfy this condition, it is necessary to sample the function at the Nyquist frequency, in accordance with Shannon's Theorem. This sets the spatial sampling frequency at $1/2a$ in the x direction and $1/2b$ in the y direction.

In a similar way, this rate may be derived from regarding the measured intensity as a system of linear equations[79, 58], one for each pixel in the detector. Then, if N pixels are used, we have N equations, but since we can only measure the magnitude of the amplitude and not the phase, we have $2N$ unknowns. In order to rectify this, Fourier space must be sampled finely enough to reduce the real-space image to a region of one half the total

array, this corresponds to sampling the intensity at the Nyquist rate, since the autocorrelation is twice as large as the object itself. This is a result of Sayre's 1952 letter on the application of Shannon's theorem to measuring structure factors[79] and is equivalent to the previous statement for a 1D argument.

In either case, we define an oversampling ratio in real-space:

$$\sigma = \frac{\text{Size of Array}}{\text{Support of Object}}. \tag{1.36}$$

This is related to the number of points per oscillation in the Fourier transform of the real-space density, so it is easily estimated from experimental data. A number of studies have been performed[61, 58] on the lower bound of σ in the case of a successful reconstruction. The equation counting argument gives $\sigma = 2$ for any dimension, and we use this as the minimum acceptable value in judging experimental data. Other studies have revealed that the oversampling ratio depends on the dimensionality of the problem[61]: the oversampling ratio should be 2^{3-N}, where N is the dimensionality of the problem.

1.3.2 Uniqueness

Since direct methods will not work, we must rely on indirect methods which will be discussed at length in Chapter 2. However, when one relies on indirect methods, the question of uniqueness of solutions is of primary interest. There are three trivial ambiguous solutions for any real-space function, $f(z)$: $f^*(-z)$, $f(z + z_0)$, and $e^{i\phi}f(z)$. These three functions all have the same Fourier modulus, $|F(k)|$, and therefore are solutions. They are trivial since they represent changes in the position or orientation of the real-space object, but do not modify the shape or other physical characteristics. We regard a solution that has only these three ambiguous solutions as unique. In contrast, if we take two functions of with compact support, $f(z)$ and $g(z)$, then their convolution is

$$h(z) = f(z) \otimes g(z) \tag{1.37}$$

and we may compose a second function

$$h'(z) = f(z) \otimes g^*(-z). \tag{1.38}$$

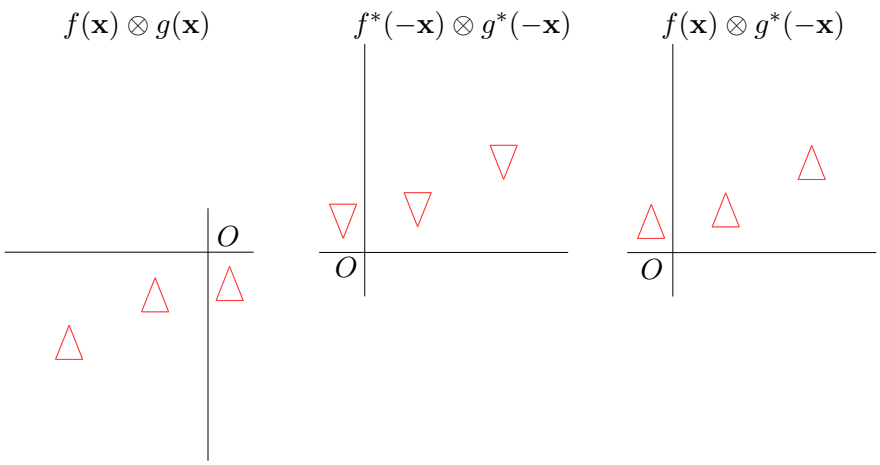


Figure 1.1: All three examples have the same Fourier modulus, demonstrating that ambiguous solutions are possible in 2D. $f(\mathbf{x})$ is the shape of the triangle. $g(\mathbf{x}) = \delta(\mathbf{x} - \mathbf{x_1}) + \delta(\mathbf{x} - \mathbf{x_2}) + \delta(\mathbf{x} - \mathbf{x_3})$, where the vectors $\mathbf{x_i}$ are from the origin to the center of each triangle. The leftmost two are the "correct" solution and its twin–a trivial ambiguity–while the third demonstrates that if $|F(\mathbf{k})||G(\mathbf{k})|$ can be factorized an ambiguous solution may exist.

Now, by the convolution theorem we can write:

$$|H(k)| = |F(k)||G(k)| = |F(k)||H^*(k)|. \tag{1.39}$$

Since $g(z)$ is not necessarily equal to $g^*(-z)$, [1.37] and [1.38] represent two real-space functions with exactly the same Fourier magnitude. In fact, it has been shown that this factorizability is a necessary condition for multiple solutions to exist[9].

Historically, the recovery of phases from a Fourier magnitude measurement was regarded as unlikely[97], as it was well known that in 1D any function can be expressed via Hadamard's product[9]:

$$F(k) = \prod_n a_n (1 - \frac{k}{k_n}). \tag{1.40}$$

This is a manifestation of the general result that any function can be expressed as an polynomial and that polynomials of a single variable with degree 2 or higher can be factorized in the complex plane. Therefore, the general 1D Fourier magnitude problem is not expected to possess a unique solution. Nevertheless, such problems have been solved in X-ray measure-

ments by using prior knowledge to eliminate unphysical solutions[94].

By the late 1970's and early 1980's, various methods of recovering complex amplitude from intensity measurements were being successfully applied to 2D data. These results[32, 84] appeared not to suffer from the non-uniqueness of the 1D case. By this time, the theory of functions of two complex variables was better understood, and it was known that while a polynomial of one complex variable was always factorisable, a polynomial of two or more such variables was generally not. In fact, in two and higher dimensions, factorisable polynomials occur with probability zero. Since this factorizability is a necessary condition, non-unique solutions to the phase retrieval problem should occur with probability no higher than zero. Unfortunately, the case elucidated in [1.37]-[1.39] still applies–with the trivial change to functions of two or more variables–so while non-unique solutions occur with probability zero, there are still many non-unique solutions if the magnitude of the amplitude can be factorized.

An example of this is demonstrated in Fig 1.1. In this case, the real-space function is taken to be the convolution of a shape with a sum of delta functions. The result of flipping the sign of the variable and conjugating the sum before convolution is shown to be a fundamentally different function from the original convolution and one of the trivial ambiguities: its twin.

A natural way to further reduce the non-uniqueness of solutions is to permit only those which satisfy, for example, a finite support constraint. In fact, it was shown in 1983 that 1D problem can be reduced to one which "almost always" has a unique solution if the real-space object is compatible with the use of a support with multiple unconnected regions[24]. Nieto-Vesperinas and Dainty[65] showed that some 2D objects could be related to 1D polynomials with integer coefficients, allowing the invocation of Eisenstein's irreproducibility theorem. This allows for unique solutions, for example, in the case of a regular array when a reference point is available. Similarly, Brames[16] showed that when the convex hull–the smallest convex set (See Fig. 2.5 for examples.) containing all points in the real-space object–has no parallel sides, a unique solution is guaranteed.

Unfortunately, neither of these situations is one in which a CXD experimenter usually finds herself. Further, the measured intensity usually has two shortcomings: it is sampled on a finite regular grid and it contains noise. It is possible, if unlikely, that two different analytic functions will possess the same sampled intensity. In which case, the multiple solutions will either be physical, *i.e.*, they are consistent with the *a priori* knowledge of the sample, or they will be unphysical. Unphysical solutions will be rejected, or at least

result in very high error metrics, by the methods discussed in Chapter 2. Multiple physical solutions should appear with equal probability, since the methods all begin with random starting points. Therefore, performing many fits with different starting points should warn the alert experimenter of the non-uniqueness of the problem at hand. The remaining difficulty is the size of the space measured. In essence, it is this that determines the resolution of the real-space reconstruction. It is unavoidable that any measurement with finite resolution will have ambiguous solutions that differ on scales smaller than the resolution, since no information about those length scales is ever measured.

Noise, unfortunately, is more subtle. The presence of noise in $|H(k)|$ means that the necessary condition for non-unique solutions is changed to: $|H(q)| + \epsilon(q)$ is factorisable, where $\epsilon(q)$ is the uncertainty in the measurement made at q. In other words, the presence of noise may greatly increase the number of ambiguous solutions to the phase retrieval problem. Again, unphysical solutions will be be rejected, as they violate the real-space constraints. This did not escape the notice of researchers in the early 1980's who studied the behavior of the algorithms in cases where no unique solution exists[43, 92]. The possibility of multiple physical solutions existing is explored at length via simulations in Chapter 4.

From this foundation of previous work from allied fields, we now move on to the iterative methods that will be used and examined in the remainder of the thesis.

Chapter 2

Algorithms

Having established the theoretical ground work for the existence and rarity of solutions to the phase retrieval inverse problem in Chapter 1, we now turn to the methods whereby a solution may be found. In general, interest in these algorithms began in the mathematics and electrical engineering community[64], but did not gain wide appeal in the physical sciences until they were applied by Gerchberg and Saxton[37] to electron diffraction and imaging data. From there, interest shifted to their utility in imaging through atmosphere and diagnosing aberrations in telescopes[32]. Many researchers contributed to the mathematical underpinnings of the algorithms in subsequent years[17, 11, 39]. The algorithms presented here operate by successively applying real- and reciprocal-space constraints to an iterate. The algorithms will be presented chronologically and then described in the mathematical context of projection operators in a feasibility problem[23], which will aid in the understanding of the differences in convergence properties among the algorithms.

2.1 Gerchberg and Saxton

One of the earliest phase retrieval methods was presented by Gerchberg and Saxton in 1972[37] and came to be known as the Gerchberg-Saxton (GS) algorithm. This was independent of the work of Lomont and Moses[64] who proposed a theorem to this effect in 1963. The motivation of Gerchberg and Saxton was to find a method whereby the phase of a wave function may be recovered from simultaneous measurements in the image and diffraction planes, as explained below. Since the amplitudes in these planes are related by a Fourier transform, a coherent illuminating wave is required. The extension of this method to X-ray crystallography was suggested by Gerchberg and Saxton, but the technical difficulties involved in making an X-ray mea-

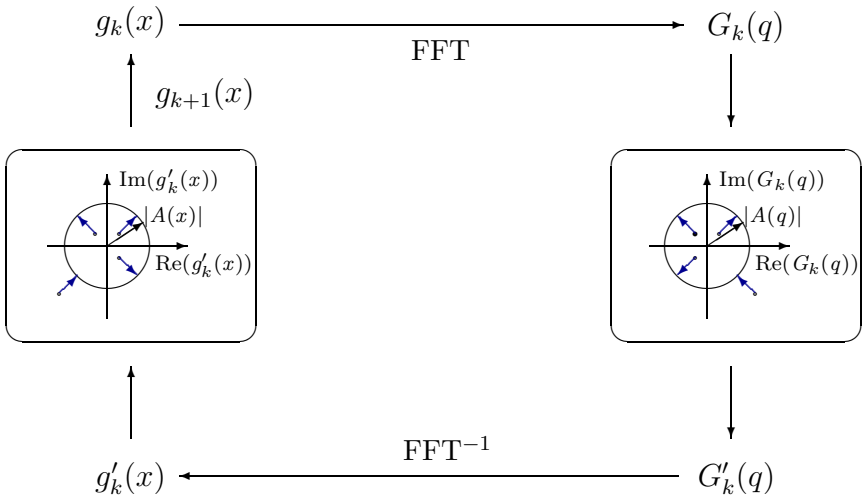


Figure 2.1: Gerchberg-Saxton Algorithm.

surement in the image plane limited its application to electron microscopy.

In the GS algorithm, we have two known constraints: the magnitudes of the amplitudes in the diffraction and image planes. An implementation of the algorithm would be the following: i) apply random phases in the interval $[-\pi, \pi]$ to the measured amplitude in the image plane, ii) perform a FFT on the image, iii) replace the magnitude of the diffracted amplitude thus acquired by the measured diffraction data, iv) perform a FFT on the diffracted amplitude to yield an image, and v) replace the magnitude of the amplitude thus acquired with the measured image data. Steps ii-v are then iterated until an acceptable solution is found. This is demonstrated schematically in Fig. 2.1.

The error in the reconstructed phase is determined by taking the squared difference between the reconstructed and measured amplitudes in either plane. One property of the GS algorithm is that this quantity decreases or remains the same on each subsequent iteration. A proof of this is presented in the next section which describes a generalization of this algorithm to alleviate the requirement of a measurement in the image plane.

2.2 Error Reduction

2.2.1 Description

In 1978 Fienup[32] suggested an extension of the Gerchberg-Saxton algorithm–sometimes called the generalized GS algorithm–to the situation where an intensity measurement is made only in reciprocal-space. In the place of the

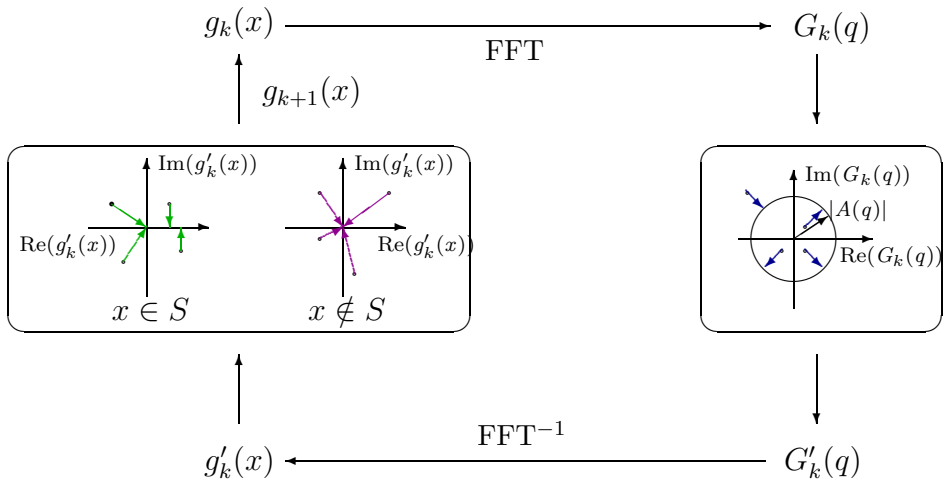


Figure 2.2: Error Reduction Algorithm.

real-space intensity measurement, he suggested the use of a more general real-space constraint. The two most common constraints are positivity and finite support. The positivity constraint is simply that the real-space density should be non-negative. In the case of x-ray diffraction from a perfect single crystal, this is physically expected. The finite support constraint is merely the requirement that the real-space density occupies a confined region within the volume of real-space measured by the diffraction data. The oversampling requirement discussed in Section 1.3.1 limits the volume of the support to less than half the total volume measured. It was initially believed that both finite support and positivity were required to find a solution to this problem, until Fienup showed[34] that a sufficiently tight support constraint without positivity will allow the reconstruction of a complex valued real-space density. Like the original GS algorithm, the reciprocal-space constraint is the measured intensity data.

Fienup's iterative method consists of four steps: i) Fourier transform an estimate of the real space density–g_k–ii) make the smallest change possible to G_k, the Fourier transform of g_k, to satisfy the Fourier modulus constraint, iii) back transform the resulting estimate, G'_k of the actual diffracted amplitude, iv) make the smallest changes possible to the calculated real-space density, g'_k so that it obeys the real-space constraint to arrive at an estimate of the real-space density g_{k+1}. This cycle is demonstrated schematically in Fig. 2.2.

2.2.2 Error metric and behavior

We must now switch from the continuous Fourier transform used in Chapter 1 to the discrete Fourier transform:

$$f(n) = DFT(F(m)) = \frac{1}{N}\sum_{m=1}^{N} F(m)e^{2\pi inm/N} \tag{2.1}$$

where f is a real-space function, F is its Fourier transform, and the discrete array occupies N points. The inverse transform is:

$$F(m) = DFT^{-1}(f(n)) = \sum_{m=1}^{N} f(n)e^{-2\pi inm/N}. \tag{2.2}$$

Further, we must recall the important result from Parseval's theorem(also called the Rayleigh energy theorem):

$$\sum_{n=1}^{N} |f(n)|^2 = \frac{1}{N}\sum_{m=1}^{N} |F(m)|^2, \tag{2.3}$$

which assures us that the integrated "energy" in the two discrete arrays are the same.

To address the behavior of this algorithm, it is necessary to introduce two quantities to measure the error in the reconstruction:

$$E_{R,k}^2 = \sum_{i=1}^{N} \left|g_{k+1}(x) - g_k'(x)\right|^2 \tag{2.4}$$

$$E_{F,k}^2 = \frac{1}{N}\sum_{i=1}^{N} \left|G_k(w) - G_k'(w)\right|^2. \tag{2.5}$$

These represent the squared error in real and reciprocal space, respectively, measuring the amount by which the estimate deviates from the correct solution. Eqn [2.5] can be related to real-space iterates by means of Parseval's Theorem:

$$\begin{aligned} E_{F,k}^2 &= \frac{1}{N}\sum_{i=1}^{N} \left|G_k(w) - G_k'(w)\right|^2 \\ &= \sum_{i=1}^{N} \left|g_k(x) - g_k'(x)\right|^2 \end{aligned} \tag{2.6}$$

Recalling the real-space constraint, g_{k+1} is generated by:

$$g_{k+1} = \begin{cases} g'_k & \text{if } x \in S \\ 0 & \text{if } x \notin S \end{cases}. \tag{2.7}$$

Therefore, if $x \in S$ then $|g_{k+1} - g'_k| = |g'_k - g'_k| = 0$ and if $x \notin S$ then $|g_{k+1}| = |g_k| = 0$, so

$$\left|g_{k+1} - g'_k\right| \leq \left|g_k - g'_k\right|, \tag{2.8}$$

which restates that $g_{k+1}(x)$ is the closest value to g'_k satisfying the real-space constraints. Then, with the relationship [2.8], we have from [2.4] and [2.6]:

$$E^2_{R,k} \leq E^2_{F,k}. \tag{2.9}$$

Now, applying Parseval's theorem to [2.5] we can write:

$$\begin{aligned} E^2_{R,k} &= \sum_{i=1}^{N} \left|g_{k+1}(x) - g'_k(x)\right|^2 \\ &= \frac{1}{N} \sum_{i=1}^{N} \left|G_{k+1}(w) - G'_k(w)\right|^2. \end{aligned} \tag{2.10}$$

Analogously to the real-space case, $G_{k+1}(w)$ and $G'_k(w)$ satisfy the reciprocal space constraint and $G'_{k+1}(w)$ is defined to be the closest point to $G_{k+1}(w)$, which satisfies the reciprocal-space constraint, we arrive at:

$$\left|G_{k+1}(w) - G'_{k+1}(w)\right| \leq \left|G_{k+1}(w) - G'_k(w)\right|., \tag{2.11}$$

which in combination with [2.5] and [2.10] gives

$$E^2_{F,k+1} \leq E^2_{R,k}. \tag{2.12}$$

Combining [2.9] and [2.12] brings us to the final result:

$$E_{F,k+1} \leq E^2_{R,k} \leq E^2_{F,k}. \tag{2.13}$$

This assures us that the error metric of subsequent iterates can be no worse than that of the current iterate, inspiring the alternate name of the algorithm: Error Reduction (ER).

In fact, ER can be regarded as a double length step gradient search algorithm[33]. To illustrate this, we take [2.5] as the function to be minimized, B. The standard procedure for the method of steepest descents calls

for the calculation of the N partial derivatives of B with respect to $g_k(x)$. Once this gradient has been calculated, a step opposite the gradient is taken, arriving at the new location $g''_k(x)$. Finally a new iterate, $g_{k+1}(x)$ would be generated from the real-space support constraint, [2.7]. Assuming $g(x)$ is real,

$$\delta_g B = \frac{\delta B}{\delta g(x)} = \frac{2}{N} \sum_w [|G(w)| - |F(w)|] \frac{\delta |G(w)|}{\delta g(x)}, \tag{2.14}$$

where the definition of B, from [2.5], is used. Using the fact that $G(w)$ is the Fourier transform of $g(x)$, we can evaluate $\frac{\delta G(w)}{\delta g(x)}$ and rewrite:

$$\begin{aligned} \delta_g B_k = & \frac{1}{N} \sum_w [G(w) - |F(w)| G(w)/|G(w)|] e^{i2\pi wx/N} \\ & + \frac{1}{N} \sum_w [G^*(w) - |F(w)| G^*(w)/|G(w)|] e^{-i2\pi wx/N} \end{aligned} \tag{2.15}$$

Recalling the definition of $G(w)$ and $F(w)$, we see that

$$G'(w) = |F(w)| G(w)/|G(w)|, \tag{2.16}$$

and we have

$$\delta_g B = \frac{1}{N} \sum_w [G(w) - G'(w)] e^{i2\pi wx/N} \quad = 2[g(x) - g'(x)]. \tag{2.17}$$

Expanding B about the point $g_k(x)$, we find the step size–determined by the first order term–to be

$$g''_k(x) - g_k(x) = \frac{1}{2} [g'_k(x) - g_k(x)], \tag{2.18}$$

which is one half the step size called for in ER; however, since B is quadratic in $g(x)$, the correct step size would be twice that given by [2.18]. Thus, we see that $g''_k(x) = g'_k(x)$, so after application of the support constraint, ER is seen to be an implementation of the method of steepest descents. This sheds light on a very important aspect of ER: its tendency to stagnate in the presence of a local minima of the error metric.

2.3 Input/Output Methods

The slow convergence and tendency to stagnate of the error reduction algorithm has led to the exploration of other algorithms. Among these are a class of algorithms called input/output algorithms[32]. In this sort of al-

gorithm, the first three steps of the ER algorithm are maintained, but the real-space constraint is enforced in a different way. To understand this, we regard these first three steps–Fourier transform to reciprocal-space, enforce Fourier modulus constraints, and back transform–as a single non-linear operation which takes some function $g_k(x)$ and returns a function $g'_k(x)$. Now, we no longer think of $g_k(x)$ as a solution, but rather as a driving function to produce the solution $g'_k(x)$. That is, we hope to make some small change to the input of this non-linear operation to change, or drive, the output to produce the desired result.

A useful property of this particular operation is that a small change typically results in a small change in the same direction of the corresponding output[33]. Thus we expect that, a small change in $g_k(x)$ will result in a change of the output of some constant, α, times the change in the input. In other words, if we choose $g_k(x) + \Delta g_k(x)$ as the input, we expect to get an output $g'_k(x) + \alpha\Delta g_k(x)$. Unfortunately since this is a nonlinear operation, there are addition terms present that prevent one from accurately predicting the exact change in $g'_k(x)$ from some small change in $g_k(x)$. Nevertheless, a logical choice for the change of the input that will produce the desired change of the output becomes $\beta\Delta g_k(x)$, where the optimal value of β would be α^{-1}. Unfortunately, α is not generally known, so in practice one must vary β to find the optimal value.

2.3.1 Hybrid Input/Output

For the problem discussed here, *i.e.*, a single intensity measurement, the desired change in the output would be:

$$\Delta g_k(x) = \begin{cases} 0 & \text{if } x \in S \\ -g'_k(x) & \text{if } x \notin S \end{cases}, \tag{2.19}$$

where S is the set of points where $g'_k(x)$ violates the real-space constraints. So if the constraints are satisfied, we do nothing; but if they are not, we change $g'_k(x)$ by the amount by which they deviate from the constraints. This leads to a choice of driving function creation called the basic input/output algorithm:

$$\begin{aligned} g_{k+1}(x) &= g_k(x) + \beta\Delta g_k(x) \\ &= \begin{cases} g_k(x) & \text{if } x \in S \\ g_k(x) - \beta g'_k(x) & \text{if } x \notin S \end{cases}. \end{aligned} \tag{2.20}$$

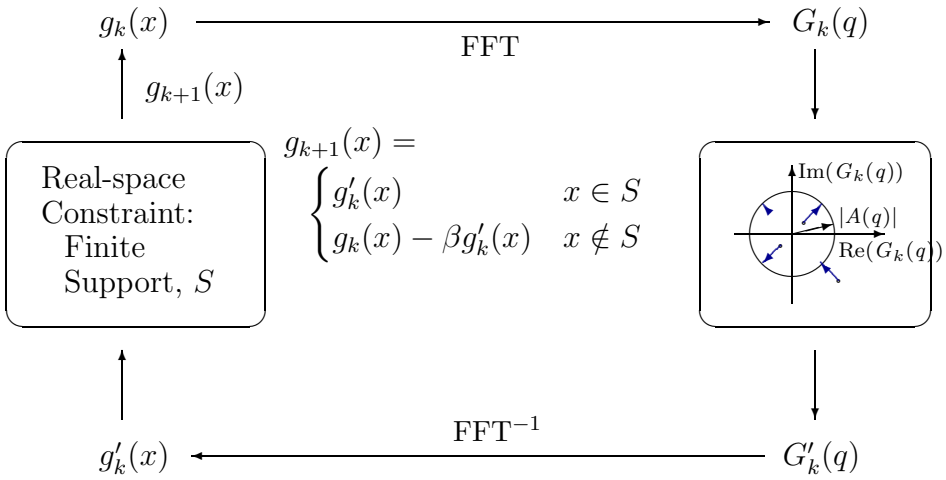


Figure 2.3: Fienup's HIO.

A second interesting feature of this nonlinear operation is that it is self-replicating in the sense that if an output $g'_k(x)$ is chosen as the input, the output is itself, since $g'_k(x)$ already satisfies the reciprocal-space constraint. Thus, no matter what input actually led to the generation of $g'_k(x)$, we can simply imagine that the input was $g'_k(x)$. This idea leads to the output/output algorithm, where the next iterate $g_{k+1}(x)$ is comprised of components from only the output of the non-linear operation:

$$\begin{aligned} g_{k+1}(x) &= g'_k(x) + \beta \Delta g_k(x) \\ &= \begin{cases} g'_k(x) & \text{if } x \in S \\ g'_k(x) - \beta g'_k(x) & \text{if } x \notin S \end{cases}. \end{aligned} \tag{2.21}$$

If we set $\beta = 1$ in [2.21], we arrive at a choice for $g_{k+1}(x)$ which is exactly the same as [2.7], *i.e.*, the output/output algorithm reduces to the error reduction algorithm for this choice of β. In fact, $\beta = 1$ is not always the best choice, so we could regard error reduction as a less than optimal special case of this algorithm.

Unfortunately, the output/output algorithm shares the tendency to stagnation seen in the error reduction algorithm, albeit for somewhat different reasons. The stagnation of the error reduction algorithm is due to its similarity to a steepest descents approach, that is, it finds local minima in the error metric and becomes trapped. On the other hand, the output/output algorithm tends to find iterates which do not change on subsequent iterations even though they are not properly solutions to the problem. To combat this

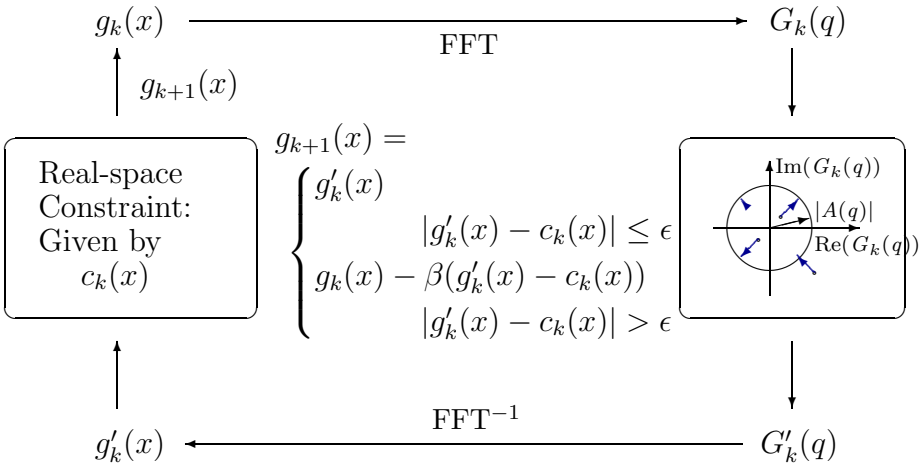


Figure 2.4: Millane's HIO.

stagnation in the output/output algorithm, a slight change is introduced:

$$g_{k+1}(x) = \begin{cases} g'_k(x) & \text{if } x \in S \\ g_k(x) - \beta g'_k(x) & \text{if } x \notin S \end{cases}, \tag{2.22}$$

giving the hybrid input/output (HIO) algorithm, so called because it creates a hybrid driving function composed of the input, $g_k(x)$, and the output, $g'_k(x)$. In this case, if at any x the output of the algorithm remains negative for more than one iteration, the value will continue to grow, eliminating the stagnation seen in the output/output algorithm. It is also important to note that $E^2_{F,k}$ is not a good choice for an error metric HIO since $g_k(x)$ is not intended to be a solution. Instead, we must use $E^2_{R,k}$. HIO is illustrated in Fig. 2.3

2.3.2 Generalization of constraints in HIO

It is possible to modify HIO to allow for a general constraint function, $c_k(x)$, to be used in place of the positivity and support constraints discussed above. One such extension is that proposed by Millane[60], whereby the condition on the application of the real space constraint given by [2.22] is altered to become:

$$g_{k+1}(x) = \begin{cases} g'_k(x) & \text{if } |c_k(x) - g'_k(x)| \leq \epsilon \\ g_k(x) - \beta\left(g'_k(x) - c_k(x)\right) & \text{if } |c_k(x) - g'_k(x)| > \epsilon \end{cases}, \tag{2.23}$$

Here, the constraint is the requirement that the iterate, $g'_k(x)$, agree with the constraint function to within a small amount, ϵ. In practice, this loosens the

constraint, presumably allowing the algorithm additional freedom to find a solution. Its obvious use is in enforcing constraints which may be uncertain, where a solution might otherwise be forbidden.

2.4 Iterative Methods as Projections

It is very difficult to visualize the effect that subsequent iterations of the preceding algorithms are having on the iterate. Indeed, it was not until Levi and Stark[52] noted that the GS algorithm was an implementation of the method of Projections onto Convex Sets (POCS) that a firm basis for understanding the algorithms could begin to be formed. It has since been noted that Fienup's Basic Input/Output Algorithm, Eqn. [2.20], is essentially the same as Dykstra's algorithm and that HIO, Eqn. [2.22], follows from the Douglas-Rachford Theorem[13]. In such optimization theories, iterates are mapped to various locations in the space and convergence implies that the trajectory of subsequent iterations is toward the intersection of two or more constraint subspaces. Naturally, any point contained in the intersection simultaneously satisfies all constraints and is therefore a solution. All of these methods can be proved effective in the case of convex constraints, unfortunately, the primary constraint in phase retrieval, *i.e.*, the collected intensity distribution is a non-convex constraint, as we shall see.

Convexity is a mathematical property: a set C_i is convex if for any $a, b \in C_i$ and real number $0 \leq \eta \leq 1$, then $(1-\eta)a + \eta b \in C_i$. In other words, a convex set is one for which every point on the shortest line between any two elements is also in the set. A few simple 2D Cartesian examples of convex and non convex sets are shown in Fig. 2.5. We will generally consider the case where each $C_i \subset \mathcal{H}$, where $\mathcal{H}$ is a Hilbert space. We then consider an N-dimensional vector ρ, in this Hilbert space as the representation of our N point discrete array. Further, a distance minimizing projection operator, π_i is required; such that $\pi_i \pi_i = \pi_i$ and $\pi_i \rho$ is the element of C_i closest to $\rho \in \mathcal{H}$. Naturally, if ρ already lines within C_i, $\pi_i \rho = \rho$.

Typically, an iterative phase retrieval method will make use of two or more constraints. The original GS algorithm made use of two modulus constraints, one in each real- and reciprocal-space. ER uses a reciprocal-space Fourier modulus constraint and two real-space constraints: positivity and finite support. Positivity and finite support are both convex constraints. In the case of positivity, it is clear that any point between two positive values will also be positive. Since the action of the support constraint is to set any value outside a region to zero, the shortest distance between any two iterates

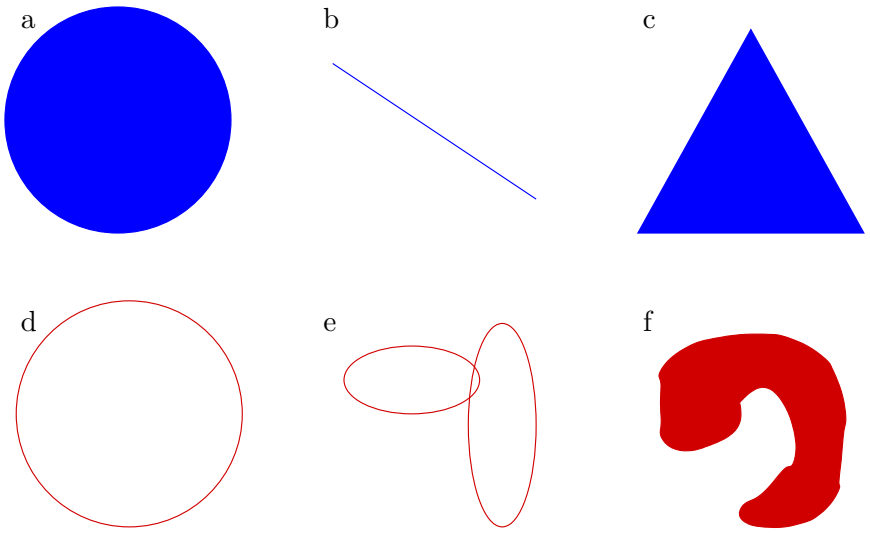


Figure 2.5: a) a filled disk, b) a 1D line, c) a filled triangle, d) a circle,e) two intersecting circles, f) a crescent-like shape. The colored region indicates the extent of the set. The first three(a-c) sets are convex, because the shortest line segment joining any two points within the set lies entirely within the set. The last three(d-f) sets are non-convex, because the shortest line segments joining any two points within the set contains at least one point outside the set.

which obey the constraint is traversed by iterates which also obey the constraint. However, in the case of a Fourier modulus constraint, the situation is quite different. The action of this constraint is to map the magnitude of the amplitude of the iterate at a point to the satisfy the measured data while keeping the iterate's phase information. This can be visualized as the case in Fig. 2.5d. It is important to note that the action of the constraint can be visualized in this way for each pixel, so the reciprocal-space space constraint set will be the surface of a cylindrical shell with rapidly varying radius as the pixel number changes. This is further complicated by the need to perform forward and back transforms before and after applying the constraint so that the operator acts on and returns a real-space object. If the modulus data were collected with some known errors, it would be possible to include that information in the constraint by means of allowing the surface to take on finite thickness, in accordance with the magnitude of the measurement error. With these tools in hand and a basic understanding of the nature of our constraints, we can proceed to investigate the trajectories of ER and HIO, and develop an introduction to the "Difference Map".

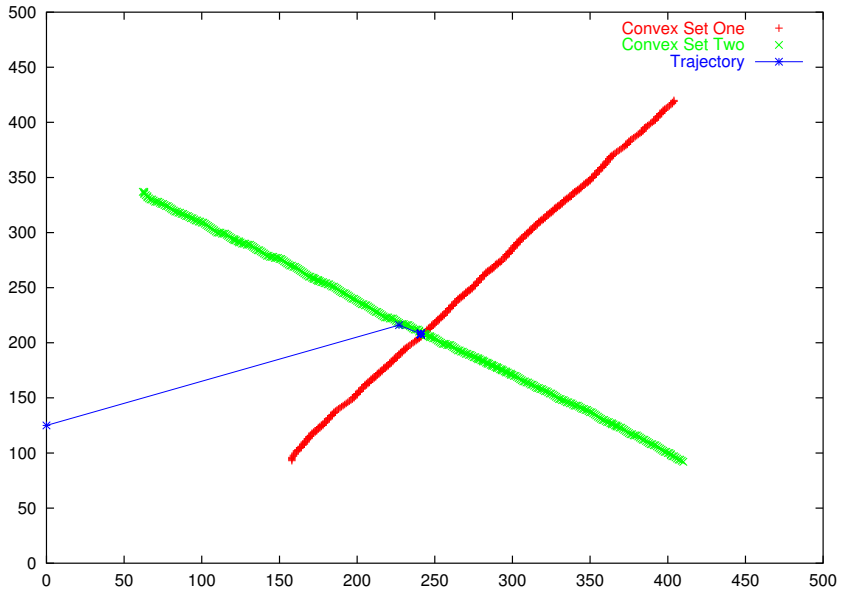


Figure 2.6: Trajectory of GS/ER. Because these two are alternating projection algorithms, the iterate will follow the contour of the boundary of a set.

2.4.1 Error Reduction

ER most closely resembles the original POCS method. Indeed, GS is simply recovered by defining two projection operators, $\pi_{m(o)}$ and $\pi_{m(i)}$. $\pi_{m(o)}$ simply maps the object space iterates onto the object-plane measurement and $\pi_{m(i)}$ consists of three operations: a Fourier transform into the image plane, a mapping of the amplitudes to the measured image plane data, and a back-transform. One iteration of GS would then be:

$$\rho_{n+1} = (\pi_{m(o)}\pi_{m(i)})\rho_n. \tag{2.24}$$

It bears reiterating that in this case neither of the constraint sets are convex, so the favorable convergence properties of POCS is not assured. ER uses slightly different constraints

$$\rho_{n+1} = (\pi_s\pi_+\pi_m)\rho_n = (\pi_{s+}\pi_m)\rho_n, \tag{2.25}$$

where π_m is the Fourier space modulus constraint previously denoted $\pi_{m(i)}$, π_s represents a support constraint, and π_+ a positivity constraint. In other words, application of the projection operator π_i is equivalent to applying the constraint i, because the action of π_i on a vector ρ is to map ρ to the set of all vectors which obey constraint i. For example, $\pi_s\rho$ is equivalent

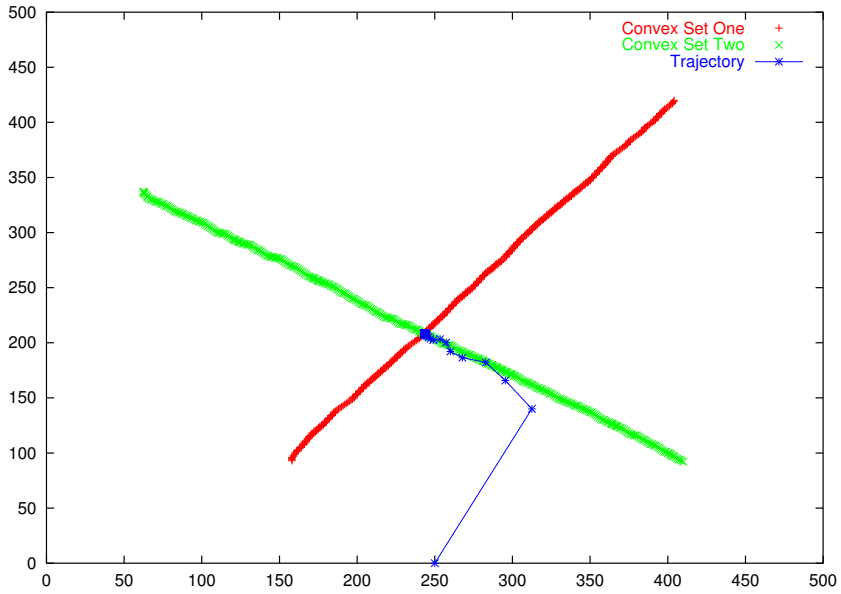


Figure 2.7: Trajectory of HIO. HIO does not follow the contour of the set as strictly as GS/ER. It has a tendency to spiral inward toward the intersection. For this example, $\beta = 0.3$ has been chosen so that the step size is short and therefore better illustrates this behavior.

to setting all elements of ρ outside the support region to zero. Positivity is rarely used except in conjunction with other constraints, so π_{s+} will denote the projection onto the combined support and positivity constraint set. A simple graphical demonstration of application of this mapping is shown in Fig. 2.6. Since during ER the iterate follows the boundary of the set, it is apparent that for the case of projections onto non-convex sets the algorithm may be unable to find the intersection. For example, if one of the sets contained a cusp, we could have a situation where the iterate is already located in a local minimum and would not move from the point, preventing the algorithm from moving the iterate on toward the intersection.

2.4.2 Hybrid Input/Output

The implementation of HIO requires a somewhat more complicated procedure than just the POCS-like extension to non-convex sets that is the case for GS and ER. Here, an iterate is generated[12] in accordance with:

$$\rho_{n+1}(u) = \begin{cases} \pi_m \rho_n(u) & \text{if } u \in S \\ (1 - \beta\pi_m)\rho_n(u) & \text{if } u \notin S \end{cases}, \tag{2.26}$$

where π_m is the same projection operator as before, β is a real number, u is the pixel number in an N element array, and S represents a region of real-space within which the object is known to exist. Thus, if an element of the N-dimensional vector ρ lies outside the real-space support constraint, that element is changed by $\beta\pi_m\rho$, yielding a combination of itself and its projection onto the Fourier modulus constraint. This acts as a sort of feedback, whereby the iterate is pushed away from its current, incorrect value rather than merely being set to zero, as is the case for ER. HIO can be expressed in the projector notation as:

$$\begin{aligned}\rho_{n+1} &= \pi_s\pi_m\rho_n + (1-\pi_s)(1-\beta\pi_m)\rho_n \\ &= \pi_s\pi_m\rho_n + \rho_n - \beta\pi_m\rho_n - \pi_s\rho_n + \beta\pi_s\pi_m\rho_n \\ &= \left[1 + (1+\beta)\pi_s\pi_m - \pi_s - \beta\pi_m\right]\rho_n.\end{aligned} \tag{2.27}$$

2.4.3 Difference Map

Following Elser[28] define the Difference Map (DM) operator in an N-dimensional Euclidean vector space E^N, $D : E^N \to E^N$, to be:

$$\rho_{n+1} = D\rho_n = (1+\beta\Delta)\rho_n = \left[1 + \beta\left(\pi_1 f_2 - \pi_2 f_1\right)\right]\rho_n, \tag{2.28}$$

where $\beta \in \Re$ is nonzero, $f_i : E^N \to E^N$, and π_i is a projection operator. We are interested in some $\rho \in E^N$, which can be uniquely represented by the sum of the projection of ρ onto the constraint sets and a vector contained in the complement of the union of the constraint sets, *e.g.*, $\rho = x_1 + x_2 + y$. Thus, the action of π_i is:

$$\begin{aligned}\pi_1(x_1 + x_2 + y) &= x_1 + a_2 + b_1 \\ \pi_2(x_1 + x_2 + y) &= a_1 + x_2 + b_2.\end{aligned} \tag{2.29}$$

We seek ρ such that $\Delta\rho = 0$. This is a fixed point of the difference map, ρ^*, which can be mapped onto a solution, *i.e.*, onto the intersection of the constraint subspaces: $(\pi_1 f_2)(\rho^*) = \rho_{1\cap 2} = a_1 + a_2 + b$. We choose f_i to be:

$$f_i = (1+\gamma_i)\pi_i - \gamma_i, \tag{2.30}$$

which maps ρ onto a point on the line defined by ρ and its projection, $\pi_i(\rho)$, and parameterized by some real number γ_i.

In our case, π_1 will be π_s and π_2 will be π_m; however, many other types of constraints[28] may be used, depending on the physical situation involved. It is important to note that not all fixed points actually lie in the intersection

of the constraint sets. In particular, it is possible that in the case noisy or incomplete data the sets will not have an intersection, in which case the algorithm may diverge along a line connecting the two sets.

One of the primary benefits of DM is that there are three parameters, β, γ_1, and γ_2, which can be varied. Of course, some values of γ_i at fixed β are unlikely to yield favorable results. To find these values, we ask what happens to $\rho = x_1 + x_2 + y$ after n iterations of the difference map. One iteration yields

$$\begin{aligned}
D\rho =&[1 + \beta(\pi_1[(1+\gamma_2)\pi_2 - \gamma_2] - \pi_2[(1+\gamma_1)\pi_1 - \gamma_1])\rho \\
=&x_1 + x_2 + y + \beta(1+\gamma_2)\pi_1\pi_2(x_1 + x_2 + y) - \beta\gamma_2\pi_1(x_1 + x_2 + y) \\
&- \beta(1+\gamma_1)\pi_2\pi_1(x_1 + x_2 + y) + \beta\gamma_1\pi_2(x_1 + x_2 + y) \\
=&x_1 + x_2 + y + \beta(1+\gamma_2)(a_1 + a_2 + b_1) - \beta\gamma_2(x_1 + a_2 + b_1) \\
&- \beta(1+\gamma_1)(a_1 + a_2 + b_2) + \beta\gamma_1(a_1 + x_2 + b_2) \\
=&x_1 + x_2 + y + \beta\gamma_2(a_1 - x_1) + \beta\gamma_1(x_2 - a_2) + \beta(b_1 - b_2) \\
=&a_1 + a_2 + y + (1 - \beta\gamma_2)(x_1 - a_1) + (1 + \beta\gamma_1)(x_2 - a_2) + \beta(b_1 - b_2).
\end{aligned} \tag{2.31}$$

Applying D once more gives

$$\begin{aligned}
D^2\rho =&[1 + \beta(\pi_1[(1+\gamma_2)\pi_2 - \gamma_2] - \pi_2[(1+\gamma_1)\pi_1 - \gamma_1])D\rho \\
=&a_1 + a_2 + y + (1 - \beta\gamma_2)(x_1 - a_1) + (1 + \beta\gamma_1)(x_2 - a_2) + \beta(b_1 - b_2) \\
&+ \beta(1+\gamma_2)\pi_1\pi_2(a_1 + a_2 + y + (1 - \beta\gamma_2)(x_1 - a_1) \\
&+ (1 + \beta\gamma_1)(x_2 - a_2) + \beta(b_1 - b_2)) \\
&- \beta\gamma_2\pi_1\times \\
&(a_1 + a_2 + y + (1 - \beta\gamma_2)(x_1 - a_1) + (1 + \beta\gamma_1)(x_2 - a_2) + \beta(b_1 - b_2)) \\
&- \beta(1+\gamma_1)\pi_2\pi_1(a_1 + a_2 + y + (1 - \beta\gamma_2)(x_1 - a_1) \\
&+ (1 + \beta\gamma_1)(x_2 - a_2) + \beta(b_1 - b_2)) \\
&+ \beta\gamma_1\pi_2(a_1 + a_2 + y + (1 - \beta\gamma_2)(x_1 - a_1) \\
&+ (1 + \beta\gamma_1)(x_2 - a_2) + \beta(b_1 - b_2)) \\
=&a_1 + a_2 + y + (1 - \beta\gamma_2)(x_1 - a_1) + (1 + \beta\gamma_1)(x_2 - a_2) + \beta(b_1 - b_2) \\
&+ \beta(1+\gamma_2)(a_1 + a_2 + b_1) \\
&- \beta\gamma_2(a_1 + a_2 + b_1 + (1 - \beta\gamma_2)(x_1 - a_1)) \\
&- \beta(1+\gamma_1)(a_1 + a_2 + b_2) \\
&+ \beta\gamma_1(a_1 + a_2 + b_2 + (1 + \beta\gamma_1)(x_2 - a_2))
\end{aligned}$$

$$
\begin{aligned}
=&a_1 + a_2 + y + (1 - \beta\gamma_2)(x_1 - a_1) + (1 + \beta\gamma_1)(x_2 - a_2) + \beta(b_1 - b_2) \\
&- \beta\gamma_2(1 - \beta\gamma_2)(x_1 - a_1) \\
&+ \beta\gamma_1(1 + \beta\gamma_1)(x_2 - a_2) \\
=&a_1 + a_2 + y + (1 - \beta\gamma_2)^2(x_1 - a_1) + (1 + \beta\gamma_1)^2(x_2 - a_2) \\
&+ \beta(b_1 - b_2).
\end{aligned}
\tag{2.32}
$$

Now, extrapolating [2.31] and [2.32] to n iterations, we have

$$
D^n\rho = a_1 + a_2 + y + (1 - \beta\gamma_2)^n(x_1 - a_1) + (1 + \beta\gamma_1)^n(x_2 - a_2) + n\beta(b_1 - b_2).
\tag{2.33}
$$

In order for [2.33] to converge to a fixed point, *i.e.*, $D^n\rho = \rho^* = a_1 + a_2 + y$, the last three terms must disappear. If we assume that a solution exists, $b_1 = b_2 = b$–since $b_1 \neq b_2$ is equivalent to the assumption that the sets do not intersect–and the last term disappears. The other two terms in question are power series in n, so as long as $-2 < \gamma_1\beta < 0$ and $0 < \gamma_2\beta < 2$, we can be assured that the series converge, as $|1 - \beta\gamma_2| < 1$ and $|1 + \beta\gamma_1| < 1$. Thus, DM converges to a fixed point, from which we can easily achieve a true solution by calculating

$$
\rho_{1\cap 2} = \pi_1 f_2 \rho^* = a_1 + a_2 + b.
\tag{2.34}
$$

It is also apparent from [2.33] that if the sets do not intersect DM will diverge for at least some values of γ_i along the line $b_1 - b_2$, due to the $n\beta(b_1 - b_2)$ term. Some examples illustrating the changes in the algorithm with different parameters will be given in the next section, but there two important cases that must be mentioned.

First, we set $\gamma_1 = -1$, so that $f_1 = (1 - 1)\pi_s + 1 = 1$, the identity map, and $\gamma_2 = 1/\beta$. Then, [2.28 becomes

$$
\begin{aligned}
\rho_{n+1} = D\rho_n &= [1 + \beta\,(\pi_s[(1 + 1/\beta)\pi_m - 1/\beta] - \pi_m)]\rho_n \\
&= [1 + (1 + \beta)\pi_s\pi_m - \pi_s - \beta\pi_m]\rho_n
\end{aligned}
\tag{2.35}
$$

which is exactly [2.27], Fienup's HIO algorithm.

Second, we set $\gamma_1 = -1/\beta$ and $\gamma_2 = 1/\beta$, so that [2.28] is

$$
\begin{aligned}
\rho_{n+1} &= [1 + \beta\,(\pi_1[(1 + 1/\beta)\pi_2 - 1/\beta] - \pi_2[(1 - 1/\beta)\pi_1 + 1/\beta])]\rho_n \\
&= [1 + (1 + \beta)\pi_1\pi_2 - \pi_1 + (1 - \beta)\pi_2\pi_1 - \pi_2]\rho_n
\end{aligned}
\tag{2.36}
$$

Recalling the definition of ρ and the action of the projectors defined in 2.29,

we can rewrite [2.36]

$$\begin{aligned}\rho_{n+1} &= [1 + (1+\beta)\pi_1\pi_2 - \pi_1 + (1-\beta)\pi_2\pi_1 - \pi_2](x_1 + x_2 + y) \\ &= (x_1 + x_2 + y) + [\beta(a_1 + a_2 + b_1) - x_1 + a_1] \\ &\quad - [\beta(a_1 + a_2 + b + 2) + x_2 - a_2] \\ &= a_1 + a_2 + y + \beta(b_1 - b_2)\end{aligned} \tag{2.37}$$

so, in one iteration of the DM, we have arrived at a position in the Hilbert space which is very close the intersection of the constraint sets, assuming it exists. In fact, in the case that the intersection is not empty, $b_1 = b_2 = b$ and one application of either π_1 or π_2 will bring the iterate into the intersection,

$$\begin{aligned}\rho_{n+1} &= \pi_i[a_1 + a_2 + y + \beta(b - b)] \\ &= a_1 + a_2 + b = \rho_{1\cap 2}\end{aligned} \tag{2.38}$$

These are the so-called maximally contractive values of γ_i, since a single iteration brings the iterate trivially close to the intersection.

A trajectory for the DM is illustrated in Fig. 2.8. For this example, the parameters are chosen so that DM is both maximally contractive and corresponds to HIO: $\beta = 1$, $\gamma_1 = -1/\beta = -1$, and $\gamma_2 = 1/\beta = 1$. To find ρ_1 from ρ_0, the starting point, we first must calculate $\pi_1 f_2 \rho_0$ and $\pi_2 f_1 \rho_0$ in [2.28]. $f_i\rho_n$ is a point on a line labeled $f_i\rho_n(\gamma_i)$ in Fig. 2.8, in the current case, $\gamma_1 = -1$ gives, from [2.30], $f_1 = (1-1)\pi_1 + 1 = 1$, the identity map, and γ_2 gives $f_2 = 2\pi_2 - 1$, which is sometimes called a reflector, because it "reflects" the iterate about its projection. Then, each of the points $f_i\rho_n$ is projected onto the set j, where $j = \{1, 2\}, j \neq i$–in other words, we obtain $\pi_1 f_2 \rho_n = \pi_1(2\pi_2 - 1)\rho_n$ and $\pi_2 f_1 \rho_n = \pi_2 \rho_n$. Finally, we add the result from the subtraction of these two vectors to the vector ρ_n to obtain ρ_{n+1}.

2.4.4 Trajectories

In an effort to develop an understanding of the convergence behavior, *i.e.*, how the iterate approaches the intersection of the sets, this section contains a series of two dimensional sets and the trajectories ER, HIO, and DM follow to the solution, represented by the intersection of the sets. A similar investigation has been performed for the case of these and other algorithms for the case of two straight lines and of one straight line and the union of two touching circles by Marchesini[54].

In order to explore the behavior of the algorithms presented, a program takes as input a UNIX bitmap (.ppm) image, the starting point, the algo-

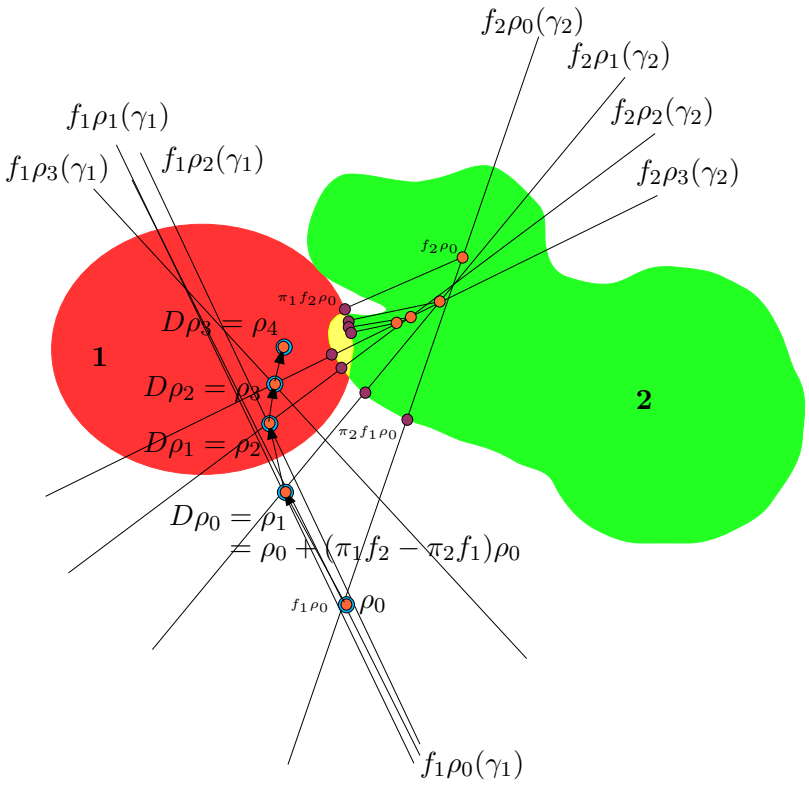


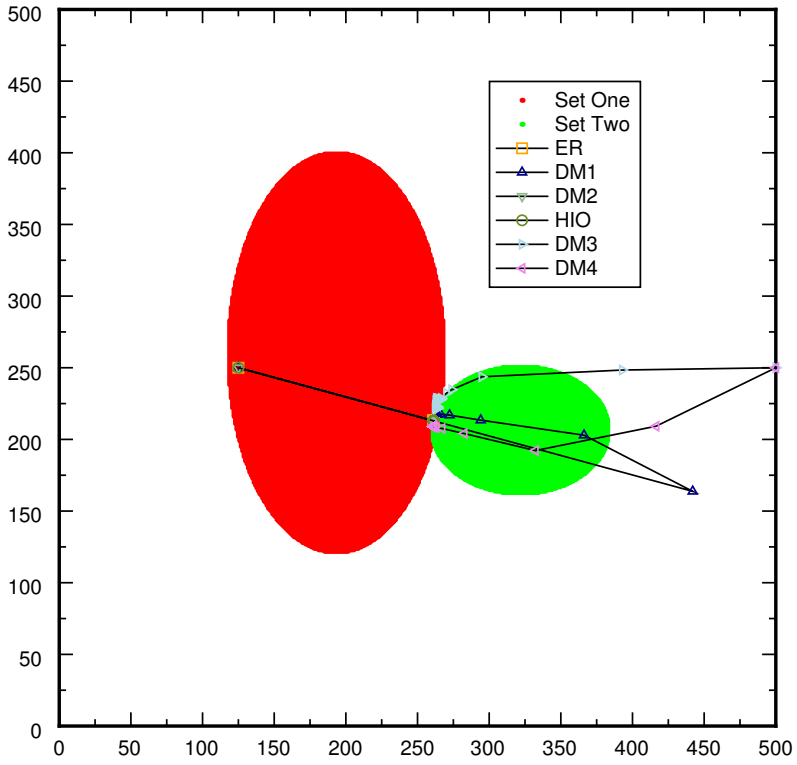
Figure 2.8: The trajectory of DM with $\beta = 1$, $\gamma_1 = -1$, and $\gamma_2 = 1$. With this choice of parameters, the DM is maximally contractive and corresponds to Fienup's HIO with $\beta = 1$. $f_i\rho_0(\gamma_i)$ is the line connecting ρ_0 with $\pi_i\rho_0$ and $f_i\rho_0$ is the point on that line selected by the choice of γ_i. The orange points correspond the the positions indicated by $f_i\rho_n$, the violet to $\pi_j f_i\rho_n$, and the blue to the iterate, ρ_n. Note the spiral trajectory of the iterate as the algorithm searches for the intersection, in yellow.

rithm to investigate, and any parameter required for the algorithm. The intent is to examine the behavior of ER and DM in several situations to help us understand results in subsequent chapters. In the case of DM, it is sometimes suggested that a final projection onto one of the constraint sets should follow the last application of DM. We have not done this in these trajectories since the behavior of DM itself is under study. The values of the parameters in the difference map are given in each of the tables, but a few words must be said about the logic behind their choice. In Fienup's original paper[33], β's near one performed well, so we choose $\beta = 0.7$ and $\beta = 1.2$ for the DM variants corresponding to HIO–*i.e.*,$\gamma_1 = -1$ and $\gamma_2 = 1/\beta$. Additionally, it is interesting to examine the maximally contractive parameters $\gamma_1 = -1/\beta$ and $\gamma_2 = 1/\beta$. Finally, we use one set of parameters that violates the condition for the convergence of the DM by choosing $\gamma_2 = 3.33$ so that $\beta\gamma_2 = 2.33 > 2$.

Fig. 2.9 represents the classical problem of finding the nonempty intersection of convex sets. Fig. 2.10 is the special case where the intersection is a unique point. As is expected, the exact shape of the set is unimportant, only its convexity affects the behavior of the algorithms. For these two cases, ER is seen to be the most efficient algorithm given the chosen starting point as it converges in a single iteration. It is interesting to note that while in the case of Fig. 2.10 ER will always converge in one iteration–because of the shape of the sets and their point of intersection–in the general case, the rapidity of the convergence will depend on starting position as well as the geometry of the sets. DM and variants, including HIO, are more leisurely in their approach, but still find the correct solution. Here we see that DM explores the space in a trajectory resembling a spiral search pattern, which is a very nice feature for avoiding local minima as will be seen.

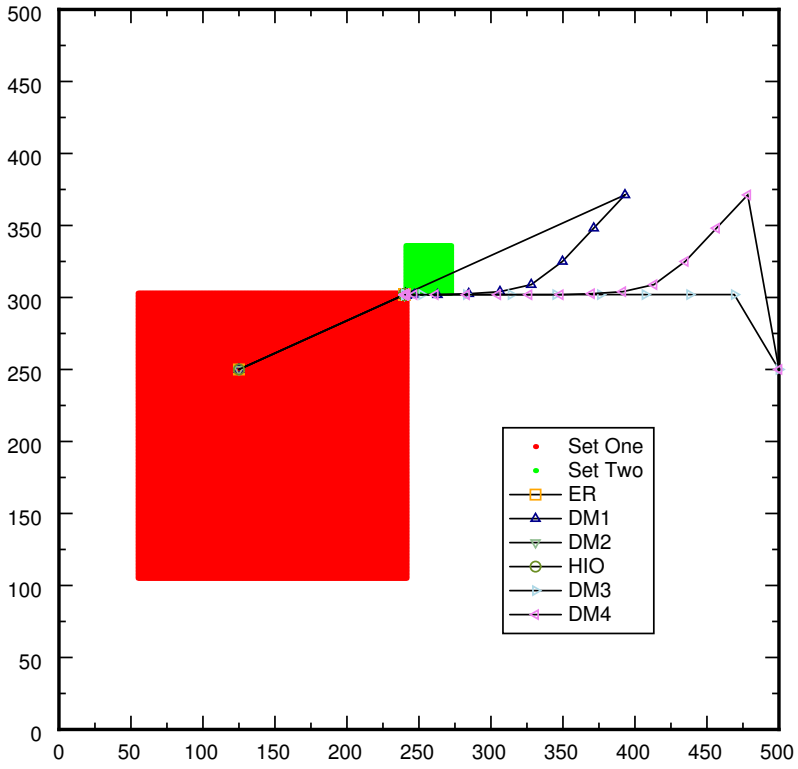
Fig. 2.11 is the case of a unique solution for two non-convex constraints. Again, all algorithms find the correct solution. In this case, ER again converges quickly due to a favorable starting point. For this geometry, the DM is seen to overshoot the intersection and then approach it from the opposite side. Fig. 2.12 again employs rings, but now the intersection is many points. Here, ER will find the closest point in the intersection to its starting position, but DM will find different solutions very far apart depending on the values of its parameters.

Fig. 2.13 is the case of one nearly straight line and a curved line. In this case, ER follows the contour of one of the sets until it finds a local minima and then stagnates. On the other hand, DM ignores the local minima entirely and spirals in toward the global minima. This is the first example



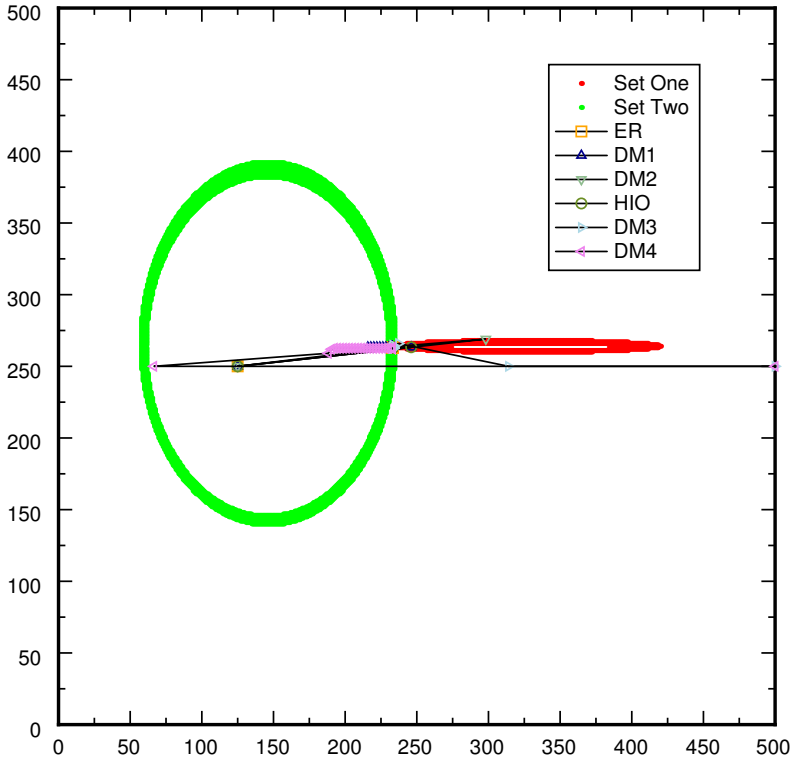
Label	Algorithm	Start x	Start y	Iterations	β	γ_1	γ_2
ER	ER	125	250	50	–	–	–
DM1	DM	125	250	50	0.7	-1	3.33
DM2	DM	125	250	50	0.7	-1.43	1.43
HIO	DM	125	250	50	0.7	-1	1.43
DM3	DM	500	250	50	0.1	-10	10
DM4	DM	500	250	50	0.7	-1	3.33

Figure 2.9: Filled ovals. The intersection contains many points, so the final solution depends upon the starting point.



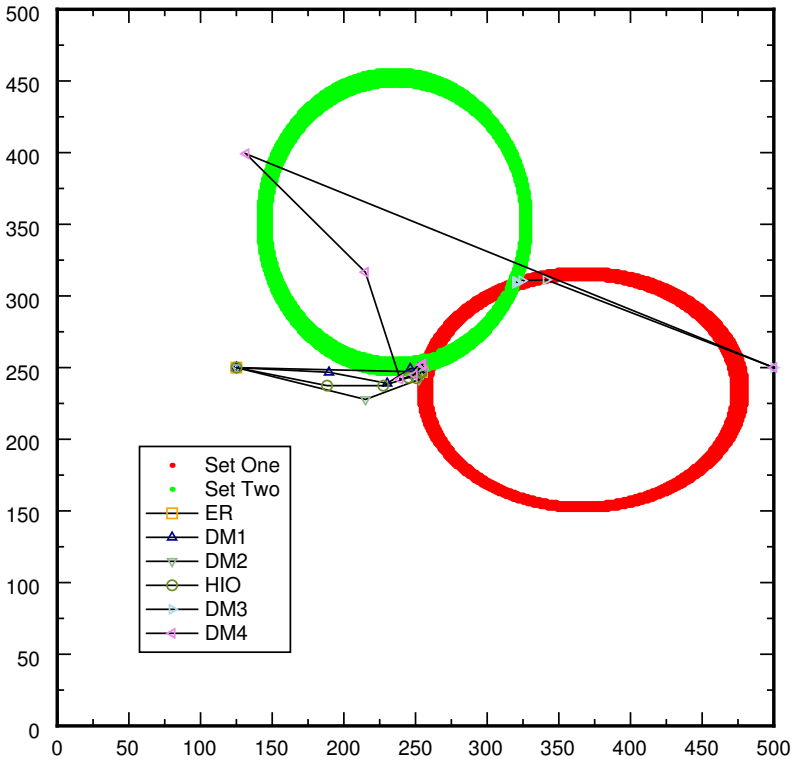
Label	Algorithm	Start x	Start y	Iterations	β	γ_1	γ_2
ER	ER	125	250	50	–	–	–
DM1	DM	125	250	50	0.7	-1	3.33
DM2	DM	125	250	50	0.7	-1.43	1.43
HIO	DM	125	250	50	0.7	-1	1.43
DM3	DM	500	250	50	0.1	-10	10
DM4	DM	500	250	50	0.7	-1	3.33

Figure 2.10: Filled Squares. Only a single point lies in the intersection of the two sets, and all algorithms find that point.



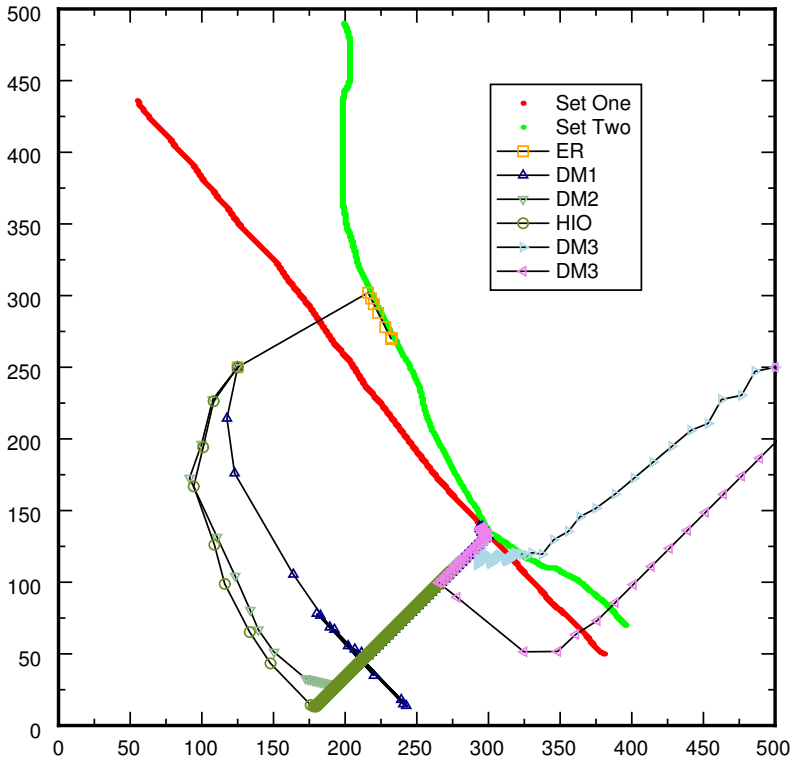
Label	Algorithm	Start x	Start y	Iterations	β	γ_1	γ_2
ER	ER	125	250	50	–	–	–
DM1	DM	125	250	50	0.7	-1	3.33
DM2	DM	125	250	50	0.7	-1.43	1.43
HIO	DM	125	250	50	0.7	-1	1.43
DM3	DM	500	250	50	0.1	-10	10
DM4	DM	500	250	50	0.7	-1	3.33

Figure 2.11: Touching Rings. Even though the sets are non-convex, all algorithms find the one point which lies in the intersection of the two rings.



Label	Algorithm	Start x	Start y	Iterations	β	γ_1	γ_2
ER	ER	125	250	50	–	–	–
DM1	DM	125	250	50	0.7	-1	3.33
DM2	DM	125	250	50	0.7	-1.43	1.43
HIO	DM	125	250	50	0.7	-1	1.43
DM3	DM	500	250	50	0.1	-10	10
DM4	DM	500	250	50	0.7	-1	3.33

Figure 2.12: Intersecting Rings. Again, although the sets are non-convex the algorithms find a particular point in the intersection. The final solution depends upon the starting point and, in the case of DM, the values of β, γ_1, and γ_2.



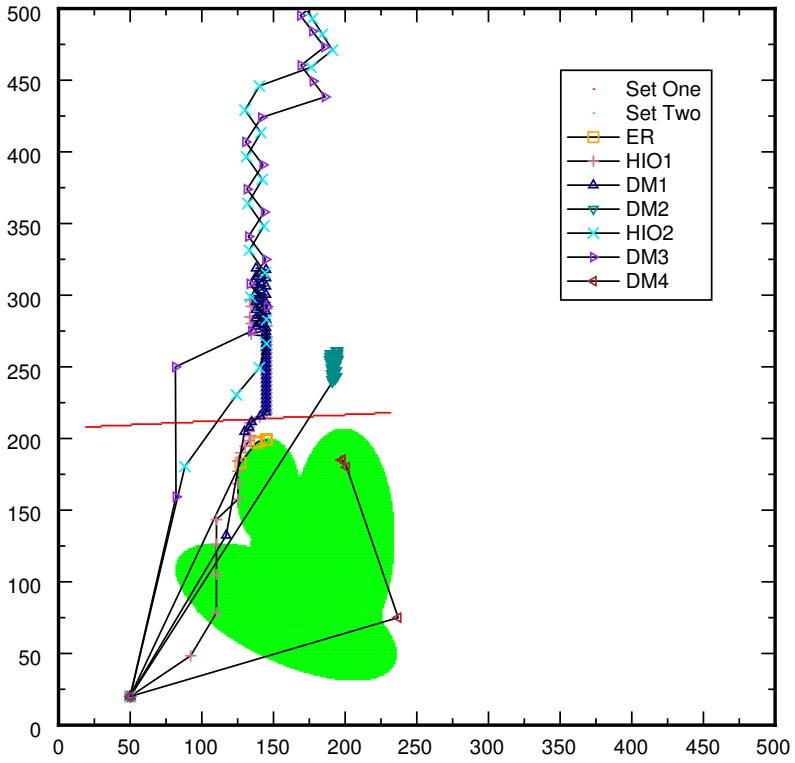
Label	Algorithm	Start x	Start y	Iterations	β	γ_1	γ_2
ER	ER	125	250	50	–	–	–
DM1	DM	125	250	50	0.7	-1	3.33
DM2	DM	125	250	50	0.7	-1.43	1.43
HIO	DM	125	250	50	0.7	-1	1.43
DM3	DM	500	250	50	0.1	-10	10
DM4	DM	500	250	50	0.7	-1	3.33

Figure 2.13: Touching lines. In this example, HIO and DM find the global minimum while ER stagnates at a point along the boundary of the green set which is locally closer to the red set than the adjacent points.

where the forbidden DM parameters–$\beta = 0.7$, $\gamma_1 = -1$, $\gamma_2 = 3.33$–at one of the two starting points does not reach the intersection. This illustrates a very interesting point, that even disallowed parameters may allow the DM to find a solution depending on the starting point.

Finally, is the case of one convex set, a 1D line, and a non-convex set, a multilobed compact object, without intersection is displayed in Fig. 2.14. Here, ER again finds a local minima and the iterate becomes stuck. The DM map behavior is quite interesting, however. For the case of $\beta = 1.2$, we see that it finds the local minima where ER stagnates and then diverges along the line joining the two sets. What is interesting is that DM eventually finds the line joining the two closest points in the sets; however, it continues to diverge along this line rather than seek an element in one set close to the other set. For the case of $\beta = 0.2$, the behavior is very similar, except that we can see that β controls the distance traveled in an iteration, which can be useful in cases like that of Fig. 2.11, where DM tends to overshoot the intersection. Again, the forbidden parameters $\beta = 0.2$, $\gamma_1 = -10$, and $\gamma_2 = 10$ give an interesting result. Namely, with these values DM overshoots the local minimum and finds the global one, where it causes the iterate to wander randomly near the point of closest approach. With these values, however, DM does not converge to particular solution. The divergent behavior of DM near the region of closest approach between the sets motivates the combination of DM and ER, where DM might be used to find this line and ER to project an iterate onto one set where it is nearest the other thus avoiding the stagnation problems inherent in ER. Finally, we note that with $\beta = -2$, $\gamma_1 = 0.5$, and $\gamma_2 = -1$ DM spirals in with the opposite chirality avoiding the local minimum completely and stagnates near the point closest approach between the two sets.

These examples illustrate what was recognized by Fienup and others early in the development of these algorithms. First, ER is an efficient algorithm provided there are no local minima in the error metric. It is also apparent that in the case of non-intersecting sets, the combination of a DM variant and ER will often help to avoid ER's failings near local minima. In fact, were the shape of the constraints known in a given experiment, it would be fairly easy to construct a plan of action which would often lead to a correct solution. Unfortunately, it is essentially impossible to visualize the Fourier modulus constraint. At each pixel in reciprocal-space, the constraint may be visualized as in Figs. 2.2-2.4 as a circle of radius given by the magnitude of the measured amplitude in the complex plane. The constraint will then project the iterate's amplitude onto this circle preserving the phase. In



Label	Algorithm	Start x	Start y	Iterations	β	γ_1	γ_2
ER	ER	50	20	50	–	–	–
HIO1	DM	50	20	50	0.2	-1	5
DM1	DM	50	20	50	0.2	-5	5
DM2	DM	50	20	50	0.2	-10	10
HIO2	DM	50	20	50	1.2	-1	0.833
DM3	DM	50	20	50	1.2	-0.833	0.833
DM4	DM	50	20	50	-2	0.5	-1

Figure 2.14: Line and Blob. In this case, the two sets do not intersect. ER stagnates at a non-global minima. Some parameters of the difference map, as discussed in the text, finds the local minima and diverge along the line connecting the two sets. Interestingly, these divergent iterates eventually find the line of closest approach between the two sets, but continue to diverge rather than seek the nearest point in one set to the other set. Also, for some values of the DM the local minima is ignored and DM stagnates near the line of closest approach between the sets.

an experiment–where a measurement spans N pixels–the Fourier modulus constraint is a cylindrical shell of varying radius along the pixel dimension. Complicating matters further is the requirement that π_2 must act in real-space, so that set 2 is related to the reciprocal-space cylindrical shell by a union of Fourier transforms of all elements in the shell.

These examples show that the shapes of the constraints are very important in attempting to predict the convergence behavior of the algorithms. The thickness, seems to be more or less irrelevant except that ER will always trace the boundary of the set whereas HIO/DM will not necessarily find the point in the intersection that's closest to one set or the other. In the real world, solutions will tend to pathological rarity [17, 11], *i.e.*, if one can be found, it's almost certainly the correct one, so the existence of multiple solutions is unlikely to cause difficulty in practice.

In the case of the DM, the approach of the iterate to the intersection is governed by the choice of parameters and shape of the constraints. If the shape were known, one could tailor parameters to converge as quickly, slowly, or any other way one might like. Again, this is in marked contrast to the real life situation where the shape of the modulus constraint is very hard to understand. This also extends to the existence of an intersection of the sets. As Fig. 2.14 shows, the choice of parameter in DM is vitally important, since the iterate will diverge if the parameters are chosen poorly.

The examples in this section do not directly relate to the problem of phase retrieval from intensity measurement; however, some insight as to how to avoid obvious mistakes, *e.g.*, really large betas, and they do foreshadow the results of noisy simulation in Chapter 4. With this feeling for the behavior of ER, HIO, and DM, exploration the behavior of these methods in reconstructing simulated objects and ask how the presence of noise alters the convergence properties in will continue in Chapter 4. Rather than extending the discussion at this time, we will instead describe a CXD experiment and present the results of applying these algorithms to reconstructing a 3D real-space density from its diffraction. This result will raise questions that will prove important for motivating the simulations of Chapter 4.

Chapter 3

CXD Imaging from Au Crystals

The experiment described here was intended as a proof of principle to establish that CXD data could be used to image 3D objects. The experiment was performed at the Advanced Photon Source (APS) located at Argonne National Laboratory. The sample is composed of many single crystals of Au, approximately $30\mu m^3$ in volume. We will use the oversampling techniques discussed in Chapter 1 to assure sufficient data is measured. This data will then be fit with the algorithms discussed in Chapter 2.

Such a CXD experiment requires a very high quality X-ray beam. The traditional measure of this is the brilliance of the source. Brilliance is defined as photons per second per mm^2 per mrad per 0.1% bandwidth. As such the brilliance is a measure of flux, with units of photons per second per mm^2, corrected for the divergence, in mrad, and the monochromaticity, in units of percent of bandwidth. Third generation sources with insertion devices, which will be addressed shortly, create very brilliant beams. We will begin by addressing a few important aspects of synchrotron radiation that determines important parameters in the experiment. The synchrotron itself determines the beam size, which is crucial in establishing the coherence of the incident beam in the transverse directions. The insertion device is responsible for the creation of the X-ray beam from the electrons orbiting the storage ring. Unfortunately, the X-ray beam must still propagate through a number of optical elements before reaching the sample that may affect the coherence of the beam.

Once we have established the properties of the incident beam, the experimental geometry of the CXD will be explained in terms of the Ewald construction and equipment used to traverse and sample reciprocal space. Special attention will be paid to the CCD detector used in the measure-

ment, making connection to the simulations of Chapter 4. The experiment described here is conducted at high temperature, so the *in situ* heating cell will be described.

The last sections of this chapter are devoted to the CXD data collected and the results of fitting this data in two and three dimensions. As part of this process, the data handling will described in detail and a similarly complete description of the computer programs used to handle and phase the data will be provided. Finally, an preliminary analysis of the 3D real-space density back transformed from the phased CXD data will be presented.

3.1 Facility

The APS is a third generation X-ray source. Such a source takes advantage of the radiation generated by the acceleration of a charged particle in a synchrotron. Essentially, the source consists of two distinct elements: the handling of charged particles and the creation of X-rays. The charged particles in this case are electrons[86, 1]. 100 keV electrons are created from a cathode ray tube and accelerated via a linear accelerator to an energy of 450 MeV before being transferred to a booster ring which imparts energy of 32 keV per orbit until the desired energy of 7 GeV is reached. The 7 GeV electrons are then injected into the storage ring, which has circumference 1104m. Typically, the APS storage ring will hold 100 mA of current.

There are many operating modes for the APS synchrotron, for example, the structure of the bunches of electrons in the ring can be varied. However, because the measurements made here last for tens of seconds–a typical electron will orbit the ring $271,000$ times a second–this type of modification is unimportant. Unfortunately, not all operating modes have such a small impact on CXD experiments. Over time, electrons are lost from the ring–for example, lost to scattering with ions in the imperfect vacuum of the ring–resulting in a slow decay in the flux of the X-ray beam produced by the insertion device.

3.1.1 Insertion device

X-ray are generated in the synchrotron when the charged particles are forced to accelerate, we refer to this as bend magnet (BM) radiation. The BM lines are used at the APS for experiments which require a broad spectrum (brilliance) of radiation. They typically have a divergence of γ^{-1}–where $\gamma = (1 - v^2/c^2)^{-1/2}$ is the usual relativistic parameter–and provide a characteristic energy of $h\nu = 0.665E_e^2B$[7].

Parameter	Value
Magnet material	Nd-Fe-B
Pole material	Vanadium permendur
Undulator period, λ_u	3.3 cm
Periods, N	72
Length	2.4 m
Minimum Gap	10.5 mm
Effective field at 11.5mm gap	0.803 ± 0.007 T
Effective field at 10.5mm gap	0.891 ± 0.009 T
Effective K value at 10.5mm gap	2.74 ± 0.03
First harmonic energy	2.96 ± 0.05 keV
RMS peak field error at 11.5mm gap	0.49 ± 0.07 %
RMS phase error at 11.5mm gap	$4.0° \pm 0.7°$

Table 3.1: Undulator A characteristics from Ref. [26].

The second type of experimental beamline is the insertion device (ID) as a source. IDs generally fall into one of two categories: wigglers and undulators. A wiggler is a series of magnets with alternating poles that force the electrons to make many oscillations. As such, the wiggler spectrum is the same as the BM, but the intensity of the beam is now proportional to the number of poles, N, since the intensity of the beam adds as if this ID were just a series of BMs. If an ID is carefully constructed so that the amplitudes of radiated X-ray field adds rather than the intensities, we have an undulator.

The ID at Sector 33, where this experiment was performed is an Undulator A[26]. Whereas the fundamental quantities of import for BM is the characteristic frequency ω_c and the relativistic parameter γ, for an undulator, we instead consider the undulator period λ_u in place of ω_c. As an electron passes through the undulator, it experiences alternating magnetic fields at the period λ_u. This causes the electrons to make very small amplitude deviations from the axis running the length of the undulator[7]. The maximum angular deviation from this axis is given by a dimensionless number, $K = eB_o/mck_u$(SI units), which is proportional to the magnetic field strength and the undulator period, through $k_u = 2\pi/\lambda_u$. In the Undulator A, K is varied by changing the gap between the poles, and thus the magnitude of the magnetic field. To achieve this addition of amplitudes, the electrons oscillate with a period such that the radiation emitted on the previous excursion from the undulator axis reaches the current excursion in an integer number of undulator periods. This results in a much sharper spectrum with a fundamental wavelength: $\lambda_1 = \lambda_u/2\gamma^2(1 + K^2/2 + (\gamma\theta)^2)$,

Parameter	Low-Emittance	High-Emittance
SR energy, E_e	7.0 GeV	7.0 GeV
SR current, I	100 mA	100 mA
Energy spread	0.096%	0.096%
Horiz. emittance	3.5×10^{-9} mrad	7.7×10^{-9} mrad
Vert. emittance	3.5×10^{-9} mrad	7.7×10^{-9} mrad
Coupling constant	1%	1%
Horiz. β func.	14.4 m	16.1 m
Vert. β func.	4.0 m	4.3 m
Dispersion func.	0.124 m	0 m
Horiz. size, σ_x	254μm	351μm
Vert. size, σ_y	12μm	18μm
Horiz. divergence, $\sigma_{x'}$	15.6μrad	21.8μrad
Vert. divergence, $\sigma_{y'}$	3μrad	4.2μrad

Table 3.2: SR characteristics from Ref. [26].

where θ is the angle between the undulator axis and the observer. Generally, the energy of a photon from the nth harmonic generated from a single electron passing through the undulator is[26]:

$$E_n[\text{keV}] = \frac{0.95 E_e^2[\text{GeV}] n}{\lambda_u[\text{cm}]\,(1 + K_{eff}/2 + \gamma^2\theta^2)}. \tag{3.1}$$

For the wiggler, the intensity generated was proportional to N, since the intensity of each oscillation of the electron was summed. In an undulator, the amplitudes are summed, so the intensity is proportional to N^2. The divergence of an undulator is symmetric about its axis and equal to $1/\gamma\sqrt{(1 + K^2/2)/nN}$. The properties of an Undulator A are listed in Table 3.1.

The usual assumption is that the undulator does not materially alter the structure of the bunches, so the X-ray beam size is the same as the cross section of the electron bunch, which is modeled as a Gaussian. Table 3.2 details two operating modes for the APS storage ring. The lattice labeled High-emittance was in operation during June of 2001 when these measurements were made, giving a source size of $\sigma_x = 351$ μm by $\sigma_y = 18$ μm. A simple geometric argument[7] based on the distance between points where the wave front is in phase, see Fig. 3.1, gives the two transverse coherence lengths $\xi_{x,y} = \frac{\lambda R}{2\sigma_{x,y}}$, where R is the source to sample distance. For this experiment, 9.5 keV X-ray were used and the sample was located 40 m from

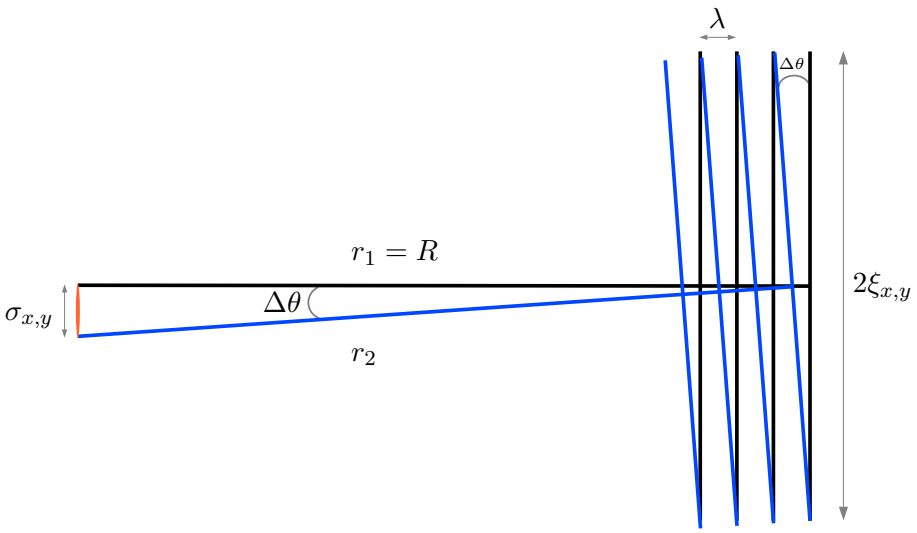


Figure 3.1: Illustration of the derivation of the transverse coherence length. Two rays, r_1 and r_2, leave the finite source with size $\sigma_{x,y}$ and travel the distance $R \gg \sigma_{x,y}$. The distance $2\xi_{x,y}$ must be traversed before the waves are again synchronized. So, $\xi_{x,y} \approx \lambda/(2\Delta\theta) \approx \lambda R/(2\sigma_{x,y})$.

the source, giving

$$\begin{aligned} \xi_x &= \frac{(1.305 \times 10^{-10})(40)}{2(351 \times 10^{-6})} = 7.44 \times 10^{-6} \text{ m} \\ \xi_y &= \frac{(1.305 \times 10^{-10})(40)}{2(18 \times 10^{-6})} = 145 \times 10^{-6} \text{ m} \end{aligned} \tag{3.2}$$

3.1.2 Monochromator

To derive the longitudinal coherence, we consider the case of two waves with the same wave front at some point[7]. By definition, the longitudinal coherence length is the distance traveled between a point where the waves are in phase and the closest point where they are out of phase. If the first wave has wavelength λ, we can write the wavelength of the second as $\lambda - \Delta\lambda$. We are interested in position at which the wave fronts are again synchronized, so we say that this occurs after some number N of wavelengths λ. Since we are interested in the shortest distance over which this is true, the we can write:

$$N\lambda = (N+1)(\lambda - \Delta\lambda), \tag{3.3}$$

which is twice longitudinal coherence length $\xi_{\|}$. Since our interest is in wave fields that are nearly coherent, we assume that $\Delta\lambda$ is a small number, which implies that $N \gg 1$, so [3.3] gives $(N+1)\Delta\lambda \simeq N\Delta\lambda \simeq \lambda$, then the

longitudinal coherence is:

$$\xi_{\parallel} = \frac{\lambda}{2}\frac{\lambda}{\Delta\lambda}. \tag{3.4}$$

Evidently, $\xi_{\parallel}$ depends on the energy of the radiation and bandwidth. In this experiment, these are properties of the undulator and monochromator. By differentiating Bragg's law, $m\lambda = 2d\sin\,\theta$, we arrive at

$$\frac{\Delta\lambda}{\lambda} = \frac{\Delta\theta}{\tan\,\theta_B}. \tag{3.5}$$

Now, we must relate $\Delta\theta$ to the experiment at hand. The first contribution is the Darwin width of the Si crystals in the double crystal monochromator. For thick crystal as we have in a monochromator, the reflectivity within a small angular region of the Bragg condition is almost unity and falls off quickly as the angle increases[98, 7]. The angular region about the Bragg condition is the Darwin width:

$$w_D = \frac{2r_o\lambda^2\,|F|}{\pi v\sin\,2\theta_B} \tag{3.6}$$

where r_o is the Thompson scattering length, λ is the wavelength of the incident radiation, F is the the structure factor for the reflection, and θ_B is the Bragg angle. For a (333) reflection from Si[19], we can calculate w_D:

$$w_D = \frac{2(2.82\times10^{-15}\text{ Å})(1.305\text{ Å})^2(36.5)}{\pi(5.43\text{ Å})^3\sin\,77.26^\circ} = 7.14\times10^{-6}\text{ rad.} \tag{3.7}$$

The second contribution to $\Delta\theta$ comes from the divergence of the X-ray beam from the undulator. This is a property of the SR of the synchrotron and has been carefully characterized[25, 26]. The double crystal monochromator (DCM) at Sector 34 is oriented vertically so that only the divergence in the vertical direction contributes to $\xi_{\parallel}$. Therefore, we arrive at $\Delta\theta = w_D + \sigma_{y'}$ and the longitudinal coherence, [3.4]–using the vertical divergence of the beam from Table 3.2–is

$$\xi_{\parallel} = \frac{1.305\times10^{-10}\text{ m}}{2}\frac{\tan\,38.63^\circ}{(7.14+4.2)\times10^{-6}} = 4.60\times10^{-6}\text{ m.} \tag{3.8}$$

In order to not lose contrast between the fringes of a diffraction pattern, the optical path length difference must be less than $\xi_{\parallel}$. For a particle of thickness, D, the path length difference is:

$$\text{PLD} = D\sin\,\alpha_i + D\sin\,\alpha_f, \tag{3.9}$$

where α_i is the incident and α_f is the exit angle of the beam. For a specular (111) reflection in Au, PLD$\sim 0.55D$, so the maximum thickness of a particle would be over 7 μm.

Perhaps more important for the experiments of this chapter, is the extinction length, defined to be distance the wave propagates into a material before its intensity is attenuated to $1e$ of the incident intensity[7]. The extinction depth, Λ_{ext}, depends inversely on the structure factor, F, as

$$\Lambda_{ext} = \frac{\sin\theta}{\lambda}\frac{v_c}{2r_o\left|F\right|}, \tag{3.10}$$

where θ is the incident angle, v_c is the volume of the unit cell, and r_o is the classical electron radius. Using tabulated data from the International Tables for Crystallography[73], the extinction length of Au with 9.5 keV radiation is about 0.1 μm for a specular (111) reflection, indicating that we should expect significant attenuation of the X-ray beam. Of course, at high temperatures, the Debye-Waller factor[98] reduces the structure factor, which increases the extinction depth, according to [3.10]. This is because at finite temperatures, the structure factor is dependent on the thermal fluctuations of atoms about their lattice sites. Debye-Waller factors are quite difficult to measure accurately, but they can be estimated[83] in cases like ours where we are only interested in a rough answer. In the case of Au at 950° C, the structure factor decreases by about 18%, giving $\Lambda_{ext} \approx 0.12$ μm.

3.1.3 Windows and mirrors

In practice, it is often unavoidable to have multiple optical elements between the insertion device source and the sample location. As the X-ray beam passes through each of these elements it gains phase structure. It is important to emphasize that these elements do not result in a loss of coherence, but rather an increase in the complexity of the partial coherence function[85, 95, 93] at the sample location. For the most part, these optical elements are either Be windows, to protect the vacuum of the SR or other beamline components, or mirrors, commonly used for focusing and harmonic rejection. It is an important result that the contribution from such elements adds incoherently to the coherence function, so that the effective function can be written as a sum of the partial coherence of the source plus the additional structure from each element. If we separate out the spatial part of the mutual coherence function [1.18], we get the mutual intensity function

(MIF)[93]

$$J(Q_1, Q_2) = \Gamma(Q_1, Q_2, 0) = \langle A(Q_1, t) A^*(Q_1, t) \rangle_T \,, \tag{3.11}$$

where Q_i represent points where the measurement is made, A is the electric field, and $\langle\rangle_T$ denotes a time average. Then the MIF transmitted through an optical element is given by

$$J_{trans}(Q_1, Q_2) = T(\mathbf{u_1}) T^*(\mathbf{u_2}) J_{in}(\mathbf{u_1}, \mathbf{u_2}), \tag{3.12}$$

where $\mathbf{u_i}$ are located in the plane of the element, $T(\mathbf{u})$ is the transmittance function, and $J_{in}(\mathbf{u_1}, \mathbf{u_2})$ is the incident MIF.

The transmittance function can written in a general form:

$$T(\mathbf{u}) = B(\mathbf{u}) e^{-i\Phi(\mathbf{u})}, \tag{3.13}$$

where $B(\mathbf{u})$ is the absorption due to the element and $\Phi(\mathbf{u})$ is the phase shift. For the case of Be windows, it is normally a good assumption at hard X-ray energies to model a low atomic number element like Be as a pure phase object, *i.e.*, the beam passing through it is not absorbed, but still gains or loses a phase offset traveling through the element, and so the contribution to [3.12]depends on the thickness of the window $d(\mathbf{u})$. The index of refraction of X-rays in matter is $n = 1 - \delta + i\beta$, where δ is a small positive number and β is the absorption of the material. Therefore, we can write down the general transmittance function

$$\begin{aligned} B_W(\mathbf{u}) &= e^{-\bar{k}\beta d(\mathbf{u})} \\ \Phi_W(\mathbf{u}) &= -\bar{k}\delta d(\mathbf{u}), \end{aligned} \tag{3.14}$$

so we see that if β can be neglected, $B_W(\mathbf{u}) = 1$. Mirrors are also commonly considered to be phase objects, but contribute to the partial coherence through the finite roughness of their surfaces. Therefore, the contribution depends on the height function $h(\mathbf{u})$.

$$\begin{aligned} B_M(\mathbf{u}) &= 1 \\ \Phi_M(\mathbf{u}) &= e^{q_z h(\mathbf{u})}, \end{aligned} \tag{3.15}$$

where q_z is the scattering vector normal to the surface.

For these cases and the case of a random optical element, it has been shown[93] that the MIF propagating through multiple optical elements is the original incident MIF summed with the contribution of each element.

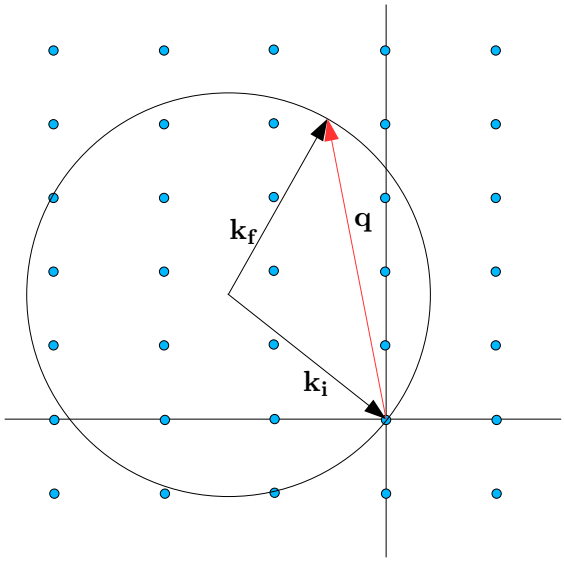


Figure 3.2: The Ewald construction can be used to visualize the 3D collection method. For example, the energy of the incident radiation may be altered slightly, thus altering the magnitude of $\mathbf{k_i}$ and sweeping the sphere through a Bragg point while collecting 2D data at each energy.

This result will prove very important in identifying partial coherence effects in the 3D reconstruction in Section 3.9.

3.2 Geometry

3.2.1 Ewald construction

To visualize the geometry for collecting a 3D CXD pattern, it is useful to recall the Ewald construction. For simplicity, Fig. 3.2 illustrates a 2D case. Each point displayed there is a Bragg point. $\mathbf{k_i}$ is the wavevector of the incident radiation and points to the origin of reciprocal space. Since we are neglecting inelastic scattering, the tail of the diffracted wave vector $\mathbf{k_f}$ is coincident with the tail of $\mathbf{k_i}$ and its direction sweeps out a sphere. All Bragg points that lie on the sphere can be measured by moving the detector to the appropriate location on the sphere. In the case of a finite crystal, there exists some finite width to the intensity distribution at a Bragg point–in fact, in accordance with [1.11] the distribution will be related to the Fourier

transform of the small crystal. If we allow our 2D detector to be represented by a 1D line in this case, an obvious method for collecting data in the third direction is by moving the plane of the detector along $\mathbf{k_f}$.

The triangle formed by $\mathbf{k_i}$, $\mathbf{k_f}$, and $\mathbf{q}$ is sometimes called the diffraction triangle. There are two obvious ways to move the place of the detector, which is always perpendicular to $\mathbf{k_f}$, through a Bragg point. First, changing the energy of the incident beam very slightly will change the radius of the Ewald sphere; however, it will also very slightly change the position of the diffraction on the detection plane. The second method is to very slightly rock the incident angle, changing the direction of $\mathbf{k_i}$, this will keep the diffraction centered in the detection plane, but the planes sampled will be out of parallel by a factor of sin $\Delta\theta$. We will use the latter experimental geometry, since the imaging experiment described here is not materially affected by this. If necessary, a geometrical correction may be applied after the reconstruction has been successful.

3.2.2 Diffraction geometry

In order to determine where to place the detector, a change of coordinates from the laboratory frame, where $\mathbf{k_i}$ and $\mathbf{k_f}$ are naturally defined, to the sample frame is necessary. This is normally done in the convention put forward by Busing and Levy[18] and later extended to allow the experimenter to control the incident and exit angles at the sample[75].

In the present experiment, a 4-circle diffractometer was used. The relevant angles are: 2θ, θ, ϕ, and χ. χ is a rotation about $\mathbf{k_i}$. θ is the incident angle. ϕ is a rotation about the sample normal. 2θ is in the direction of θ. A further two angles are provided by the goniometer head and are used to align the sample so that it is perfectly flat with respect to the axis. 2θ is in the direction of θ and is constrained by $\mathbf{q}$ in the following way:

$$|\mathbf{q}| = 2\frac{2\pi}{\lambda}\sin\, 2\theta/2. \tag{3.16}$$

As we will show later, a combination of 2θ, θ, and χ will position the detector near a $\{1\bar{1}1\}$ Au Bragg point and ϕ will allow the selection of individual crystals from among those illuminated.

3.3 Detection

A CXD imaging experiment results in a large amount of data. As discussed above, the geometry lends itself to the use of a 2D detector. The two primary

choices for hard X-ray experiments are image plates and CCDs. Image plates have a much greater dynamic range–up to 10^8 photons[89]–but exhibit non-linearity, *i.e.*, the detection efficiency tends to decrease as the exposure time increases. CCDs have a much smaller dynamic range, but exhibit good linearity with increasing photon numbers. Although there are further differences between these two technologies, pixel size, readout time, etc., the most important for CXD imaging is the linearity of the detector, since we seek to find a set of phases consistent with the amplitude of the diffracted wave at the detector.

Charge coupled devices were invented at Bell Telephone Laboratories in 1969 by Boyle and Smith[46]. Over the next five years, the technology metamorphosed from a device composed of metal plates to one built on a Si wafer. The original motivation for these devices was a form of bubble memory, but their promise for reliable imaging devices did not escape the interest of astronomers and astrophysicists who were at the time actively seeking an camera for the Hubble Space Telescope. The subsequent revolution in consumer imaging has exposed everyone to these devices.

In this Section, we will explore the method of operation of this solid state device, focusing on the characteristics that effect its utility as an imaging device. We will then explore the origin of noise in a CCD measurement, and explore analogies with the noisy simulations of Chapter 4. Finally, specifics concerning the detector used in the CXD imaging experiment described later in this chapter will be provided.

3.3.1 Scientific CCDs

CCD X-ray detectors work either by directly detecting X-ray energy photons, called direct read CCDs, or by detected visible light emitted by a phosphor coated screen or scintillator. In either case, incoming photons interact with some probability in the depletion region of the device and create a cascade of electron-hole pairs. The device is biased so that recombination is discouraged and the electrons move under an applied electric field toward a collector.

CCDs are typically made by first growing a highly doped substrate layer on Si wafer. The doping provides two benefits to the completed device: it is highly conductive, so it can be grounded as the CCD transfers charge, and it is essentially optically dead, because any electron-hole pairs created by incoming photons will quickly recombine. This substrate layer also serves to support the epitaxial Si layer, which will be p-doped, grown on top of it.

For a direct read CCDs, so-called deep depletion CCDs are used and have epitaxial layers up to 300μm thick.

Any photon with energy larger than the gap energy, E_g, will create one or more electron-hole pairs; however, for high photon energies, the average E_{e-h} will exceed E_g due to phonon emission. In fact, a wide range of semiconductor materials have E_{e-h} of about $3E_g$[6]. The energy gap of Si at 300 K is 1.12 eV[8] and the average energy required per created pair in Si is $E_{e-h} = 3.65$ eV[46].

One problem in early CCDs was the low charge transfer efficiency due to charge trapping at the surface Si/SiO_2 interface of the device. This was corrected through the use of buried channel CCD. In this case, the epitaxial layer is doped to create an n-type region near the surface, which reshapes the potential and leads to the collection of electrons below the surface of the device avoiding the Si/SiO_2 problem. This leads to a p-n junction inside the layer and therefore to a depletion region. In a CCD, the junction is reverse biased, so that the n-doped region is fully depleted when exposure begins. As photons interact with carriers in this region the trapping potential flattens–due to freed electrons–allowing other electrons to escape as a pixel becomes saturated.

Charge is transfered in a CCD array by shifting rows down the array into a horizontal register and then reading the register out into an analog to digital converter (ADC). This charge transfer is accomplished by sequentially depleting a pixel allowing it act as a sink for the charge in the neighboring pixel. In a buried channel CCD, charge transfer efficiencies of 99.9999% are attainable.

3.3.2 Noise in CCDs

The measurement of an X-ray is complicated by the fact that each incident photon generates a different number of electrons in the depletion region of the CCD, which makes the recovery of the number of incident photons extremely difficult. In practice, this problem is tractable with low count rates. For example, in the limit of low count rate, of order 10^{-2} photons/s/px, the Droplet algorithm[27, 53] can be used to recover the number of photons incident on a pixel. In the experiment described later in this chapter, we will have count rates as high as 45 photons/s/px, for which the recovery of the photon number has not been demonstrated. Instead, we will fit the data as it arrives from the ADC, that is a number of analog to digital units for each electron generated in each pixel of the CCD chip. We must therefore accept

that the error in the measurement depends on electron generation mechanism and electronics noise in addition to the inherent Poisson distribution of the diffracted photons.

The uncertainty in the amount of charge generated by a photon event is not characterized by a simple Poisson distribution, but rather by the Fano factor[29], F_a, which accounts for pair creation events being correlated to one another. The Fano factor is experimentally determined to be 0.1 in Si[46]. This quantity relates the energy of the incident photon to the root mean square (RMS) uncertainty in the number of electrons created by:

$$\sigma_e = \sqrt{E_{ph} E_{e-h} F_a}/E_{e-h}, \tag{3.17}$$

where the photon and the electron-hole creation energies, E_{ph} and E_{e-h}, are in eV. This statement is easily derived from the above statement of the statistics:

$$\begin{aligned} \sigma_e^2 &= N_e F_a \\ &= E_{ph} F_a / E_{e-h} \\ &= E_{ph} E_{e-h} F_a / E_{e-h}^2. \end{aligned} \tag{3.18}$$

At any finite temperature, T, there is some probability that an electron-hole pair will split and provide an apparent contribution to the signal. This phenomenon is called "dark current", because it generates electrons in the pixel even the detector is shuttered. The average rate of dark current generation can be calculated:

$$D[\mathrm{e}^-/\mathrm{s/pixel}] = C[\mathrm{e}^-/\mathrm{s/pixel/K}] T^{3/2} e^{-E_g/(2k_b T)}, \tag{3.19}$$

where C is a constant depending on the pixel size, room temperature dark current generation, and constants. Given the pixel size, P, and the "dark current figure of merit", D_{FM}, C can be evaluated so that [3.19] becomes

$$D = 2.5 \times 10^{15} P D_{FM} T^{3/2} e^{-E_g/(2k_b T)}. \tag{3.20}$$

This is the source of a bias level in our CXD experiment. The dark current can be greatly reduced by cooling the detector, but the level of the background will still vary in time and its distribution on the chip will be random.

The statistics of the dark current are Poisson. Therefore, $\sigma_e = \sqrt{N_e} = \sqrt{Dt}$ is the noise per pixel due to dark current in the CCD measurement. A background image can be collected by capturing an exposure of the same length with the X-ray source shuttered, but this image will also have noise

from the random nature of the dark current. Therefore, a background subtracted image of N pixels will still have a noise contribution of $\sigma_e = \sqrt{2DNt}$.

Finally, the electronics involved in converting and amplifying the analog signal from the CCD will contribute noise to the final measurement. This is generally referred to as $1/f$, or flicker, noise, so-called because of its power spectrum. The spectrum has a $1/f$ behavior, meaning that the power of the noise decreases by a factor of 10 for each factor of 10 increase in the frequency, f. This is believed to be caused by fluctuations in the tunneling current into the SiO_2 layer[5]. It is usual to filter out known components of $1/f$ noise in the electronics themselves [2].

In summary, the SNR of a pixel in a CXD experiment can be expressed as

$$SNR = \frac{PQt}{\sqrt{PQt(\bar{N}_e + \sigma_e^2) + (SQt(\bar{N}_e + \sigma_e^2))^2 + N_r^2 + 2Dt}}, \tag{3.21}$$

where P is the flux of photons diffracted from the sample, S is the flux scattered from anything except the sample, Q is the quantum efficiency of the detector, $\bar{N}_e$ is the average number of electrons generated by a photon, N_r is the read noise associated with the detector systems, and $2Dt$ is the dark current noise arising from a background subtraction. The first term in the denominator is just the expected variance due to the Poisson distribution of the signal. The second term is due to incoming photons scattered by alien objects in the path of the beam. In practice it is essentially impossible to exactly identify all such incoming photons. The third and fourth terms are the read noise of the electronics and the contribution from dark current in the chip, where it is assumed that both a background subtraction and a flat field correction have been applied.

3.3.3 Princeton Instruments CCD X-ray detector

A typical CCD detector system, such as the Princeton Instruments PI-LCX 1300 with ST133 controller, will have a depletion region about 50 μm deep in each of approximately 10^6 20×20 μm^2 pixels. This detector is a front-illuminated, deep depletion device fitted with a Be window to prevent visible light from interfering with the measurement. A plot of quantum efficiency (QE) against photon energy is shown in Fig. 3.3. To reduce the impact of dark current, the chip is thermoelectrically cooled by means of a Peltier junction that is water cooled, giving an operating temperature of about $-50°$C.

CCDs are graded according to the number and type of defects they ex-

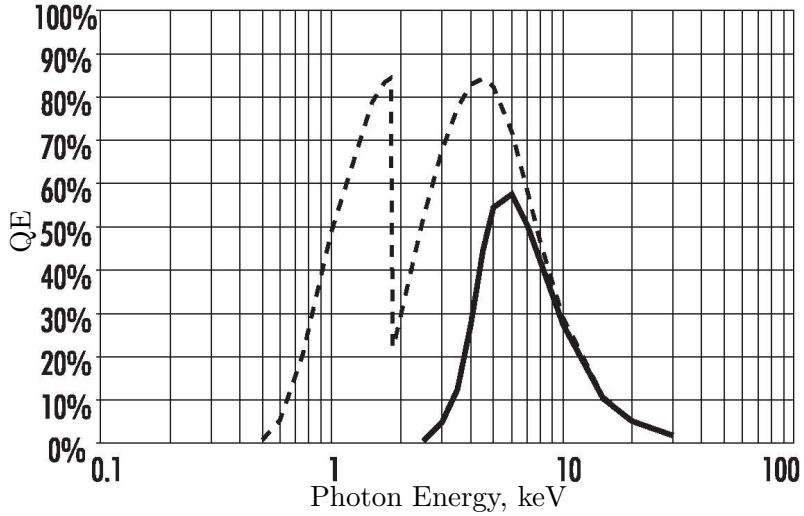


Figure 3.3: Plot of quantum efficiency versus photon energy demonstrating the sensitivity of a deep depletion CCD. The dotted line is the native sensitivity and solid line includes the absorption from the Be window. Figure from Ref. [3].

hibit. Defects range from traps, which collect perhaps 100 e^-, to dead pixels, which show no response to photons, to bright pixels, which accumulate more charge per photon than average. More serious defects sometimes manifest themselves, for example, a column of pixels failing to operate due to a short in the charge transfer path. Variations of the size of pixels may be corrected by applying a flat-field correction, which entails the collection of a data image and a second exposure of the same length and scaling each pixel in the former by the ratio of the mean to the dark current signal in that pixel in the latter. If a bright field image can be acquired, it may be used in this way to account for the variation in sensitivity from one pixel to the next, but creating the requisite diffuse illumination is difficult with X-rays and the sensitivity is known to change with increasing radiation dose. The chip used in the experiment here is labeled grade 2, with grade zero being the best available and grade 3 generally unsuitable for scientific instrumentation. Descriptions of the quantities and types of defects allowed in a particular grade are documented in Ref. [4].

The CCD detector in this experiment was an EEV 1152×1242 pixel array with 22.5 μm^2 pixels. The full well depth is about 60 photons with each photon event causing 700-1000 analog to digital units (ADUs) to be generated. A high speed shutter is used to protect the detector when it is not in use and while charge is being transferred out of the CCD array. The chip itself operates in a small evacuated region within the housing. This camera

was attached to a vacuum pump and the pressure within its chamber was reduced to less than 10 mtorr before each experimental run. The vacuum is primarily to protect the chip from damage, for example, since it is cooled to -50° C, any adsorbed water on the chip will freeze and possibly damage it as the chip is cooled from room temperature to its operating temperature.

3.4 Experiment and Sample Preparation

To test the algorithms of Chapter 2, we performed experiments the high temperature shape of Au crystals. Heyraud and Métios[40], previously measured the size of facets on the equilibrium crystal shapes of Au, to deduce relative free energies of facets. Their experiment was conducted under ultra high vacuum conditions and included the creation of the sample by depositing and annealing a thin film grown on pyrolytic graphite. In analogy to this experiment, we prepared 1000 Å Au films in a vacuum deposition chamber on 1 cm^2 pieces of a Si(001) wafer, whose native oxide was left intact to prevent strain in the film, which might be caused by either by the lattice mismatch of Au and Si or the creation of AuSi compounds at the interface.

The Au coated Si samples were scribed and broken to yield squares of approximately 2 mm^2. These samples were then placed in the chamber shown in Fig. 3.4. The top and bottom were machined from a soft ceramic, and cured to improve their mechanical properties. Four grooves were machined into each piece creating a square of approximately 8 mm^2 in the center of each 1.5 cm^2 ceramic slab. The sample was placed in the center of this square and the walls slide into place. 1 cm^2 Quartz slides coated with a Au film were used as sides for the chamber. A Cr wetting layer was used in the growth of the Au on Quartz samples. When positioned correctly, the slides enclosed the sample and served as support for the top slab. These materials were chosen so that they would survive the annealing of the sample, which was performed at up to 1050°C, slightly below the melting point of Au, 1064°C.

Once the chamber was assembled with the sample at its center, it was placed in a programmable oven. Many heating procedures were carried out, but the final result was dependent only on the total time at high temperature. A typical cycle would be: place the chamber in the oven, heat to 1000°C at a rate of 5.5 degrees per minute, dwell at 1000°C for 18 h, stop heating. The sample was either the 1 cm or 2 mm Au on Si system described above. In the case of the small sample, after its removal from the oven, it was mounted on a ceramic post and placed in the *in situ* heater shown in

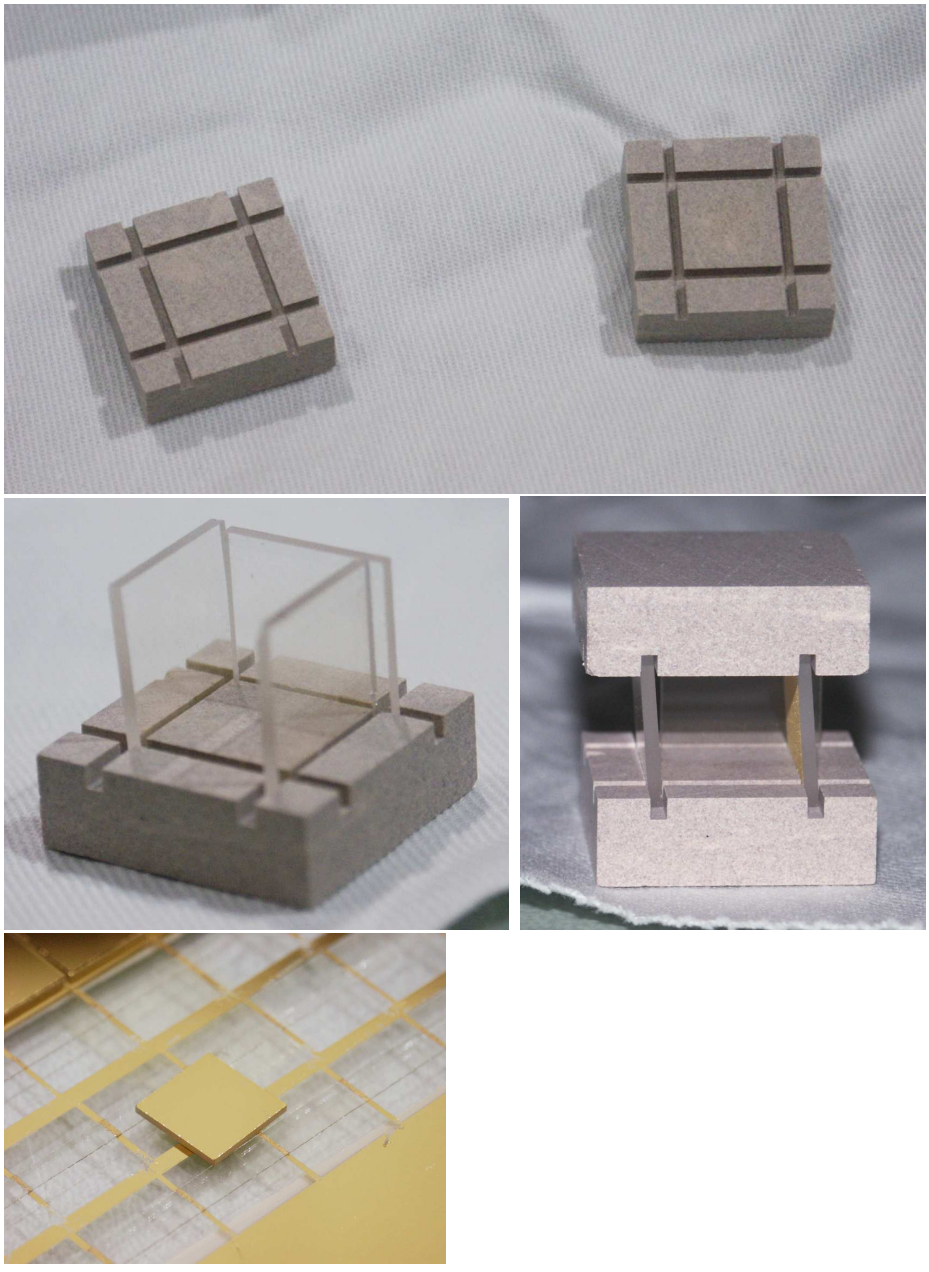


Figure 3.4: Top: disassembled chamber made from machinable ceramic. Middle left: Au slides(below) are used as sides. Middle right: the assembled chamber with the fourth wall removed to reveal the interior. Bottom: Au coated Quartz slides with a Cr wetting layer are used as sides for the chamber.

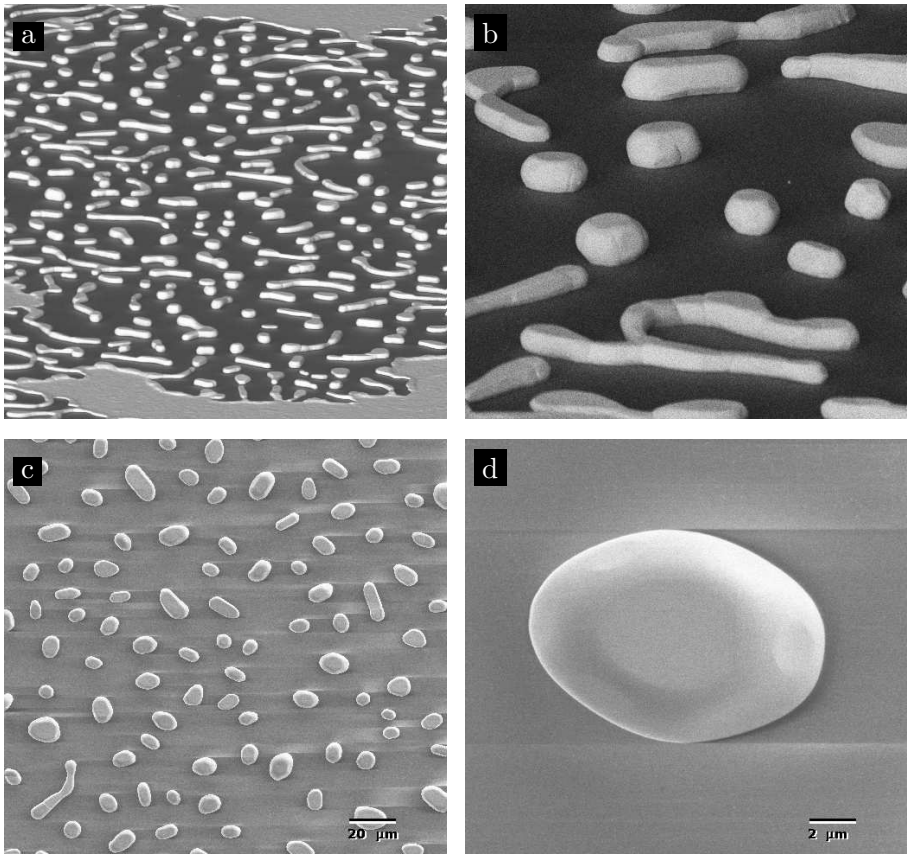


Figure 3.5: SEM micrographs of Au on Quartz samples acquired after the CXD experiment. As the film dewets, long "rivers" form as seen in a and the closeup in b. The sample in c was heated for a longer time at high temperature and most of the rivers have coalesced into isolated crystals like the one in d.

Fig. 3.6. This arrangement was used to collect CXD from the sample at high temperatures. The original 1 cm Au on Si and the Au on Quartz were measured at ambient temperature of the X-ray hutch. SEM micrographs are shown in Fig. 3.5.

The two piece heater assembly for the smaller sample was comprised of the base, which fit precisely into a stock goniometer head, and the body topped by the heater, a halogen light bulb, both machined from stainless steel. A thermocouple was threaded into a ceramic post and the sample was affixed to it and the post by means of a ceramic adhesive. Great care was taken to keep the sample perpendicular to the post, as this simplified measurement. The K-type thermocouple was then fed into the base, with each wire in a separate insulated channel to avoid electrical contact. The post was held in place by a set screw. The body had entrance and exit windows through which the X-ray beam may pass. Kapton covered the windows to prevent excessive convective cooling while still permitting the X-ray beam to pass through the heater. The heat source was a 150 W light bulb glued into a Au-coated parabolic reflector with a ceramic adhesive. The final height of the sample was chosen so that the sample was in the focal spot created by the parabolic reflector, which was secured to the body by means of six clamps. The most likely failure mode of the heat source was mechanical failure between the bulb and the reflector. To mitigate this, an Al heatsink was attached to the leads of the bulb, cooling the contacts and lengthening its lifetime. One of the contacts was electrically isolated from the heatsink by means of a Mica washer placed between the heatsink and the Cu connector. Although the device was low voltage, less than 15 V, a protective mesh guarded the higher voltage contact for safety, while the other–and the heatsink–was grounded. This device allowed us to heat a sample *in situ* near the melting point of Au, 1337 K.

3.5 CXD Data

Placing one of the three samples described above in the X-ray beam and moving the detector to a $\{11\bar{1}\}$ Bragg point yielded an intensity distribution similar to those in Fig. 3.7. Since the intensity distribution is related to the square of the Fourier transform of the crystal shape, we can identify the prominent features of the patterns. Each flare corresponds to a facet in the crystal, which are recognized as truncation rods[74]. The modulation of these flares is indicative of interference created by parallel facets of the crystal. The latter relation allows the estimation of the size of the crystal

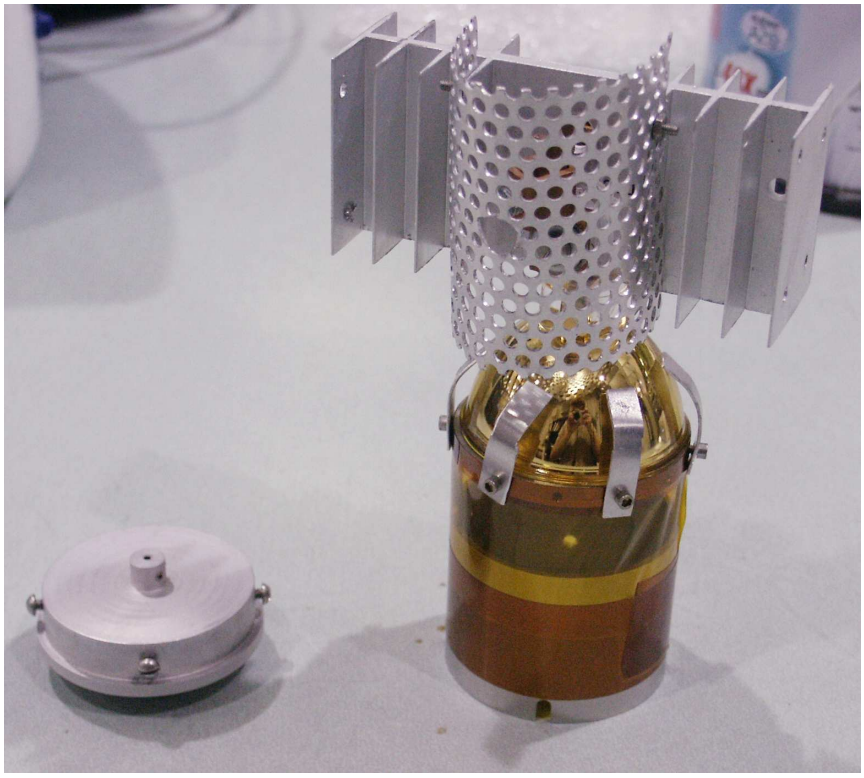


Figure 3.6: Left: The base of the heater which is connected to the goniometer head of diffractometer. A ceramic post is inserted in the base to place the sample in the focus spot of the heater and in the path of the X-ray beam. Right: The heater assembly that sits on the base, shown left. The heat source is a 150 W halogen light bulb–model OSRAM HLX 64635)–with a Au coated parabolic reflector. A heat sink is attached to the leads of the bulb to mitigate failure of the ceramic used to cement the bulb into the reflector. Power is supplied to the bulb through the Cu contacts, one of which is seen here behind the protective mesh. The X-ray beam enters and exits through the windows in the body. Kapton tape covers these windows to limit convective cooling of the sample.

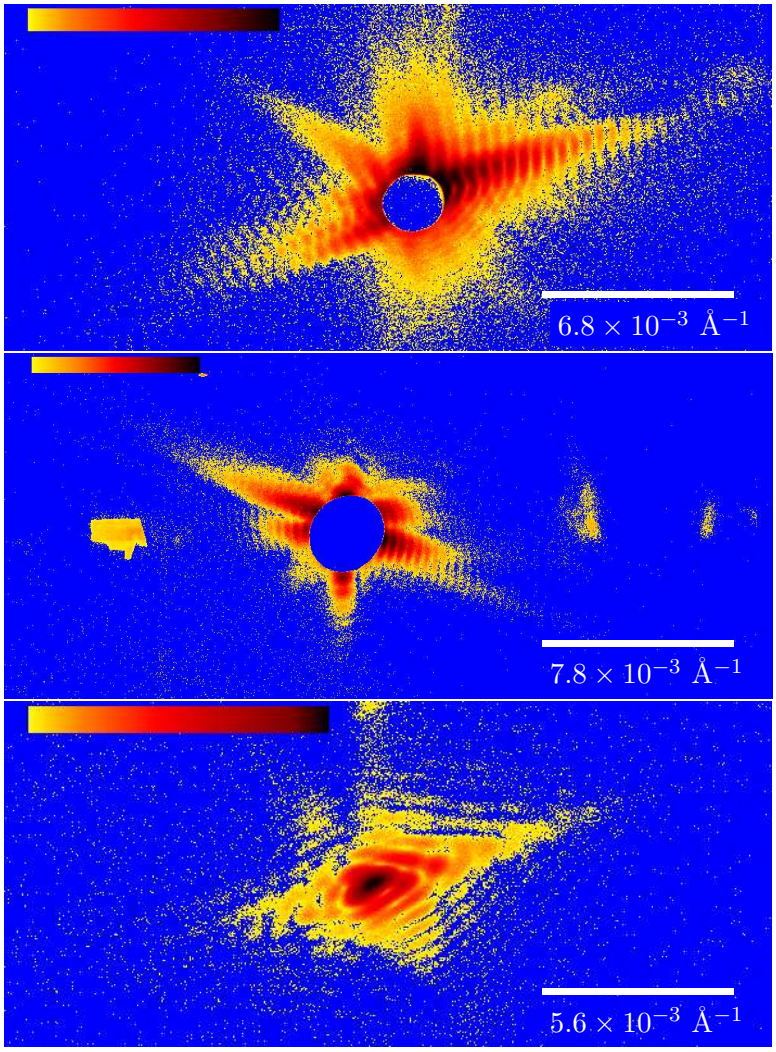


Figure 3.7: CXD patterns near a Bragg point from the three sample described in the text. Top: Au on Quartz. Middle: Au on Si, heated *ex situ*. Bottom: Au on Si at 950°.

according to

$$\Lambda_i = \frac{2\pi}{K_i q_{px}} = \frac{2\pi}{K_i k \sin\theta} = \frac{2\pi}{K_i (2\pi/\lambda)(\Lambda_{px}/L)} = \frac{\lambda L}{K_i \Lambda_{px}} \tag{3.22}$$

where the Fourier relationship between the characteristic length scale of a real-space feature, Δx, and its conjugate in reciprocal-space, Δq: $\Delta x \Delta q = 2\pi$ has been used, K_i is the period of the modulation in pixels along the ith fringe, λ is the wavelength of the radiation, θ is the angle subtended by a pixel in the detector, L is the sample to detector distance, and Λ_{px} is the spatial dimension of a pixel.

Rocking the sample θ showed that the diffraction was essentially centrosymmetric about the Bragg point. As mentioned previously, slight asymmetries may be the result of either strain or failing to find the Bragg point precisely. Occasionally, the region in the immediate vicinity of the Bragg point was so intense that it was more convenient to measure the fringes during a long exposure and the center in a shorter one. The method used to combine the two measurement will be described in Section 3.8.3.

3.6 CXD Data Handling

Princeton Instruments (Now a subsidiary of Roper Scientific.), the maker of the CCD detector control system used here, provides a Microsoft Windows™-based control program called WinView™. This program allows the user to create a sequence of accumulations at a constant exposure time grouped into a single frame. This is an easy way to take many short exposures at a particular data point when a single long exposure would saturate the detector. The trade off is that for N accumulations, a factor of $\sqrt{N}$ times the dark current noise will be present in a background subtracted image. The controller provides a transistor-transistor logic (TTL) pulse indicating a shutter event. This TTL pulse was used as the monitor in a scan of θ to automatically move to the next position when a frame had been collected.

WinView™ stored the data in a file whose structure is publicly available. Within the data file, the number of ADUs per pixel and a few parameters pertaining to the controller were stored. We extracted this data from the file and stored it as double precision floating point numbers for manipulation by our programs. Our data file format is a simple structured format we call an sp4 array and so the program is called **`2dspetosp4`**. The first four bytes are an integer specifying the version of the file format. The next 2048 bytes are a character string which may contain information about the creation

of the array. The next four are an integer declaring the array to be real or complex. A complex array stores the complex number contained in pixel one by placing the real part of the number in the first eight bytes (one double) and the imaginary in the next eight. This is followed by an integer number of dimensions, which may take the value of 1, 2, or 3. The next four bytes contain a long integer which is the total number of data elements in the array. The next element is the physical dimension, as a long integer, of the array, which occupies one to three times four bytes. Finally, the data itself is written into the array.

Background subtraction requires two arrays. The data which should be subtracted and an identical frame taken while the X-ray beam was shuttered. Unfortunately, cosmic rays will penetrate the Pb lined hutches at the APS and so occasionally appear in a "background" image. A primitive threshold exists within the background subtraction program for the rejection of these events. For simplicity, a configuration file must be passed to **`flatten`** that contains the location of the data, the location of the background, the section of the background image to be used–in case a full frame background has been acquired and the data has been collected in a smaller region of interest–a scaling factor to account for different exposure times in the two array, and a threshold, to allow the user to specify that any pixel below the threshold should be set to zero or converted to its absolute value.

Further, two modes of background subtraction are possible. In the first mode(**`flatten`**), a simple point by point subtraction is performed using the values in the configuration file. In the second mode **`flatten2`**, an initial subtraction is applied as above, followed by the calculation of a histogram of the background subtracted data. A second subtraction is then performed to move the background peak so that its mean value is 0 ADUs. This is useful because the amount of dark current generated in the camera is, in practice, dependent on the flux incident on the chip, so different scaling factors are optimal for different frames in a 3D measurement.

A program exists to perform a flat-field correction **`ffcorrect`**, but is rarely used in practice. Since we do not have ready access to a diffuse source, it is usually not possible to obtain an image when the flux density is uniform across the detector. This correction can be made using a shuttered frame and will in this case correct for the deviation in physical size of the pixels, but not their sensitivity, for example, due to variations in doping concentration. The correction is accomplished as described above using the mean and the dark current signal in the dark image. At this time we also perform any other correction that may be necessary to compensate for experimental conditions,

for example, in Section 3.8.3 we will discuss the merging of the center of a short exposure with an exposure where the central region of the pattern is obscured by a beamstop. Since the earliest form of the fitting programs did not allow a complex real-space object–the real-space density was constrained to be purely real forcing the reconstructed reciprocal space amplitude to be centrosymmetric–some CXD patterns were symmetrized via **`2dsymmtest`**. The symmetrization was accomplished by my minimizing a χ^2 function , see [4.4] and [3.25], where the difference is between two points at a given radius with an angle of π between them. This function is minimized at the point about which the array is most centrosymmetric. A correlation coefficient is also calculated, which is maximized when χ^2 is minimized.

Finally, we must convert the intensity measurement into the modulus constraint required by the algorithm. This was done by simply calculating the square root of the ADUs recorded in a pixel and using that value as the magnitude of the complex amplitude. After this was completed, the data was converted into a complex sp4 array and fitting began.

3.7 Methods and programs

This section will describe in detail the methods and utilities used to fit the data in the following sections. A more complete description of the programs written by the author appears in Appendix B and should serve as a kind of guide to their use. The programs are written in the C programming language and developed under the GNU/Linux operating system. They are standards compliant, so and so could be used under another operating system. They have additionally been tested under the Cygwin program, which provides a UNIX™-like API on Microsoft Windows operating systems.

We will begin with the methods generally employed to generate the real-space constraint needed by the iterative algorithms. Following that is a description of the invocation of the fitting programs themselves. Lastly, the programs used to display and analyze the resulting reconstructions will be presented and their use described.

3.7.1 Support and Fourier modulus constraints

Before invoking the algorithms discussed in Chapter 2, a real-space support array must be generated. For the purpose of using our fitting program, a support region is defined as the region of the array where the pixel value is above some threshold, commonly chosen to be 0.5. An array of this kind was commonly created in one of three ways: automatically from another array,

by generating an array with the appropriate support from an estimate of the particle size, or by hand in an image manipulation program. Automatic generation can be accomplished by thresholding the autocorrelation, which is easily generated by squaring the measured intensity and back transforming. Unfortunately, the appropriate threshold will be highly specific to the input data, so even this was not truly automatic. The second use of thresholding–as in **2dspt4tocon**–was creating a constraint from an existing estimate of the particle's density. This was more commonly applied to tightening a constraint as fitting progresses.

It was relatively easy to obtain approximate crystal size from the diffraction by either calculating the autocorrelation function and using half the area, or volume in 3D, as a support or counting the number of pixels per fringe and dividing the array size by that number. Once the approximate size was known, a program exists to create an arbitrary polygon for use as a constraint, called **2dpoly**. If a more exotic support was desired, an image manipulation program may be used to draw a support. This was occasionally a more useful way of using the autocorrelation. In this method, the image must be saved as an 8-bit bitmap (ppm) image and then converted to an sp4 array via **2dppmtosp4**.

The last preparatory step required for the modulus constraint was to "wrap" the complex array so that the center pixel lies in the first element of the array in preparation for application of the DFT. We made use of the FFTW libraries, which provide a fast Fourier transform algorithm for arbitrary, *i.e.*, not limited to power of 2, array sizes. In general, it was easiest to use even array sizes as this simplifies the wrapping and unwrapping procedure. Utilities such as **2dcrop** and its derivatives–**2dquickcrop**, **2dcroppad**, etc.–have been written to alter the array size.

3.7.2 Fitting programs

It is useful to calculate a numerical quantity that can be used to compare fits to the original data and one another. To this end, we define two error metrics and measure of the reproducibility of the two best fits. The error metric in reciprocal-space is defined:

$$\chi^2 = \frac{\sum_{i=1}^{N} (|A_i^{calc}| - \sqrt{I_i^{meas}})^2}{\sum_{i=1}^{N} I_i}, \tag{3.23}$$

Table 3.3: Summary of the modes available in **2dmultifit8**. For HIO algorithms, the first parameter is β and the second is ϵ. In ER modes, the parameters are not used unless specified.

type	mode	description
gs		Gerchberg-Saxton
hio	4	Millane's HIO(mHIO)
	5	ER enforcing the density to be real positive by rotating the magnitude of the amplitude onto the real axis.
	6	mHIO which alters the density according to the values in the real-space support
	7	ER enforcing support and allowing the real-space phase to vary according to the phase of the support array
	8	ER allowing a phase gradient to exist within the real-space density
	9	mHIO allowing real-space phase to take on one of three values: 0, $\pm 2\pi/3$
	10	ER allowing real-space phase to take on one of three values: 0, $\pm 2\pi/3$
	11	mHIO enforcing the density to be real positive by projecting the amplitude onto the real axis
	12	ER enforcing the density to be real positive by projecting the amplitude onto the real axis
	13	mHIO with only a support constraint
	14	ER enforcing support and real density
	15	HIO, Fienup variant with support and positivity
	16	ER, support only
mov	1	moves the center of mass of the real-space reconstruction to an offset from the center of the array given by the parameters, horizontal first then vertical
vor	1	Removes vortices as previously discussed (Sec. 4.5) the parameters are the tolerance and the radius
	2	Same as 1, but tries to find vortices by finding near-zero points and calculating phase wrap, the first parameter is unused

where A_i^{calc} is the amplitude of the ith pixel in the current iterate in reciprocal-space, I_i is the measured intensity, and the sum is taken over the N pixel detector. A similar metric is defined in real-space,

$$\chi^2_{\mathrm{Re}} = \frac{\sum_{x=1}^{N} |f_n(x) - f'_{n-1}(x)|^2}{\sum_{i=1}^{N} I_i}, \tag{3.24}$$

where $f_n(x)$ is the current best estimate of the object, and $f'_{n-1}(x)$ is the result of applying the reciprocal-space constraint to $f_{n-1}(x)$. Finally, we quantify the reproducibility by

$$\xi^2_1 = \frac{\sum_{x=1}^{N} |f^{(1)}(x) - f^{(2)}(x)|^2}{\sum_{i=1}^{N} I_i}, \tag{3.25}$$

where $f^{(1)}(x)$ is the best fit and $f^{(2)}(x)$ is the second best fit. For [3.23]-[3.25], a value of 0 indicates a perfect match, and we interpret the square root of these quantities as an average error or disagreement per pixel. These quantities are all discussed in greater detail in Section 4.1.1.

The basic invocation of the fitting program was with only a single command line parameter: a configuration file. This was a structured plain text file read sequentially by **`2dmultifit8`**. The first line contained the total number of fits to perform and the number to keep separated by spaces. The next three lines contained the name of the real-space array, the name of the reciprocal space array, and the number of cycles per fit that follow in the file. The name must contain the full path and was used to name the reconstructions by appending a string, for example, `mysupport.002.realfit` would be the second best real-space fit to the data with real-space support array "mysupport". The remainder of the file was composed of two line segment declaring the algorithm, mode, iterations, and parameters to be used for that cycle of the fitting. The first of the two lines must be either 'gs', 'hio', 'mov', 'vor': 'gs' was the historical Gerchberg-Saxton algorithm, 'hio' included ER and HIO, 'mov' moved the real-space reconstruction, and 'vor' invoked the vortex removal algorithms. The second line had four space-separated numerals: the number of iterations(integer), the mode(integer), first parameter(double), second parameter(double). Table 3.3 contains a summary of the modes available in the current version of **`2dmultifit8`**. All lines preceded by **`'#'`** were assumed to be comments and ignored. The

real-space output was saved just before the real-space constraint was applied and the reciprocal-space was saved just before the modulus constraint was applied.

Once invoked, **2dmultifit8** would load the specified constraint arrays into memory and allocated memory equal to the number of fits to keep specified in the configuration file plus one times the size of the array. Additionally, one more array's worth of memory must be allocated to perform iterations of HIO. For each fit, a set of random numbers was generated to be used in calculating the initial estimate of the reciprocal-space phases. These phases are assigned to the support array and it was Fourier transformed to start the fitting. An equivalent method was to begin in real-space with a constant phase and random density; both have been used with no difference in fitting efficacy observed. As the fit proceeded, two error metrics–the real- and reciprocal-space defined by [3.23] and [3.24]–were written to disk in a file called `recon.log`. This was used to track the progress of a fit and judge the utility of the fitting recipe being used.

2dmultifit8 would also take up to two additional command line parameters. The default behavior was to use the system clock to seed the standard C uniform random number generator drand48(). If two parameters were passed on the command line, the second was assumed to be the seed. This is very useful for repeating the result of a particular fit, especially if a mistake was made in the original configuration file. **2dmultifit8** wrote the seed used to create the random set of starting phases for each array in the sp4 array header for this purpose. The final optional parameter was an integer, N_f. When specified, **2dmultifit8** would generate a gray scale bitmap (pgm) once every N_f iterations. This bitmap contained four images: top left was real-space amplitude, top right was real-space phase, bottom left was reciprocal-space amplitude, and bottom right, was reciprocal-space phase. This feature was very useful for monitoring the progress of the fit.

The program (**2ddiffmap**) for applying DM was markedly different. Its configuration file consisted of seven plain text lines: the number of fits to perform and the number to keep, the real-space support array, the reciprocal space constraint, the number of iterations, β, γ_1, γ_2. **2ddiffmap** wrote only the real-space result, since the language in which it is naturally discussed does not operate in reciprocal-space. Therefore, the reciprocal-space reconstruction was obtained in analogy to that obtained by **2dmultifit8**, but the real-space output of **2ddiffmap** differs in that there was a final application of the real-space constraint. As described in Chapter 4, only a support constraint was applied in this program.

3.7.3 Plotting programs

Various utilities have been written for creating bitmap (ppm) images from sp4 data arrays. These programs created an 8-bit gray scale image of the magnitude of the amplitude and, if the array was complex, an 8-bit color image of the corresponding phases. The color image was generated in the HSV color scheme, with saturation and value set to unity and hue varied according the to the phase, which lay within $[-\pi, \pi]$. In this way, the phase was mapped onto a color circle with π mapping to red, $2\pi/3$ to yellow, 0 to cyan, $-2\pi/3$ to magenta, and $-\pi$ to red. The phase was thus represented continuously rather than having a black/white transition at π, as might have been the case were the phase displayed in gray scale. Reciprocal-space arrays were unwrapped with **`2dinvert`** prior to plotting.

The simplest plotting program was **`2dplot`**, which requires an integer between 0 and 255 and the name of the sp4 array to be plotted. The integer was the number of levels desired in the amplitude image, with 0 meaning the full 8-bit range. **`2dplot`** automatically found the amplitude range in the input array and used the range to scale the array to the desired number of levels in the output image. A scale bar was automatically placed in the upper left hand corner of all images with width greater than 256 pixels. This was important to minimize the variations in printers and monitors when examining the data. The scale bar started at the leftmost pixel and corresponded to the value 0 in the bitmap image. While displaying many images, especially those in reciprocal space, a user-defined threshold was sometimes used making it appear as though the scale bar started away from the leftmost pixel. The reciprocal-space images tended to have very large dynamic range, so it was useful to plot them as the logarithm of the intensity. A separate program, **`logscale`**, did this by finding the lowest value in the array, normalizing that to unity and calculating the the logarithm base 10. Zero valued pixels in the intensity were not altered.

Occasionally, one wishes to do a side by side comparison between two arrays with different amplitude ranges. One way to do this was through the program **`2dmanplot`**, which allowed the user to specify the minimum and maximum values. Values in the sp4 array below the minimum were set to zero and those above the maximum were set to the maximum specified level. Alternatively, **`sp4scale`** could be used to multiply every element in an array by a supplied constant, yielding a new array.

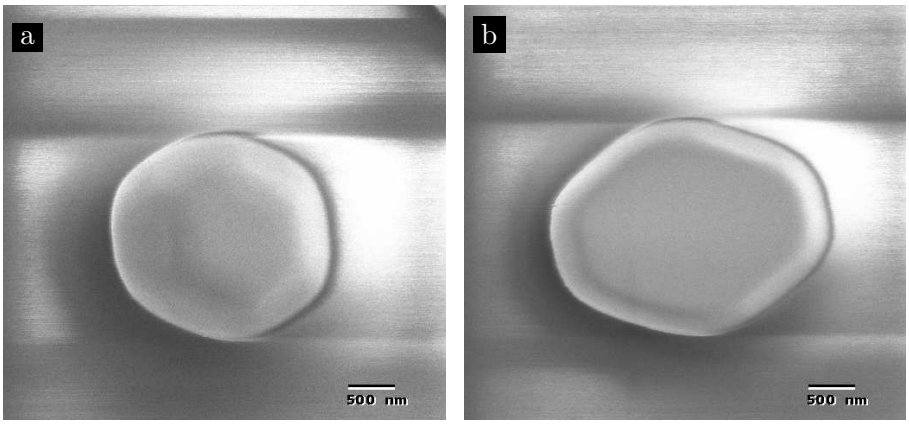


Figure 3.8: SEM micrographs of Au on Si samples heated at 1050° C.

3.7.4 Analysis programs

Several programs were of practical import when analyzing the reconstructions. **`2dreprod`** calculated the χ^2 quantity ([3.23]-[3.25]) of the two arrays named on the command line, which was used to quantify the agreement between the two arrays. The program centered each array by calculating the Fourier transform and extracting the linear gradient from the phase of the transform. Using the new centered arrays, the point by point sum of squared differences was calculated. Since a twin image may have reconstructed, one array was rotated by π and the procedure was repeated. The lowest of the two numbers was reported where a value of 0 would be a perfect match. In some cases one is interested in comparing two arrays which may not have the same total integrated intensity so **`2dnormreprod`** was written to normalize the arrays before this calculation.

Another useful program is **`2dsp4toascii`**, which read a section of an array and output the magnitude of the amplitudes and phases to a plain text file. This was useful for generating quick plots of a 1D slice through a 2D object.

A host of other programs exists and the reader is invited to scan the program listing in Appendix B. Names are generally descriptive and almost all program print usage information when called incorrectly.

3.8 2D Results

3.8.1 Sample

Figs. 3.9-3.11 show the results from reconstructing CXD data from a Au on Si sample. The sample was prepared by depositing 1000 Å of Au onto a 1 cm^2 piece of Si with its native oxide intact. It was then placed in an oven inside the cell previously described for 10 h at 1050° C and held at 850° C for an additional 10 h. The sample was cooled from 850° C by turning off the power to the oven and opening it when the temperature dropped below 200° C. The cooling process took about 3 h. After cooling, the sample was placed on the diffractometer.

3.8.2 Experiment

For this experiment[77], we positioned the detector in the vicinity of $(11\bar{1})$ Bragg peak, assuming that the substrate normal direction was the same as the (111) direction of the crystal. This was not unreasonable, since (111) was the lowest energy face and so was likely be the preferred face for the substrate interface. By rotating the sample about its normal, an individual grain could be selected, since the non-specular {111} peaks are randomly oriented about the normal. A small slit, $\sim 30\ \mu$m $\times 100\ \mu$m, was used to limit the number of grains illuminated. The slit was large enough to allow many coherence volumes in the beam to pass through, so it did not contribute to the coherent illumination of a particular crystal, rather it was the size of the crystal–small compared with the coherence volume–that assured coherent illumination. The transverse coherence lengths given in Section 3.1.1 apply here, but the monochromator was Si(111) using a (111) reflection rather than a (333) reflection as was assumed in Section 3.1.2, so the longitudinal coherence length was decreased from that estimate. For a Si(111) reflection at 7.5 keV, we have $w_d = 5.5 \times 10^{-5}$ rad, from [3.6], and so $\xi_{\|} = 0.57\ \mu$m, from [3.4].

For micron sized single crystals of a high Z material like Au, the exposure time was short, about $1 - 2$ s, so the data collection time was dominated by the data transfer rate of the CCD detector system, which required about 15 s to read a full frame image. To expedite this process, we placed a beam stop at a known position in front of the detector. The beam stop was a piece of Pb/Sn solder formed by dropping small amounts of molten solder into a large container of water. It was then sandwiched between a sheet of Kapton and a piece of Kapton tape. This assembly was affixed to stage so

that rough adjustments to the beam stop position could be made. With the beamstop in place, a short scan of θ was conducted about the suspected location of the Bragg point. The scan consisted of 11 points 0.02° apart and at each position five accumulations of 60 s were collected, for a total exposure time of 300 s. The beam stop was then removed and the scan was repeated, collecting five accumulations of 3 s, for a 15 s exposure.

3.8.3 Data handling

Fig. 3.9 details the images used to recreate the CXD pattern from these images. The original long exposure image, Fig. 3.9a, was background subtracted with a frame of the same duration captured while the X-ray shutter was closed, Fig. 3.9b. The large roughly rectangular area of intensity in Figs. 3.9a-b was attributed to a particle of dust sitting on the surface of the CCD chip. As can be seen in the final image, Fig. 3.9d, the background subtraction removed it. Similarly, the short exposure, Fig. 3.9c, was background subtracted (Short exposure background image not shown.). A program, **`bmstpcorrect2`**, was used to merge the two background subtracted images. **`bmstpcorrect2`** calculated a scaling factor to apply to the portion of the short exposure which was obscured by the beamstop in the long exposure. The scaling factor was given by the ratio of the deviations of intensity inside a user defined annulus in each of the images:

$$\frac{\sigma_{\text{long}}}{\sigma_{\text{short}}} = \frac{\sum_{n=1}^{N} (I_n - < I_{\text{long}} >)^2}{\sum_{n=1}^{N} (I_n - < I_{\text{short}} >)^2}, \tag{3.26}$$

where $< I >$ is the average intensity inside the annulus in either the long or short exposure. Naturally, the annulus should be a region close to, but outside the area of the detector affected by the beamstop. Finally, a linear interpolation between the two images was performed. The resulting image is shown in Fig. 3.9d. It should be noted that a threshold has been used in displaying each of these images, as can be seen from the scale bar in the upper right hand corner of an image, which starts at the leftmost edge of the image. Further, it is the logarithm of the intensity that is displayed.

3.8.4 Fitting

The data in Fig. 3.9d were cropped to reduce the time necessary for the FFT and to remove the scattering that appears to the right in the image.

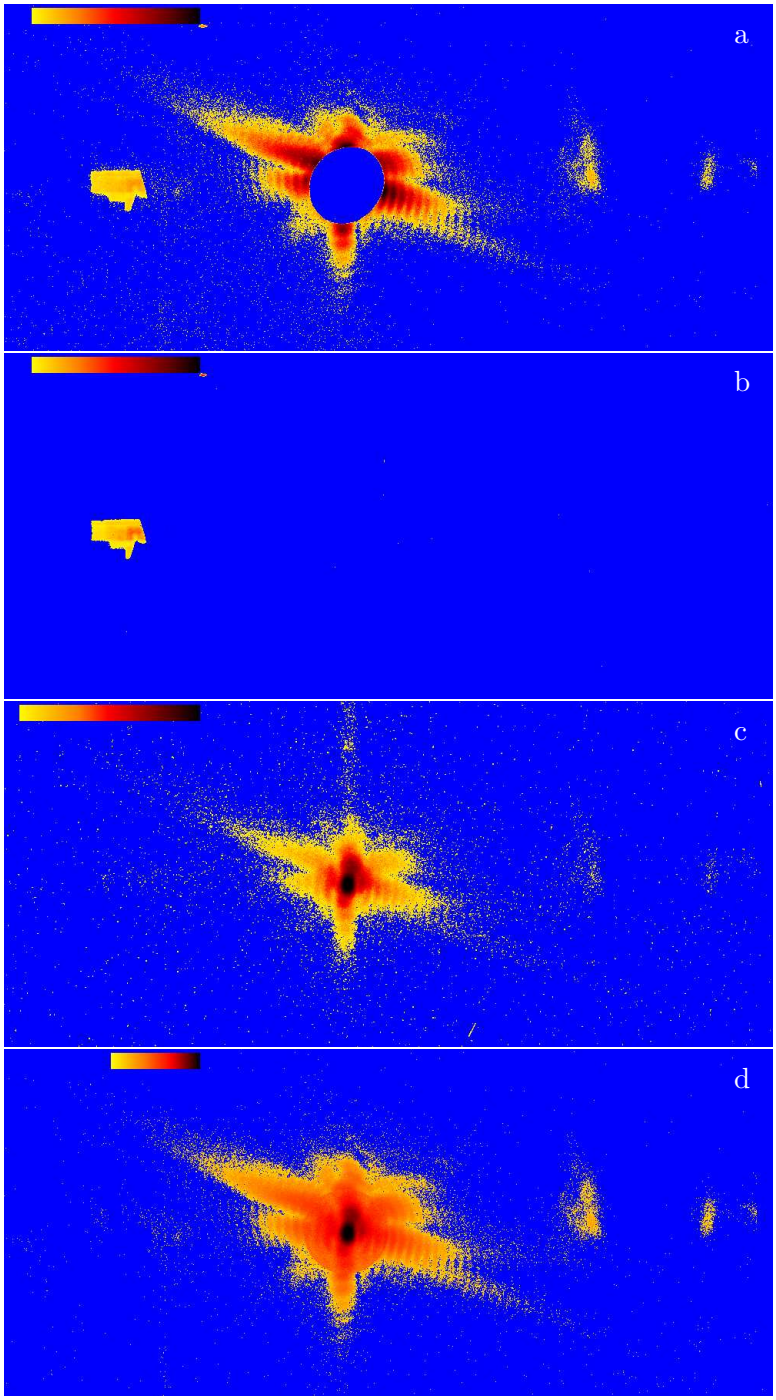


Figure 3.9: A long exposure, a, was taken to gather contribution from the edges of the pattern. Some artifacts, for example the bright roughly rectangular object in b, can be removed through background subtraction. A short exposure, c, captured the center of the pattern. a and c are merged to form the Fourier modulus constraint, d. The images are the logarithm of the intensity.

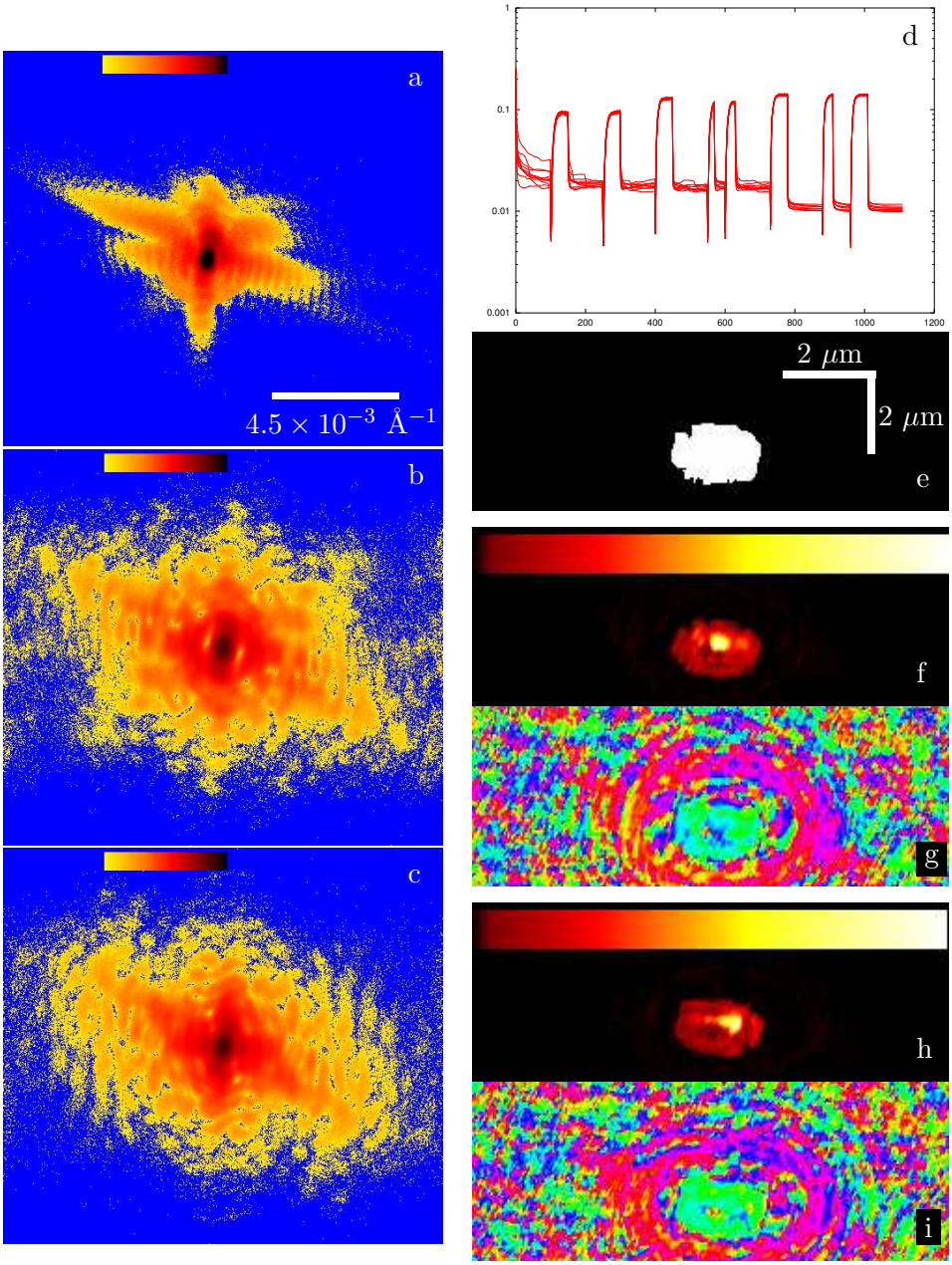


Figure 3.10: Results from fitting 2D data whose preparation is detailed in Fig. 3.9. a is the corrected intensity of the CXD measurement, while b and c are the two best fits to it. d is a plot of the error metric during the fitting, starting with ER and then alternating cycles of HIO and ER. e is the support created from an earlier fit. f and h are the magnitudes of the complex valued real-space density derived from b and c. The accompanying phase is shown in g and i. Images a-c are the logarithm of the intensity and e-i are linearly scaled.

Fig. 3.10a is the logarithm of the intensity of the array used as the Fourier modulus constraint. The fitting consisted of 10 runs, each with a different set of random starting phases. During each run, 17 different cycles of ER or HIO was performed. Fig. 3.10d is a plot of the real-space error metric against iteration for the ten fits.

In total, 1110 iterations of the algorithms were performed. The β parameter for HIO was varied between 0.45 and 0.95. The last three cycles of ER enforced only the support constraint, while the first six enforced both support and that the density be real positive. Relaxing the real positive constraint allows the reconstruction of an asymmetric Fourier modulus, which we have in this case. The support, Fig. 3.10e, was generated by thresholding an earlier fit of the same duration, except that the real positive constraint was never relaxed. Generally, a tight support was required when allowing the phase of the real-space object to vary to limit the degrees of freedom of the problem.

The two best resulting reciprocal-space fits are shown in Figs. 3.10b-c. The images in Figs. 3.10a-c have not been scaled identically, although the scaling for Fig. 3.10b and Fig. 3.10c is very similar. It is immediately obvious that there is far more intensity near the boundaries of the array in the reconstructed patterns than in the original. This is not an artifact of the scaling and will be addressed further shortly. The real-space best fit, Figs. 3.10f-g, corresponds to Fig. 3.10b. Fig. 3.10f is the amplitude of the reconstructed density and Fig. 3.10g is the phase of this complex amplitude. This is a linear plot that has been colorized according to the scale bar in the image. Similarly, the second best fit, corresponding to Fig. 3.10b, is shown in Figs. 3.10f-g. The difference between the two fits is calculated to be $\xi_1^2 = 0.2$, and their agreement to the CXD data is $\chi^2 = 0.0099$ and $\chi^2 = 0.010$, for the best and second best fit.

The scale bar shown in Fig. 3.10e has slightly different length in the two directions. This is due to the rectangular array–500×450 px–in which the fitting was performed. To calculate the amount of space spanned by a pixel, we use the Fourier relationship for discrete arrays: $\Delta x \Delta q_x = 2\pi/N_x$, where Δx and Δq_x are the amount of real- and reciprocal-space spanned, and N_x is the number of elements in the array in the x direction. An identical result holds for the y direction, defined to be up and down in the array. For this experiment and array size,

$$\Delta x = \frac{2\pi}{N_x k \sin\,\theta} = \frac{\lambda L}{N_x \Delta x_{px}} = \frac{(1.65\text{Å})(2.743\text{m})}{(500)(22.5\mu\text{m})} \simeq 400\text{Å} \tag{3.27}$$

is the the amount of real-space corresponding to a single pixel in the horizontal direction of the reconstructed density. The size in the vertical direction is $\Delta y \simeq 450$ Å.

The size of the reconstructed particle is well within the distribution of sizes known to exist on our sample from the SEM micrographs taken after the CXD experiment. Unfortunately, there is no way to know which particle on the sample we are collecting diffraction from, so we cannot compare the reconstructed shape to the projection of the known particle shape. The two best reconstructions do show similar features. The real-space reconstructions are twins, as can be seen be examining the fit and by the calculation of the irreproducibility, ξ_1^2. Both particles have a "notch", seen in Fig. 3.10f on the upper left, and one region of very low density and one of very high density. These are both unexpected, since the crystals should be solid, *i.e.*, without voids, and the density should be uniform, leaving the bright spot in the projected density unexplained. The complicated phase structure in the complex density is also difficult to explain and the two phase structures do not agree well.

3.8.5 Revised fitting

In order to get a better fit to this data, two modifications were made[77]. First, the data was symmetrized to encourage the reconstructed density to be real positive. Second, the data were multiplied by a radial Gaussian with half width of 41 px. The processed data are shown in Fig. 3.11a. The former should correct for being slightly off the Bragg point in the original measurement, since the 2D pattern containing a Bragg peak from a real density should be centrosymmetric. The Gaussian filter reduces the likelihood of aliasing. It is this aliasing effect that is believed to have caused the intensity seen above to appear in the reconstructed complex amplitude where none exists in the measured data. Multiplying the reciprocal-space pattern by a Gaussian has the effect of convolving the real-space density with the Fourier transform of that Gaussian, which is a Gaussian with half width $2\pi/41$ px. This is a common method for "smoothing" an image.

This processed data was then used to run 20 fits each of which was identical to those performed on the unprocessed data. The support was taken to be a square of size 100×100 px in an array of 464×426. Since the modulus constraint was already symmetric, ER utilized both the support and positivity constraints throughout. Fig. 3.11d is a plot of χ^2_{Re} against iteration. Figs. 3.11b-c are again the first and second best fits with the ac-

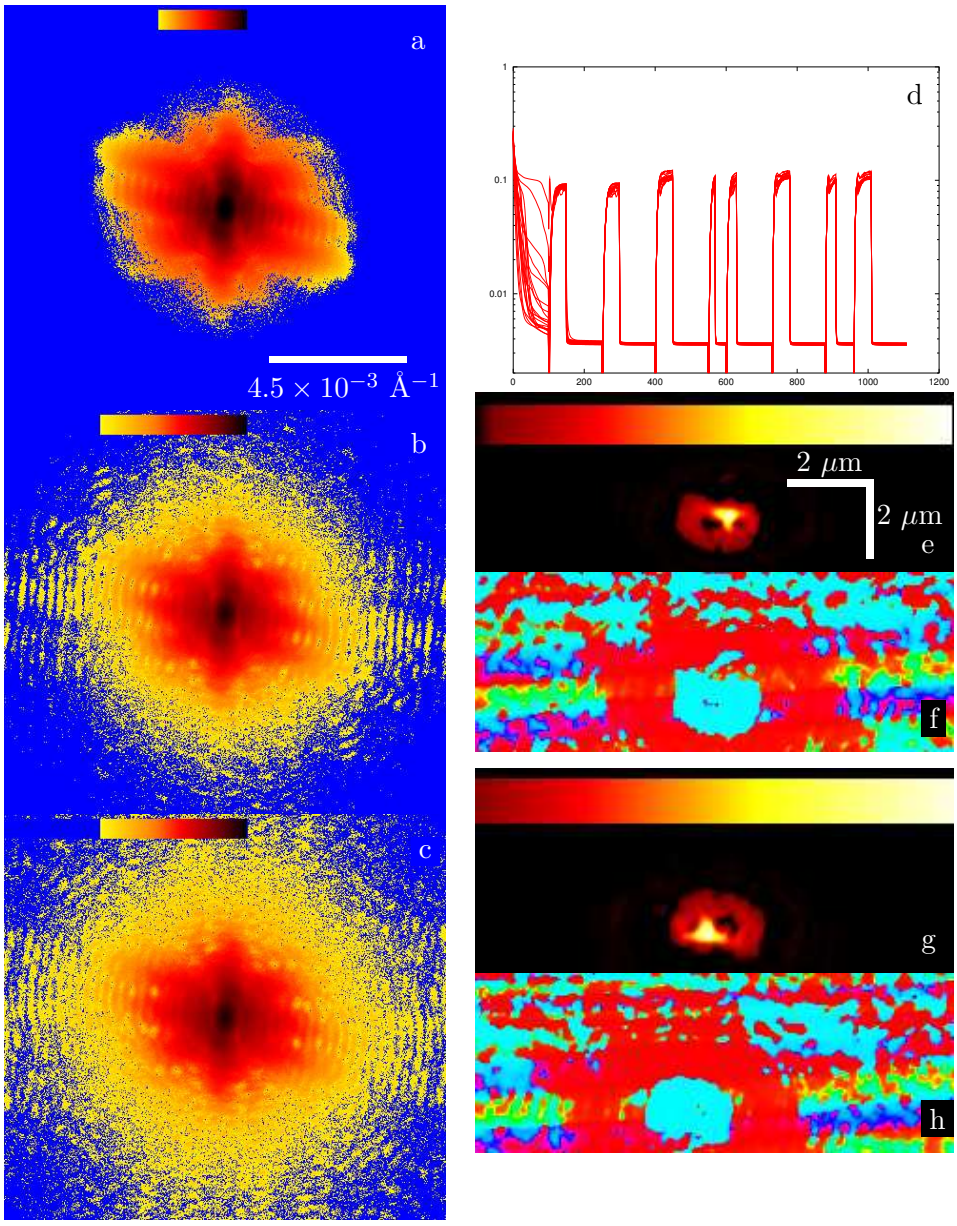


Figure 3.11: Results from fitting symmetrized 2D data whose preparation is detailed in Fig. 3.9. a is the corrected, symmetrized, and Gaussian filtered intensity from the CXD measurement, while b and c are the two best fits to it. d is a plot of the error metric during the fitting, starting with ER and then alternating cycles of HIO and ER. e and g are the magnitudes of the complex valued real-space density derived from b and c. The accompanying phase is shown in f and h. Here, a real positive constraint was imposed on the density and the support was a 100×100 px square. For reference, the real-space images, e-h, are 256 px in width. As before, a-c are log-scaled and e-h are linearly scaled.

companying real-space density reconstructions in Figs.3.11e-f and g-h. The reproducibility of the real-space reconstructions is quantified by $\xi_1^2 = 0.02$ with error metrics $\chi^2 = 0.003518$ and $\chi^2 = 0.003521$.

Despite the much better agreement with the modulus constraint and good reproducibility, the real-space reconstructions still exhibit a "hole" and a bright spot. In the symmetrized case, the same basic shape is reconstructed in real-space with only small details differing. It should be noted that the real-space phases are very similar.

These reconstructions are encouraging, but lead to many unanswered questions. What is the origin of the "hole" in the reconstruction? It is possible for the projection of a complex density to produce such structure, but should we expect this? In order to obtain a complex density the phase of the incident beam must change as it traverses the crystal, *e.g.*, the particle's crystal planes could be displaced from the ideal Bravais lattice or the coherence of the beam might be lacking. There is, in fact, good reason to suspect that the coherence of the beam may be insufficient: the visibility of the pattern is much less than unity, in other words, the intensity does not drop to zero between the fringes. Indeed, the penetration depth in the kinematical approximation of 7.5 keV X-rays in Au is about 0.37 μm, which is of the order of our longitudinal coherence length, $\xi_{\|} = 0.57$ μm. No data was collected in the third direction, but since the density projections give the dimensions perpendicular to $\mathbf{k_f}$ to be about 2×1.5 μm, we expect the crystal extends at least 1 μm in the third direction.

Also, heavy processing was necessary to merge the center and the edges of the pattern. As can be seen from Fig. 3.9 this was not perfectly accomplished. Finally, the pattern was symmetrized to account for the detector being away from the Bragg point and to reduce the effect of aliasing. We address these problems through improvement in the experimental procedure and data handling.

3.9 3D Result

3.9.1 Sample

In this experiment, the sample was a 2×2 mm^2 section of the Au film on Si/SiO_2 sample earlier discussed. This small sample was placed into the chamber shown in Fig. 3.4 and heated in air at 1050° C for ten hours. The temperature of the oven was then decreased to 850° C and the sample was held at this temperature for ten more hours before cooling to room

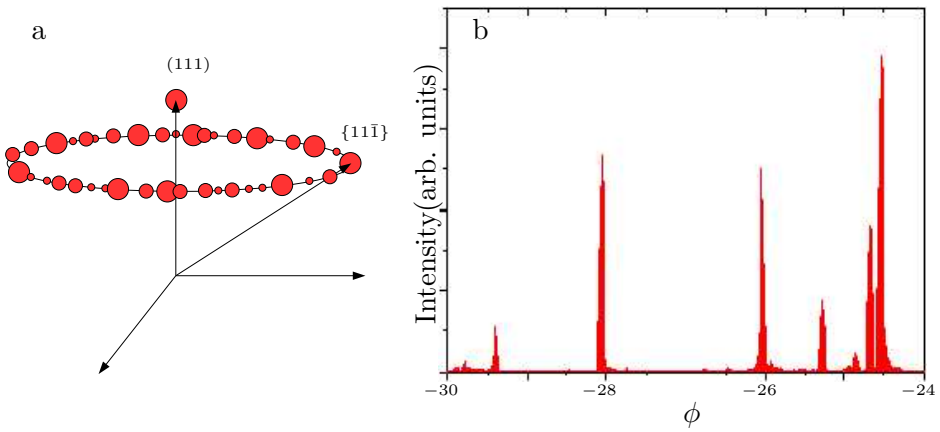


Figure 3.12: a) Illustration of diffracted intensity from a small number of illuminated Au crystals. b) Scan about the surface normal of the sample showing approximately one reflection per degree implying the number of illuminated crystals is of order 100.

temperature. After cooling, the sample was mounted to a ceramic post by means of a ceramic adhesive and the post was placed in the *in situ* heating cell shown in Fig. 3.6. The heater was then placed on the diffractometer and the sample illuminated by the X-ray beam.

3.9.2 Experiment

Roller blade slits[71] were used to limit the footprint of the beam on the sample so that only a small number of grains were illuminated. The diffraction near the specular (111) was found to be the incoherent addition of many overlapping patterns, so the diffraction from an off-specular (11$\bar{1}$) was measured. If the film were completely melted–and the substrate were assumed perfectly amorphous–we would expect a random orientation of the crystals upon freezing with uniform texture over 4π of solid angle. However, in this case the sample was not taken above the melting point of Au, so the film dewetted and maintained its (111) texture. This was confirmed on earlier samples using the facilities of the Center for Microanalysis of Materials. An illustration of this is displayed in Fig. 3.12. Fig. 3.12b is the diffracted intensity as ϕ, the rotation about the sample normal, was scanned. This scan indicates that, on average, there is signal from one particle per degree diffracting into the the detector slit. Since Au is an FCC material, we expect three such reflections from each illuminated crystal in this geometry; therefore, we estimate that between 100 and 150 crystals were in the footprint of the beam on the sample.

The light bulb *cum* heat source was turned on and the sample temper-

ature was raised to approximately 950° C. Although a thermocouple was buried in the ceramic adhesive beneath the sample, the temperature was also estimated by tracking the Bragg point as the sample was heated. Since the Au lattice expands with increasing temperature, the $(11\bar{1})$ Bragg point changes position. Using this change of position in conjunction with tabulated values of the lattice expansion for bulk Au, the temperature was estimated. The agreement with the thermocouple was quite poor, which is not surprising as it was embedded in ceramic adhesive. Once a suitable reflection was found, the incident angle, θ, of the sample was rocked in steps of 0.002° to acquire a 3D CXD pattern. At each θ position 15 accumulations of a 1.5 s exposure were captured, for a total exposure of 22.5 s per slice of the 3D pattern. In all, 31 2D data sets were collected in this way. At the completion of the scan, the X-ray shutter was closed and a background exposure captured.

3.9.3 Data Handling

In the case of this data set, better results were achieved without performing a background subtraction. The data handling began with extracting the data from the .SPE file produced by the camera and storing it as a sequence of 2D data files. To determine the center of the 3D data array, the center point of the most symmetric 2D slice was found and a square array was carved out of each 2D slice with that center. This was done to balance the calculation time–primarily the time spend performing an FFT–with the possibility of using an array so small that aliasing effects may manifest. A square array was chosen so that pixels in the x and y directions have an equivalent real-space size. The array cropped in this way was too small, and was therefore padded to create a larger array.

Since background subtraction was not performed, the padding was carried out through the use of **`2dcroppad2`**, which placed the old array inside a new array and filled the boundaries with a primitive calculated "background." This is accomplished automatically: the mean level is calculated from regions of the array from the center and some uniform random deviation is added, or subtracted, from the mean. In this case, the deviation was ±5% of the mean. This procdure led to the creation of a 440×440 array for each slice of data. Finally, the 2D arrays were stacked into a three dimensional array–using **`2dto3d`**–and reordered to prepare for fitting. Since the center-most slice of the scan missed the Bragg point, an average slice was calculated from the data collected farthest from the Bragg point, where

there was very little diffracted intensity, by looking at each pixel in three or more such slices and rejecting contribution to that pixel from any slice which had more ADUs than the lowest value plus a fraction of the ADUs generated by a photon in that pixel. The resulting empty slice was used to pad the array up to its final size of $440 \times 440 \times 30$.

3.9.4 Fitting

The support chosen for this fit was a rectangular region of width 90 px, height 120 px and depth 20 slices and it is this region of the array that is displayed in the figures. This is well in excess of the expected largest dimension of the crystal, 100 px, calculated from the fringe spacing of the 2D data. The recipe for the fitting was 19 cycles of alternating ER and HIO, beginning and ending with ER. The β parameter for HIO was 0.4 for the first three cycles, then increased to 0.8 for the next three, to 0.9 for the next two, and finally one cycle with $\beta = 0.7$. A plot of the error metric against iteration is shown in Fig. 3.13. The red line is the real-space metric, [3.24], while the green line is the reciprocal-space metric, [3.23]. Ordinarily in simulations the two have the same value during iterations of ER. This is, as was discussed in Chapter 2, not the case for iterations of HIO or DM. Here, after the first cycle of HIO the metrics in real- and reciprocal-space do not have the same value during cycles of ER, this is not unusual while fitting experimental data.

The first cycle is 150 iterations of ER and the error metrics both decrease during this cycle. The second cycle is 100 iterations of HIO, where we see that χ^2 greatly increases, although the trend with subsequent iterations is for it to decrease. As expected, the metrics did not agree during HIO, indicating that HIO primarily decreases the real-space metric, not the reciprocal-space one. The third cycle was another 150 iterations of ER. Immediately, two of the fits fell to a lower χ^2 than the others and a third fit began a slower descent to this level before the next cycle of HIO. Here, the two metrics began to disagree with one another. This is believed to be caused by the cycles of HIO, since it was not seen in ER only fitting recipes. The fitting continued in this way, with alternating cycles of ER and HIO. Over the course of these cycles, we see that the error metrics of the fits are distributed about one of two levels. In this particular case, 9 of the 10 fits find the lower level, which is slightly better than a typical fitting run. The error metrics calculated while fitting this data were always higher during iterations of HIO. Further, χ^2 displays a highly oscillatory behavior with solutions that have not found

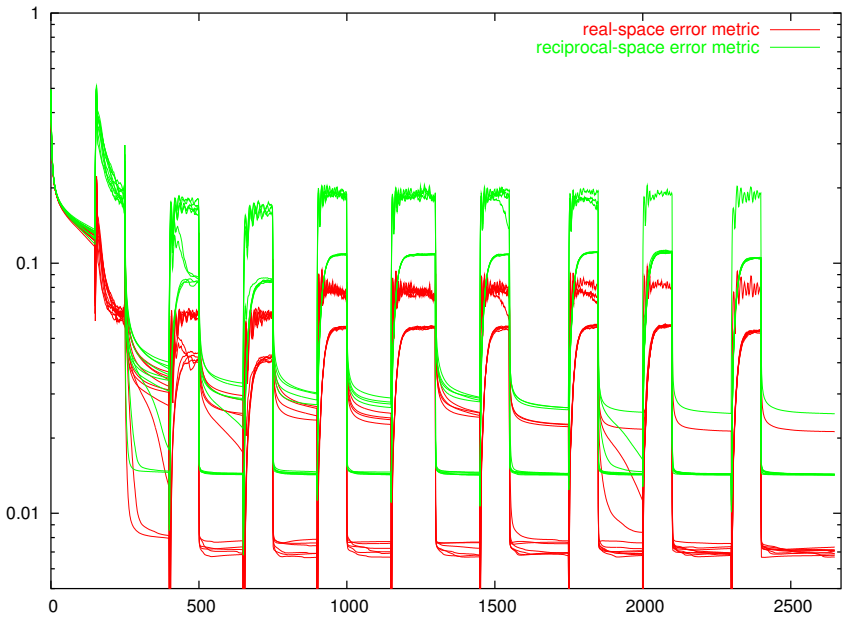


Figure 3.13: Error metric versus iteration for the 3D fit to CXD from Au. The fitting starts with one cycle of ER and then alternates between cycles of HIO and ER. It is observed that the error metric varies more during HIO for those fits that do not fall to the lower of the two preferred levels of the error metric.

the lower level during a cycle of ER. After an iterate found the lower level, subsequent iterations of HIO did not result in a rise of the error metric back to the higher level. It should also be noted that iterates at the lower level do continue to experience slight changes in the error metric as they were subjected to further iterations.

The final values of the reciprocal-space error metric, χ^2, and the real-space metric, χ^2_{Re}, for the best fit are $\chi^2 = 0.01422$ and $\chi^2_{Re} = 0.0068$ while for the second best fit $\chi^2 = 0.01427$ and $\chi^2_{Re} = 0.0067$. The irreproducibility between the two best fits is $\xi^2_1 = 0.0099$. For all of these values, a lower values implies a better agreement. We may interpret $\sqrt{\chi^2}$ as an RMS error between the fit and the measured data, which gives 11.9% average disagreement per pixel in both cases. Slices through the 3D complex reconstructed amplitude are shown in Fig 3.14. Shown is the amplitude on a linear scale, further scaled as indicated by the bar atop each image, as discussed in Section 3.7.3. All images are scaled in the same way.

For this experiment, a CCD detector with $\Lambda_{px} = 22.5\ \mu\text{m}^2$ pixels was positioned $L = 2.93$ m from the sample. Using $\lambda = 1.3$ Å for the wavelength

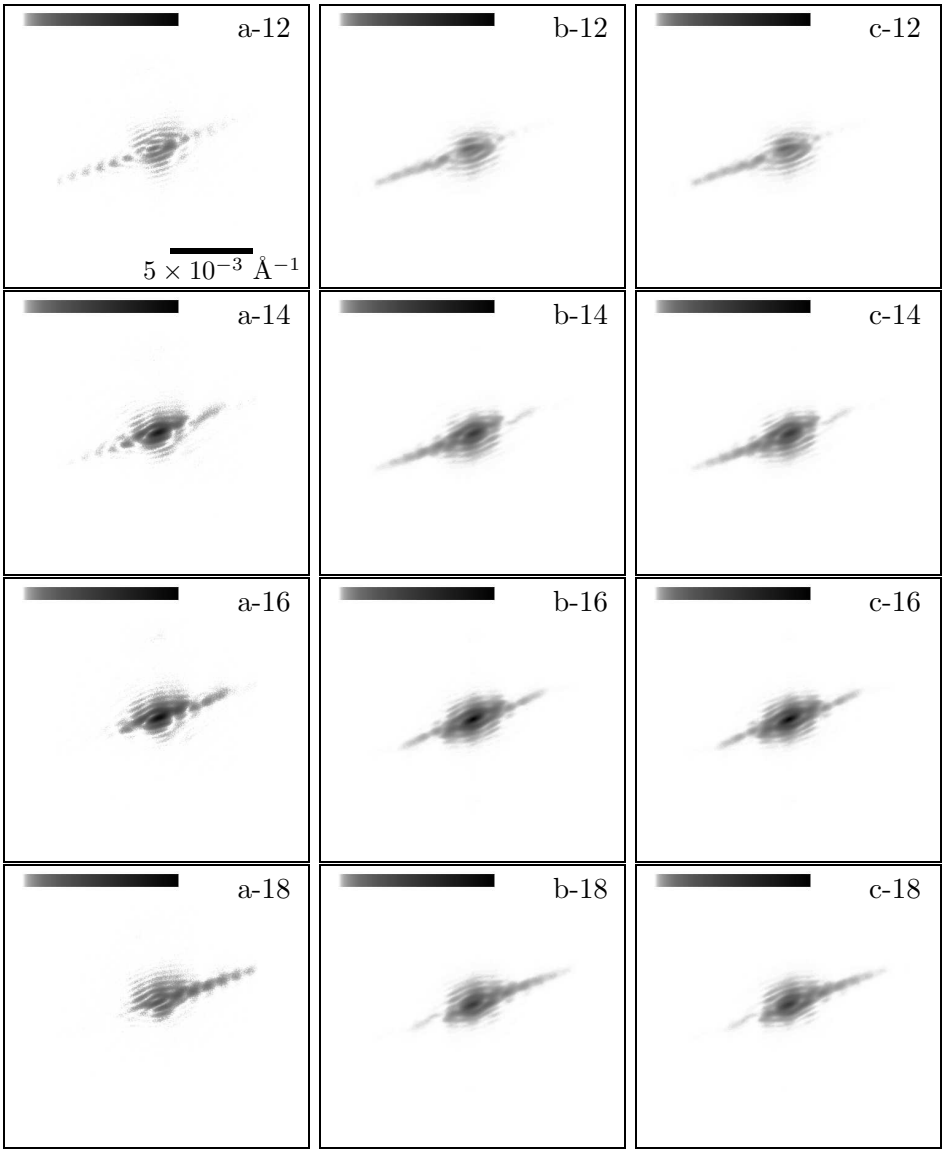


Figure 3.14: 2D slices through 3D complex amplitude. a- through c-12 is the 12th of 30 slices, where a is the original data, b is the best fit, and c is the second best fit. Slices are separated by 1.86×10^{-4} Å^{-1}. The magnitude of the amplitude is plotted using a linear scale given by the bar on each image.

of 9.5 keV X-rays, a pixel captured

$$\Delta q_{px} = \frac{2\pi}{\lambda}\frac{\Lambda_{px}}{L} = 3.7 \times 10^{-5} \text{ Å}^{-1}. \tag{3.28}$$

Rocking the sample in steps of $\Delta\theta = 0.002^\circ$ gave a spacing in the third direction of

$$\Delta q_\perp \approx |q|\Delta\theta \approx 9.3 \times 10^{-5} \text{ Å}^{-1}. \tag{3.29}$$

Thus, $\Delta q_\perp$ was 5.84 times larger than Δq_{px}; however, as can be seen from Fig. 3.14, the data were very oversampled in the plane of the detector, so the step size in the third direction is not too large to properly sample the diffracting intensity. Notably, Fig. 3.14b-16 and c-16 are centrosymmetric, while the data, 3.14a-16, are not. This no doubt contributed to the error of nearly 12% per pixel on average. It is possible to release the phase constraint in real-space, thereby allowing the reciprocal-space pattern to become asymmetric in this center-most slice, but for a real-space support of this size, allowing the phase to vary did not result in a true solution to the problem. If the phase were allowed to vary in this case, the best real-space fits would not agree, an indication that the algorithms did not find a solution. Examining 3.14a-14 and a-18, we see that, as expected, the pattern is symmetric about the Bragg point in pairs of planes, but not necessarily in any single plane away from the center.

Another interesting feature of the reconstructed reciprocal-space amplitude is that the pattern has lower contrast in the fringes than the original CXD data. In fact, we can estimate the visibility, $\mathcal{V}$, by calculating the ratio of the difference of intensity between a ridge and a valley and the sum of the intensities. An estimate can be found by averaging over 9 pixels near the first maximum and the nearby minimum. Using this method, we estimate the visibility to be $\mathcal{V} = 0.50$ for the CXD pattern and $\mathcal{V} = 0.23$ for the reconstructed pattern. The lower contrast is undoubtedly a sign that the reciprocal-space amplitude has not been perfectly recovered and may be a result of the real positive real-space constraint.

3.9.5 Real-space result

A simple inverse FFT of the recovered reciprocal-space amplitude yields the desired 3D real-space density map. 2D slices of this density are displayed in Fig. 3.15, where they are plotted linearly and a color gradient is applied to enhance the features inside the crystal. The bar at the top of the figure summarizes the color scheme as described in Section 3.7.3. Using the normal

relationship between conjugate variables and [3.28], we find a single pixel in our $N_x = N_y = 440$ element array spans an amount

$$\Delta r_{px} = \frac{2\pi}{\Delta q_{px} N_x} = 385 \text{ Å} \tag{3.30}$$

of real-space. Analogously, the space between each of the $N = 30$ slices, using [3.29], in the array is

$$\Delta r_{\perp} = \frac{2\pi}{\Delta q_{\perp} N_z} = 2250 \text{ Å}. \tag{3.31}$$

The slices in Fig. 3.15 begin with the sixth slice of the 3D array, the first slice in which the support allows density to exist. Every third slice, up to the twenty-seventh is shown, the twenty-fifth being the last in which the support allows density. These images give the size of the Au crystal to be approximately 1.1 μm by 2.8 μm in the plane. Examining the slices, we find substantial density in 18 of the thirty, giving the dimension in the third direction to be about 4 μm. The maximum density value reconstructed is 3305 and cut off for the extent of crystal was chosen to 150, by examining the drop-off in density with increasing distance from the center. The absolute values of these numbers have no meaning, being determined by the total integrated intensity of the measured data. It is important to remember that each slice is perpendicular to the vector $\mathbf{k_f}$, so we are seeing a "k_f-view" of the crystal in which it appears to be elongated. Nevertheless, the crystal is definitely of an oval shape, rather than the circular one expected in equilibrium[40].

Indeed, significant structure has been reconstructed inside the crystal, whose interior is expected to have uniform density–assuming no strain is present. Before attempting to analyze the specific features within the crystal is it important to establish the resolution of the measurement. In this case, the smallest period that can be measured in real-space is governed by how far from the Bragg point the data extend. Examining column a of Fig. 3.14, we find that the data extend approximately 170 px from Bragg point, giving an in-plane resolution of

$$\Lambda_{x,y} = \frac{2\pi}{\Delta q_{px}(170)} \sim 1000 \text{ Å} \tag{3.32}$$

and in the third direction

$$\Lambda_{\perp} = \frac{2\pi}{\Delta q_{\perp}(11)} \sim 6000 \text{ Å}, \tag{3.33}$$

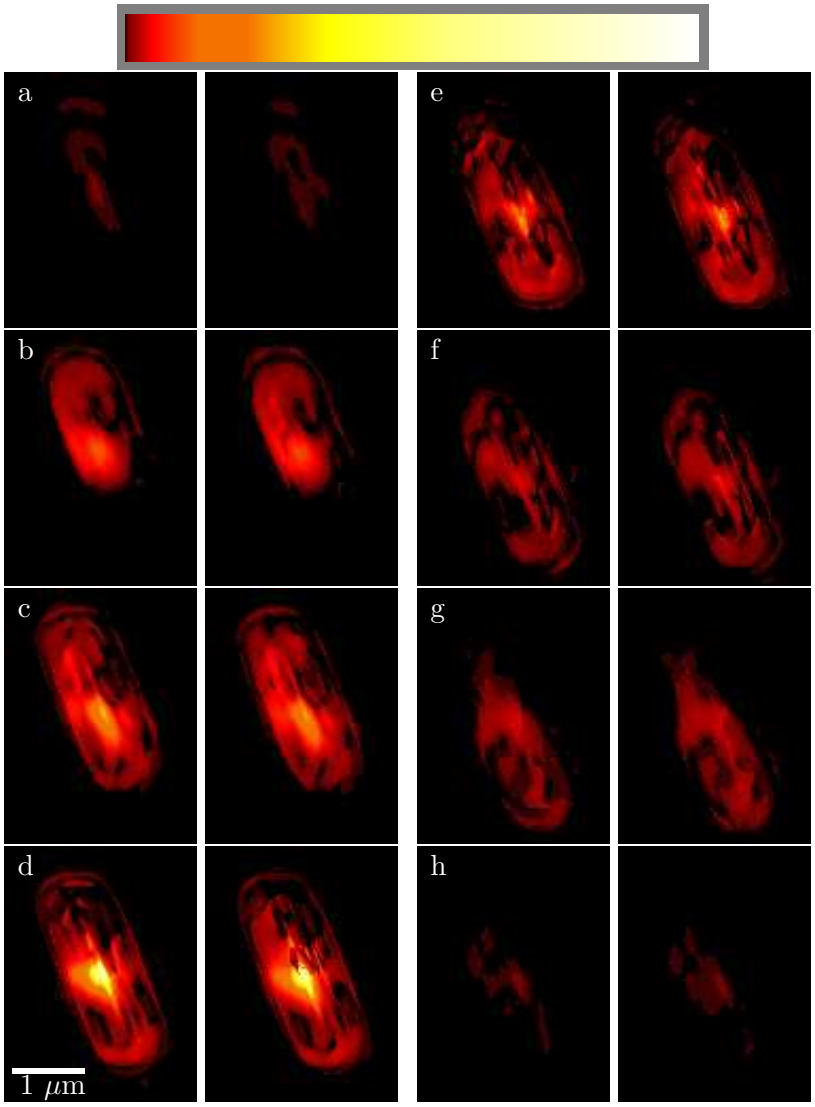


Figure 3.15: 2D slices in a "k_f-view" through the reconstructed 3D real-space density, linearly scaled and colorized as demonstrated in the bar at the top of the image. In each lettered pair, the left slice is from the best fit and the right from the second best. Unexpected density contrast is revealed, most notably a bright spot in the center-most slices and fine modulations diagonally from top left to bottom right in the (111) direction. Slices are separated by 0.675 μm and in this view, the crystal is about 4 μm thick in the direction normal to the slices.

examining all 30 slices.

It is important to consider the effect of the partially coherent X-ray beam might be having on these reconstructions. Various studies of the effect of beamline optics on the partial coherence have been performed[48]. From Section 3.1.3, we can derive an understanding of the ≈ 0.5 μm bright spot in the center of the real-space reconstructions. If we consider an idealized beamline consisting of only the undulator and a Be windows between the source and sample, we can estimate the form of the MIF[93]. In the limit of small crystals, the MIF–composed of the source, J_S, and window, J_W, components–can be approximated by neglecting intensity variations across the incoming beam, $J_S(\mathbf{r_1}, \mathbf{r_2}) = I_S(0)\gamma_S(\Delta\mathbf{r})$ and $J_W(\mathbf{r_1}, \mathbf{r_2}) = I_W(0)\gamma_W(\Delta\mathbf{r})$. γ_S and γ_W are the complex degrees of coherence of the source and the Be windows respectively, in analogy to [1.22] of Chapter 1.

The complex degree of coherence of the source can be calculated in the far field from the van Cittert-Zernike Theorem, [1.19], to be

$$\gamma_S(\Delta\mathbf{r}) = e^{\frac{-\Delta r_x^2}{2\xi_x^2} + \frac{-\Delta r_y^2}{2\xi_y^2}}, \tag{3.34}$$

where $\Delta\mathbf{r}$ lies in the 2D transverse plane and $\xi_{x,y}$ are the coherence lengths derived from the Gaussian source size we expect from a synchrotron source. The Be window contribution must be calculated in near field since far field limit gives the bound on the distance from the window to the sample, L_2: $L_2 \gg 2\sigma_{eff}\xi_{in}/\bar{\lambda} \approx 100$ m–using the ring parameters in 3.2 and the calculated coherence lengths from earlier in the chapter–while in actuality the optical element is often the exit window, with $L_2 \approx 5$ m. The contribution from such a beamline component has been calculated[93] and the expected result is that $\gamma_W(\Delta\mathbf{r})$ is a sharply peaked function with width 0.8 μm. Therefore, it is likely that the bright spot in the center of the reconstruction is the result of the partial coherence of the incident beam, which is itself the addition of a coherent intensity from the source and a much shorter coherence length intensity scattered by the beamline optics.

It does not seem likely that voids would be present within a Au crystal prepared in this way; however, the apparent density modulation inside the crystal could result from other physical phenomenon. Notably, if a small section of the crystal were misaligned with the bulk, that section would diffract into a totally different direction in reciprocal-space and its diffracted intensity would therefore not be measured. One property of the reconstruction might be explained by twinning[38]. In a twinned region within the crystal, the stacking order is reversed and therefore diffracts into a region of recip-

rocal space 60° away from the reflection measured in this experiment. It does seem to be the case that these stripes have reconstructed perpendicular to the (111) direction, since the vertical direction in the reconstruction is the $(11\bar{1})$[71]. Unfortunately, these features are too close to the limit on our resolution, 50 nm, to attribute them conclusively to twinning within the crystal, although it does remain the best explanation.

The remaining "holes" within the particle may be due to relatively large regions of crystallographically misoriented crystal, or an artifact of the tight constraints enforced during ER. These holes are not seen in reconstructions from other samples. For example, the results from a similar experiment on Pb are presented in Chapter 5 do not have this feature, so the holes are not believed to be an artifact of the method. If the crystal were highly strained, it is possible that repeated iterations of forcing the reconstructed density to be real positive might also contribute to this contrast.

Fig. 3.16 shows the 15th slice in the 3D array for the two best fits. As can be seen from Fig. 3.16a and b, there is only a slight difference between them. Perhaps more interesting is the real-space phase shown in Fig. 3.16c and d. Here we see a very complicated structure has been recovered. Although we do enforce a real-positive constraint, the real-space image shown is recovered by a back transform from the reciprocal-space amplitude, so it can be, and in this case is, a complex valued function. Given this result, one might suspect that relaxing the real-positive constraint would give a better result; however, in this case loosening the real positive constraint does not reduce the apparent density contrast and the phase is left essentially unchanged.

3.10 Summary

This section opened by describing the equipment used in our coherent diffractive imaging experiments. A brilliant synchrotron source is a very natural choice for this type of experiment, giving both large transverse coherence lengths and enough flux to capture data in a reasonable amount of time. The method of operation of the insertion device gives rise to Gaussian complex degree of coherence incident upon the sample, which is sufficiently broad to image samples with dimensions of microns cubed. Our longitudinal coherence was given by the inherent Darwin width of the monochromator and is normally on the order of a micron. Direct read CCD detectors have physical limitations–shallow well depth so that short exposures are required, only 30% efficient at 10 keV, difficult to count photons in high-flux situations, etc.–but they are still the best available device for this experiment.

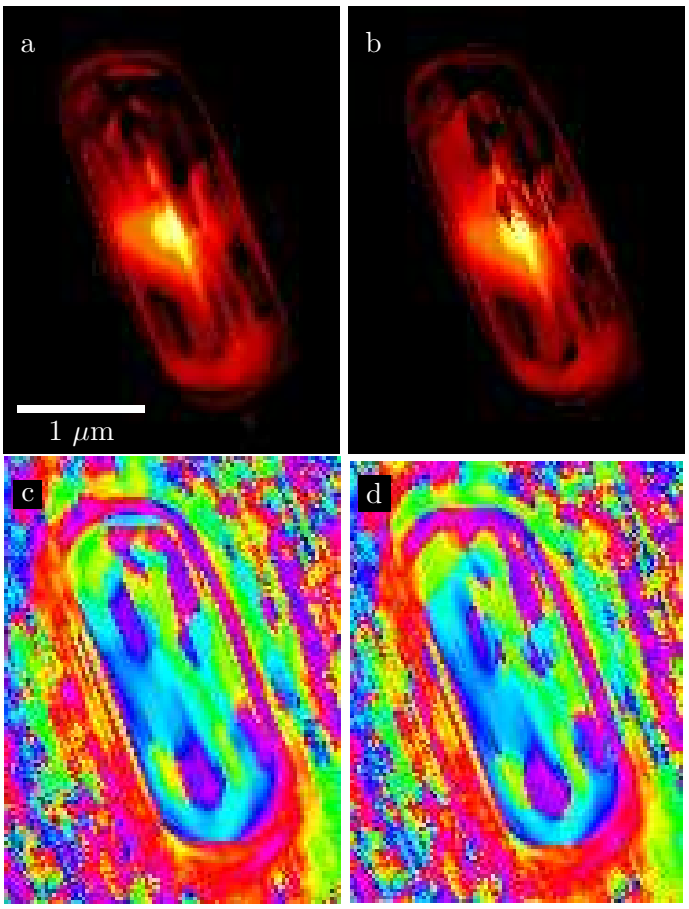


Figure 3.16: The 15th slice from the two best 3D real-space reconstructions from Au CXD patterns. The real-space density is a complex valued function with the amplitude of the best fit shown in a, and its phase in c. b and d are the amplitude and phase of the second best fit. Real-space scaling is the same as in Fig. 3.15 and phase is plotted as described in Section 3.7.3. The two show only minor differences.

The experimental geometry has also been described and two methods of acquiring 3D CXD patterns–rocking the incident angle and changing the incident energy–were discussed. The method described here requires only slight changes to the incident angle, 0.002° steps in this case, alleviating the precise alignment of the sample with the rotation axes required by other methods. Data handling was also discussed both in the theoretical and the practical way that was performed here. Programs were written to automate many of these procedures and their operation discussed.

The resultant real-space density from a phased 3D CXD pattern was presented. Unexpected density modulations were observed at high temperature in small crystals formed by dewetting a Au film from an Si/SiO_2 surface. A very high apparent density near the center of the crystal is believed to be the result of the partially coherent illumination of the crystal. This partial coherence effect arises from the rescattering of the beam by optical elements along its path and poses a complication to experiments of this kind. A second modulation occurs perpendicular to the (111) direction and may be due to twinned regions; although, the resolution here is too poor to conclusively state this.

Finally, it must be acknowledged that some or all of these modulations may be due to other factors: aliasing–due to the size of the array–dynamical scattering, or measurement noise may all be important. Artifacts from aliasing can be simulated by changing the size of the array, as this will materially change the aliasing. The contribution from dynamical scattering is a more complicated problem whose solution calls for a detailed calculation. The possibility that these features may be attributed to noise is treated by means of a simulation in Chapter 4.

Chapter 4

Simulations

4.1 Motivation

There are fundamental differences between the real-space density maps recovered from experimental data and those obtained from simulated objects. The latter may be recovered from the magnitude of their Fourier transform with relative ease and yet experimental data will tend to stagnate, possess artifacts, and arrive at multiple solutions. We seek to answer the question: what must done to simulated CXD data to create the problems inherent in fitting experimental data?

We begin by examining the convergence behavior of the algorithms on five different shapes, including the number of iterations required, the fidelity to the original object, and the reproducibility between the two best fits of several. The important question to answer is what calculable quantity is related to the fidelity of the reconstruction to the original object? As we will show, the error metric alone is not a sufficient indicator of good fidelity to the truth, or simulated, object. Fortunately, we may also measure the reproducibility between the best two fits, and these two quantities together may be used to eliminate false solutions.

As a precursor to exploration of the causes of difference between simulations and data, we will give examples of how well the algorithms perform with various kinds of simulated objects ranging from handwriting, to gray-scale objects, and finally to polygons. This section gives a feeling for the strengths of the algorithms, but should not be overemphasized as special care has not been taken to optimize the fitting–*e.g.*, by altering algorithm parameters or tailoring real-space support constraints–which might dramatically alter the results.

Finally, we will move on to simulating a noisy reciprocal-space data set and examining the artifacts introduced by different sources of noise,

experimental factors, and fitting methods, including changing the number of "photons" in the simulation, as well as changing algorithm parameters. The primary difference between these simulations and those of earlier sections of this chapter is that all simulated patterns will be generated with a particular number of photons. This is in contrast to the previous simulations, which are discrete Fourier transforms fit in the same in which they were generated. As such, the early simulations possessed considerable aliasing, as a sharp edged object contains very high spatial frequency terms.

4.1.1 Error metrics and fidelity

To monitor the progress of an iterate, we use an reciprocal-space error metric defined as:

$$\chi^2 = \frac{\sum_{i=1}^{N} (|A_i^{calc}| - \sqrt{I_i^{meas}})^2}{\sum_{i=1}^{N} I_i}, \tag{4.1}$$

where A_i^{calc} is the amplitude of the ith pixel in the current iterate in reciprocal-space, I_i is the measured intensity, and the sum is taken over the N pixel detector. This is [2.5], the metric used in the discussion of ER in Chapter 2. In the case of HIO, the reciprocal-space is not properly thought of as the best estimate of the true diffraction pattern, but rather as a driving function, so the error metric is altered to become a real-space metric:

$$\chi^2_{\mathrm{Re}} = \frac{\sum_{x=1}^{N} |f_n(x) - f'_{n-1}(x)|^2}{\sum_{i=1}^{N} I_i}, \tag{4.2}$$

where $f_n(x)$ is the current best estimate of the object, and $f'_{n-1}(x)$ is the result of applying the reciprocal-space constraint to $f_{n-1}(x)$. Finally, the progress of DM is monitored by tracking the step size of the algorithm, i.e., calculating the difference between iterates:

$$\zeta_{\mathrm{DI}} = \frac{\sum_{x=1}^{N} |\rho_n(x) - \rho_{n-1}(x)|^2}{\sum_{i=1}^{N} I_i}, \tag{4.3}$$

which depends only on the difference between the current step and the previous step. We can also define a measure of agreement between two real-space

objects using the same form,

$$\xi_a^b = \frac{\sum_{x=1}^{N} |\rho^{(a)}(x) - \rho^{(b)}(x)|^2}{\sum_{i=1}^{N} I_i} \tag{4.4}$$

This last quantity will be helpful in identifying weaknesses in the algorithms. The metrics defined by [4.1]-[4.3] are used internally; however, often only values for χ^2 and ξ_a^b will be presented, since it is interesting to examine the difference in the accuracy of the reconstruction in the real- and reciprocal-spaces. Typically, the real-space measure will be ξ_1^2, quantifying the reproducibility between the two best fits, or ξ_1^{orig}, quantifying the accuracy of the best fit to the original object. For the the quantities [4.1]-[4.4] a value of zero indicates a perfect match between the two functions.

4.2 Comparison of Algorithms

Four algorithms are included in this section: Error Reduction(ER)–Section 2.2–Difference Map(DM)–Section 2.4.3–Fienup's HIO(HIO)–Section 2.3.1–and Millane's HIO(mHIO)–Section 2.3.2. The FFT of each object is taken and the phases are removed. This is used as the "input data" for the reconstruction. The standard operating procedure is to begin each fit with a randomly chosen uniform distribution of phases between $-\pi$ and π and includes 1000 iterations of one of the algorithms. A run consists of ten such fits each with new starting phases. Each figure in this section contains a plot of χ^2 and lists ξ_1^2, the irreproducibility between the two best fits for a particular algorithm. The real-space constraint is support for DM and support plus positivity for ER, HIO, and mHIO. β is chosen to be 1, and for DM we choose the maximally contractive values for γ_i so that the only difference between HIO and DM is the real-space constraint chosen. $\beta = 1$ is not necessarily the best choice for a particular object, but the goal is not to completely optimize all parameters, merely to examine the general behavior. The ϵ parameter for mHIO is set at 0.01. The region displayed in the figures is of dimension 256×256 px and the original object, its FFT, and the fitting were all carried out in a 512×512 array. Therefore, all examples shown here are well over the theoretical requirement for the oversampling discussed previously.

The first object is a handwriting sample meant to test the ability of an algorithm to reconstruct a shape with very high spatial frequency terms. Fig.

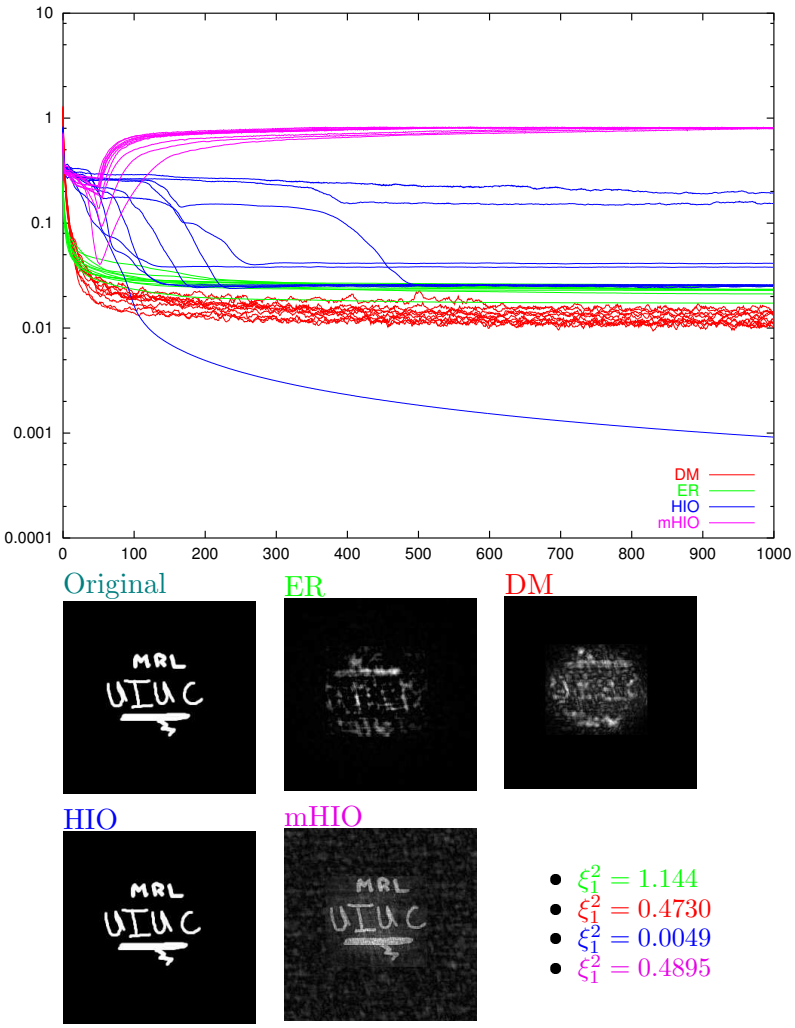


Figure 4.1: High spatial frequency object with rectangular real-space support. HIO clearly excels, while ER and DM fail to fit image.

4.1 is the result for the object with a rectangular constraint of size 135×120 px, while Fig. 4.2 is the result for the same object with an oval constraint– of size 162×139 px. For this object, HIO provides the best fit and the irreproducibility between the two best fits is quite low at 4×10^{-9}. ER has clearly stagnated and has not reached a suitable solution in the allowed 1000 iterations. Although the difference map fit achieves lower values of χ^2 and ξ_1^2, the fit is clearly worse than the result from mHIO. It is interesting to note that the only difference between HIO and DM is the positivity constraint, which clearly plays an important role in finding the correct solution in this case. Note that HIO finds the twin in the case of Fig. 4.2. This is an allowed

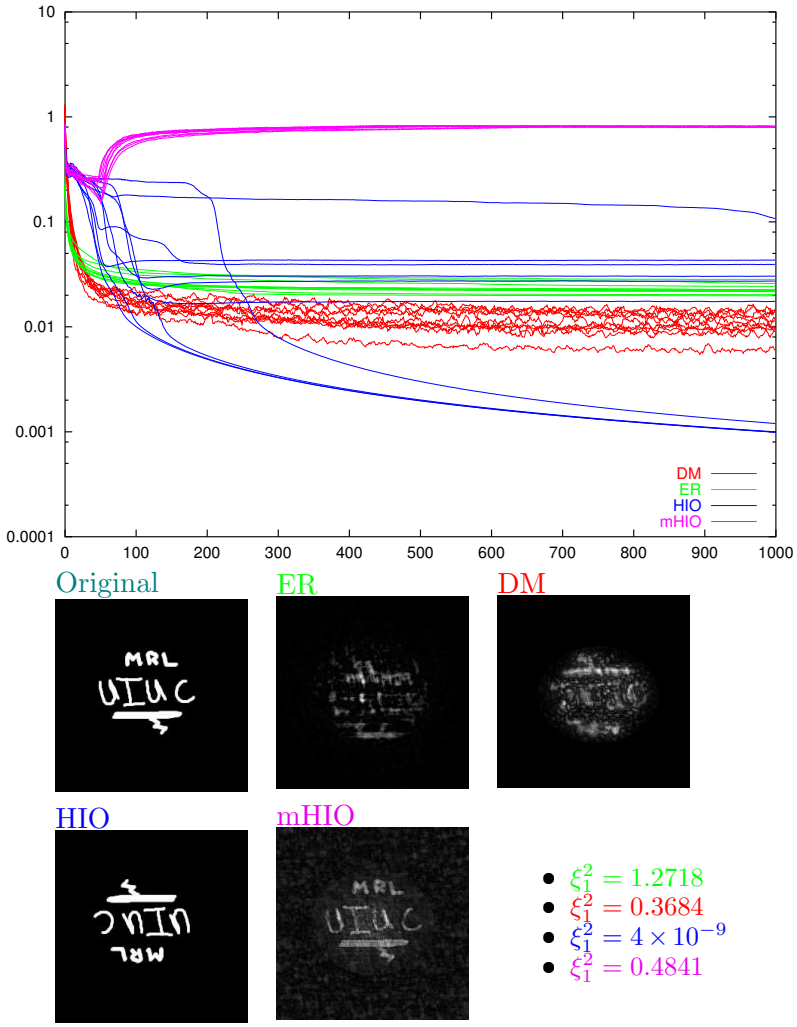


Figure 4.2: High spatial frequency object with oval real-space support. HIO clearly excels, while ER and DM fail to fit image. mHIO does slightly worst with this constraint, while HIO produces two fits nearly identical.

solution and the possibility of a twin is taken into account in calculating ξ_1^2.

Fig. 4.3 is one of the few situations in which a solution is found to within machine precision. These fits assumed a square support of 128×128 px. mHIO is clearly the best algorithm for fitting an object of this nature: all ten of the fits result in a solution. ER also finds the correct solution in one out of ten fits. Here, the difference between the first and second best fits is striking with $\xi_1^2 = 1.0749$, indicating that the two best fits are not correlated. HIO also comes very close to finding the correct solution, however, DM is still lagging due to the lack of the positivity constraint. In the case of Fig. 4.4, it is seen that the results are worse for all the algorithms, even mHIO does not converge each time. The only difference between this and the previous run is that the shape of the support has been changed to an oval of size 158×128 px. Thus we see that the shape of a support may affect the outcome of the fits.

It is also interesting to explore what happens when objects with more than two levels of intensity are used with the same support constraints. Fig. 4.5 uses the same arrangement of rectangles as the previous two cases, but this time it is positioned on a gray square. Even this seemingly slight change drastically reduces the effectiveness of the algorithms. HIO now provides the best fit to the object with DM a close second. mHIO still performs slightly better than ER. The most interesting feature in this figure is that all four algorithms seem to have reached an iterate that is composed of an iterate mixed with its twin, as can be seen from the presence of a faint outline of the largest rectangle positioned symmetrically about the center from the large rectangle of higher density. Fig. 4.6 uses the oval constraint that is not quite large enough to contain the gray square. The result is that all four algorithms fail, with mHIO results being the worst with $\chi^2 > 1$, indicating that the iterate is far from the correct solution.

The final object is a four sided irregular polygon whose sides are not oriented with the edges of the array. The rectangular constraint used in Fig. 4.7 had size 150×150 px. All four algorithms successfully find the outline of the polygon, but only the HIO fit reconstructs an object without internal fluctuations. DM is the second best, beating out mHIO which seems to have a slight mixing between an iterate and its twin, demonstrated by the sharp line along the left side of the reconstruction. ER suffers from large internal fluctuations. Using an oval constraint of dimension 170×174 px, Fig. 4.8, the quality of the fits suffer. While HIO and mHIO exhibit a slight increase in internal fluctuation, the boundary of the DM fit becomes blurred and ER shows signs of a strong mixing with an iterate and its twin.

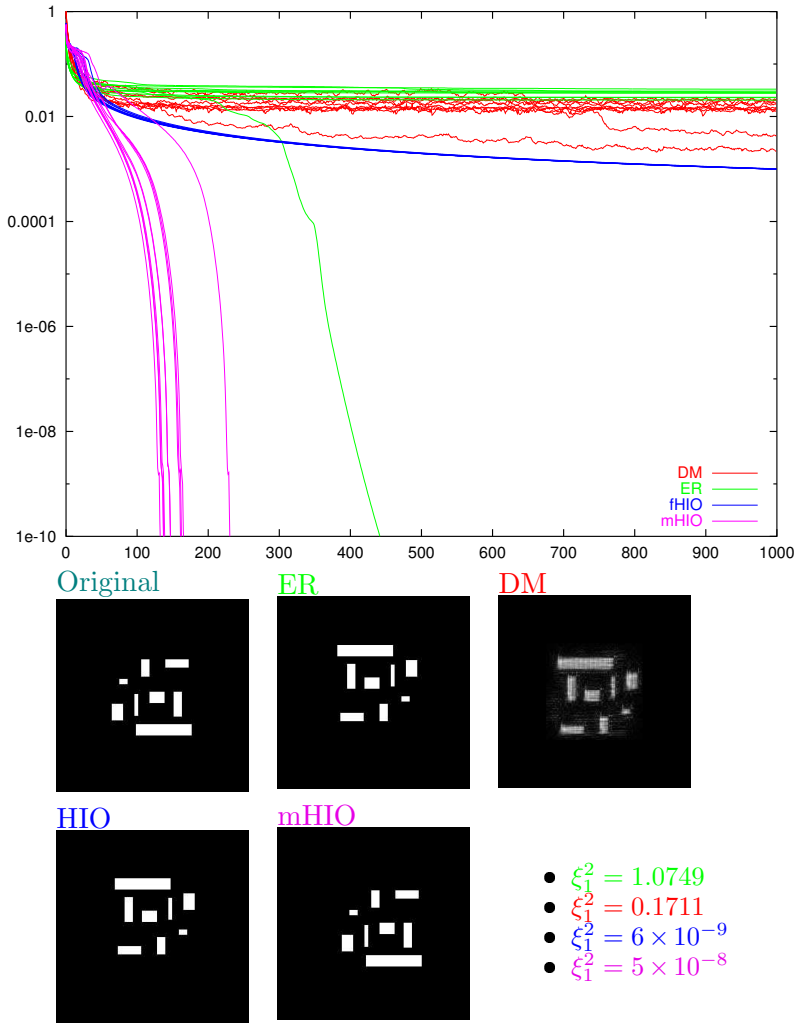


Figure 4.3: A collection of rectangles possessing many spatial frequencies fit with a rectangular real-space support. All sets of starting phases for mHIO find the correct solution, while only one set succeeds with ER. While the HIO fits are good, they do not approach the quality of mHIO or the single ER fit to find the solution. Note the very high value of ξ_1^2 for ER, indicating that the other ER fits are far from the solution.

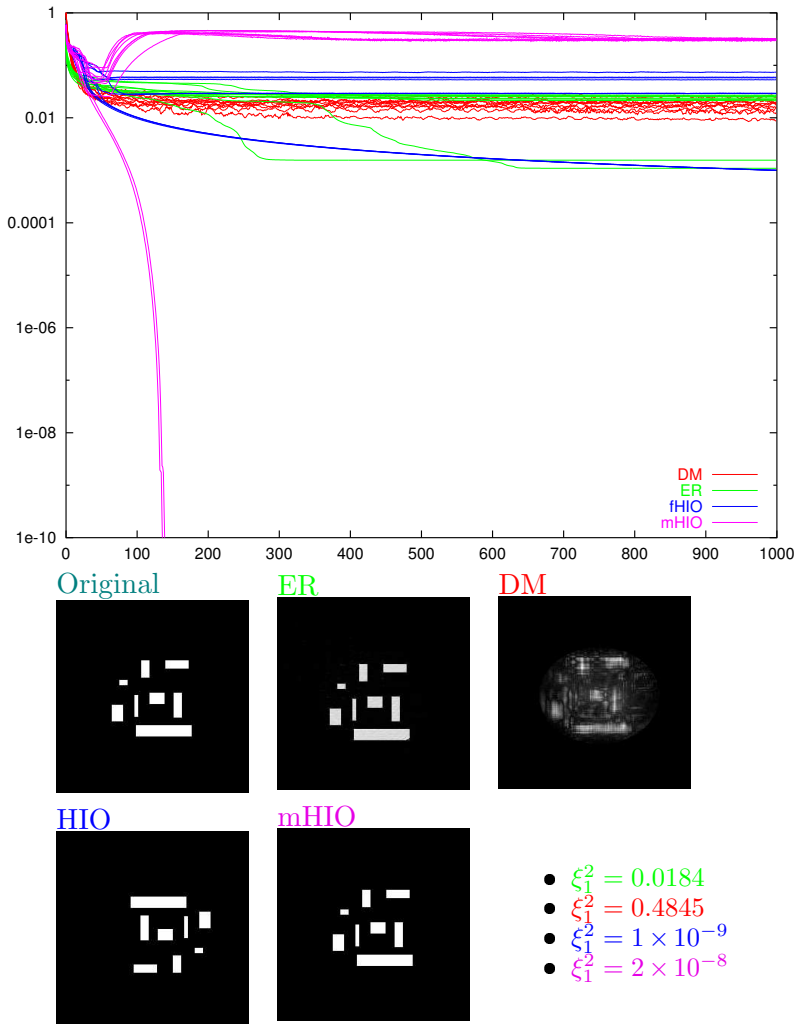


Figure 4.4: A collection of rectangles possessing many spatial frequencies fit with a oval real-space support. With this support, none of ER fits find the solution and only about half of the mHIO do. HIO perform similarly with the change of support, but DM performs poorly.

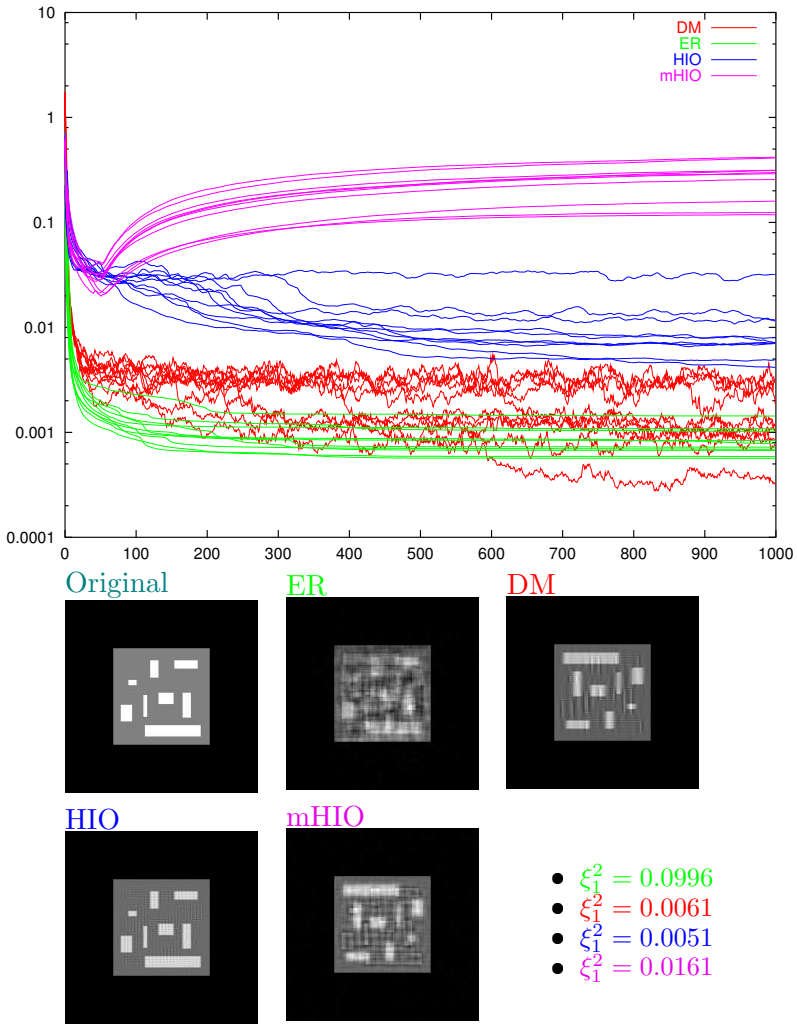


Figure 4.5: A collection of rectangles possessing many spatial frequencies within a larger square with amplitude less than the small rectangles fit with a rectangular real-space support. The addition of the large square at a different amplitude destroys the mHIO algorithms convergence behavior. HIO now gives the best fit, but algorithms have arrived at an iterate which is mixed with its twin, as can be seen from the overlapping shadows in the reconstructions.

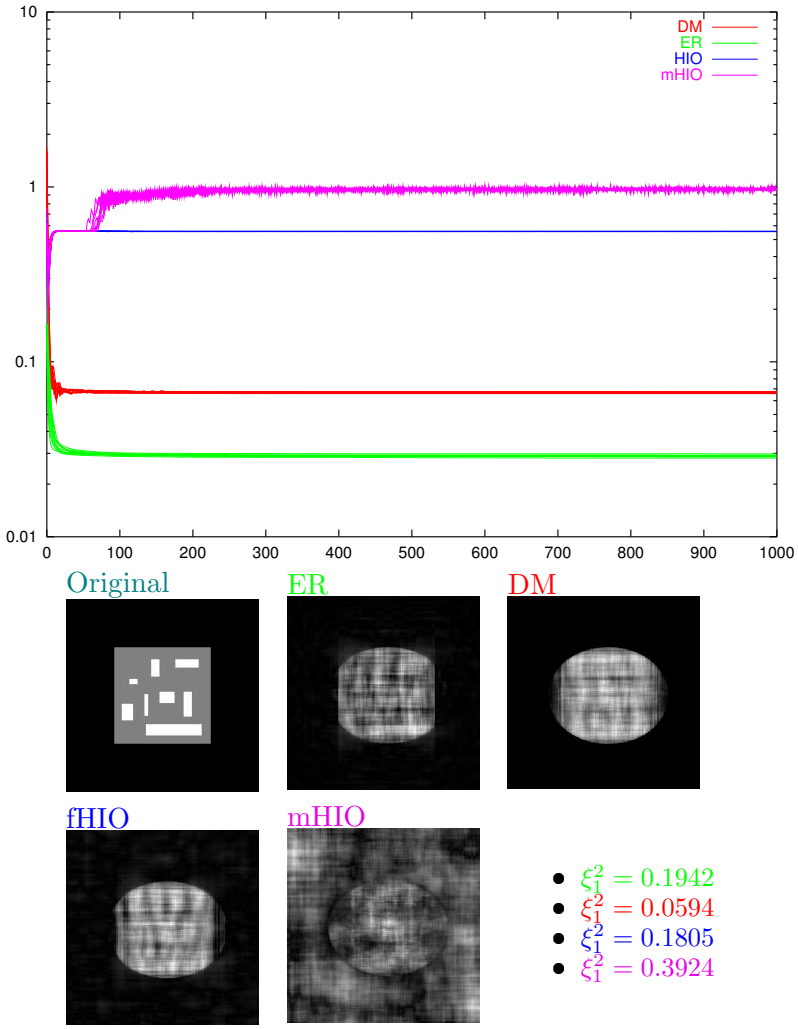


Figure 4.6: A collection of rectangles possessing many spatial frequencies within a larger square with amplitude less than the small rectangles fit with an oval real-space support. In this case, the constraint is too small for the reconstructed object. mHIO clearly misbehaves when the support is too small, as demonstrated by the large amplitude which reconstructs outside the support and the very poor χ^2.

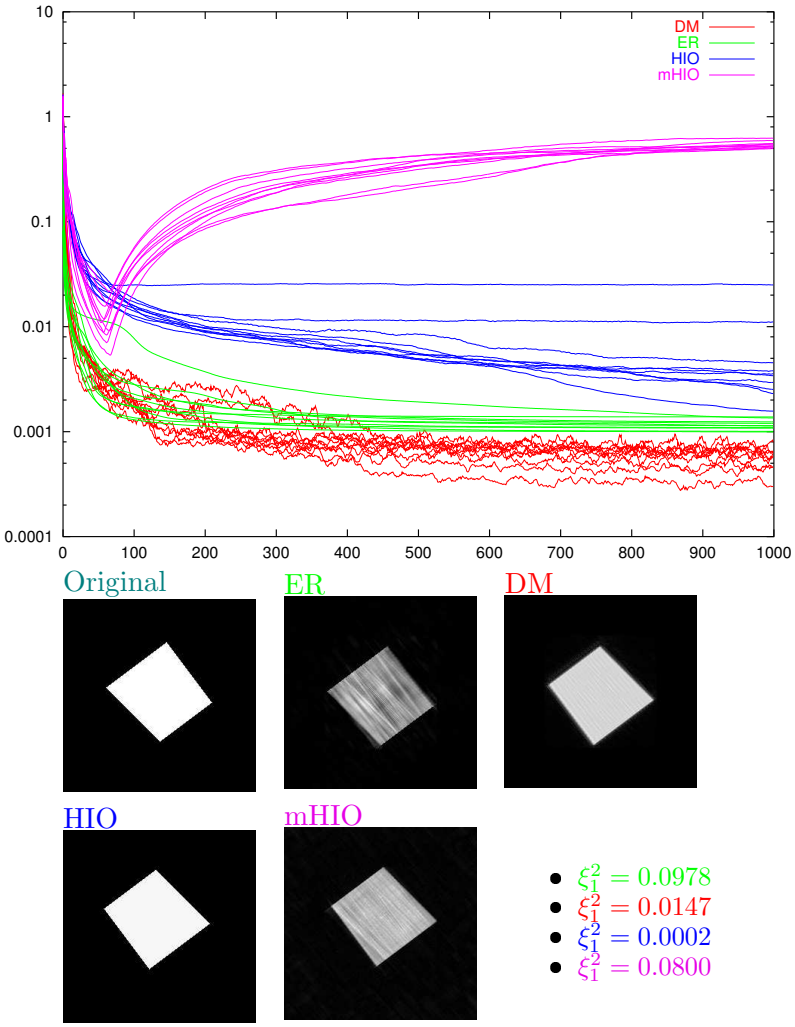


Figure 4.7: Four sided polygon with square constraint. The sides of the polygon are not parallel to the sides of the array to mitigate aliasing. In this case, all algorithms reconstruct the exterior shape, with the primary deficiency lying in the reconstructed modulation within the object. HIO has nearly succeeded here, followed by the DM, then mHIO, and finally ER.

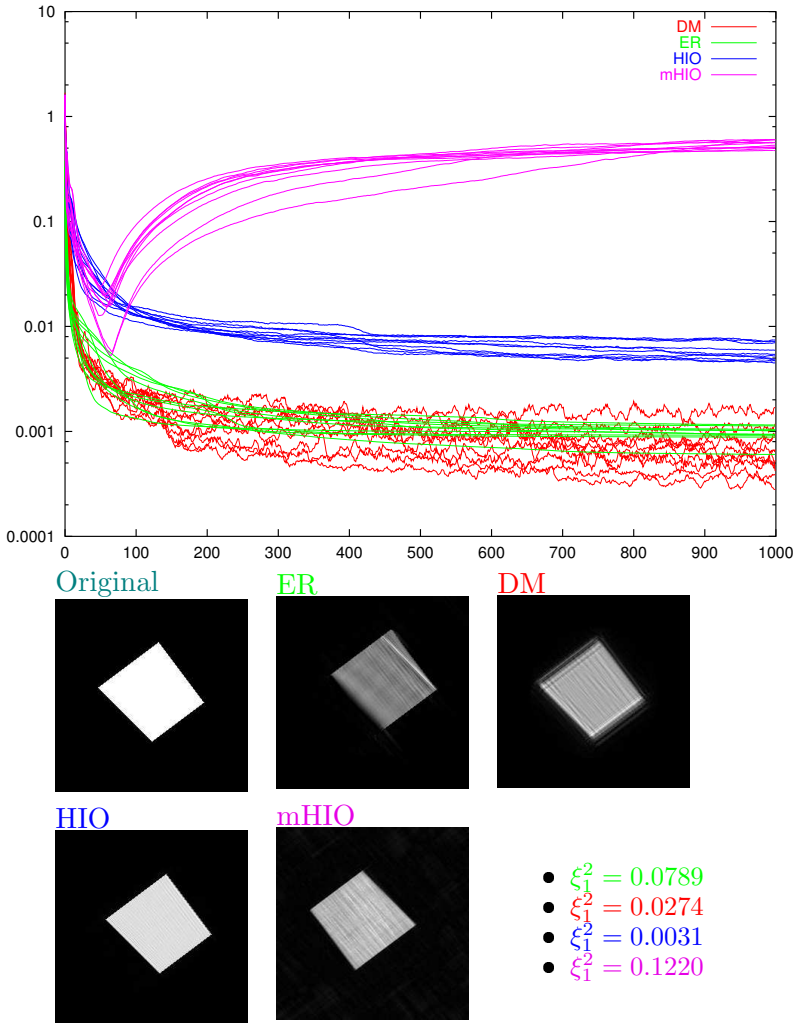


Figure 4.8: Four sided polygon with an oval support. The sides of the polygon are not parallel to the sides of the array to mitigate aliasing. Again, the oval constraint, which is of slightly larger total area, gives worse reconstructions. In this case, ER is clearly mixed with its twin, as exhibited by the bright line which would be the exterior edge of the twin of the dominant reconstruction.

Finally, we examine the case of a polygon which has been blurred with a Gaussian, Fig. 4.9. This case most closely resembles the experimental case discussed in Chapter 3, since a 2D CXD pattern corresponds not to a 2D slice through the diffracting crystal, but rather to the projection of the crystal's density onto a plane, leading to smooth edges. For this case a square real-space support of dimension 150×150 px was used. Now, DM, HIO, and mHIO all seem to perform well, while ER still reconstructs internal fluctuations after 1000 iterations.

These examples demonstrate that no one algorithm is ideally suited for all situations; however, we can conclude that DM has made a respectable showing considering that there is a huge parameter space associated with its three variable parameters. ER has demonstrated its well known propensity to stagnate at an intermediate iterate. mHIO does have one additional parameter, ϵ, that was not varied during these tests, so changing that parameter could lead to better convergence in those cases where it performed poorly. The difference between HIO and DM strikingly highlights the import of the positivity constraint, when it is physically applicable. It must also be noted that DM variants, and mHIO, tend to converge more quickly than ER.

4.3 Simulation of CXD Patterns

4.3.1 Creating a "crystal"

The first step in simulating a diffraction pattern is to choose the basic shape of the real space image. This can be done by choosing a set of lines bounding the object, for example the image in Fig. 4.10a was created using the lines perpendicular to the vectors: $(6,-4)$, $(-6,4)$, $(2,-12)$, $(-10,2)$, $(-10,-11)$, $(10,10)$, $(10,-4)$ and $(3,11)$. Additionally, the image may be convolved with some function to avoid an abrupt black-white transition in the image. In the case of this example, we chose to convolve the image with a radial function of the form $(1+e^{\sqrt{x^2+y^2}})^{-1}$ maintaining a rather abrupt cutoff of the density. An alternative method is to define a 3D image with a set of bounding planes and project the resulting bounded density onto a plane. The latter method is more exact in that the FFT of the projected 2D image is a single slice through the 3D Fourier transform of the original object. Nevertheless, reasonable results may be obtained using the former method, as will be shown in the following sections.

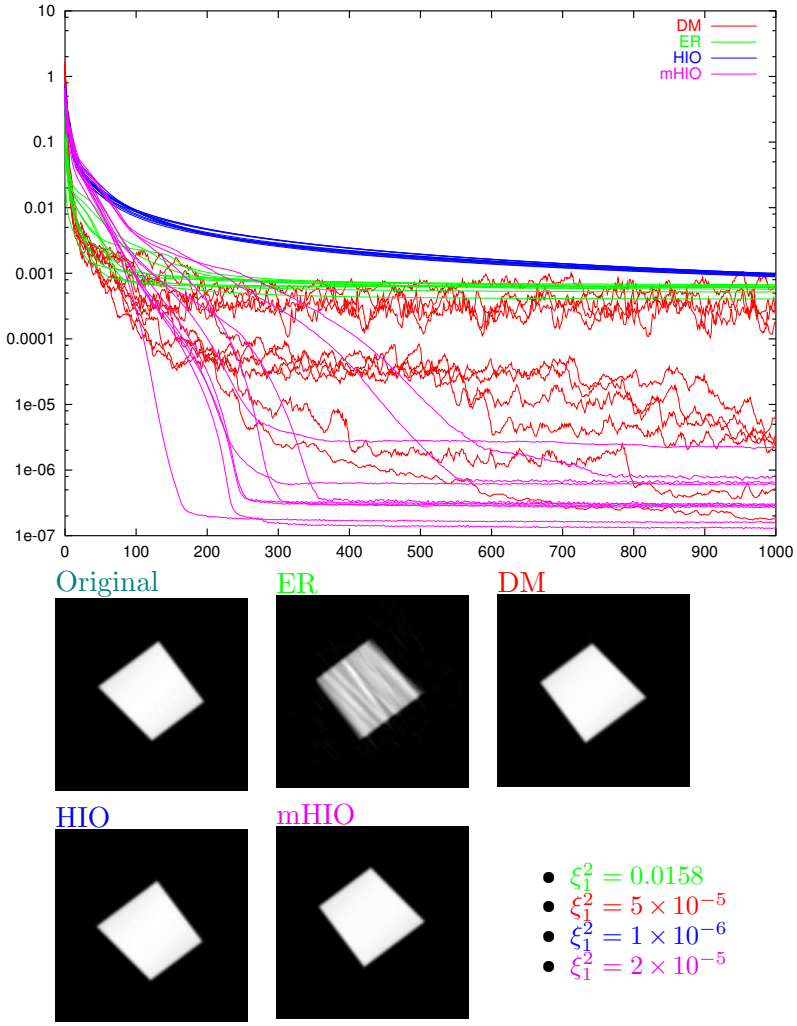


Figure 4.9: Polygon smoothed by convolution with a Gaussian. Here, all algorithms succeed in finding the shape and only ER reconstructs false contrast within the object. It may be that smooth objects are easier due to the reduced aliasing in reciprocal-space.

4.3.2 Noise in CXD experiments

In general we can group the expected noise in an experiment of this kind into three categories. The inherent photon shot noise arises from the probability that a diffracted photon will arrive in a pixel. This probability follows a Poisson distribution and so the expected uncertainty in a measurement of N photons is $\sqrt{N}$. This shot noise is simulated by first choosing the number of photons desired in the simulation. Then, the square modulus of the Fourier transform of the real space image is calculated. The ratio of desired photons to integrated intensity is then found and each pixel in the simulated diffraction pattern is assigned a number of photons drawn from a Poisson random number distribution with mean equal to the square modulus of the Fourier transform of the real space image multiplied by the ratio calculated previously. In this way, the intensity is used as a probability distribution to determine the pixel location of each photon.

Although great lengths are taken to eliminate the contribution of nearby scatterers in the sample, it is rarely possible to completely remove their effect in a real experiment. Therefore, we add additional shot noise to the simulated diffraction pattern. This is achieved by declaring an expected mean number of photons for each pixel in the detector. This mean is completely independent of the calculated Fourier transform and is used as the mean for a Poisson random distribution from which a number is drawn for each pixel in the array. The returned number is then cast to an integer, since fractional photons are disallowed in the simulation. At this point, the "photons" are converted to ADUs by drawing a number from a Gaussian random distribution with mean and variance extracted from experimental data by fitting photon peaks in a histogram.

The last source of noise in this simulation is an approximation of the noise caused by the background of the CCD detector. For simplicity, this is estimated by a constant background level which serves as the mean in a Gaussian random number distribution. Unlike the previous two sources, this number is not cast to an integer, although the ADC counts in the actual experiment are always integer values. This is done so that very small values of the background relative to a photon's ADC count level may be explored.

Once the level of noise from the three preceding sources has been chosen,

a signal to noise ratio is calculated according to:

$$SNR \equiv \frac{\sum_{n=1}^{N} I_n}{\sum_{n=1}^{N} \sqrt{I_n} + P_n + B_n}, \tag{4.5}$$

where I_n is the intensity collected in the nth pixel of the N pixel detector, $\sqrt{I_n}$ is the inherent error in measuring an intensity I_n, P_n is the noise added due to scattering measured in a pixel from sources other than the sample, and B_n is an estimate of the background of the measurement in a pixel.

4.3.3 Simulation procedure

The simulated diffraction patterns for the next section were created by means of the following procedure:

i) choose a crystal shape,

ii) perform an FFT on the crystal shape,

iii) select the number of photons collected in the detector,

iv) choose a mean value for the anomalous scattering noise,

v) optionally select a number of ADUs per photon,

vi) set the variance about the mean ADUs per photon,

vii) choose a background level, and

viii) set the variance about the background level

For example, the simulated pattern in Fig. 4.10c was created from a real space object's, shown in Fig. 4.10a, FFT, shown in Fig. 4.10b. In this case, we chose to add 9×10^4 photons to the pattern based on the probability of a photon arriving at a pixel given by the FFT of the real space object. This procedure yields a simulated measurement, shown in Fig. 4.10c. We then take the phase information contained in the FFT of the real space object, apply it to the simulated diffraction pattern, and back transform it to yield a real space object which we consider to be the "truth image", that is, Fig. 4.10d is the solution to the phased simulated measurement that we seek. This is important in judging the robustness of the fitting algorithms.

A program was written in C, **`interactivenoise`**, to generate a noisy data array from an input array. The input array can be generated either

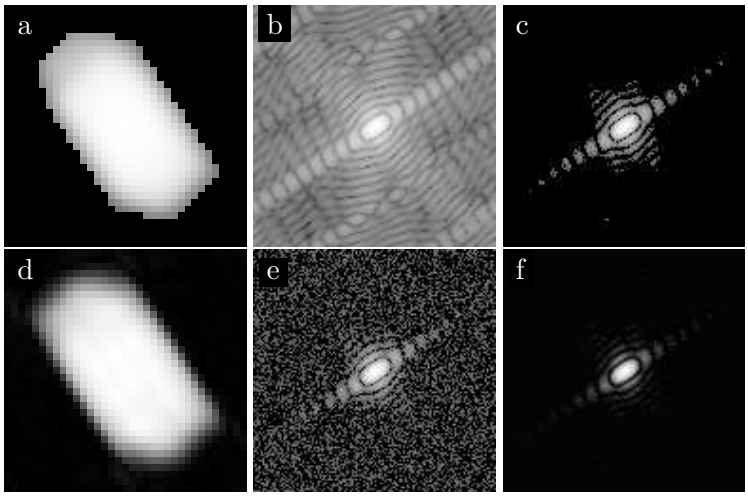


Figure 4.10: a is shape, b is FFT(a), c is b with only x photons, d is the real-space object which corresponds to c, e is c with added shot noise, f is e with added background

by specifying the planes, as described above, or by creating a bitmap image (ppm) in an image manipulation program and converting it to a data array. The FFT of this array is the input expected by the **interactivenoise** program. The program may be invoked on the command line with all of its parameters, for example

`interactivenoise <input> <output> <seed> <`$N_{\mathtt{ph}}$`> <`$N_{\mathtt{ADU}}$`> <`$\sigma^2_{\mathtt{ph}}$`>`
`<`$N_{\mathtt{back}}$`> <`$\sigma^2_{\mathtt{back}}$`> <`$N_{\mathtt{alien}}$`>`

where the seed is used to initialize the standard drand48() C function, N_{ph} is the desired number of photons, N_{ADU} is used to specify the mean number of ADUs per photon in the output array, σ^2_{ph} is the Gaussian variance of the distribution from which the counts per photon is actually drawn, N_{back} is the mean background level, σ^2_{back} is the variance of the Gaussian distribution about N_{back}, and N_{alien} the mean number of photons per pixel to add while simulating scatter from alien particles. The order of operations is the same as above, first the intensity is calculated, then a number of photons is assigned to each pixel based on the ratio of the intensity in the pixel to the total intensity and the number of photons to be assigned in the array. Then, photons are added to pixels to simulate alien scattering by drawing a number from a Poisson random number distribution[72] with mean N_{alien} and cast to an integer number of photons. In this way, the probability of alien scatterer is uniform across the array. Next, a Gaussian random number distribution[72] with mean N_{ADU} and variance σ^2_{ph} is used to convert the number of photons in a pixel into a number of ADUs collected in that pixel.

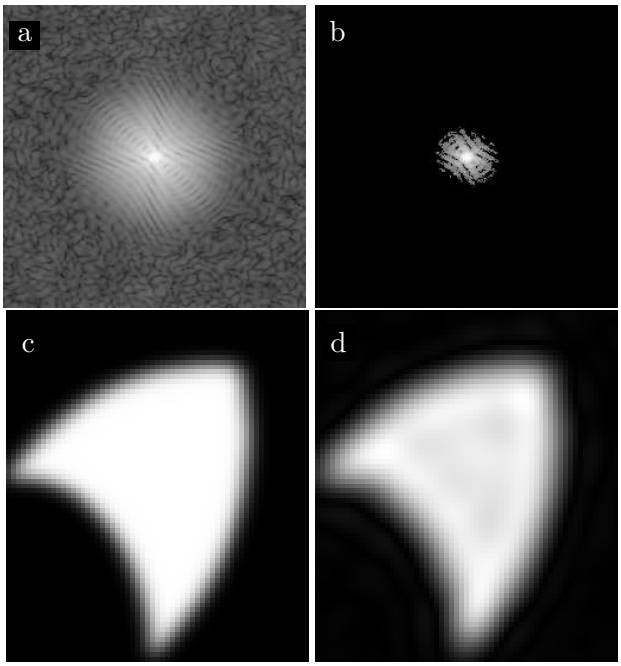


Figure 4.11: a is the "diffraction pattern"–*i.e.*, the FFT of the real-space object–b is the result of allocating9×10^4 photons using the FFT in a as a probability distribution, c is the real-space object, and d is the real-space object which corresponds to b, when the phases of a are used.

For each photon, one number is drawn from the distribution and the sum of all such numbers is the final number of ADUs in that pixel due to photons incident on the detector. Finally, contribution due to dark current, the ADC offset–which is generally larger than the amplifier noise to avoid zero crossings in what is generally a positive signal–and other electronic noise is grouped together into a bias added to each pixel. The bias is calculated from a second Gaussian random number distribution, this time with mean N_{back} and variance σ^2_{back}. A single number of ADUs is drawn from this distribution for each pixel and added to the ADU total due to photons.

4.4 Testing the Algorithms

Following the procedure discussed in the previous section, a simulated real-space density was used to create a "noisy diffraction pattern." One important difference is that the real-space object chosen for these examples is strongly asymmetric. This is done to elucidate the problem of stagnation between twins, which is known to arise when utilizing these iterative methods[33]. Fig. 4.11a,c shows the real- and reciprocal-space images used

to obtain 4.11b,d, of which b is used as the Fourier magnitude data to be fit and d is used to calculated ξ_1^{orig}, given by [4.4]. By calculating the histogram of an experimental CXD pattern, we found that on average, one photon generates 324 ADUs and the distribution of ADUs is roughly Gaussian, with $\sigma^2 = 1024$. These values are used for all simulated photons incident on the array, whether they are supposed to arise from scattering from the sample or are attributed to some alien background scattering.

Fig. 4.12 shows a histogram calculated from CXD data, red dots, and the Gaussian distributions used to generate the simulated pattern, green line. This particular data set is a very long 3D series measured from the diffraction of a Pb crystal. Since both the number of photons captured and the dark current accumulated far exceed the values used for the 2D simulation here, the amplitudes of the background and n-photon peaks in the data do not agree with the simulation, but both the ADUs per photon and the distributions about the mean background level and about the mean n-photon peak positions show good agreement between the values selected for simulation and the measurement. There is a more complete discussion of the characteristics of the detector in Chapter 3.

This section compares the behavior of ER, HIO, and DM in the presence of different kinds on noise–with fixed parameters for HIO and DM–in Sections 4.4.1 and 4.4.2 and the effect of the parameters on HIO and DM in Sections 4.4.3 and 4.4.4. In Section 4.4.5 we will increase the number of photons in the simulation to explore the behavior of the algorithms with increasingly large signal. Finally, we study the utility of the mixing of ER and HIO cycles used to fit experimental CXD data in Section 4.4.6.

The fitting was conducted in an array of the same size as the generated patterns shown in Fig. 4.11, which was 256×256 px. In each test, the reciprocal-space Fourier modulus constraint was the same simulated pattern for all algorithms. ER, HIO, and DM all use a 60×70 px rectangular region as a finite support constraint. This is the dimension of the viewable area in Fig. 4.11c, so it is evident that the resultant Fourier transform is oversampled by the earlier definition. For reference, if the real-space truth object is compared to the part of itself that lies only within the real-space support, a baseline for comparing ξ_1^{orig} may be arrived at: $\xi_{trunc}^{orig} = 0.005$. Any value of $\xi_1^{orig} > \xi_{trunc}^{orig}$, is an imperfect fit and if $\xi_1^{orig} < \xi_{trunc}^{orig}$, the fitting has extracted information from the "measurement" that contradicts the real-space support. This specific value of ξ_{trunc}^{orig} depends on the number of photons, and so will be recalculated and presented whenever this quantity is altered.

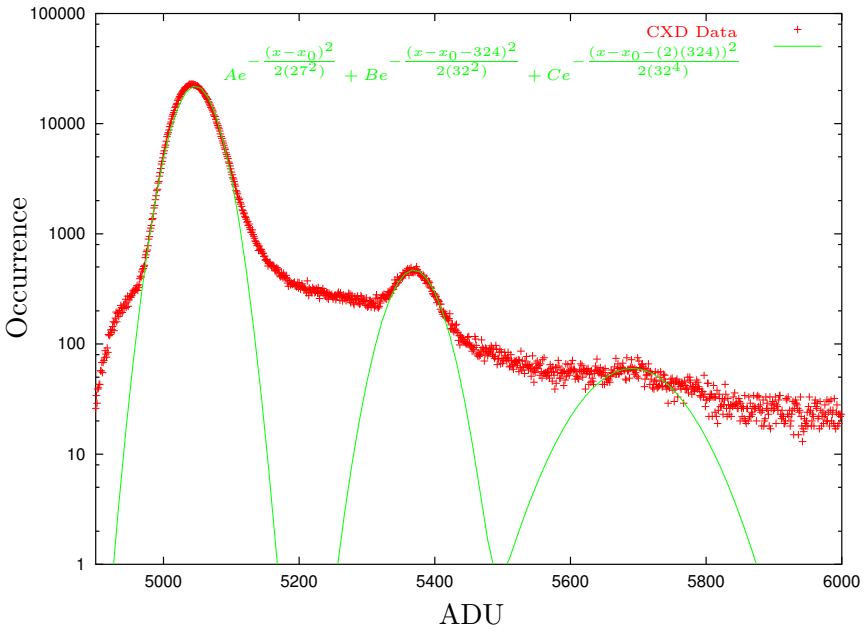


Figure 4.12: Histogram of CXD data with overlaid functional form of the simulation. Each term has been rescaled to show agreement with data, but n-photon peak positions relative to background, distribution about the mean background level, and photon peak distributions are the same as those used in simulation.

The precise real-space constraint varies from algorithm to algorithm, so we must be precise in specifying them. An iterate in ER is required to have only positive real density in real-space, so any complex number reconstructed inside the support is projected onto the real axis in the complex plane. Any value violating the support constraint is set to zero. Although HIO is a special case of DM, it has been implemented independently and uses both the finite support and a positivity constraint, meaning that all negative values are set to zero. DM, in this case, uses only the finite support constraint, so all values which reconstruct within the support are kept. Each run consists of five fits of 500 iterations of a particular algorithm for each simulated pattern. A different set of random starting phases is used for each fit. The best fit for each run is displayed in Appendix A. A more complete description of the details of fitting is given in Chapter 3.

4.4.1 Alien scattering

Shot noise levels of this kind are highly variable in CXD experiments. For the case of large Au crystals–discussed in Chapter 3–it is possible to reduce the number of illuminated crystals so that the CXD patterns are well sep-

Table 4.1: Shot noise table

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
319	DM	1	-1	1	500	0.1704	–	0.02418	0.02705	0.02918
319	ER	–	–	–	500	0.006136	0.006044	–	0.02848	0.053
319	HIO	1	–	–	500	0.1711	0.07629	–	0.05975	0.04731
296	DM	1	-1	1	500	0.1679	–	0.02485	0.05854	0.03807
296	ER	–	–	–	500	0.005258	0.005167	–	0.00272	0.03585
296	HIO	1	–	–	500	0.175	0.07603	–	0.1529	0.1157
163	DM	1	-1	1	500	0.169	–	0.03122	0.04723	0.0397
163	ER	–	–	–	500	0.008922	0.008743	–	0.04658	0.04506
163	HIO	1	–	–	500	0.1765	0.07702	–	0.04492	0.02151
32.9	DM	1	-1	1	500	0.1796	–	0.05076	0.02536	0.1263
32.9	ER	–	–	–	500	0.03001	0.02923	–	0.2168	0.1509
32.9	HIO	1	–	–	500	0.1811	0.07085	–	0.0586	0.03078
3.56	DM	1	-1	1	500	0.2963	–	0.2342	0.07448	0.03076
3.56	ER	–	–	–	500	0.1346	0.1308	–	0.272	0.361
3.56	HIO	1	–	–	500	0.3001	0.09856	–	0.224	0.1141

arated in reciprocal-space, leaving the shot noise contribution to be almost entirely air and slit scatter. In this case, the distribution of noise across the detector is essentially uniform. If the CXD patterns from multiple incoherently illuminated areas on the sample are too close together in reciprocal space, there is an increased probability for a photon to land in a particular region of the detector. The former case is assumed for the purpose of these simulations.

Since the level of alien scattering depends upon many factors, including the amount of air the beam must pass through, the location of mirrors, windows, and other optical elements, and possibly the scattering from the sample's substrate, a wide variety of values should be simulated. To explore the realistic range of shot noise contamination, a spatially uniform probability is assumed with mean photons per pixel chosen to be $0, 0.0005, 0.005, 0.05$, and 0.5. These values yield SNRs, from [4.5], that range from 319 to 3.5. Comparison by eye (See Appendix A, Section A.1.) would lead one to suspect that a mean of 0.005 is approximately what is seen in best case experiments where air scatter is not fully eliminated.

For these fits, we chose the DM parameters $\beta = 1$, $\gamma_1 = -1$, and $\gamma_2 = 1$, which may be recalled as both the HIO and maximally contractive parameters. Since $\beta = 1$ was also chosen for HIO, the DM and HIO would be expected to give the same result, were it not for the additional positivity constraint during HIO.

Table 4.1 summarizes the results of fits to simulated data with various levels of alien scatter. It is particularly interesting to note that for fits using ER ξ_1^{orig} gets larger, *i.e.*, produces a worse fit to the truth image, much more quickly than χ^2. This is most noticeable for the simulation with $SNR = 32$ where during ER $\chi^2 = 0.03$ and $\xi_1^{orig} = 0.22$ while for HIO $\chi^2 = 0.18$ and $\xi_1^{orig} = 0.03$, implying that ER does a much better job at fitting the data and a much worse job at finding the correct solution to the inverse problem.

Another important feature is the difference in the behavior of the error metrics for different algorithms. Before implementing DM, χ^2 and χ^2_{Re} were used to characterize the quality of the fit to CXD data. These two metrics have essentially the same value during iterations of ER, but since HIO makes no real attempt to fit the data directly, the latter is often used to track the progress of an iterate. Because of the way DM is works, it is very difficult to calculate χ^2_{Re}, so instead ζ_{DI} and χ^2 are calculated. While χ^2_{Re} and ξ_1^{orig} seem to be correlated during HIO, this is true of neither ER nor of DM with ζ_{DI} and ξ_1^{orig}.

Finally, there does appear to be a correlation between ξ_1^2 and ξ_1^{orig}. This

is promising, since ξ_1^2 is directly measurable at the conclusion of fitting real CXD data. Indeed, if no such correlation existed for low noise simulation, it might be an indication that the problem possesses multiple solutions. In general, the effect of added shot noise on the real-space result seems to be to a random "speckling" or generally noisy appearance in the case of ER and the presence of random amplitude fluctuations inside the object for HIO and DM.

From these examples, it seems that realistic values of shot noise will cause a degradation that manifests itself primarily by slowly varying density contrast within the real-space object. The algorithms did, however, perform very well in recovering the lowest frequency information, *i.e.*, the general shape of the object, and almost always recovered the essentially correct compact shape.

4.4.2 Bias level

A bias level, or background, in our CXD measurement comes primarily from the so-called "dark current" of the CCD. The dark current signal is approximately linear in time, so its precise value will depend critically upon the exposure time in the experiment. The origin of the current is in the thermal excitation of electron-hole generation, thus the expected distribution is Poisson. However, as a simplifying measure, all aspects of the CCD that might give rise to a bias level–read noise, $1/f$ amplifier noise, *etc.*–have been combined and through a fit to a histogram of a CXD pattern the distribution of the background is seen to be very nearly Gaussian. For the simulations here, the bias level is set at 1000 ADUs with $\sigma^2 = 729$. This image is in Appendix A, Section A.2, and the $SNR = 0.4$ entry in Tables 4.2 and 4.3. To generate higher SNR simulations, a constant is subtracted from each pixel in the array, mimicking a primitive experimental background subtraction. These subtractions are 10%, 30%, 50%, 70%, 90%, 95%, 99%, and 100% of the background level, yielding SNR values between 2 and 52. The DM and HIO parameters are the same as those used in Section 4.4.1.

In general, the same trend of ξ_1^{orig} with SNR is observed. Perhaps the most striking result is the difference in ξ_1^{orig} for $SNR = 40$ and $SNR = 52$, where the quality of the fits actually seem to degrade slightly with the removal of noise. Further, we can clearly see that added bias has a markedly different effect on the iterate than random shot noise discussed above. The most telling sign of this is that while very high levels of shot noise caused a generally blurry real-space result to occur with DM and HIO, high noise,

Table 4.2: Bias table

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
0.437	DM	1	-1	1	500	0.1325	–	0.0281	0.5319	0.5031
0.437	ER	–	–	–	500	0.01681	0.0168	–	0.5785	0.9429
0.437	HIO	1	–	–	500	0.1307	0.05713	–	0.5563	0.9147
1.91	DM	1	-1	1	500	0.1372	–	0.02894	0.4305	0.4564
1.91	ER	–	–	–	500	0.01856	0.01855	–	0.8166	0.9023
1.91	HIO	1	–	–	500	0.1351	0.05771	–	0.5496	0.7942
2.42	DM	1	-1	1	500	0.1492	–	0.03038	0.3656	0.4178
2.42	ER	–	–	–	500	0.0158	0.01578	–	0.7657	0.907
2.42	HIO	1	–	–	500	0.1454	0.06346	–	0.3638	0.5571
3.33	DM	1	-1	1	500	0.1626	–	0.03227	0.3362	0.4049
3.33	ER	–	–	–	500	0.01611	0.01608	–	0.6634	0.8734
3.33	HIO	1	–	–	500	0.1627	0.06892	–	0.3161	0.5287
5.33	DM	1	-1	1	500	0.1833	–	0.03602	0.2342	0.3928
5.33	ER	–	–	–	500	0.01358	0.01354	–	0.3984	0.6857
5.33	HIO	1	–	–	500	0.1853	0.07717	–	0.2322	0.3912

Table 4.3: Bias table, continued

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
13.3	DM	1	-1	1	500	0.2121	–	0.05651	0.1114	0.1691
13.3	ER	–	–	–	500	0.01168	0.01157	–	0.1861	0.6413
13.3	HIO	1	–	–	500	0.2201	0.08064	–	0.2465	0.2469
21.1	DM	1	-1	1	500	0.2148	–	0.08062	0.09147	0.09583
21.1	ER	–	–	–	500	0.01617	0.01587	–	0.137	0.2847
21.1	HIO	1	–	–	500	0.2265	0.07835	–	0.1074	0.0766
40.3	DM	1	-1	1	500	0.1986	–	0.05525	0.03527	0.02956
40.3	ER	–	–	–	500	0.02369	0.02319	–	0.2358	0.202
40.3	HIO	1	–	–	500	0.2025	0.07251	–	0.07802	0.056
52.1	DM	1	-1	1	500	0.2087	–	0.05154	0.1718	0.1119
52.1	ER	–	–	–	500	0.01956	0.01916	–	0.1366	0.05835
52.1	HIO	1	–	–	500	0.1986	0.0755	–	0.06138	0.03288

bias added simulations result in a real-space result with rapidly varying amplitude inside a coarsely defined region. Indeed, it is not until SNR reaches 40 that the real-space object becomes well defined.

The reason for this modulation is an artifact that has been reported to occur in iterative fitting methods: phase singularities[36]. These singularities, or vortices, are clearly visible in the images in Appendix A. They are comprised of a group of two to four neighboring pixels whose complex amplitudes approach zero and a rapid change in the phase, of magnitude $2\pi n$–where n is an integer–occurs as a loop around the group is traversed. It is extremely uncommon for any one pixel to be a true zero, although one pixel generally has a smaller amplitude than its neighbors. It has been posited[36] that the true zero is preferentially not on a sampled pixel, since this would be a true zero in the analytic continuation of the reconstruction which does not exist in the analytic continuation of diffraction data, hence the two reciprocal-space functions cannot be same nor have the same Fourier transform.

These vortices are clearly present in simulations that have as little as 10 ADUs of bias, about 3% of the ADUs caused by a one photon event. Further, the vortices appear in ER and DM reconstructions, but not during HIO. Their formation must then be inhibited by the enforcement of the positivity constraint, since that is the only difference between DM and HIO as described earlier. These vortices will be discussed in greater detail in Section 4.5.

Examining the error metrics for these fits, we notice something quite unusual: for very high noise, $SNR < 10$, the error metrics are all astonishingly low while ξ_1^{orig} remains over 0.2, $\sim 45\%$ error per pixel. This is worrying if we only use the error metrics as a measure of the quality of fit, but fortunately ξ_1^2 remains high, indicating a poor fit even when ξ_1^{orig} is unmeasurable. It is likely that this fault in the error metric is due to the combination of vortices, which are point-like defects, and the increased integrated intensity that comes from adding a bias level to each of 256^2 pixels. It is now clear that bias is potentially much more damaging to the quality of the reconstruction than the alien scatter discussed earlier.

4.4.3 HIO parameters

As demonstrated by the illustrations of 2D trajectories in Chapter 2, it is not clear what the best values for the β parameter of HIO should be. Since we cannot predict *a priori* what value might be best for noisy data, we try

Table 4.4: HIO parameter table

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
52	HIO	0.2	–	–	500	0.1092	0.03135	–	0.04133	0.03881
52	HIO	0.5	–	–	500	0.166	0.05613	–	0.05118	0.03536
52	HIO	0.7	–	–	500	0.1871	0.06564	–	0.1808	0.1316
52	HIO	0.9	–	–	500	0.1944	0.07333	–	0.06196	0.04242
52	HIO	0.99	–	–	500	0.2009	0.07931	–	0.0666	0.03333
52	HIO	1.01	–	–	500	0.2021	0.0785	–	0.07234	0.03045
52	HIO	1.2	–	–	500	0.2017	0.08097	–	0.1641	0.142
52	HIO	1.5	–	–	500	0.1825	0.07731	–	0.1053	0.0601
52	HIO	1.7	–	–	500	0.1541	0.06643	–	0.2527	0.1045
52	HIO	-0.2	–	–	500	0.1092	0.02981	–	0.04976	0.04056
52	HIO	-0.7	–	–	500	0.1879	0.07005	–	0.06566	0.03445
52	HIO	-0.9	–	–	500	0.1979	0.07643	–	0.06229	0.1547
52	HIO	-1.2	–	–	500	0.208	0.08482	–	0.06558	0.03211
52	HIO	-1.5	–	–	500	0.183	0.07866	–	0.1425	0.07786

a range of values with a single simulated data set. For this trial, we chose the 100% background subtracted image from Section 4.4.2. This is a logical choice, since it mimics the procedure for background subtracting CXD data.

Table 4.4 details the result of fitting this simulated data for values of beta from -1.5 to 1.7. Again, the results are all plotted in Appendix A. The general trend is that as $|\beta|$ increases, so too does ξ_1^{orig}, meaning that small βs and therefore small steps in approaching the intersection are favored. It's interesting to note that $\beta = \pm 0.2$ give almost identical results, despite the fact that $\beta\gamma_1 = (-0.2)(-1) > 0$, which violates the convergence condition of DM derived from [2.33]. Of course, it has already been shown that for some values of its parameters and some shapes of the constraint sets, DM will wander in the vicinity of a solution even if it does not converge to it. This might be manifesting itself in the current situation as χ^2_{Re} is lowest for $\beta = -0.2$, but ξ_1^{orig} is lowest for $\beta = 0.2$. Nevertheless, at very high SNR, *i.e.*, very strict background subtraction, the reconstructions from the simulated data begin to resemble those from simulations of low SNR with only shot noise. This is expected, since at perfect background subtraction, half of all pixels that do not contain true signal will be zero, hence the strongly background subtracted simulation should begin to resemble a shot-noise added simulation.

4.4.4 DM parameters

The situation for DM is even worse than for HIO: it is profoundly difficult to choose parameters without merely trying a few to see how they work out. Table 4.5 details various choices of $\beta > 0$, γ_1, and γ_2. We immediately see that there is no correlation between β and ξ_1^{orig} as was seen for HIO. DM fits with $\beta = 0.7$, $\gamma_1 = -2.85$, $\gamma_2 = 2.85$ and $\beta = 1.2$, $\gamma_1 = -1$, $\gamma_2 = 0.833$ both have $\xi_1^{orig} \simeq 0.02$. The former is case where nearly violating the convergence guarantee yields final iterates close to the solution. The latter is a case of the HIO parameters that appeared in Table 4.4. For this value of β and this simulated diffraction pattern, it seems that the positivity constraint does more harm than good since with positivity we get $\xi_1^{orig} = 0.16$ and without it we get $\xi_1^{orig} = 0.02$. Interestingly, examining Table 4.6–similar to Table 4.5, except with $\beta < 0$–we see that another set of parameters– $\beta = -0.4$, $\gamma_1 = 2.5$, $\gamma_2 = -2.5$, which is maximally contractive–also gives a comparable value of ξ_1^{orig}.

In summary, of the parameter values listed in Tables 4.5 and 4.6, the best fits–with errors per pixel of $\sim 15\%$– are obtained by one of three sets of

Table 4.5: DM parameter table, with positive β

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
52.1	DM	0.2	-10	10	500	0.04667	–	0.01171	0.1309	0.08005
52.1	DM	0.2	-10	2	500	0.08817	–	0.01	0.3532	0.2689
52.1	DM	0.2	-1	5	500	0.1131	–	0.004454	0.06686	0.05836
52.1	DM	0.2	-5	5	500	0.09357	–	0.002977	0.2284	0.14
52.1	DM	0.7	-1.4286	1.4286	500	0.1761	–	0.02382	0.1765	0.08359
52.1	DM	0.7	-1	1.4286	500	0.1823	–	0.02149	0.07773	0.0275
52.1	DM	0.7	-2.85	0.5	500	0.2256	–	0.1252	0.316	0.1067
52.1	DM	0.7	-2.85	2.85	500	0.07214	–	0.05591	0.0203	0.1719
52.1	DM	0.9	-1.11	1.11	500	0.1948	–	0.0394	0.1364	0.1164
52.1	DM	0.9	-1.22	2.22	500	0.1335	–	0.04231	0.07262	0.1489
52.1	DM	0.9	-1.22	2.5	500	0.123	–	0.07407	0.04279	0.06177
52.1	DM	0.9	-1	1.11	500	0.1886	–	0.03815	0.04683	0.02937
52.1	DM	1.0	-1.5	5	500	0.7618	–	2.454	0.1951	0.1332
52.1	DM	1.0	-2	2	500	0.08228	–	0.1114	0.1417	0.1035
52.1	DM	1.0	-3	0.5	500	0.05244	–	0.4037	0.2649	0.2607
52.1	DM	1.2	-0.833	0.833	500	0.2176	–	0.1035	0.08854	0.03794
52.1	DM	1.2	-1.66	0.8	500	0.1104	–	0.1683	0.1715	0.1106
52.1	DM	1.2	-1.66	1.66	500	0.08077	–	0.1504	0.05079	0.1074
52.1	DM	1.2	-1	0.833	500	0.195	–	0.0777	0.02485	0.01088
52.1	DM	1.7	-0.1	1.17	500	0.3167	–	0.9691	0.8628	0.5132
52.1	DM	1.7	-0.58	0.58	500	0.4553	–	2.362	0.9747	1.328
52.1	DM	1.7	-1.17	1.17	500	0.07293	–	0.3586	0.04378	0.09156
52.1	DM	1.7	-1	0.58	500	0.1405	–	0.3433	0.108	0.02312

parameters: a set which ought not give convergence to a solution, a set corresponding to HIO with positive β, and a set that is maximally contractive with negative β. Further, there are 11 sets with errors per pixel of $\sim 25\%$.

The DM is clearly a very versatile algorithm, but its enormous parameter space makes it similarly hard to use effectively. It is unfortunate that unlike the case of HIO, where small β's seem preferable when used on noisy images, no clear pattern has emerged in the case of the DM parameters.

4.4.5 Photon number

It is also useful to investigate the behavior of these algorithms with changing photon number–*i.e.*, changing the number of photons present in the simulation–so that a heuristic might be developed to aid in experimental design. Tables 4.8 and 4.9 list the results of this trial. The general trend is for ξ_1^{orig} to decrease with increasing photon number. It is worth noting that we again observe that χ^2_{Re} and χ^2 are not well correlated to ξ_1^{orig}, although ξ_1^2 does tend to track with ξ_1^{orig}.

Interestingly, the relationship between ξ_1^{orig} and photon number is not strictly monotonic. This is slightly troubling, since the expected behavior is for ξ_1^{orig} to decrease with increasing photon number. While generally ξ_1^{orig} does decrease with the number of photons in the simulated measurement, ER fails to find good solutions for the $SNR = 1.6 \times 10^4$ and $SNR = 2.4 \times 10^3$ cases, but finds much better solutions at intermediate values. HIO displays this behavior to a lesser extent, for example, finding a much better solution at $SNR = 2.4 \times 10^3$–with $\xi_1^{orig} = 0.006$–than at $SNR = 2.3 \times 10^4$–with $\xi_1^{orig} = 0.011$. DM, on the other hand, demonstrates very little of this behavior, with values of ξ_1^{orig} dropping with increasing photon number. The errant behavior of ER and HIO is probably caused by particularly bad random starting points or the inability to completely explore the solution space.

4.4.6 Photon number and mixed algorithms

Finally, to close this section, an investigation of the properties hinted at in Chapter 2 for the case of non-intersecting sets is presented. Here, the simulated CXD patterns from the last section on photon number are used to demonstrate the effectiveness of a mixture of algorithms. Rather than 500 iterations of each of the algorithms, we use a combination of ER and HIO: 150 ER, 50 HIO, and 100 ER. The results are summarized in Table 4.10 and the appendix.

Table 4.6: DM parameter table, with negative β

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
52.1	DM	-0.4	1	-2.5	500	0.08632	–	0.005405	0.09088	0.08054
52.1	DM	-0.4	2.5	-2.5	500	0.08479	–	0.009633	0.02522	0.02702
52.1	DM	-0.4	4.5	-5.5	500	0.06692	–	0.07095	0.2983	0.28
52.1	DM	-0.4	5	-1	500	1.437	–	0.154	1.225	0.9121
52.1	DM	-0.4	5	-5	500	0.06904	–	0.02778	0.1041	0.09892
52.1	DM	-0.8	1	-1.25	500	0.1121	–	0.02895	0.1456	0.105
52.1	DM	-0.8	1.25	-1.25	500	0.1133	–	0.03524	0.145	0.07394
52.1	DM	-0.8	2.5	-1	500	1.197	–	0.64	1.895	1.919
52.1	DM	-0.8	2.5	-2.5	500	0.09929	–	0.1001	0.06086	0.02282
52.1	DM	-0.8	2.6	-2.2	500	0.1607	–	0.1683	0.04083	0.02456
52.1	DM	-1.2	0.833	-0.833	500	0.118	–	0.06585	0.08756	0.05565
52.1	DM	-1.2	1	-0.833	500	0.1216	–	0.08045	0.03579	0.02685
52.1	DM	-1.2	1.66	-0.833	500	0.7947	–	1.437	1.318	1.254
52.1	DM	-1.2	1.66	-1.66	500	0.6039	–	1.276	0.1063	0.1453
52.1	DM	-1.5	0.66	-0.66	500	0.1325	–	0.0878	0.1071	0.08742
52.1	DM	-1.5	1	-0.66	500	0.1218	–	0.1704	0.05975	0.02811
52.1	DM	-1.5	1.2	-1.2	500	0.1034	–	0.4056	0.1276	0.1227
52.1	DM	-1.5	1.33	-1.33	500	0.4971	–	2.25	1.254	1.263

Table 4.7: As photon number increases, the best possible fidelity, ξ_{trunc}^{orig}, decreases, as expected. These numbers represent the best expected ξ_1^{orig} values for the corresponding SNR in following tables.

Photon Number	SNR	ξ_{trunc}^{orig}
9×10^4	2.867×10^2	5.794×10^{-3}
2×10^5	3.965×10^2	3.510×10^{-3}
5×10^5	5.895×10^2	1.830×10^{-3}
1×10^6	8.026×10^2	1.201×10^{-3}
2×10^6	1.104×10^3	7.124×10^{-4}
5×10^6	1.701×10^3	3.649×10^{-4}
1×10^7	2.374×10^3	2.316×10^{-4}
5×10^7	5.194×10^3	1.073×10^{-4}
1×10^8	7.308×10^3	8.674×10^{-5}
5×10^8	1.623×10^4	6.992×10^{-5}
1×10^9	2.290×10^4	6.690×10^{-5}

Examining this table demonstrates that the values of ξ_1^{orig} obtained in this way are at least as low as those of the previous section, and are in most cases even lower. Fig 4.13 is a plot of error metric versus iteration for DM, ER, HIO, and ER/HIO for the case of 5×10^6 photons at $SNR = 1700$. For DM, the metric plotted is ζ_{DI} while χ^2_{Re} is shown for the others. This figure clearly shows that the mixture of ER and HIO gives a lower error metric and, perhaps more importantly, tends to drive all the iterates to the same solution, as demonstrated by the close grouping of error metrics in Fig. 4.13 and the numerical values in Table 4.10.

One unexpected result of this simulation is that for a few simulated CXD data sets with very large numbers of photons, that is, high SNR, the fidelity, ξ_1^{orig}, falls below the baseline value, ξ_{trunc}^{orig}. This is seen to occur twice in Fig. 4.13, at $SNR = 5.19 \times 10^3$ and 1.62×10^4. Apparently, the combination of ER and HIO has led to a slight violation of the support constraint which has not been undone in later iterations of ER.

It should also be noted that the behavior of ξ_1^{orig} with increasing photon number is monotonic when fitting with ER and HIO together, at least until ξ_1^{orig} is of order 10^{-4}. An interesting question which remains to be answered, is if the best performing values of β in HIO alone and ER/HIO combinations are the same.

Table 4.8: Photon number table one

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
287	DM	1	-1	1	500	0.1792	–	0.03144	0.05735	0.05126
287	ER	–	–	–	500	0.007012	0.006883	–	0.02442	0.152
287	HIO	1	–	–	500	0.1907	0.08124	–	0.1751	0.1314
396	DM	1	-1	1	500	0.1409	–	0.01781	0.02079	0.05401
396	ER	–	–	–	500	0.006937	0.006838	–	0.3722	0.1639
396	HIO	1	–	–	500	0.1458	0.06661	–	0.02759	0.01718
589	DM	1	-1	1	500	0.1088	–	0.01248	0.02229	0.007585
589	ER	–	–	–	500	0.002718	0.002678	–	0.02987	0.05489
589	HIO	1	–	–	500	0.1123	0.05687	–	0.01208	0.00926
803	DM	1	-1	1	500	0.08642	–	0.01069	0.0686	0.02679
803	ER	–	–	–	500	0.002061	0.002037	–	0.05825	0.3476
803	HIO	1	–	–	500	0.09202	0.04993	–	0.006504	0.00621
1.1e+03	DM	1	-1	1	500	0.08824	–	0.008515	0.1662	0.01094
1.1e+03	ER	–	–	–	500	0.001805	0.001789	–	0.04065	0.1668
1.1e+03	HIO	1	–	–	500	0.07496	0.04401	–	0.005319	0.004195
1.7e+03	DM	1	-1	1	500	0.05458	–	0.003422	0.007026	0.1526
1.7e+03	ER	–	–	–	500	0.0007829	0.0007755	–	0.01552	0.1835
1.7e+03	HIO	1	–	–	500	0.05733	0.03642	–	0.005671	0.002426

Table 4.9: Photon number table two

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
2.37e+03	DM	1	-1	1	500	0.04305	–	0.002486	0.008317	0.1437
2.37e+03	ER	–	–	–	500	0.002279	0.002275	–	0.159	0.138
2.37e+03	HIO	1	–	–	500	0.0465	0.03073	–	0.006179	0.001505
5.19e+03	DM	1	-1	1	500	0.02764	–	0.0009463	0.004203	0.04702
5.19e+03	ER	–	–	–	500	0.0009551	0.0009541	–	0.032	0.05932
5.19e+03	HIO	1	–	–	500	0.02894	0.02157	–	0.008437	0.0004979
7.31e+03	DM	1	-1	1	500	0.02271	–	0.0005787	0.002278	0.05455
7.31e+03	ER	–	–	–	500	0.0004232	0.0004226	–	0.01959	0.09105
7.31e+03	HIO	1	–	–	500	0.02319	0.01761	–	0.009033	0.000823
1.62e+04	DM	1	-1	1	500	0.01418	–	0.000194	0.0007644	0.02587
1.62e+04	ER	–	–	–	500	0.00204	0.00204	–	0.1386	0.1791
1.62e+04	HIO	1	–	–	500	0.01403	0.01123	–	0.009566	0.001558
2.29e+04	DM	1	-1	1	500	0.01202	–	0.000141	0.0004941	0.002187
2.29e+04	ER	–	–	–	500	0.001198	0.001197	–	0.06849	0.1789
2.29e+04	HIO	1	–	–	500	0.01158	0.009779	–	0.01087	0.0001473

Table 4.10: Photon number with a mixture of ER/HIO.

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
287	ER/HIO	1	–	–	300	0.006041	0.005879	–	0.01059	0.007408
396	ER/HIO	1	–	–	300	0.003773	0.003672	–	0.004173	0.01504
589	ER/HIO	1	–	–	300	0.001838	0.001794	–	0.00271	0.001053
803	ER/HIO	1	–	–	300	0.001154	0.001125	–	0.0009925	0.0004145
1.1e+03	ER/HIO	1	–	–	300	0.0006684	0.0006522	–	0.0003488	4.397e-06
1.7e+03	ER/HIO	1	–	–	300	0.000315	0.0003068	–	0.0003657	0.0005749
2.37e+03	ER/HIO	1	–	–	300	0.0001696	0.0001658	–	0.0002458	0.0006027
5.19e+03	ER/HIO	1	–	–	300	4.462e-05	4.337e-05	–	6.84e-05	0.0001615
7.31e+03	ER/HIO	1	–	–	300	3.34e-05	3.268e-05	–	0.0002363	0.0002566
1.62e+04	ER/HIO	1	–	–	300	5.994e-06	5.846e-06	–	5.477e-05	0.0003455
2.29e+04	ER/HIO	1	–	–	300	1.435e-05	1.427e-05	–	0.0003609	0.0006403

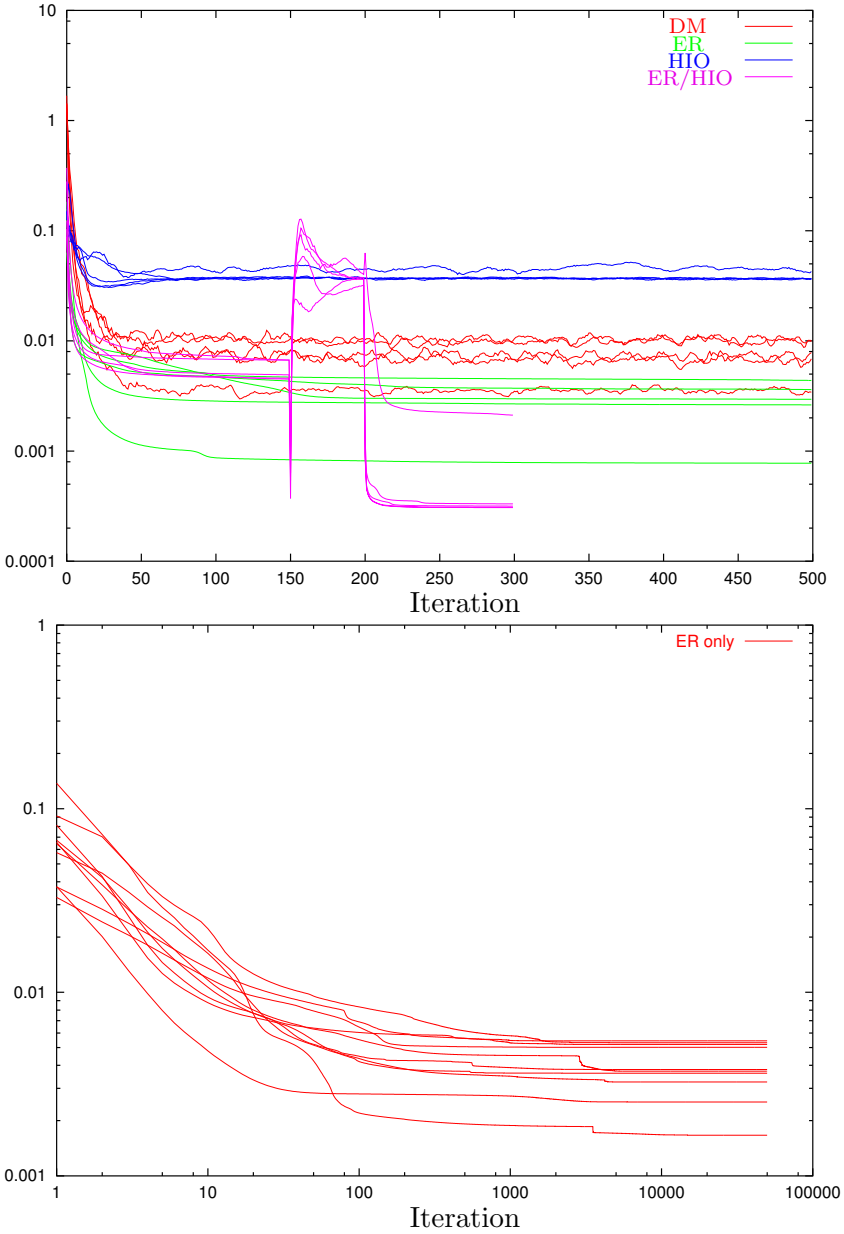


Figure 4.13: Error metric plots for ER, HIO, DM, and ER/HIO. Top: a comparison of ER, HIO, DM, and HIO/ER showing the efficiency of the combination of HIO and ER. Bottom: after 50,000 iterations of ER, the iterate begins to approach the χ^2 value of the combination of ER/HIO.

4.4.7 HIO parameters in ER/HIO combination

Since a combination of ER and HIO seems to have the effect of reducing both ξ_1^{orig} and χ^2, we ask what value of β is the best choice for HIO? Table 4.11 provides details of the result of fitting the $SNR = 1700$ CXD simulation with the values of β used in section 4.4.3. For this role of HIO, it appears that positive values are overwhelmingly more useful than negative ones, which tend to give results of about the same quality expected of fits using only ER at small negative values and grow steadily worse as β becomes more negative. The optimal values for β seem to be positive and either slightly less than or greater than 1.2, with $\beta = 0.9$ giving the best result(lowest ξ_1^{orig}). As might have been suspected, the optimal values of HIO used in combination with ER are different from the optimal values of HIO alone. In particular, while $\beta < 0$ were useful when using only HIO(See Table 4.4), these values perform the worst of those tested in a combination of ER and HIO. Since it is presumed that HIO assists ER by moving the iterate away from a local minima of the error metric, we conclude that in this case $\beta < 0$ moves the iterate in the wrong direction to escape the minima.

4.4.8 Summary of simulations

This section began with an investigation of the importance of two of the types of noise generally observed in CXD experiments: shot noise from alien scatterers and bias levels generated by the detector and its electronics. The algorithms examined here perform well in the presence of the former source of noise and it is not likely to be a concern in CXD imaging so long as care is taken to minimize this scattering experimentally. Bias added by the detection apparatus is a different matter. While noise from other scatterers does not introduce artifacts into the reconstructed complex amplitude, bias levels as low as 10% of the ADUs attributable to a single photon have been shown to cause vortices.

Each pair of these vortex artifacts in the reciprocal-space amplitude leads to a striped modulation of the corresponding real-space density. These stripes could then be mistaken for actual, physical properties of the sample– a mistake which is unlikely to be true of artifacts due to alien scatterers. Fortunately, a complete background subtraction will result in reduction of these effects, but will leave some bias due to the Gaussian distribution of the detector noise. This noise seems to lead to behavior of the algorithms similar to the case of shot noise from alien scatters and a solution may be arrived at, as shown by Table 4.3, as well as Tables 4.4-4.5 and their corresponding

Table 4.11: Photon number with a mixture of ER/HIO, varying β.

SNR	Algo	β	γ_1	γ_2	Iter	χ^2	χ^2_{Re}	ζ_{DI}	ξ_1^{orig}	ξ_1^2
1.7e+03	ER/HIO	0.2	–	–	300	0.0003216	0.0003134	–	0.001415	0.001708
1.7e+03	ER/HIO	0.5	–	–	300	0.0003207	0.0003127	–	0.0006432	0.0009106
1.7e+03	ER/HIO	0.7	–	–	300	0.0003134	0.0003058	–	0.0005517	0.0009241
1.7e+03	ER/HIO	0.9	–	–	300	0.0003172	0.0003091	–	0.0003623	0.00059
1.7e+03	ER/HIO	0.99	–	–	300	0.0003125	0.0003043	–	0.0005485	0.0004762
1.7e+03	ER/HIO	1.01	–	–	300	0.0003136	0.0003058	–	0.001031	0.0004414
1.7e+03	ER/HIO	1.2	–	–	300	0.0003102	0.0003017	–	0.001347	0.001246
1.7e+03	ER/HIO	1.5	–	–	300	0.0003136	0.000306	–	0.0006966	5.392e-05
1.7e+03	ER/HIO	1.7	–	–	300	0.0003136	0.0003059	–	0.0006911	0.003766
1.7e+03	ER/HIO	-0.2	–	–	300	0.002189	0.002181	–	0.05228	0.05763
1.7e+03	ER/HIO	-0.7	–	–	300	0.002022	0.002016	–	0.09138	0.06912
1.7e+03	ER/HIO	-0.9	–	–	300	0.002299	0.002292	–	0.1564	0.2022
1.7e+03	ER/HIO	-1.2	–	–	300	0.002752	0.002746	–	0.1385	0.1991
1.7e+03	ER/HIO	-1.5	–	–	300	0.001744	0.001737	–	0.06915	0.1772

images in Appendix A.

From the investigation of a background subtracted simulated CXD pattern–Tables 4.4-4.5–it is determined that smaller values of β are generally more effective than larger values, perhaps because large values move the iterate too quickly toward the perceived intersection of the sets, in analogy to Fig 2.14. Further, in these simulations, some combinations of β and γ_i's which do not lead to convergence of DM nevertheless lead to good solutions of the inverse problem. It is difficult to predict which parameter sets in DM will be useful, as these will almost certainly vary from data set to data set since the shape of the corresponding modulus constraint can change markedly. Therefore, it appears that, in the absence of a more complete cataloging of the behavior, the parameters of DM will have to be explored with each new CXD pattern.

In general, increasing photon number in the absence of other noise, leads to a reduction in ξ_1^{orig}, and hence to a more correct fit. This is an expected result, but the fact that reasonably good agreement is achievable with $\sim 10^5$ photons is particularly pleasing, because this is a realistically achievable intensity measurement for the CXD experiment of Chapter 3.

Further, we have seen that a combination of ER and HIO is generally more effective in finding the correct solution in fewer iterations compared to ER or HIO alone. This is not entirely unexpected given the investigation of an iterate's trajectory in the N-dimensional Hilbert space in Chapter 2, where it was shown that in some cases neither DM or ER alone could find the correct solution. Fig. 4.14 displays the result of fitting the simulated CXD pattern generated with 5×10^6 photons used in Section 4.4.5. Fig. 4.14a used DM, b used ER, c used HIO, and d was the combination of ER and HIO in Section 4.4.6. The fidelity to the truth image for the combination of algorithms, given by $\xi_1^{orig} = 4 \times 10^{-4}$, is essentially the same as the best possible value $\xi_{trunc}^{orig} = 3 \times 10^{-4}$ achievable. The second closest fidelity was an order of magnitude worse at $\xi_1^{orig} = 6 \times 10^{-3}$ given by the fit with HIO only.

Another interesting result that can be gleaned from an examination of the images in Appendix A is that many of the results of fitting with DM variants result in real-space images that appear to be the result of a mixture of the image and its twin. This is an interesting result, since HIO was originally devised by Fienup to avoid such stagnation. That even DM and HIO can be fooled into finding local rather than global minima is an indication of the complexity of the Fourier modulus constraint, which is likely increased by the added noise. A few examples are shown in Fig. 4.15. The

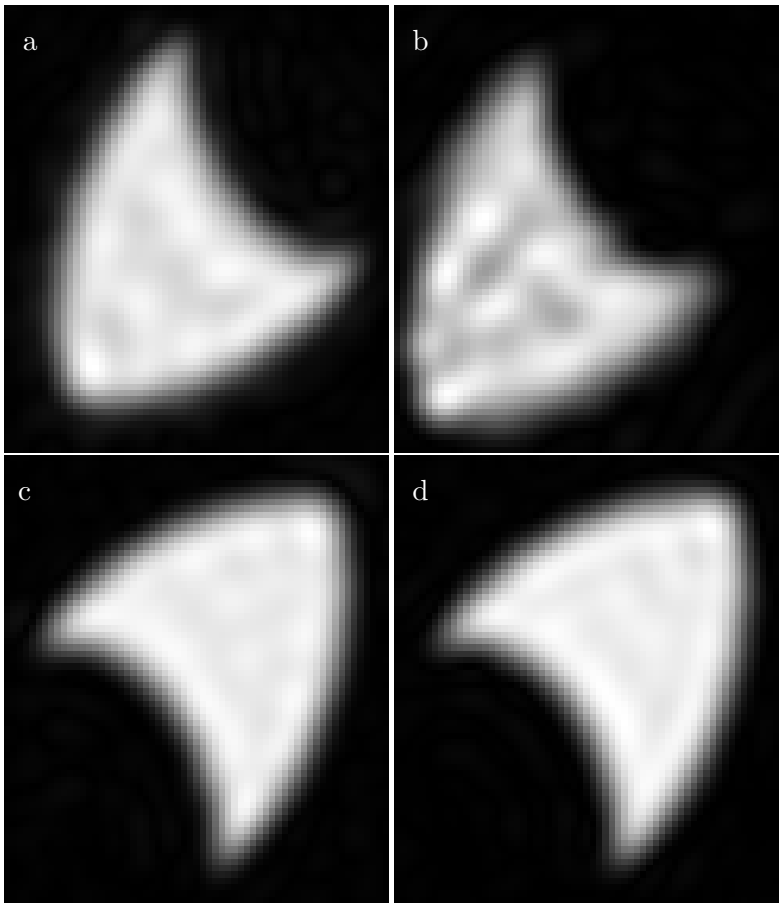


Figure 4.14: Comparison of best results obtained by fitting the 5×10^6 photon image in 4.4.5 with DM(a), ER(b), HIO(c), and ER/HIO(d). The ER/HIO is the best result as indicated by Fig. 4.13, which shows the plot of the error metric with iteration, and the values of ξ_1^{orig} in Tables 4.8 and 4.10.

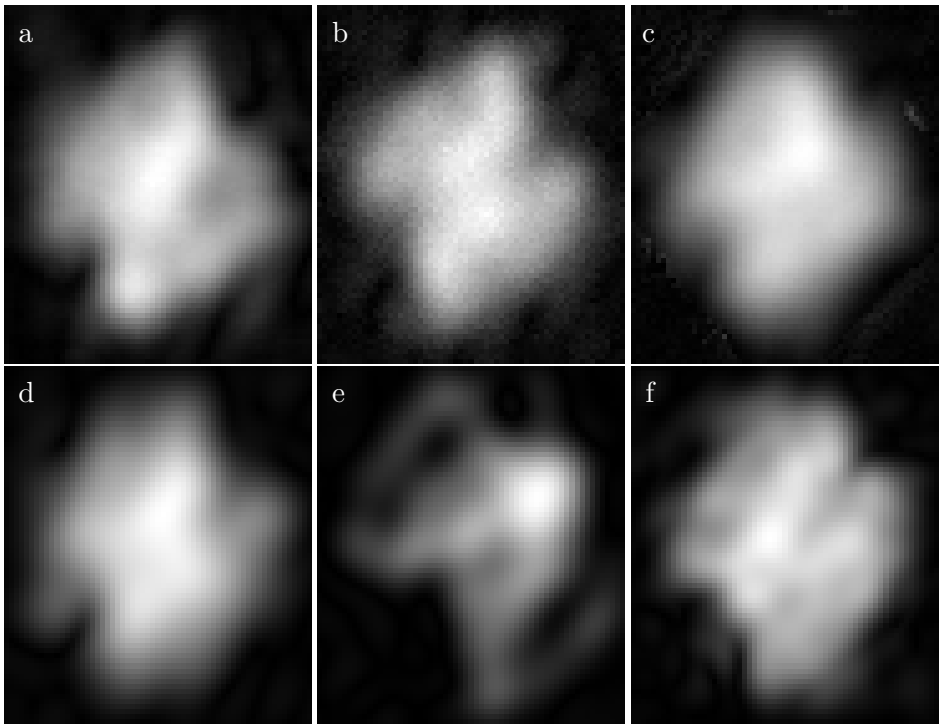


Figure 4.15: Twins can occur during fitting with all algorithms when noisy simulated data is used for the Fourier modulus constraint. As described in the text, the top row is from background subtracted simulated patterns fit with DM variants of various parameters from Sections 4.4.3 and 4.4.4. The bottom row is a fit from simulated patterns with only photon counting noise added, left to right: HIO with 9×10^4 photons, ER with 2×10^5 photons, and DM with 2×10^6 photons

top three images resulted from DM variant fitting of the simulated CXD pattern generated by assigning 9×10^4 photons, adding a background level with Gaussian variance, and subtracting the mean of the distribution from each pixel–the results from Sections 4.4.3 and 4.4.4. Fig. 4.15a is DM with $\beta = 1.2$, $\gamma_1 = -1.66$, and $\gamma_2 = 0.8$, which are neither HIO nor maximally contractive parameters, but do satisfy the requirements for convergence of the DM. Fig. 4.15b used DM with $\beta = 0.7$, $\gamma_1 = -1.43$, and $\gamma_2 = 1.43$; the maximally contractive parameters. Fig. 4.15c used HIO with $\beta = 1.2$, which is not considered to be an outlandish value. The bottom row consists of fits conducted in Section 4.4.5 concerning photon number: d is HIO with 9×10^4 photons, e is ER with 2×10^5 photons, and f is DM with 2×10^6 photons; demonstrating that all algorithms are vulnerable. By contrast, when a combination of ER and HIO is used, e.g., the results of Sections 4.4.6 and 4.4.7, this behavior manifests only when $\beta < 0$ in HIO.

Reconstruction of experimental data is described in detail in Chapter 3, but for the sake of comparison it must be said that CXD data is generally reconstructed by applying cycles of ER and HIO as in sections 4.4.6 and 4.4.7, so we examine the results of simulations using this method to answer the question of how much signal is needed to succeed in a reconstruction of this sort. For this simulated object, the $SNR = 1100$ case, which was generated with 2×10^6 photons, gives $\xi_{trunc}^{orig} = 7 \times 10^{-4}$ which is an RMS error per pixel of about 2.5%. A typical CXD experiment, like the one discussed in Chapter 3, can collect approximately this many photons in 120 s, which is an encouraging result. In this particular case, the experiment involves a strong scatterer, a Au crystal, of approximately 25 μm^3. However, the error in the case of $SNR = 287$, with 9×10^4 photons, gives $\xi_1^{orig} \sim 0.01$ or $\sim 10\%$ per pixel RMS uncertainty. The latter is still acceptable, especially in cases where only the gross shape of the sample is desired. The important revelation that χ^2 and ζ_{DI} do not necessarily correlate with ξ_1^{orig} is slightly unnerving.

Fortunately,the combination of χ^2 or ζ_{DI} with ξ_1^2 is shown to provide a reliable measure of the fidelity of the best fits to the truth image. Fig 4.16 contains scatter plots of ξ_1^{orig} versus ξ_1^2. These plots contain all of the trials described in this chapter. Of particular interest is the lower plot which shows that all runs where ξ_1^2 was less than 0.01 resulted in a best fit with fidelity to the truth image $\xi_1^{orig} < 0.025$, or about 16% error per pixel. To verify this, it is important to test other objects, but we nevertheless expect a combination of the error metric and the irreproducibility, ξ_1^2, to give a strong indication whether or not a solution has been found.

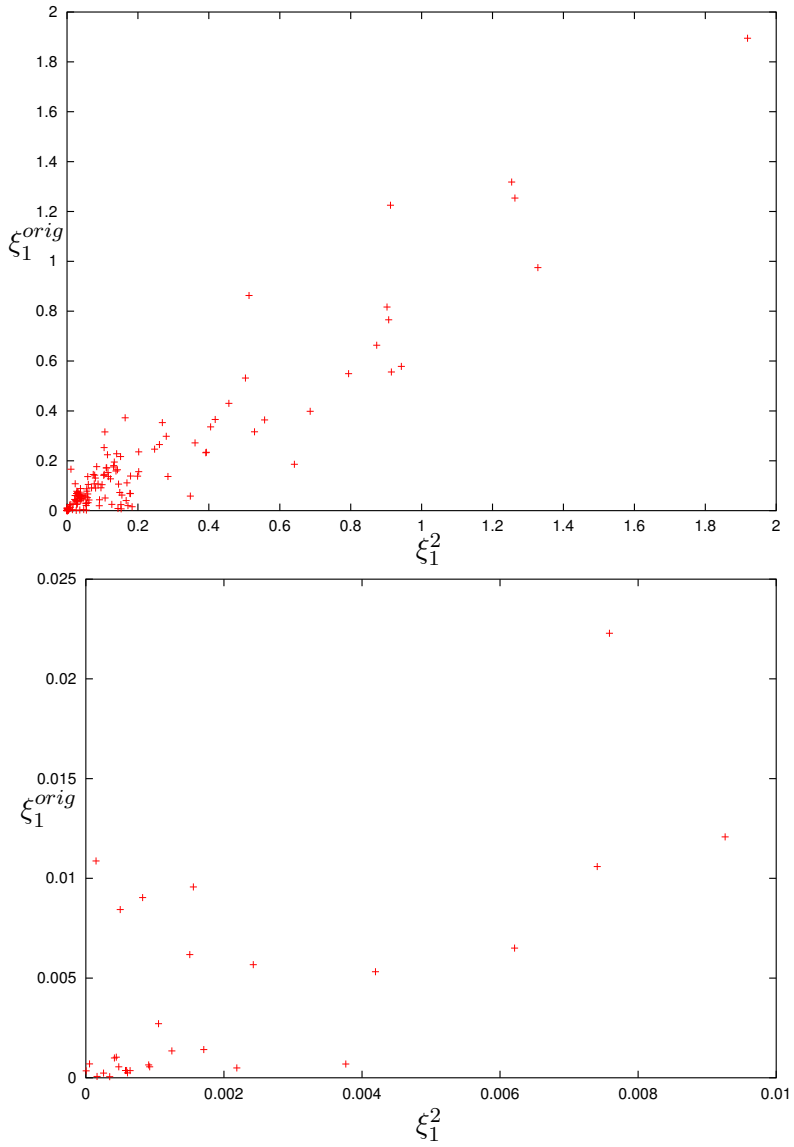


Figure 4.16: Scatter plots of ξ_1^{orig} versus ξ_1^2. There is a roughly linear relationship between the two figures of merit. In this simulation, all fits resulting in a value of $\xi_1^2 \leq 0.01$ gave $\xi_1^{orig} \leq 0.025$, or about 16% RMS error per pixel compared to the truth object.

Lastly, it must be reiterated that since the shape of the modulus constraint changes strongly from CXD pattern to CXD pattern it is expected that results from the simulations presented here will not hold across all datasets and experimental conditions. In fact, that would be far beyond the scope of the trials presented here, where the goal was to determine heuristic guidelines for experimental design and the behavior of these iterative algorithms.

4.5 Dealing with Vortices

As mentioned previously, these phase singularities have been discussed by Fienup and Wackerman[36]. In that paper, several methods of overcoming the problem of "striped images" is put forward. The stripes occur in the real-space iterate and arise from the interference of two phase singularities in reciprocal-space. These vortices occur in pairs located centrosymmetrically in the image with opposite phase wraps, *i.e.*, if the phase changes by $2\pi n$ in a clockwise loop about one vortex in the pair, the phase will change by $-2\pi n$ around the other. It is important to realize the complex, global nature of these singularities. For example, it is not feasible to remove the vortices one at a time, since that would leave an overall non-zero phase wrap in the phase and the removed vortex would reconstruct on subsequent iterations. Instead, they must be removed in pairs of opposite chirality. The following two methods due to Fienup and Wackerman attempt to remove all singularities simultaneously.

First, the "voting" algorithm utilizes three possible solutions arrived at via iterative fitting starting with three different random phase sets. The three real-space objects must be centered on the same point in the array. This is most easily accomplished in reciprocal-space, where removing all linear phase terms will center the real-space object to within a fraction of a pixel in the array. Next, the phases in reciprocal-space must be aligned. Since an overall phase offset–a zero order phase term–has no effect on the real-space object's shape, it is common to have such an offset between two fits. For the most part, this is easily accomplished, with the caveat that one must be certain that the two reciprocal-space patterns correspond to the same twin. If they do not, it is necessary to conjugate one before aligning the phases. Once this has been done to all three images, the complex amplitude at every point in the array is replaced by some average of the complex amplitude in the two arrays which have the closest agreement. In other words, the amplitude at each pixel is voted on with the final result being

a compromise between the two parties who agree most closely. The exact value chosen might be a simple average or even the measured magnitude of the amplitude with the average phase from the two winning reconstructions. Typically, the resulting possible solution would then be used as the starting point for a new fit.

Second, the "patching" algorithm begins just as the voting algorithm does, by ensuring that the phases of the reciprocal-space image are properly aligned for a point-by-point comparison. Only two inputs are required for patching, but rather than a direct comparison, further manipulation is called for. To isolate the stripes in real-space, all amplitude inside the support is set to zero. It is, of course, important to use a smooth apodization to avoid problems in reciprocal-space. This image is then transformed to reciprocal-space where it is smoothed and thresholded to create a mask for each of the two images. If the masks intersect, one must either re-threshold the image or try again with a different set of starting phases. The corrected reciprocal-space is then generated by using the complex amplitude of the first fit everywhere except inside its masked region, where the amplitude of the second is used instead. Again, this data is generally used as the input for a new fit.

Both of these methods utilize the property that the singularities typically do not reconstruct in the same pixel in iterates with different starting phases. While this has generally been our experience (Had it not, one might suspect that the vortices were actual singularities in the wavefield!), it should be noted that with very low SNR data, such as that collected from 0.2 μm Pb crystals, there is a greater likelihood of artificial singularities occurring in low intensity regions with rapidly varying amplitude of the data, e.g., on the edges of the central maxima, than in other regions.

Neither of the above methods have worked very well with our CXD data. Instead, we use a sort of hybrid patching approach. Essentially, the vortices were located in reciprocal-space by either calculating the gradient of the amplitude and seeking large negative spikes or finding low amplitude points and summing the phase around a loop surrounding those points. For the former method, a threshold was set as some fraction of the highest magnitude spike and all points within that thresholded set of values was identified as a vortex. This is a flawed approach, as it can both miss vortices and detect vortices where none exists. Nevertheless, this method is generally reliable. In the latter case, a vortex is identified by a $2\pi n$ result, but due to the discrete grid it is still possible to misidentify a point as a vortex. In any event, once a vortex has been identified, all pixels within a user-defined radius are altered

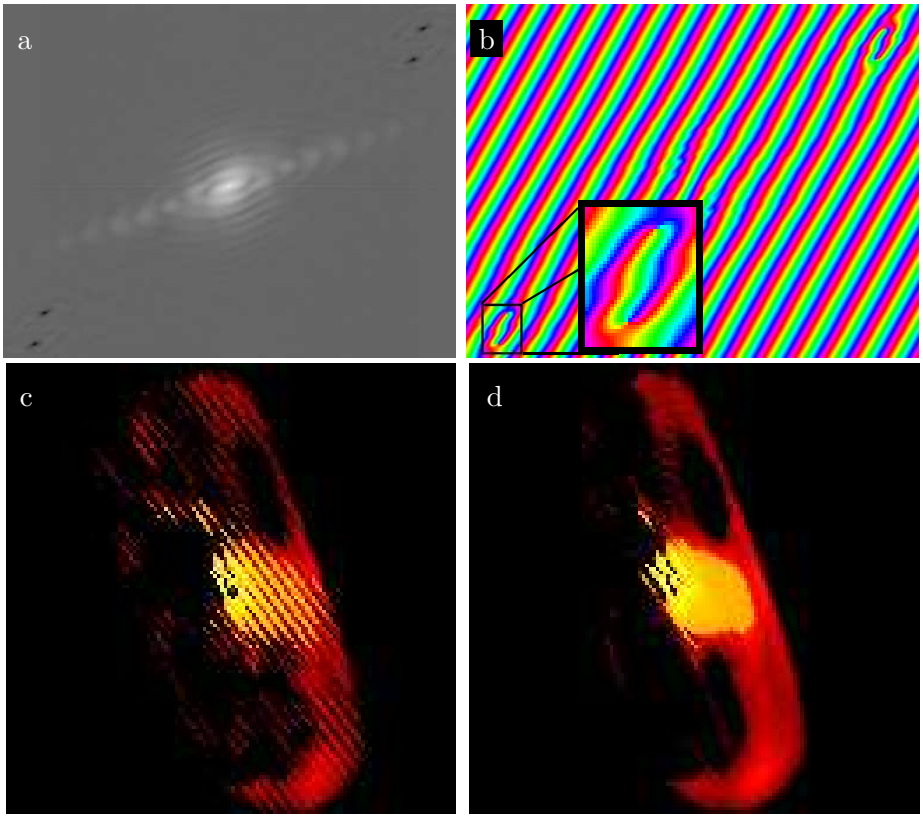


Figure 4.17: Reconstructed images from Au CXD data, showing the presence and effect of vortices on the real-space image. a and b are the reciprocal-space complex amplitudes that have reconstructed with vortices, as can be determined by the areas of very low amplitude and rapid phase change(See the inset in b.). c is the real-space density projection corresponding to the complex amplitude, showing the "stripping" indicative of this artifact. d is the real-space image immediately after the vortices have been removed by the method discussed in the text.

by setting the magnitude of their amplitudes to the experimental value and setting the phase at that pixel to a random number in $[-\pi, \pi]$.

Unlike the earlier methods, this requires only a single input, facilitating the batch processing of data and allowing the procedure to be applied any number of times. On the other hand, a vortex will only be removed via this method if another of opposite chirality is located within the radius of the patch. If this condition is met, they may annihilate, but this result is not certain.

Fig. 4.17 shows the effect of vortices on a the fit to a 2D slice from the 3D CXD data of Chapter 3. The top images show the complex amplitude that was reconstructed. The phase is shown as described in Chapter 3, with cyan corresponding to zero and red to $-\pi$ and π. Each vortex is characterized by a region of small amplitude around which the phase varies rapidly by an integer multiple of 2π. In this case, the two vortices, shown expanded in the inset of Fig. 4.17b, have opposite chirality and so can be removed by the above method. Since these artifacts appear symmetrically, the other pair is also removable. Fig. 4.17c shows the characteristic stripes described by Fienup and Wackerman[36]. The hybrid patching algorithm is applied once and after one iteration of ER the stripes have disappeared, as can be seen in Fig. 4.17d. This method is explicitly a 2D method. In 3D, the vortices manifest as loops rather than points, meaning that this method would require the randomization of all points between the two sides of the loop, which is far more disruptive of the reconstructed phase. Fortunately, in 3D persistent vortices occur much more rarely than in 2D, so the problem is not as serious as it could be, but a good background subtraction is clearly important to reduce the appearance of vortices.

4.6 Summary

The intent of this chapter was to examine the differences between straight forward simulations, usually generated by applying a FFT to some crystal shape, and the fitting of experimental data, which tends to be more difficult. Our initial object was chosen to be a single compact, smooth-edged, asymmetric image without internal contrast. With the exceptions of the asymmetric shape, chosen to reveal stagnation with the twin image, and the Gaussian-smoothed edges, this object is similar to the single compact Au crystal we measured in Chapter 3. With other samples and objects these methods may behave very differently, so caution must be taken when attempting to generalize this result to other objects.

Adding noise to the FFT of a simulated crystal similar to those measured resulted in a pattern very similar to the measured CXD data. Indeed, for the first time, we found vortices in our reconstructions of a simulated pattern; something that we had long seen in data, but been unable to reproduce in simulations. Another similarity to reconstructing CXD data is the utility of a mixture of HIO and ER cycles to quickly find a solution. Here, only a single iteration of HIO was required, while with CXD data application of HIO must sometimes be performed three or four times to achieve this result. We can not explain this discrepancy here.

An important difference is that the patching discussed in Section 4.5 failed for simulated patterns. From previous experience, it has been observed that data with apparent low SNR data, *i.e.*, weakly scattering samples or situations with high levels of scattering from alien particles, that patching is much more difficult. It is, therefore, possible that other simulations may exhibit tolerance to patching.

It is also important to explicitly state what we have not attempted to model. As discussed in Chapter 1, creating a complex, rather than real, density allows the simulation of CXD from a strained crystal. One could also simulate a pattern corresponding to a measurement that has missed the exact Bragg point, resulting in an asymmetric pattern. These are both interesting problems not addressed here. Further, the shot noise from alien scatterers was explicitly assumed to occur with uniform probability across the detector. One might instead ask what effect a spatially non-uniform shot noise probability might have. This amounts to the simulation of a "contaminated" CXD pattern, where some intensity from a different crystal is diffracting into the region of reciprocal-space being measured.

Nevertheless, for the simulation described in this chapter, we may draw some conclusions. First, without a very carefully chosen support, DM and its variants alone generally provide some improvements to ER with regard to the fit to the truth object. These algorithms are also far more noise immune than ER, so if a support is carefully chosen, they may substantially outperform ER. Interestingly, ER tends to preferentially fit the reciprocal-space pattern as demonstrated by very low χ^2 values of its fits. This is, naturally, not extremely beneficial when the pattern contains substantial uncertainties. However, mixing HIO and ER is enormously effective, with the combination providing a far better fit than either algorithm alone. One possibility for improving these algorithms is by loosening the Fourier modulus constraint to allow for the inherent photon shot noise. It is not clear what effect this would have on the convergence of the algorithm, but it is worth trying.

The noise immunity of DM variants makes its combination with ER a very potent method for finding a solution quickly, this agrees with experience from fitting experimental data. Similarly, the fits to simulated patterns containing comparable numbers of photons give reproducible results. No noticeable ambiguity in the uniqueness of a solution was introduced by the addition of noise. Further, simulated background subtracted images reconstructed well. These results from our simulations give no cause to doubt the reconstructions from experimental data, and, in fact, reinforce the resilience of the method.

Chapter 5

Conclusion and Outlook

Chapter 1 set forth an inverse problem whose solution would allow the 3D reconstruction of a diffracting density. It was shown that the scattered intensity in a carefully controlled coherent X-ray diffraction experiment is related to this density, albeit without any information concerning the phase of the diffracted wave. The recovery of that phase would then amount to the recovery of the real-space diffracting density, since in the kinematical approximation the two are related by a Fourier transform. Since X-ray wavelengths are comparable to atomic spacings in materials, this phase retrieval could be used in 3D X-ray microscope. Applications for such an instrument range the gamut from basic materials research to studying electromigration in semiconductor devices to imaging biological materials. Since this recovery is actually the solution of the phase retrieval inverse problem, a solution can not generally be found analytically. This inspires the use of iterative methods used in other fields to recover the lost phases and thereby make possible a kind of lensless X-ray microscopy.

In Chapter 2 the iterative algorithms that can be used to phase the CXD data were described. These include the original GS algorithm, its generalization by Fienup, also known as ER, Fienup's HIO–and its generalization by Millane–and the DM developed by Elser, of which Fienup's HIO is a special case. Through the use of projector notation used by others[23, 12, 28] a few simple examples where used to demonstrate the difference between ER and DM variants, as well as the effect of the DM's parameters on its behavior. Since the real-space constraints are typically convex, while the reciprocal-space Fourier modulus constraint is decidedly non-convex, no general proof of convergence is known. Using the projector notation, several simple illustrations were utilized to highlight the behavior of these algorithms in the presence of convex and non-convex constraint sets. A particularly interesting example comprised of a convex set and a non-convex set that did not

intersect, vividly shows the utility of non-converging parameters in the DM and also that combinations of DM variants and ER can be amazingly useful.

From the foundations laid by these chapters, we could move on to describe the successful reconstruction of a Au crystal in Chapter 3. By measuring the distance from the Bragg point to the farthest detectable fringe, we confirm that we can see objects with a period of 100 nm and thus are sensitive to structure as small as 50 nm, in the plane, while the resolution in the third direction is somewhat worse. This chapter also contained detailed information concerning the sample preparation and the equipment used to conduct this experiment. The reconstructed real-space 3D density, whose shape was within the bounds of expected sample dimension, displayed marked internal density contrast. A region of very high effective density was reconstructed near the center of the crystal. This was shown to result from the partial coherence of the incident beam, to good agreement with theoretical calculations. Additional contrast may be explained by misoriented regions within the crystal. Such regions diffract into different–unmeasured–Bragg spots and therefore are expected to appear "dark" in the reconstructed density. In particular, we reconstruct stripes perpendicular to the (111) direction that may be indicative of twinning within the crystal. Nevertheless, one is left to wonder what effect noise in the CXD measurement has on the reconstructed density.

This last question was addressed through the use of a simulated CXD pattern in Chapter 4. Here, a "crystal" was created and its "diffraction pattern" calculated by means of a FFT. The behavior of the algorithms was then explored in the presence of the inherent photon counting noise, as well as addition shot noise, as from alien scatterers, and bias, as might result from a poor background subtraction of the data. As a result of adding bias to the image, vortex-like artifacts were present in the reconstructed reciprocal-space amplitude. This was judged an important step, since up to this point these vortices had only appeared in the phasing of experimental data. Similarly, simulations mimicking the experimental procedure of background subtraction and fitting with alternating cycles of ER and HIO were seen to result in far better reconstructions than any of the algorithms alone. The final result of this chapter was that there was no apparent effect upon the uniqueness of the reconstructed density, even in very high noise simulations. Further, simulations with noise levels similar to those in the experiment of Chapter 3 displayed no artifacts likely to be misconstrued as physical phenomena, as those artifacts that do appear are not reproducible when many fits are performed.

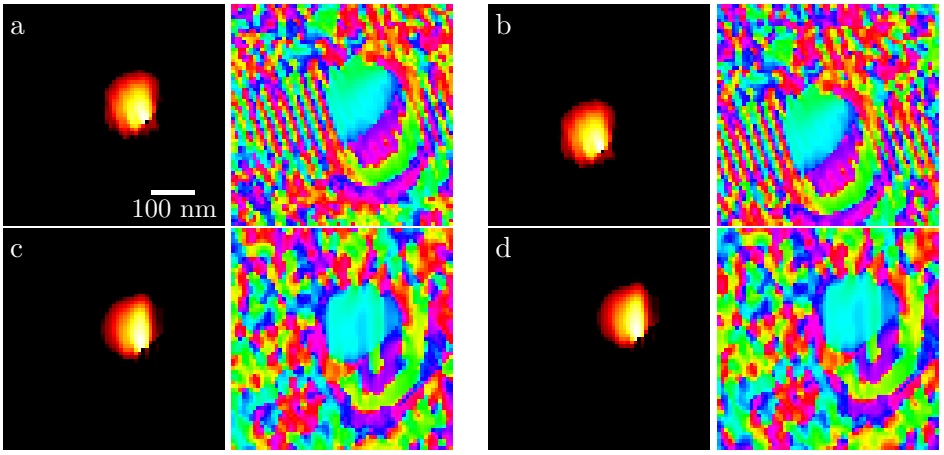


Figure 5.1: 3D density projections recovered from Pb crystals at ~ 575 K in a UHV chamber. a and b are the two best fits from the first measurement. c and d are the two best fits to a measurement made on the same crystal in the same experimental geometry 10 min after the start of the first data collection. Despite the instability of the sample, a very reproducible shape is reconstructed from each measurement. The density, left, and its phase, right, is shown for each reconstruction.

The remainder of this chapter will be used to discuss possible future directions for this technique. First, a short section on a slightly different experiment on a Pb film treated similarly to the Au film of Chapter 3 will be presented to demonstrate that the internal structure of those results is not inherent in the method. Finally, a few words on how the technique may be improved are presented.

5.1 2D CXD Imaging of Pb

As a follow up to the Au experiment in Chapter 3, an ultra high vacuum (UHV) experiment on Pb was conducted at Sector 33 of the APS. In this case, a Pb film was grown *in vacuo* on a Si wafer with its native oxide intact, in analogy to the earlier experiment. However, here the film was completely melted, resulting in a collection of small crystals with random orientation, rather than the (111) texture of Au film derived crystals.

The 2D reconstructions of CXD patterns taken from this sample are shown in Fig. 5.1. In this case, the diffracted intensity is much smaller and vortex formation is a very real problem. Multiple applications of the vortex removal algorithm discussed in Section 4.5 were applied to remove these artifacts. These data were collected while the sample was approximately

25° C below the melting point of Pb, 600 K. Stability issues forbade the collection of a 3D series in this experiment. Fig. 5.1a and b are the two best fits to the first data set collected at this position. The irreproducibility is $\xi_1^2 = 0.00021$ with error metrics $\chi^2 = 0.144287$ and $\chi^2 = 0.144289$. Fig. 5.1c and d are the reconstructions from data taken approximately 10 min later and have irreproducibility $\xi_1^2 = 0.00016$ and error metrics $\chi^2 = 0.09585$ and $\chi^2 = 0.09590$. The error metrics indicate a rather poor fit to the measured intensities, likely due to the relatively short exposures and difficulty in obtaining the measurement. Nevertheless, the reproducibility of the fit is quite striking and a dramatic change in the shape of the projection of the Pb crystal is evident.

In this case, a much smaller crystal has been reconstructed. There appears to be some contribution from the partial coherence in the form of high apparent density near the center of the crystal, but it is less dramatic than in the case of Au. The reconstruction of a smooth density, and a relatively flat phase in real-space, indicates that the internal density modulations in the Au reconstructions were most likely due to strain or imperfection in the crystal rather than any inherent flaw in the methodology. Most importantly, this result demonstrates that the method can be effective in recovering the density map of small crystals measured in interesting environments–in this case, under UHV and near melting.

5.2 Future Directions

Although the methods presented here have been largely successful, it is important to carefully consider the scope of the technique and reasonable ways to enhance it. The experiments presented here make use of a very small crystal, which in effect acts as the final slit in the path of the incident beam, at least in terms of describing the coherence. For other samples, it will be necessary to control the number of coherence volumes of the incident beam incident on the sample, since in this case the intensity from each coherently illuminated region will add, rather than the addition of amplitudes that results from illumination with a single coherence volume. This can likely be accomplished by the use of a slit very close to the sample, at the cost of requiring that the illumination function of the slit be modeled and used as part of the real-space constraint. Of course, the true virtue of this method is in imaging objects on a very small scale and so the effect of focusing elements on the reconstruction should also be studied.

In addition to extending the technique to different kinds of samples, we

can also address changes in the iterative reconstruction algorithms. Building on the simulations of Chapter 4, a great deal of work can be done in further characterizing the behavior of the DM parameters. In fact, a much more ambitious project would be to attempt to develop some sort of classification of the shape of Fourier modulus constraints for various objects. This would be very useful, since such a scheme might greatly reduce the space of useful DM parameters for a given problem. Additionally, DM variants are strongly dependent on the real-space support constraint, a property that has not been explored in this thesis. It might prove very useful to discover exactly how tight the support should be, and if this varies, for example, from smooth objects to sharp edged ones.

Finally, it may prove useful to account for additional physical phenomena present in the measurement. A method for deconvolving the partial coherence contribution to the reconstructed density would be a very useful step forward. Failing that, characterization of the partial coherence function of the beamline might assist in ameliorating its contribution. In the particular experiment in Chapter 3, it is also possible that the kinematical approximation is less than perfectly valid. Therefore, if the dynamical scattering from such a sample were calculated, it might allow us to express more confidence in the result presented here.

Appendix A

Reconstructions from Noise-Added Simulation

This appendix contains images generated from the simulations of Chapter 4. The figures are one of two types: a collection of the original and starting images or a summary of the best fit of five–each of which started with a different set of random phases–with for a specific noise level and/or parameter choice. In the latter case, the images are the logarithm of square of the reciprocal-space amplitude, its phase, and the resultant real-space density are shown in that order from left to right. Further, the three important calculated quantities–χ^2, ξ_1^2, and ξ_1^{orig}–are provided to the right of each set of images. These are the best fit of five, each of which started with a different set of random phases.

A.1 Alien Scattering Added Simulations

Images corresponding to fits performed in Section 4.4.1. Fig. A.1 displays the original alien scattering added simulated CXD patterns and the truth image, or target of the fit.

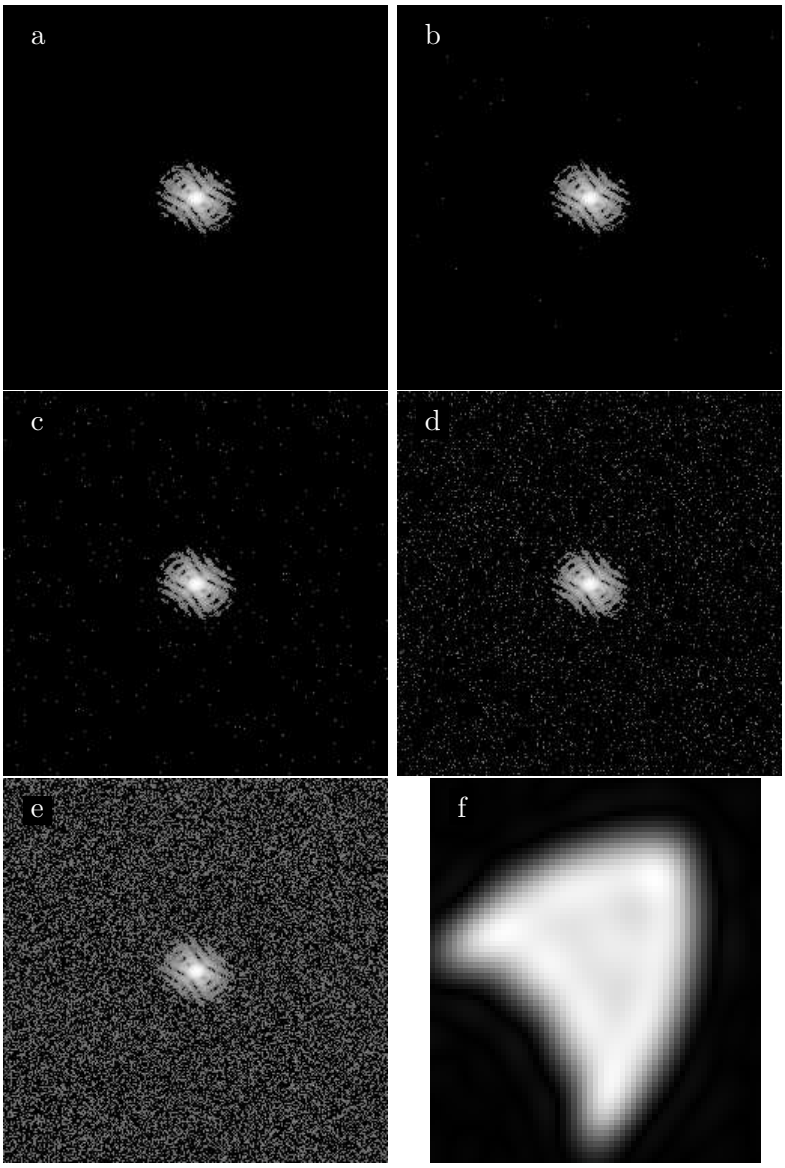


Figure A.1: Original simulated CXD patterns for the alien scattering simulation of Section 4.4.1. The noise is specified by the mean of the Poisson random distribution from which it is drawn. The images here have mean: 0 in a, 0.0005 in b, 0.005 in c, 0.05 in d, 0.5 in e. The images are plotted as the logarithm of the intensity. The truth image for all simulations, excluding those varying photon number, is shown in f. The viewable area in f is the size of the real-space support constraint.

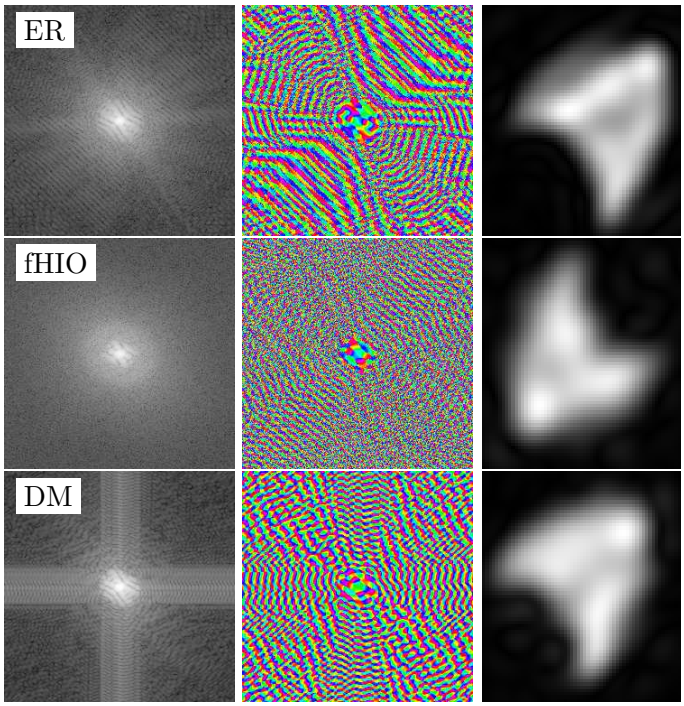


Figure A.2: Poisson mean: 0.

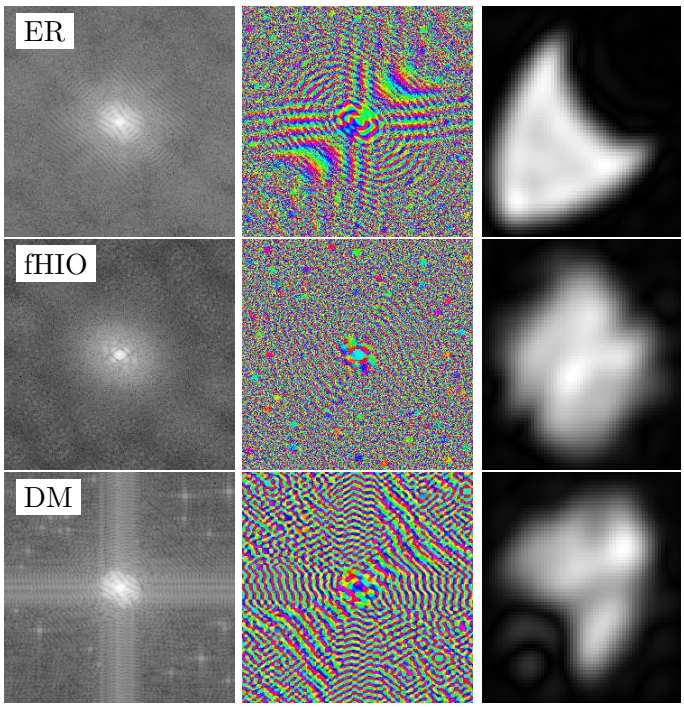


Figure A.3: Poisson mean: 0.0005.

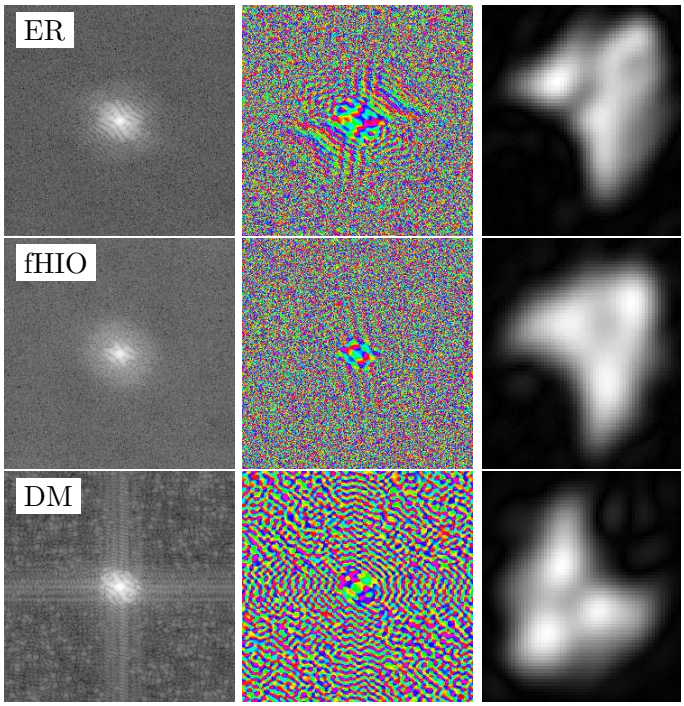


Figure A.4: Poisson mean: 0.005.

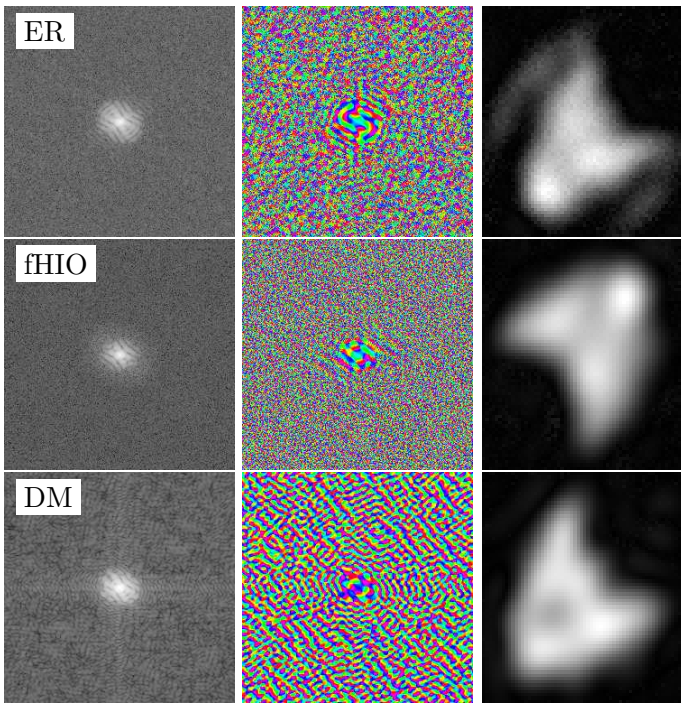


Figure A.5: Poisson mean: 0.05.

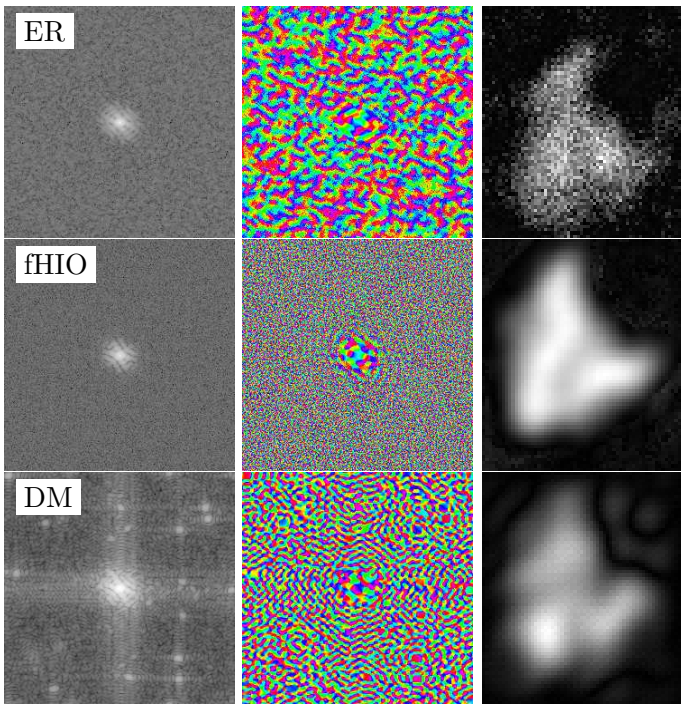


Figure A.6: Poisson mean: 0.5.

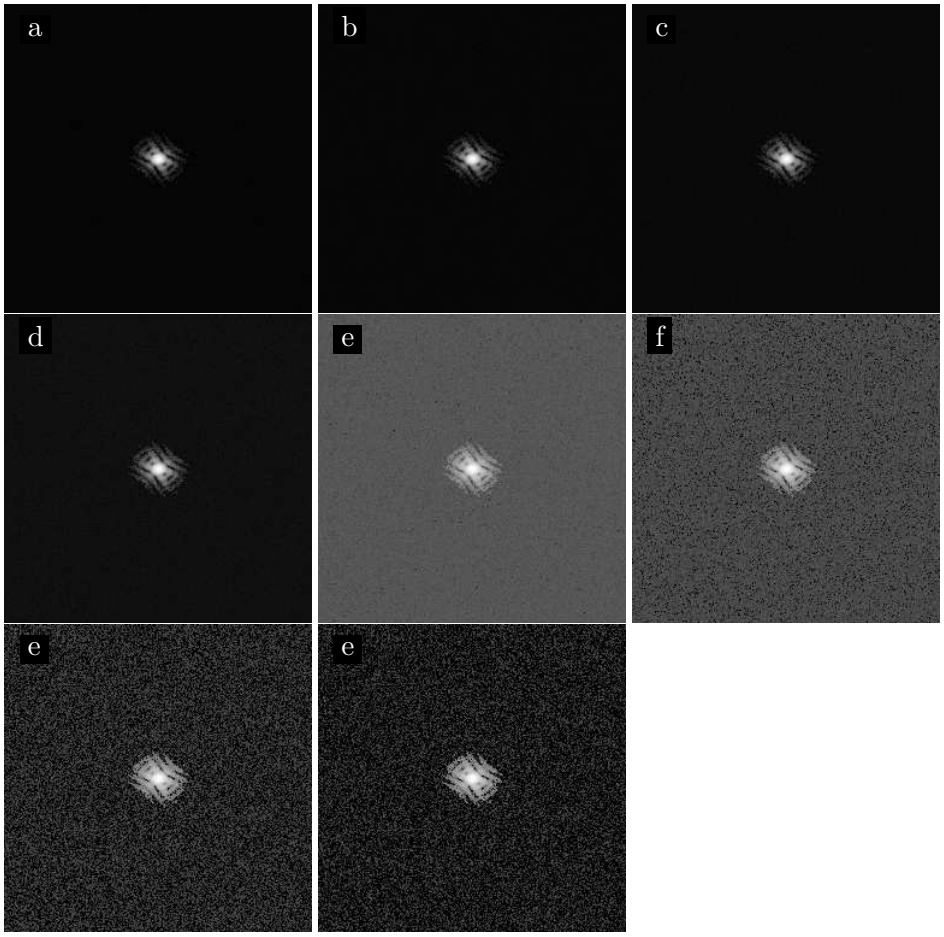


Figure A.7: Original simulated CXD patterns for the bias added simulation of Section 4.4.2. The noise is specified by the mean of the bias, in analog to digital units (ADUs), added to the pixels, after the subtraction is performed. The images here have "noise": 900 in a, 700 in b, 500 in c, 300 in d, 100 in e, 50 in f, 10 in g, 0 in h. The images plotted are the logarithm of the intensity.

A.2 Bias Added Simulations

Images corresponding to fits performed in Section 4.4.2. Fig. A.7 displays the original bias added simulated CXD patterns. The bias level in each pixel is determined by drawing a number from a Gaussian random distribution with variance obtained from a histogram of an experimental CXD pattern. Lower noise images are obtained by removing a constant fraction of the original mean of the distribution from each pixel, mimicking a background subtraction in experimental data. The cases are labeled by what the Gaussian mean would be, were it not cut off at 0 ADUs. Again, this is done in analogy to experimental background subtraction where values less than

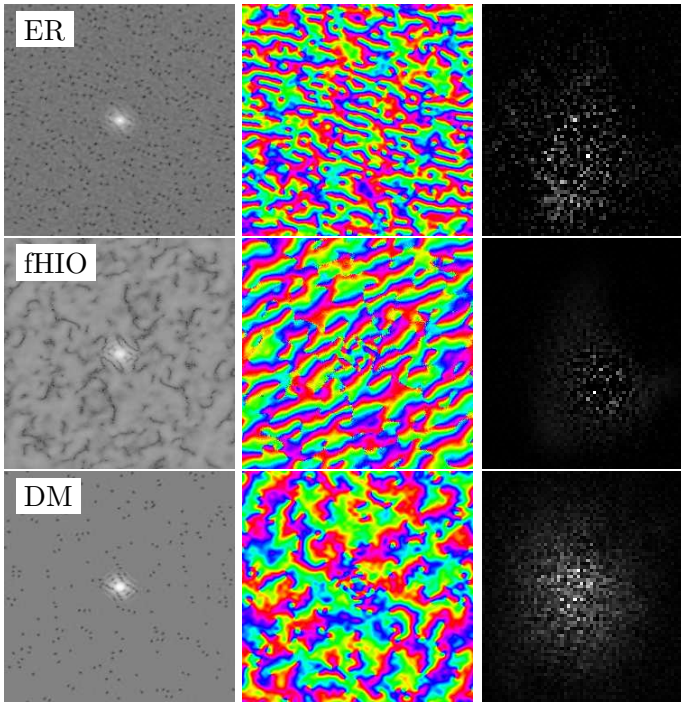


Figure A.8: Bias mean 900 ADUs.

0 are set to 0.

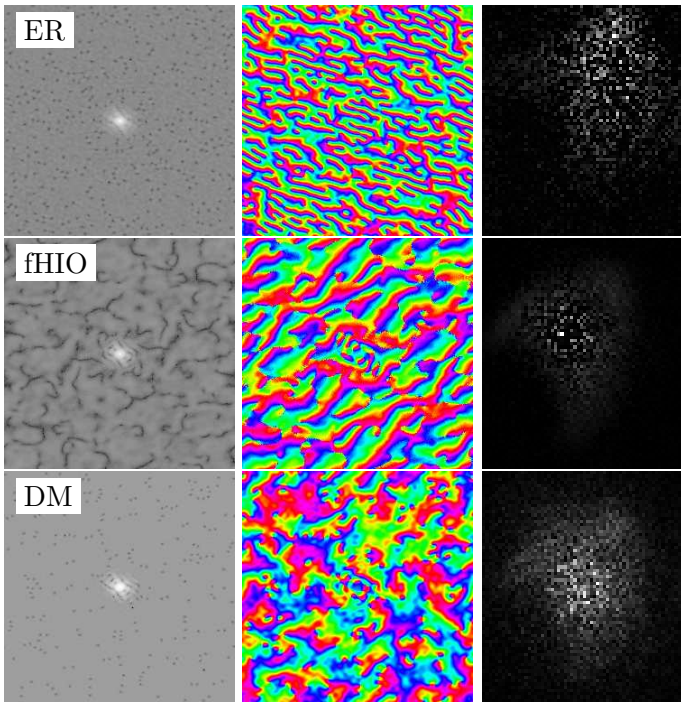


Figure A.9: Bias mean 700 ADUs.

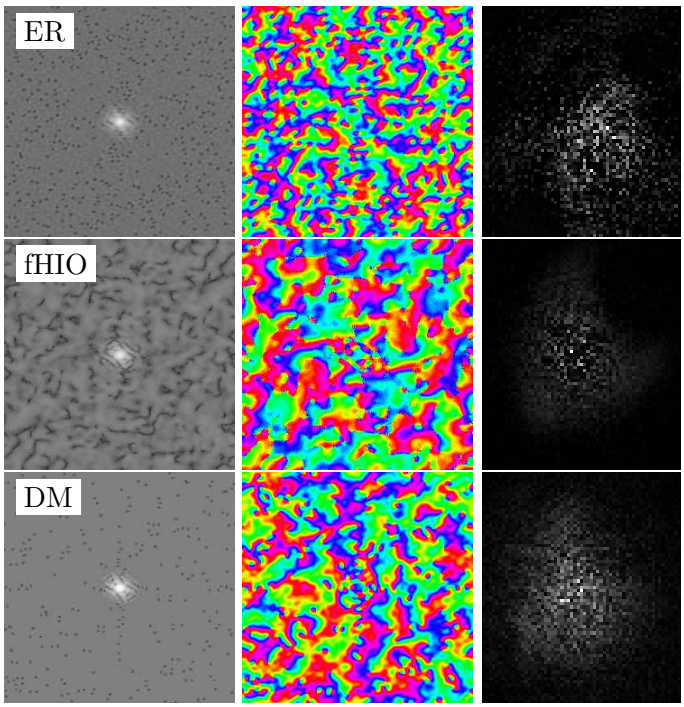


Figure A.10: Bias mean 500 ADUs.

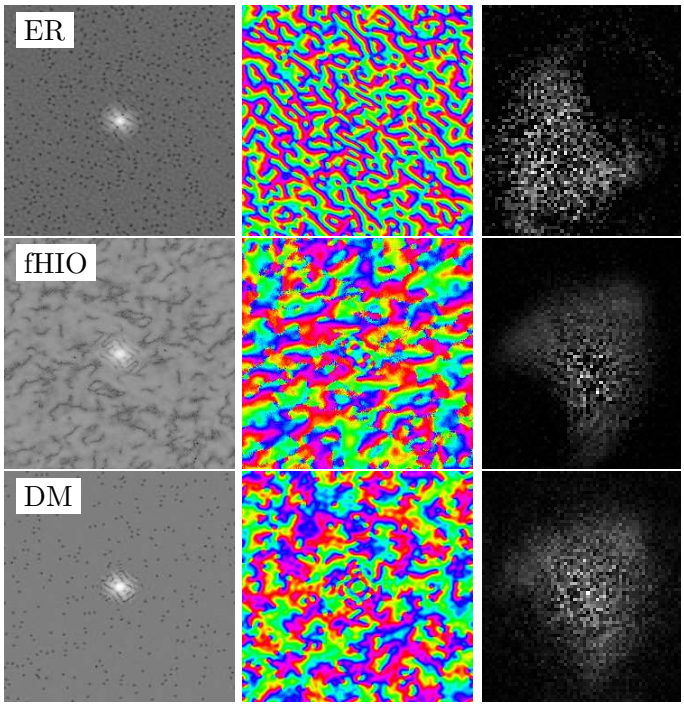


Figure A.11: Bias mean 300 ADUs.

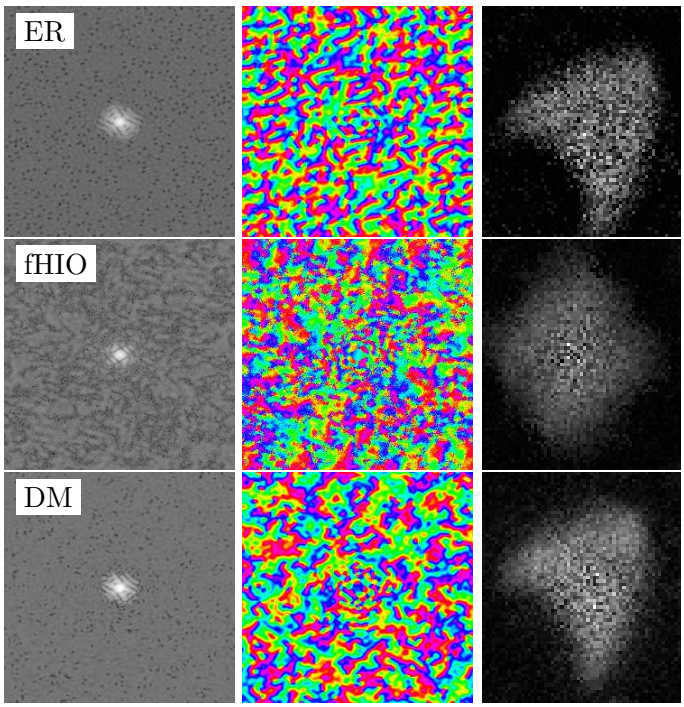
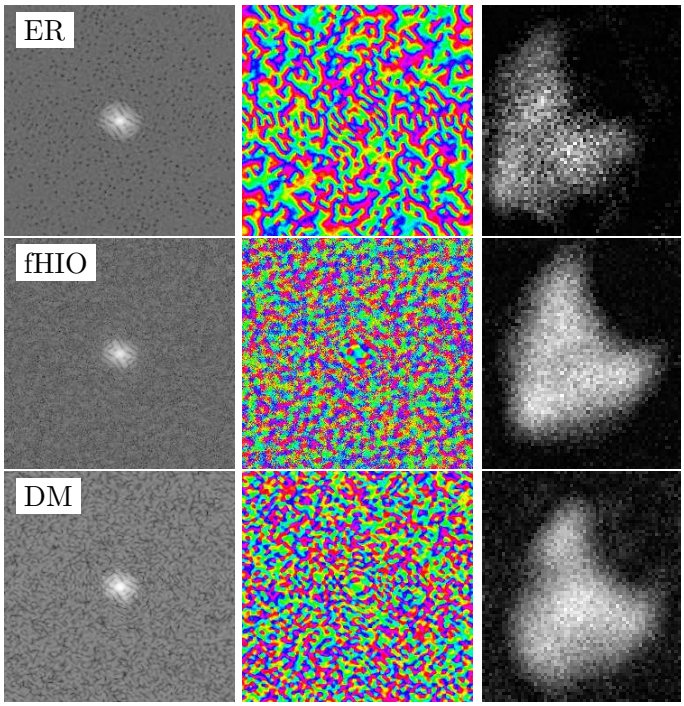


Figure A.12: Bias mean 100 ADUs.



$\chi^2 = 0.01617$

$\xi_1^2 = 0.2847$

$\xi_1^{orig} = 0.137$

$\chi^2 = 0.2265$

$\xi_1^2 = 0.0766$

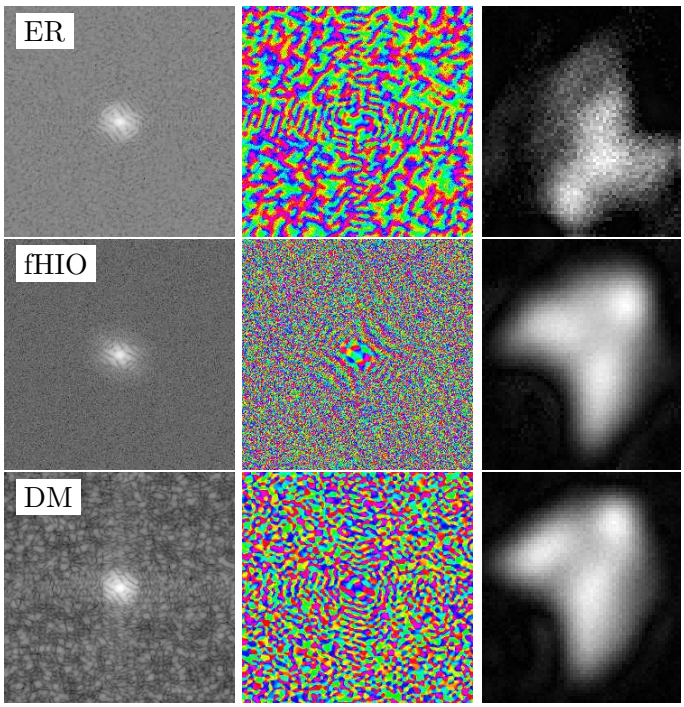
$\xi_1^{orig} = 0.1074$

$\chi^2 = 0.2148$

$\xi_1^2 = 0.09583$

$\xi_1^{orig} = 0.09147$

Figure A.13: Bias mean 50 ADUs.



$\chi^2 = 0.02369$

$\xi_1^2 = 0.202$

$\xi_1^{orig} = 0.2358$

$\chi^2 = 0.2025$

$\xi_1^2 = 0.056$

$\xi_1^{orig} = 0.07802$

$\chi^2 = 0.1986$

$\xi_1^2 = 0.02956$

$\xi_1^{orig} = 0.03527$

Figure A.14: Bias mean 10 ADUs.

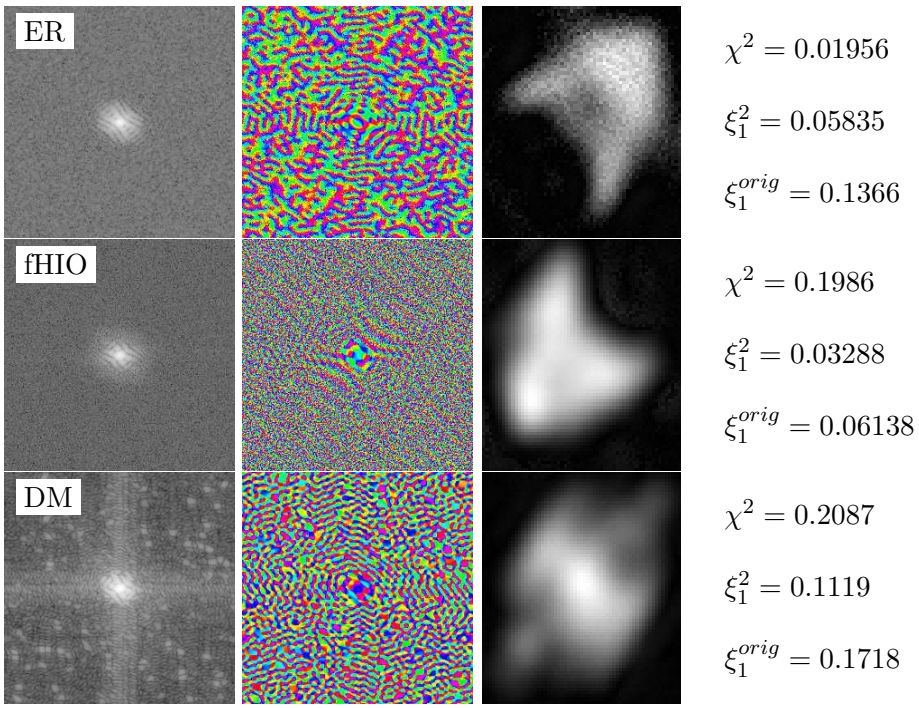


Figure A.15: Bias mean 0 ADUs.

A.3 Using fit parameters to Alleviate Noise effects

Using the 100% Background subtracted simulated CXD pattern, discussed in Section 4.4.2 and shown in Fig A.7h we present the reconstructed amplitudes and real-space objects from Sections 4.4.3 and 4.4.4.

A.3.1 HIO Parameter Study

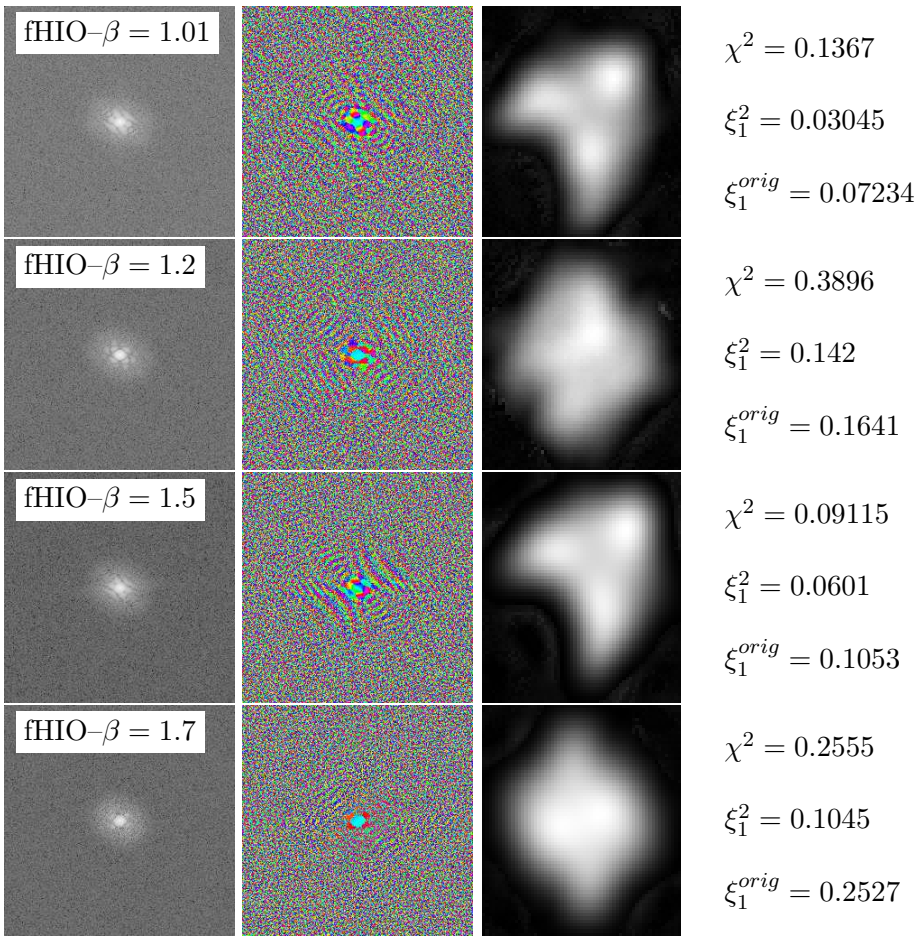


Figure A.16: HIO with $\beta > 1$.

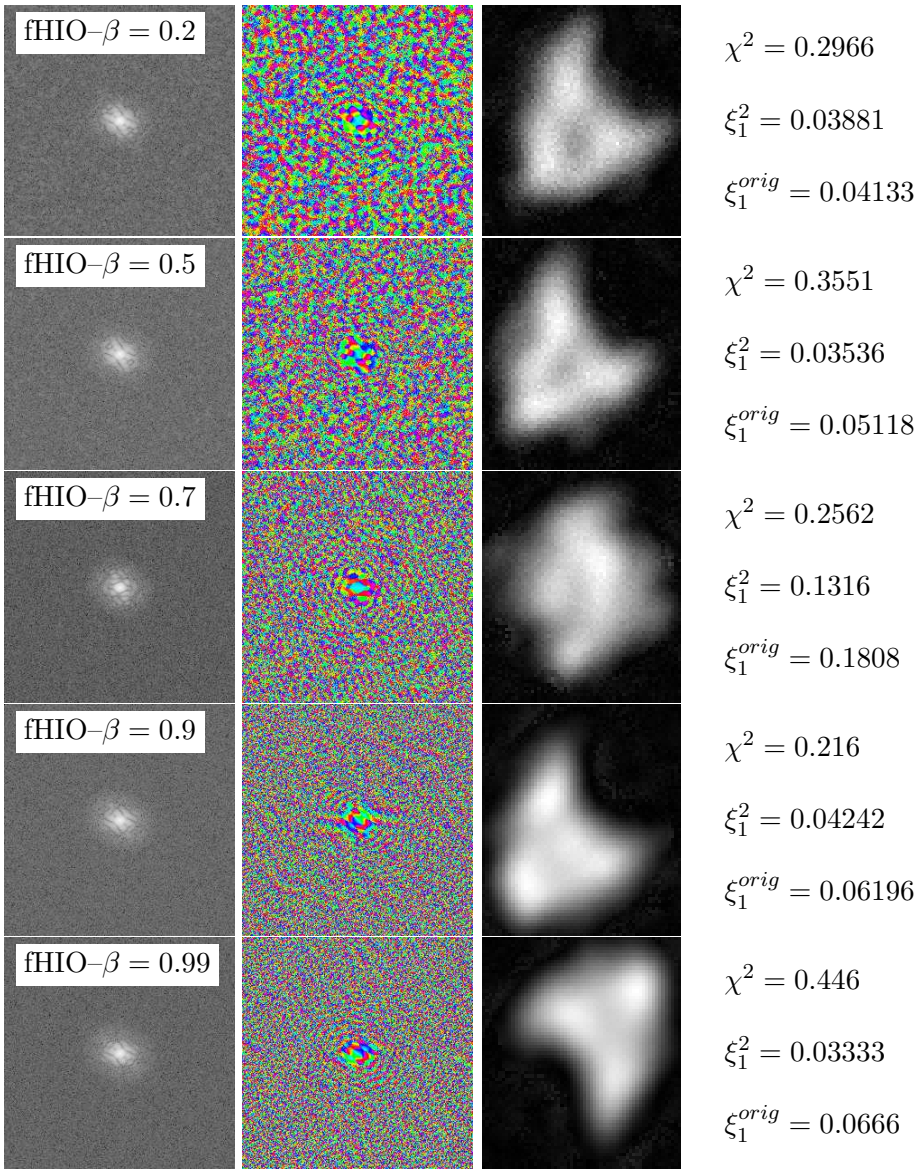


Figure A.17: HIO with $0 < \beta < 1$.

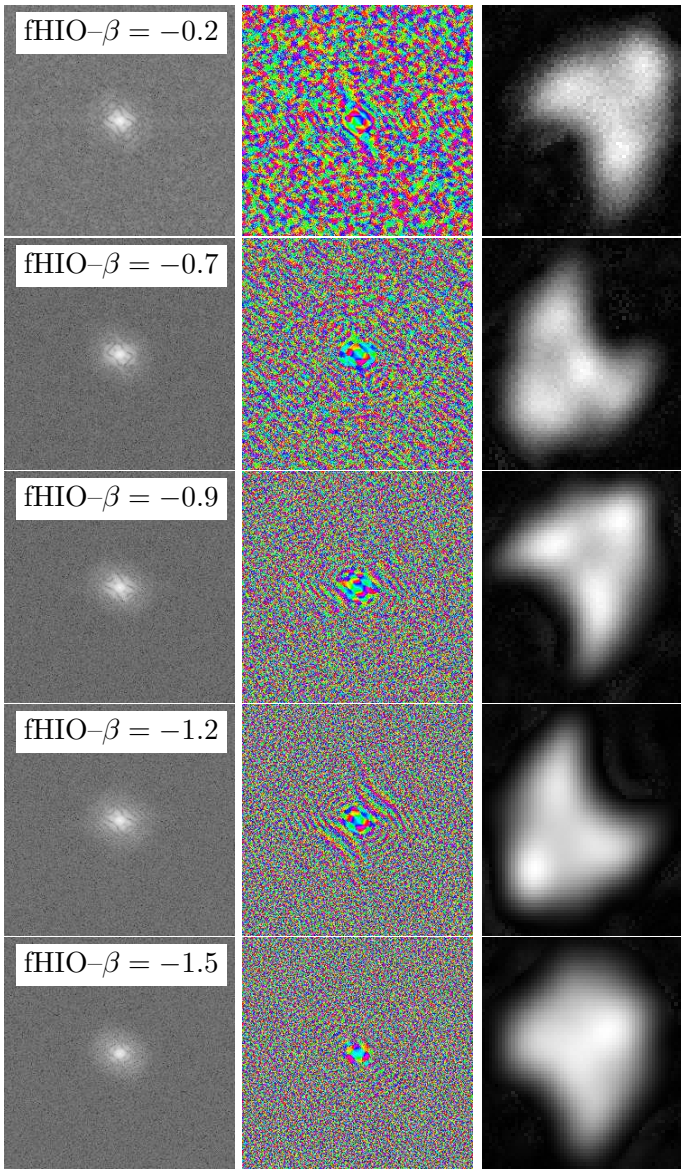


Figure A.18: HIO with $\beta < 0$.

A.3.2 Difference Map Parameter Study

Elser's Difference Map uses three parameters: β, γ_1, and γ_2. Two parameter sets go by special names: HIO with $\gamma_1 = -1$ and $\gamma_2 = 1/\beta$ and the maximally contractive parameters $\gamma_1 = -1/\beta$ and $\gamma_2 = 1/\beta$. In the following simulations these three parameters are varied.

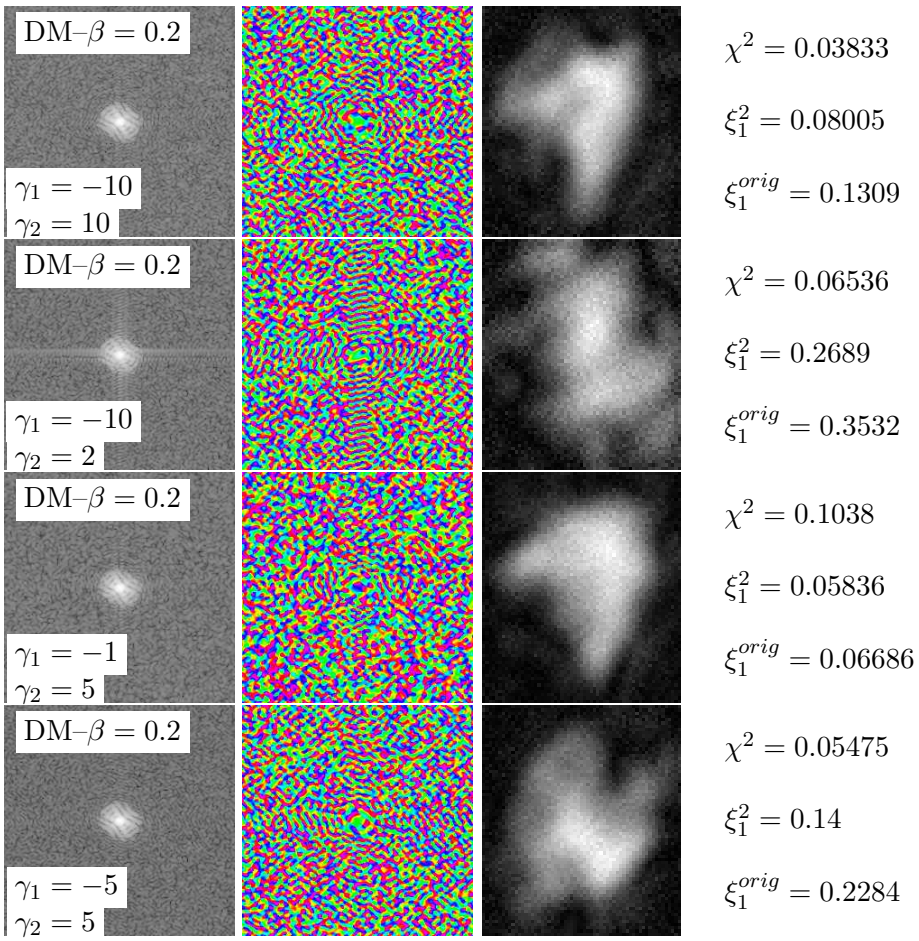


Figure A.19: DM with $\beta = 0.2$ and various γ_i's.

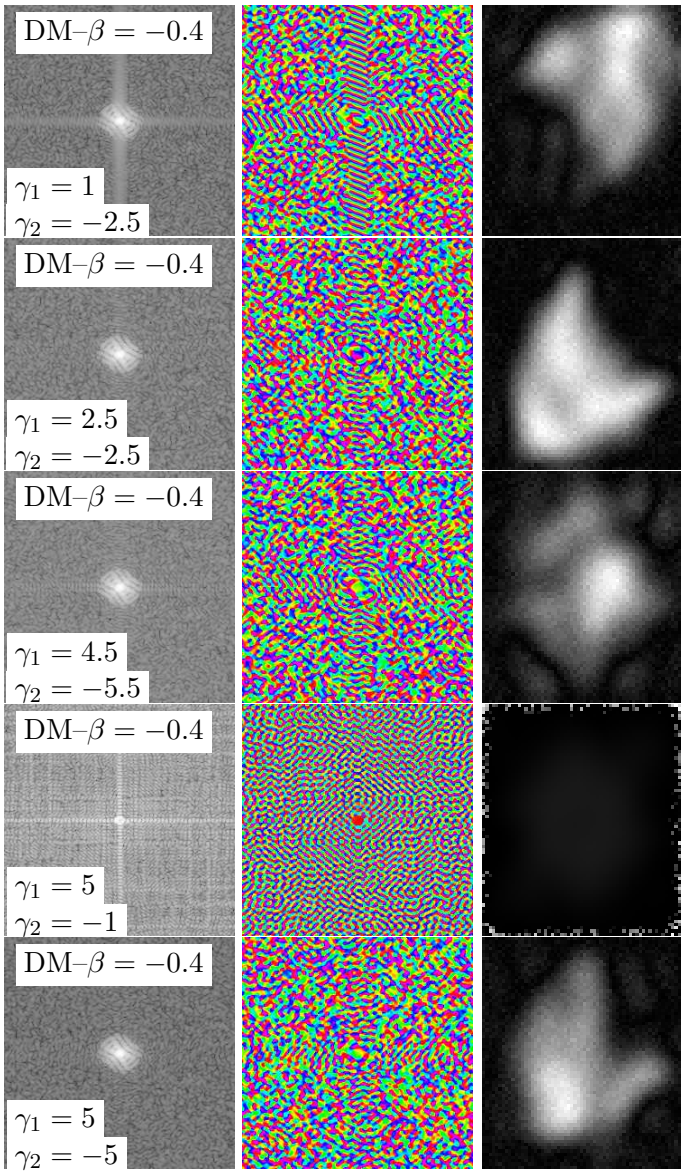


Figure A.20: DM with $\beta = -0.4$ and various γ_i's.

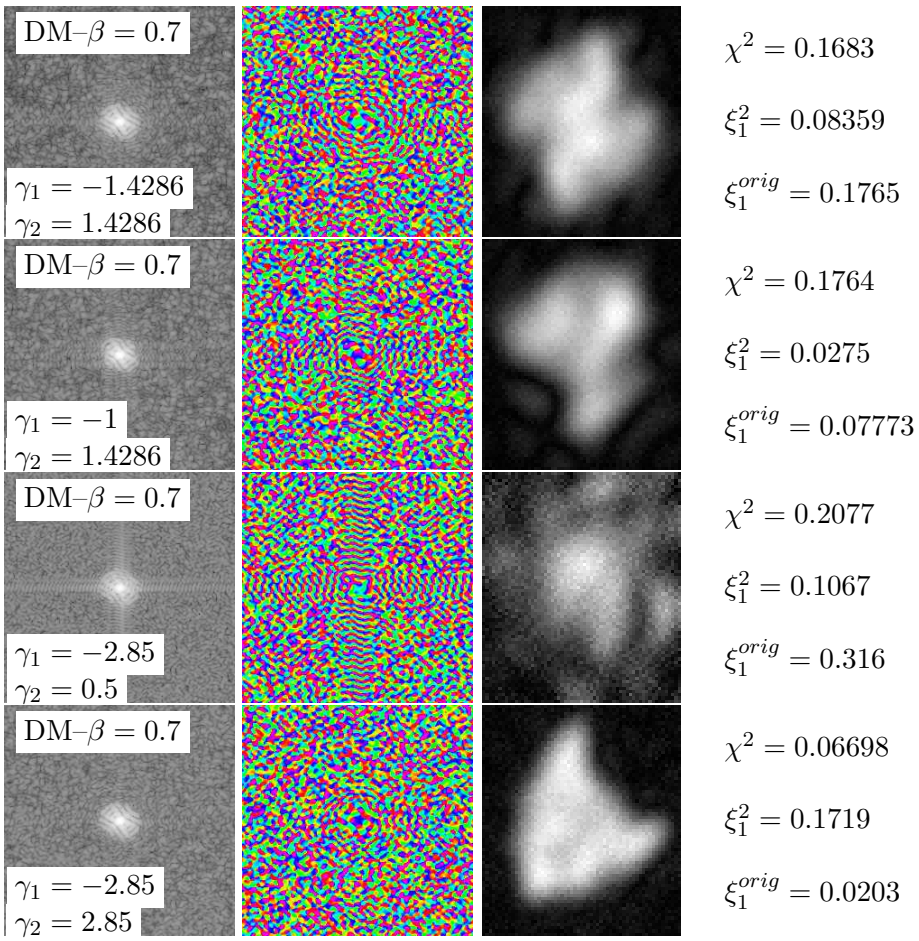


Figure A.21: DM with $\beta = 0.7$ and various γ_i's.

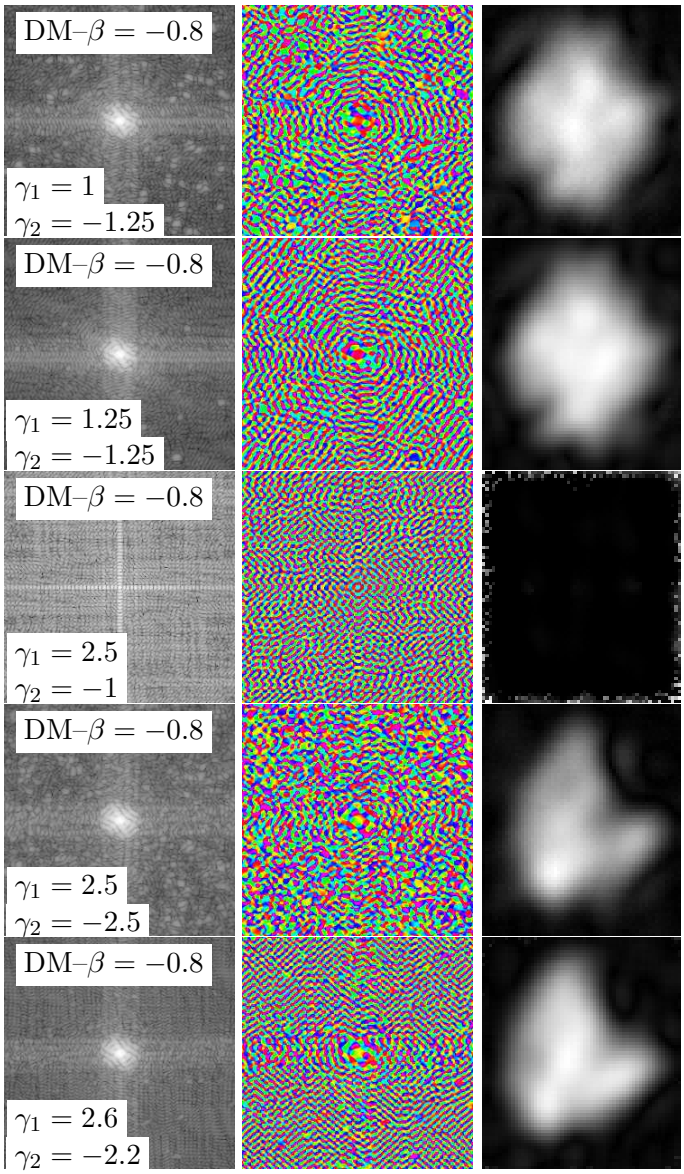


Figure A.22: DM with $\beta = -0.8$ and various γ_i's.

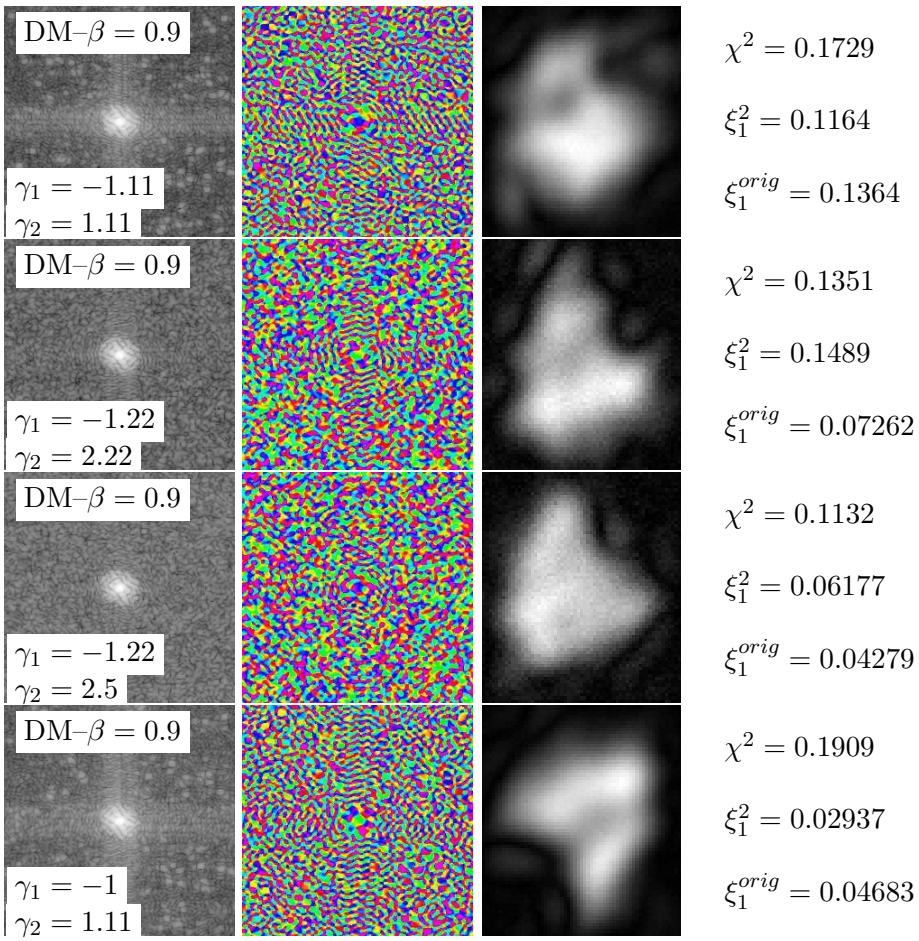


Figure A.23: DM with $\beta = 0.9$ and various γ_i's.

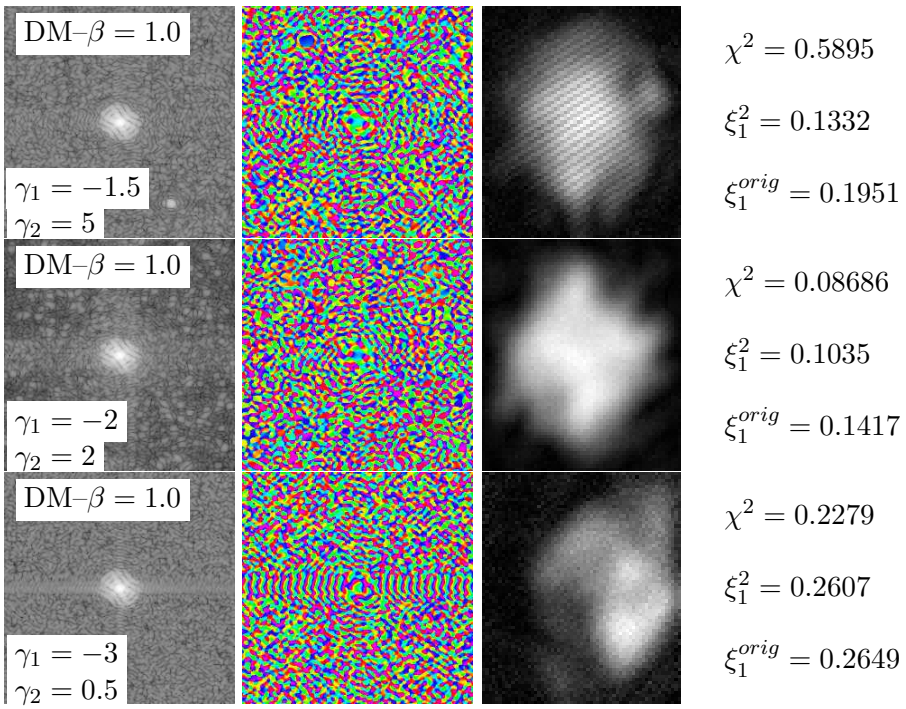


Figure A.24: DM with $\beta = 1.0$ and various γ_i's.

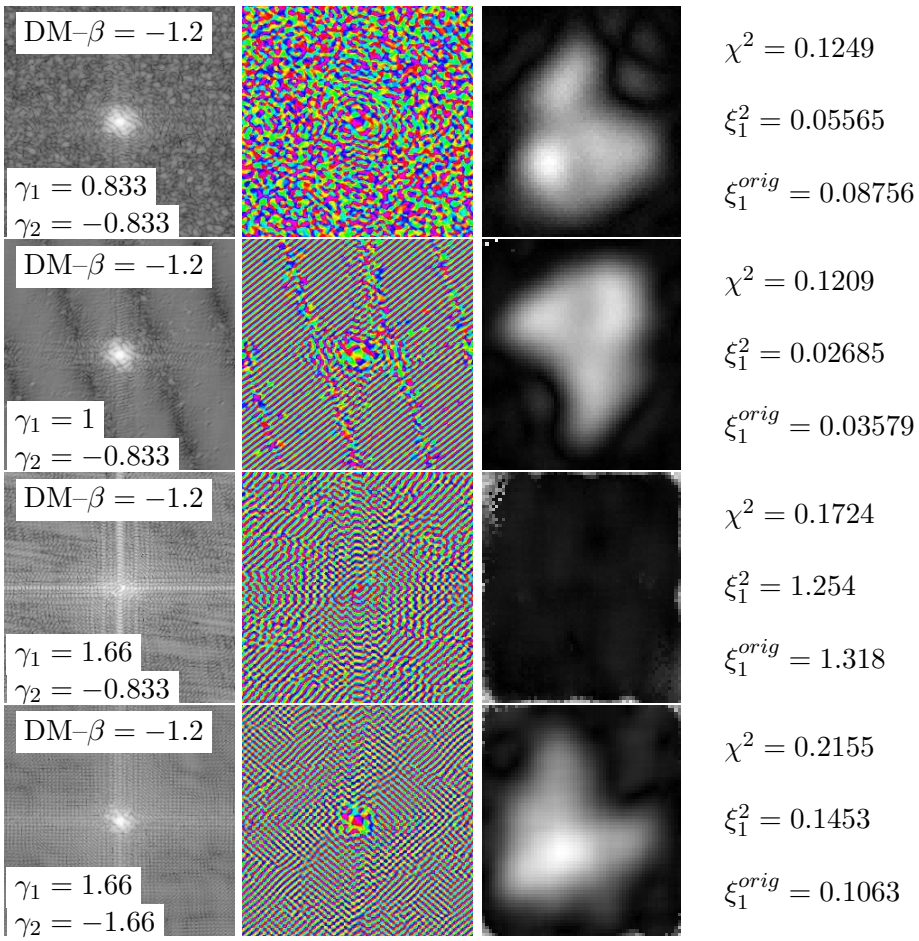


Figure A.25: DM with $\beta = -1.2$ and various γ_i's.

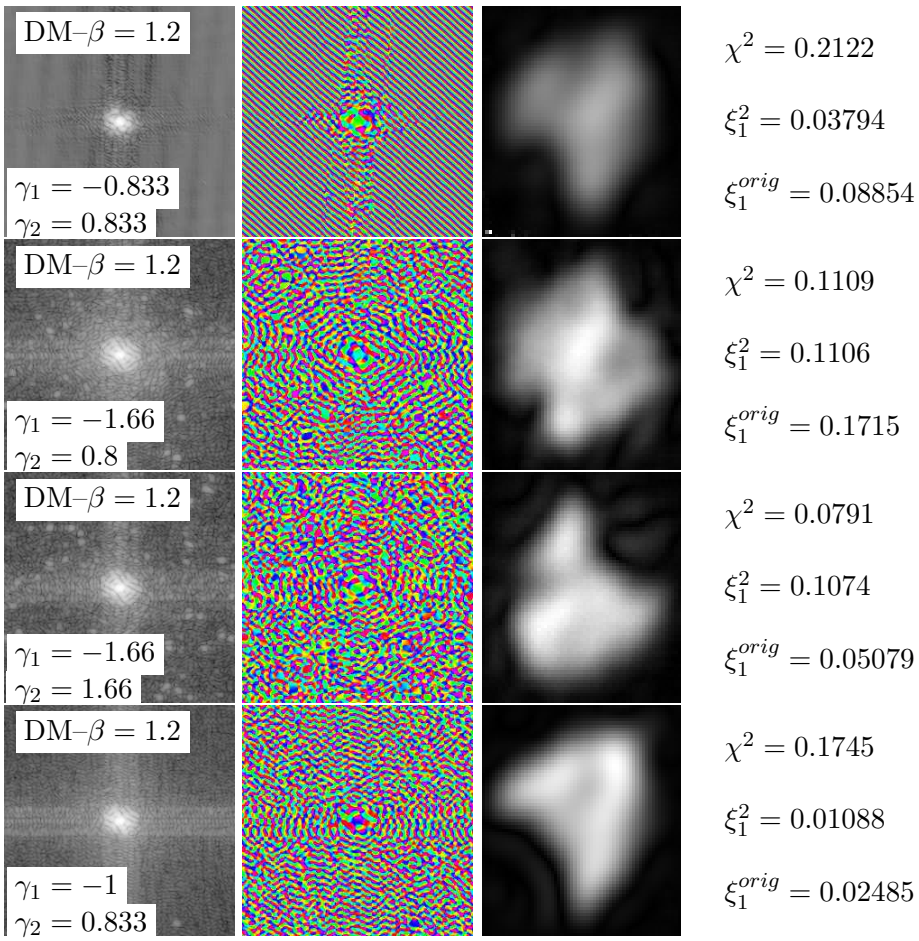


Figure A.26: DM with $\beta = 1.2$ and various γ_i's.

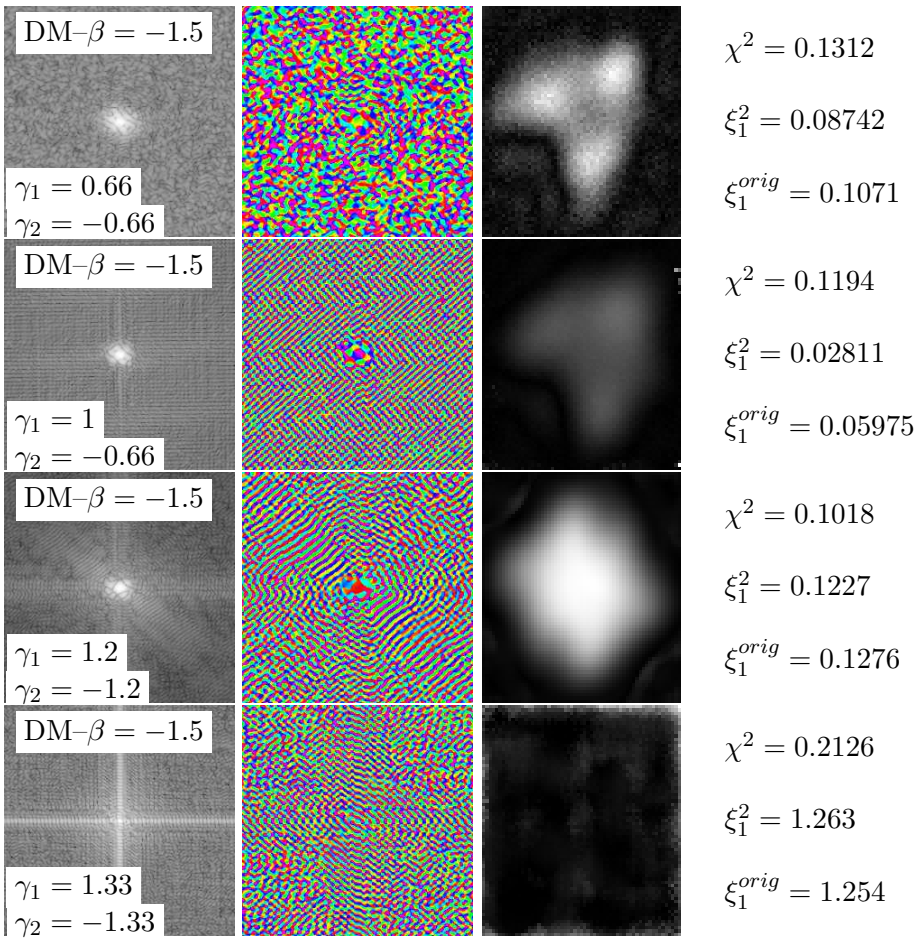


Figure A.27: DM with $\beta = -1.5$ and various γ_i's.

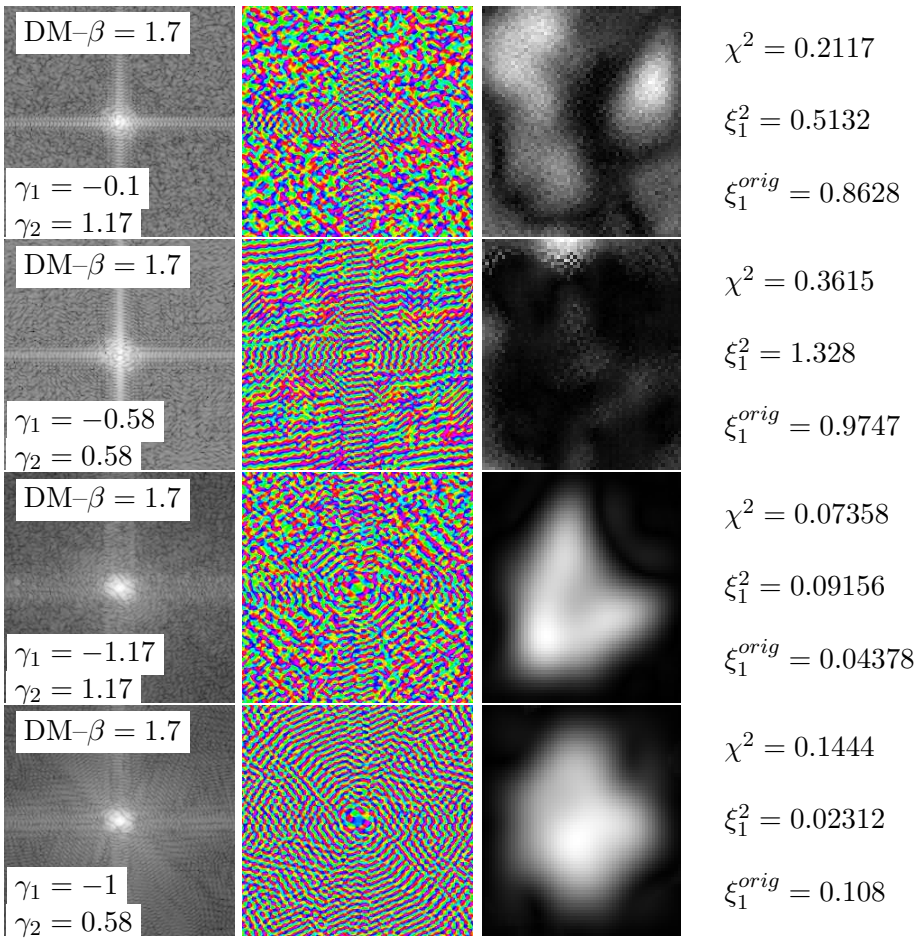


Figure A.28: DM with $\beta = 1.7$ and various γ_i's.

A.4 Photon Number

The results of fits to simulated CXD patterns with changing photon number, from Section 4.4.5 are shown here. Fig. A.29. In these images, only counting noise is simulated.

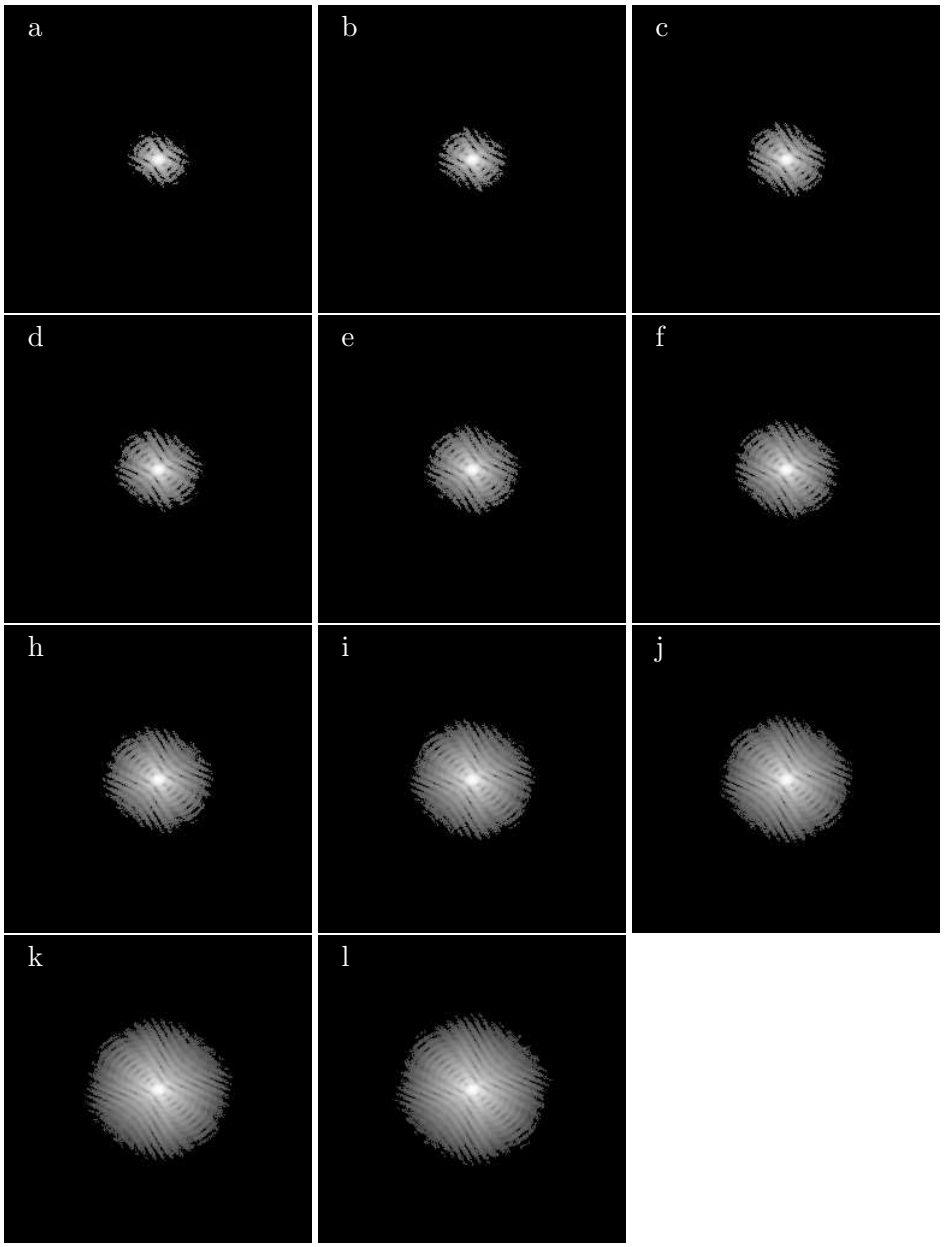


Figure A.29: Simulated CXD patterns with different photon number used as Fourier modulus constraints in Sections 4.4.5 and 4.4.6. The number of photons in each simulation are: 9×10^4 in a, 2×10^5 in b, 5×10^5 in c, 1×10^6 in d, 2×10^6 in e, 5×10^6 in f, 1×10^7 in h, 5×10^7 in i, 1×10^8 in j, 5×10^8 in k, and 1×10^9 in l. The images are the logarithm of the intensity.

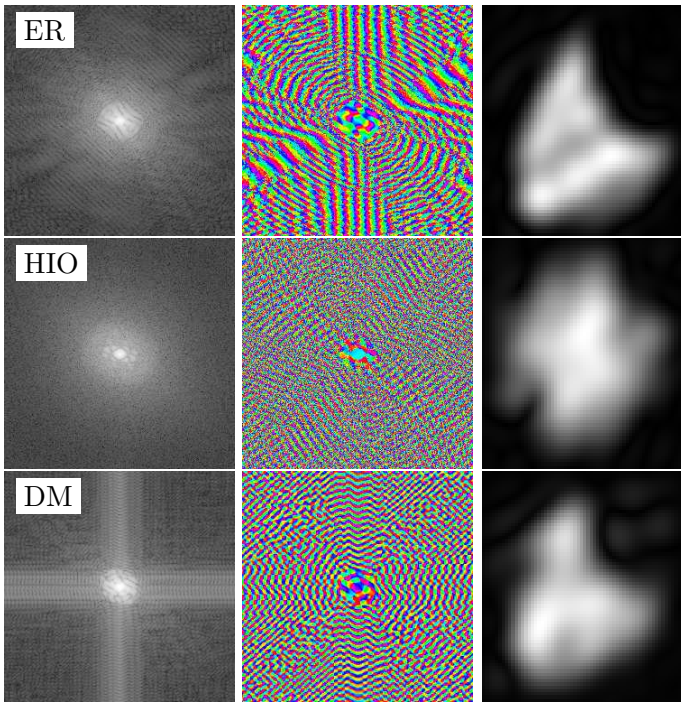


Figure A.30: 9e4 photons.

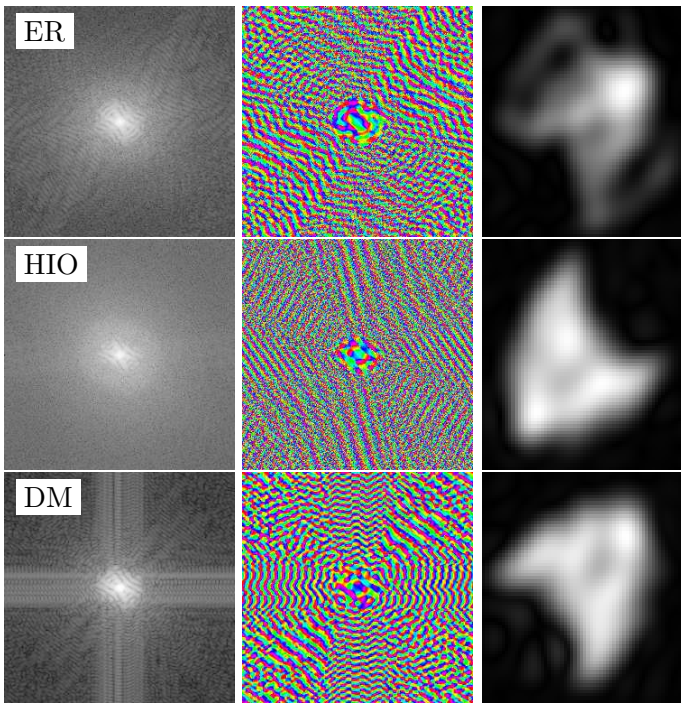


Figure A.31: 2e5 photons.

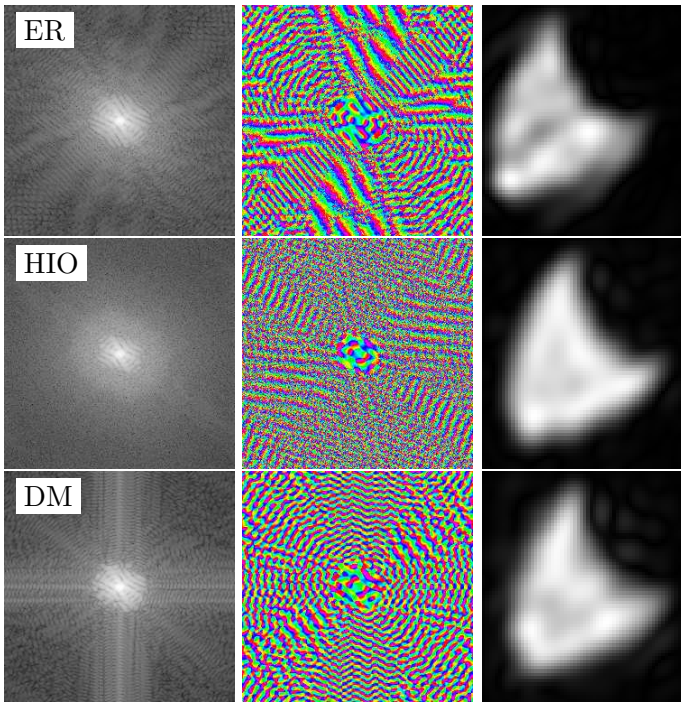


Figure A.32: 5e5 photons.

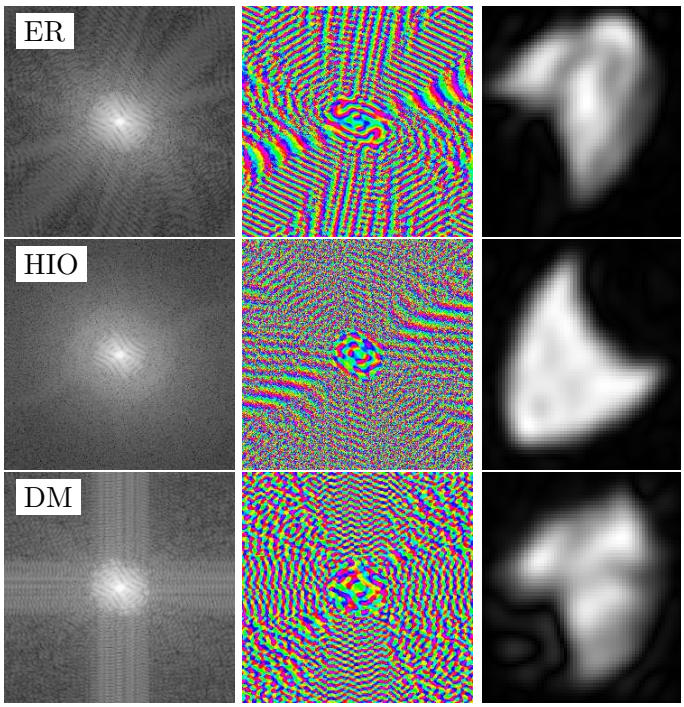


Figure A.33: 1e6 photons.

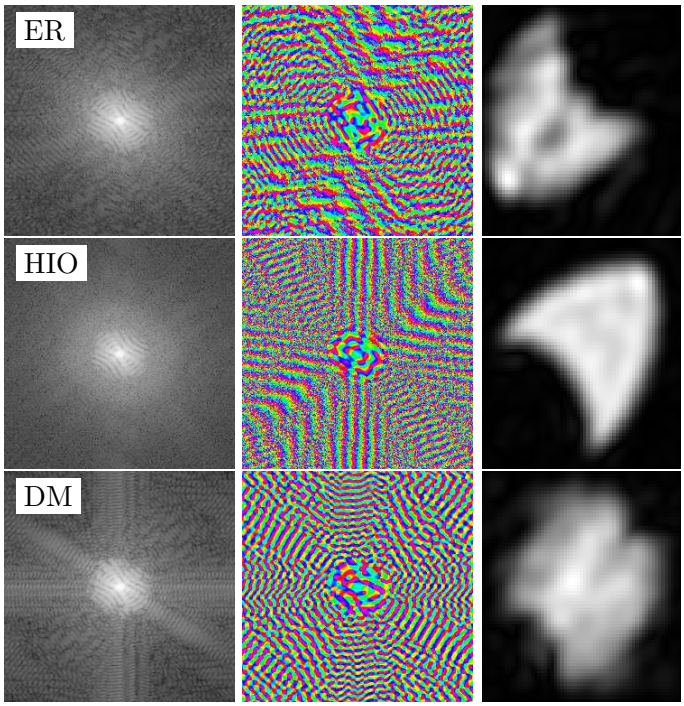


Figure A.34: 2e6 photons.

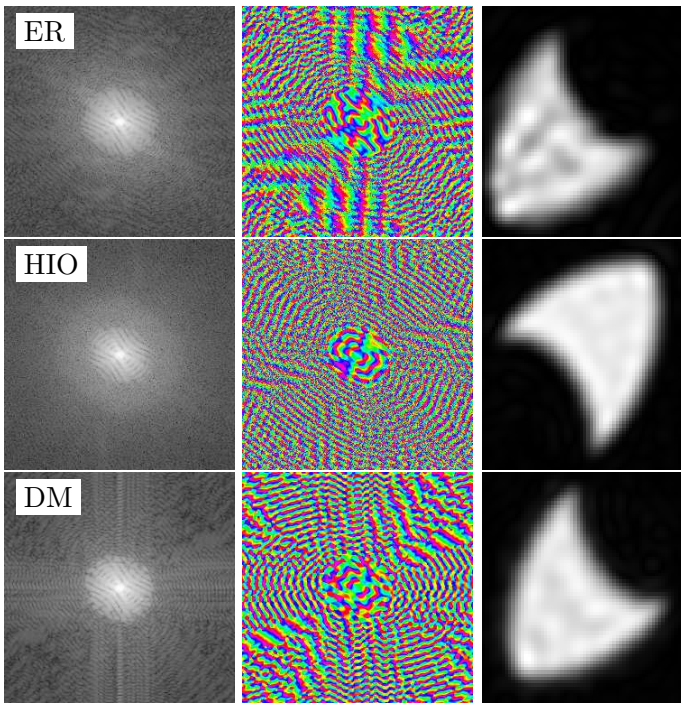


Figure A.35: 5e6 photons.

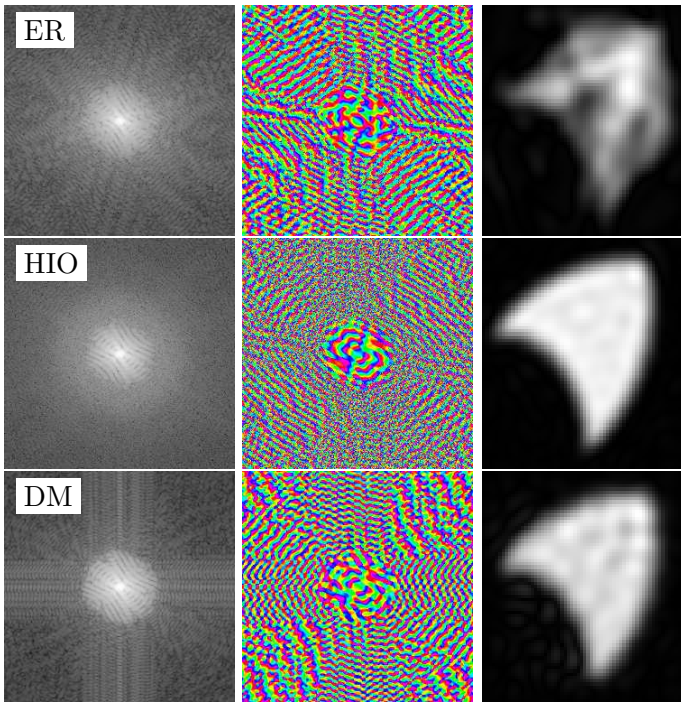


Figure A.36: 1e7 photons.

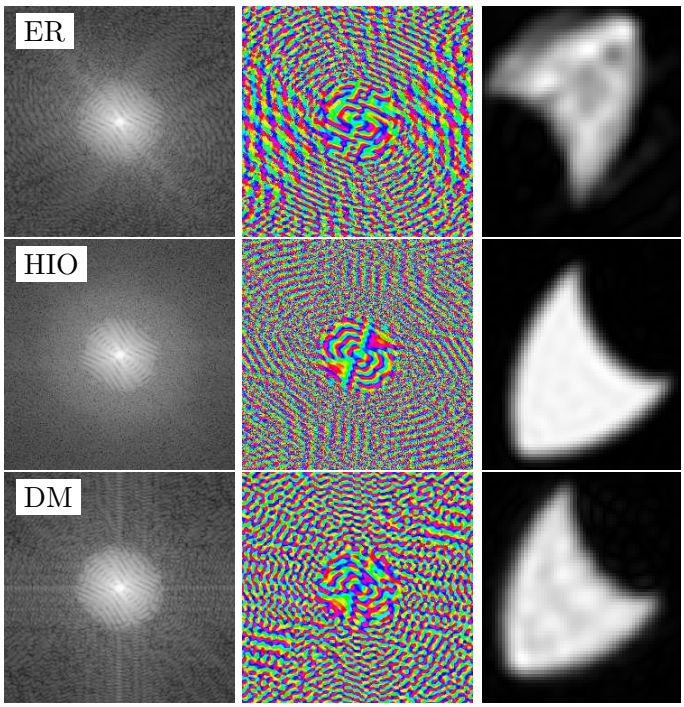


Figure A.37: 5e7 photons.

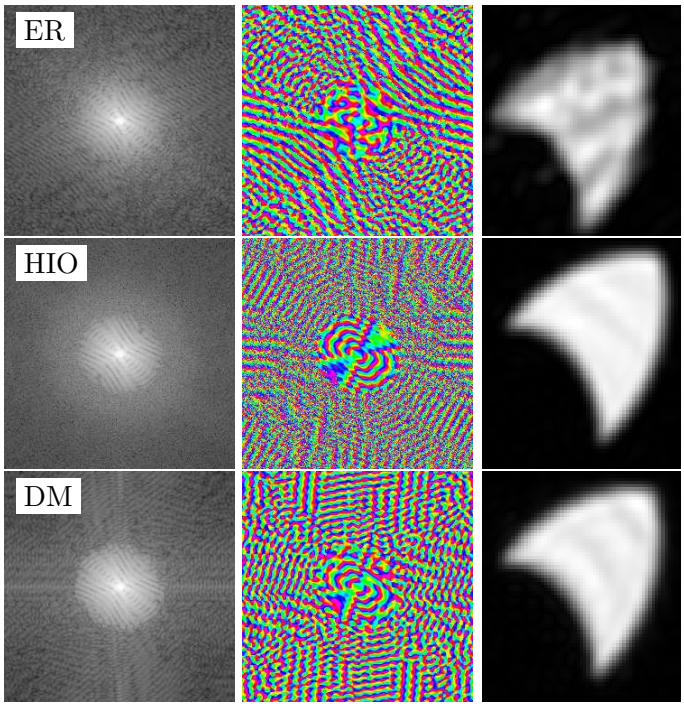


Figure A.38: 1e8 photons.

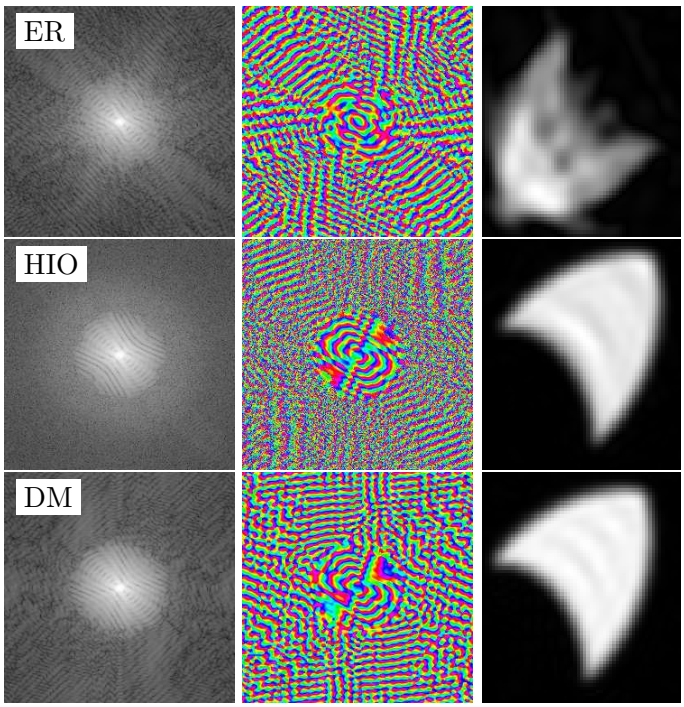


Figure A.39: 5e8 photons.

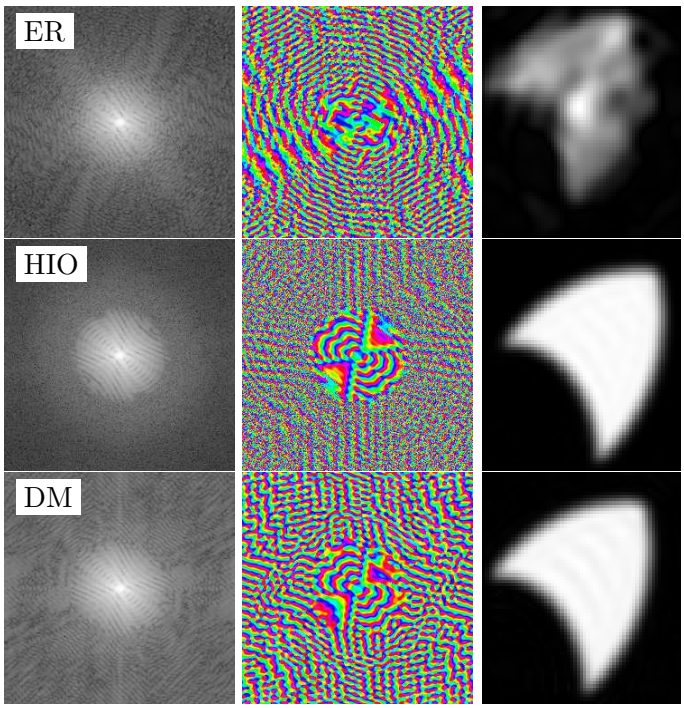


Figure A.40: 1e9 photons.

A.5 Mixture of ER and HIO

This section contains the images corresponding to Section 4.4.6. The simulated patterns are those in in Fig. A.29. Rather than using ER, HIO, or DM, the fitting is performed using 150 cycles of ER, 50 cycles of HIO, and 100 cycles of ER. The HIO parameter is $\beta = 1$.

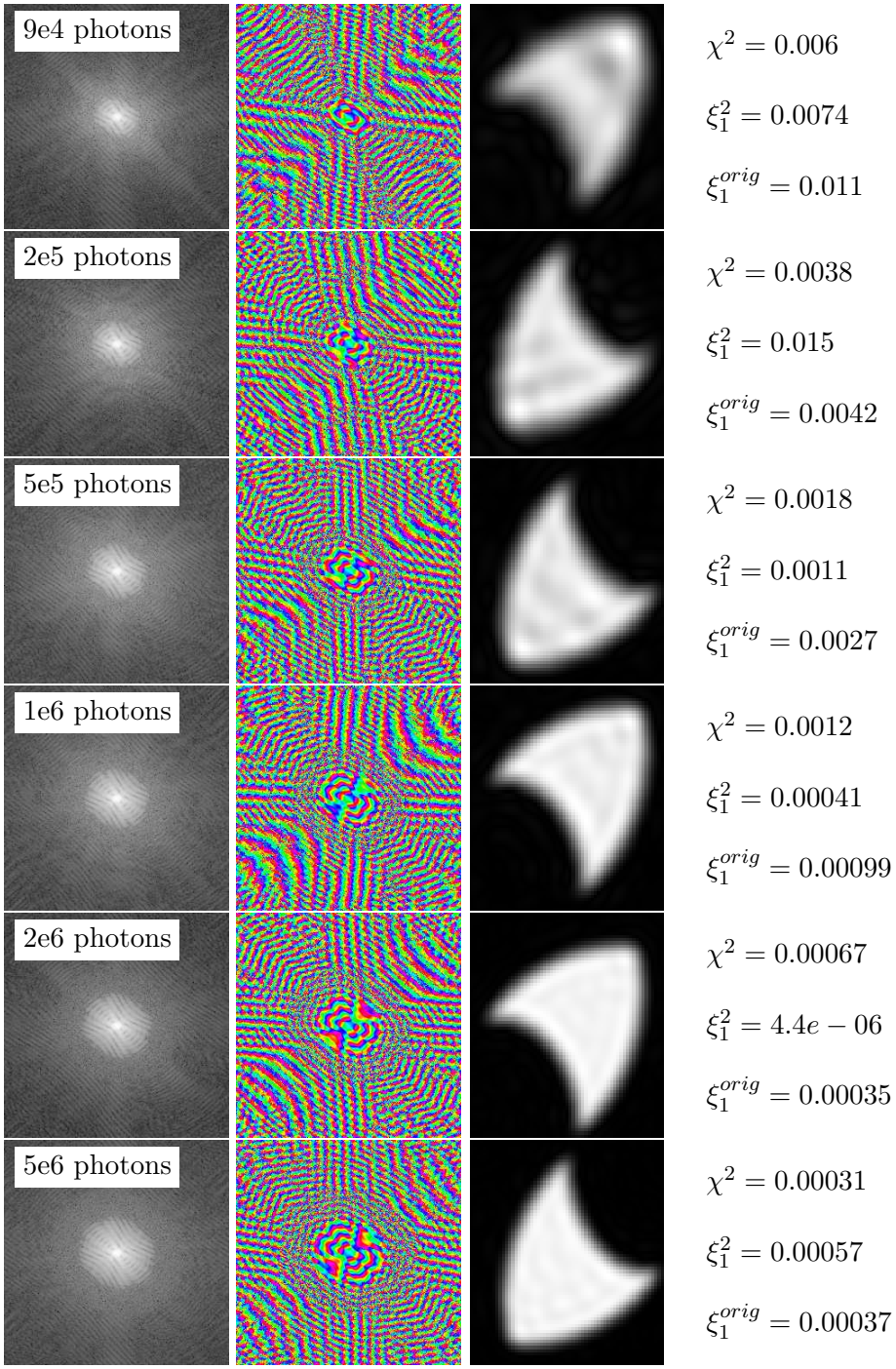


Figure A.41: ER/HIO mixture with varying photon number.

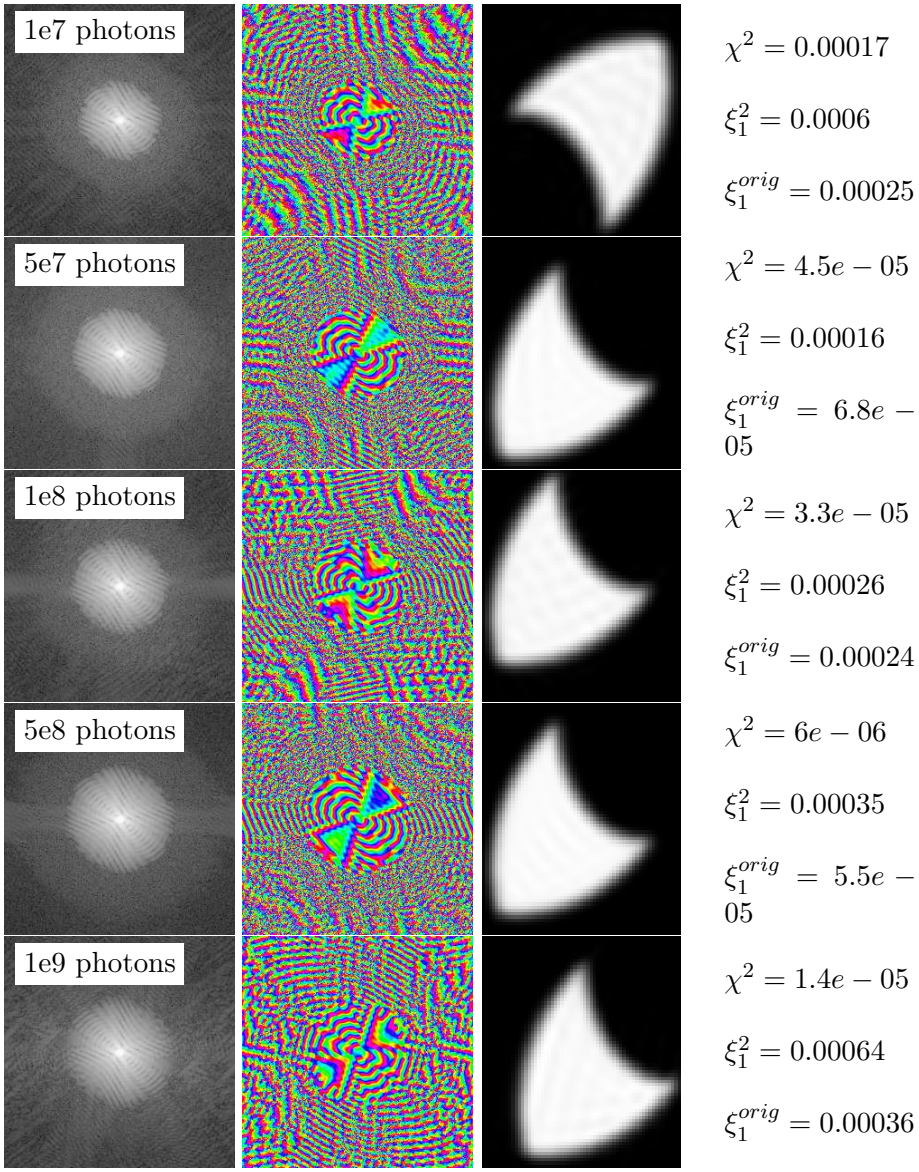


Figure A.42: ER/HIO mixture with varying photon number.

A.6 ER/HIO Mixture varying β

This section contains the images corresponding to Section 4.4.7. The simulated pattern is that of Fig. A.29f with 5×10^6 photons. Rather than using ER, HIO, or DM, the fitting is performed using 150 cycles of ER, 50 cycles of HIO, and 100 cycles of ER. The HIO parameter is varied and takes values: $\beta = -0.2$, $\beta = -0.7$, $\beta = -0.9$, $\beta = -1.2$, $\beta = -1.5$ $\beta = 0.2$, $\beta = 0.5$, $\beta = 0.7$, $\beta = 0.9$, $\beta = 0.99$ $\beta = 1.01$, $\beta = 1.2$, $\beta = 1.5$, and $\beta = 1.7$.

.

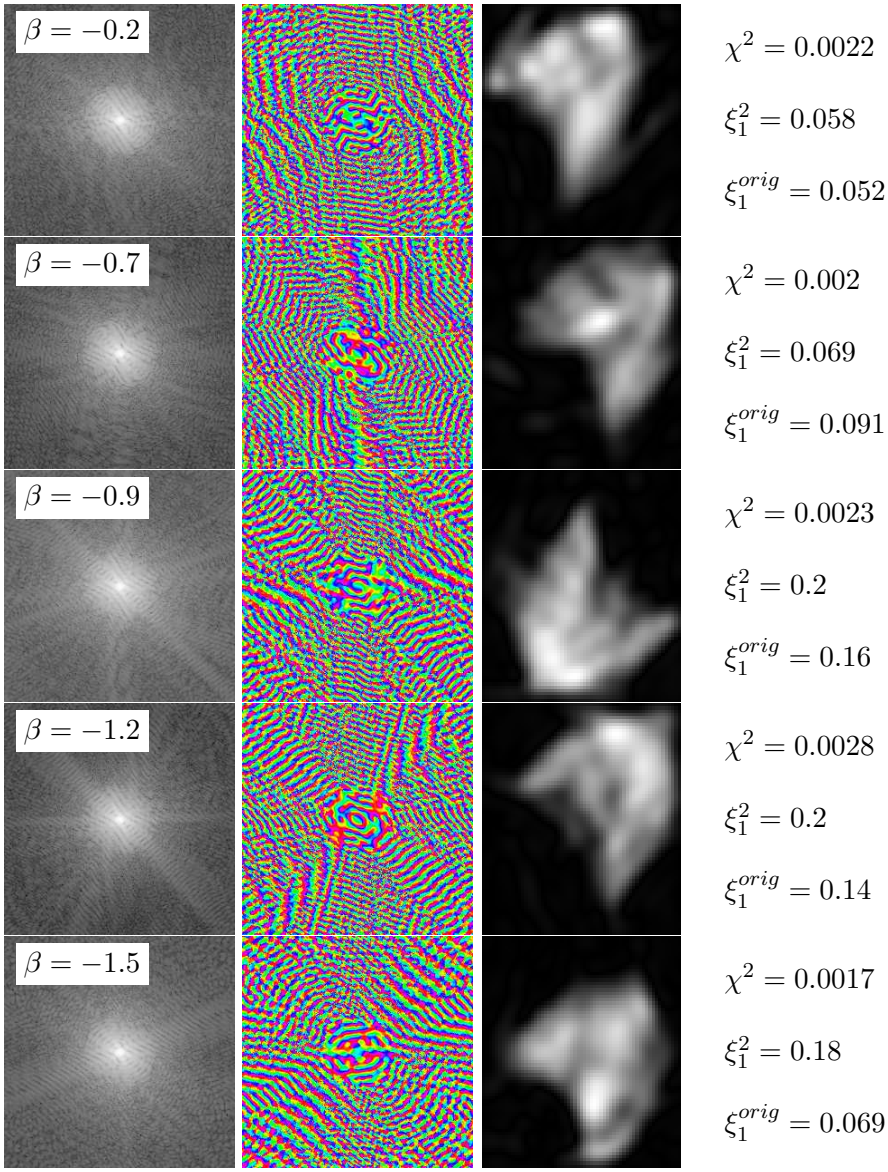


Figure A.43: ER/HIO mixture with HIO's βs under zero.

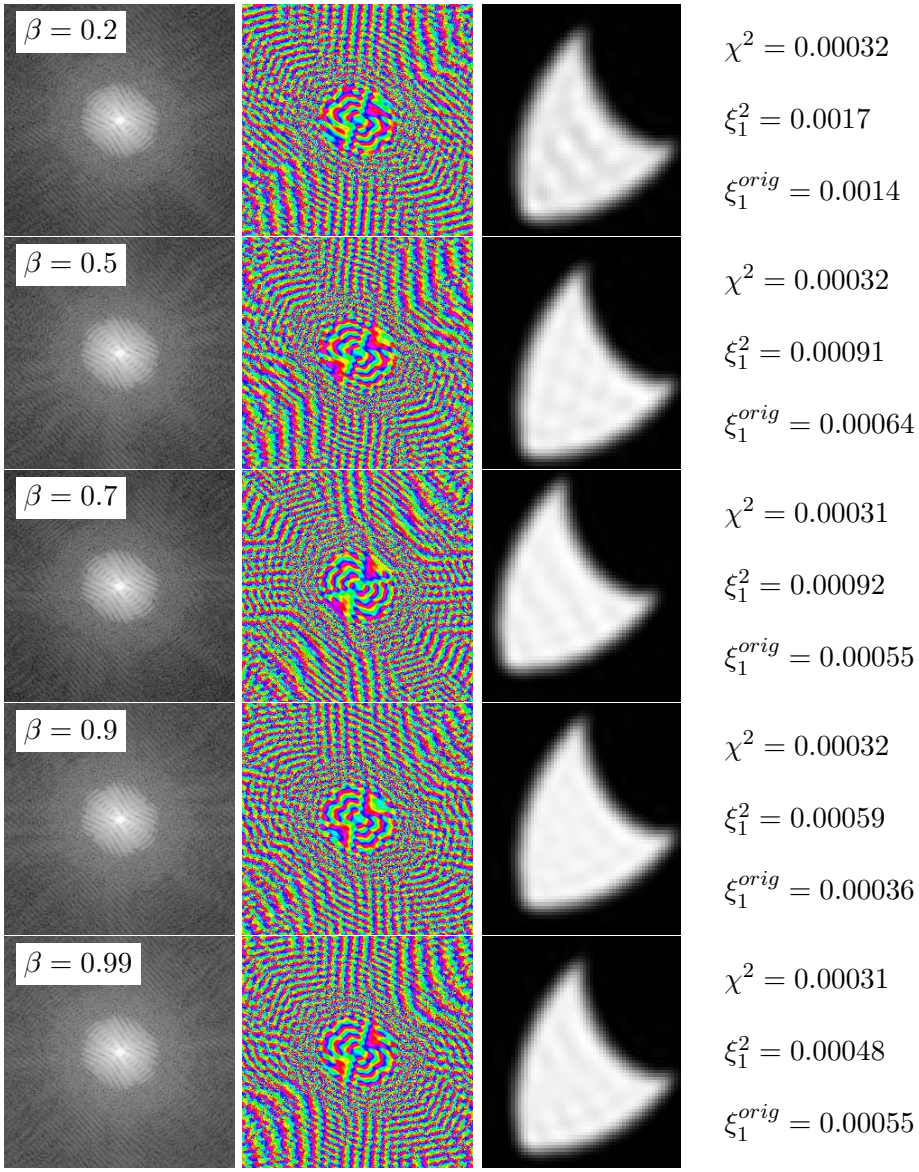


Figure A.44: ER/HIO mixture with HIO's βs over zero, under one.

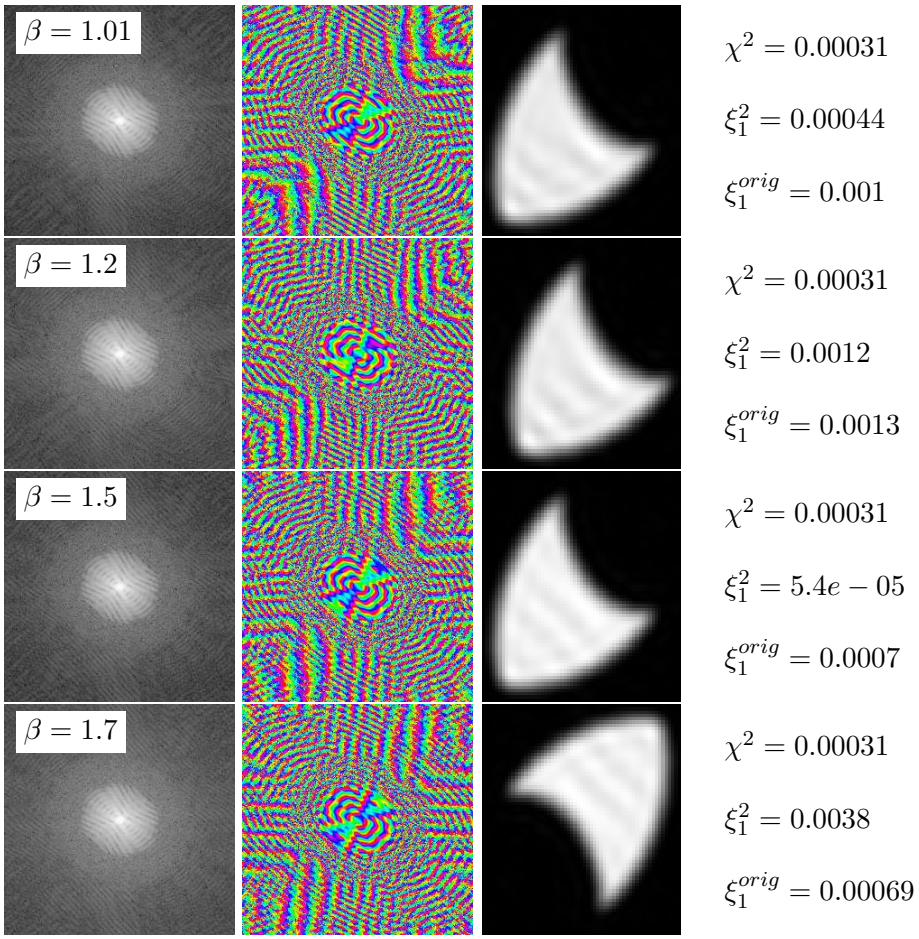


Figure A.45: ER/HIO mixture with HIO's βs over one.

Appendix B

Listing of Programs

To facilitate the fitting of CXD taken with our Princeton Instruments CCD, J. Pitney devised an alternate file format for storing and handling the data to overcome, among other things, the fact that data are stored as long integers in the spe file format. Section B.1 briefly introduces the format.

The programs in listed in Section B.2 were written by J. Pitney, G. Williams, M. Pfeifer, and I. Vartanyants. These are a sampling of the utilities commonly used in 2D and 3D fitting as described in Chapter 3.

B.1 SP4 Array

The sp4 data array is defined as:

```
typedef struct _Sp4Array
/* define build number for future compatibility*/
int build;
/* string to hold information about the array */
char comments[MAX_COMMENT_LENGTH];
/* real or complex array */
int datatype;
/* the dimensionality of the array */
int ndim;
/* the total number of elements in the array */
unsigned long int nelements;
/* the x, y (and optionally z) dimensions */
unsigned long int* nn;
/* data stored as double precision floating point numbers */
double* data;
Sp4Array;
```

This is the structure currently used our programs.

B.2 Useful Programs

- `2danticorr <infile.sp4> <outfile.sp4>`

  Writes the anticorrelation of `<infile.sp4>` into `<outfile.sp4>`. Calculated by element-wise product of the modulus of the amplitude of the input array with itself.

- `2dautocorr <infile.sp4> <outfile.sp4>`

  Writes the autocorrelation of `<infile.sp4>` into `<outfile.sp4>`. Calculated by element-wise product of the square of the amplitude of the input array with itself.

- `2dblackout.c <in> <out> <x-center> <y-center> <x-length> <y-length>`

  Simply replaces all elements in the specified region with zeroes. Useful, for example, in Fourier filtering to find vortices.

- `2dclip <in.sp4> <out.sp4>`

  Removes the highest valued pixel in an array by setting it to zero. Useful in removing the zero-order peak in the Fourier transform of an array with very high average amplitude per pixel, so that **`2dplot`**'s dynamic range is reduced and the array may be more easily examined.

- `2dcrop <x0> <y0> <new-width> <new-height> <in> <out>`

  Crops an array. `<x0> <y0> is the upper left corner of the new array.`

- `2dcroppad2 <N-x> <N-y> <extend-mode> <infile.sp4> <outfile.sp4>`

  A more versatile cropping and padding program. The center point of the new array is the same as the input array. `<N-x> <N-y>` give the new array dimensions. `<extend-mode>` takes ones of three values: 1 gives zero padding, 2 pads with a constant by copying the value of the nearest pixel in the original array into the new one, and 3 pads with a random deviation (compiled into the program, currently ± 1) about a mean calculated from a square of predefined size (currently 15×15 px), and the two are merged by linear interpolation across a predefined distance (currently 40 px).

- `2ddiff <first> <second> <diff>`

  Calculates the sum of squared differences in two arrays and outputs the result.

- `2ddiffmap config-file`

  Implementation of Elser's Difference Map[28]. A configuration file is expected. An example would be:

```
#number of fits and number to keep
10 3
#real-space support
support
#reciprocal-space Fourier modulus
measured
#iterations
200
#beta
1.00
#gamma1
-1.00
#gamma2
1.00
```

- `2dgfilter <input> <outout> <alpha>`

  Multiplies an array by a radial Gaussian. The half width of the Gaussian is given by α in $\exp(-\alpha r^2)$ =$\exp(-r^2/2\sigma^2)$. The radial distance is calculated from the center of the array. Used to reduce the effect of aliasing on the reconstruction.

- `<infile.sp4> <outfile.sp4>`

  Unwraps an array in preparation for fitting. In this case, it's implemented by moving the center to the corners of the array and adding π to every other point in the array.

- `2dplot:  2dmanplot <input sp4 file> <min> <max>`

  Plots a data array with the specified minimum and maximum values, rather than allowing **`2dplot`** to automatically determine the dynamic

range. Also used to assure that images are plotted with the same range for comparison.

- `2dmovieprep <list of ppms>`

  Takes a series of UNIX bitmap (ppm) files, usually generated by `2dmultifit` and draws a progress bar through the middle. The bar is color coded based on the name of the input file. These images may then be used as input to `mpeg_encode` or another utility for conversion into an mpeg movie.

- `2dmultifit8 <configfile> [<pgm period> <seed>]`

  Implements Fienup's ER and HIO, Millane's HIO, and other iterative fitting routines. The configuration file is of the form:

```
#number of fits and number to keep
10 3
#real-space support
support
#reciprocal-space Fourier modulus
measured
#number of cycles
200
#cycles #each cycle requires:  type, iterations,
# parameter1(usually beta in HIO),
# parameter2(usually epsilon in mHIO)
#current modes for hio are listed in table 3.3
#examples
#100 iterations of ER
hio
100 5 0.9 0.00
#50 iterations of mHIO with beta=0.9, epsilon=0.01
hio
50 4 0.9 0.01
#50 iterations of HIO with beta=0.9
hio
50 15 0.9 0
#move real-space object to center of array
mov
1 1 0 0
#patch vortices.
```

```
#include all points were gradient is within 10%
#of steepest, use a radius of 15 px vor
1 1 0.1 15
```

Optionally, a gray scale bitmap (pgm) image can be written out with the provided period. Further, a fit may be repeated by providing the second optional parameter. This allows the fitting of the same random starting phase set with different methods. The seed used in a fit is included in the sp4 data array structure's text field and can be read using **`readheader`**. Any points of negative amplitude in the Fourier modulus constraint as taken to be "flagged out" and the constraint will not be applied in these regions.

- `2dflagsp4 <in> <out> <x0> <y0> <R0> [<x1> <y1> <R1>...]`

  Used to flag out regions of an array so that the Fourier modulus constraint will not be applied in those regions. Such regions are specified by giving the center and radius of a circle containing them. Any number of such regions may be specified. This can be useful when scattering from another particle is intruding upon the one measured.

- `2dplot <nbands> <input sp4 file>`

  Writes a UNIX bitmap (ppm) image of the data array. `nbands` is the number of levels in the image, with 0 being 256 levels. If the array is complex, an amplitude and phase image will be generated. The range of the image is calculated and the image rescaled to fit within the 8-bit range of the output image.

- `2dpoly <configuration-file> <output-file>`

  Creates an array composed of ones within the area bounded by the planes provided in the configuration file. A configuration file might look like:

  ```
  #0 for sharp edges, -1 for blank array
  #1 for edges smoothed by ‘‘Fermi function’’
  0
  #size of sp4 array in pixels
  234 232
  #(x,y) pairs describing bounding planes of the polygon
  -20 0
  0 -20
  ```

```
0 20
20 0
```

This example gives a square with sharp edges.

- `2dppmtosp4 <ppm-filename>`

  Converts an 8-bit ppm to a data array. Useful for hand made constraints and testing the algorithms.

- `2dquickcrop <in> <out> <x0> <y0>`

  Crops the array into the largest square possible with center `<x0> <y0>`.

- `2dreprod <ar1> <ar2>`

  Calculates the summed difference squared of all elements in the two arrays, normalized by the total squared energy in first. The arrays are centered by calculating the Fourier transform, removing the linear term from the phase, and back transforming. A check for the twin image is performed.

- `2dsp4bin <infile> <outfile> <x scaling> <y scaling>`

  Bins the data array by the scaling factor provided.

- `2dsp4toascii <x0> <y0> <x1> <y1> <infile.sp4>`
  `<outfile.text>`

  Dumps the requested portion of a data array into the output file as ASCII text. Each line has the row, column, amplitude, and phase of the pixel.

- `2dsp4tocon <input> <output> <threshold>`

  Takes a data array and reduces it to a 1-bit (black and white) data file suitable for use as a support constraint. The threshold is not a compared to each pixel, rather the portion of a row between the first and last occurrence of a pixel above the threshold is recorded. The same procedure is done for each column and the two lists are compared. If a point is on both lists, the corresponding pixel in the output is high. Next if a pixel is within a predefined distance (currently 3 px) to a pixel on both lists, it is set high. Otherwise, the pixel value is zero. The second step can help to make the output continuous.

- `2dspetosp4 <spe-filename> <outfile>`

  Converts the Roper Scientific data format (spe) to the sp4 data array. `<outfile>` is a string to which the frame number of a particular slice will be appended.

- `2dsymmtest2: 2dsymmtest2 <infile.sp4> <outfile.sp4> <X> <Y> where <X> and <Y> are the inversion point.`

  Calculates a correlation coefficient and a summed squared difference of pixels located centrosymmetrically in the image. This is used to test and symmetrize data that is very nearly symmetric. Normally, several inversion points will be tried and the one with the highest correlation coefficient (lowest sum of squared differences) will be used in symmetrizing the data. The symmetrized file is calculated and saved as `<outfile.sp4>`.

- `2dto3d <config file>`

  Converts a group of 2D data arrays into a 3D data array. The configuration file has the form:

  ```
  #new array name
  3darray
  #number of 2d slices
  5
  #2d arrays
  blank
  sq
  sq
  sq
  blank
  ```

  This would create a 3D array from 5 slices. In this case, it would perhaps be a support array where `sq` is a 2D filled square and `blank` is a zero filled array. This program is also used to stack the 2D data arrays recovered from the spe data format for use as the Fourier modulus constraint.

- `3dblanknth <in.sp4> <out.sp4> <n>`

  An early program designed as a rough test of simulations. It sets the amplitudes of every nth slice from a 3D array to test the requirements on oversampling in the third direction.

- `3drmnth <in.sp4> <out.sp4> <n>`

  An early program designed as a rough test of simulations. It removes every nth slice from a 3D array to test the requirements on oversampling in the third direction.

- `3dblur <in.sp4> <out.sp4> <alpha>`

  Inappropriately named program to multiply the input array by a Gaussian, in analogy to **`2dgfilter`**.

- `3dcmplxsqrt <in.sp4> <out.sp4>`

  Calculates the square root of the magnitude of the value in a pixel and gives it zero phase.

- `3dcrop <in> <out> <w> <h> <d> <x0> <y0> <z0> <dx/dy> <dx/dz> <dy/dz>`

  Crops a 3D data array. `<x0> <y0> <z0>` is the corner of the array and output size is given by `<w> <h> <d>`. The last three parameters allow a parallelepiped to cropped out rather than a cubic shape. This is useful when the 3D image is obtained by changing the incident X-ray beam energy, because in this case the CXD pattern moves on the detector.

- `3dcroppad: 3dcroppad <N-x> <N-y> <N-z> <infile.sp4> <outfile.sp4>`

  Performs the same function in 3D that **`2dcroppad2`** with mode 3 does in 2D. That is, puts the input array into a larger array and fills the new area with some random amplitude derived from the corner of the 3D array and a small predefined random deviation.

- `3dinvert <infile.sp4> <outfile.sp4>`

  Moves the center of the array to the corners. Used for preparing the Fourier modulus constraint. See **`3dinvert2`**

- `3dinvert2 <infile.sp4> <outfile.sp4>`

  Unwraps the FFT of a 3D array. Works in analogy to **`2dinvert`**. In preparing a 3D array for fitting, the array is normally complex by the time one wishes to move the Fourier modulus constraint to the corners,

but this program should not be used for this application, since half of the pixels will then possess negative amplitude and will be ignored for the purposes of applying the constraint in 3dmultifit2.

- `3dmultifit2: 3dmultifit2 <configfile>`

  An extension of **`2dmultifit`** to 3D. The configuration file is identical, but the mode numbers are not. Modes are documented in the source code.

- `3dpoly <data-file> <output-file>`

  Works analogously to **`2dpoly`**. This is not heavily used since it is normally necessary to use the 2D slices to determine an appropriate constraint, but it is useful for simulating diffraction patterns.

- `3dreduce <in> <out> <factor>`

  Bins a data array in the plane of the detector, but leaves the third dimension alone.

- `3dslice <in> <out> <x0> <y0> <z0>`

  Slices a 3D array for viewing. `<x0>` and `<y0>` should always be 1, and `<z0>` the number of the slice to remove from the 3D array. `3dreprod <ar1> <ar2>`Calculate a summed squared difference between two 3D arrays. Works by calculating the center of mass, so it is less effective than **`2dreprod`**. `3dnormreprod <ar1> <ar2>`Calculate a summed squared difference between two 3D arrays. Assumes arrays are centered, so it should be used in conjunction with **`3dcenter`**. Also renormalizes the arrays, if necessary. `3dcenter <in> <out>`Creates a centered array from the input by applying an FFT, removing a linear phase term, and back transforming.

- `2dsp4to16bbin <input.sp4> <basename>`

  Converts a 2D sp4 data array into a binary dump of 16-bit range.

- `3dsp4to8bbin <input.sp4> <basename>`

  Converts a 3D sp4 data array into a binary dump of 8-bit range.

- `add2sp4 <in1> <in2> <out>`

  Adds to sp4 arrays together.

- `addphases <infile.sp4> <phases.sp4> <outfile.sp4>`

  Replaces the phases of the first array with those of the second.

- `algoexplore <ppm> <startx> <starty> <algorithm>`
  `<iterations> [<beta> <gamma1> <gamma2>]`

  Program used to illustrate the dependence on the shape of the constraint sets of the algorithms. A color ppm is given which the program takes to understand as containing up to 3 sets. A set is composed of all points with a value of 255 in the red, green, or blue channel. For example, the intersection of the red and green sets would be yellow in the image. The use specifies a starting point in the image, the algorithm, the iterations, and any parameters the algorithms may take. At present ER and DM are supported.

- `alignph <sp4 file> <sp4 file to align> <output sp4 file>`

  Attempts to remove the constant phase offset between to data files.

- `bmstpcorrect2 <config file>`

  Program used to merge a short exposure of the center of a CXD pattern with a long exposure of fringes of the pattern. It requires a configuration file:

  ```
  #centerless array
  slow
  #array with center
  fast
  #output
  slow+fast
  #center of beamstop
  120 120
  #inner radius
  40
  #outer radius
  80
  ```

  The inner and outer radii are used to calculated the standard deviations of each array and smoothly merge the two together. WORKS IN 2D ONLY.

- `calctotamp in.sp4`

Calculates the sum of the magnitude of the amplitudes of a data array.

- `dofft <isign> <infile.sp4> <outfile.sp4>`

  Calculates the FFT of a data array. `<isign>` is the sign on the argument of the exponential in the transform.

- `dosqrt <infile.sp4> <outfile.sp4>`

  Calculates the square root of each pixel in an array. Only works on real arrays.

- `extractphases <in> <out>`

  Takes the phase information from the input array and writes it to a new array.

- `introducephases <in.amp> <in.phase> <out>`

  Creates an array with the amplitudes of one input array and the phases of the other.

- `flatten <config file>`

  Performs a background subtraction. The configuration file looks like this:

```
#choose zero rather than absolute value (1)
#for values less than threshold
0
#background filename
back
#data filename
58-40
#filename for background subtracted data
b58-40
#the data and background arrays are the same size
1 1 1 1
#multiply each pixel by 1
1
#set threshold at zero
0
```

  For a given threshold value, the program will either set such values to zero or to their absolute value. If the background file is larger than

the data file, one can give the section of the background array that corresponds to the data array. Finally, the background subtraction is performed by multiplying the value of a background pixel by the scaling factor and subtracting it from the data. If the resulting value is below the threshold it is modified as discussed above.

WORKS IN 2D ONLY.

- `flatten2 <config file>`

  Performs a background subtraction in the same way as **`flatten`** except that it attempts to correct the scaling factor. To do this, a background subtraction, up to application of the threshold condition is performed. Then, a histogram of the background and subtracted data array is calculated. If the first peak of the subtracted data is not centered about zero, a second subtraction (possibly an addition) is performed to move the peak to zero.

- `logscale <infile.sp4> <outfile.sp4>`

  Takes a data array and writes a real array whose elements are the logarithm base 10 of the squared value of each element in the original array.

- `readheader <sp4>`

  Reads the header information from a data array.

- `realtocomplex <in-file> <out-file> (where the files are binary Sp4Array's`

  Converts a real data array to a zero phase complex one.

- `removephases <cmplx-array> <real-array>`

  Removes all phase information from a complex array, yielding a real one.

- `rotate <data-file> <rotation about x> <rotation about y> <output-file>`

  A quick program to rotate a set of planes contained in a text file given to **`simpoly`**. Using the Euler angles. For example, if one were interested in an face centered cubic crystal, it might be interesting to

create an expected shape. In this case, a truncated octagon, composed of eight {111} planes and six {200}, and rotating it to see the FFT corresponding to different projections.

- `simpoly <data-file> <output-file>`

  Simulates the projection of a 3D polyhedron onto the $x - y$ plane. Its configuration is simply a list of intercepts, 3 for each plane, one set per line.

- `simpolyz <data-file> <output-file>`

  Same as simpoly, except that it accounts for a phase offset, as would be seen due to inverting a 2D CXD pattern collected off the Bragg condition.

- `sp4multiply <first> <second> <product>`

  Calculates the product of two data arrays.

- `subtrphases <infile.sp4> <frfile.sp4> <outfile.sp4>`

  Subtracts the phases of two data arrays.

- `sp4hist <input.sp4> <output> <min> <max> <bin>`

  Calculates the histogram of a data array and writes it to a text file. Minimum and maximum values, as well as, the bin size are user selected.

B.3 Example of a 3D Fit

This section gives an example of the procedure used to retrieve the phase of a 3D CXD pattern. A great deal of output from the programs has been omitted for brevity.

1 Extract a background image from an spe data file.

```
$ 2dspetosp4 0616-4.spe back
Read exposure time of 1.500000 seconds
Read xdim of 700 pixels
Read ydim of 420 pixels
Read datatype of 1
Read numframes of 1
Read numaccums of 20
```

2 Extract CXD data from an spe data file

```
$ 2dspetosp4 0616-2.spe au0616-2
```

3 Optionally perform a background subtraction.

```
$ flatten2 flconf
```

4 Plot the center-most slice and examine it with a viewing application.

```
$ 2dplot 0 16-2.11
2dplot:  found max and min are 932255 and 1427
```

5 It's sometimes easier to deal with a log scale image, so take the logarithm.

```
$ logscale 16-2.11 l16-2.11
```

6 Plot the log scaled data.

```
$ 2dplot 0 l16-2.11
```

7 From the image, find the approximate center and then search for the pixel which minimizes(maximizes) `chi squared`
(`Correlation coefficient`).

```
$ 2dsymmtest2 16-2.11 symm 312 256
nn[2] 624 nn[1] 512
Input array size 700 420; Output array size 624 328
Correlation coefficient = 0.000169193
chi squared = 0.047953
.
.
.
$ 2dsymmtest2 16-2.11 symm 311 256
nn[2] 622 nn[1] 512
Input array size 700 420; Output array size 622 328
Correlation coefficient = 0.000164999
chi squared = 0.052718
.
.
.
```

```
$ 2dsymmtest2 16-2.11 symm 312 257
nn[2] 624 nn[1] 514
Input array size 700 420; Output array size 624 326
Correlation coefficient = 0.00017071
chi squared = 0.055537
```

8 Using the bash shell, crop all data files about the center of the center-most slice.

```
$ for i in $(seq 1 30);do 2dquickcrop 16-2.$i c16-2.$i
312 256;done
```

9 Using the bash shell, pad all slices.

```
$ for i in $(seq 1 30);do 2dcroppad2 440 440 3 16-2.$i
c16-2.$i;done
```

10 Calculate the square root of the intensity for use as the Fourier Modulus constraint.

```
$ for i in $(seq 1 30);do dosqrt c16-2.$i ac16-2.$i;done
```

11 Stack the 2D slices into a 3D array.

```
$ 2dto3d 3dkey
```

12 Convert from a real data array to a complex one.

```
$ realtocomplex ac0616.2 ac0616.2
```

13 Wrap the data so that the center-most point is in the first pixel.

```
$ 3dinvert ac0616.2 ac0616.2
```

14 Create a 2D box to use a support constraint. Here, the size is 90 px by 120 px.

```
$ 2dpoly key sq
Plane 1, 0:  4.50e+01 Plane 2, 0:  0.00e+00
Plane 1, 1:  0.00e+00 Plane 2, 1:  6.00e+01
Plane 1, 2:  0.00e+00 Plane 2, 2:  -5.90e+01
Plane 1, 3:  -4.40e+01 Plane 2, 3:  0.00e+00
```

15 Create a blank array, since the object is not supposed to fill the array in the third direction.

```
$ 2dpoly key blank
```

16 From the 2D slices just written, create a 3D support constraint array.

```
$ 2dto3d 3dsup
```

17 Pass a configuration file containing the fitting procedure and the names of the constraint arrays just written to the fitting program.

```
$ 3dmultifit2 conf
```

18 Since the support may be too large, move the best real-space reconstruction to the center of the array.

```
$ 3dcenter sq_20_90_120.001.realfit creal1
```

19 Also move the second best real-space reconstruction to the center of the array.

```
$ 3dcenter sq_20_90_120.002.realfit creal2
```

20 Calculate the irreproducibility, ξ_1^2.

```
$ 3dnormreprod creal1 creal2
res = 1.47146e+08, tot = 1.4808e+10
Error metric = 0.00993697
```

21 Using the bash shell, extract 2D slices from the 3D array.

```
$ for i in $(seq 1 30); do 3dslice
sq_20_90_120.001.realfit real1-$i 1 1 $i; done
```

22 Using the bash shell, plot each 2D slice of the best real-space fit.

```
$ for i in $(seq 1 30); do 2dplot 0 real1-$i; done
minamp = 0.000599118, maxamp = 67.4347,
minph = -3.14159, maxph = 3.14159
amprng = 67.4341, phrng = 6.28319
junk = 255
.
```

.

.

23 Using the minimum and maximum from the plotting program's output, replot all 2D slices with the same range for comparison.

```
$ 2dmanplot real1-$i 0 3305
```

24 Unwrap the reciprocal-space best fit.

```
$ 3dinvert2 ac0616.2.001.recipfit recip1
```

25 Calculate the log scale array.

```
$ logscale recip1 lrecip1
```

26 Using the bash shell, extract 2D slices from the 3D array.

```
$ for i in $(seq 1 30); 3dslice lrecip1 lrec1-$i 0 0 $i
; done
```

27 Using the bash shell, plot each 2D slice of the best reciprocal-space fit.

```
$ for i in $(seq 1 30); do 2dplot 0 lrec1-$i
; done
```

References

[1] Advanced Photon Source list of parameters. Argonne National Laboratory Technical Bulletin ANL/APS/TB-26, 1996.

[2] Keep the noise down! low noise: an integral part of high-performance CCD camera systems. Roper Scientific Technical Bulletin 4, 1999.

[3] Direct detection of X-rays (4 keV to $\sim$ 20 keV) using detectors based on deep-depletion CCD technology. Roper Scientific Technical Bulletin 2, 2000.

[4] Grading scientific CCD sensors. Roper Scientific Technical Bulletin 10, 2000.

[5] G. B. Alers, K. S. Krisch, D. Monroe, B. E. Weir, and A. M. Chang. Tunneling current noise in thin gate oxides. *Appl. Phys. Lett.*, 69:2885–2887, 1996.

[6] R.C. Alig, S. Bloom, and C.W. Struck. Scattering by ionization and phonon emission in semiconductors. *Phys. Rev. B*, 22:5565–5582, 1980.

[7] Jens Als-Nielsen and Des McMorrow. *Elements of Modern X-ray Physics*. Wiley, 2001.

[8] Neil W. Ashcroft and N. David Mermin. *Solid State Physics*. Harcourt Brace, college edition, 1976.

[9] R. Barakat and G. Newsam. Necessary conditions for a unique solution to two-dimensional phase recovery. *J. Math. Phys.*, 25:3190–3193, 1984.

[10] S. Basu and Y. Bresler. Uniqueness of tomography with unknown view angles. *IEEE Transactions on Image Processing*, 9:1094–1106, 2000.

[11] R.H.T. Bates. Fourier phase problems are uniquely solvable in more than one dimension. I: Underlying theory. *Optik*, 61:247–262, 1982.

[12] H.H. Bauschke, P.L. Combettes, and D.R. Luke. On the structure of some phase retrieval algorithms. *IEEE ICIP*, pages II841–II844, 2002.

[13] H.H. Bauschke, P.L. Combettes, and D.R. Luke. Phase retrieval, error reduction algorithm, and Fienup variants: a view from convex optimization. *J. Opt. Soc. Am. A*, pages 1334–1345, 2002.

[14] Brett Borden. Mathematical problems in radar inverse scattering. *Inverse Problems*, 18:R1–R28, 2002.

[15] M. Born and E. Wolf. *Principles of Optics: Electromagnetic Theory of Propagation, Interference and Diffraction of Light.* Cambridge Univ. Press, 7th edition, 1999.

[16] B. J. Brames. Unique phase retrieval with explicit support information. *Opt. Lett.*, 11:61–63, 1986.

[17] Yu.M. Bruck and L.G. Sodin. On the ambiguity of the image reconstruction problem. *Optics Communications*, 30:304–308, 1979.

[18] W. R. Busing and H. A. Levy. Angle calculations for 3- and 4-circle X-ray and neutron diffractometers. *Acta Cryst.*, 22:457–464, 1967.

[19] CHESS. Structure factor & darwin width calculations. http://www.chess.cornell.edu/Operations/Computing/X-Ray_Calcs/darwin_width.htm.

[20] W. Cochron. Relations between phases of structure factors. *Acta Cryst.*, 8:473–478, 1955.

[21] David Colton, Joy Coyle, and Peter Monk. Recent developments in inverse acoustic scattering theory. *SIAM Review*, 42:369–414, 2000.

[22] David Colton, Klaus Giebermann, and Peter Monk. Regularized sampling method for solving three-dimensional inverse scattering problems. *SIAM Journal of Scientific Computing*, 6:2316–2330, 2000.

[23] P.L. Combettes. Hilbertian convex feasibility problem: Convergence of projection methods. *Appl. Math. Opt.*, 35:311–330, 1997.

[24] T. R. Crimmins and J. R. Fienup. Uniqueness of phase retrieval for functions with sufficiently disconnected support. *J. Opt. Soc. Am.*, 73:218–221, 1983.

[25] R. J. Dejus, B. Lai, E. R. Moog, and E. Gluskin. Undulator A characteristics and specifications: enhanced capabilities. Argonne National Laboratory Technical Bulletin ANL/APS/TB-17, 1994.

[26] R. J. Dejus, I. B. Vasserman, S. Sasaki, and E. R. Moog. Undulator A magnetic properties and spectral performance. Argonne National Laboratory Technical Bulletin ANL/APS/TB-45, 2002.

[27] E. Dufresne, R. Brüning, M. Sutton, B. Rodricks, and G.B Stephenson. A statistical technique for characterizing X-ray position-sensitive detectors. *Nuc. Inst. & Meth. in Phys. Res. A*, 364:380–393, 1995.

[28] V. Elser. Phase retrieval by iterated projections. *J. Opt. Soc. Am. A*, 20:40–55, 2003.

[29] U. Fano. Ionization yield of radiations. ii. the fluctuations of the number of ions. *Phys. Rev.*, 72:26–29, 1947.

[30] John A. Fawcett. Geocoding sidescan sonar data - an inverse problem. *Oceans Conference Record (IEEE)*, 1:11–15, 2000.

[31] J. R. Fienup. Space object imaging through the turbulent atmosphere. *Opt. Eng.*, 18:529–534, 1979.

[32] J.R. Fienup. Reconstruction of an object from the modulus of its Fourier transform. *Optics Letters*, 3:27–29, 1978.

[33] J.R. Fienup. Phase retrieval algorithms: a comparison. *Applied Optics*, 21:2758–2769, 1982.

[34] J.R. Fienup. Reconstruction of a complex-valued object from the modulus of its Fourier transform using a support constraint. *J. Opt. Soc. Am. A.*, 4:118–123, 1987.

[35] J.R. Fienup, J.C. Marron, T.J. Schulz, and J.H. Seldin. Hubble Space Telescope characterized by using phase-retrieval algorithms. *Appl. Opt.*, 32:1747–1767, 1993.

[36] J.R. Fienup and C.C. Wackerman. Phase-retrieval staganation problems and solutions. *J. Opt. Soc. Am. A.*, 3:1897–1907, 1986.

[37] R.W. Gerchberg and W.O. Saxton. A practical algorithm for the determination of phase from image and diffraction plane pictures. *Optik*, 35:237–246, 1972.

[38] H.W. Hayden, W.G. Moffat, and J. Wulff. *Structure and Properties of Materials III.* Wiley, New York, 1965.

[39] M.H. Hayes. The reconstruction of a multidimensional sequence from the phase or magnitude of its fourier transform. *IEEE Transactions on acoustics, speech, and signal processing*, ASSP-30:140–154, 1982.

[40] J.C. Heyraud and J.J. Metois. Equilibrium shape of gold crystallites on a graphite cleavage surface: surface energies and interfacial energy. *Acta Metallurgica*, 28:1789–1797, 1980.

[41] O. Hignette, P. Cloetens, W.-K. Lee, W Ludwig, and G. Rostaing. Hard X-ray microscopy with reflecting mirrors status and perspectives of the ESRF technology. *Journal De Phsysique IV: JP*, 104:231–234, 2003.

[42] M. R. Howells, H. Chapman, S. Hau-Riege, H. He, S. Marchesnini, J. Spence, and U. Weierstall. X-ray microscopy by phase-retrieval methods at the advanced light source. *H. De Phys. IV: JP*, 104:557–561, 2003.

[43] A. M. Huiser and P. van Toorn. Ambiguity of the phase-reconstruction problem. *Opt. Lett.*, 5:499–501, 1980.

[44] Gene E. Ice, Eliot D. Specht, Jonathan Z. Tishler, Ali Khounsary, Lahsen Assoufid, and Chian Liu. At the limit of nondispersive micro and nanofocusing mirror optics. *Proceedings of the SPIE*, 5347:1–8, 2004.

[45] J.D. Jackson. *Classical Electrodynamics.* Wiley, 3th edition, 1998.

[46] James R. Janesick. *Scientific Charge-Coupled Devices.* SPIE Press, 2001.

[47] A. C. Kak and M. Slaney. *Principles of Computerized Tomographic Imaging.* Society of Industrial and Applied Mathematics, 2001.

[48] I.K. Robinson C.A. Kenney-Benson and I.A. Vartanyants. Sources of decoherence in beamline optics. *Physica B: Condensed Matter*, 336:56–62, 2003.

[49] V. A. Kochnev. Adaptive methods for the solution of geophysical inverse problems (reasoning for the use, review of results and prospects). *Modelling, Measurement & Control C: Energetics, Chemistry, Earth,Environmental & Biomedical Problems*, 42:47–54, 1994.

[50] C. Kostov, J. Barsch, K. Deal, S. Grion, and L. Walker. Good seismic data in bad weather. *Proceedings of the Annual Offshore Technology Conference*, 1:123–126, 2000.

[51] Patrica K. lamm. Inverse problems and ill-posedness. *Proceedings of the 1st International Conference on Inverse Problems in Engineering*, pages 1–10, 1993.

[52] A. Levi and H. Stark. Image restoration by the method of generalized projections with application to the restoration from magnitude. *J. Opt. Soc Am. A*, 1:932–943, 1984.

[53] F. Livet, F. Bley, J. Mainville, R. Caudron, S.G.J. Mochrie, E. Geissler, G. Dolino, D. Abernathy, G. Grübel, and M. Sutton. Using direct illumination CCDs as high-resolution area detectors for X-ray scattering. *Nuc. Inst. & Meth. in Phys. Res. A*, 451:596–609, 2000.

[54] S. Marchesini. Benchmarking iterative projection algorithms for phase retrieval. arXiv:physics/0404091, 2004.

[55] S. Marchesni, H. N. Chapman, S. P. Hau-Riege, R. A. London, A. Szoke, H. He, M. R. Howells, H. Padmore, R. Rosen, J. C. H. Spence, and U. Weierstall. Coherent X-ray diffractive imaging: applications and limitations. *Opt. Express*, 11:2344–2353, 2003.

[56] J. Miao, P. Charalambous, J. Kirz, and D. Sayre. Extending the methodology of X-ray crystallography to allow imaging of micrometre-sized non-crystalline specimens. *Nature*, 400:342–344, 1999.

[57] J. Miao, T. Ishikawa, B. Johnson, E.H. Anderson, B. Lai, and K.O. Hodgson. High resolution 3D X-ray diffraction microscopy. *Phys. Rev. Lett.*, 89:088303, 2002.

[58] J. Miao, D. Sayre, and H. N. Chapman. Phase retrieval from the magnitude of the Fourier transforms of nonperiodic objects. *J. Opt. Soc. Am. A.*, 15:1662–1669, 1998.

[59] R. P. Millane. Phase retrieval in crystallography and optics. *J. Opt. Soc. Am. A*, 7:394–411, 1990.

[60] R. P. Millane and W. J. Stroud. Reconstructing symmetric images from their undersampled Fourier intensities. *J. Opt. Soc. Am. A.*, 14:568–579, 1997.

[61] R.P. Millane. Multidimensional phase problems. *J. Opt. Soc. Am. A.*, 13:725–734, 1996.

[62] Allen F. Moench. Computation of type curves for flow to partially penetrating wells in water-table aquifers. *Ground Water*, 31:966–971, 1993.

[63] John E. Molyneux. Ground penetrating radar tomography. *IEEE International Conference on Acoustics, Speech and Signal Processing - Proceedings*, 5:2813–2816, 1995.

[64] H. E. Moses and R. T. Prosser. Phases of complex functions from the amplitudes of the functions and the amplitudes of the Fourier and Mellin transforms. *J. Opt. Soc. Am.*, 73:1451–1454, 1983.

[65] M. Nieto-Vesperinas and J. C. Dainty. A note on Eisenstein's irreproducibility criterion for two-dimensional sampled objects. *Opt. Comm.*, 54:333–, 1985.

[66] A. Yu. Nikulin. Uniqueness of the complex diffraction amplitude in x-ray Bragg diffraction. *Phys. Rev. B*, 57:11178, 1998.

[67] A. Yu. Nikulin, J.R. Davis, and N.T. Jones. X-ray phase-amplitude contrast mapping of single-crystal alloys near the absorption edge of the alloy impurity. *J. Appl. Phys.*, 84:4815–4821, 1998.

[68] K. A. Nugent, A. G. Peele, H. N. Chapman, and A. P. Mancuso. Unique phase recovery for nonperiodic objects. *Phys. Rev. Lett.*, 91:203902, 2003.

[69] J. Baruchel P. Rejmánková-Pernot, P. Cloetens and J.-P. Guigay. Phase retrieval by combined bragg and fresnel X-ray diffraction imaging. *Phys. Rev. Lett.*, 81:3435–3438, 1998.

[70] Franco Pettenati and Livio Sirovich. Tests of source-parameter inversion of the u.s. geological survey intensities of the whittier narrows 1987 earthquake. *Bulletin of the Seismological Society of America*, 93:47–60, 2003.

[71] John A. Pitney. *Coherent X-ray diffraction*. PhD thesis, University of Illinois at Urbana-Champaign, 2000.

[72] William H. Press, Brian P. Flannery, Saul A. Teukolsky, and William T. Vetterling. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, 1992.

[73] E. Prince, editor. *International tables for crystallography.* Kluwer, 2004.

[74] I. K. Robinson. Crystal truncation rods and surface roughness. *Phys. Rev. B*, 33:3830–3836, 1986.

[75] I. K. Robinson. Four-circle diffractometry for surfaces. *Rev. Sci. Instrum.*, 60:1541–1544, 1989.

[76] I.K. Robinson and I.A. Vartanyants. Use of coherent X-ray diffraction to map strain fields in nanocrystals. *Appl. Surf. Sci.*, 182:97–100, 2001.

[77] I.K. Robinson, I.A. Vartanyants, G.J. Williams, M.A. Pfeifer, and J.A. Pitney. Reconstruction of the shapes of gold nanocrystals using coherent X-ray diffraction. *Phys. Rev. Lett.*, 87:195505, 2001.

[78] J. L. C. Sanz and T. S. Huang. Unique reconstruction of a band limited multidimensional signal from its phase or magnitude. *J. Opt. Soc. Am.*, 73:1446–1450, 1983.

[79] D. Sayre. Some implications of a theory due to Shannon. *Acta Cryst.*, 5:843, 1952.

[80] D. Sayre. The squaring method: a new method for phase determination. *Acta Cryst.*, 5:60–65, 1952.

[81] D. Sayre, H. N. Chapman, and J. Miao. On the extendibility of X-ray crystallography to noncrystals. *Acta Cryst.*, A54:232–239, 1998.

[82] C. G. Schroer, M. Kuhlmann, U. T. Hunger, T. F. Günzler, O. Kurapova, S. Feste, F. Frehse, B. Lengeler, M. Drakopoulos, A. Somogyi, A. S. Simionovici, A. Snigirev, I. Snigireva, C. Schug, and W. H. Schröder. Nanofocusing parabolic refractive x-ray lenses. *Appl. Phys. Lett.*, 82:1485–1487, 2003.

[83] V.F. Sears and S.A. Shelley. Debye-waller factor for elemental crystals. *Acta Cryst.*, A47:441–446, 1991.

[84] J.H. Seldin and J.R. Fienup. Numerical investigation of the uniqueness of phase retrieval. *J. Opt. Soc. Am. A.*, 7:412–427, 1990.

[85] S. K. Sinha, M. Tolan, and A. Gibaud. Effects of partial coherence on the scattering of x rays by matter. *Phys. Rev. B*, 57:2740–2758, 1998.

[86] Advanced Photon Source. APS Introduction. http://www.aps.anl.gov.

[87] J. C. H. Spence, M. Howells, L.D. Marks, and J. Miao. Lenless imaging: A workshop on new approaches to the phase problem for non-periodic objects. *Ultramicroscopy*, 90:1–6, 2001.

[88] A. Szoke. Holographic microscopy with a complicated reference. *J. Image. Sci. Technol.*, 41:332–341, 1997.

[89] M. Thoms. The dynamic range of X-ray imaging with image plates. *Nucl. Instr. and Meth. in Phys. Res. A*, 389:437–440, 1997.

[90] Johan R. Valstar, Dennis B. McLaughlin, Chris B. M. Te Streot, and Frans V. Van Geer. A representer-based inverse method for groundwater flow and transport applications. *Water Resources Research*, 40:W051161–W0511612, 2004.

[91] Friso van der Veen and Franz Pfeiffer. Coherent x-ray scattering. *Journal of Physics: Condensed Matter*, 16:5003–5030, 2004.

[92] P. van Toorn, A. H. Greenaway, and A. M. J. Huiser. Phaseless object reconstruction. *Opt. Acta*, 31:767–774, 1984.

[93] I. A. Vartanyants and I. K. Robinson. Origins of decoherence in coherent X-ray diffraction experiments. *Opt. Comm.*, 222:29–50, 2003.

[94] I.A. Vartanyants, J.A. Pitney, J.L. Libbert, and I.K. Robinson. Reconstruction of surface morphology from coherent X-ray reflectivity. *Phys. Rev. B*, 55:13193–13202, 1997.

[95] I.A. Vartanyants and I.K. Robinson. Partial coherence effects on the imaging of small crystals using coherent X-ray diffraction. *J. Phys. Cond. Mat.*, 13:10593–10611, 2001.

[96] I.A. Vartanyants and I.K. Robinson. Imaging of quantum array structures with coherent and partially coherent diffraction. *Journal of Synchrotron Radiation*, 10:409–415, 2003.

[97] A. Walther. The question of phase retrieval in optics. *Opt. Acta*, 10:41–49, 1963.

[98] B.E. Warren. *X-ray Diffraction*. Dover Publications, Inc., 1990.

[99] G.J. Williams, M.A. Pfeifer, I.A. Vartanyants, and I.K. Robinson. Three-dimensional imaging of microstructure in Au nanocrystals. *Phys. Rev. Lett.*, 90:175501, 2003.

[100] W. Yun, B. Lai, Z. Cai, J. Maser, D. Legnini, E. Gluskin, Z. Chen, A. Krasnoperova, Y. Vladimirsky, F. Cerrina, E. Di Fabrizio, and M. Gentili. Nanometer focusing of hard x rays by phase zone plates. *Rev. Sci. Instrum.*, 70:2238–2241, 1999.

[101] Z. M. Zuo, I. Vartanyants, M. Gao, and L. A. Nagshan. Atomic resolution imaging of a carbon nanotube from diffraction intensities. *Science*, 300:1419–1421, 2003.

Author's Biography

Garth Jonathan Williams was born August 6, 1975 in Mansfield, Ohio to Christine and Randall Williams. In August of 1993, he began undergraduate study at the University of Akron, Akron, Ohio. While at the University of Akron, he received support from the University Honors Program. He was graduated as a University Scholar with a B.S. in Physics and Pure Mathematics. In August of 1998, Garth began his graduate studies at the University of Illinois at Urbana-Champaign. While at the University of Illinois, served three semesters as a Teaching Assistant, two in an undergraduate Electromagnetism Lab and one in a Digital Electronics Lab. Garth joined the research group of Ian K. Robinson in June of 1999 as a Research Assistant.