Phase retrieval algorithms applied to the reconstruction of photonic quantum states

Mário Foganholi Fernandes

Presented august 2016

Universidade Federal de Minas Gerais

Instituto de Ciências Exatas – Departamento de Física

Phase retrieval algorithms applied to the reconstruction of photonic quantum states

Mário Foganholi Fernandes

Advisor: Leonardo Teixeira Neves

Co-advisor: Miguel Ángel Solís Prosser*

**Center for Optics and Photonics, Departamento de Física,*
Universidad de Concepción, Chile

Dissertação apresentada ao departamento de Física da Universidade Federal de Minas Gerais, para a obtenção do título de Mestre em Física.

The phrase *shoshin* means 'beginner's mind'. The goal of practice is always to keep our beginner's mind. In the beginner's mind, there are many possibilities; in the expert's mind there are few.

—Shunryu Suzuki-roshi

All life demands struggle. The very striving and hard work that we so constantly try to avoid is the major building block in the person we are today.

—Pope Paul VI

Acknowledgements

Eu gostaria de agradecer, primeiramente, ao meu orientador, Leonardo Neves, e ao meu co-orientador, Miguel Solís Prosser. Trabalhar com vocês tem sido incrivelmente enriquecedor, tanto pelo lado acadêmico quanto pelo pessoal, pois vocês me ensinam muito. Acho que o aspecto mais marcante e mais inspirador de trabalhar com vocês é a maneira como vocês fazem Física, não por vaidade ou para buscar ”natures” e ”sciences”, mas simplesmente porque Física é apaixonante. Tanto no laboratório quanto nas reuniões, o clima de trabalho é extremamente acolhedor e aberto, e eu não tenho como agradecer o suficiente a vocês.

Também sou grato aos meus professores da UFMG e da USP, em especial Rafael de Sá Freitas, Jorge deLyra, Manuel Robillota, Ana Regina Blak, Arnaldo Gammal, Henrique Barbosa, Philippe Gouffon, Osvaldo Pessoa, Jafferson Kamphorst, Ronald Dickman e Mario Mazzoni. Vocês fizeram muita diferença na minha formação, e me ensinaram não só Física e Matemática, mas formas de pensar.

Also, I want to thank my professors ans friends from CWRU, where I spent a wonderful year. In special, professors Michael Martens, Robin Snyder, Thomas Shutt, Daniel Akerib and Colin McLarty, for all your dedication in teaching. You have also made a huge difference in my education, both professionally and personally. And my friends Youngmin Park, Priscilla Ambrosia, Renato Marques, Larissa Ishikawa, Julián Rojo, Sheerin Sandhu, Heather Wojanowski, I hope you are all doing good and that we bump into each other at some point. I miss you so much.

Agradeço também ao pessoal da secretaria de pós-graduação e da biblioteca do DF da UFMG, por serem sempre atenciosos e por quebrarem tantos galhos. Em especial, quero agradecer a Shirley Maciel, por ser tão incrível como pessoa e como profissional. A biblioteca do DF é um lugar incrível, sempre com coisas legais expostas pra gente ver, com tantos livros interessantes, e sempre com um clima maravilhoso, que dá vontade de não sair nunca mais depois de entrar. Muito, muito, muito obrigado por toda a sua dedicação pela biblioteca, e venha sempre nos visitar, pois vamos sentir muitas saudades!

I am also very thankful to the people at Case Kung FU Club, specially sifu James van Doren. Também agradeço muito ao pessoal do Inst. de Kung Fu Shaolin, em especial sifu Marcos Serra, sifu Cláudio Pereira, Profa. Ângela Forninho e Profa. Isadora Gouvêa, e ao pessoal da Yamashi Yoshin Dojo, em especial sensei Davi Maciel e sensei Arthur Corrêa.

Além de artes marciais, vocês me ensinam a ser uma pessoa melhor, e eu certamente não teria chegado até aqui são se não fosse por vocês.

Também agradeço aos meus amigos da USP, em especial Baldi (que ainda tá devendo uma visita!), Raissa, André, Anderson, Júlio, Gubolin, Edson, Hideki, Luana, Nicholas, Alessandro, Gabriel, Beiço, Flávio, Helo, Capeta e Luis Guilherme, da UFMG, em especial Marcello, Lud, Mila, Caio, Thamires, Davi (irmão gêmeo!), Cobra, Tati, Diego, Alana, Julia, Denise, João, Gil, Tutu, Bluesão, Olímpio, Ludmila, Arthur, Regiley, Wilder, Wilmer, Leo e Timas. Também quero agradecer aos meus velhos amigos do Porto, especialmente Camali, Rodolfo, Ogeda, Lima, Bia, Carol e Larissa. Eu não tenho como dizer o quão sortudo eu me sinto de ter trombado com pessoas legais como vocês, sintam-se abraçados bem apertado.

Finalmente, agradeço à minha família, em especial a meus pais, Marizelda e Francisco, e a meus irmãos, Francisco e Beatriz. Nem sempre eu falo, mas eu morro de saudades de vocês aqui em BH. Amo vocês com todo meu coração.

A todos vocês, dedico esse trabalho com gratidão que não cabe em palavras. Muito obrigado mesmo, por tudo.

Contents

Abstract

Reconstructing a wavefront is a problem that appears in many areas of physics and engineering, and leads to many important applications. X-ray crystallography, electron microscopy, femtosecond laser temporal characterization, blind deconvolution of degraded images, and tomographic imaging are examples of areas that, if not directly involving the reconstruction of a wavefront, have benefited from the techniques developed to solve this problem. In general, the measurement devices are able to record the intensities of the wavefront, but not its phases; the numerical tools used to recover them thus became known as *phase retrieval algorithms*.

In this work we propose the use of these algorithms to reconstruct pure quantum states encoded into transverse spatial modes of single photons – the so-called *spatial qudits*. We made significant adaptations on the algorithms found in the literature in order to fit experimental features of this kind of encoding. The most striking of these was to magnify Fourier-plane amplitudes in order to compensate for the small range of sampled frequencies, leading to great improvements in the quality of the results.

To demonstrate this technique, we performed a proof-of-principle experiment with an optical beam mimicking spatial qudits states of dimensions $D = 2, 3, 4, 7$ and 9. After compensating for some experimental deviations, the recovered states presented fidelities frequently above 99% with respect to the target state, showing that the phase retrieval algorithms may be a useful tool for quantum state characterization.

Resumo

A reconstrução de frentes de onda é um problema que aparece em muitas áreas da física e da engenharia, e resulta em muitas aplicações importantes. Cristalografia de raios-x, microscopia eletrônica, caracterização temporal de lasers de femtosegundo, deconvolução cega de imagens degradadas e imageamento tomográfico são exemplos de áreas que, quando não envolvendo diretamente a reconstrução de uma frente de onda, se beneficiaram das técnicas desenvolvidas para resolver este problema. Em geral, os dispositivos de medição são capazes de obter as intensidades da frente de onda, mas não suas fases; as ferramentas numéricas utilizadas para recuperá-las ficaram portanto conhecidas como *algoritmos de recuperação de fase*.

Neste trabalho, nós propusemos o uso desses algoritmos para reconstruir estados quânticos puros codificados em modos espaciais transversais de fótons individuais – os chamados *qudits espaciais*. Nós fizemos adaptações significativas aos algoritmos encontrados na literatura, de modo a adequá-los às características experimentais desse tipo de codificação. A mais eminente dessas adaptações foi magnificar as amplitudes no plano de Fourier, de forma a compensar o pequeno intervalo de frequências medidas. Isso resultou em uma grande melhora na qualidade dos resultados.

Para demonstrar essa técnica, nós realizamos um experimento de prova de princípio com um feixe óptico que mimetizava estados de qudits espaciais com dimensões $D = 2, 3, 4, 7$ e 9. Depois de compensados certos desvios experimentais, os estados recuperados apresentaram fidelidades frequentemente acima de 99% com relação ao estado que se desejava preparar, mostrando que os algoritmos de recuperação de fase podem ser uma boa ferramenta para a caracterização de estados quânticos.

Introduction

When we study undulatory phenomena, we learn that waves have *phases*. But they are not evident in our everyday life, since when we hear or see we are actually interacting with *incoherent* waves, whose phase fluctuations are too big. For instance, these fluctuations make it impossible for *interference phenomena* to take place: how many times has the reader seen two people talk at the same time and the volume of their speech increase fourfold? Or their words cancel each other, with just silence remaining? In fact, the phases of a wave are also not easy to measure, as most detectors – and certainly all photodetectors – only record intensities. Nevertheless these phases exist, and both their control and measurement have been used in astonishing applications.

A wave is usually a cyclic phenomena, and the point of its cycle that the wave finds itself is called its *phase*. After a wave interacts with something else – a non homogeneous material, for example – each of its points might have evolved differently in their cycles, since each interacted with a different part of the material. The phases that a wave carries after such an interaction can actually tell us something about the interaction itself. How could one infer the phase of a wave then? One strategy would be to transform the phase information into intensity information, for example through an interference phenomenon, which could then be readily measured. After that, one could infer back the phases, and finally infer the characteristics of the interaction (or the material) by using some physical model.

Here we will study another such strategy that relies on the knowledge (total or partial) of both the wave's and its Fourier transform's amplitudes. In this strategy, in order to recover the phases one has to use numerical algorithms known as *phase retrieval algorithms*. There are two kinds of problems [1] that can be fit in this context. The first kind is that of the *reconstruction problems*, in which the phase of the wave carries some information one is interested in. Surface metrology is an example of such a task: imagine you have a surface and want to check it for very slight deformations; you could impinge a plane wave on that surface so that eventual deformations would delay or advance the wavefront. The mirrors of the James Webb Space Telescope (JWST), planned successor of Hubble telescope, have been checked for possible deformations that could hinder the images with this technique [2], demonstrating the astonishing precision that can be reached. Not less impressive is the use of these algorithms to align the different sections of its mirrors [3] after it is launched (the JWST will have a very large mirror composed of 18 sections that will be folded on top of each other during the launch, but should be unfolded after it reaches its solar orbit for the observations [4]).

The second kind is that of the *synthesis problems*, in which instead of desiring to recover phases based on intensity measurements, one wants to discover what phases should be *imposed* on a wave in order to give it some desired intensity profile. This has been used to control laser beam profiles in inertial confinement fusion [5], and to improve the quality of images made with holograms [6], for example. Phase retrieval algorithms can also be employed to solve this kind of problem.

In principle, every scientific area that involves coherent waves interacting with matter could benefit from phase retrieval techniques, as has been the case of electron microscopy [7] and x-ray crystallography [8].

In this work we propose the use of phase retrieval algorithms as a tool for characterizing pure quantum states encoded in transverse spatial modes of photons (spatial qudits); we do so by approaching state characterization as a phase reconstruction problem (of the kind described above). Spatial qudits have been receiving increasing attention because of its potential for applications in quantum information, quantum cryptography and fundamental tests of quantum mechanics with high-dimensional states. All of these involve the knowledge of the states at some level, for which the phase retrieval algorithms might be one more alternative.

We have organized this dissertation in the following manner: chapter 1 discusses the spatial qudits and how the phase retrieval problem arises when we use them; chapter 2 presents the phase retrieval algorithms themselves; chapter 3 outlines the specific adaptations we had to make in the standard algorithms in order to adapt them to our problem; chapter 4 introduces the experimental setup we used for preparing the spatial qudit states and also presents the results we obtained when reconstructing them with the algorithm; finally, chapter 5 brings the conclusions and future perspectives.

Chapter 1

Spatial qudits and the phase retrieval problem

Quantum information and quantum computing are fields that have been receiving increasing attention, both because of its technological applications and the point-of-view it offers for approaching fundamental questions [9]. For example, some interesting applications that have been devised are superdense coding [10], information teleport [11], quantum cryptography [12] and quantum speed-ups of algorithms [13]; an example of the fundamental questions that have been studied is the attempt to understand whether quantum mechanics can be formulated as a hidden-variable theory or not [14].

Another intriguing possibility that quantum computing offers is to use a quantum system to simulate another, as has been pointed by R. Feynman [15]. While it may be difficult to simulate the time evolution of a given quantum system in a classical computer, it might be possible to make another quantum system mimic this system of interest instead. This would be a valuable tool to understand systems too complicated to control in practice.

In the pursue of quantum computers, several implementations using different quantum systems have been studied. Quantum information and quantum computing protocols have been implemented with ion traps [16], superconducting circuits [17], quantum dots [18] and photons [19]. Photons are good candidates for these tasks since they are easily transported, both by free-propagation and optical fibers. They also have several degrees of freedom in which one can encode information, for instance the polarization [20], the angular momentum [21], the temporal profile [22] and the transverse position-momentum [23], on which we will be focusing from now on.

The transverse spatial modes have the versatility of easily allowing for high-dimensional states to be encoded, offering advantages for applications in cryptography [24], communication [25] and Bell-like experiments [26], for example. The states codified in this manner are called *spatial qudits*[1], and they have already been used in experiments of quantum algorithms [27], quantum games [28], quantum contextuality [29], simulation of decoherence [30] and generation of maximally-entangled pairs [23].

The ability to characterize states is important in all of these applications. In general, this is done through state tomography, which involves carrying out measurements with an informationally complete set of operators [31,32]. Because these sets comprise many operators, this technique becomes somewhat costly, specially for higher-dimensional states. In particular, several tomoghaphic schemes have been devised for spatial qudits [33–36].

In this chapter we will study how electromagnetic fields can behave similar to a D-level quantum system. We will see a scheme that can be used to emulate qudit states in classical beams as well as encode them in single photons. It will become clear that this scheme relies on the use of phases, so next we will discuss the problem of phase information loss that arises when we use photodetector arrays to detect the beam profile, and introduce the main idea behind the algorithms that can solve it. Finally, we introduce the fidelity between two states, which we will use later to quantify how good the results of the algorithms were.

1.1 Mimicking quantum states with an electromagnetic field

In this section we want to introduce an approach to mimic pure quantum states with a classical electromagnetic field.

A pure state of a quantum system is described as a vector in a D-dimensional Hilbert space $\mathcal{H}$ [37]. If we let $\{|n\rangle\}_{n=1}^{D}$ be a basis of $\mathcal{H}$ (orthonormal, for convenience), we can write any arbitrary state in this space as

$$|\psi\rangle = \sum_{n=1}^{D} c_n |n\rangle , \tag{1.1}$$

where the coefficients c_n are complex numbers that must satisfy $\sum_{n=1}^{D} |c_n|^2 = 1$ for the qudit state $|\psi\rangle$ to be normalized.

Using a scalar representation of an electromagnetic wave (which we are allowed to do, if we assume that the polarization of the field is fixed throughout the entire space), we can arrive at a set of fields that obey an expression similar to equation (1.1). In this sense, it is possible to mimic these states with electromagnetic waves.

[1] In quantum information, *qudits* are the D-dimensional analogues of the two-dimensional quantum bits (qubits).

Imagine we have an opaque screen with D rectangular slits, each having a very thin film that controls its complex transmission coefficient. That is, the film at each slit controls both the field amplitude that is transmitted and phase that it gains, as depicted in figure 1.1. Thus the transmission [38] function of this screen over the plane is given by

$$t(x,y) = \sum_{i=1}^{D} t_i e^{i\phi_i} \mathrm{rect}\left(\frac{x-x_i}{X}\right) \mathrm{rect}\left(\frac{y}{Y}\right), \tag{1.2}$$

where t_i is the real transmission coefficient, ϕ_i the phase gain and x_i the center of the ith slit; also, X ans Y are widths of the slits in the x and y directions, respectively, and rect is the rectangle function, given by

$$\mathrm{rect}(\eta) = \begin{cases} 1, & \text{if } \eta \in [-0.5, 0.5] \\ 0, & \text{otherwise} \end{cases}. \tag{1.3}$$

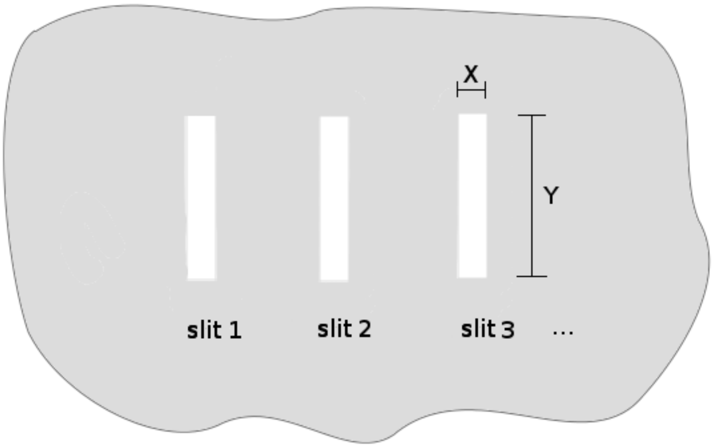


Figure 1.1: Screen with D slits; its transmission functions is given by equation (1.2).

Now if we impinge a monochromatic plane wave of frequency ω and amplitude E_0 on this screen, propagating in the z direction, the field immediately after the screen (supposed at $z = 0$) will be

$$\begin{aligned} E_t(x,y,t) &= t(x,y)E_0 e^{-i\omega t} && (1.4) \\ &= e^{-i\omega t} \sum_{i=1}^{D} t_i e^{i\phi_i} E_0 \, \mathrm{rect}\left(\frac{x-x_i}{X}\right) \mathrm{rect}\left(\frac{y}{Y}\right) \\ &= \sum_{i=1}^{D} t_i e^{i\phi_i} E_i(x,y,t). && (1.5) \end{aligned}$$

This field has the same form as equation (1.1), with the coefficients $t_i e^{i\phi_i}$ playing the roles of each c_i, and the single-slits fields

$$E_i(x,y,t) = E_0 e^{-i\omega t}\mathrm{rect}\left(\frac{x-x_i}{X}\right)\mathrm{rect}\left(\frac{y}{Y}\right) \tag{1.6}$$

playing the roles of the basis states $|i\rangle$. This shows that, by controlling the transmission coefficients of each slit (making $t_i = |c_i|$ and $\phi_i = \arg c_i$), we can mimic an arbitrary state $|\psi\rangle$.

Of course, equation (1.5) is an approximation. We did not use diffraction theory to find the transmitted field, we just used the approximation that it is given by the product of the transmission function and the incident wave. In fact, the vectorial theory of near field diffraction can be a quite hard subject. However, if we admit that the incident wavelength is much smaller than the dimensions of the slits, this is a good approximation.

Isomorphism between slit fields and quantum states

We have seen how an electromagnetic field can be similar to a qudit. Actually, we would like them to be so similar that any operation and any quantity defined for the quantum states can be also implemented or calculated for the slit fields. Mathematically, what assures this is possible is a relation called *isomorphism*.

Let us call the set of all possible slit fields $\mathcal{H}'$. What we are claiming is that the correspondence

$$\begin{aligned} h:\ \mathcal{H} &\longmapsto \mathcal{H}' \\ \sum_{i=1}^{D} c_i\,|i\rangle &\longmapsto \frac{1}{E_0}\sum_{i=1}^{D} t_i e^{i\phi_i} E_i, \end{aligned} \tag{1.7}$$

with $t_i = |c_i|$ and $\phi_i = \arg(c_i)$ is a isomorphism and thus preserves the structure of the Hilbert space of the possible states $|\psi\rangle$, so that the space $\mathcal{H}'$ of transmitted fields E_t is a faithful copy[2] of $\mathcal{H}$. This is to say that h satisfies the following properties:

(i) it is a bijection[3];

(ii) preserves linear combinations: $h(\alpha\psi_1 + \beta\psi_2) = \alpha h(\psi_1) + \beta h(\psi_2)$;

(iii) preserves inner products: $\langle h(\psi_1)|h(\psi_2)\rangle = \langle\psi_1|\psi_2\rangle$;

[2] We are claiming that h is an *isomorphism* between the Hilbert spaces; this is not surprising: one can check that $\mathcal{H}'$, which is generated by the slit fields E_i, is indeed a Hilbert space of D dimensions, just as $\mathcal{H}$, and remember that a mapping that takes an orthonormal basis in one finite-dimensional Hilbert space to another orthonormal basis in the other will be an isomorphism.

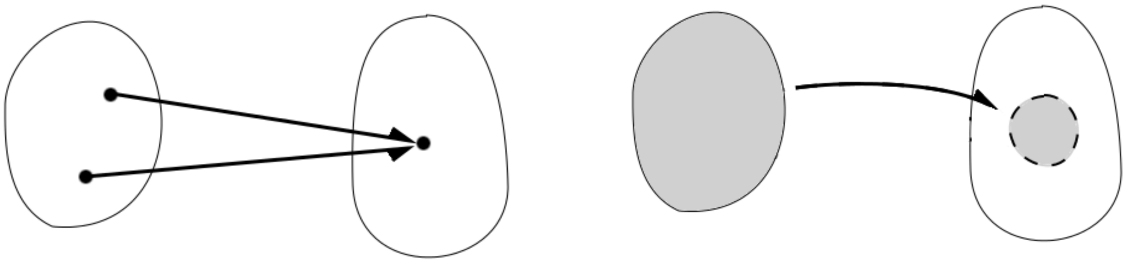
[3] An *isomorphism* is just a mapping that is both *injective* (no two elements are mapped to the same image) and *surjective* (the image of the domain covers all of the codomain)

which are all straightforward to check; for property (iii), it is useful to note that the slit fields are orthogonal: if we let $j \neq i$, then

$$\begin{aligned}\langle E_i|E_j\rangle &= \iint_{\mathcal{R}^2} E_i^*(x,y,t)E_j(x,y,t)dxdy \\ &= E_0^2 \underbrace{\left[\int_\infty^\infty \mathrm{rect}\left(\frac{x-x_i}{X}\right)\mathrm{rect}\left(\frac{x-x_j}{X}\right)dx\right]}_{0} \left[\int_\infty^\infty \mathrm{rect}\left(\frac{y}{Y}\right)\mathrm{rect}\left(\frac{y}{Y}\right)dy\right] \\ &= 0, \end{aligned} \tag{1.8}$$

where we have used the fact that the slits should have a separation bigger than their widths, $X > |x_i - x_j|$, and in this case the product of the rectangle functions inside the first integral is zero for all x.

In order to get some appreciation for these three conditions, we can think of the possible "defects" that could arise in case our map did not satisfy them. Some defects are depicted in figure 1.2. For property (ii), we have to keep in mind that we are trying to *simulate* a quantum system through another physical system: in this second system, we want not only to encode states, but also mimic operators on the first system.


(a)

(b)

Figure 1.2: Pictorial examples of: (a) a non-1-to-1 mapping (b) a non-surjective mapping.

- If h were not injective (one-to-one), we would have different states $|\psi_1\rangle$ and $|\psi_2\rangle$ mapped to the same field E_1; it is as if in the process of mapping, we had lost points of the original space.

- If h were not surjective (onto), we would have fields E without correspondent states $|\psi\rangle$; it is as if we have gained points through the process of mapping.

- If h did not preserve linear combinations, an important class of linear operators on $\mathcal{H}$ would have to be mapped to non-linear operators in $\mathcal{H}'$: let A be a linear operator in $\mathcal{H}$ with no zero-valued eigenvectors, and let A' be the corresponding operator in $\mathcal{H}'$ defined by[4]

$$A'(E) = h \circ A(\psi),$$

with E and ψ corresponding by $E = h(\psi)$; if it happens that

$$h(\alpha\psi_1 + \beta\psi_2) \neq \alpha h(\psi_1) + \beta h(\psi_2)$$

for some α, β, ψ_1 and ψ_2, we make ψ_{01} and ψ_{02} such that $A\psi_{0i} = \psi_i$ and thus

$$\begin{aligned} A'(\alpha E_{01} + \beta E_{02}) &= h \circ A(\alpha\psi_{01} + \beta\psi_{02}) \\ &= h(\alpha\psi_1 + \beta\psi_2) \\ &\neq \alpha h(\psi_1) + \beta h(\psi_2) = \alpha h \circ A(\psi_{01}) + \beta h \circ A(\psi_{02}) \\ \Rightarrow A'(\alpha E_{01} + \beta E_{02}) &\neq \alpha A'(E_{01}) + \beta A'(E_{02}). \end{aligned}$$

- If h did not preserve inner products, some probability distributions calculated from a state $|\psi\rangle$ would have a different value than that calculated with the corresponding field E: let $|\psi_m\rangle$, $|\psi_n\rangle$, E_m and E_n be such that $E_m = h(\psi_m), E_n = h(\psi_n)$ but

$$\langle E_m|E_n\rangle \neq \langle\psi_m|\psi_n\rangle\,;$$

we can analyse two cases to see that some distribution probability will be altered because of this defect in h. We can look, for example, at the probability that we have a positive result when applying on state ψ_β the projective measurement onto state ψ_α, which is given by $P(\alpha|\beta) = |\,\langle\psi_\alpha|\psi_\beta\rangle\,|^2$, and the corresponding probability $P'(\alpha|\beta) = |\,\langle E_\alpha|E_\beta\rangle\,|^2$.

1. If $|\,\langle E_m|E_n\rangle\,| \neq |\,\langle\psi_m|\psi_n\rangle\,|$, then the probabilities $P(m|n) = |\,\langle\psi_m|\psi_n\rangle\,|^2$ and $P'(m|n) = |\,\langle E_m|E_n\rangle\,|^2$ are clearly different.
2. If $\langle E_m|E_n\rangle = e^{i\phi}\,\langle\psi_m|\psi_n\rangle$ with ϕ not a multiple of 2π, then for the state

$$|+\rangle = \frac{1}{\sqrt{2 + 2\,\mathrm{Re}(\langle\psi_m|\psi_n\rangle)}}(|\psi_m\rangle + |\psi_n\rangle)$$

we have

$$\begin{aligned} P(m|+) &= \frac{1}{2 + 2\,\mathrm{Re}(\langle\psi_m|\psi_n\rangle)}\Big(1 + |\,\langle\psi_m|\psi_n\rangle\,|^2 + 2\,\mathrm{Re}(\,\langle\psi_n|\psi_m\rangle\,)\Big), \\ P'(m|+) &= \frac{1}{2 + 2\,\mathrm{Re}(\langle\psi_m|\psi_n\rangle)}\Big(1 + |\,\langle\psi_m|\psi_n\rangle\,|^2 + 2\,\mathrm{Re}(\,e^{i\phi}\,\langle\psi_n|\psi_m\rangle\,)\Big), \end{aligned} \tag{1.9}$$

[4] We will use $\circ$ to denote composition: $h \circ A(\psi) = h(A(\psi))$.

which will also be different given that $\langle\psi_m|\psi_n\rangle \neq 0$ (the case $\langle\psi_m|\psi_n\rangle = 0$ and $\langle E_m|E_n\rangle = 0$ is ruled out by the hypothesis $\langle\psi_m|\psi_n\rangle \neq \langle E_m|E_n\rangle$; the case $\langle\psi_m|\psi_n\rangle = 0$ and $\langle E_m|E_n\rangle \neq 0$ falls in the previous item).

Therefore, it is very important for the fields of the form (1.5) to have the same mathematical structure as the pure quantum states, otherwise one or more of the problems mentioned above will take place.

As a final remark before we proceed to the next section, there are two simplifications we would like to introduce in the basis of slit fields (1.6). We can treat the slits as having infinite width in the y direction, so that they behave as one-dimensional fields. Also, since the temporal behaviour is the same for all the slits, it can be factored out. Thus the single slit fields can be simplified to

$$E_i(x) = E_0 \mathrm{rect}\left(\frac{x - x_i}{X}\right). \tag{1.10}$$

1.2 Spatial qudits

Many of the applications we mentioned in the beginning of this chapter used single photons instead of the classical fields described above. It seems natural to think of the wavefront of the laser beam as being proportional to the probability amplitude of a single-photon multimode field [39]. Following this line of thought, we could represent the single-photon field in the one-dimensional position coordinate[5] as

$$|\psi\rangle = \int \psi(x)\,|1\,x\rangle\,dx, \tag{1.11}$$

where $\psi(x)$ is its normalized transverse probability amplitude, which is proportional to the transverse spatial profile of the beam. The states $|1\,x\rangle$ are defined here as

$$|1\,x\rangle = \frac{1}{\sqrt{2\pi}} \int e^{i\,kx}\,|1\,k\rangle\,dk, \tag{1.12}$$

in analogy to the Fourier transform of the Dirac delta function[6]; the states $|1\,k\rangle$ are the single-photon, plane-wave modes.

[5] The existence of a position operator for photons is a tough and ongoing debate; we are not recurring to a position operator to define the states $|1\,x\rangle$, but rather using the well defined plane-wave states.

[6] Definition 1.12, by imitating the Fourier transform of the Dirac delta, tries to capture the notion of a *localized* photon state; this is another tough and ongoing debate, but it is worth noting that we are thinking about localization in a scale that is small compared to our experimental apparatus' dimension, but still large compared to the photon wavelength.

Let us imagine that this single-photon field is sent through a screen with a D-slit transmission function

$$t(x) = \sum_{j=1}^{D} \tilde{c}_j \mathrm{rect}\left(\frac{x - \eta_j a}{X}\right), \tag{1.13}$$

where the $\tilde{c}_j$ are the complex transmission coefficients, a is the separation between the slits, X is their width and $\eta_j = (j-1) - (D-1)/2$. The field immediately after the screen will be

$$\begin{aligned} |\psi_t\rangle &= C \int t(x)\psi(x) \, |1\,x\rangle \, dx \qquad (1.14)\\ &= C \sum_{j=1}^{D} \tilde{c}_j \int \psi(x)\mathrm{rect}\left(\frac{x - \eta_j a}{X}\right) |1\,x\rangle \, dx, \qquad (1.15)\end{aligned}$$

where C is a normalization constant. Finally, if we assume that $\psi(x)$ is constant across all the slits [say $\psi(x) = \psi_0$], we can write the state as

$$|\psi_t\rangle = \sum_{j=1}^{D} c_j \, |j\rangle \,, \tag{1.16}$$

with $c_j = \tilde{c}_j / \sqrt{\sum_{i=1}^{D} |\tilde{c}_j|^2}$ and

$$|j\rangle = \frac{1}{\sqrt{X}} \int \mathrm{rect}\left(\frac{x - \eta_j a}{X}\right) |1\,x\rangle \, dx. \tag{1.17}$$

The field states 1.16 only represent the part of the field that is trasmitted through the screen, and do not account for the part that is reflected. Since this part is the one that can be detected by a detector after the screen, we say that these states are *postselected*.

These are the *spatial qudits* [40], which are D-dimensional quantum systems on their own right. As we have mentioned in the beginning of the chapter, they have been increasingly used as a resource for applications involving higher-dimensional states (cryptography, communication and fundamental tests, for example).

From what we have seen before, they are also isomorph to the slit fields (1.5), and by extension also isomorph to any other D-level quantum system. Therefore, we can also use them to simulate other quantum systems.

As we will see later, we used a classical beam in our experiment. However, the phase retrieval method that we are proposing would apply equally well to sigle-photon spatial qudits.

1.3 Loss of phase information

It is common to attribute more weight to the intensity than to the phase profile of a field. However, one striking example of the importance of the phase information was given by Eliyahu Osherovich [41] in his PhD thesis (see section 1.3.4), which we will reproduce now (but with two different characters).

Imagine we get two grayscale pictures, which can be thought of as two real fields (in the sense of having constant, zero phase profiles) f_1 and f_2. We can take their Fourier transforms[7] $F_1 = |F_1|e^{i\phi_1}$ and $F_2 = |F_2|e^{i\phi_2}$, exchange their phase profiles and finally take the inverse transform of the resulting functions. This whole operation would be

$$\begin{array}{c} f_1 = \mathcal{F}^{-1}(|F_1|e^{i\phi_1}) \\ f_2 = \mathcal{F}^{-1}(|F_2|e^{i\phi_2}) \end{array} \longrightarrow \begin{array}{c} f_1' = \mathcal{F}^{-1}(|F_1|e^{i\phi_2}) \\ f_2' = \mathcal{F}^{-1}(|F_2|e^{i\phi_1}) \end{array} . \tag{1.18}$$

Figure 1.3 depicts the result of this process: in the end we get exchanged pictures. This example illustrates how the phase information is important, as we could exchange the pictures themselves only by exchanging the phase profiles of their Fourier transforms.

Concerning the phase profiles of the slit fields, in equation (1.5) one can see that without controlling the phase of the field inside each slit region, it is impossible to generate all of the desired fields. Phase control is essential for this regard. Moreover, if we were to determine which field was generated with this scheme, we would need to measure the phase of the field in the slit regions[8].

Photodetectors, however, do not record the phases of a field when it impinges on them, they only record intensities. To overcome this difficulty, several schemes have been devised to make the intensity at a given point dependent of the the field's phase — for example by interfering the field under investigation with a plane wave reference field.

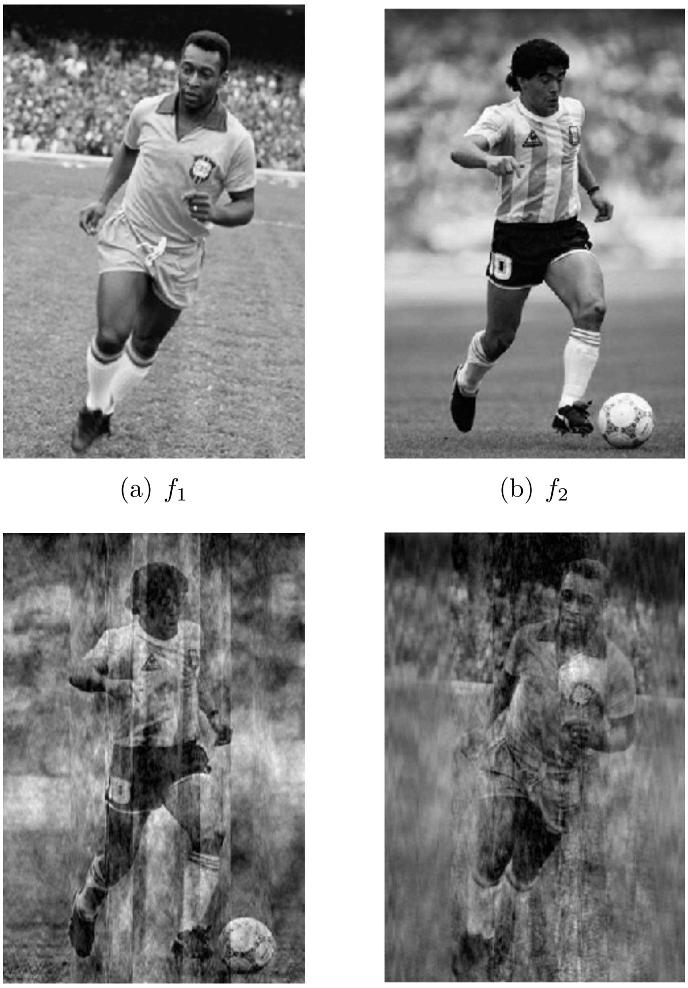
There is another such scheme, that takes advantage of the fact that the amplitudes of a Fourier transform depend on the phases of the original function. For instance, let us take the double-slit field

$$E_t(x) = \frac{1}{\sqrt{2}}\text{rect}\left(\frac{x-a/2}{X}\right) + \frac{e^{i\phi}}{\sqrt{2}}\text{rect}\left(\frac{x+a/2}{2X}\right), \tag{1.19}$$

[7] Throughout the text, we will be denoting the Fourier transform operation by $\mathcal{F}$, and Fourier-transform pairs as the same letter with lower and upper cases; the Fourier transform F of a discrete function f can be defined [42] as

$$F_k = \sum_n f_n e^{-i\,2\pi kn/N}$$

[8] To be more precise, we would need to measure the phases inside each slit *with respect* to the phase at some arbitrary point (for example, the phase inside the leftmost slit), as there is always a global phase that does not have physical meaning.



(a) f_1

(b) f_2

(c) $|f_1'|$

(d) $|f_2'|$

Figure 1.3: Illustration of the process given in equation (1.18); (a) and (b) show the original functions f_1 and f_2, while (c) and (d) show the absolute values of the resulting functions, f_1' and f_2', after exchanging their phase in the Fourier domain. Besides adding some noise, the net result is to interchange the pictures, which illustrates how important the phase information is.

(a is the separation of the slits) which has the same amplitude in both slit regions, but has an arbitrary phase ϕ in the second. According to the scalar theory of diffraction, after propagating a long distance, the diffracted field becomes a scaled Fourier transform of the original field [38]:

$$U_{\text{far}}(x) = \frac{-i}{\lambda z} \int U(\eta) e^{-i\frac{2\pi}{\lambda z}x\eta} d\eta. \tag{1.20}$$

where z is the propagation distance. Therefore, the far field corresponding to 1.19 will be

$$E_{\text{far}}(x) = \frac{e^{i\Phi(z)}}{i\lambda z}\sqrt{2}X \cos\left(\frac{2\pi^2 a}{\lambda z}x + \frac{\phi}{2}\right) \operatorname{sinc}\left(\frac{2\pi X}{\lambda z}x\right), \tag{1.21}$$

where λ is the wavelength of the electromagnetic field. We are regarding the longitudinal distance z as a fixed parameter and the phase $\Phi(z)$ as a global phase.

The amplitude profile of this field can be thought of as composed by two factors: a sinc envelope with a cosine oscillation. The phase ϕ has the role of displacing the oscillation pattern inside the envelope, as can be seen in figure 1.4

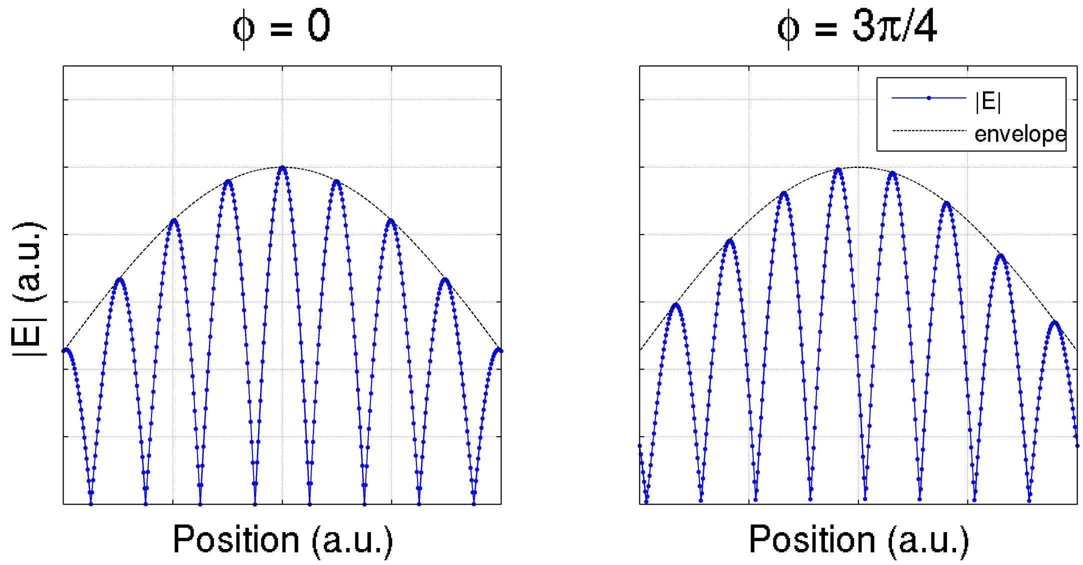


Figure 1.4: Amplitude profile of diffracted fields given in (1.21) for (a) $\phi = 0$ and (b) $\phi = \frac{3\pi}{4}$. The phase difference ϕ between the slits causes the oscillatory pattern to shift inside the envelope. This illustrates that, in general, the amplitudes of a Fourier transform also depend on the phases of the original function; this fact is what underlies the algorithm we will see in the next chapter.

The dependence of the Fourier amplitudes on the original function phases is not trivial. In order to use the phase information that is somehow "buried" in the Fourier amplitudes, we have to resort to numerical algorithms, which we will discuss in the next chapter.

1.4 Fidelity between two quantum states

Later on we will be interested in quantifying how similar two quantum states are, and we will be using a measure called *fidelity*. The fidelity between two pure state $|\psi\rangle$ and $|\phi\rangle$ is defined by [9]

$$F(\phi,\psi) = |\langle\phi|\psi\rangle|. \tag{1.22}$$

(The fidelity can be defined in a manner that includes non-pure states, but this pure state version will be enough for us). In inner product spaces (of which Hilbert spaces are a special case), the angle θ_{uv} between two unit vectors u and v is defined by

$$\cos\theta_{uv} = \langle u|v\rangle\,, \tag{1.23}$$

so that the fidelity between the two pure states $|\psi\rangle$ and $|\phi\rangle$ is just the absolute value of the cosine of their angle[9].

It is easy to see that when the first state lies on the subspace generated by the second (and thus their angle is either 0 or π), their fidelity equals one, and that when it lies on the subspace orthogonal to the second, their fidelity equals zero. Also, in intermediate situations the fidelity will lie between 0 and 1.

[9] It is also common to encounter the fidelity defined as the square of that defined in equation (1.22); if the reader is to compare the fidelities we found in this work with those found in another, this should be kept in mind.

Chapter 2

Phase retrieval algorithms

In the last chapter we have seen that the phase of a function and the amplitudes of its Fourier transform are related, though this relation is not trivial. In this chapter, we are going to see a few numerical algorithms that take advantage of this fact in order to solve the phase retrieval problem. For now, we will be more concerned in explaining the general features of the algorithms and leave the specific adaptations we made for our experimental setup to the next chapter.

2.1 The Gerchberg-Saxton algorithm

In 1972, Gerchberg and Saxton published an iterative algorithm to solve the phase retrieval problem in the context of electron microscopy [43]. In their experiments, an electron beam was impinged onto a sample which then scattered it, and assessing the scattered wavefunction of the electrons in the beam gave information about the sample object [7].

The electrons in the beam would propagate until they reach a two-dimensional detector, which would record a signal proportional to the intensity profile $|\psi(x)|^2$ of the electron beam [1]. By controlling the current in electromagnetic lenses and thus tuning a magnetic field that the beam had to transpose, they could control the propagation of the electron beam. Specifically, they could make the plane where the detector lied an *object plane* or a *Fourier plane*, just as with optical lenses. That is, they could make the wavefunction at the detector plane the same as immediately after the sample, when the *object plane* configuration was being used, or they could make it equal its Fourier transform with the *Fourier plane* configuration.

Having the measured intensity profiles $|f(x)|^2$ and $|F(u)|^2$ of the Fourier transform

[1] Even though x is a two-dimensional variable at this point, we will omit the vector notation because our work was done with one-dimensional functions, and the algorithm can be applied in the very same way.

pair $f - F$ at hand, one iteration of the Gerchberg-Saxton algorithm proceeds in four steps:

Gerchberg-Saxton algorithm

(i) make an estimate $\phi(x)$ of the phases in the object domain, then build an estimate $g(x) = |f(x)|e^{i\phi(x)}$ of the function f;

(ii) take the Fourier transform of the estimate g to find $G(u) = |G(u)|e^{i\psi(u)}$;

(iii) correct the amplitudes of the Fourier transform G, but keep its phases: $G'(u) = |F(u)|e^{i\psi(u)}$;

(iv) take the inverse Fourier transform of G' to arrive at a new estimate $g'(x) = |g'(x)|e^{i\phi'(x)}$ of f.

By the end of step (iv), the new phase estimate of f is of course $\phi'(x) = \arg(g'(x)))$.

In the first iteration, we should start the algorithm using a random phase estimate in step (i) or an educated guess, if we have enough prior knowledge. But by the end of an iteration, we feed the phases $\phi'(x)$ of the new estimate g' back into step (i) and start another iteration. Thus, step (i) imposes the object-domain amplitudes $|f(x)|$ at g', in a similar fashion to what step (iii) does in the Fourier domain. The iterative process is illustrated in figure 2.1.

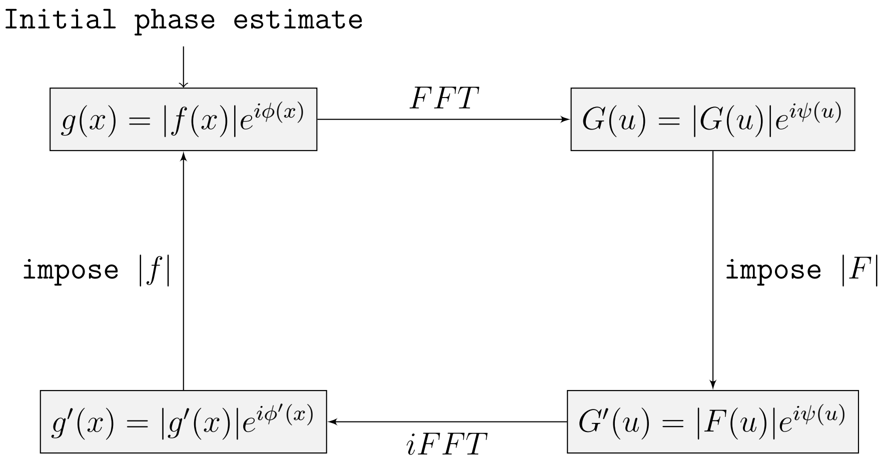

Initial phase estimate
$g(x) = |f(x)|e^{i\phi(x)}$
FFT
$G(u) = |G(u)|e^{i\psi(u)}$
impose $|f|$
impose $|F|$
$g'(x) = |g'(x)|e^{i\phi'(x)}$
$iFFT$
$G'(u) = |F(u)|e^{i\psi(u)}$

Figure 2.1: Gerchberg-Saxton iterative algorithm.

To keep track of the progress of the algorithm, it is useful to define the object-domain normalized error:

$$E_o = \frac{\sum_x (|g'(x)| - |f(x)|)^2}{\sum_x |f(x)|^2}. \tag{2.1}$$

This is just the squared residual of the amplitudes of the estimate function g' at the end of one iteration, normalized by the total object-domain intensity, and is calculated repeatedly by the end of each iteration. Since step (i) will change the amplitudes of g' but not its phases, the sum in the denominator is also a measure of the correction our estimate will suffer when the next iteration begins, and thus a measure of how far we are from the answer in the current iteration.

The iterative process is repeated until a numerical criterion is satisfied, for example, a certain number of iterations being reached. Another common choice is to repeat the iterations until the difference between errors in successive iterations is smaller than a fraction of the error in the current iteration. By the end, the algorithm should reach a good estimate of the phase of f.

2.1.1 Some intuition on the workings of the Gerchberg-Saxton algorithm

At first, it might not be obvious why this algorithm works at all. But one has to remember that by changing the amplitudes in either the Fourier or the object domains, we also change both the amplitudes and phases in the other domain, since these are inter-related. By making successive modulus impositions in steps (i) and (iii) of the algorithm, we are actually correcting the phase estimate $\phi(x)$ until it (hopefully) converges to the actual phase of the function f.

In figures 2.2 and 2.3, we illustrate this process with the results of the first iterations of the algorithm when ran with the chirp function

$$f(x) = e^{i16\pi x^2}\mathrm{rect}(x). \tag{2.2}$$

In these figures, the red full lines represent actual values of phase or modulus, while the black or blue dots show those quantities in the current estimate of the algorithm.

The first row of figure 2.2 shows (from left to right) $|g|$, $\arg(g)$ and $|G|$ obtained in steps (i) and (ii) of the first iteration. One can see that, in spite of having the correct object-domain amplitudes, the random phases of g give its Fourier transform a spectrum that resembles white noise. In the second row there are (from right to left) plots of $|G'|$, $\arg(g')$ and $|g'|$ obtained after steps (iii) and (iv) of the same iteration. It is possible to see that by imposing the correct Fourier-domain amplitudes $|F|$ made the object-domain phase estimate much less random than it was initially.

The third and fourth rows in figure 2.2 show plots of the same quantities as in the first and second rows, respectively, but for the estimates in the second iteration of the program. We can see great improvement in $|G|$ and $|g'|$ with respect to the first iteration,

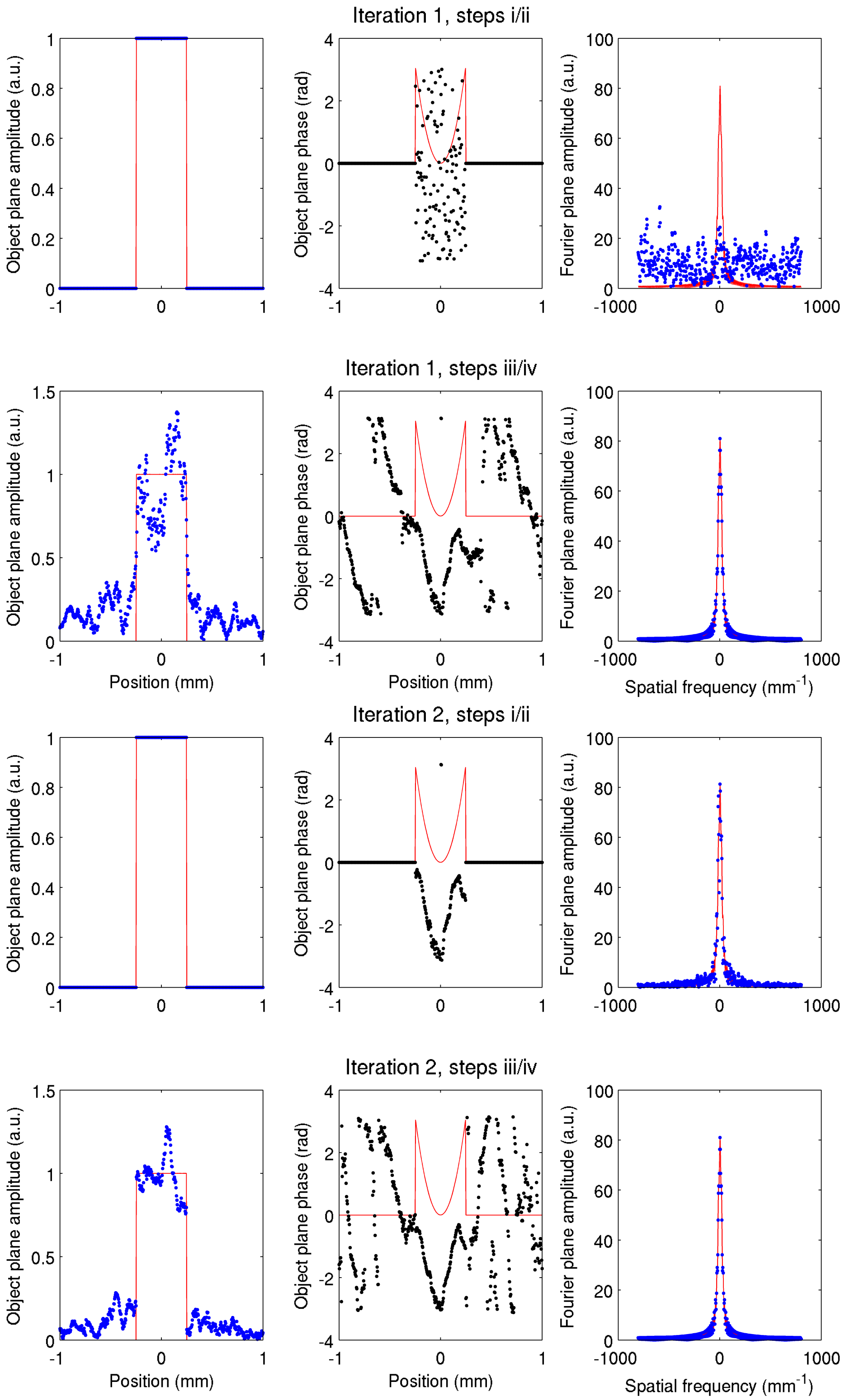


Figure 2.2: Modulus and phase profiles of estimates g and g', and modulus profile of G and G' for iterations 1 (first two rows) and 2 (last two rows) of the GSA; the target function in this example is given in equation (2.2). Full lines represent the actual quantities, while dots represent the estimates at the current iteration.

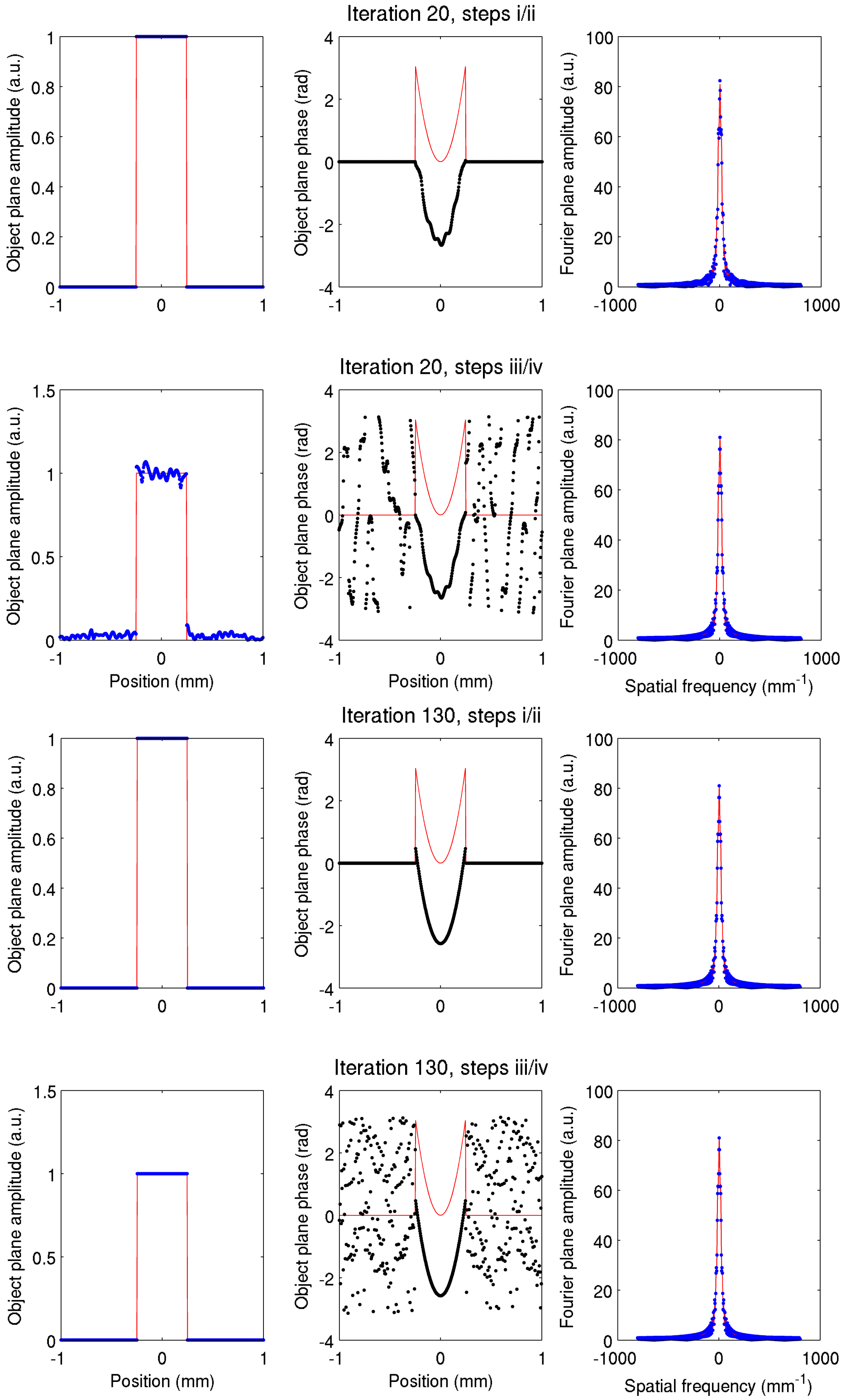


Figure 2.3: Same quantities as in figure 2.2, but for iterations 20 (first two rows) and 130 (last two rows) of the GSA. Full lines represent the actual quantities, while dots represent the estimates at the current iteration. One can see that, as the iterations proceed, the estimates approach the actual chirp function (2.2) both in amplitude and phase.

and the phase estimates now start to resemble the actual phase profile.

Figure 2.3 top two rows shows the same quantities described above, but now for iteration 20. By now, $|g'|$ seems quite close to the actual values of $|f|$. The object-domain phases resemble the original values, but they are vertically shifted due to a global phase of the estimate with respect to the actual phases $\arg(f)$, which is actually not a problem.

Finally, figure 2.3 bottom two rows shows these quantities once again in iteration 130. The phase estimate is now quite close to the actual values in the region with non-zero amplitudes (apart from the constant phase shift) and $|g'|$ is also close to the rectangle function.

2.1.2 Weak convergence of the Gerchberg-Saxton algorithm

One striking feature of the Gerchberg-Saxton algorithm is that the error E_o [given in equation (2.1)] never increases after one iteration, it can only decrease or stay unchanged. We say thereafter that the Gerchberg-Saxton algorithm is *weakly convergent*. For enstance, figure 2.4 shows the value of $\log_{10}(E_o)$ at each iteration of the previous example with the chirp function.

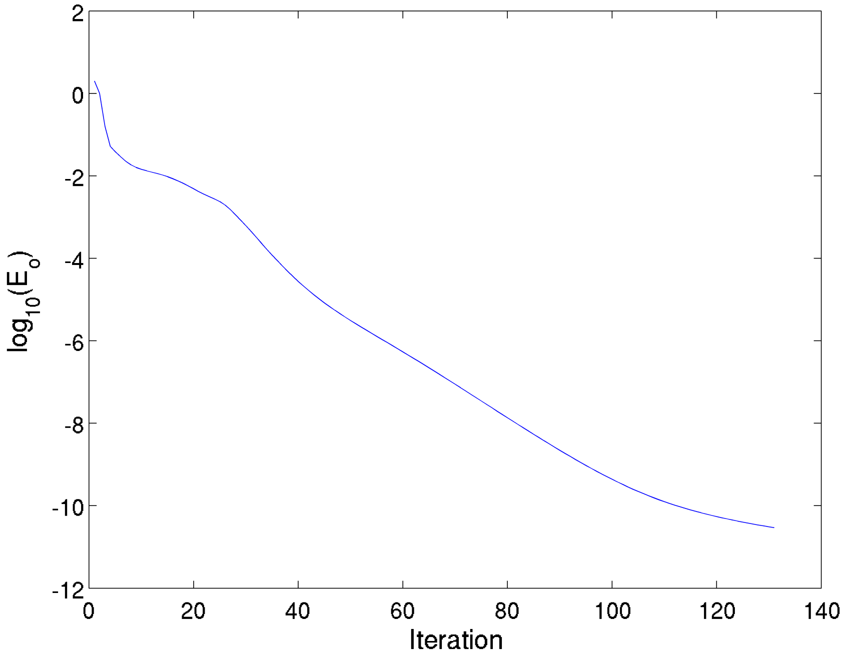


Figure 2.4: Progression of $\log_{10}(E_o)$ when the GSA was run with the chirp function (equation (2.2) and figures 2.2 and 2.3).

We will follow the proof of weak convergence given in [44]. But first, let us establish some useful notations.

Preliminary definitions

The object-domain error has been defined in equation (2.1); following that definition, this error in k-th iteration is given by

$$E_{o,k} = \frac{\sum_x \left(|g'_k(x)| - |f(x)|\right)^2}{\sum_x |f(x)|^2}, \tag{2.3}$$

but if we remember that in this algorithm $\arg(g'_k(x)) = \arg(g_{k+1}(x))$ and that $|g_{k+1}(x)| = |f(x)|$, we can rewrite $E_{o,k}$ as

$$E_{o,k} = \frac{\sum_x |g'_k(x) - g_{k+1}(x)|^2}{\sum_x |f(x)|^2}. \tag{2.4}$$

Similarly, we can define the Fourier-domain error at iteration k as

$$E_{F,k} = \frac{\sum_u (|G_k(u)| - |F(u)|)^2}{\sum_u |F(u)|^2}, \tag{2.5}$$

and if we remember that $\arg(G'_k(u)) = \arg(G_k(u))$ and that $|G'_k(u)| = |F(u)|$, we can rewrite it as

$$E_{F,k} = \frac{\sum_u |G_k(u) - G'_k(u)|^2}{\sum_u |F(u)|^2}. \tag{2.6}$$

Weak convergence

Parseval's theorem states [42] that for any function h, its total intensity and the intensity of its Fourier transform H are related by

$$\sum_x |h(x)|^2 = \frac{1}{N} \sum_u |H(u)|^2 \tag{2.7}$$

with N the number of points x in the object-domain (and thus also in the Fourier-domain).

First we look at the alternative definition of $E_{o,k}$ in equation (2.4), and use Parseval's theorem with $h(x) = g'_k(x) - g_{k+1}(x)$ to arrive at

$$E_{o,k} = \frac{1}{NI_f} \sum_u |G'_k(u) - G_{k+1}(u)|^2 = \frac{1}{I_F} \sum_u |G'_k(u) - G_{k+1}(u)|^2,$$

where we have made $\sum_x |f(x)|^2 = I_f$ and $\sum_u |F(u)|^2 = I_F = NI_f$. However, for each u we have that, among all the numbers in the complex circle of radius $|F(u)|$, $G'_{k+1}(u)$ is the closest complex number to $G_{k+1}(u)$, since they have the same phase. Therefore we have

$$\frac{1}{I_F}\sum_u |G'_k(u) - G_{k+1}(u)|^2 \geq \frac{1}{I_F}\sum_u |G'_{k+1}(u) - G_{k+1}(u)|^2$$

and thus

$$E_{o,k} \geq E_{F,k+1}. \tag{2.8}$$

On the other hand, we can use Parseval's theorem with $H(u) = G'_{k+1}(u) - G_{k+1}(u)$ to get from equation (2.6)

$$E_{F,k+1} = \frac{1}{I_F}\sum_u |G'_{k+1}(u) - G_{k+1}(u)|^2 = \frac{1}{I_f}\sum_x |g'_{k+1}(x) - g_{k+1}(x)|^2.$$

Similarly, we note that, for each x, $g_{k+2}(x)$ is the complex number closest to $g'_{k+1}(x)$ among the numbers in the circle of radius $|f(x)|$, since they have the same phase. Hence

$$\frac{1}{I_f}\sum_x |g'_{k+1}(x) - g_{k+1}(x)|^2 \geq \frac{1}{I_f}\sum_x |g'_{k+1}(x) - g_{k+2}(x)|^2$$

and therefore

$$E_{F,k+1} \geq E_{o,k+1}. \tag{2.9}$$

From equations (2.8) and (2.9), we have finally that

$$E_{o,k+1} \leq E_{o,k}, \tag{2.10}$$

which completes the proof.

2.1.3 Nonuniqueness and local-minima stagnation problems

On one hand, the weak convergence of the Gerchberg-Saxton algorithm establishes that it is a rather safe algorithm, in the sense that the estimate in the end of each iteration will not be worse than the estimate at the end of the previous iteration in terms of the error E_o. On the other hand, it also tells us that the algorithm is somewhat similar to gradient-descent algorithms, in the sense that it can get stagnated at a local minimum of the error that is not the global minimum. This can happen because the algorithm cannot "climb out" from an eventual error "well".

Another difficulty that one has to deal with is the nonuniqueness of solutions of the phase retrieval problem. It is known [1] that there are more that one transform-pair $f - F$ that conform to $|f|$ and $|F|$ in some situations — for example the conjugate pair $f^* - F^*$ when $|f|$ and $|F|$ are centro-symetric. In addition, it has been reported in [1, 45] that the one-dimensional phase retrieval problem suffers more severely from non-uniqueness than the two-dimensional problem, and that *a priori* information on the function can also affect how severe the non-uniqueness will be. One way to avoid these two problems is to run the algorithm several times with different starting phase estimates, as we will discuss in the next chapter.

2.2 Fienup's family of algorithms

In spite of being safe, the Gerchberg-Saxton algorithm (to which we will refer as GSA from now on), usually converges slowly. We will turn our attention to a family of variations of the GSA that was developed to improve its performance by J. Fienup [6]. One remark before we begin though: Fienup devised his algorithms for a version of the phase retrieval problem that is different from the one we are interested in. In our experiment, it is easy to assess both the moduli $|f|$ and $|F|$. Fienup was confronting a problem with less prior information about f: Instead of $|f|$, what is known is only that f is real and non-negative. Therefore, we are presenting versions of the algorithms that are adapted to our problem. The best-performing algorithm of this family of variations, called the *hybrid input-output algorithm* [44], did not have such an adaptation (in fact, it would reduce to the so-called *input-output* algorithm, which we are about to see), but the interested reader is encouraged to pursue it.

We can think of the last three operations in each iteration of the GSA (see figure 2.1 and the box above it) as a non-linear system that receives g_k as input and gives g'_k as output. Although the steps (ii) and (iv) are linear, step (iii) is not. Therefore this sequence of operations as a whole is non-linear (as sketched in figure 2.5). Because of step (iii), g'_k already has the desired modulus in the Fourier-domain; it only lacks the correct object-domain modulus in order to be a solution. Once we have calculated $g'_k(x)$, we can find the correction $\Delta g_d(x)$ it needs in order to satisfy the object-domain constraint, that is, we can find Δg_d so that $|g'_k(x) + \Delta g_d(x)| = |f(x)|$. For example, one possibility is

$$\Delta g_d(x) = -g'_k(x) + \frac{|f(x)|}{|g'_k(x)|} g'_k(x), \tag{2.11}$$

but the question then is how to change the input g_k so that the output g'_k receives this desired correction.

After running some iterations of the GSA, one would expect to be somewhat close to a solution, so the desired correction Δg_d should be small. However, when one makes a small perturbation in the input, the corresponding perturbation in the output is approximately proportional. In other words, the system responds almost linearly to small perturbations,

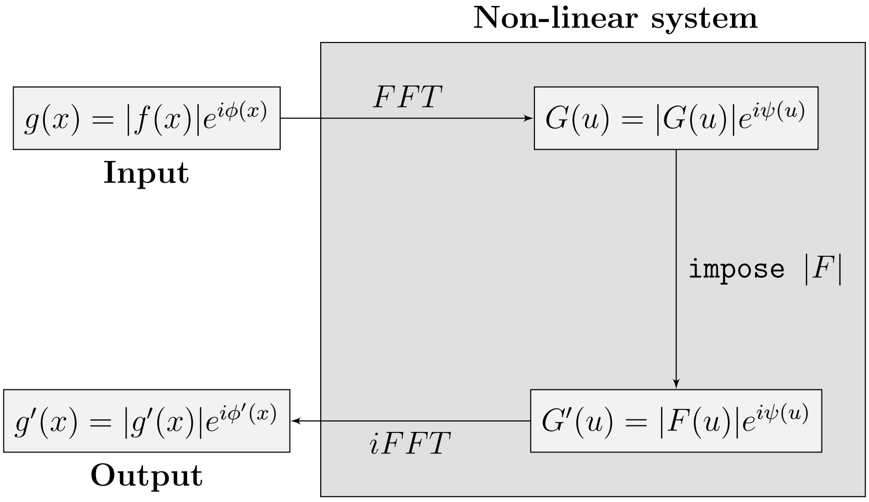

Non-linear system
$g(x) = |f(x)|e^{i\phi(x)}$
Input
FFT
$G(u) = |G(u)|e^{i\psi(u)}$
impose $|F|$
$G'(u) = |F(u)|e^{i\psi(u)}$
$iFFT$
$g'(x) = |g'(x)|e^{i\phi'(x)}$
Output

Figure 2.5: Steps (ii), (iii) and (iv) of the Gerchberg-Saxton algorithm as a non-linear system.

so the input

$$g_{k+1}(x) = g_k(x) + \Delta g(x) \tag{2.12}$$

will give as output

$$g'_{k+1}(x) = g'_k(x) + \alpha\Delta g(x), \tag{2.13}$$

where α is the proportionality coefficient between the perturbations, which is characteristic of the non-linear system. Therefore, by choosing $\Delta g(x) = \alpha^{-1}\Delta g_d(x)$ we can induce the desired correction in g'_k.

The difficulty in this error compensation approach is that the exact value of α is hard to assess, as it depends on the statistics of $|F(u)|$ and on the current solution estimate g'_k [6]. This means the coefficient can change as we get closer to the solution or even depend on the function one is trying to reconstruct. It is possible to estimate the value of α^{-1} numerically as reported in [44]: one can make several runs from the same starting phase estimate, each with a different value of β (an estimate for α^{-1}) and all with the same total number of iterations, and then compare the error curves to determine which β value gave the algorithm the best error progression.

It is also possible to make an estimate of the value of β through the expected value $\langle|F|/|G|\rangle$, as shown in the appendix of [6]. Such an estimate can help in determining a suitable testing range for β.

We made an estimate of the optimal value for β by generating several curves of E_o against the iteration number (shown in figure 2.6), following the method we will describe now. We ran 20 iterations of the GSA algorithm with the function of a spatial qutrit state (more details in the next chapters), followed by 100 iterations of the output-output variation (which we will see very soon), and then 50 more iterations of the GSA. The first iterations with the GSA were intended to make the estimate somewhat close to an answer. The last were used because it has been reported [44] that Fienup's family of variations might increase the error E_o while actually improving the image quality; it could be the case that one of the higher-error estimates was actually better in a visual criterium, so that running a few GSA iterations again would place its error below the other ones. This was not the case here, though[2].

To generate the curves in figure 2.6, we ran the algorithm several times with a fixed initial phase estimate, but each time using a different value for β. At the left figure, one can see that too high values of β (namely $\beta = 2, 2.5$, 3 and 3.5) made the error increase after some iterations, meaning that the algorithm was unstable with those values. On the other hand, at the right figure, one can see that the lower values of β did not suffer from this instability, and that $\beta = 1.5$ gave the fastest decreasing curve. We chose to use $\beta = 1.3$ in our routine in order to have a safety margin.

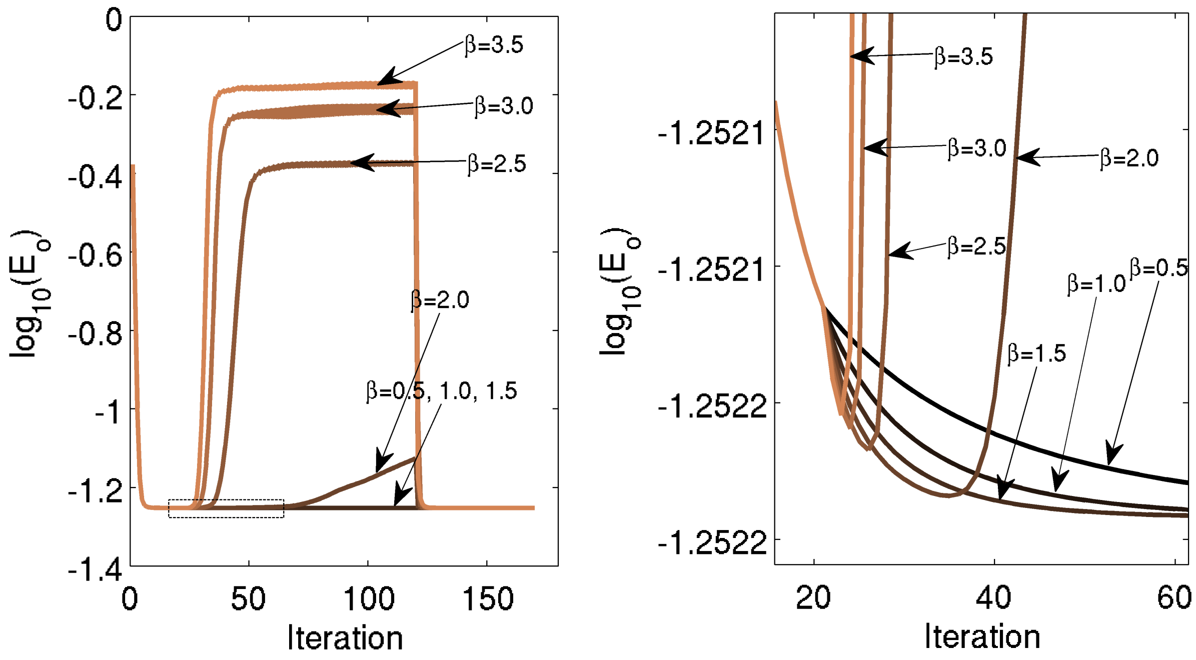


Figure 2.6: Numeric method for estimating the best value for β; (*Left*) E_o vs. iteration number for several values of β; (*right*) zoom at the beginning of non-GSA iterations. The value $\beta = 1.5$ gave the fastest-decreasing curve without making the algorithm unstable, and was therefore the best tested value.

With an estimate β of α^{-1} at hand, the first variation of the GSA would be to correct g_k by $\beta \Delta g_d$ [see equation (2.11)] when starting iteration $k + 1$:

[2] As we mentioned earlier, Fienup was treating a different version of the phase retrieval problem. The object-domain error in his case was different than ours, and somewhat less linked to the visual quality of the image.

Input-output algorithm

- proceed with steps (i) to (iv) of the GSA;
- correct the estimate $g_k(x)$ by $\beta\Delta g_d$:

$$g_{k+1}(x) = g_k(x) + \beta\left(\frac{|f(x)|}{|g'_k(x)|} - 1\right) g'_k(x).$$

This is the so-called *input-output* algorithm, in which we compensate the input to the next iteration based on the error of the current output.

Another variation of the GSA is the *output-output* algorithm, which relies on another characteristic of the non-linear system (figure 2.5). If an output $g'(x)$ is fed as input to the system, the new output is again $g'(x)$, since the third step will not make any change in $\mathcal{F}[g']$. Therefore, we can consider every output as resulting from itself as input. Hence we can use a perturbed input similar to (2.12),

$$g_{k+1}(x) = g'_k(x) + \Delta g(x), \tag{2.14}$$

to get the same output as in equation (2.13). Following the same reasoning as before and making $\Delta g = \beta\Delta g_d$ [see equation (2.11)], we arrive at another variation of the GSA, called *output-output algorithm*:

Output-output algorithm

- proceed with steps (i) to (iv) of the GSA;
- correct the output $g'_k(x)$ by $\beta\Delta g_d$:

$$g_{k+1}(x) = g'_k(x) + \beta\left(\frac{|f(x)|}{|g'_k(x)|} - 1\right) g'_k(x).$$

It is interesting to note that the output-output algorithm is equivalent to the GSA when we choose $\beta = 1$. Thus we can think of the GSA as a special case of the output-output algorithm, but with a suboptimal parameter value since it converges somewhat slowly. We can also infer that, while $\beta = 1$ might not the best value, it is probably not very far off, since the GSA is still very reliable. Larger values of β can make this family of variations converge faster, as they would make more significant changes to the input after each iteration. However, excessively large values might make them unstable, as we have seen in figure 2.6.

The best strategy reported in [44] is to alternate between the original GSA and one of its variations, and was the strategy we followed when treating our data.

Chapter 3

Phase retrieval algorithms for spatial qudits: Problem-specific adaptations

In the last chapters we have seen how the phase retrieval problem arises when we try to determine the state of a spatial qudits, and also a few numerical algorithms that could solve it using object- and Fourier-plane intensity measurements. Our discussion so far was about the general working of these algorithms, but in this chapter we shall turn our attention to the specific adaptations we had to make so they could suit our problem.

3.1 Match of frequencies by expanding object-domain amplitude vector

In the last chapter, we saw some algorithms to reconstruct a (complex) field given its modulus profile and that of its Fourier transform. In other words, these two profiles are the input that the algorithm uses, and therefore are the quantities we need to measure.

In our experiment, the Fourier transform of the qudit field was obtained by means of a lens. It can be shown that, in the Fraunhoffer diffraction regime, the field at a distance l_f after a spherical lens of focal length also equal to l_f is given by [38]

$$U_{\text{fo}}(x) = \frac{-i}{\lambda l_f} \iint U_{\text{im}}(u) e^{-i\frac{2\pi}{\lambda l_f}(ux)} du, \tag{3.1}$$

with U_{im} being the field at a distance l_f *before* the lens and λ the light wavelength. In other words, the field at the output focal plane corresponds to a scaled Fourier transform of the field at the input focal plane. In our apparatus, the field U_{im} was a spatial qudit field of the form (1.5). U_{im} will play the role of f in the phase retrieval problem, while U_{fo} will play that of F (see chapter 2).

We placed a lens with its input focal plane over the plane where the qudit field U_{im} was prepared, and set a camera at the output focal plane, so that it measured the Fourier transform intensities of the qudit field (we will discuss the experimental setup in more details in the next chapter). Each pixel of the camera measured the intensity of the field that was entering it; therefore, our experimental measurement gave us a vector of Fourier amplitudes, sampled at the positions

$$x_{\mathrm{fo}}^{(n)} = n\Delta x_{\mathrm{fo}}, \quad n = -\left\lfloor \frac{N_{\mathrm{fo}}}{2} \right\rfloor, \ldots, \left\lceil \frac{N_{\mathrm{fo}}}{2} \right\rceil, \tag{3.2}$$

where Δx_{fo} is the camera pixel size and N_{fo} is the number of pixels in the image. Our camera had a pixel size of $\Delta x_{\mathrm{fo}} = 5.2\,\mu\mathrm{m}$ and the images had up to 1268 pixels, the size of the camera. According to equation (3.1), the field U_{fo} at each position $x_{\mathrm{fo}}^{(n)}$ corresponds to the Fourier transform at the spatial frequencies

$$p_{\mathrm{fo}}^{(n)} = \frac{2\pi}{\lambda l_f} x_{\mathrm{fo}}^{(n)} = \frac{2\pi}{\lambda l_f} \Delta x_{\mathrm{fo}} n. \tag{3.3}$$

However, the discrete Fourier transform (DFT) of a vector $V(j)$ with N entries, sampled at positions $x^{(j)} = j\Delta x$, is another vector with N entries $V_{\mathrm{DFT}}(n)$, given by [42]

$$V_{\mathrm{DFT}}(n) = \sum_{j=-\lfloor N/2 \rfloor}^{\lceil N/2 \rceil} V(j) e^{i\frac{2\pi}{N}jn} = \sum_{j=-\lfloor N/2 \rfloor}^{\lceil N/2 \rceil} V(j) e^{i\frac{2\pi}{N\Delta x}nx^{(j)}}, \tag{3.4}$$

which is sampled at the spatial frequencies

$$p_{\mathrm{DFT}}^{(n)} = \frac{2\pi}{N\Delta x} n, \quad n = -\left\lfloor \frac{N}{2} \right\rfloor, \ldots, \left\lceil \frac{N}{2} \right\rceil. \tag{3.5}$$

These frequencies $p_{\mathrm{DFT}}^{(n)}$ are the ones the computer will be using in its internal representation while running the FFT and $iFFT$ routines. But since we will be imposing experimentally measured amplitudes during step (iii) of the GSA (see section 2.1), it is essential that these frequencies match those corresponding to the sampling of the camera pixels, otherwise we will be imposing Fourier amplitudes at the wrong values of frequency. Therefore, the frequencies (3.3) and (3.5) need to match. This will be achieved as long as we have

$$N = \frac{\lambda l_f}{\Delta x_{\mathrm{fo}} \Delta x} = \frac{\lambda l_f}{(\Delta x)^2}. \tag{3.6}$$

Here Δx is the sampling step size of the field U_{im}, but in our setup the camera used to sample it was identical to the camera sampling the field U_{fo} in the Fourier-plane, and thus we have $\Delta x_{\mathrm{fo}} = \Delta x$.

Imposing this value for N meant to change the size of the vector $V(j)$ — whose role is played by the spatial qudit field U_{im}. By using $\lambda = 691\,\text{nm}$, $l_f = 0.3\,\text{m}$ and $\Delta x = 5.2\,\mu\text{m}$, one would conclude that $N = 7666$ pixels were needed in our image, while our camera had only 1268, and thus we needed to extend our object-domain image[1]. This was achieved by placing 3199 zeroes before and another 3199 after the vector of measured intensities (these are called the *trailing zeroes*).

Since the spatial qudit fields are zero-valued outside the slit regions, this procedure did not compromise the results, since the object-domain camera was wide enough to capture all the slits. In oher words, completing the object-domain vector with trailing zeroes mimics what the camera should have measured if it had more pixels.

3.2 Partial imposition of Fourier-domain amplitudes

When dealing with other variations of the phase retrieval problem, in which one has different *a priori* information about f and F, the usual approach is to modify step (iii) and the update of g_k in the GSA. Instead of imposing $|f|$ and $|F|$ as in our case, one can simply impose those conditions which f and F are known to satisfy.

In its original form, the GSA works with known moduli $|f|$ and $|F|$ across the whole image; this rather strong condition can be grasped in the proof of the weak convergence (section 2.1.2), which uses Parseval's theorem and therefore presupposes that the sums over x and u, in the calculus of the errors E_o or E_F, span the whole object or Fourier domain. This entails that $|f(x)|$ and $|F(u)|$ are known at every point in both domains.

Our amplitude measurements in the Fourier domain, however, yielded us only up to 1268 pixels (the reason why we are saying *up to* will be clear soon), while the whole domain had 7666. We only knew the values for $|F|$ at the central portion of the spectrum therefore. Even though this is not the case corresponding to the original GSA, we carried our iterations with a slight modification in step (iii): the imposition of $|F|$ was only *partial*, in the measured region. We let the values of the estimate G float freely outside that region.

Before running our routine on experimentally measured data, we made several tests using simulated values. In those tests, we used $|F|$ vectors with 1268 entries, but this limitation in the domain did not seem to be too serious, as only a small part of the intensity in the Fourier plane fell off of the region in which $|F|$ was known. However, our experimental data had only 608 pixels, and the intensity falling off the measured region

[1] To be precise, we have extended three vectors to a length of 7666 entries: the one with the object-domain measured amplitudes $|f|$, and the estimates in both domains, g and G. The first was extended with trailing zeroes, and the other two just followed naturally to have more entries, since they are fourier transforms of the first.

was considerable. To overcome this difficulty with the experimental data, we had to resort to the modification described in section 3.6.

3.3 State estimation from the algorithm results

The output of the phase retrieval algorithm was a vector g with many complex entries (namely 7666). This vector corresponded to an optical field of the form (1.5), expected to be piecewise constant. We built the estimated state $|\tilde{\psi}_g\rangle$ corresponding to g with the median of the amplitudes and phases inside each slit region. That is, we built the estimated state as

$$|\tilde{\psi}_g\rangle = \frac{1}{C}\begin{bmatrix} \tilde{A}_1 e^{i\tilde{\phi}_1} \\ \tilde{A}_2 e^{i\tilde{\phi}_2} \\ \vdots \\ \tilde{A}_D e^{i\tilde{\phi}_D} \end{bmatrix}, \tag{3.7}$$

where

$$\tilde{A}_j = \underset{x\in X_j}{\text{median}}\big(|g(x)|\big), \tag{3.8}$$

$$\tilde{\phi}_j = \underset{x\in X_j}{\text{median}}\left(\arg\big(g(x)\big)\right), \tag{3.9}$$

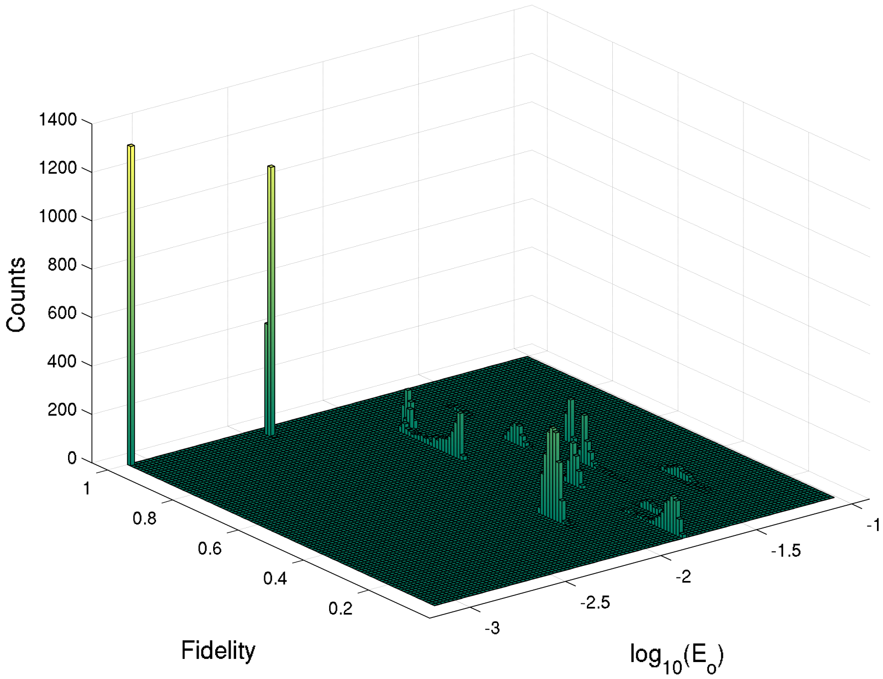
X_j denotes the j-th slit region for $j = 1, \ldots, D$ and $C = \sqrt{\sum_j \tilde{A}_j^2}$ is a normalization constant.

3.4 Reinitialization and post-selection of estimates

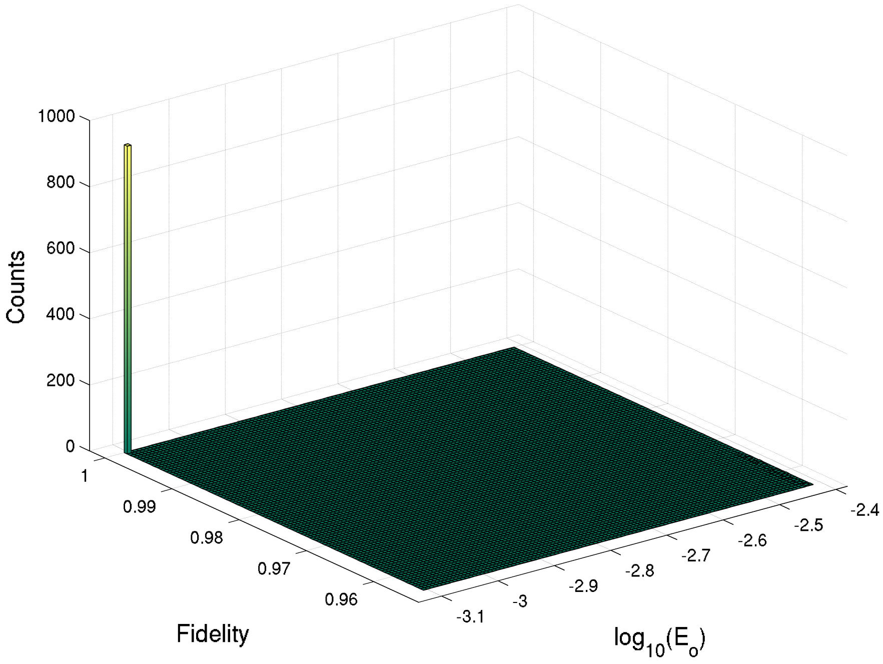
As we have seen in section 2.1.2, the GSA is weakly convergent: an iteration of this algorithm cannot make the error E_o increase. This characteristic makes the GSA both safe, in the sense that it will not worsen any estimate g in terms of E_o, and vulnerable to local minima, since it cannot climb out an error well in case it falls in.

One possibility to avoid this difficulty is to run the algorithm several times with the same data, using a different initial phase estimate each time. One would expect that the algorithm does not get caught in the same local minimum in all the runs, hence one can collect several final estimates and select the one with the smaller error E_o. The reinitialization and post-selection process is illustrated in figure 3.1, which we will explain in better detail now.

We have chosen *one* $D = 10$ qudit state at random and ran the algorithm several



(a)



(b)

Figure 3.1: Solutions found by our algorithm when simulating a $D = 10$ qudit state with (a) no reinitializations (10000 runs) (b) 19 reinitializations per run (1000 runs); note the reduction in the fidelity and error ranges in the second figure. Using this procedure elliminated the spurious solutions and took the algorithm to find only the global solution, as desired.

times with the corresponding simulated data. First, we run the algorithm 10000 times and just saved the estimates it arrived when the stopping criterium was met (absolute change in the error being less than 10^{-8}; the errors themselves were of the order of 10^{-3}). There was no selection of the estimates whatsoever in this first test, shown in figure 3.1(a), as we were trying to discover how serious the problem of local minima was. After the algorithm converged to an estimate, we calculated its fidelity $\digamma$ (section 1.4) with the original state, and made a histogram of fidelities and errors. One can see that there are several count peaks across the histogram, implying that the algorithm got stuck at several estimates that are local minima. The estimate with $E_o \approx 10^{-3.5}$ and $\digamma \approx 1$, the error global mimimum and fidelity maximum, was still the most probable peak, being reached about 1400 times in the 10000 runs.

Next we repeated the process, but now selecting the best among 20 estimates (which means to reinitialize the algorithm 19 times). In this manner we could check how well the post-selection strategy would perform, in particular if this number of reinitializations would be reliable. Also, we only saved 1000 estimates instead of 10000, because of the increase in the computing time. As figure 3.1(b) shows, this strategy yielded the error global minimum estimate very reliably, as the histogram has now only one peak. Most importantly, the error global minimum was also the fidelity global maximum (being very close to 1), so that this approach in fact helped the algorithm to arrive at better state estimates.

As a final remark, we note that a histogram such as the one in figure 3.1(a) can aid in estimating how many reinitializations are needed for the algorithm to have a given reliability figure. If the global minimum peak has a fraction p of all the estimates (in our case, for example, p amounted roughly to 0.14, as the peak had 1400 estimates in a total of 10000), and if we admit that the algorithm converges independently to each local minimum (which is plausible, since reinitializing it amounts to taking another random phase estimate), then the probability that no estimate in N_{est} will fall in the global minimum will be

$$P(\text{no global min.}|N_{\text{est}}) = (1-p)^{N_{\text{est}}}, \tag{3.10}$$

which is monotonically decreasing with N_{est}. Hence, if a reliability figure δ is desired, one needs to select among a number of estimates such that $1 - P(\text{no global min.}|N_{\text{est}}) > \delta$, which translates to

$$N_{\text{est}} > \frac{\log(1-\delta)}{\log(1-p)}. \tag{3.11}$$

3.5 Some results with simulated data

Before running the algorithm with our experimental data, we used simulated data to check if it was working properly. In fact, the adaptations we described so far were developed during this stage. Our greatest worry was that the sampled region in the Fourier domain was too small, and the data we would have to feed the algorithm would be insufficient for it to find good estimates.

In this stage we fed the algorithm data corresponding to perfectly rectangular slits and their corresponding interference pattern, without including imprefections from realistic experiental situations such as vignetting [38] from the lenses or detection noise. Another aspect of the simulated data was its width: the simulated amplitude vectors had 1268 pixels. However, the experimental data was more limited, having only 608 pixels in the Fourier-domain amplitude vector. We will discuss how we dealt with this in the next section. Figure 3.2 depicts a typical result obtained with the algorithm at this stage. For

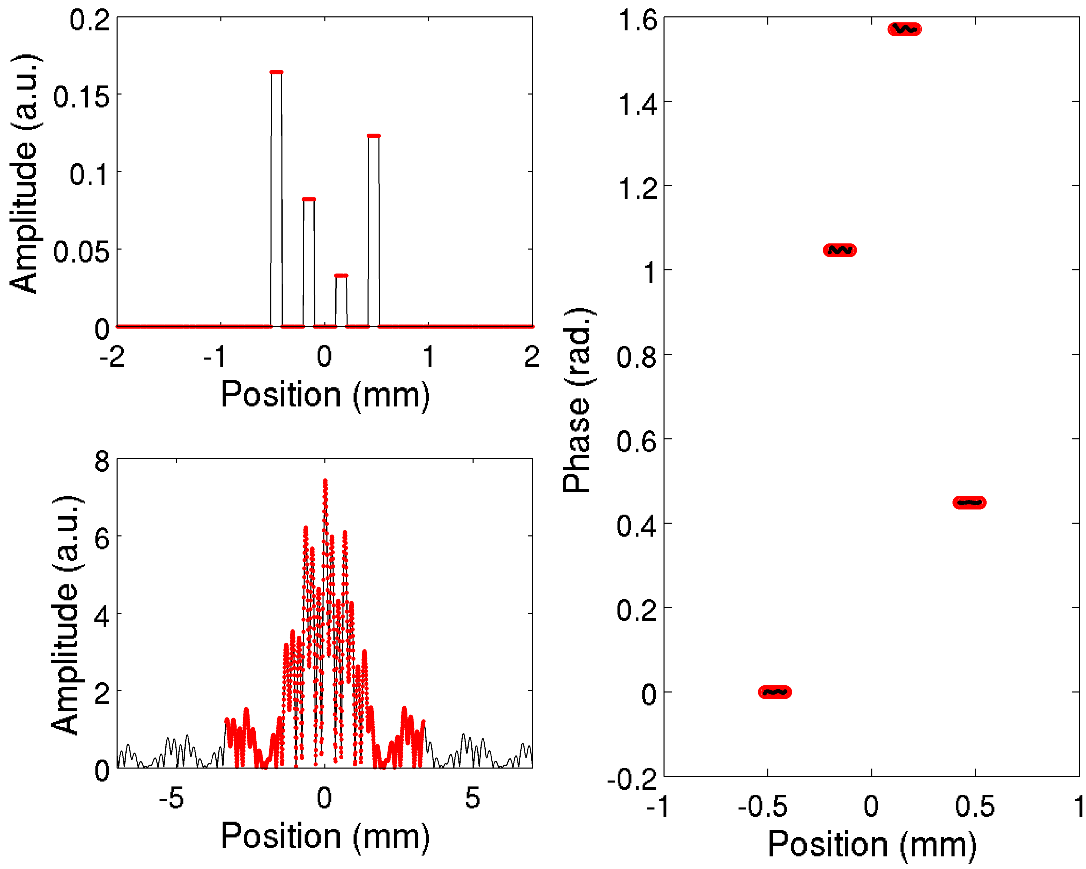


Figure 3.2: Result with simulated data for a $D = 4$ state: object-domain amplitudes (*upper left*), Fourier-domain amplitudes (*lower left*) and object-domain phases (*right*). Simulated data is in *red dots* and the final algorithm estimate is in *black lines*. It used a total of about 4500 iterations in 19 reinitializations, or about 225 iteration for each one.

this specific run, the fidelity between the state estimated by the algorithm and state that was being simulates was

$$\mathcal{F} = 0.9999999924,$$

which was quite high.

Another test that we made at this point was to randomly generate 1000 different $D = 10$ qudit states, feed each one to the algorithm (configured to use 15 reinitializations at the time) and record the fidelities of the estimated states with respect to the generated states. Figure 3.3 shows two histograms of these fidelities.

The average fidelity among all the estimates was $\mathcal{F}^{(\text{ave})} = 0.980$, while its median was $\mathcal{F}^{(\text{median})} = 0.999999697$. One could think that, since we were using *ideal* data, an average of 0.980 fidelity was actually not a good performance. However, the difference between these values suggests that there were some spurious, lower-fidelity results which decreased the average, but were not representative of the whole collection of results. One can in fact see this in the left panel of figure 3.3: There are many lower-fidelity bins with very low count, and one high-fidelity bin which concentrated most of the results.

Therefore we ruled out the results having a fidelity of less than 0.995 and histogrammed the remaining ones. This margin can look exaggeratedly demanding, but we still had 926 of the 1000 results in this quota. The averaged and median fidelities in these selected results were $\mathcal{F}^{(\text{ave})}_{\text{filt}} = 0.99999962$ and $\mathcal{F}^{(\text{median})}_{\text{filt}} = 0.99999972$, much closer to the expected value of 1. These results gave us confidence that the algorithm was working properly and could recover the prepared states, so we proceeded to using experimental data.

3.6 Fourier amplitudes magnification

When we ran the algorithm with experimental data, it often converged to estimates with phase profiles showing peaks and high standard deviation inside the slit regions, while spatial qudit fields are constant both in phase and amplitude inside them. In addition, the results often showed Fourier amplitudes at high frequencies quite higher than what was expected. These two deviations are shown in the left column of figure 3.4, for which we ran the algorithm with the qubit state $\frac{1}{\sqrt{2}}\left[\begin{smallmatrix}1\\ i\end{smallmatrix}\right]$. From what we have seen before, one can infer that the phase deviations [shown in figure 3.4(e)] have some counterpart in the Fourier amplitudes; indeed, by comparing the figures 3.4(a) and 3.4(c), which shows the theoretical amplitudes, one can see that the amplitudes of the algorithm's result do not decay as fast as expected.

Presumably these deviations were due to the small region of measurement in the Fourier domain, from which we had images only 608 pixels wide. Since our routine normalized the total intensity in the object plane to unity, the total intensity in the

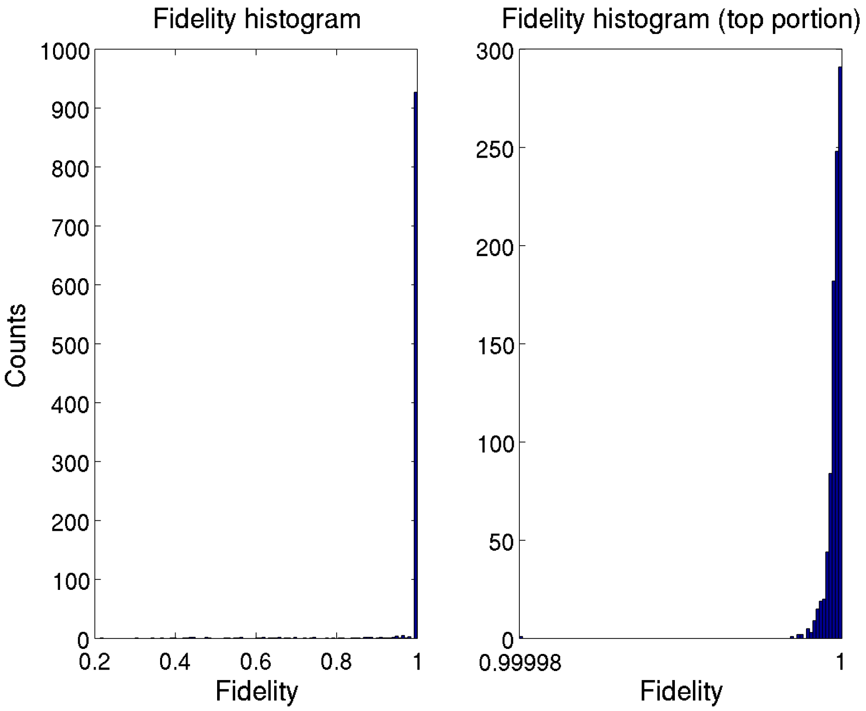
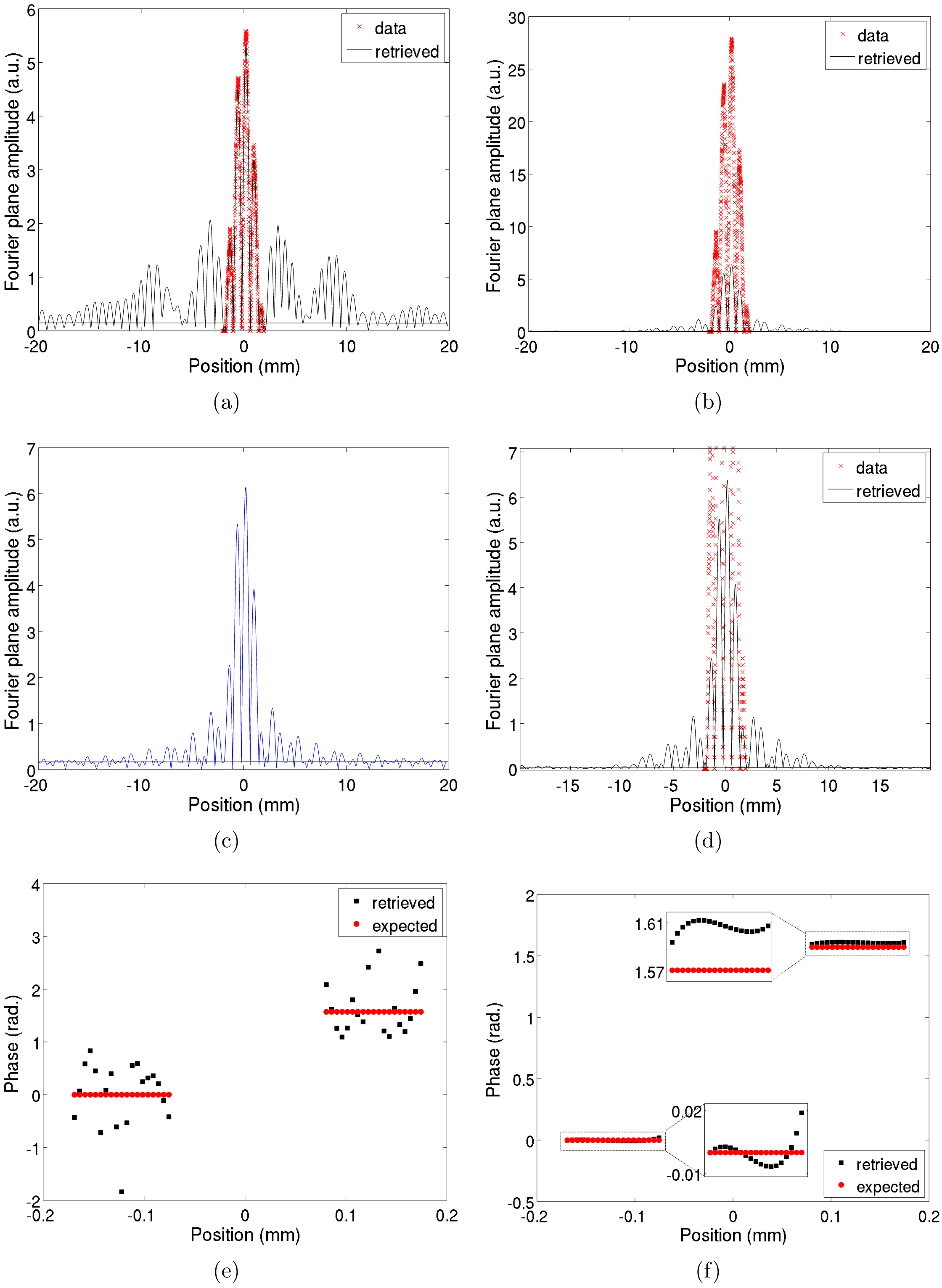


Figure 3.3: Histograms of the recovered fidelities for 1000 $D = 10$ random states: (*left*) all the 1000 results; (*right*) top 926 results. At this point, the algorithm was arriving at very satisfactory results with ideal data.

Fourier plane was also limited (because of Parseval's theorem); we inferred that the small measurement region, and thus small region of amplitudes imposition in the Fourier plane, made it difficult for the algorithm to move amplitudes at higher spatial frequencies back to the low frequency region.

We found that magnifying the Fourier plane amplitudes helped the algorithm converge to better behaved results. We increased the measured amplitudes five (for $D = 2$) and tenfold (for $D = 3, 4, 7, 9$), and used the increased amplitudes in the phase retrieval routine. This was a way of forcing the amplitudes back into the low-frequency region: at step (iii) of the phase retrieval routines (section 2.1), this procedure would give the intensity in the central region of the spectrum a higher increment, at the cost of violating the total intensity in the Fourier domain. This violation would be compensated, however, at the beginning of the next iteration, when making the new estimate g_{k+1}.

The right column of figure 3.4 shows the results obtained after the magnification: (b) shows the magnified and the retrieved Fourier amplitudes; (d) shows a zoom of these amplitudes, revealing that the retrieved amplitudes behaved much more similarly to the expected [shown in (c)]; (f) shows the retrieved phases in the two slit regions. In the phase figure, there are also zooms to show their behaviour in better detail. We can see that there was an oscillating tendency in both slits, plus a positive shift in the second,



(a) (b) (c) (d) (e) (f)

Figure 3.4: Typical profiles of results with experimental data (red dots: measured amplitudes, black line: converged amplitudes): amplitude profiles in Fourier domain without (a) and with (b) magnification; (c) expected Fourier-domain amplitudes; (d) zoom of (b), for comparison with (c); phase profiles inside slit regions without (e) and with (f) magnification. The fourier domain magnification enabled the algorithm to arrive at much better results when using experimental data, as can be seen in the right column.

but both still happening at a small scale. These results illustrate the improvement that the Fourier-amplitudes magnification brought.

3.7 Final remarks

Before we proceed to the results obtained, let us make some final remarks on the configuration we used with the algorithm.

We found the input-output algorithm (see section 2.2) often made the estimate diverge, and did not work properly. After several tries, we gave up using it, and remained with the Gerchberg-Saxton and the output-output algorithms. This was perhaps due to the different philosophies of the algorithms: while Gerchberg-Saxton and output-output algorithms use the current output as the new input, the input-output algorithm sees input and output separately, with the input not necessarily being close to a solution. Perhaps the input-output variation was thus incompatible with the other routines for our problem. We found, through the procedure described also in section 2.2, that the best value for β to be used with the output-output algorithm was $\beta = 1.3$.

Our routine used 20 reinitializations for each state, and in each of them it proceeded with 50 initial GSA iterations, followed by alternating 10 output-output and 10 Gerchberg-Saxton iterations until the absolute change in the error was of 10^{-8}, and finally 200 more GSA iterations before reinitializing the current estimate. It took typically about a total of 4000 iterations for an estimate of a state to be reached, corresponding to about 30 s in our modest-power computer (2 GHz dual core processor). The more robust computer we have in our lab (1.4 GHz quadruple core processor) could achieve the same results in about half the time.

Chapter 4

Experiment and results

In this chapter we will present and discuss the results obtained for the experimental reconstruction of spatial qudits states using our numerical routine, based in the phase retrieval algorithms presented in chapter 2, and with the modifications discussed in chapter 3 (our source code can be found in appendix A).

First we describe the experimental setup, then we provide some remarks on the parametrizations we used to the states and on the quantities we analyzed. Finally, we will see the results themselves.

4.1 Experimental Setup

Figure 4.1 shows the experimental setup that we used to generate and characterize spatial qudit states. As demonstrated in [46], it allows one to prepare arbitrary pure states of spatial qudits [equation (1.1)] with a single phase-only spatial light modulator (SLM). We will stay with a general overview of the setup, but the interested reader can find a more detailed description, as well as the theory behind the preparation of states, in [46–48].

Let us divide the setup (figure 4.1) in three parts: (i) the optical path the beam went through until the SLM, (ii) the SLM itself, and (iii) the path it went afterwards, until reaching the two CMOS cameras.

Path until SLM

In the first part, the laser beam goes through a set of neutral density filters, a beam expander and spatial filter, a polarizer and is then reflected by a mirror and a 50-50 beam splitter (BS) to finally impinge on the SLM:

- the *neutral density filters* just absorb some light, thus allowing us to control the total intensity of the beam that goes forth in the setup; it was important for preventing

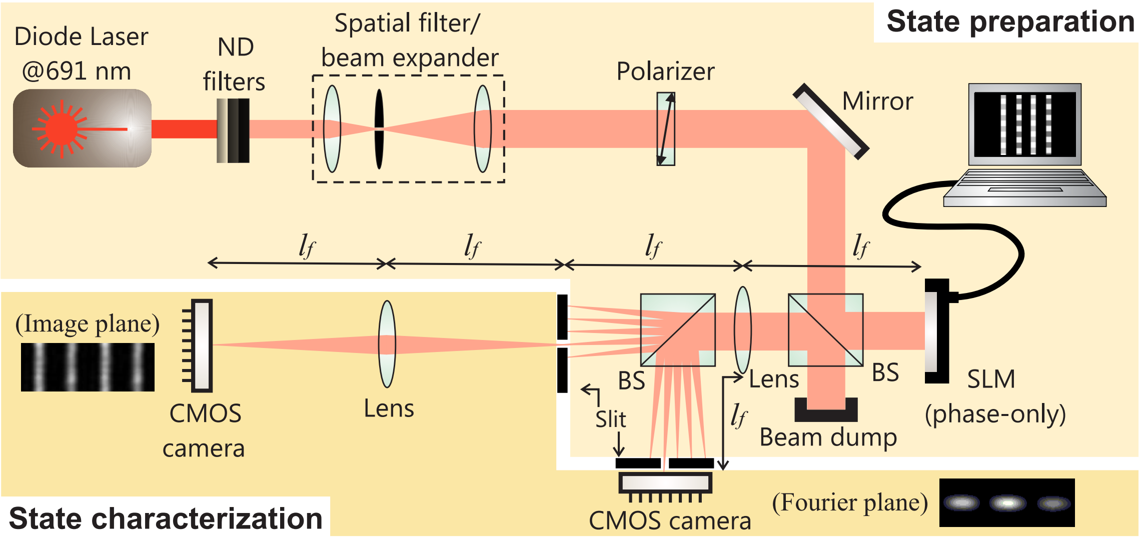


Figure 4.1: Our experimental setup; see text for details.

saturation in the images of the cameras;

- the *beam expander* is composed of two lenses of 10 and 40 cm focal lengths in a confocal arrangement; it magnified the collimated beam by a factor of 4, so that we could illuminate the SLM more uniformly;

- the *spatial filter* was a 50 μm pinhole placed at the confocal plane of the beam expander; it blocked higher spatial frequency components of the transverse profile of the beam, leaving it more more gaussian-like;

- the *polarizer* fixed the polarization of the beam in the vertical direction (the working polarization of the SLM);

- the *mirror* just bent the beam path;

- the *beam splitter* made the beam impinge normally on the SLM, which reflected the beam back.

SLM

The phase-only SLM (Holoeye PLUTO) we had at our disposal was of the normal-incidence, reflecting type. It is a thin layer of liquid crystal fixed on top of a reflective silicon matrix that is divided in an rectangular array of many pixels (in our case, a 1920 × 1080 array of square pixels with 5.2 μm sides). The silicon matrix can have electrical voltages individually set at each pixel region, which causes the liquid crystal to expand in the normal direction. The higher the voltage, the greater the expansion [49].

The portion of the beam that impinges on a given pixel, going through the liquid crystal, reflecting at the matrix and going through the liquid crystal once more, will thus travel through an amount of material that depends on the voltage that was being applied. As the liquid crystal has a higher refraction index than the air's, the net result will be that part of the beam gaining a phase that can be controlled by the voltage on the pixel!

The SLM can therefore be used to give the transverse profile of the input beam a position-dependent phase gain. This device is operated with a computer, with the user setting a grayscale image that adresses the desired voltage at each pixel, with higher gray values being converted to higher voltages. An example of the phase profiles we used to generate the spatial qudit states is shown in figure 4.2, with a blazed diffraction grating being formed inside each slit region and a constant zero-phase being set in the background.

The part of the beam that impinged on the background was simply reflected. However, the phase patterns in the slit regions acted as diffraction gratings, and the part of the beam the impinged on these regions got diffracted into several orders. A spatial filter at the focal plane of the lens after the SLM selected only the light in the +1 diffraction order, and blocked all the other orders as well as the light that was just reflected from the background. But to understand why this was desired, we have to look at the remaining part of the setup.

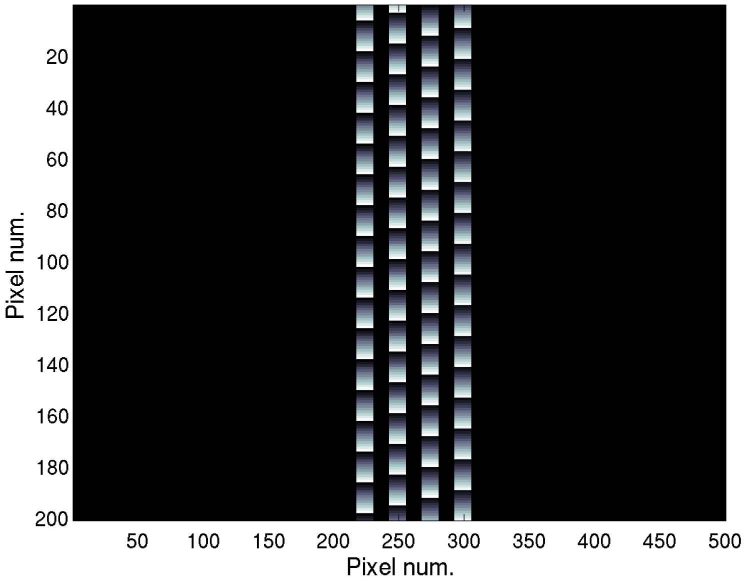


Figure 4.2: Example of phase mask used in the qudits generation; this mask was used to prepare the $D = 4$ state $\frac{1}{2}[1,\ i,\ -1,\ -i]$.

Path after SLM

After impinging on the SLM and passing through the BS once again, the beam went through a lens and a second BS, which branched the optical path in two; the reflected branch is spatially filtered by a slit and then ended in one of the cameras right afterwards, while the transmitted branch was almost identical, except for having a lens between the spatial filter and the camera:

- the *lens* in between the two BSs is taking a Fourier transform of the beam profile modulated by the SLM (see discussion at the beginning of section 3.1): it was placed with its input focal plane on the SLM, and the spatial filter was placed at its output focal plane (there are two because of the second BS);

- the *beam splitter* is just making another copy of the beam that is coming from the SLM; one half of the beam wax sent to a camera that would measure $|F|^2$ and the other half to a camera for $|f|^2$;

- the *slits*, acting as *spatial filters*, were blocking all the diffraction orders that came from the slit regions in the SLM, except for order $+1$; therefore the light that went forth in the setup came only from the regions with diffraction gratings (namely, the slit regions) in the SLM;

- the *camera* in the reflected path measured the intensity of the diffraction-filtered Fourier transform of the field immediately after the SLM; that is, it measured the Fourier-transform intensities of the D-slit field and thus provided the $|F|$ data to be fed into the phase retrieval algorithm);

- the *lens* after the spatial filter was taking another Fourier transform (input focal plane at spatial filter, camera at output focal plane), so that the field at the camera corresponded to the diffraction-filtered field just leaving the SLM, namely the D-slit field;

- the *camera* in the transmitted path measured the object-domain intensities, which were just the D-slit field (from which we got $|f|$ to be fed into the algorithm).

Therefore, the images of the slit fields were formed by only allowing the light that was diffracted at the SLM to reach the camera, and of course, using the SLM in a manner that only light in the desired slit regions was diffracted. The light that impinged on the background regions was simply reflected and further blocked at the spatial filter.

Two other interestings points in this setup are the manner in wich the slit phases and amplitudes were controlled. In order to control the amplitudes, we used different grating formats at each slit region, as each format spreads the beam with different intensities at the several diffraction orders. Therefore, by varying the maximum value of the gratings, we controlled how much light was scattered into the +1 order. On the other hand, to control the phase of each slit we needed only to sum a constant phase along the phase region. This turns out to be equivalent to displacing vertically the grating of a slit region with respect to the first, as one can note from figure 4.2

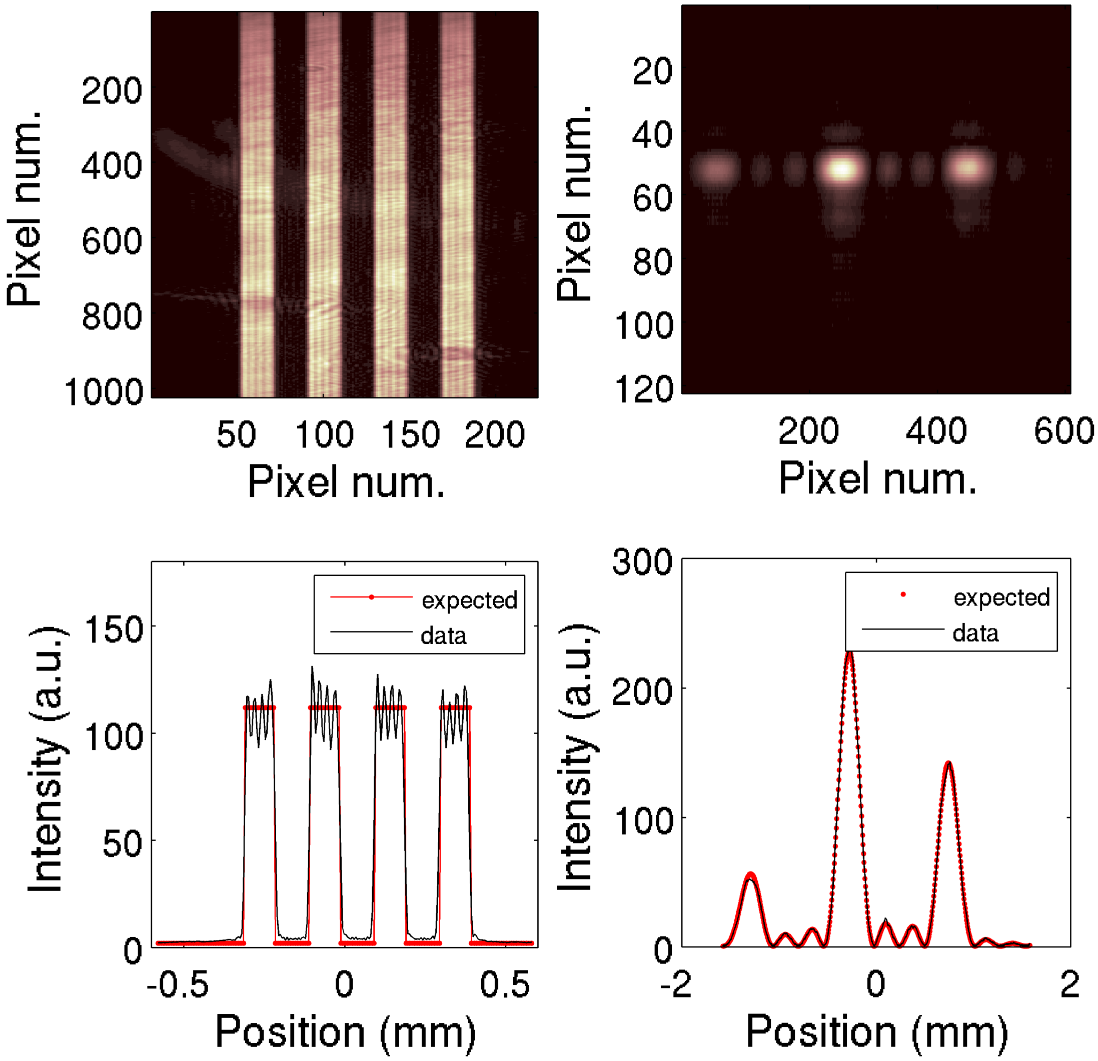


Figure 4.3: *Upper panels*: camera shots of the object (left) and Fourier (right) planes for the $D = 4$ state prepared with the phasemask shown in figure 4.2. *Lower panels*: data obtained from them by summing in the vertical direction, which played the role of $|f|$ and $|F|$ in the algorithm, respectively.

The upper panels of figure 4.3 show typical images of the two cameras: D-slit field (left) and its Fourier transform (right). The last step before feeding the algorithm was to sum the images in the vertical direction, so as to obtain one-dimensional intensities. The square roots of these one-dimensional profiles were then fed to the computer program.

4.2 Intermediate remarks

4.2.1 State parametrization

Qubit states $(D = 2)$

We first ran the algorithm with qubits. Using a basis $\{|1\rangle, |2\rangle\}$, an arbitrary pure state of a qubit can be written as

$$|\psi^{(2)}\rangle = c_1 |1\rangle + c_2 |2\rangle , \tag{4.1}$$

with c_1 and c_2 being two complex numbers satifying $|c_1|^2 + |c_2|^2 = 1$. We can also regard the phase of the coefficient c_1 as a global phase, which will not affect the state ψ, and therefore remove it. Thus it is possible to write this state in a manner that satisfies these two conditions automatically as:

$$|\psi^{(2)}\rangle = \cos\frac{\theta}{2} |1\rangle + e^{i\phi} \sin\frac{\theta}{2} |2\rangle , \tag{4.2}$$

with $\theta \in [0, \pi]$ and $\phi \in [0, 2\pi]$. Here θ and ϕ can be thought as the two angular spherical coordinates, and the space of states can be viewed as the three-dimensional sphere with unit radius, often called the *Bloch sphere* [9]. This was the parametrization we used when generating qubit states in our experiment.

Qutrit states $(D = 3)$

For qutrits there is a similar parametrization with two angular coordinates and two phases:

$$|\psi^{(3)}\rangle = \sin\frac{\theta_1}{2} \cos\frac{\theta_2}{2} |1\rangle + e^{i\phi_1} \sin\frac{\theta_1}{2} \sin\frac{\theta_2}{2} |2\rangle + e^{i\phi_2} \cos\frac{\theta_1}{2} |3\rangle , \tag{4.3}$$

with $\theta_1, \theta_2 \in [0, \pi]$ and $\phi_1, \phi_2 \in [0, 2\pi]$.

We did not, however, use arbitrary qutrit states. In fact, we only used the so-called symmetric states, as the primary focus of the experiment at that time was on the generation and discrimination of non-orthogonal symmetric states of qudits [39]. A set of N states $\{|\phi_k\rangle\}_{k=0}^{N-1}$ is said to be symmetric if there exists a unitary operator $\hat{U}$ for which [50]

$$|\phi_k\rangle = \hat{U} |\phi_{k-1}\rangle = \hat{U}^k |\phi_0\rangle , \tag{4.4}$$

$$|\phi_0\rangle = \hat{U} |\phi_{N-1}\rangle , \tag{4.5}$$

and the qutrit states we have used in our experiment had a form slightly different than

(4.3). The phases ϕ_1 and ϕ_2 were not arbitrary; instead, the states were of the form

$$|\psi_k^{(3)}\rangle = \sin\frac{\theta_1}{2}\cos\frac{\theta_2}{2}\,|1\rangle + \omega_{(3)}^{k}\sin\frac{\theta_1}{2}\sin\frac{\theta_2}{2}\,|1\rangle + \omega_{(3)}^{2k}\cos\frac{\theta_1}{2}\,|3\rangle\,, \tag{4.6}$$

where $\omega_{(3)} = e^{i2\pi/3}$, as $N = D = 3$, and $k = 0, 1, 2$. These states are symmetric with respect to the transformation $\hat{U} = \sum_{j=1}^{3}\omega_{(3)}^{j-1}\,|j\rangle\langle j|$.

Qudit states with $D \geq 4$

For dimensions $D \geq 4$ we also used sets of symmetric states [equations (4.4) and (4.5)] for the reasons stated above. The algorithm was applied to reconstruct states of the form[1]

$$|\psi_k(\alpha, j_0)\rangle = \sum_{j=1}^{D}\omega_{(D)}^{k(j-1)}c_j(\alpha, j_0)\,|j\rangle\,, \tag{4.7}$$

for $k = 0, \ldots, D-1$, $\omega_{(D)} = e^{i2\pi/D}$, and

$$c_j(\alpha, j_0) \propto \begin{cases} 1, & \text{if } j < j_0 \\ \sqrt{1 - \sqrt[D]{\frac{j-j_0}{D-j_0}}\alpha}, & \text{if } j \geq j_0 \end{cases}, \tag{4.8}$$

for $j_0 = 1, \ldots, D-1$ and $\alpha \in [0, 1]$. In the formula above, we are omitting the normalization factor of c_j as it is quite cumbersome, thus the proportionality instead of an equality in equation (4.8). These states are symmetric with respect to $\hat{U} = \sum_{j=1}^{D}\omega_{(D)}^{j-1}\,|j\rangle\langle j|$.

4.2.2 Fidelity analysis

For each generated state, we assessed the quality of the phase retrieval algorithm result by calculating the fidelity (see section 1.4) between the reconstructed state $|\tilde{\psi}_g\rangle$ and the *target* state (the state which was intended to be generated) $|\psi_T\rangle$:

$$F_T = |\,\langle\psi_T|\tilde{\psi}_g\rangle\,|, \tag{4.9}$$

which we will call *target fidelity*. Ideally, the recovered state should have a target fidelity of 1.

[1] This parametrization was defined in [39] in order to represent the discrimination probabilities for dimensions $D \geq 4$ as a surface.

4.3 Results with experimental data

4.3.1 Qubits

Figure 4.4 shows spherical projection plots of the target fidelity [equation (4.9)] and the phase standard deviation of each slit for the spatial qubit states ($D = 2$). The values of θ and ϕ on the projection refer to the state that was being generated, according to equation (4.2). We used 27 values for ϕ and 13 for θ, all equally spaced, for a total of 351 different states around the Bloch sphere.

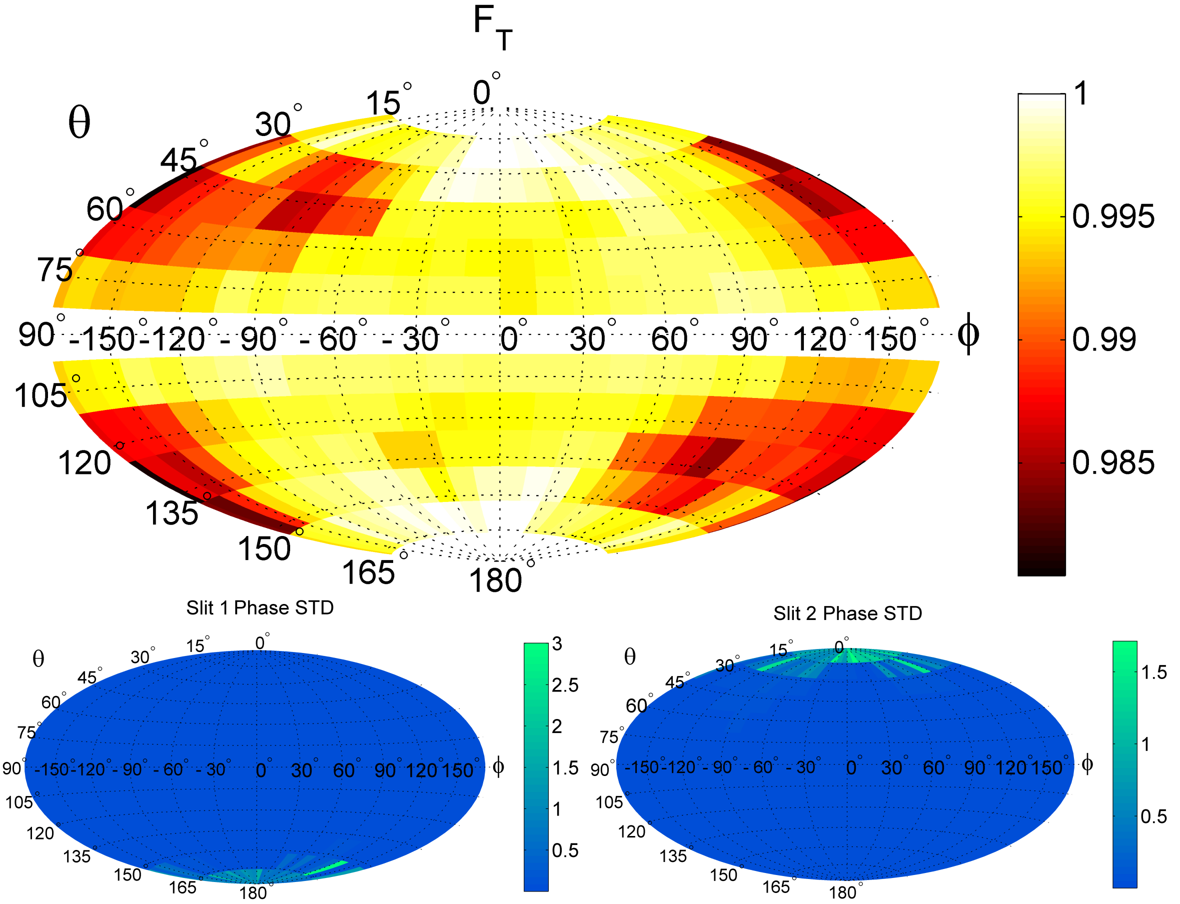


Figure 4.4: Results for $D = 2$.

In the fidelity plot (upper panel), we can see that the highest fidelities occurred close to the poles and to the equator line. High fidelities were already expected near the poles, as one of the slits has zero amplitude in these points and so whatever phase the other slit has would be just a global phase. Among these states, the smallest fidelity that the algorithm arrived at was $F_T^{(\mathrm{min})} = 0.980$, and the average over all the states was of $F_T^{(\mathrm{ave})} = 0.996$, which were quite satisfactory results.

We also calculated the standard deviation of the phase inside each slit region. They are shown in the two lower panels of figure 4.4. One can see that they were rather small for most states, but rose sharply near the poles (near $\theta = 0$ in the first slit and near $\theta = \pi$ in the second). When a slit has zero amplitude, the phase on it just makes no difference in the Fourier-domain intensity pattern, so the algorithm would just be incapable of determining it. The smallest phase standard deviation was $\text{STD}^{(\text{min})} = 0.0003$, and was of this order of magnitude for most states.

These results are summarized in table 4.1.

Table 4.1: Results for $D = 2$.

	D = 2		
	ave.	**min**	**max**
$\mathcal{F}_{\mathbf{T}}$	0.996	0.980	1.000
$\mathbf{STD_1}$	0.1153	0.0003	1.7146
$\mathbf{STD_2}$	0.1095	0.0007	3.0029
No. of states		351	

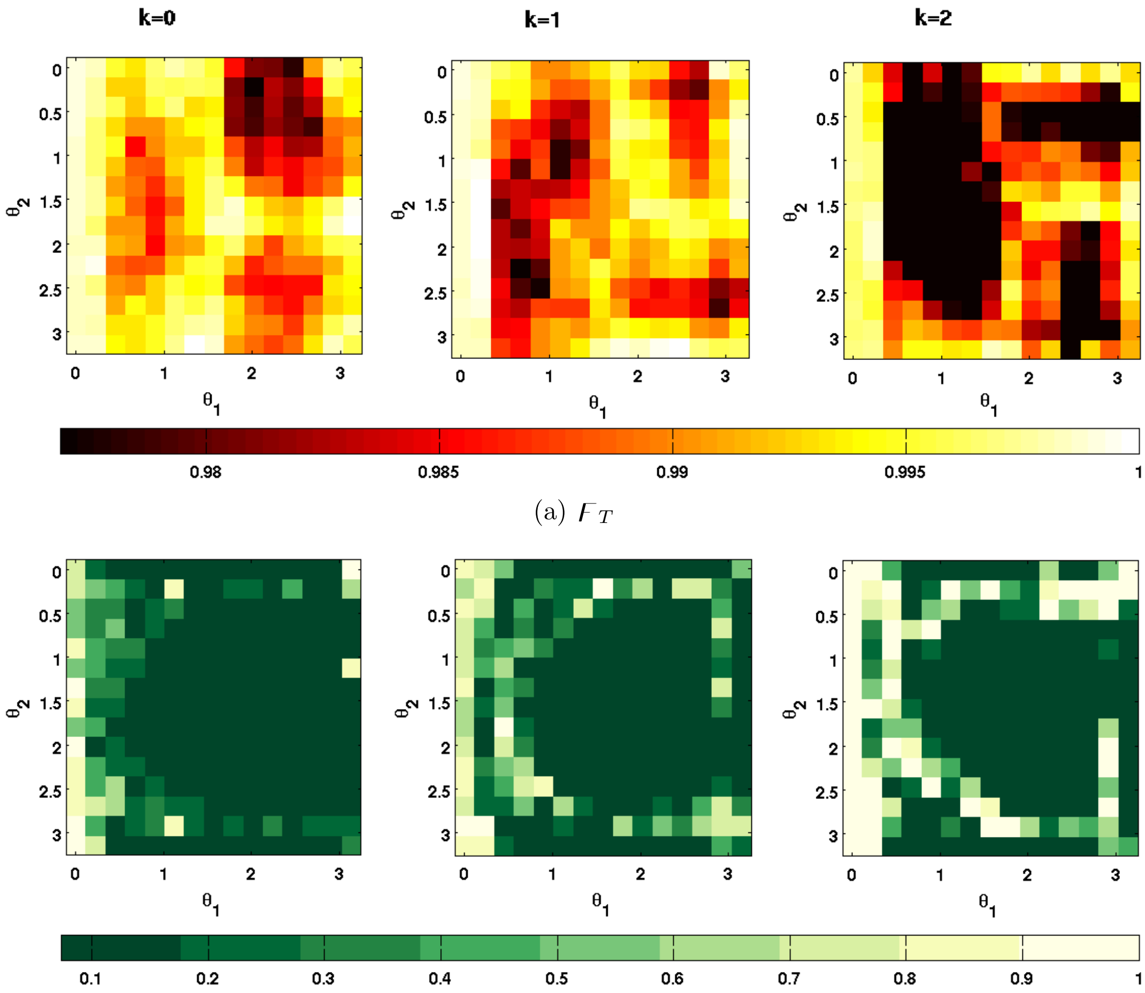
4.3.2 Qutrits

In figure 4.5 we have, for each qutrit state ($D = 3$), the plots of the target fidelity and the greatest phase standard deviation (which we will denote by σ_ϕ^M) among the three slits. The values of θ_1 and θ_2 refer to the state that was prepared, following equation (4.3). There were 15 values for θ_1, 15 for θ_2 and 3 for k, totalizing 675 states. The results for each θ_1 and θ_2 were all very close to one another across the different k values, thus we will present only the average over k of each quantity. In this section we will still present the plots of each quantity separately for each k, so as to illustrate their similarity.

In the target fidelity plot (upper row), we can see that the highest fidelities ocurred near $\theta_1 = 0$ and $\theta_1 = \pi$, where states have nearly zero amplitudes in the first and second slits. Still, the fidelities of the recovered states were quite high overall, with and average of $\mathcal{F}_T^{(\text{ave})} = 0.9919$. The fidelities for $k = 2$ were also somewhat lower than for the other values (~ 0.01 difference), but still compatible.

The values of σ_ϕ^M were rather small for most states, rising near regions where one slit had small amplitude once again. Its minimum value was $\sigma_\phi^{M(\text{min})} = 0.07$, considerably higher than the $D = 2$ case, but still quite acceptable.

These k-averaged results are summarized in table 4.2.



(a) F_T

(b) σ_ϕ^M

Figure 4.5: Results for $D = 3$.

Table 4.2: k-average results for $D = 3$.

	D = 3		
	ave.	**min**	**max**
$F_{\mathbf{T}}$	0.9919	0.9769	0.9997
σ_ϕ^M	0.42	0.07	2.62
No. of states		675	

4.3.3 Qudits with $D = 4, 7$ and 9

For the $D = 4, 7$ and 9 states, the results across the different values of k [see equation 4.7] were all similar, so we will present here the quantities averaged over the k values. Figure 4.6 show the plots of the target fidelity and σ_ϕ^M for each α and j_0 values (see equation 4.8), respectively for $D = 4, 7$ and 9. We used 20 values for α, $D-1$ for j_0 and D for k, in a total of 240, 840 and 1440 states for each of these dimensions.

The fidelity plot (left column) shows that the target fidelities were somewhat smaller for lower values of j_0, and a somewhat periodic behaviour on α. The minimum and average values for $D = 4, 7, 9$ were respectively $\mathcal{F}_T^{(\mathrm{min})} = 0.982$, 0.978, 0.894 and $\mathcal{F}_{ST}^{(\mathrm{ave})} = 0.994$, 0.994, 0.955, showing a gradual decrease as with the dimension.

The values of σ_ϕ^M (right column) were once again small for most states, increasing for higher j_0 values near regions where one slit had small amplitude once again. In fact, they showed an unexpected tendency for $D = 7$ and $D = 9$: After peaking, they became small again for the highest possible value of j_0 and higher α values. The minimum and average values for $D = 4, 7, 9$ were respectively $\sigma_\phi^{M(\mathrm{min})} = 0.006$, 0.004, 0.005 and $\sigma_\phi^{M(\mathrm{ave})} = 0.075$, 0.164, 0.112.

All these results are summarized in table 4.3.

Table 4.3: k-average results for $D = 4, 7, 9$.

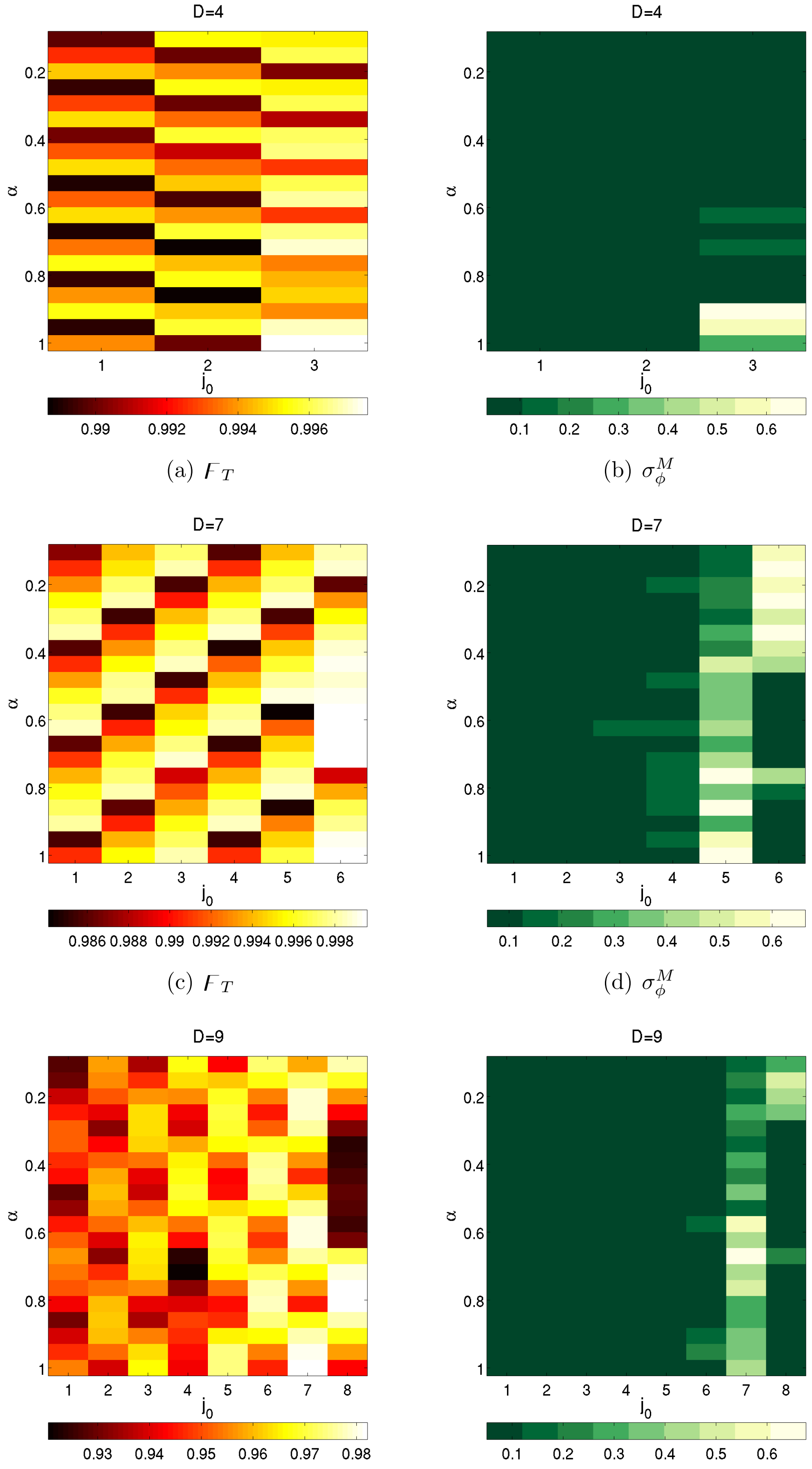
	D = 4			**D = 7**			**D = 9**		
	ave.	**min**	**max**	**ave.**	**min**	**max**	**ave.**	**min**	**max**
$\mathcal{F}_{\mathbf{T}}$	0.994	0.982	0.999	0.994	0.978	1.000	0.955	0.894	0.990
σ_ϕ^M	0.075	0.006	1.133	0.164	0.004	2.660	0.112	0.005	3.002
No. of states		240			840			1440	

4.4 Discussion and final remarks

Overall, the retrieved states were very satisfactory: the average fidelities were $\mathcal{F}_T^{(\mathrm{ave})} = 0.996, 0.992, 0.994, 0.994, 0.955$ for $D = 2, 3, 4, 7, 9$, respectively.

The lower fidelities for $D = 9$ might have been caused by non-uniform illumination of the SLM. Our setup expected a homogeneous illumination, but our laser beam was gaussian, and therefore was less intense at its periphery. Of course, the beam spot was quite wide, so that only its central portion was lighting slit regions in the SLM, so that this imperfection was expected to be small. Still, having the outer slits of a state less illuminated would lead to a deviation in the amplitudes of the prepared states, and the higher the dimension (and thus the number of slits) of a state, the more serious this deviation would be. The minimum fidelities $\mathcal{F}_T^{(\mathrm{min})}$ show this tendency – the lowering of the retrieved fidelities with D – more clearly.

Aside from this, we can conclude that the state generation stage was working properly. All the results we had with the phase retrieval algorithm were directly dependent on the data we fed it with. If the state generation was preparing states too corrupted, it would not be possible to the algorithm to reach reasonable estimates.



(a) F_T

(b) σ_ϕ^M

(c) F_T

(d) σ_ϕ^M

(e) F_T

(f) σ_ϕ^M

Figure 4.6: k-averaged results for $D = 4$ (upper row)), 7 (mid row) and 9 (lower row): F_T (left column) and σ_ϕ^M (right column).

Chapter 5

Conclusions and future perspectives

In this work we have studied phase retrieval algorithms and their applicability to reconstructing pure spatial qudit states. We have built an adapted version of the algorithms found in the literature, fitting it to this encoding scheme. This algorithms has worked well with simulated data, reaching very accurate estimates of the target states. Finally, we realized a proof-of-principle experiment that succesfully demonstrated the reconstruction of over 3500 states with dimensions $D = 2, 3, 4, 7$ and 9, thus showing its feasibility as a tool for reconstructing qudit states.

After the study we have presented so far, we can draw a few conclusions and outline some future perspectives, presented below.

Phase reconstruction of the spatial qudit states was feasible.

The fidelities of the recovered states were quite satisfactory. This means that the recovered phases were very similar to those of the target state, and thus that the phase retrieval alogrithm arrived at accurate estimates.

The state preparation scheme was working properly.

The results of the phase retrieval routine depends entirely on the data we feed into the algorithm. If this data is somehow corrupted (e.g., if the state generation stage is not working properly), it is very unlikely that the routine will arrive at good results. Therefore, we can also conclude that the state preparation scheme used in the experiment was working properly, and only because of that was the phase retrieval able to work.

Lower recovered fidelities were probably due to non-uniform illumination of the SLM.

One tendency in our results was that F_T decreased with the dimension D of the generated state. We believe that this was due to the gaussian profile of the beam illuminating the SLM, while our setup was conceived expecting a uniform illumination. States with greater D have more slits, the outermost of which were illuminated by more peripheral, less intense regions of the beam. This presumably caused a deviation on the generated states.

Magnifying Fourier-domain amplitudes was an effective strategy to compensate the small range of sampled spatial frequencies.

From what we have seen in section 3.6, the small number of pixels in the measurement of $|F|$ made it difficult for the algorithm to arrive at a good estimate. Magnifying the vector of Fourier-domain amplitudes was effective in helping the algorithm reaching accurate phase estimates, but it would be probably better still to have sampled a wider region of spatial frequencies.

Use of two-dimensional spatial encoding schemes.

In this work we converted the two-dimensional images of the spatial qudits into one-dimensional functions by summing in one direction; these were the functions that were fed to the algorithm. However, the phase retrieval algorithm does not demand this conversion, and treating two dimensional data could in fact bring benefits.

By using two dimensional images, we could use this method with spatial qudits encoded in a two-dimensional array of squares instead of a one-dimensional array of slits. These could make the diffractive regions in the SLM better centered, or even achieve higher dimensions more easily. Also, this would make the method able to work with other codification schemes, shuch as those using the orbital angular momentum [21].

Phase retrieval with fractional Fourier transforms.

Versions of the phase retrieval problem involving two planes related by the more general *fractional* Frourier transform have also been considered [51]. There is always the possibility that two different spatial qudit states have the same intensity profiles in the object and Fourier planes, so that the phase retrieval problem has not a unique solution for that case[1]. However, since these states are different, there should be a difference in their profiles at

[1] A dramatic example are the states forming mutually unbiased bases for spatial qudits with $D \geq 3$; there are states within the same basis (and thus orthogonal) that have exactly the same intensity patterns in both the object and Fourier domains.

some intermediate plane, where the propagation would correspond to a fractional Fourier transform instead of the conventional one. Thus, it could be useful to have a fractional version of the algorithm.

Appendix A

Source code

In this appendix we will present the source code used for phase retrieval in our work. There were five phase retrieval functions and one script to coordinate them; we designed the functions in this manner because the best strategy reported in the literature [44] was to keep interchanging a few iterations of each algorithm instead of proceeding with one of them all the time. This code was written in the `Matlab` language.

Though the coordinating script was initially designed to control five different phase retrieval, only two of them proved to work well, namely the Gerchberg-Saxton and the output-output algorithms (see chapter 2). Therefore, we will only show these two routines after the coordinating script.

Besides interchanging the algorithms, the coordinating function also managed the detection of estimate stagnation (with further storing of final estimate and reinitialization of the algorithm). There was a main function that would call this routine and later choose the least-error estimate.

Coordinating routine

```
%Phase Retrieval Algorithm, with "jumps", and outputs to search the best
%no-jump threshold value.
%   This version uses intensity measures of both the object and its
%   Fourier transform!
%
%   If the algorithm sees no significant changes in the error for n_stuck
%   consecutive iterations, it adds some noise to the current phase
%   (and thus "jumps" out of the current local minimum it is stuck at).
%
%   Usage:
%      [Obj_Estimate, Transf_Estimate, Eo, Ejump, g_jump] = pramix_jumps_search(Npre,Nio
    ,Noo,Nhgi,Nhgo,Njumps,...
%      Beta_io,Beta_oo,Beta_hgi,Beta_hgo, Supp, Obj_Phases_init, Obj_Int, Transf_Int)
%
%   Input list:
```

```
%       Obj_Phases_init   current estimate of object phases
%       Obj_Int           object intensity values
%       Transf_Int        transform intensity values
%       N                 number of jumps to be performed
%       N...              number of iterations of each pra
%       Beta_...          beta values for the fienup-family algorithms
%       err_thresh        threshold for the reduction in the error Eo
%
%   Output list:
%       Obj_Estimate      estimate of the object after the N iterations
%       Transf_Estimate
%       Eo              vector of errors in each iteration
%       Ejump             vector of errors at jumps
%       g_jump            matrix wiht the g's at the jumps in its columns

% FIX: phylosophical difference of the "input" methods...

function [Obj_Estimate, Transf_Estimate, Eo, Ejump, g_jump] = pramix_jumps_search_F_part_impos(OutModuliCorr,Npre,Nio,Noo,Nhgi,Nhgo,Ngs,Npost,Njumps,...
    Beta_io,Beta_oo,Beta_hgi,Beta_hgo, Obj_Phases_init, Obj_Amp, Transf_Amp, err_thresh, nslits, slits_rise, slits_fall)

  %ERROR CHECKS

  if(~isreal(Obj_Phases_init))
        error('Phase vector should have only real components (phases)!!')
    end
    if( length(slits_rise)~=nslits || length(slits_fall)~=nslits)
        error('Number of slits rises/falls different than number of slits!')
    end

    %INITIALIZATIONS
  g = Obj_Amp.*exp(1i*Obj_Phases_init);
  Eo = [];
    Ejump = [];
    g_jump = [];
    Supp = [];
    for r=1:1:nslits
        Supp = [Supp slits_rise(r):1:slits_fall(r)];
    end

    %JUMP PARAMETERS
    n_stuck = 40;           %number of "stuck iterations" that will trigger the jump
    noise_level = 1*2*pi;   %noise level to add the phase ("jump amplitude")


    %PHASE RETRIEVAL ALGORITHMS
    %to be repeated until N jumps have occurred

    i_stuck = 0;    % stuck iterations counter
    r = 0;          % jump counter
    i_pra = 1;      % counter to mark which PRA should be used at the current iteration
    Eo = [];
    %g = X;
    [g, G, eo] = gerch_sax_F_part_impos(OutModuliCorr, Npre, angle(g), Obj_Amp,
    Transf_Amp);
    Eo = [Eo eo];

    if(Njumps==0)
        while(i_stuck<n_stuck)
            if(i_pra<Nio)
                [g, G, eo] = input_output_F_part_impos(OutModuliCorr, 1, Beta_io, angle(
    g), Obj_Amp, Transf_Amp);
                Eo = [Eo eo];
            elseif(i_pra<Nio+Noo)
                [g, G, eo] = output_output_F_part_impos(OutModuliCorr, 1, Beta_oo, angle
    (g), Obj_Amp, Transf_Amp);
                Eo = [Eo eo];
            elseif(i_pra<Nio+Noo+Nhgi)
                [g, G, eo] = hgi_F_part_impos(OutModuliCorr, 1, Beta_hgi, Supp, angle(g)
    , Obj_Amp, Transf_Amp);
                Eo = [Eo eo];
            elseif(i_pra<Nio+Noo+Nhgi+Nhgo)
                [g, G, eo] = hgo_F_part_impos(OutModuliCorr, 1, Beta_hgo, Supp, angle(g)
    , Obj_Amp, Transf_Amp);
                Eo = [Eo eo];
            elseif(i_pra<Nio+Noo+Nhgi+Nhgo+Ngs)
                [g, G, eo] = gerch_sax_F_part_impos(OutModuliCorr, 1, angle(g), Obj_Amp,
     Transf_Amp);
                Eo = [Eo eo];
            end
            i_pra = i_pra+1;
            if(i_pra==Ngs+Nio+Noo+Nhgi+Nhgo)
                i_pra = 1;
            end

            % Check if error was stuck in this iteration
            if( length(Eo)>2 && Eo(end-1)-Eo(end)<err_thresh )
               i_stuck = i_stuck+1;
            end
        end

        [g, G, eo] = gerch_sax_F_part_impos(OutModuliCorr, Npost, angle(g), Obj_Amp,
    Transf_Amp);
        Eo = [Eo eo];
        Ejump = [Ejump Eo(end)];
        g_jump = [g_jump g'];
        r = r+1;
    end

  while(r<Njumps)
      % Phase Retrieval Algorithms mix
        if(i_pra<Nio)
            [g, G, eo] = input_output_F_part_impos(OutModuliCorr, 1, Beta_io, angle(g),
    Obj_Amp, Transf_Amp);
            Eo = [Eo eo];
        elseif(i_pra<Nio+Noo)
            [g, G, eo] = output_output_F_part_impos(OutModuliCorr, 1, Beta_oo, angle(g),
     Obj_Amp, Transf_Amp);
            Eo = [Eo eo];
```

```
        elseif(i_pra<Nio+Noo+Nhgi)
            [g, G, eo] = hgi_F_part_impos(OutModuliCorr, 1, Beta_hgi, Supp, angle(g), Obj_Amp, Transf_Amp);
            Eo = [Eo eo];
        elseif(i_pra<Nio+Noo+Nhgi+Nhgo)
            [g, G, eo] = hgo_F_part_impos(OutModuliCorr, 1, Beta_hgo, Supp, angle(g), Obj_Amp, Transf_Amp);
            Eo = [Eo eo];
        elseif(i_pra<Nio+Noo+Nhgi+Nhgo+Ngs)
            [g, G, eo] = gerch_sax_F_part_impos(OutModuliCorr, 1, angle(g), Obj_Amp, Transf_Amp);
            Eo = [Eo eo];
        end
        i_pra = i_pra+1;
        if(i_pra==Ngs+Nio+Noo+Nhgi+Nhgo)
            i_pra = 1;
        end

        % Check if error was stuck in this iteration
        if( length(Eo)>2 && Eo(end-1)-Eo(end)<err_thresh )
            i_stuck = i_stuck+1;
        end

        % Make jump if n_stuck stuck iterations have been achieved
        if( i_stuck == n_stuck )
            [g, G, eo] = gerch_sax_F_part_impos(OutModuliCorr, Npost, angle(g), Obj_Amp, Transf_Amp);
            Eo = [Eo eo];
            Ejump = [Ejump Eo(end)];
            g_jump = [g_jump g'];

            if(r~=Njumps-1)
                %making new phase guess
%                phi = angle(g) + noise_level*rand(1,length(g)); %this was completely random...
                phi = zeros(1,length(g));
                for q=1:1:nslits
                    phi(slits_rise(q):slits_fall(q)) = noise_level*rand;    %this is constant within each slit
                end
                g = Obj_Amp.*exp(1i*phi);

                [g, G, eo] = gerch_sax_F_part_impos(OutModuliCorr, Npre, angle(g), Obj_Amp, Transf_Amp);
                Eo = [Eo eo];
            end
            i_stuck = 0;
            i_pra = 1;
            r = r+1;
           %display( ['Jumped: r=' num2str(r)] )
        end
    end

  Obj_Estimate = g;
    Transf_Estimate = G;
```

```
%endfunction
```

Gerchberg-Saxton routine

```
%Gerchberg-Saxton Phase Retrieval Algorithm
%(actually, a fixed number of iterations of)
%   This version uses intensity measures of both the object and its
%   Fourier transform!
%
%   Usage:
%      [Obj_Estimate, Transf_Estimates, Eo] = gerch_sax_data(N_gs, Obj_Phases_init, Obj_Amp, Transf_Amp, w)
%
%   Input list:
%      Obj_Phases_init  current estimate of object phases
%      Obj_Amp          object amplitude values
%      Transf_Amp       transform amplitude values
%      N_gs             number of iterations
%
%   Output list:
%      Obj_Estimate     estimate of the object after the N iterations
%      Transf_Estimate  estimate of the transform after the N iterations
%      E0               vector of errors in each iteration

%THIS IS THE MATLAB VERSION

%FIX
%-return transform phases?
%-make error checks

function [Obj_Estimate, Transf_Estimate, Eo] = gerch_sax_F_part_impos(OutModuliCorr, N_gs, Obj_Phases_init, Obj_Amp, Transf_Amp)

  %% ERROR CHECKS
  %(maybe later...)

  % Initializations
  X = Obj_Amp.*exp(1i*Obj_Phases_init);
  Eo = [];
    Nfft = size(Obj_Amp,2);
    n = size(Transf_Amp,2);

    if(N_gs==0)
        Obj_Estimate = X;
        Transf_Estimate = fft(X);
        return
    end



    %% GERCHBERG_SAXTON ALGORITHM
    % Remember X and F will have different lengths

    % Calculations for "outside" region modulus correction
    I_F = sum(Transf_Amp.^2);
    Nfft = length(Obj_Amp);
    if( Nfft>I_F )
        c = sqrt(Nfft - I_F);
    else
        c = 0;
    end

  for r = 1:1:N_gs
        %disp(r)
        F = fft(X);
        G = F;
        %F = fftshift(fftshift(fft(fftshift(X))));
    %F(w:w+n-1) = Transf_Int.*exp(1i*angle(F(w:w+n-1)));
        F(1:ceil(n/2)) = Transf_Amp(1:ceil(n/2)).*exp(1i*angle(F(1:ceil(n/2))));
        F(end-floor(n/2):end) = Transf_Amp(end-floor(n/2):end).*exp(1i*angle(F(end-floor(n/2):end)));
        if(OutModuliCorr == 1 && c>0)
            F(ceil(n/2)+1:end-floor(n/2)-1) = c*F(ceil(n/2)+1:end-floor(n/2)-1)/sqrt(sum( abs( F(ceil(n/2)+1:end-floor(n/2)-1) ).^2 ));
        end
    g = ifft(F);
        %g = ifftshift(ifft(ifftshift(ifftshift(F))));
    X = Obj_Amp.*exp(1i*angle(g));

    Eo = [Eo, sum( (abs(g)-Obj_Amp).^2 )/sum( abs(Obj_Amp).^2 )];
    end

%   Obj_Estimate = g;
%   Transf_Estimate = G;
    Obj_Estimate = X;
    Transf_Estimate = fft(X);


%endfunction
```

Output-output routine

```
%Fienup INput-Output Phase Retrieval Algorithm
%(actually, a fixed number of iterations of)
%   This version uses intensity measures of both the object and its
%   Fourier transform!
%
%   Usage:
%      [Obj_Estimate, Transf_Estimate, Eo] = output_output(N, Beta, Obj_Phases_init, Obj_Int, Transf_Int)
%
%   Input list:
%      Obj_Phases_init  current estimate of object phases
%      Obj_Int    object intensity values
%      Transf_Int transform intensity values
%      N    number of iterations
%
%   Output list:
```

```
%      Obj_Estimate     estimate of the object after the N iterations
%      Transf_Estimate  estimate of the transform after the N iterations
%      EO               vector of errors in each iteration

%THIS IS THE MATLAB VERSION

%FIX
%-return transform phases?
%-make error checks

function [Obj_Estimate, Transf_Estimate, Eo] = output_output_F_part_impos(OutModuliCorr,
     N, Beta, Obj_Phases_init, Obj_Amp, Transf_Amp)

  %ERROR CHECKS (... maybe later)

    % Calculations for "outside" region modulus correction
    I_F = sum(Transf_Amp.^2);
    Nfft = length(Obj_Amp);
    if( Nfft>I_F )
        c = sqrt(Nfft - I_F);
    else
        c = 0;
    end

  %J. R. FIENUP ALGORITHM
  Eo = [];
  X = Obj_Amp.*exp(1i*Obj_Phases_init);
    Nfft = size(Obj_Amp,2);
    n = size(Transf_Amp,2);

    if(N==0)
        Obj_Estimate = X;
        Transf_Estimate = fft(X);
        return
    end

  for r = 1:1:N
    F = fft(X);
        G = F;
    %G(w:w+n-1) = Transf_Int.*exp(1i*angle(G(w:w+n-1)));
        G(1:ceil(n/2)) = Transf_Amp(1:ceil(n/2)).*exp(1i*angle(G(1:ceil(n/2))));
        G(end-floor(n/2):end) = Transf_Amp(end-floor(n/2):end).*exp(1i*angle(G(end-floor
    (n/2):end)));
        if(OutModuliCorr == 1 && c>0)
            G(ceil(n/2)+1:end-floor(n/2)-1) = c*G(ceil(n/2)+1:end-floor(n/2)-1)/sqrt(sum
    ( abs( G(ceil(n/2)+1:end-floor(n/2)-1) ).^2 ));
        end
    g = ifft(G);
    Delta_g = ( 1 - Obj_Amp./abs(g) ).*g;
    X = g - Beta*Delta_g;

    Eo = [Eo, sum( (abs(g)-Obj_Amp).^2 )/sum( abs(Obj_Amp).^2 )];
    end

%   Obj_Estimate = g;
%   Transf_Estimate = F;
```

```
    Obj_Estimate = X;
    Transf_Estimate = fft(X);

%endfunction
```

Bibliography

[1] J R Fienup. **Phase retrieval algorithms: a personal tour**. *Applied Optics* **52**, 45 (2013).

[2] H P Stahl. **Rules for optical metrology**. *International Commission for Optics 22nd General Congress* (2011).

[3] T Zielinski. **Wave front sensing for large telescopes**, Phase Retrieval & Image Science research group at Rochester University page.

[4] **Vital facts about the JWST**. NASA's webpage about the James Webb Space Telescope.

[5] J A Marozas. **Fourier transform–based continuous phase-plate design technique: a high-pass phase-plate design as an application for OMEGA and the National Ignition Facility**. *Journal of the Optical Society of America A* **24** 74 (2007).

[6] J R Fienup. **Iterative method applied to image reconstruction and to computer-generated holograms**. *Optical Engineering* **19**, 193297 (1980).

[7] W O Saxton. ***Computer Techniques for Image Processing in Electron Microscopy***. *Academic Press* (1978).

[8] R P Millane. **Phase retrieval in crystallography and optics**. *Journal of the Optical Society of America A* **7** 394 (1990).

[9] M Nielsen and I Chuang. ***Quantum Computation and Quantum Information***. *Cambridge University Press* (2000).

[10] C H Bennett and S J Wiesner. **Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states**. *Physical Review Letters* **69** 2881 (1992).

[11] C H Bennett, G Brassard, C Crépeau, R Jozsa A Peres, and W K Wootters. **Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels**. *Physical Review Letters* **70** 1895 (1993).

[12] N Gisin, G Ribordy, W Tittel, and H Zbinden. **Quantum cryptography**. *Reviews of Modern Physics* **74** 145 (2002).

[13] L K Grover. **Quantum Mechanics Helps in Searching for a Needle in a Haystack**. *Physical Review Letters* **79** 325 (1997).

[14] A Peres. **Generalized Kochen-Specker Theorem**. *Foundations of Physics* **26** 807 (1996).

[15] R P Feynman. **Simulating physics with computers**. *International Journal of Theoretical Physics* **21** 467 (1982).

[16] J J García-Ripoll, P Zoller, and J I Cirac. **Speed Optimized Two-Qubit Gates with Laser Coherent Control Techniques for Ion Trap Quantum Computing**. *Physical Review Letters* **91** 157901 (2003).

[17] A Blais, Ren-Shou Huang, A Wallraff, S M Girvin, and R J Schoelkopf. **Cavity quantum electrodynamics for superconducting electrical circuits: An architecture for quantum computation**. *Physical Review A* **69** 062320 (2004).

[18] D Loss and D P DiVincenzo. **Quantum computation with quantum dots**. *Physical Review A* **57** 120 (1998).

[19] J L O'Brien, A Furusawa, and Jelena Vuckovic. **Photonic quantum technologies**. *Nature Photonics* **3** 687(2009) .

[20] C Ottaviani, D Vitali, M Artoni, F Cataliotti, and P Tombesi. **Polarization Qubit Phase Gate in Driven Atomic Media**. *Physical Review Letters* **90** 197902 (2003).

[21] S Franke-Arnold, L Allen, and M Padgett. **Advances in optical angular momentum**. *Laser and Photonics Review* **2** 299 (2008).

[22] I Marcikic, H de Riedmatten, W Tittel, V Scarani, H Zbinden, and N Gisin. **Time-bin entangled qubits for quantum communication created by femtosecond pulses**. *Physical Review A* **66** 062308 (2002).

[23] L Neves, G Lima, J G A Gomez, C H Monken, C Saavedra, and S. Padua. **Generation of Entangled States of Qudits using Twin Photons**. *Physical Review Letters* **94** 100501 (2005).

[24] S Etcheverry, G Cañas, E S Gómez, W A T Nogueira, C Saavedra, G B Xavier, and G Lima. **Quantum key distribution session with 16-dimensional photonic states**. *Scientific Reports* **3** 2316 (2013).

[25] H Bechmann-Pasquinucci and A Peres. **Quantum Cryptography with 3-State Systems**. *Physical Review Letters* **85** 3313 (2000).

[26] D Collins, N Gisin, N Linden, S Massar, and S Popescu. **Bell Inequalities for Arbitrarily High-Dimensional Systems**. *Physical Review Letters* **88** 040404 (2002).

[27] B Marques, M R Barros, W M Pimenta, M A D Carvalho, J Ferraz, R C Drumond, M T Cunha, and S Pádua. **Double-slit implementation of minimal Deutsch algorithm**. *Physical Review A* **86** 032306 (2012).

[28] P Kolenderski, U Sinha, L Youning, T Zhao, M Volpini, A Cabello, R Laflamme, and T Jennewein. **Aharon-Vaidman quantum game with a Young-type photonic qutrit**. *Physical Review A* **86** 012321 (2012).

[29] G Borges, M Carvalho, P L de Assis, J Ferraz, M Araújo, A Cabello, M T Cunha, and S Pádua. **Quantum contextuality in a Young-type interference experiment**. *Physical Review A* **89** 052106 (2014).

[30] B Marques, A A Matoso, W M Pimenta, A J Gutiérrez-Esparza, M F Santos, and S Pádua. **Experimental simulation of decoherence in photonics qudits**. *Scientific Reports* **5** 16049.

[31] J M Renes, R Blume-Kohout, A J Scott, and C M Caves. **Symmetric informationally complete quantum measurements**. *Journal of Mathematical Physics* **45** 2171 (2004).

[32] M Paris and J Rehacek. ***Quantum State Estimation***. *Springer-Verlag* (2004).

[33] G Lima, F A Torres-Ruiz, L Neves, A Delgado, C Saavedra, and S Pádua. **Measurement of spatial qubits**. *Journal of Physics B* **41** 185501(2008).

[34] G Taguchi, T Dougakiuchi, N Yoshimoto, K Kasai, M Iinuma, H F Hofmann, and Y Kadoya. **Measurement and control of spatial qubits generated by passing photons through double slits**. *Physical Review A* **78** 012307 (2008).

[35] W M Pimenta, B Marques, M A D Carvalho, M R Barros, J G Fonseca, J Ferraz, M T Cunha, and S Pádua. **Minimal state tomography of spatial qubits using a spatial light modulator**. *Optics Express* **18** 24423 (2010).

[36] G Lima, L Neves, R Guzmán, E S Gómez, W A T Nogueira, A Delgado, A Vargas, and C Saavedra. **Experimental quantum tomography of photonic qudits via mutually unbiased basis**. *Optics Express* **19** 3542 (2011).

[37] L E Ballentine. ***Quantum Mechanics: A Modern Development***. *World Scientific Publishing* (2000).

[38] J W Goodman. ***Introduction to Fourier Optics***. *McGraw-Hill* (1998).

[39] M A Solís-Prosser, M F Fernandes, O Jiménez, A Delgado, and L Neves. **Experimental minimum-error quantum-state discrimination in high dimensions**. *Physical Review Letters* **118**, 100501 (2017).

[40] M A Solís-Prosser. **Estudio de la medición de qubits y qudits espaciales codificados en fotones individuales y aplicaciones**. Master's thesis, *Universidad de Concepción* (2011).

[41] E Osherovich. ***Numerical methods for phase retrieval***. PhD thesis, *Israel Institute of Technology* (2011).

[42] J L deLyra. ***Transformadas de Fourier***. *Livraria da Física* (2014).

[43] R W Gerchberg and W O Saxton. **A practical algorithm for the determination of phase from image and diffraction plane pictures**. *Optik* **35**, 237 (1972).

[44] J R Fienup. **Phase retrieval algorithms: a comparison**. *Applied Optics* **21**, 2758 (1982).

[45] J R Fienup. **Reconstruction of an object from the modulus of its Fourier transform**. *Optics Letters* **3**, 27 (1978).

[46] M A Solís-Prosser, A Arias, J J M Varga, L Rebón, S Ledesma, C Iemmi, and L Neves. **Preparing arbitrary pure states of spatial qudits with a single phase-only spatial light modulator**. *Optics Letters* **38**, 4762 (2013).

[47] M A Solís-Prosser. **Optimización de la fidelidad de preparación de qubits y qudits espaciales mediante SLMs en régimen de modulación de sólo-fase**, private communication (2012).

[48] J J M Varga, L Rebón, M A Solís-Prosser, L Neves, S Ledesma, and C Iemmi. **Optimized generation of spatial qudits by using a pure phase spatial light modulator**. *Journal of Physics B* **47** 225504 (2014).

[49] C Maurer, A Jesacher, S Bernet, and M Ritsch-Marte. **What spatial light modulators can do for optical microscopy**. *Laser & Photonics Reviews* **5**, 81 (2011).

[50] A Chefles and S M Barnett. **Optimum unambiguous discrimination between linearly independent symmetric states**. *Physics Letters A* **250** 223 (1998).

[51] Wen-Xiang Cong, Nan-Xian Chen, and Ben-Yuan Gu. **Recursive algorithm for phase retrieval in the fractional Fourier transform domain**. *Applied Optics* **37** 6906 (1998).

[52] J R Fienup. **Reconstruction of a complex-valued object from the modulus of its Fourier transform using a support constraint**. *Journal of the Optical Society of America* **4**, 118 (1987).

[53] G Grynberg, A Aspect, and C Fabre. ***Introduction to Quantum Optics: From the Semi-classical Approach to Quantized Light***.