Molecular movies and geometry reconstruction using Coulomb explosion imaging

by

Ali Ramadhan

A thesis
presented to the University of Waterloo
in fulfillment of the
thesis requirement for the degree of
Master of Science
in
Physics

Waterloo, Ontario, Canada, 2017

AUTHOR DECLARATION

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I understand that my thesis may be made electronically available to the public.

ABSTRACT

Coulomb explosion imaging is a technique of imaging the structure of small molecules in the gas phase and their ultrafast dynamics by inducing the rapid ionization and dissociation of the molecule into its constituent atomic fragments. The momentum vectors of the atomic fragments facilitate the retrieval of the molecule's structure, however, few attempts at geometry reconstruction appear in the published literature, whose vague methodology casts serious doubts on the geometry reconstructions that have been performed, and motivating the need for an investigation into the feasibility of geometry reconstruction. We develop a method for the fast and precise reconstruction of triatomic molecular geometries by casting the task as a nonlinear constrained optimization problem. We use this method to investigate the uncertainty in geometry reconstructions as a function of measurement uncertainty as well as the existence and nature of multiple solutions to the geometry reconstruction problem. We map out the conditions under which molecular geometries may be accurately reconstructed and propose a framework for reconstructing geometries, and therefore producing molecular movies using Coulomb explosion imaging.

ACKNOWLEDGMENTS

I feel extremely priveleged to have had the opportunity to work on a problem of personal interest to me with sufficient financial support to put a roof over my head as well as food (and drink) on the table. To that end I would like to first and foremost take this opportunity to thank the Canadian taxpayers for their important investment in scientific research in these tough economic times, which has funded my research and education on multiple occasions. I hope I will be able to do more to give back in the future.

I would like to of course express my gratitude to my supervisor, Joseph Sanderson, who took me into his lab as a complete newbie and who provided me with numerous exciting (and sometimes endlessly frustrating) research experiences as well as for not kicking me out after I broke an expensive neutral density filter wheel (expensive on a grad student's budget). I definitely learnt a lot in my time here.

I would like to thank my fellow graduate student Benji Wales for being a perpetual source of free research ideas without which this thesis would have looked quite different, and Reza Karimi for being a genuinely cool and always helpful dude.

I sincerely thank Dr. Shoja'eddin Chenouri of the Department of Statistics & Actuarial Science and the Statistical Consulting and Collaborative Research Unit for his extensive support and guidance in the development of a rigorous Bayesian inference approach to geometry reconstruction. I regret not working harder to see the project through to the end.

I would also like to thank Dr. James Martin for his extensive support and mentorship over the past few years, and for the many discussions which I believe have shaped my interests. I also thank Dr. Wing-Ki Liu, giver of the most difficult undergraduate exams, for generously providing the computational resources which made a large part of this thesis possible. I thank Dr. Marcel Nooijen for originally proposing the idea of using a lookup table to reconstruct molecular geometries which started this research project. I also thank Dr. Donna Strickland for being on my thesis committee and for her willingness to share her infinite yet practical laser wisdom.

I have learned a great deal about science and research from very wise collaborators and through numerous side projects over the past few years (unrelated to this thesis) and from whom I have learned a tidy amount.. To that end, I would like to thank Michael MacDonald and Lucia Zuin (Canadian Light Source, Saskatchewan), Haruo Shiromaru and Jun Matsumoto (Tokyo Metropolitan University), Tomonari Wakabayashi (Kindai University, Osaka), Khaled Ibhrahim (Waterloo

Institute for Nanotechnology), and Nhat Ly (Department of Mechanical and Mechatronics Engineering).

Judy McDonnell and Anja Drygala have helped tremendously with the administrative side of being a grad student, thank you!

I would like to thank everyone involved in the production of the television series *The Wire* and *Breaking Bad* for providing excellent and thoughtful procrastination material, without which this thesis would have been finished earlier.

Part of this work was made possible by the high-performance computing facilities and the support staff of the Shared Hierarchical Academic Research Computing Network (SHARCNET: www.sharcnet.ca) and Compute Canada.

I gratefully acknowledge financial support provided by the Natural Sciences and Engineering Research Council of Canada (NSERC) through their Undergraduate Student Research Awards (USRA) Program and their Canada Graduate Scholarships Master's (CGS M) Program.

DEDICATION

Look mom and dad! I got my name on a thesis!

Also dedicated to you, the reader, for reading 0.05% of this thing!

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1

INTRODUCTION

To image the structure and dynamics of a molecule by destroying it seems paradoxical, yet that is precisely what Coulomb explosion imaging (CEI) does. Exposing a molecule to the intense electric field of an ultrashort laser pulse causes it to rapidly ionize and dissociate into its constituent atomic fragments, in the case of complete dissociation, termed a *Coulomb explosion*. The momentum vectors of the "atomic shrapnel" contain a wealth of information about the molecule and its dissociation, thus by measuring them the molecule's original geometry may be reconstructed. This is the basic idea behind CEI. The laser pulse may be used to induce specific molecular dynamics then dissociation at particular times, either through the temporal lengthening of the pulse or the use of a pump then a probe pulse, which CEI can image forming a series of frames constituting a *molecular movie*.

MOTIVATION AND RESEARCH PROBLEM

A cursory review of CEI research articles would suggest that the reconstruction of molecular geometries is a well-understood process. For example, Xu et al. (2016) writes, "CEI is an effective and straightforward tool to retrieve molecular structural properties, requiring no prior assumptions about the molecule", and Matsuda, Takahashi, and Hishikawa (2014) states that "Coulomb explosion provides a direct access to the instantaneous structure of the target molecule" (Matsuda, Takahashi, and Hishikawa, 2014). Vager (2001), in an older review article of CEI, writes that "simultaneous determination of the final fragments velocity vectors from each single molecule, which is what CEI detectors do, resolves the 3D initial conformation", however the article contains no molecular geometries or structures. This is true of the majority of research articles employing CEI which tend to study the momentum vectors themselves, inferring molecular dynamics and structure from the kinetic energy spectra of the atomic fragments, and from the arrangement of the vectors through the use of Newton plots and Dalitz plots (Ramadhan et al., 2016). For the few that do attempt to determine the molecular structures, the methodology is vague, sometimes even reduced to a single sentence, such as "we find a three-dimensional structure that reproduces the measured fragment velocities" (Légaré et al., 2005b).

In this thesis, we will attempt to address this knowledge gap by developing a computational framework for geometry reconstruction using CEI, with the aim of providing fast, accurate reconstructions

and quantifying the uncertainty on these geometry reconstructions. The problem of geometry reconstruction may be thought of as an ill-posed *inverse problem* with no analytic solution or iterative process that guarantees a solution. However, inverse problems in science are routinely tackled effectively using mathematical optimization techniques, which we employ, and Bayesian inference, which we will propose and discuss.

THESIS OUTLINE

Chapter 2 provides a brief overview of CEI including the method's aspirations and an experimental outline as well as a review of past geometry reconstruction attempts.

Chapter 3 details the multi-step process of measuring the momentum vectors for each atomic fragment following a Coulomb explosion, and the quantification of measurement uncertainty for the momentum vectors. We will also perform some exploratory data analysis of the measurements then discuss how to computationally simulate a Coulomb explosion using Hamiltonian mechanics. Finally, we will detail some necessary conventions for describing molecular geometries and momentum vectors.

Chapters 4 and 5 constitute the main portion of this thesis, detailing the two different approaches taken to geometry reconstruction. Chapter 4 discusses the lookup table approach and its motivation to supersede a previous approach relying on the Nelder-Mead simplex method. A lookup table is implemented and used to perform geometry reconstruction using the OCS molecule as an example, and to study the existence of *degenerate geometries*. Chapter 5, motivated by the drawbacks of the lookup table, formulates the task of geometry reconstruction as an optimization problem, which is tackled using nonlinear constrained optimization algorithms. Some theory from mathematical optimization is introduced to understand the methods employed, and an implementation in MATLAB is used to reconstruct the OCS molecule and the two reconstructions are compared. The optimization approach also allows for the further investigation of degenerate geometries.

Chapter 6 addresses the important task of uncertainty quantification for reconstructed geometries, which has not been addressed by previous studies. A heuristic approach is employed at first and used to provide insights into the effects of uncertainty. A more rigorous and sophisticated approach in the Bayesian inference framework is discussed and suggested as a next step.

2

COULOMB EXPLOSION IMAGING

Coulomb explosion imaging (CEI) is a technique for studying the structure and ultrafast dynamics of small molecules in the gas phase, essentially by ionizing the molecule to induce fragmentation after which the positively-charged fragments repel each other in a *Coulomb explosion*[1] and the momentum vector of each fragment is measured. In principle, it is possible to reconstruct the molecular structure with knowledge of the momentum vector. We will develop and implement multiple general methods for doing this in the following chapters, however we will slowly come to see the problem's pathological nature. On the other hand, the momentum vectors may themselves be studied to infer molecular dynamics and changes in molecular structure.

In this section we will briefly discuss what makes CEI an appealing technique and provide a cursory description of an experimental CEI apparatus. This is followed by a review of molecular geometry reconstructions performed using CEI, which are rather few in number.

2.1 AN APPEALING PROPOSITION

This does not sound like a technique of interest in the 21st century—the structure of virtually all small gas molecules is well known from spectroscopic measurements, so what is left for CEI to tell us about? One main goal would be the production of *molecular movies* that image the ultrafast dynamics of small molecules as a function of time, lending significant insights into important ultrafast chemical and biochemical reactions. Once these ultrafast chemical processes are imaged and understood, it may become possible to control them (Miller

1 It is worth mentioning that the concept of a Coulomb explosion is independent of CEI and refers to a cluster of many atoms repelling each other under their mutual Coulomb repulsion following ionization, for example by the intense electromagnetic field of a short laser pulse. Interestingly, a Coulomb explosion seems to be the mechanism responsible for the explosive reaction of alkali metals, such as sodium or potassium, with water, finally explaining the chemistry behind the classic high school experiment (Mason et al., 2015).

et al., 2010; Weinstein and Hunt, 2012; Zewail, 2000). One recent example is the imaging of proton migration using acetylene as a target molecule by laser-induced CEI (Ibrahim et al., 2014) and x-ray core ionization (Liekhus-Schmaltz et al., 2015).

Another application is determining the absolute (or stereochemical) configuration of chiral molecules that cannot be crystalized and probed by x-rays, first proposed by Kitamura et al. (2001). Chiral molecules are of particularly great interest in biochemistry (Gardner, 2005). Recent studies have looked at, for example, bromochlorofluoromethane (CHBrClF) (Pitzer et al., 2013) as well as *cis* and *trans* molecular isomers of 1,2-dibromoethene (Ablikim et al., 2016). CEI can also provide information about non-classical molecular structures that elude other methods, such as helium trimer systems (Voigtsberger et al., 2014) and the quantum halo state of He_2 (Zeller et al., 2016) as well as image processes such as quantum unidirectional rotation (Mizuse et al., 2015).

X-ray diffraction can also provide detailed information regarding the structure of small molecules such as mercury fulminate (Beck et al., 2007), whose structure eluded determination for over 200 years and which was famously featured in "*Crazy Handful of Nothin'*", the sixth episode of the hit television series *Breaking Bad* during Heisenberg's first meeting with Tuco.

Of course, it must be acknowledged that x-ray diffraction results have contributed an absolutely incredible amount of scientific knowledge—any accolade would be an understatement (Hendrickson, 1995). However, I find a certain appeal in the direct structure measurement aspect of CEI. Certainly at least one author became interested with the prospect of *imaging molecules* using "lasers and [Coulomb] explosions". CEI directly answers the question of "what are we made of and what does it look like?" first posed by the ancient Greeks and Indians, most famously by Democritus and his mentor Leucippus, who may have been fictional (Taylor, 2010).

The idea of the four classical elements: earth, air, fire, and water was proposed by many cultures to explain natural phenomena and the complexity of matter in terms of simpler entities with some explanations being tied to atomism, the idea that matter was made of tiny, indivisible entities, such as Plato's association of the Platonic solids with the classical elements (see figure 1) first written about in his dialogue *Timaeus* circa 360 B.C. (Cornford, 2014). While the ideas evolved over time, it was not until the 1600's that the theory was subject to experimental verification and eventually completely disproved (Maxwell, 1873). Even today, the classical elements still feature prominently in popular media.

One barely feels the sea breeze blowing during the sun's rise every morning, as if the air was made up of smooth octahedrons. But not as smooth as the almost spherical icosahedron which rolls and flows,

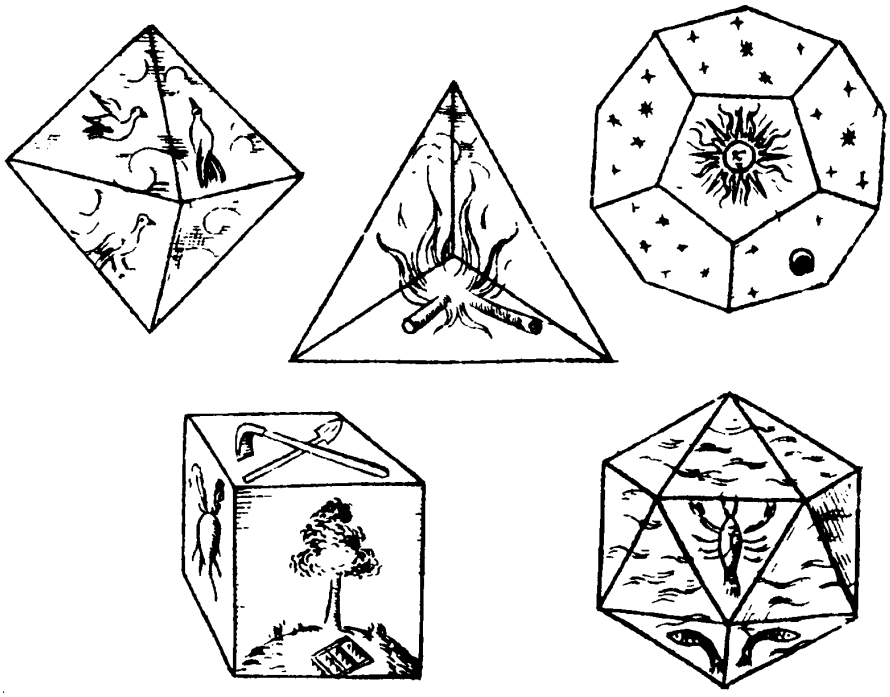

Figure 1: The classical elements associated with the five Platonic solids. Clockwise from the top left: the octahedron with air, tetrahedron with fire, dodecahedron with the universe, the icosahedron with water, and the cube with earth. Figure from Kepler (1619, Book 2, p. 53). English translation available (Kepler et al., 1997).

just like the water contained in all the Earth's rivers and oceans. The sharp pain felt in close proximity to a warm crackling fire would sting like the pointy ends of a tetrahedron. The cube is the only Platonic solid to tessellate and fill Euclidean space, giving it the solidity and stability of the unchanging earth. The remaining Platonic solid, the dodecahedron, then must form everything else in the heavens. Just as there is beauty in the simplicity of the Platonic solids and classical elements, there is beauty in the simplicity of Coulomb explosion imaging.

2.2 EXPERIMENTAL OUTLINE

CEI must be performed under high-vacuum conditions and so the molecule must be ionized and detected within a position-sensitive time-of-flight (PSToF) spectrometer that lies inside a vacuum chamber. Figure 2 shows a very basic schematic of a CEI PSToF spectrometer used by Ramadhan et al. (2016). The molecules of interest may be introduced simply as an effusive gas jet but for laser CEI, a well-collimated supersonic jet of cold molecules is highly desired to increase the resolution with which the time-of-flight and position measurements may be made (Dörner et al., 2000) as well as prepare molecules with a known velocity distribution.

In pump-probe CEI the ultrashort laser pulse is split into two pulses through the use a beam splitter. The first is called the *pump pulse* and is usually weaker than the second, the *probe pulse*. A time delay τ between the pulses is created by exposing the probe pulse to a longer optical path length such that the pump arrives at the molecule at time $t = 0$ followed by the probe pulse at time $t = \tau$. The job of the pump pulse is to initiate some change in the molecule, ideally a change that we are interested in imaging. One important example would be an isomerization of the molecule (Ibrahim et al., 2014; Liekhus-Schmaltz et al., 2015). Thus the pump pulse *pumps* the molecule into some excited state.

The job of the more powerful probe pulse is to engulf the molecule in an intense enough electric field such that multiple electrons (≥ 2) are stripped off of it. The molecule is left in a highly charged and unstable state where the chemical bonds between the individual atoms cannot hold it together any longer, and the molecule dissociates. It may dissociate completely into its constituent atoms or dissociate partially into a mixture of atoms and molecular fragments. The fragments are left in a highly-charged state (although some neutral fragments may be produced) and begin to behave as individual point charges in a weak Coulombic potential. The entire process occurs in the presence of a constant electric field and so the positively-charged ions accelerate upwards towards the PSToF detector, which allows for

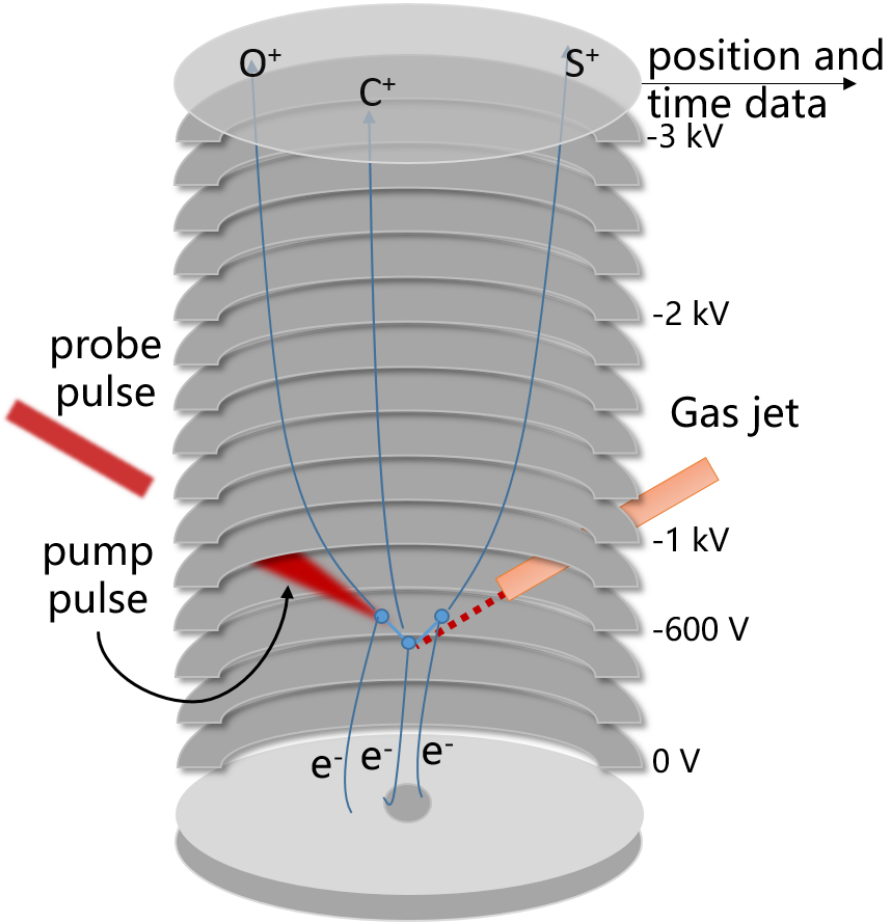

Figure 2: Basic schematic of a position sensitive time-of-flight spectrometer. A cross-section of the rings is shown, which set up the constant electric field by virtue of being biased to different voltages through the use of a resistor chain. The size of the example triatomic molecule is *greatly* exaggerated for illustrative purposes. Note that electrons will be accelerated downwards, and may be detected using a separate second detector. This figure was adapted from an earlier version meant to accompany Ramadhan et al. (2016).

the measurement of their momentum vectors (section 3.1). Thus the probe pulse allows for the *probing* of the excited state.

The time delay τ between the pulses may be varied to image the dynamics induced by the pump pulse at various times. Knowledge of the molecular structure at each time delay allows for the production of a series of frames showcasing the dynamics, or a *molecular movie*.

An alternative to pump-probe CEI uses a single laser pulse to both pump and probe the molecule. In this case, the laser pulse length is varied (not the time delay between two laser pulses) acting like the shutter speed of a camera. This allows for the imaging of ionization dynamics, and the variable pulse length allows for some control over the ionization process. Karimi, Liu, and Sanderson (2016) provide a review of this technique, termed Femtosecond Multiple Pulse Length Spectroscopy (FEMPULS).

For details on the physical principles, Posthumus (2004) provides a detailed review on the dynamics of small molecules in intense laser fields.

2.3 MOLECULAR GEOMETRY RECONSTRUCTIONS USING CEI

The original CEI experiment is usually traced back to Vager, Naaman, and Kanter (1989) in which the Coulomb explosion is initiated by passing a molecular beam through a thin foil. This may be because it was the first work suggesting that full molecular structures may be recovered by measuring the velocity (or momentum) vectors of the atomic fragments, and even reported on a non-classical molecular structure. However, previous works utilizing CEI do exist, even some significant works that report on molecular structures (Kanter et al., 1979).

Ultrashort laser pulses[2]as a means of inducing Coulomb explosions made their entrance in the 1980's where they were utilized to infer molecular dynamics using covariance mapping (Frasinski, Codling, and Hatherly, 1989). Highly charged ion impact is another method of inducing a Coulomb explosion, and was first done in the 1990's in parallel with the development of more sophisticated coincidence mapping techniques. Since then, the laser has emerged as the more popular tool and has further developed the coincidence mapping technique. There do exist other methods of inducing Coulomb explosions, for example, single photons from a synchrotron source utilizing the Auger effect, x-ray pulses from a free-electron laser source, or electron collision.

In this section we will trace the history of CEI back to the 1970's where it started with foil-induced fragmentation. We will then follow it's development to the present day where ultrashort laser pulses are the most popular means of performing CEI. Throughout we will

2 In 1987, ultrashort would be referring to 0.6 ps laser pulses (Frasinski et al., 1987).

focus solely on the achievements of CEI in determining molecular structures, and in creating molecular movies using these recovered structures.[3]

Interestingly, the first-ever mention of the term "Coulomb explosion" in the published literature comes from an unrelated study of the fine structure of singly ionized helium by Novick, Lipworth, and Yergin (1955). They measured the energy difference of the $2\,^2S_{1/2}$ and $2\,^2P_{1/2}$ states of ionized helium as a sensitive test of quantum electrodynamics. Coulomb explosion (or space charge explosion) was the dominant ion removal mechanism which they accounted for in modeling the quenching rate[4]of metastable $2\,^2S_{1/2}$ ions by radio frequency radiation to describe the observed resonance lineshapes (spending two appendices on it).

2.3.1 *Foil-induced dissociation*

CEI was first performed by passing a molecular beam containing the molecular ion of interest from a storage ring through a thin atomic film. While in the solid film, the probability for Coulomb scattering of the individual atomic nuclei is small due to their small size and consequently small interaction cross-section. The electrons will be scattered to very wide angles due to their interaction with the many electron clouds in the film. This process rapidly ionizes the molecule, typically within the first few atomic layers, or the first femtosecond. This time scale is faster than the characteristic times scale for molecular vibration ($> 10^{-14}$ s) and rotation ($> 10^{-12}$ s). The now highly ionized molecule exits the foil and rapidly breaks up into its constituent atomic ion fragments which repel each other under their mutual Coulomb repulsion in what is termed a "Coulomb explosion" (Vager, Naaman, and Kanter, 1989). Figure 3 shows a schematic of such an experiment. This specific experiment set out to measure the absolute configuration of atoms in a chiral molecule in the gas phase, which remains challenging. Early foil-induced CEI experiments can be described by this schematic except for the fact that they did not employ a mass selector and used a more primitive but still position-sensitive detector. The atomic fragment trajectories (dashed lines) assume that no rearrangement of the atoms occurs and that the system evolves under a Coulomb potential.

The premise behind foil-induced CEI is that during this explosion, the atoms simply repel each other and do not rearrange, thus preserving the angles between them from the time they exit the foil to

3 Much of the molecular dynamics are inferred in CEI from studying the distribution of the fragment momentum vectors (e. g. through the use of Newton and Dalitz plots) and the distribution of kinetic energy carried away by each fragment. We will be focusing on the original aim of CEI, that is, to measure molecular structures.

4 The term was more popular in decades past but simply means the extinction rate or loss rate of metastable ions.

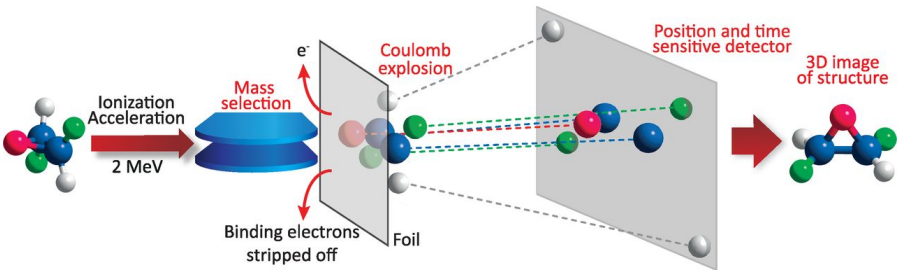

Figure 3: Schematic of a foil-induced Coulomb explosion imaging experiment. From Herwig et al. (2013b). Reprinted with permission from AAAS.

the time they are detected at a position and time-sensitive detector. As the potential energy of each pair of fragments i, j is converted to kinetic energy according to

$$\frac{4q_iq_j}{|\mathbf{r}_i - \mathbf{r}_j|} = \frac{\mu|\mathbf{V}_i - \mathbf{V}_j|^2}{2} \tag{2.1}$$

it suggests that measurement of the asymptotic vector velocities completely defines the initial geometry of the molecule. Here q_i, $\mathbf{r}_i$, and $\mathbf{V}_i$ are the charge, position vector, and velocity vector of the atomic fragment i while μ is the reduced mass of the two-body system of fragments i, j. Vager, Naaman, and Kanter (1989) argues that the density of the measured $\mathbf{r}_i$ vectors is an experimental measurement of the square of the three-dimensional nuclear ground-state wave function, and would additionally describe all possible correlations between the molecule's constituent nuclei.

The earliest example of a molecular geometry recovered using CEI is reported by Gaillard et al. (1978). They used the foil-induced Coulomb explosion to image the structure of the H_3^+ molecular ion, showing that it mainly exhibits an equilateral triangular shape using separate pieces of evidence from three different experiments.[5] Figure 4 gives a few examples of the geometries they recovered.

It is unclear how the idea for such an imaging experiment came to be, however it is worth noting that Gaillard et al. (1978) and coauthors have been studying the effects of molecular beams passing through thin foils for quite some time, mainly at Argonne National Laboratory. See for example their studies of wake potentials generated behind charged particles as they pass through a solid (Gemmell et al., 1975; Vager and Gemmell, 1976) and their study of the dissociation of fast HeH^+ ions traversing thin foils (Vager, Gemmell, and Zabransky, 1976). It seems quite reasonable that studying the dissociation of small molecules and the angular distribution of the atomic

5 It is interesting to note that the experiment was repeated by three separate teams, then reported on coherently in one manuscript. Each team, having access to different equipment, produced separate pieces of data. It was the team in Rehovot, Israel that recorded the projections of the exploded ions while the others measured energy spectra and angular distributions.

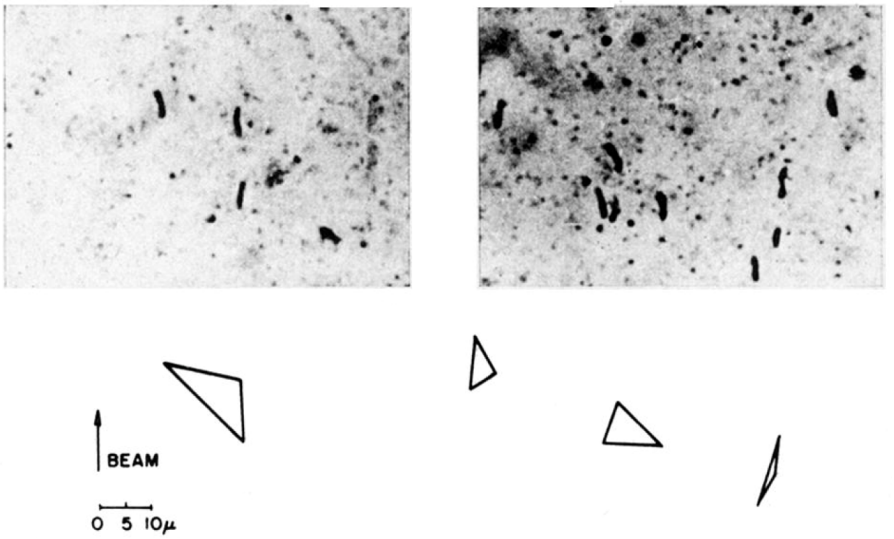

Figure 4: Reconstructions of exploded H_3^+ following foil-induced dissociation. On the top row are two photographs of exploded H_3^+ recorded on photographic emulsion at a tilt angle of 30°. The bottom row shows a few reconstructions (normally projected) made by inspection from the photographs. The authors analyzed 350 such photographs and concluded that H_3^+ mainly exhibits an equilateral triangle geometry. A distribution of angles for the equilateral geometries is given as well. From Gaillard et al. (1978). Reprinted with permission from APS.

fragments would inspire researchers to attempt to infer molecular structures using this data.

A better known CEI experiment was performed by Vager, Naaman, and Kanter (1989) almost a decade later employing a ∼30 Å carbon film. Their work was motivated by the opportunity of imaging non-classical molecular structures that more established methods were incapable of imaging. They were also the first to suggest that measuring the velocity (or momentum) vectors of each fragment would be provide all the information required to describe the molecule's structure. Figure 5 shows one reconstruction for the $C_2H_3^+$ molecule based on the measured velocity vectors. Assuming equation (2.1) holds, it shows an inference of the molecule's geometry based on the argument that is an experimental measurement of the square of the nuclear wave function, and is a convincingly pretty one at that.

However, they do not perform any geometry reconstruction and report their fragment ion densities in a coordinate system defined by the asymptotic velocity of each particle. Considering that they provide equation (2.1) for converting from velocity vector measurements to position vectors, it seems rather unusual for them to report a distribution in velocity space as opposed to position space.

The promise of recovering molecular geometries in this simple fashion seems quite empty now after glancing at the published literature in the few decades since. One reason for this include the difficulty

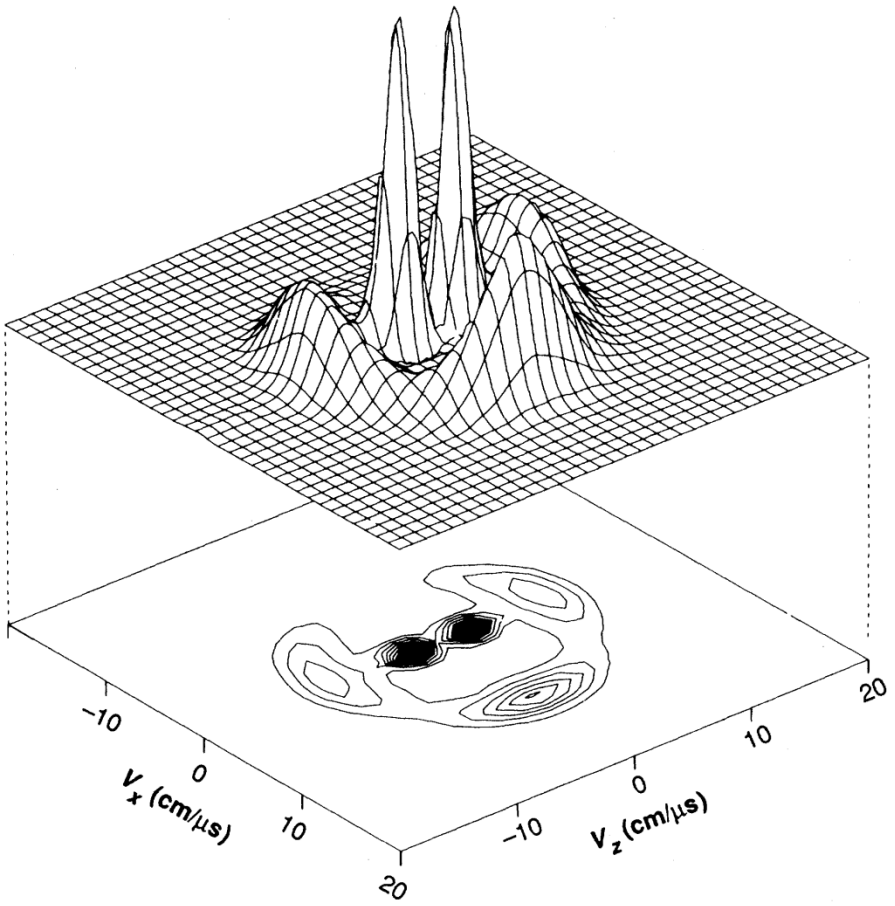

Figure 5: Reconstruction of $C_2H_3^+$ following foil-induced dissociation. The densities of the fragment ions are plotted in a coordinate system defined by the final velocities of each particle (relative to the mean carbon-ion velocity). The carbon ion densities were reduced by a factor of 5 for display purposes. From Vager, Naaman, and Kanter (1989). Reprinted with permission from AAAS.

and inconvenience of preparing the molecular beam required, which may not be possible for every molecule to be imaged. Another reason is that the method assumes that no molecular rearrangement occurs after the molecule passes through the thin film, which requires the complete dissociation of the molecule. This is a crucial issue no matter the ionization method used, however a thin film may not be very effective at inducing complete dissociation in many molecules of interest, while an ultrashort laser pulse can be more effective.

Foil-induced CEI has found some uses and produced some interesting work in recent years such as the imaging of the rovibrational wave functions of H_2^- and D_2^- by studying the kinetic energy released by the fragments (Herwig et al., 2013a; Jordon-Thaden et al., 2011), and imaging the absolute configuration of chiral molecules in the gas phase by studying the measured velocity vectors of the fragments with Newton plots (Herwig et al., 2013b). Vager (2001) also provides a review of some experiments. However, for the purposes of studying molecular structures and dynamics, the ultrafast laser shortly thereafter became the tool of choice for CEI. The reason for the scarcity of foil-induced CEI experiments in the published literature is mainly due to its limitations which also include the scarcity of experimental storage ring facilities, and the requirement the the molecule must be prepared as a molecular ion beam. The latter may prohibit the study of many molecules that cannot be prepared as such or which change molecular structure upon excitation away from the neutral ground state.

Of course, there are some assumptions that must hold for a complete and accurate recovery of the initial geometry. However, just by inspection of the schematic in figure 3 we can see that no rearrangement of the atoms must occur and that the molecular system must evolve on a purely Coulombic potential, which requires the rapid stripping of many electrons off the atom. Thus CEI becomes increasingly difficult to perform with larger molecules so it is best used to study smaller molecules. However, it is precisely these small molecules in the gas phase that need to be studied using CEI as they cannot be probed using other more established methods. Moreover, many smaller molecules may exhibit non-Coulombic behaviour unless placed into a highly charged state, which may be impossible depending on the apparatus in use.

2.3.2 *Imaging with ultrashort laser pulses*

Due to the limitations of foil-induced dissociation for CEI, the ultrashort laser emerged as a powerful tabletop solution for rapidly ionizing molecules. This occurred in the early 1990's following the development of the first broad-bandwidth solid-state Ti:sapphire laser by Moulton (1986) and the first demonstration of a mode-locked Ti:sapphire

laser producing femtosecond laser pulses by Spence, Kean, and Sibbett (1991), as well as the introduction of a chirped-pulse amplification scheme by Strickland and Mourou (1985) that allowed for the generation of ultrahigh peak power laser pulses (Maine et al., 1988).

Attempt at an analytical solution

Before taking a tour of the molecular geometries recovered using laser-induced CEI, it is worth mentioning that Nagaya and Bandrauk (2004) have attempted to arrive at an analytical solution for calculating molecular geometries from the measured momentum vectors. They were able to derive *classical imaging formulae* giving the image of the squared vibrational wavefunction inverted from the momentum distribution of the atomic ions for the Coulomb explosion of a diatomic molecule, a linear symmetric triatomic molecule, and a linear asymmetric triatomic molecule. They are able to derive similarly simple formulae for the diatomic as well as the linear, symmetric, triatomic molecule, but the more general case of the asymmetric, linear, triatomic molecule proves much more formidable.[6] The bulk of their article focuses on that case, deriving a three-dimensional classical imaging formula in terms of Jacobi and hyperspherical coordinates then reducing it to two dimensions. An extension to three dimensions for bent triatomic molecules is promised but could not be found in the published literature.

As an example, their formula for the Coulomb explosion of a diatomic molecule AB is given as

$$|\Psi_{\text{image}}(R_I)|^2 = S(p)\frac{1}{P_{\text{ion}}(R_I)}\sqrt{\frac{\mu q_A q_B}{8\pi\epsilon_0 R_I^3}} \tag{2.2}$$

where $S(p)$ is the momentum distribution measured in the asymptotic region (when the atomic fragments are far apart and barely interact), $P_{\text{ion}}(R_I) = |T_{\text{ion}}(R_I)|^2$ is the ionization probability, μ is the reduced mass of the diatomic molecule, q_A and q_B are the electric charges on atoms A and B respectively, and $R_I = \mu q_A q_B / 2\pi\epsilon_0 p^2$ is described elsewhere to be the R-coordinate corresponding to the total energy E of the exploding fragments determined by conservation of energy, $E = E_0 + q^2/R_I$ (Chelkowski and Bandrauk, 2002).

They proceed to compare their "classical" reconstructions for the vibrational wavefunction of various linear helium trimer systems (He_3^{3+} and He_3^{6+}) to the predictions of the quantum theory, noting small discrepancies. It seems that similar formulae were derived and actually used in previous studies (Bandrauk and Chelkowski, 2001; Chelkowski and Bandrauk, 2002), however, more recent studies do not seem to employ these classical imaging formulae.

6 While a highly commendable effort, their unsaid conclusion seems to be that this is an intractable problem as their research group seems to have gone silent on this

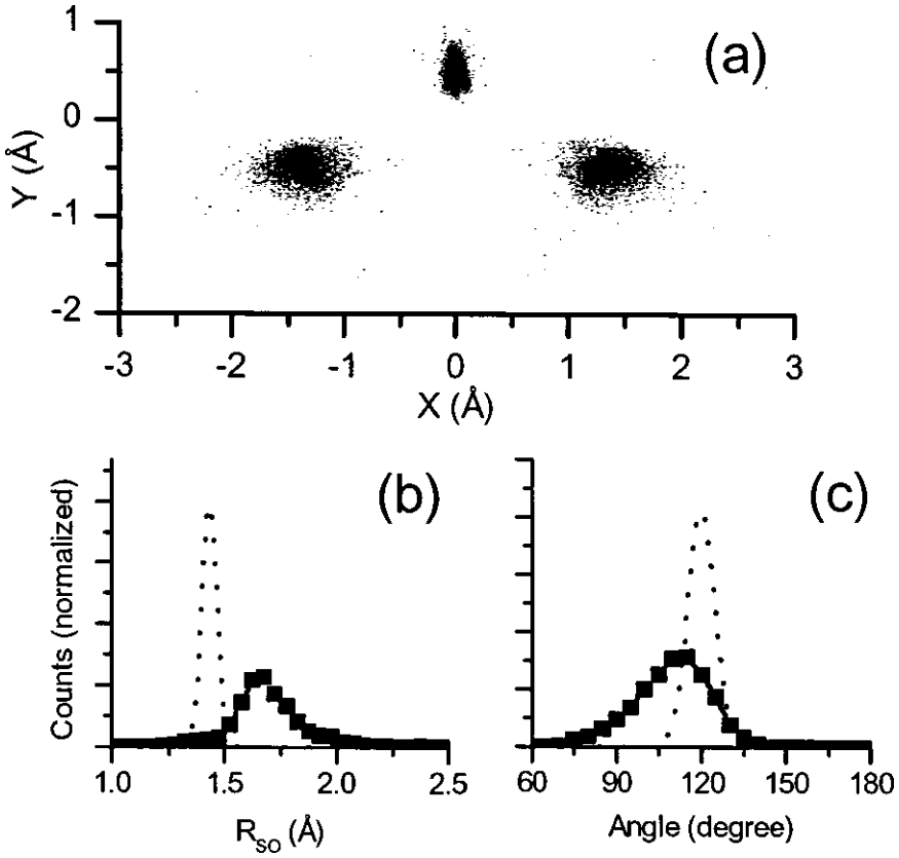

Figure 6: (a) Molecular structure of SO_2 using the SO_2^{7+} charge state ($SO_2^{7+} \rightarrow O^{2+} + S^{3+} + O^{2+}$). The center of mass is at $x = 0$, $y = 0$, and the y-axis is the bisector of the angle. (b) Radial distribution and (c) angular distribution of the reconstructed geometries with the dotted lines showing the expected distributions for the $v = 0$ stationary state structure of SO_2. From Légaré et al. (2005b). Reprinted with permission from APS.

Experimental reconstructions

Légaré et al. (2005a,b) were the first to use ultrashort laser pulses (8 fs) and CEI to report on molecular structures and dynamics. Figure 6 shows a reconstruction of SO_2 using the $SO_2^{7+} \rightarrow O^{2+} + S^{3+} + O^{2+}$ charge state.

While an intuitive way to plot geometries, we will show that such a plot can hide unphysical correlations in the reconstructed geometries. It would be interesting to see the radial distributions of both bond lengths to help ascertain the robustness of their reconstruction method. These marginal distributions are typically of the greatest interest but we will show that they can be also used to hide unphysical correlations (section 5.3.2), and that joint distributions, plotted using a scatter plot for example, should be reported.

problem and future studies do not refer back to Nagaya and Bandrauk (2004) except when discussing the difficulty of the problem.

To obtain the structures, they assume the explosion system evolves under a purely Coulombic potential and use optimization methods to make guesses at the structure that most accurately reproduces the observed data. Treating the geometry reconstruction as an optimization problem is exactly what we do in chapter 5. However, disappointingly they only allot a couple of sentence to describing their methodology and do not report the optimization methods employed, sidestepping the question of whether their methods were appropriate for the optimization problem as well as the nuances of the reconstruction process that we will discuss in chapters 4–6. Without knowledge of the optimization methods used, it is impossible to tell whether appropriate methods were used.

While Légaré et al. (2005b) may have not provided sufficient information regarding their methods for a third-party to reproduce their results, they do provide some important insights into the general problem of geometry reconstruction using CEI. They use simulations of dissociative ionization to estimate the role of intermediate charge state dynamics and show that discrepancies between reconstructed geometries and true geometries of the equilibrium state are mainly caused by ion motion during the ionization process. The ion motion is introduced mainly due to the interaction between the molecule and the finite pulse length of the laser (7 fs), which was not as much of a concern with foil-induced dissociation's interaction time of ~ 0.1 fs (Vager, Naaman, and Kanter, 1989). They subsequently argue that their half bond length resolution images are sufficient for the observation of large-scale rearrangements of small molecules such as isomerization processes.

Gagnon et al. (2008) reported the reconstruction of dichloromethane (CH_2Cl_2) using a home-made[7] stochastic-based simulated annealing algorithm that globally optimizes the molecular spatial configuration. Such an algorithm is an example of a heuristic derivative-free optimization algorithm, a class of algorithms that is sometimes described as a "last resort", such as by Conn, Scheinberg, and Vicente, 2009 who provide an introduction to derivative-free methods. Gagnon et al. (2008) attempt to minimize an objective function in the form of a scaled ℓ_∞ norm $\max_i |\mathbf{v}_i - \mathbf{v}_i^\star|/\mathbf{v}_i$ where the index i ranges over the atomic fragments, $\mathbf{v}_i$ is the theoretical or simulated velocity vector for the atomic fragment i, and $\mathbf{v}_i^\star$ is the experimentally measured velocity vector for the atomic fragment i. Gagnon (2006, p. 49) suggests in his thesis that choosing an objective function in the form of a χ^2 statistics may have been more desirable as it accounts for errors in the velocity vectors as well. Figure 7 shows an example of a reconstruction.

7 There is nothing wrong with writing your own code here but nonconvex optimization algorithms are tricky to get right and professional optimization libraries (both proprietary and open-source) do exist. Fortunately, the methodology and implementation's source code are publicly available in the main author's thesis (Gagnon, 2006).

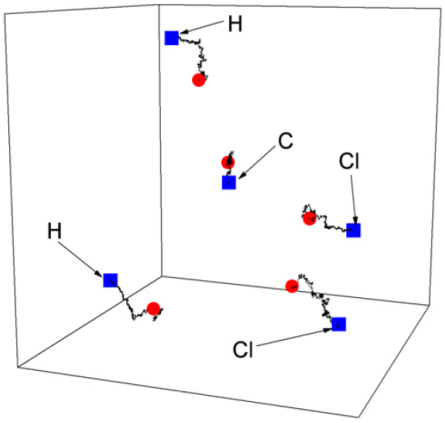

Figure 7: An example of a geometry reconstruction of dichloromethane (CH_2Cl_2) for the $CH_2Cl_2 \rightarrow H^+ + H^+ + C^+ + Cl^+ + Cl^{2+}$ fragmentation channel. The three-dimensional stochastic motion showing the trajectory of the guesses or iterates of the simulated annealing algorithm is shown (small black dots apparently but looks more like a solid black line) from the initial guesses for the positions (red circles) to the optimal solution found by the algorithm (blue squares). From Gagnon et al. (2008). Reprinted with permission from IOP.

They are only able to obtain the molecular structure for five sets of measured velocity vectors, out of potentially hundreds. Their reconstructed geometries possess the expected tetrahedral structure of CH_2Cl_2 and they compare the reconstructions to spectroscopic measurements, suggesting that the uncertainty in their geometry reconstruction is not due to the algorithm itself, but rather to the uncertainty in the velocity vectors, a relationship we explore quantitatively in chapter 6.

The most recent, and perhaps the most interesting geometry reconstruction effort using CEI so far, is the imaging of the long-predicted but experimentally elusive Efimov state of the helium trimer ($_4He_3$) by Kunitski et al. (2015), coming full circle to the very first images of the hydrogen trimer (Gaillard et al., 1978), both excellent examples of non-classical molecular structures which may be imaged effectively by CEI (Vager, Naaman, and Kanter, 1989). The ideal Efimov trimer is about 100 times larger than a typical triatomic molecule and does not exhibit a linear or equilateral structure. An impressive *Physics Today* article by Greene (2010) further explores Efimov states and universality in few-body physics. Figure 8 shows the theoretical and experimentally reconstructed molecular structures for the helium trimer. They plot their geometries in the center-of-mass coordinate frame which we employ for our geometry plots as well.

Kunitski et al. (2015, supplementary material) describe their reconstruction method which uses a lookup table (with a somewhat similar approach to the one we use in chapter 4). The lookup table we describe in chapter 4 maps geometries (r_{12}, r_{23}, θ) to asymptotic momentum vectors (described in a dimensionality-reducing convention). However, the lookup table they employed maps geometries (R_1, R_2, R_3) described using Dalitz coordinates to momentum space, also described using Dalitz coordinates. Dalitz coordinates were introduced by Dalitz (1953), and a more modern description of their use in

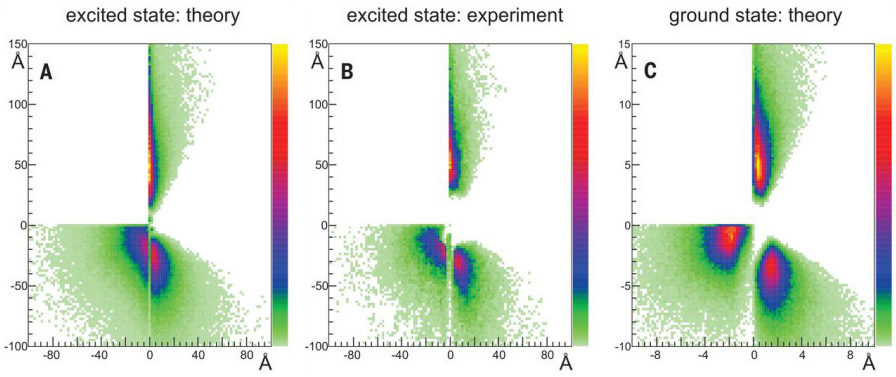

Figure 8: Theoretical and experimental molecular geometries of the helium trimer. The (A) theoretical and (B) experimental reconstructed excited-state molecular structures are shown along with the (C) theoretical ground state structure, with a much smaller length scale on both axes. The trimer's center of mass is shifted to the origin. The structures are also rotated such that the principal axis with the smallest moment of inertia points along the y-axis, and mirrored with respect to the x or y-axis such that one helium atom lies in the first quadrant and the other two in the third and fourth quadrants. From Kunitski et al. (2015). Reprinted with permission from AAAS.

CEI as Dalitz plots (a type of ternary plot) for describing momentum vector arrangements can be found in Ramadhan et al. (2016).

An important feature of Dalitz coordinates is that they require 2 coordinates to describe a triatomic molecule, effectively reducing the dimensionality of the geometry reconstruction problem to finding the two Dalitz coordinates as opposed to finding the three bond lengths. The third bond length is then calculated from the kinetic energy released (KER) by the three ions. Assuming the Coulomb explosion begins with the molecule at rest, the KER must equal the potential energy of the trimer (in atomic units)

$$\mathrm{KER} = \frac{1}{R_{12}} + \frac{1}{R_{13}} + \frac{1}{R_{23}} \tag{2.3}$$

and so the kinetic energy release, which is easily calculated from the measured momentum vectors using $\mathrm{KER} = \mathbf{p}^2/2m$, is related to the equilateral structure and a third bond length may be calculated if two others are known. If the ions pick up some initial momentum from the laser ionization process, however, (2.3) may not hold to a some degree.

They simulate the Coulomb explosion of 1000^2 structures with different Dalitz coordinates. The forward simulation of the Coulomb explosions is done using Newton's equations of motion while on the other hand we solve Hamilton's equations (section 3.3). They launch each trajectory six times with different randomly generated small initial momenta whose distribution is taken from measured KER spectra of the single charged helium ion to account for ion motion due to the laser ionization. They find that the initial values of the momenta do

not alter their reconstruction results significantly, however initial momentum kicks smaller than one hundredth of the KER ($\sim 0.25\,\text{eV}$) can produce significant shifts in the Dalitz plot for other molecules such as OCS^{4+} in the $(2,1,1)$ fragmentation channel (Ramadhan et al., 2016). This introduces additional error when attempting to reconstruct other molecules unless accounted for.

They also notice an "ambiguity of momentum-to-structure relation" for a small region in phase space (or structural space) which results in the reconstruction of some irrelevant geometries. This is due to the existence of multiple structures that produce the exact same set of momentum vectors, which we refer to as degenerate geometries in this thesis. We discuss them extensively in the following chapters. It is interesting to note that even a simple structure such the helium trimer's results in degeneracies.

They filter our their degenerate geometries, which seem to be "irrelevant" or physically unrealizable as they cannot correspond to the excited state of the helium trimer. While this may be true in this case, it is possible that each degeneracy for another molecular structure may represent a physically realistic geometry (section 5.4).

They believe an iterative approach to geometry reconstruction is impracticable due to the highly nonlinear relation between the initial spatial geometry and the asymptotic momentum vectors coupled with the initial momentum each ion picks up during the laser ionization process. While true, especially for the simple structure of the helium trimer, we believe an iterative optimization approach is superior in general. Computing the last bond length from the KER of the system using equation (2.3) would produce wildly inaccurate geometries as it assumes the process is 100% Coulombic, that is, the potential energy of the molecular system is completely converted into kinetic energy. However, this is not true for many molecules such as OCS (Wales et al., 2014), and even for highly charged states of some molecules such as CS_2^{10+} for both the $(4,2,4)$ and $(3,4,3)$ fragmentation channels (Matsuda, Takahashi, and Hishikawa, 2014). If we cannot compute the last bond length from the KER of the system then we would have to start with a three-dimensional lookup table, and lookup table storage space requirements scale up in an exponential manner (section 4.3.1).

They suggest that extending their approach to four-body system seems feasible, in which case four-particle Dalitz plots may be used (Schulz et al., 2007), and may be an idea worth pursuing for cases when (2.3) holds.

3

DATA MEASUREMENT AND ANALYSIS

The premise of CEI is simple enough for a quirky elevator pitch yet the collection and analysis of data is a nuanced multi-step process as we are attempting to measure the momentum vector of each single atomic fragment precisely. This chapter will discuss the data measurement and analysis process in detail from the measurement of position and time to the calculation of the momentum vector components and theoretical lower-bound estimates of the measurement uncertainty. We will explore the collected data set as well as provide some information regarding the simulation of a Coulomb explosion and the conventions used throughout this thesis for describing the geometries and momentum vectors.

3.1 DATA MEASUREMENT

In this section we will go through the process of how the momentum vectors of each atomic fragment are measured. This will require some discussion regarding the apparatus, algorithms, and intricacies of the process, all of which are essential to understand exactly how the data is collected so that it can be analyzed appropriately. We will also quantify the uncertainty in those measurements, which will be useful in quantifying our uncertainty in the reconstructed geometries in chapter 6.

3.1.1 *Time and position measurement*

The time-of-flight of each atomic fragment and its position on the detector are required to calculate its momentum vector. The mea-

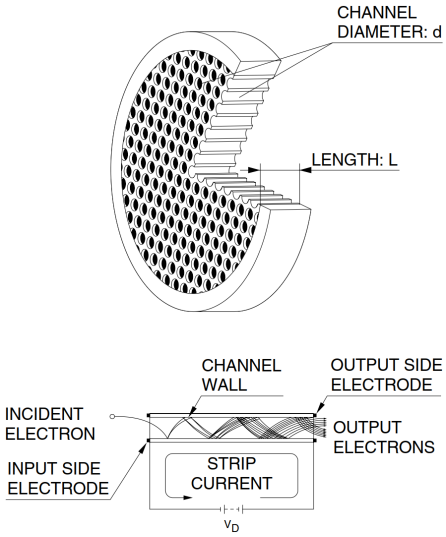

Figure 9: Schematic of a multi-channel plate (MCP). The incident particle need not be an electron, and may in fact be any charged particle, a high-energy photon, or a neutrally charged particle with kinetic energy larger than the work function of the glass.

surement of time and position is carried out by a two-stage apparatus feeding electrical signals into a data acquisition (DAQ) computer which analyzes the signals to determine time and position.

The first part of the two-stage apparatus is a set of two micro-channel plates (MCP) placed in a chevron configuration.[1] Figure 9 shows a schematic of an MCP. Once a charged particle is incident on an MCP and collides with a channel wall, multiple secondary electrons are emitted and accelerated up the channel due to the applied voltage V_D setting up a potential gradient along the channel and replenishing the emitted electrons. Due to the angled channels, the emitted electrons follow parabolic trajectories hitting the other wall and continuing the amplification process until a large number of particles ($\sim 10^4$, depending on the applied voltage) are emitted at the channel output.

The job of the MCP is to amplify the signal of a single charged particle enough such that it may be detected as an electrical signal by an oscilloscope, much like a photomultiplier tube. Thus the output of an MCP is a shower of charged particles, or rather a charged cloud. The charged cloud may be fed into a second MCP to further amplify the signal. A two-stage chevron MCP setup produces an amplification of approximately 10^5 to 10^7 depending on the applied voltage V_D across the channels.

You may notice that while most of the MCP's surface is covered in channels, not all of it is, leading to an imperfect detection rate. Only 60% of the area is open to incident particles, and if a particle is incident on the other 40% then it is not detected. Thus the detection efficiency of a triple coincidence event is $(0.6)^3 \approx 0.2$ and so we see

1 The channels of an MCP are slanted, usually at a (bias) angle of 5° to 15° to increase the probability that an incident particle collides with the channel wall. To further increase this probability, the second MCP is oriented such that its channels are slanted in the opposite direction forming a V-like (or chevron-like) channel configuration.

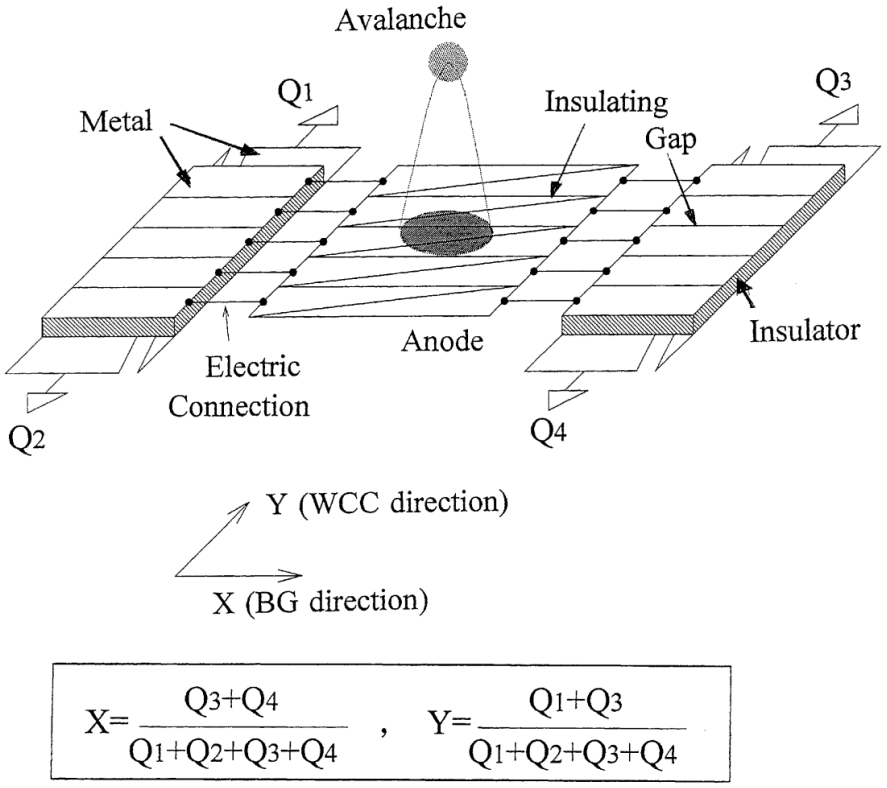

Figure 10: Schematic of a symmetrized "modified backgammon with weighted capacitors" (MBWC) anode for position detection. Figure from Mizogawa et al. (2002). Reprinted with permission from Elsevier.

that detection efficiency decreases rapidly with the number of fragments that must be detected, suggesting that larger molecules are more difficult to study. There do exist "funnel" MCPs with an open area ratio of 90% that increase the detection efficiency.

By itself this MCP setup is enough to provide time-of-flight information but to obtain position information, this charged cloud output is made incident on a position sensitive anode such as a "modified backgammon with weighted capacitors" anode or readout pad built as described by Veshapidze et al. (2002) and used by Ramadhan et al. (2016). Figure 10 shows a schematic of such an anode.

The avalanche of charged particles, or the charged cloud, hits the anode and induces a charge on it. This charged is induced via the capacitive couplings from the feedback capacitors of the preamplifiers connected to the triangular conducting strips. The lines on the anode are insulating gaps, splitting the anode into a series of triangles whose arrangement resemble that of the backgammon board game. The metal strips are capacitively coupled to the triangular strips through the insulator. If the cloud lands on the right side of the anode, then a larger fraction of the induced charge will flow to Q_3 and Q_4. So we can see that $x = 0$ corresponds to the left side of the anode, and $x = 1$ to the right side. If the cloud lands further

up the anode, then a larger fraction of the induced charge will flow through Q_1 and Q_3 so we see that $y = 0$ corresponds to the bottom side of the anode and $y = 1$ corresponds to the top side. The design gets its name as it is a combination of two older designs, the "backgammon" (BG) and the "weighted coupling capacitor" (WCC) designs. Mizogawa et al. (1992) provides a more detailed explanation of its operation. This setup provides position information on a scale $0 \leq x, y \leq 1$ and must be multiplied by appropriate scaling factors for each dimension, which depend on the physical dimension of the setup, to retrieve the correct position information in meters.

These four Q_i signals are subsequently fed into an Ortec 142 preamplifier in energy output mode, essentially acting as an operational amplifier integrator. The integrated signal is then fed into a data acquisition (DAQ) computer equipped with a four-channel oscilloscope. Every time a laser pulse is fired into the experimental apparatus, an electrical signal is sent to the oscilloscope as a trigger. The software running on the DAQ computer examines the four signals following a trigger and saves them if it detects evidence of charged particle detection, using a coincidence detection algorithm for improving detection rates as described by Wales et al. (2012a).

A particularly good example of a triple coincidence detection event can be seen in figure 11, wherein the fragmentation event showcased is the concerted breakup process $CS_2 \rightarrow CS_2^{3+} \rightarrow C^+ + S^+ + S^+$.

As the carbon atom is lighter, the first step at 500 ns is the detection of the carbon atom. All four channels increase by roughly similar amounts hinting that the carbon atom was detected near the center of the detector by (3.1). Then the second and third events belong to the two sulfur atoms, arriving later due to sulfur's larger atomic mass. If the molecule was aligned in a plane parallel to the detector, and both sulfur atoms started with zero initial momentum, then they would have remained at the same height throughout the Coulomb explosion and been detected at the same time, producing one step in the signal. However, one must have had some downward initial momentum with the other having some upward initial momentum, which may be a result of the molecule having been oriented vertically such that one sulfur atom was closer to the detector. During the Coulomb explosion, the closer atom will initially experience a kick towards the detector while the other sulfur atom will initially experience a kick away from the detector before being accelerated upwards due to the constant electric field. This results in one sulfur atom arriving earlier, and the other later. Looking at the individual signals, we see a significant increase in Q_1 and Q_2 at 800 ns suggesting that the first sulfur atom was on one side of the detector while the more significant increase in Q_3 and Q_4 at 900 ns suggest that the second sulfur atom was on the other side. This makes some intuitive sense as we expect the carbon atom to land somewhere near the middle and the two sulfurs

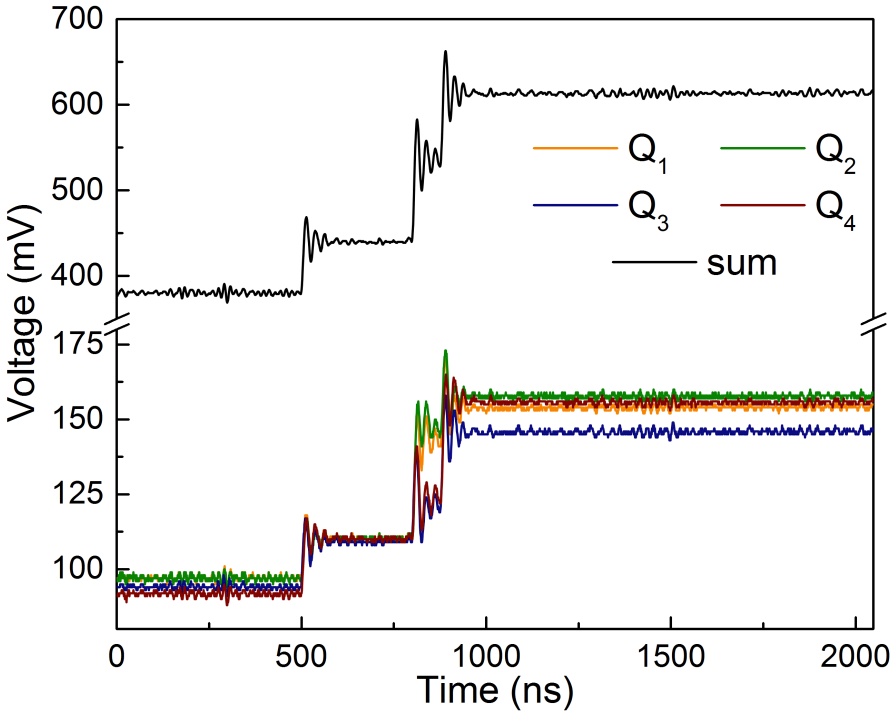

Figure 11: Spectrometer response during a triple coincidence event. This was taken during a CEI experiment studying the dynamics of CS_2 at the Canadian Light Source. The fragmentation event showcased is the concerted breakup process $CS_2 \rightarrow CS_2^{3+} \rightarrow C^+ + S^+ + S^+$.

to land on opposite sides of the detector. The oscilloscope cards sport an 8-bit bus and so the individual Q_i channels were limited to 200 mV to increase position detection accuracy. Triple coincidence events tend to make up approximately $(0.6)^3 \approx 0.2$ of detected events due to the detector's imperfect detection rate, however rich and interesting signals such as this one can be rarer. The majority of events detected tend to be single or double coincidences of course.

As a side note, if too many atomic fragments arrive at the detector in a short enough time period, steps due to different molecules may get mixed. The laser typically produces pulses with a repetition rate of 1 kHz which will keep different events from overlapping. When the experiment is carried out at a synchrotron facility, adjusting the light intensity until a lower event rate is observed (100 Hz) works better due to the lower level of control we have over the event rate (Ramadhan et al., 2016). Another reason to keep the count rate low is that the signals need time to decay back down post-integration otherwise the signals will saturate the oscilloscope at 200 mV. The ringing artifacts on the signal are due to impedance mismatches between the anode and preamplifiers.

Software can analyze the signals and determine the magnitude of each Q_i signal. If the change in baseline before and after an event is denoted Q_i' then the position of the electron is then calculated using

$$x = \frac{Q_1' + Q_2'}{Q_1' + Q_2' + Q_3' + Q_4'}, \quad y = \frac{Q_1' + Q_3'}{Q_1' + Q_2' + Q_3' + Q_4'} \tag{3.1}$$

where $x, y \in [0, 1]$ are fractional positions. Multiplying x and y by the dimensions of the MCP detector will yield the physical position of the cloud's centroid.

3.1.2 *Calculating the atomic fragments' momenta*

Calculating the momentum vector of each atomic fragment is an elementary physics problem once we have the time and position measurements. The components of the three-dimensional momentum vector $\mathbf{p} = (p_x, p_y, p_z)$ for each atom are calculated as

$$p_x = \frac{m(x - x_0)}{t}, \; p_y = \frac{m(y - y_0)}{t}, \; p_z = \frac{qV}{2\ell}\left(\frac{t_0^2 - t^2}{t}\right) \tag{3.2}$$

where m is the atom's mass, (x, y) is the location the atom collided with the MCP detector, and (x_0, y_0) is the location that the Coulomb explosion originated. The location $(0.5, 0.5)$ corresponds to the physical center of the MCP detector. q is the net charge of the atom, V is the value of the constant electric field the atom is subjected to, and ℓ is the distance from the location of the Coulomb explosion to the detector. t is measured time of flight (between Coulomb explosion and detection) of the atom and

$$t_0 = \sqrt{\frac{2d\ell}{V}\left(\frac{m}{q}\right)} \tag{3.3}$$

is the atom's time of flight assuming no external forces act on it during its trip to the detector.

3.1.3 *Measurement uncertainty in the momenta*

For any relation $f = f(x_1, x_2, \ldots, x_n)$, assuming independent variables (neglecting correlations), the standard deviation (or absolute uncertainty) in a quantity f, which we denote Δf, may be calculated using the variance formula (Ku, 1966), which has been very popular among physical scientists,

$$\Delta f = \sqrt{\sum_{i=1}^{n}\left(\frac{\partial f}{\partial x_i}\Delta x_i\right)^2} \tag{3.4}$$

where Δx_i is the standard deviation in the independent variable x_i and where the partial derivatives $\partial f / \partial x_i$ are evaluated at the mean of x_i. This formula relies on the linear characteristic of the gradient of f and so it's a good estimate for the standard deviation of f as long as the standard deviations Δx_i are small compared to the partial derivatives. However, as it employs a truncated Taylor series, it may even be a biased estimate in some cases. Much can be said about which formula to use in the propagation of uncertainty, and expository articles on the subject have been written by Birge (1939) and Ku (1966), the latter of which provides an insightful derivation of (3.4). In our case, we will not attempt to make accurate nor precise calculations of any uncertainty, so we will not fuss about which method we choose to propagate our uncertainties forward. We are simply interested in making order-of-magnitude estimates of the uncertainties on the measured momentum vectors in order to make order-of-magnitude estimates on the uncertainty of reconstructed geometries in chapter 6.

Using (3.4) we may calculate the uncertainty in the measured momentum values, which will be different for each component. In our case, p_x is a function $p_x(m, x, x_0, t)$ and $p_y = p_y(m, y, y_0, t)$, however, the uncertainty in the atomic mass m is orders of magnitude smaller than the uncertainty in the other variables and so we will ignore its effects. Thus we get that

$$\Delta p_x = \sqrt{\left(\frac{\partial p_x}{\partial x}\Delta x\right)^2 + \left(\frac{\partial p_x}{\partial x_0}\Delta x_0\right)^2 + \left(\frac{\partial p_x}{\partial t}\Delta t\right)^2} \tag{3.5a}$$

$$\Delta p_y = \sqrt{\left(\frac{\partial p_y}{\partial y}\Delta y\right)^2 + \left(\frac{\partial p_y}{\partial y_0}\Delta y_0\right)^2 + \left(\frac{\partial p_y}{\partial t}\Delta t\right)^2} \tag{3.5b}$$

where the partial derivatives can be calculated from (3.2) as

$$\frac{\partial p_x}{\partial x} = \frac{m}{t}, \quad \frac{\partial p_x}{\partial x_0} = \frac{m}{t}, \quad \frac{\partial p_x}{\partial t} = -m\frac{x - x_0}{t^2} \tag{3.6a}$$

$$\frac{\partial p_y}{\partial y} = \frac{m}{t}, \quad \frac{\partial p_y}{\partial y_0} = \frac{m}{t}, \quad \frac{\partial p_y}{\partial t} = -m\frac{y - y_0}{t^2} \tag{3.6b}$$

and so after plugging in the partial derivatives and rearranging slightly we get that

$$\frac{\Delta p_x}{p_x} = \sqrt{\left(\frac{\Delta x}{x - x_0}\right)^2 + \left(\frac{\Delta x_0}{x - x_0}\right)^2 + \left(\frac{\Delta t}{t}\right)^2} \tag{3.7a}$$

$$\frac{\Delta p_y}{p_y} = p_y\sqrt{\left(\frac{\Delta y}{y - y_0}\right)^2 + \left(\frac{\Delta y_0}{y - y_0}\right)^2 + \left(\frac{\Delta t}{t}\right)^2} \tag{3.7b}$$

Repeating the process for $p_z = p_z(q, V, \ell, t_0, t)$ but ignoring the tiny uncertainties in q, V, and ℓ, we get

$$\frac{\Delta p_z}{p_z} = p_z \sqrt{\left(\frac{2tt_0}{t_0^2 - t^2}\Delta t_0\right)^2 + \left(\frac{t^2 + t_0^2}{t(t_0^2 - t^2)}\Delta t^2\right)^2} \tag{3.8}$$

To calculate the uncertainties in position, Δx and Δy, we repeat the same process for equations (3.1) to arrive at

$$\Delta x = \frac{1}{Q_T}\sqrt{(1-x)^2(\Delta Q_1^2 + \Delta Q_2^2) + x^2(\Delta Q_3^2 + \Delta Q_4^2)} \tag{3.9a}$$

$$\Delta y = \frac{1}{Q_T}\sqrt{(1-y)^2(\Delta Q_1^2 + \Delta Q_3^2) + y^2(\Delta Q_2^2 + \Delta Q_4^2)} \tag{3.9b}$$

where $Q_T = Q_1 + Q_2 + Q_3 + Q_4$. The values of Q_i are digitized with the same resolution thus denoting $\Delta Q_i = \Delta Q$ for $i = 1, 2, 3, 4$ yields simpler relations

$$\Delta x = \sqrt{2}\frac{\Delta Q}{Q_T}\sqrt{(1-x)^2 + x^2} \tag{3.10a}$$

$$\Delta y = \sqrt{2}\frac{\Delta Q}{Q_T}\sqrt{(1-y)^2 + y^2} \tag{3.10b}$$

3.2 EXPLORATORY DATA ANALYSIS

Before we attempt to reconstruct geometries based on the momentum vectors measured, it would be a good idea to inspect and analyze the raw data first. Such an analysis will help us identify any issues with the data and really we should be making sense of the raw data before attempting to analyze it further. Such an analysis is typically termed exploratory data analysis, first promoted by Tukey (1977).

3.2.1 *Momentum measurements*

For each Coulomb explosion event, our measurements for OCS include a momentum vector (p_x, p_y, p_z) for each atom or fragment. Thus each event involves the measurement of 9 scalars, ultimately forming a 9-dimensional multivariate dataset with some structure, mainly imposed by the condition that momentum must be conserved along each physical axis in the reference frame of the molecule, that is the center-of-momentum (COM) frame.

For this section we will look at the momentum vectors measured for OCS in the $(2, 2, 2)$ fragmentation channel following Coulomb explosion by a 7 fs laser pulse, with all measurements kept in the lab frame. This allows us to inspect the raw data before it is transformed to the COM frame with the added benefit that it allows for the inspection of any instrumental or systematic errors. Qualitatively, the data

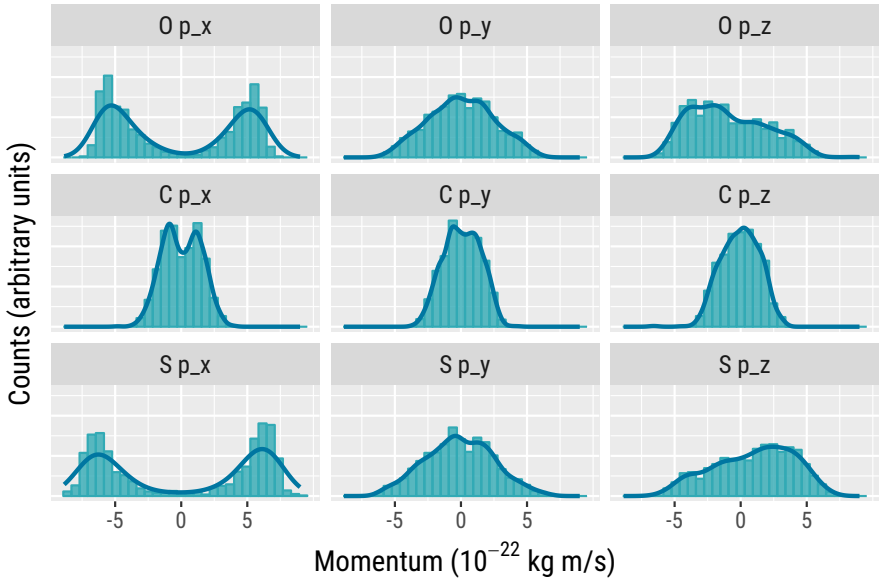

Figure 12: Distributions for each atom's momentum components measured after Coulomb explosion by a 7 fs laser pulse for the $(2, 2, 2)$ fragmentation channel (in the lab frame). Kernel density estimates (with a Gaussian kernel, see section 3.2.3) are overlaid to estimate probability distributions.

looks quite similar in the two frames, but the geometry reconstruction can only be done in the COM frame as the simulation assumes that the molecule starts from rest such that momentum is always conserved as the molecular systems undergoes a Coulomb explosion.

In figure 12 we plot histograms for each atom's momentum components and overlay the histograms with kernel density estimates (using a Gaussian kernel) to estimate the probability density function of each momentum component. Glancing at the distributions, they seem to make sense qualitatively with some peculiarities.

We can infer some basic dynamics from these plots. For example, we see that the oxygen and sulfur atoms tend to fly off with high velocity in the x-direction while the carbon atom flies off with little velocity in the x-direction. As conservation of momentum must hold, the p_x component of the molecular system must sum to zero suggesting that the oxygen and sulfur tend to fly off in opposite x-directions. This is expected as they are both terminal atoms and confirms that the data makes physical sense. The other distributions do not say much except for the slight asymmetry in the oxygen and sulfur's p_z distributions which may be due to instrumental bias in measuring the arrival time of atomic fragments. We also notice that the momentum distribution is rather isotropic in the y and z directions but is bimodal in the x direction, suggesting that the laser pulse's electric field was polarized along the x-axis. Kernel density estimates (with a Gaussian kernel) are used to estimate the probability density function of each momentum component. Such estimates are inherently not as effective

at estimating bimodal distributions thus the carbon and sulfur's p_x estimates appear over smoothed.

This analysis should be done for each data set collected. In our case, this includes the 30 fs, 60 fs, 100 fs, and 200 fs data sets which we include in appendix A as figures 31–34.

While figure 12 gives us some insight into the measurements, we are looking at a multidimensional data set and might want to look at correlations between each measurement. We do this using a pairs plot, or a scatter plot matrix, which is a grid of scatterplots showing the bivariate relationships between all pairs of variables in a multivariate dataset. The pairs plot was first introduced by Hartigan (1975), however we use the more modern and generalized version introduced by Emerson et al. (2013) as they provide an open-source implementation in the form of an R package.

Figure 13 uses a pairs plot to showcase the relationship between every pair of variables in our dataset. Due to the 9-dimensional nature of our dataset, we end up with a 9×9 grid of plots with some redundancy. The diagonal repeats the kernel density estimates shown in figure 12 but they are quite useful as a reference here. Below the diagonal are the scatter plots, however due to the high density of points, contour plots are employed to showcase the same relationship above the diagonal. So only 36 scatter plots are required but this format provides us with greater insight of our dataset, especially in scatter plot regions with a large density of points.

For example, the scatter plot in row 7, column 1 plots the oxygens's p_x component on the x-axis and the sulfur's p_x component on the y-axis. The contour plot in row 1, column 7 shows the same relationship. We see a negative correlation between oxygen's p_x and sulfur's p_x as predicted in figure 12's caption. Similarly we see negative correlations between oxygen's p_y and sulfur's p_y as well as between oxygen's p_z and sulfur's p_z, which makes physical sense due to the two atoms being terminal atoms, so they should fly off in approximately opposite directions following a Coulomb explosion.

Pairs plots for the 30, 60, 100, and 200 fs data sets are included in appendix A as figures 35–38.

3.2.2 *Discrepancy in the momentum measurements*

It is also worth looking at the measurements of the momentum vectors for the three atoms to check if they sum to zero, that is if momentum is conserved. We will look at the momentum measurements for a laser pulse length of 7 fs. We plot a histogram of the momentum discrepancy $\|\mathbf{p}_O + \mathbf{p}_C + \mathbf{p}_S\|$ exhibited by the measurements in figure 14(a). In (b) we also plot a histogram for the total momentum $\|\mathbf{p}_O\| + \|\mathbf{p}_C\| + \|\mathbf{p}_S\|$ for comparison.

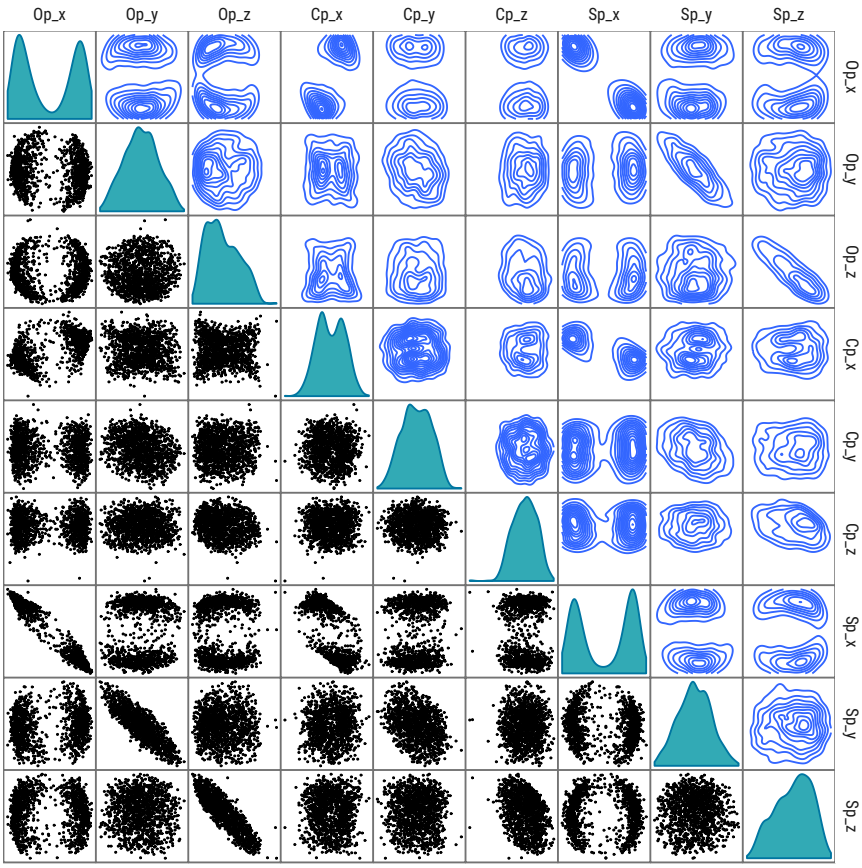

Figure 13: Pairs plot showing the bivariate relationship between each atom's momentum components measured after Coulomb explosion by a 7 fs laser pulse for the $(2, 2, 2)$ fragmentation channel. On the diagonal, kernel density estimates (with a Gaussian kernel, see section 3.2.3) of the momentum component designated by the label at the top of the column and end of the row are given. Below the diagonal, scatter plots show the relationship between the momentum components belonging to that row and column. Above the diagonal, the same relationship is given using a contour plot instead.

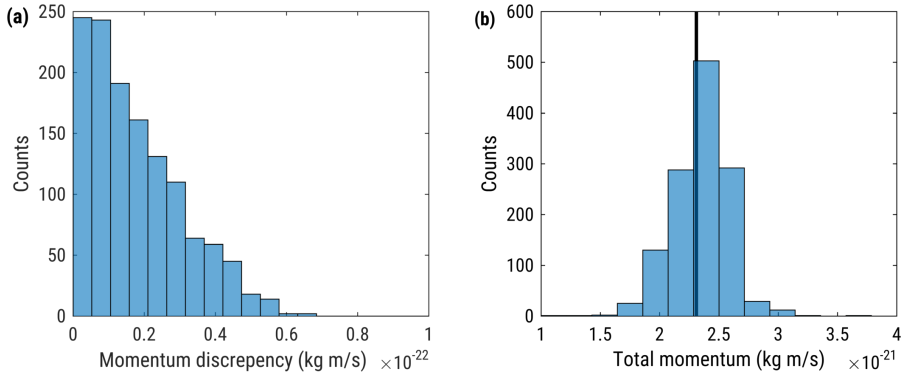

Figure 14: (a) Momentum discrepancy and (b) total momentum for momentum vectors measured for the OCS $(2, 2, 2)$ molecule after Coulomb Coulomb by a 7 fs laser pulse. The vertical black line indicates the expected momentum when exploding from the ground-state geometry.

We see that our the measured momentum vectors sum to zero with a discrepancy of less than $5 \times 10^{-23}\,\mathrm{kg\,m\,s^{-1}}$ in the vast majority of cases. This is compared to the total momentum which peaks around $2.5 \times 10^{-21}\,\mathrm{kg\,m\,s^{-1}}$. Thus the discrepancy corresponds to a $< 2\%$ error due to momentum not being conserved.

The vertical black line in figure14(b) indicates the expected momentum when exploding from the ground-state geometry ($r_{CO} = 1.15\,\text{Å}, r_{CS} = 1.56\,\text{Å}, \theta = 172°$), which we sometimes call a 100% Coulombic explosion. In fact the total momentum is distributed about the expected momentum, suggesting that the OCS $(2,2,2)$ fragmentation channel is "highly Coulombic", or that the Coulomb potential is a good approximation. The agreement is not perfect, however, and this will introduce some error in the reconstructed geometries as we assume a purely Coulombic potential. Significant amounts of initial momentum carried by any of the atomic fragments will also introduce error in the reconstructed geometries.

3.2.3 *Kernel density estimation*

Kernel density estimates (KDE's) are used to estimate probability distributions in figures 12 and 13, and will be used to estimate probability distributions for the atomic positions of reconstructed geometries in chapters 4 and 5. They are a method of performing nonparametric statistics, that is, of fitting observations to a probability distribution that has no dependency on a parameter (Stuart, Ord, and Arnold, 2010, §20.2-20.3). They serve roughly the same purpose as a histogram, however histograms tend to be non-smooth and their shape depends on both the width of the bins and the ends points of the bins. To solve at least the first two problems, we can use a KDE. They are especially effective for estimating high-dimensional probability distributions where histograms can be very sparse (see also curse of dimensionality, section 5.1.6).

For independent and identically distributed univariate samples $x_1, x_2, \ldots, x_n$ drawn from an unknown probability distribution f, its KDE is

$$\hat{f}_h(x) = \frac{1}{n}\sum_{i=1}^{n} K_h(x - x_i) = \frac{1}{nh}\sum_{i=1}^{n} K\left(\frac{x - x_i}{h}\right) \tag{3.11}$$

where K is the *kernel*, a function with zero mean that integrates to one, and $h > 0$ is the *bandwidth*, a smoothing parameter analogous to the bin width (Scott, 2015, p. 137). Many kernel choices exist such as the rectangle or triangle function but the standard normal (or Gaussian) kernel is the most popular. Multivariate KDE's replace the scalar bandwidth parameter h with a symmetric and positive bandwidth matrix $\mathbf{H}$ with a variable number of smoothing parameters

(Wand and Jones, 1993) however full bandwidth matrices seem to give markedly better performance (Duong and Hazelton, 2003).

Selection of the bandwidth h, as expected, is the most difficult aspect of KDE. Analytical formulae can be derived for the bandwidth h (Scott, 2015, p. 143) that accurately estimate the unknown density f by minimizing the mean integrated squared error (MISE) however they require knowledge of the unknown density f and so cannot generally be used. Fortunately, when estimating a normal (or Gaussian) density, the bandwidth that minimizes the MISE is given by Silverman (1986) as

$$h = \left(\frac{4\hat{\sigma}^5}{3n}\right)^{1/(d+4)} \approx 1.06\left(\frac{\sigma^5}{n}\right)^{1/(d+4)} \tag{3.12}$$

where n is the number of observations, $\hat{\sigma}$ is the standard deviation of the samples, and d is the dimensionality of the observations (so $d = 1$ for univariate observations and $d = 2$ for bivariate). This is termed the *normal distribution approximation* or the "rule of thumb". As most of our data, especially the atomic positions of reconstructed geometries, seems to roughly follow a normal distribution, we used the Silverman (1986) rule of thumb in computing our kernel bandwidths. It is worth noting that it does not perform well on multimodal data which is why certain estimates in figure 12 appear over smoothed.

Other methods do exist, and univariate estimators are especially effective as briefly surveyed by Jones, Marron, and Sheather (1996), but bandwidth selection for multivariate kernel density estimates is quite difficult for non-Gaussian densities (Scott, 2015, §6.5.2).

3.3 COMPUTATIONALLY SIMULATING A COULOMB EXPLOSION

To simulate a Coulomb explosion of a molecule containing n atoms, we will make some simplifying assumptions to arrive at the simplest possible simulation that will allow us to investigate the problem of reconstructing geometries. We will assume that the motion of the ions are governed only by their mutual Coulomb repulsion, so that the chemical bonds are broken instantaneously at $t = 0$ and have no effect on the trajectories of the ions, and that neutral fragments do not interact with any other fragment. We model each atom as a point particle with a fixed electric charge assigned at $t = 0$ so no charge redistribution can occur. We assume that the atoms each begin at rest at $t = 0$ and that their initial positions are determined by the molecule's equilibrium or ground-state geometry. Of course, these assumptions force us to ignore the rearrangement of the atoms under the influence of the laser pulse's intense electromagnetic field, and any initial momentum imparted on the atoms by this interaction.

Under such assumptions, we can solve the classical equations of motion for each ion right after the explosion. We choose to use Hamiltonian mechanics to acquire a system of $6n$ first-order ordinary dif-

ferential equations (ODE's) which may be easily solved by numerical methods such as the ubiquitous fourth-order Runge-Kutta. If Lagrangian mechanics is used, then the resulting second-order must be recast as a system of first-order ODE's as numerical algorithms are developed to solve systems of first-order ODE's. The Hamiltonian of the molecular system is

$$\mathcal{H}(\mathbf{r}_i, \mathbf{p}_i, t) = \sum_{i=1}^{n} \frac{\mathbf{p}_i^2}{2m_i} + \frac{1}{4\pi\epsilon_0} \sum_{\substack{\{i,j\} \\ i \neq j}} \frac{q_i q_j}{|\mathbf{r}_i - \mathbf{r}_j|} \tag{3.13}$$

where $i, j \in \{1, 2, \ldots, n\}$ and so the second summation is over all i, j pairs where $i \neq j$. Calculating Hamilton's equations for the system, we get

$$\frac{d\mathbf{r}_i}{dt} = \frac{\partial \mathcal{H}}{\partial \mathbf{p}_i} = \frac{\mathbf{p}_i}{m_i} \tag{3.14a}$$

$$\frac{d\mathbf{p}_i}{dt} = -\frac{\partial \mathcal{H}}{\partial \mathbf{r}_i} = -\frac{1}{4\pi\epsilon_0} \sum_{j,\, j \neq i} \frac{\mathbf{r}_i - \mathbf{r}_j}{|\mathbf{r}_i - \mathbf{r}_j|^3} \tag{3.14b}$$

where i is held fixed over the second summation. With appropriate initial conditions this system of $6n$ scalar first-order ordinary differential equations may be easily solved using, for example, the classical fourth-order Runge-Kutta method for numerically solving ordinary differential equations. The atoms are assumed to be at rest so that $\mathbf{p}_i(t = 0) = 0$, while the initial positions, $\mathbf{r}_i(t = 0)$, are chosen to correspond to the molecular geometry.

One way to think of the problem being tackled in this thesis is: which initial geometry $\mathbf{r}_i(t = 0) = 0$ results in the momentum values measured at the detector? The atoms are far enough apart after just a few nanoseconds that by the time they arrive at the detector, they feel almost no forces due to each other and their momenta attain asymptotic values which we can denote $\mathbf{p}_i(t \to \infty)$.

For some perspective, Slater et al. (2015) discuss the computational simulation of ion trajectories for larger systems such as the the 3,5-dibromo-3',5'-difluoro-4'-cyanobiphenyl (DBrDFCNBph) molecule, and who interestingly employ a pixel-imaging mass-spectrometry camera for position detection.

3.4 DESCRIBING GEOMETRIES AND MOMENTA

While tackling the problem of geometry reconstruction, it will be crucial to choose a convention for describing the geometries and momentum vectors especially so that geometries and vector arrangements can be compared with ease. Even more importantly, it provides us with an opportunity to reduce the dimensionality of the problem from $3N$ to $3N - 6$ for a molecular system with N atoms. This stems

from the fact that we only need to describe the relative position of each atom, not its absolute position.

For example, a triatomic molecule can be described by two bond lengths r_{12}, r_{23} and a bond angle θ, or even three bond lengths (r_{12}, r_{13}, r_{23}) rather than three position vectors $(\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3)$.

For larger molecules, dihedral angles are required to describe the geometry in addition to bond lengths and angles. Such coordinates are called *internal coordinates* and multiple possible descriptions may exist (Peng et al., 1996) although it is unclear how significantly the choice of coordinate system will come into effect for molecules containing several atoms. The Z-matrix, a tool from computational chemistry, may be used to store and convert between them and Cartesian coordinates using the Natural Extension Reference Frame (NERF) algorithm (Parsons et al., 2005). However, concerns regarding the Z-matrix have been discussed (Baker and Chan, 1996; Baker and Hehre, 1991) and further investigation is required

Before attempting to compare simulated momentum vectors with experimentally measured data, the momentum vector measurements must be converted from the laboratory frame to the center-of-momentum (COM) frame by removing any center-of-mass motion of the molecular system (see the `removeCOMMotion.m` code listing in appendix B).

The exact same molecule can produce different momentum vectors after a Coulomb explosion depending on its initial orientation with respect to the detector. We must use a momentum convention to ensure a one-to-one mapping between geometries and measured momentum vectors as we are strictly interested in the molecular structure, and not the molecule's orientation.

For triatomic molecules, the momentum vectors can be rotated into a plane (see the `rotateMomentum.m` code listing in appendix B). We further restrict the central atom's momentum vector to lie along the $+x$-axis and that the second terminal atom sits in the $+x$ half plane. Thus we are only left with five nonzero momentum components as $\mathbf{p}_1 = (p_{1x}, p_{1y}, 0)$, $\mathbf{p}_2 = (p_{2x}, 0, 0)$, and $\mathbf{p}_3 = (p_{3x}, p_{3y}, 0)$. With conservation of momentum, $\mathbf{p}_3 = -(\mathbf{p}_1 + \mathbf{p}_2)$, and so we just need three momentum components before we can determine all the momentum vectors.

For larger molecules, the momentum vectors cannot always be rotated into a plane although we can still employ a very similar procedure. After the first three vectors are rotated into a plane, the remaining vectors may point out of the plane.

4

GEOMETRY RECONSTRUCTION BY LOOKUP TABLE

We begin our attempts at reconstructing geometries by taking a very simple approach, the use of a lookup table. Simulating Coulomb explosions is computationally cheap, or fast, so we can simulate the explosion of many geometries and create a large mapping of geometries to asymptotic momentum vectors. Then reconstructing a geometry from measured momentum vectors is simply a matter of looking up the geometry that produces the most similar momentum vector arrangement. In this chapter we will describe an implementation of this approach and use it to reconstruct molecular geometries for carbonyl sulfide (OCS) in the $OCS \rightarrow O^{2+} + C^{2+} + S^{2+}$ concerted[1] fragmentation channel, denoted as the $(2, 2, 2)$ channel.

First doing so however, we will motivate the need for a new approach to geometry reconstruction, the lookup table, by investigating the failures of the previous attempt at reconstructing molecular geometries, which relied on the Nelder-Mead simplex method. We will also take a brief historical look at lookup tables.

4.1 PREVIOUS ATTEMPT USING THE NELDER-MEAD METHOD

Brichta, Seaman, and Sanderson (2009) proposed the reconstruction of small triatomic molecules using a "simplex" algorithm. It should

1 Meaning that the molecule dissociates into its atomic fragments at the beginning of the Coulomb explosion, simplifying the system's dynamics and simulation. This is opposed to, for example, stepwise dissociation in which the two bonds break at different times, allowing for the formation of a diatomic rotor, or dumbbell, which rotates numerous times before further dissociating into its atomic fragments.

not be confused with the much better known simplex algorithm by George Dantzig, also an optimization algorithm but for linear programming (see section 5.1.1). The algorithm employed should be referred to as the Nelder-Mead method, downhill simplex method, or amoeba method to avoid confusion between the two.[2] We refer to it as the *Nelder-Mead method* for this chapter, as Wikipedia does.

4.1.1 *Previous reconstructions*

Unfortunately, Brichta, Seaman, and Sanderson (2009) only report on the reconstruction of molecular structures based on a few simulated geometries for carbon dioxide (CO_2) and formaldehyde (CH_2O). In an earlier work, Brichta et al. (2007) used this algorithm to report on reconstructions of CO_2 geometries from experimental data (see figure 15), which seems ripe for discussion and investigation, however the experimental data is not discussed and instead the focus is placed on reconstructing simulated data. They plot the ground state geometry of CO_2 as perfectly linear even though it should have a bond angle of 172.5° (Mathur et al., 1992; Siegmann et al., 2002). Each plot is reported to contain approximately 10^3 events (or geometries), however figures 15(a) and (d) appear to contain quite different amounts of data. It was also used by Bocharova et al. (2011) to report on the molecular structure of CO_2 $(2, 2, 2)$ (see figure 16). They report a molecular geometry of $\langle r_{CO} \rangle \approx 1.3\,\text{Å}$, $\langle \theta_{OCO} \rangle \approx 168°$ which is close the equilibrium geometry of $\langle r_{CO} \rangle \approx 1.16\,\text{Å}$ (Greenwood and Earnshaw, 1997) and $\langle \theta_{OCO} \rangle \approx 172°$ (Mathur et al., 1992; Siegmann et al., 2002).

They both produce nice and intuitive plots, showing what appears to be an approximation to a molecular wavefunction with two broad position distributions as one atom is fixed. Both works, however, do not report the set of geometries used to form the initial simplex.

4.1.2 *Testing the Nelder-Mead method*

Before using the Nelder-Mead to reconstruct geometries besides CO_2, such as OCS, we decided to investigate the accuracy and reliability of the Nelder-Mead method for geometry reconstruction, by testing to see whether it could reconstruct simulated geometries. By this we mean that we would chose a molecular geometry then use it to simulate a Coulomb explosion (see section 3.3) and compute the resulting asymptotic momentum vectors. These momentum vectors are then fed into the Nelder-Mead method to see whether it could recover the

2 It seems that Dantzig's simplex algorithm finds more use in fields such as optimization, operations research, and economics while the Nelder-Mead simplex method has been historically popular in scientific and engineering fields due to it's ability to run on microcomputers.

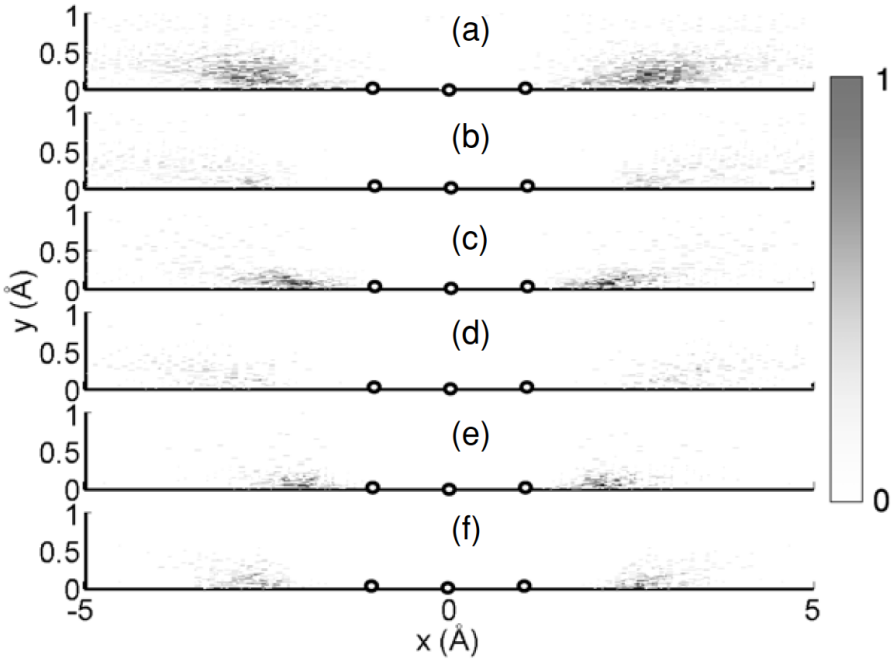

Figure 15: Reconstructed CO_2 geometries after Coulomb explosion by 50 fs laser pulses using the Nelder-Mead simplex method for the (a) $(1,1,1)$, (b) $(1,2,1)$, (c) $(1,1,2)$, (d) $(1,2,2)$, (e) $(2,1,2)$, and (f) $(2,2,2)$ fragmentation channels. The carbon atom is placed at the origin with the higher charged oxygen atom in the $+x$-quadrant. A 2D histogram is used to plot the spatial distribution of the two terminal oxygen atoms with darker colors indicating a higher count rate. The histogram is normalized such that the maximum count rate is 1. The three circles represent the equilibrium or ground state geometry of CO_2. Figure from Brichta et al. (2007).

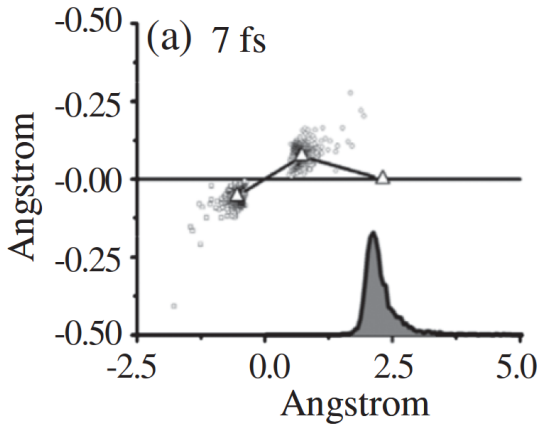

Figure 16: Reconstructed CO_2 geometries in the $(2,2,2)$ charge state after explosion by a 7 fs laser pulse using the Nelder-Mead simplex method. The oxygen in the $x > 0$ plane is restricted to the x-axis with a probability density curve showing its distribution in space, presumably calculated using a kernel density estimate, although unreported. The circles pinpoint the location of the carbon atom and the oxygen atom in the $x < 0$ plane relative to the fixed oxygen atom and the triangles presumably pinpoint the centroid (or average) position of each atom. Figure from Bocharova et al. (2011).

CO_2 simplex			OCS simplex		
r_{12} (Å)	r_{23} (Å)	θ_{OCO} (°)	r_{CO} (Å)	r_{CS} (Å)	θ_{OCS} (°)
0.9	1.2	165	1.8	2.4	165
1.4	1.0	165	0.8	1.5	165
0.9	1.9	165	1.8	3.8	165
0.9	1.2	180	1.8	2.4	180

Table 1: Initial CO_2 and OCS simplices used for testing the Nelder-Mead method.

original molecular geometry, as done by Brichta, Seaman, and Sanderson (2009).

Methodology: attempting to reconstruct simulated geometries

Simulated geometries are used for this analysis because we can check exactly how far off the reconstructed geometry is from the known correct solution. The simulated geometry only experiences the Coulomb force and so at least one solution is guaranteed to exist by our deterministic classical simulation. Simulated geometries are thus the easiest geometries to reconstruct and serious doubts regarding an algorithm's utility must be raised if it cannot reconstruct them. When dealing with experimentally measured data, we do not know the geometries *a priori* and so we need a trustworthy reconstruction algorithm in order to trust any of the results it produces.

We attempted to reconstruct molecular structures for CO_2 and OCS, both in the $(2, 2, 2)$ fragmentation channel. Starting from their equilibrium geometries we varied one parameter at a time to test whether the Nelder-Mead method could recover the geometry. The initial simplex we used for reconstructing CO_2 and OCS consist of four points (or initial guess geometries) and are tabulated in table 1.

Results

Figure 17 shows the reconstruction results for CO_2 $(2, 2, 2)$. We take the ground state geometry of CO_2 to be $r_{CO} = 1.16\,\text{Å}$ (Greenwood and Earnshaw, 1997) and $\theta_{OCO} = 180°$ although $\theta_{OCO} = 172°$ would be more accurate (Mathur et al., 1992; Siegmann et al., 2002).

In the first row of figure 17, the first C−O bond length (r_{12}) was varied to create an "input geometry" which underwent a simulated Coulomb explosion. The resulting momentum vectors from the explosion were fed into the Nelder-Mead method which converged to a geometry we call the reconstructed or "output" geometry. The second C−O bond length (r_{23}) and the bond angle θ were varied in the second and third rows respectively. Solid black lines indicate the expected output geometry, so deviations indicate a failure on the method's

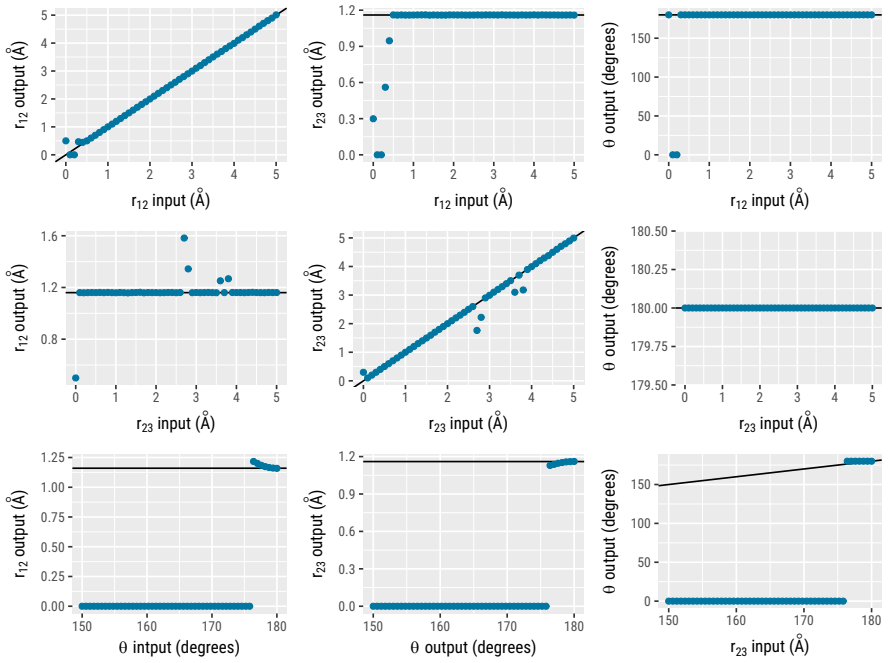

Figure 17: Testing the Nelder-Mead method's ability to reconstruct CO_2 geometries in the (2,2,2) fragmentation channel by starting with the ground-state geometry of CO_2 and varying each parameter one-by-one. Solid black lines indicate the expected output geometry.

ability to reconstruct the geometry. We see that the algorithm can accurately reconstruct the geometry when a single bond length is varied but completely fails once the bond angle is below approximately 176°, at which point the reconstruction code was instructed to simply return a row of zeros, thus the large number of data points with output bond lengths and angles of zero in the third row.

Figure 18 shows the reconstruction results for OCS $(2,2,2)$. For OCS, we took the ground state geometry to be $r_{CO} = 1.1578$ Å, $r_{CS} = 1.5601$ Å (Lide, 2007), and $\theta_{OCS} = 180°$ although $\theta_{OCS} = 175°$ (Wales et al., 2012b) would be more accurate (Wales et al., 2012b).

In the first row of figure 18, the C−O bond length ($r_{CO} \equiv r_{12}$) was varied to create an "input geometry" which underwent a simulated Coulomb explosion. The resulting momentum vectors from the explosion were fed into the Nelder-Mead algorithm which converged to a geometry, the "output geometry". The C−S bond length ($r_{CS} \equiv r_{23}$) and the bond angle θ were varied in the second and third rows respectively. A triatomic molecule with a bond angle greater than 180°, $180° + x°$ for example, is the same as a molecule with a bond angle of $180° - x°$, and so it should not be necessary to attempt the reconstruction of geometries with bond angles greater than 180° but we attempt it nonetheless with the aim of testing the method's robustness. Solid black lines indicate the expected output geometry, so deviations indicate a failure on the algorithm's ability to reconstruct the geometry.

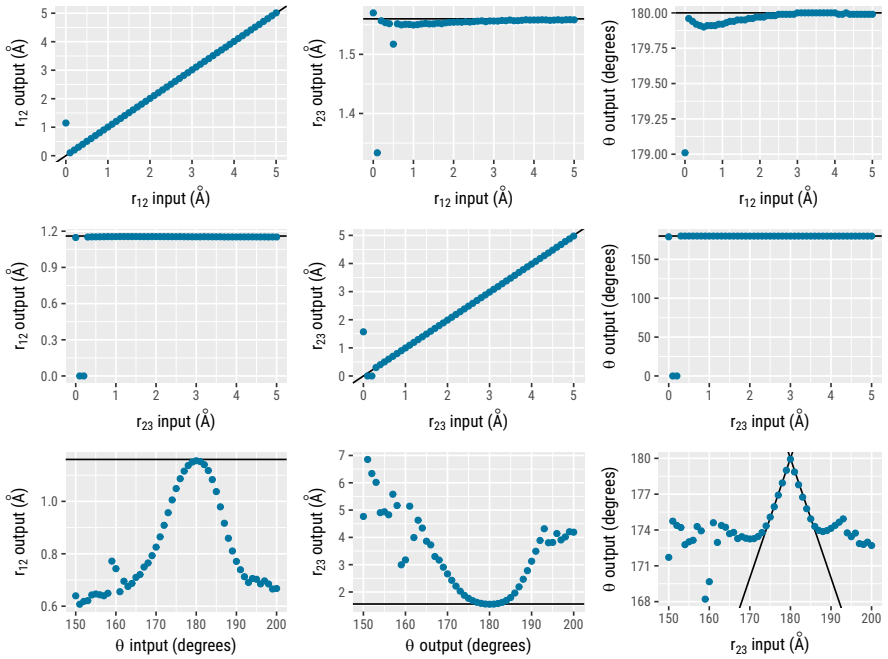

Figure 18: Testing the Nelder-Mead method's ability to reconstruct OCS (2,2,2) geometries by starting with the ground-state geometry of OCS and varying each parameter one-by-one. Solid black lines indicate the expected output geometry.

We see that the algorithm can accurately reconstruct the geometry when a single bond length is varied but performs worse and worse as the molecule bends, even slightly. There are some slight differences between molecules with bond angles of $180° + x°$ and $180° - x°$, but the absolute errors on the reconstructed bond lengths and angles seems to be similar for both cases.

Discussion

So we see that the Nelder-Mead method successfully retrieves the bond lengths for the majority of cases but not the bond angles. In the case of CO_2 it seems to completely fail at retrieving the geometry once the bond angle falls below approximately 176°. For OCS, the absolute error on the reconstructed bond angle seems to increase quadratically as the molecule bends, even slightly.

A more thorough analysis should vary multiple parameters at once to more fully explore the state space of the problem, however if the method cannot even reconstruct geometries that slightly different from the equilibrium state by a single parameter as we have done, then the results of a more thorough analysis will probably be even more worrying.

It should be noted that the Nelder-Mead method is quite sensitive to the geometries chosen to represent the initial simplex. Changing them could significantly impact the algorithm's ability to converge on

the correct geometry. Of course, this does suggest that there may exist a set of initial geometries that significantly improve the algorithm's reliability, however, I could not find such a set even after several dozen attempts. Some choices improved the recovery of bond angles at the cost of failing to recover the correct bond lengths. The initial simplex used to produce figures 17 and 18 was chosen to maximize the number of successful reconstructions for very straight molecules which constitute the majority of cases, however molecules with bond angles $\theta < 175°$ are still very common and so even with this choice, we are very unsure about the accuracy of the majority of reconstructions.

The Nelder-Mead algorithm is an ad-hoc or heuristic algorithm for nonlinear optimization that can be used without computing derivatives of the objective function[3]. It was first generalized to minimizing functions by Nelder and Mead (1965) based off ideas by Spendley, Hext, and Himsworth (1962). Nelder and Mead were researchers at the National Vegetable Research Station in Warwick, England, leading an unidentified laboratory to doubt if these "turnip bashers could be numerate" in response to their optimization algorithm (Wright, 2010). It has enjoyed widespread popularity due to its ease of implementation and intuitive inner workings but it is not appropriate to every problem. In fact, it is not guaranteed to converge except for strictly convex problems in 1 and 2-dimensions (Lagarias et al., 1998) and thus fails when applied to some problems. It can even converge to non-stationary points in some cases (McKinnon, 1998). Studies would sometimes introduce modifications to the Nelder-Mead method that would improve its performance on a specific problem Wright (2010). Unfortunately, I believe geometry reconstruction is not an appropriate problem for the Nelder-Mead method.

Conclusion

Perhaps Brichta et al. (2007) and Bocharova et al. (2011) happened to find an optimal set of geometries to form their initial simplex used to produce figures 15 and 16, however no mention of it is made anywhere, and thus we are unable to replicate their results or use the Nelder-Mead method for further reconstructions. The necessity and difficulty of fine tuning required should cast some doubt over their geometry reconstructions and any consequent conclusions in their respective works. We already have enough information to distrust the Nelder-Mead method and begin searching for a new approach.

3 Wright (2010) provides a great discussion of the Nelder-Mead method, ending with a comment by John Nelder regarding his algorithm, "Mathematicians hate it because you can't prove convergence; engineers seem to love it because it often works." Press et al. (2007, section 10.4) describes the algorithm in detail and provides a well-commented C++ implementation.

4.2 AN ASIDE ON LOOKUP TABLES

The idea of using a lookup table to speed up calculations is as old as mathematics itself with examples dating back to some of the earliest mathematical texts produced by ancient Egyptian scribes during the Twelfth Dynasty of Egypt (circa 1990–1800 BC) (Neugebauer, Sachs, and Götze, 1945, p. 1, footnote 4). The Egyptian Mathematical Leather Roll, a particularly well-preserved example of Egyptian mathematics dating back to circa 1650 BC, contains perhaps the first such complete lookup table and tabulates 26 sums of unit fractions equaling another unit fraction. Glanville (1927) reports on the leather roll, housed at the British Museum, and gives a photograph (figure 19) and schematic (figure 20) of the table.

The Egyptian Mathematical Leather Roll, as well as the more famous and extensive Rhine Mathematical Papyrus, were brought to the British Museum in 1864, however it took decades before archaeologists knew how to treat the leather to prevent its disintegration upon unrolling it. Scott and Hall (1927) provides details of this process and claim that, "the dissemination of the knowledge of the chemical treatment of the leather, is of greater value than the publication of the contents inscribed on it" upon discovery of what he considered to be a rather mundane table.

An even more impressive lookup table is found in the extensive Rhind Mathematical Papyrus, which (seemingly) methodologically expresses the fractions $2/n$ for odd $n \in \{3, 5, \ldots, 101\}$ as the sum of 2–4 unit fractions! We would have chosen to present photographs of the papyrus had the figures of (Glanville, 1927) not been of such high clarity. Gillings (1982) reports extensively on this $2/n$ table and the several dozen problems posed and solved on the papyrus, making extensive use of the table. The same table, verbatim, has found to be in use by scribes more than a millennium after its creation suggesting that it was of great utility. To this day, scholars argue about exactly how the scribes knew to construct the table and what methods they used (Abdulaziz, 2008; Gillings, 1974).

Maybe it is not so surprising that our first approach is to use a lookup table, after all many others have done the same.[4]

4 Neugebauer, Sachs, and Götze (1945) reports on mathematical cuneiform texts used by the ancient Babylonians during similar time periods as well. Other lookup tables of historical importance include a 98-column Roman multiplication table from 493 A.D. (Maher and Makowski, 2001) and an ancient Indian sine table circa 499 C.E. (Hayashi, 1997).

Figure 19: Photograph of columns 3 (right) and 4 (left) of the Egyptian Mathematical Leather Roll. Columns 3 and 4 are duplicates of columns 1 and 2. For example, row 9 of column 3 translates to $\frac{1}{50}\frac{1}{30}\frac{1}{150}\frac{1}{400}\quad\frac{1}{16}$ in modern fraction notation where addition is implied. Figure 20 is a schematic of the table photographed here. Figure from Glanville (1927) which is accompanied by a translation of the hieroglyphics.

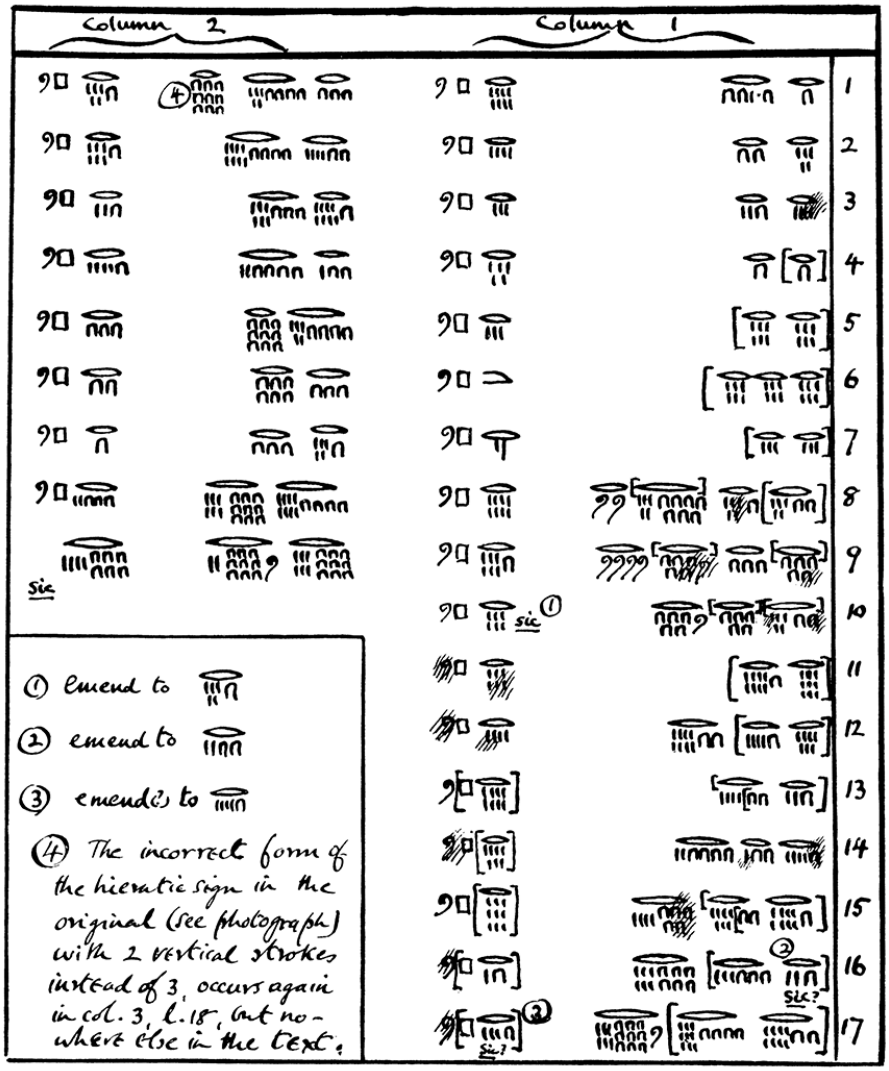

Figure 20: Schematic of columns 1 and 2 of the Egyptian Mathematical Leather Roll. Columns 3 and 4 are duplicates of columns 1 and 2. For example, row 1 of column 1 translates to $\frac{1}{10}\frac{1}{40}\quad\frac{1}{8}$ in modern fraction notation where addition is implied. Figure 19 is a photograph of the table outlined here. From Glanville (1927) which is accompanied by a translation of the hieroglyphics.

4.3 IMPLEMENTATION

The idea behind using a lookup table is quite simple: many Coulomb explosions are simulated (see section 3.3) using a wide variety of molecular structures as the initial condition, and the resulting momentum vectors from each simulation are stored. This creates a mapping from molecular structures to momentum vectors. Thus by storing the results from many simulations, we end up with a lookup table. To determine the structure belonging to a certain set of observed momentum vectors, you simply read the table in reverse and search for the momentum vectors that most closely match the observed set.

In our case, we quantify this through the use of the ℓ_2-norm squared between the sets of vectors

$$\left\|\mathbf{p} - \mathbf{p}'\right\|_2^2 = \sum_{i=1}^{3N} (p_i - p_i')^2 \tag{4.1}$$

for an N-atom molecule and where i sums over each momentum component for each atom, e. g. $O_x, O_y, \ldots, S_y, S_z$ for the OCS molecule. We may sometimes refer to as the *objective function* and its value as the *absolute error*.

Each molecular geometry and its corresponding post-explosion momentum vectors can be stored in a single row consisting of 9 entries, 3 to describe the molecular geometry of a triatomic molecule and 6 to describe the momentum vectors. While only 3 scalars are enough to describe the momentum vectors produced by a simulated Coulomb explosion (the last momentum vector can be calculated from the others using conservation of momentum, see section 3.4), we store all 6. This is to allow for comparison with real momentum data where ignoring some component measurements may skew the reconstruction, especially if these specific components happened to carry a large uncertainty. We have not tested this approach when only a subset of our measurements are used, however it may provide a powerful dimensionality-reduction measure when reconstructing larger molecules.

A technical detail that is important to mention is the reason for storing and using bond lengths in units of picometers and bond angles in degrees. The bond lengths and angles differ numerically by approximately 12 orders of magnitude in SI units and so this was done to keep the parameters all on the same order of magnitude, mainly to avoid potential numerical instabilities and to make data analysis and plotting more convenient. This becomes especially important for the optimization approach we take in chapter 5 where derivatives and Jacobian matrices need to be calculated. Equivalently, we could have chosen angstroms (Å) and radians.

A cursory argument in favor of the lookup table would suggest that this approach is simple to implement, fast, and precise as the time-

consuming task of simulating many geometries is done only once then stored. Searching through the lookup table to find a match takes linear time $\mathcal{O}(n)$ where n is the number of table entries,[5] and the precision is set when simulating the many geometries (0.05 Å and 0.25 Å in our case). Big-$\mathcal{O}$ notation is used to refer to the asymptotic behavior of an algorithm's run time or storage space requirements. Even issue we can foresee is that this approach assumes the true geometry is in the vicinity of the geometry found using the lookup table. This is equivalent to assuming that every local minimum is a global minimum (or that this is a convex optimization problem).

Momentum lookup tables can be generated using the `simulateMomenta.m` function and the lookup table is implemented in the `lookupGeometry.m` file (see appendix B for code listings).

4.3.1 *Computational space complexity*

One of the disadvantages of using a lookup table for this particular problem is the amount of storage space it occupies. As a concrete example, our lookup table for OCS contained geometries spanning a cube in phase space ($0.50\,\text{Å} \leq r_{CO}, r_{CS} \leq 5.00\,\text{Å}$, $140° \leq \theta_{OCS} \leq 180\,\text{Å}$) that we believe should contain all physically realizable geometries. Individual geometries spanned the discrete ranges $r_{CO}, r_{CS} \in [0.50, 0.55, \ldots, 4.95, 5.00]$Å and $\theta_{OCS} \in [140.00°, 140.25°, \ldots, 179.75°, 180.00°]$ giving a precision of 0.05 Å for the bond lengths and 0.25° for the bond angle. This gives us a lookup table with $91 \times 91 \times 161 = 1,333,241$ entries. Since each entry contains 9 64-bit floating-point numbers, it requires $9 \times 8\,\text{B} = 72\,\text{B}$ to store each entry.[6] To store the entire table, that is $1,333,241 \times 72\,\text{B} = 95.993\,\text{MB}$. Such a table, if stored in human-readable ASCII such as in a comma-separated value file would take up more space (262 MB) but can be compressed efficiently (24 MB using 7-zip).

For a molecule with N atoms, each entry would require $3N - 6$ entries to describe the geometry and $3N - 3$ to describe the momentum vectors for a total of $6N - 9$ entries per geometry. If d_i denotes the number of values simulated for parameter i (e. g. $d_1 = 91$, $d_2 = 91$,

5 It takes linear time because we must search through every single entry before concluding that we have found the best match. If we could sort our entries in some way then we could perform a binary search instead taking logarithmic time $\mathcal{O}(\log n)$ however that would require distilling each entry to an appropriate scalar, which would differ for each geometry and take linear time to perform anyway.

6 There are 8 bits in a byte (B) so a single 64-bit float would take 8 bytes to store. MATLAB uses 64-bit double precision floating point numbers by default but other programming languages may use single precision 32-bit floating point numbers by default, even when running on a 64-bit processor.

and $d_3 = 161$ for our lookup table described above) then the total number of entries will be

$$\prod_{i=1}^{3N-6} d_i = d_1 \times d_2 \times \cdots \times d_{3N-6}, \tag{4.2}$$

or simply d^{3N-6} if $d_i = d$ for all i so that we use the same number of simulated values for each parameter. We require 8 B for each of the $6N - 9$ enetries requires to describe each geometry, which we store for d^{3N-6} geometries. Thus the total storage space used up by such a lookup table, in bytes, is

$$8(6N - 9)d^{3N-6} \sim \mathcal{O}(Nd^N) \tag{4.3}$$

which increases exponentially with an increase in the number of atoms N and follows a power law in the number of simulated values d per parameter. The number of simulated values d required to achieve a certain precision $\epsilon \ll 1$ is $d = (p_{\max} - p_{\min})/\epsilon$ where $[p_{\min}, p_{\max}]$ is the range of possible parameter values we wish to simulate. So the size of the lookup explodes very quickly with increased precision requirements ($\epsilon \to 0$) as well, as the desired precision ϵ is inversely proportional to the step size d.

Let us now calculate the size of a higher resolution lookup table and a lookup table for a 4-atom molecule. For example, a lookup table that is five times more precise than ours ($\epsilon_r = 0.01$ Å and $d_r = 451$ for bond lengths, $\epsilon_\theta = 0.05°$ and $d_\theta = 801$ for bond angles) takes up $8\,\text{B} \times 9 \times 451^2 \times 801 = 11.7\,\text{GB}$ of storage space, or $122 \approx 5^3$ times more space. Storing such a table in memory, and thus searching speed which used to only take linear time, starts to become a major concern. Going up to a larger 4-atom molecule such as acetylene and using the same coarse precision as our lookup table did ($\epsilon_r = 0.05$ Å and $d_r = 91$ for bond lengths, $\epsilon_\theta = 0.25°$ and $d_\theta = 161$ for bond angles), a lookup table would take up $8\,\text{B} \times 15 \times 91^3 \times 161^3 = 377\,\text{TB}$ or almost 4 million times larger, now requiring a distributed database management system; completely overkill to solve a relatively simple problem. It seems that this lookup table approach will not scale at all to larger molecules without massive sacrifices in precision. Plus, using gigabytes to store a lookup table suggests we must seriously look for a method that does not require large amounts of data to reconstruct geometries.

4.3.2 *Zooming in for more precise reconstructions*

Another immediate disadvantage of such an approach is that you are limited to recovering only the geometries that are included in the table. One way to increase the precision of the lookup table approach without using up additional storage space is to use the table as a

coarse first-order approximation. Once a geometry is found using the lookup table, you can dynamically simulate many similar geometries (neighbouring geometries in phase space) on-the-fly which can then be searched for a more precise match. This can be applied iteratively, reducing the volume of phase space searched at each iteration, until a desired precision is reached.

In our case, we chose to increase the precision by a factor of 5 each iteration so that we search through a cube in phase space with a volume that is 5^3 times smaller each iteration (in an attempt to precisely locate the local minimum). Each iteration we simulate geometries with 10 values for each parameter, so 10^3 geometries in total. We stop after 5 iterations or after a desired error threshold is reached. Typically an error of 10^{-48} gives 1-2 significant figures of precision while an error of 10^{-50} gives 2-3 significant figures. This step can be computationally expensive, requiring the additional simulation of thousands of geometries per geometry recovered, however a single simulation takes on the order of 10 ms to complete on a personal laptop so it will complete in a reasonable timeframe and this lets us arbitrarily increase the precision of our reconstruction without using up an exorbitant amount of storage space. It does feel quite wasteful though as it seems that the entire field of mathematical optimization attempts to solve this very problem of finding minima.

One possible improvement is to store these thousands of extra simulations as extra lookup table rows, effectively extending the lookup table. This storing or caching of the results of expensive function calls is termed *memoization*. It may help if the geometries being reconstructed tend to lie close to each other, however in practice the sparsity of reconstructed geometries relative to the resolution of these extra rows results in almost no improvements in performance.

4.3.3 *Using simulations to test accuracy*

Since we used simulations to test the accuracy of the Nelder-Mead simplex method in section 4.1.2, it is only fair that we subject the lookup table to the same analysis. Figure 21 shows the results of this test.

We see that the lookup table is able to reconstruct the vast majority of the geometries quite well, except for a single outlier, suggesting that this approach is already much more reliable as no fine tuning is required at all. It is important to note that we did not simulate geometries that are already contained in the lookup table otherwise the reconstruction is trivial as the table already contains the answer, which is extremely unlikely if we are dealing with experimental data.

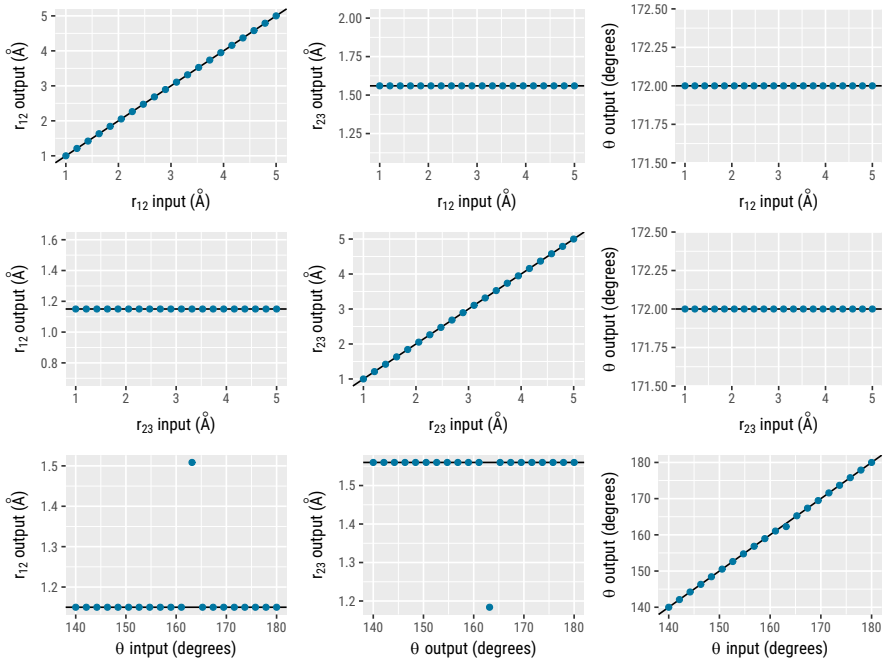

Figure 21: Testing the lookup table's ability to reconstruct OCS (2,2,2) geometries by starting with the ground-state geometry of OCS and varying each parameter one-by-one. Solid black lines indicate the expected output geometry.

4.4 RECONSTRUCTION OF EXPERIMENTAL DATA

We will now attempt to use the lookup table to reconstruct the OCS $(2, 2, 2)$ geometry as an example. Figure 22 shows one such geometry for the Coulomb explosion by a 7 fs laser pulse and may form the first frame of a molecular movie.

Here we have plotted the geometries by placing the center of mass at the origin, giving us three probability distributions, one for each atomic fragment which allows us to see the variance in each atomic fragment's position. We could have placed the carbon at the center resulting in just two distributions, one for each of the terminal atoms much like Brichta et al. (2007) did in figure 15 however we felt that this approach provides less information. Bocharova et al., 2011, in figure 16, plot the geometries with a terminal oxygen at a fixed position along the x-axis and include a one-dimensional marginal distribution of the oxygen's position along the x-axis, which is an improvement but we feel that it still provides less information. We use the same visualization method as Légaré et al. (2005b) (see figure 6) who revert to a plot like the one by Bocharova et al. (2011) in their later work (Légaré et al., 2005a). The modal geometry is calculated to be $r_{CO} = 1.74$ Å, $r_{CS} = 1.59$ Å, $\theta_{OCS} = 172.7°$ while the average geometry of $r_{CO} = 1.93$ Å, $r_{CS} = 1.61$ Å, $\theta_{OCS} = 171.6°$ is slightly larger due to outliers.

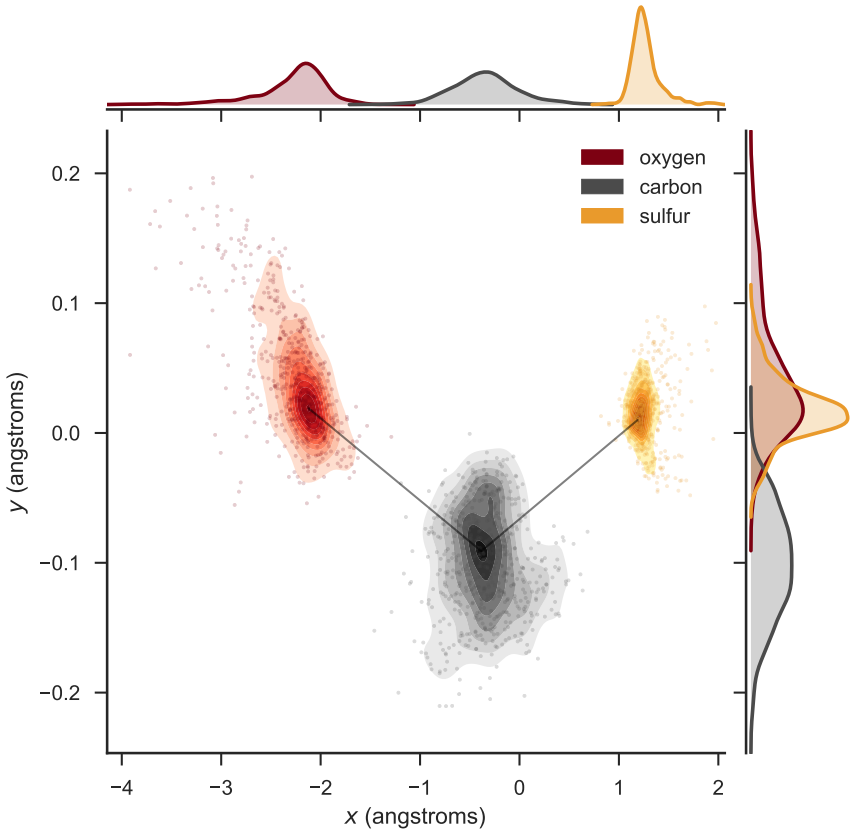

Figure 22: Scatter plot showing the molecular geometry of OCS following Coulomb explosion by a 7 fs laser pulse for the $(2, 2, 2)$ fragmentation channel. Each geometry is represented by three colored points, one for each atomic fragment; red for oxygen on the left, black for carbon in the center, and yellow for sulfur on the right. The colors were chosen to imitate the CPK coloring convention. Geometries are plotted such that the molecule's center of mass is at the origin to showcase the variance in each atomic fragment's position, and are rotated such that a vertical line bisects the O−C−S bond angle. Bivariate kernel density estimates (with a Gaussian kernel), plotted as shaded-in contours, are used to estimate the the probability density of each atomic fragment's position. Solid black lines are drawn between the peaks of each atomic fragment's kernel density estimate to illustrate the *modal geometry* or most likely geometry. Along the top of the plot, marginal distributions show the probability density of each atomic fragment's position along the x-axis and the same is done for the y-axis along the right. The molecule is almost straight but an aspect ratio of approximately $12 : 1$ is employed to showcase variability in the y-axis.

Pulse width (fs)	Geometries	Reconstructions	Success
7	795	584	73%
30	1000	518	52%
60	358	154	43%
100	531	226	43%
200	500	190	38%

Table 2: Statistics for geometry reconstruction by lookup table.

An argument could be made for each visualization method, however we will come back the issue of geometry plotting later in chapter 5, where we will look at correlations between bond lengths.

The calculated modal and average geometries may appear quite worrying as they suggest a stretching of the C−O bond while the C−S bond and the bond angle remains close to equilibrium. This is mainly due to ion motion from the molecule's interaction with the laser pulse during the ionization process.

A critical issue worth investigating is quantifying how much uncertainty there is in these reconstructed molecular geometries. This is a complicated problem we will tackle in chapter 6.

We can repeat this analysis for the other laser pulse widths (30 fs, 60 fs, 100 fs, and 200 fs). In table 2 we tabulate the number of geometries we were able to successfully reconstruct, which decreases as the pulse length increases indicating a greater difficulty in reconstructing geometries that are further from equilibrium. Or it could indicate a greater difficulty in reconstructing geometries when the atomic fragments had some significant initial momentum. The longer the laser pulse, the more time the molecule has to distort its structure in response to the laser's intense electric field, reducing the validity of the assumption that the molecule begins the Coulomb explosion process in its equilibrium geometry.

To conserve space, the geometry reconstructions for Coulomb explosion by 30 fs, 60 fs, 100 fs, and 200 fs laser pulses are provided in appendix A (figures 39–42).

Of particular interest might be the average and modal (most likely) geometries recovered, which we tabulate in table 3.

Pulse length (fs)	Modal geometry r_{CO}	r_{CS}	θ_{OCS}	Average geometry r_{CO}	r_{CS}	θ_{OCS}
(Equilibrium)	1.16	1.56	172	1.16	1.56	172
7	1.74	1.59	173	1.93	1.61	172
30	2.34	1.74	172	2.50	2.04	170
60	2.43	1.96	171	2.56	2.16	170
100	2.52	2.16	171	2.72	2.46	169
200	3.17	2.22	171	3.10	2.72	163

Table 3: Average and modal geometries reconstructed using a lookup table. Bond lengths are given in angstroms ($1\,\text{Å} = 10^{-10}\,\text{m}$) and bond angles in degrees.

4.5 DEGENERATE MOLECULAR GEOMETRIES

In chapter 1 we hinted at the fact that due to the ill-posed nature of the geometry reconstruction inverse problem, multiple solutions may be possible. While investigating the reconstructed geometries, this feature surprisingly did appear for the OCS molecule and the lookup table can help provide some insight into these multiple solutions, which we will call *degenerate geometries*.

Figure 23 shows an example of two degenerate geometries that both produce the same set of momentum vectors after a Coulomb explosion, $\mathbf{p}_O = (0.398, -1.75, 0) \times 10^{-20}\,\text{kg m/s}$, $\mathbf{p}_C = (0.853, 0, 0) \times 10^{-20}\,\text{kg m/s}$, and $\mathbf{p}_S = (-1.25, 1.75, 0) \times 10^{-20}\,\text{kg m/s}$ (described in our convention).

More specifically, it is a 3D scatter plot in phase space of the 200 best geometries matching this particular set of measured momentum vectors. Each point is colored according to the base-10 logarithm of the absolute error (2-norm) squared between the measured momentum vectors and the momentum vectors resulting from Coulomb exploding the geometry corresponding to that data point. So the darkest purple corresponds to an error of just under $10^{-47.8}$. The 200 best geometries are clustered into two separate regions, indicating that the particular set of measured momentum vectors mentioned above could have resulted from the Coulomb explosion of two very different molecular geometries, one where the C−O bond is more stretched, and another where the C−S bond is more stretched and the molecule is less bent. Interestingly, the points do not seem to form balls or blobs in phase space, but rather possess elongated and angled rod-like distributions. The region with the stretched C−O bond seems to have more points but this does not suggest that this geometry is more likely. Rather, it may suggest that it is easier to converge to, or that it may have a larger *basin of attraction*. We see that there are many yellow data points (relatively high error) surrounding one or two purple

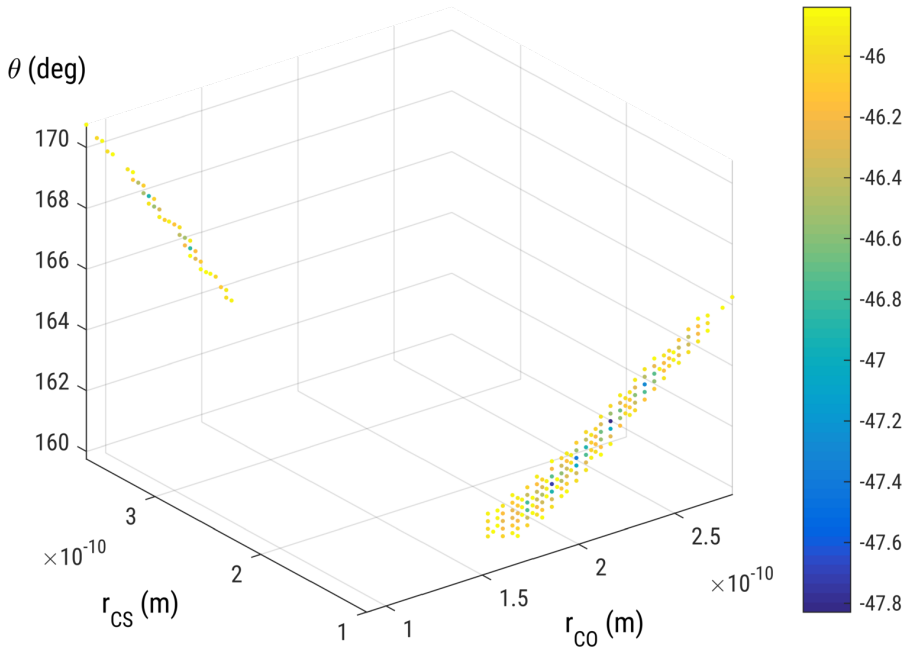

Figure 23: 3D scatter plot in phase space of the 200 best geometries matching a particular set of measured momentum vectors, showing the existence of two degenerate geometries. Each point is colored according to the base-10 logarithm of the absolute error (2-norm) squared between the measured momentum vectors and the momentum vectors resulting from Coulomb exploding the geometry corresponding to that data point.

points (relatively low error) for each region indeed showing the low-resolution nature of our lookup table. This same information may be plotted using 2D color mapped volumetric slices or contour slices which may even be animated to provide a visual scan through the full phase space volume. However due to the angled distribution formed by the spatial sets and the significance of only one or two data points, it becomes very difficult to visually locate the local minima even with angled slices. Another visualization method may be to use convex hulls or alpha shapes which will also allow us to assign each region a shape and volume (see section 6.1.2).

In figure 24 we take these two degenerate geometries, Coulomb explode them, and plot their trajectories to show that even though they show slightly different dynamics, both simulations result in the exact same momentum vectors (and kinetic energy) being measured.

The plots on the left correspond to an OCS molecule with $r_{CO} = 1.8949\,\text{Å}$, $r_{CS} = 1.2990\,\text{Å}$, and $\theta_{OCO} = 160.601°$ while the plots on the right correspond to an OCS molecule with $r_{CO} = 2.486\,\text{Å}$, $r_{CS} = 1.0755\,\text{Å}$, and $\theta_{OCO} = 164.568°$. Triatomic molecules explode in a plane so their momentum vectors can be plotted in the $p_x p_y$-plane although the atomic fragments will possess some momentum in the z-direction due to the presence of the constant electric field, it will not deviate from equation (3.2) and is irrelevant for these simulated

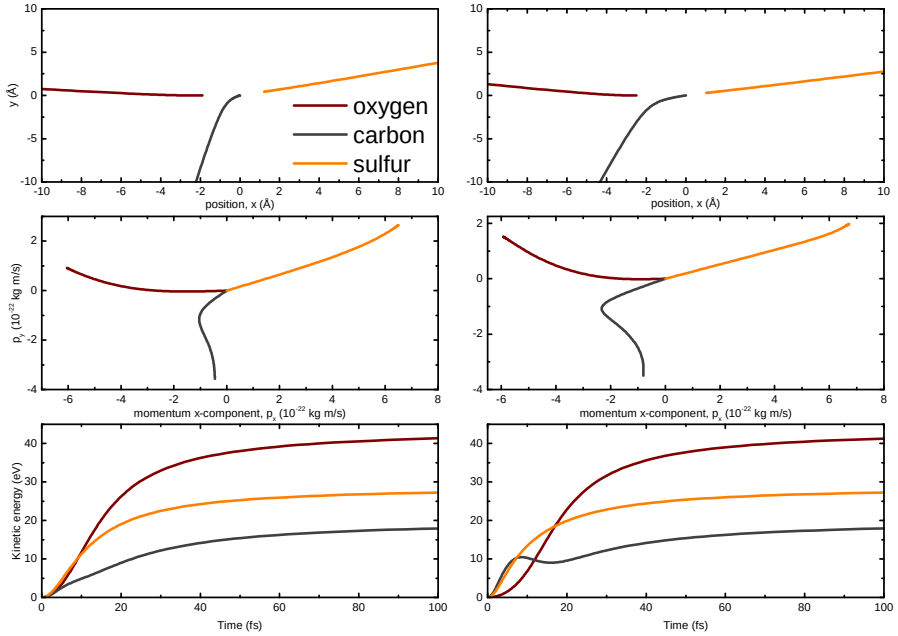

Figure 24: Atomic trajectories in position and momentum-space, and kinetic energy, for the oxygen, carbon, and sulfur atoms of two degenerate OCS molecular geometries in the $(2, 2, 2)$ charge state undergoing a Coulomb explosion staring from rest.

Coulomb explosions. The molecular geometries are quite different yet when they undergo a Coulomb explosion, they produce the exact same set of momentum vectors once rotated into our convention (see section 3.4). By that we mean that we can make the absolute error (or 2-norm) between the two sets of momentum vectors arbitrarily small with increased precision in describing the geometries. We see that the molecular dynamics are a little different yet in both cases, all three atoms emerge with the exact same kinetic energy. These two geometries were found by searching through the entire lookup table for the geometries best matching a particular set of measured momentum vectors (see figure 23). The lookup table actually does not give us enough precision to recover these geometries to several decimal places so we used the lookup table to find a measurement corresponding to two degenerate geometries, and then used the optimization method from chapter 5 to actually recover the geometries with much higher precision.

The high precision to which we were able to recover the two degenerate geometries suggests that presence of degenerate geometries corresponds to the existence of multiple global minima. This suggests the use of a method that can locate multiple minima, which we will employ in the next chapter.

4.6 CONCLUSIONS

Geometries may be reconstructed using a lookup table—We find this approach more reliable and robust that the Nelder-Mead simplex method approach employed by Brichta, Seaman, and Sanderson (2009), and it has provided us with some additional insight into the problem of reconstructing geometries especially regarding the existence of degenerate geometries. This makes the lookup table potentially useful for future investigations and reconstructions of triatomic molecules. However, it seems that it will not scale to molecules containing more than 3 atoms, motivating the need for another method.

4.6.1 *Lessons learnt*

Inability to precisely reconstruct geometries—Using just the lookup table itself, you are limited to recovering molecular geometries contained in the lookup table and nothing in between, severely limiting the resolution of reconstructed geometries.

Inability to differentiate between global and local minima—The lookup table cannot choose between two degenerate geometries and due to its low resolution it cannot distinguish between global and local minima. It may be easily modified to report on multiple solutions but doing so may be computationally expensive. A more sophisticated approach should be sought first.

Inability to scale to larger molecules—As discussed in section 4.3.1, trying to use the lookup table to reconstruct molecules containing even as little as 4 atoms requires prohibitive amounts of storage space (>100 TB) to store the lookup table. This may be circumvented by fixing some molecular parameters, such as maybe the C≡C triple bond length in acetylene, if they vary very little and are not of interest. However, this fix only reduces the dimensionality by one so it is unlikely to be an overall solution.

4.6.2 *Future applications for the lookup table*

Finding an initial or first-order solution or initial guess—As the lookup table may quickly find coarse-grained geometries, the recovered geometry may be used as an initial guess or first-order solution for another algorithm.

Visualizing the reconstruction problem for a specific set of momentum vectors—Such a plot would assign a scalar error value to every point in phase space, allowing us to quickly visualize the objective function (as done in figure 23 but for all the geometries, not just the

best 200). This would visualize the location of degenerate geometries and provide some insight into the nature of the geometry reconstruction problem. For example, if it is smooth and convex with a single global minimum, it would suggest that geometry reconstruction in that particular case is computationally easy. However, a jagged and discontinuous function with multiple minima and saddle points would suggest that geometry reconstruction is much more difficult. For triatomic molecules the 3-dimensional space may be visualized using colormapped volumetric slices or contour slices. Unfortunately this becomes unfeasible for larger molecules due both the size of the lookup table and the difficulty of visualizing an n-dimensional scalar function ($n > 3$). Instead of generating a lookup table for larger molecules, Coulomb explosions can be simulated for every geometry of interest, if the resolution is reduced. Fixing certain parameters would reduce the dimensionality of the objective function, allowing 2-dimensional slices of it to be easily visualized. Or one could look towards methods of visualizing high-dimensional scalar functions.

Finding degenerate regions of phase space quickly—At least for triatomic molecules, where generating and storing a lookup table is feasible, it may be quickly searched to determine whether a set of measured momentum vectors correspond to just one geometry, or multiple degenerate geometries.

5

GEOMETRY RECONSTRUCTION USING CONSTRAINED NONLINEAR OPTIMIZATION

In the previous chapter, we saw that a lookup table approach could be used to perform geometry reconstruction using Coulomb explosion imaging. However, numerous drawbacks including the lookup table's exponential computational space complexity (severely limiting its scalability to larger molecules) and its inability to both provide precise reconstructions as well as distinguish between local and global minima leave much to be desired.

In this chapter, we will approach the task of geometry reconstruction as an optimization problem, allowing us to utilize the much more sophisticated methods of constrained nonlinear optimization. Before we describe our reconstruction methods, it will be worthwhile to provide some background on the theoretical framework underpinning the optimization algorithms we utilize. We will then describe our implementation and use it to reconstruct the same molecular structures seen in the previous chapter, allowing for some comparison between methods and for some verification of the results. We will then further study the individual geometries recovered and the nature of the degenerate geometries seen in section 4.5.

5.1 MATHEMATICAL OPTIMIZATION

We will take a massively expedited tour of mathematical optimization with the aim of explaining the inner workings of the primal-dual

interior point methods used for nonlinear constrained optimization in this chapter. To understand how these methods operate, we will need to introduce some important concepts in optimization, namely the duality principle, the Karush-Kuhn-Tucker (KKT) optimality conditions, and the general concept of a trust region. This is prefaced with a brief introduction to the subject.

No knowledge of mathematical optimization is required, however we will assume some background knowledge throughout this section, namely a familiarity with linear algebra, matrix algebra, vector calculus, and some elementary concepts in analysis. Boyd and Vandenberghe (2004) provides an excellent introduction to mathematical optimization, particularly convex optimization, and their freely-available textbook is accompanied by video lectures and lecture slides. We will follow their textbook for sections 5.1.1 to 5.1.3. Our problem is nonconvex, however, and so we also turn to Nocedal and Wright (2006) for sections 5.1.4 to 5.1.5, who discuss the more advanced interior-point methods suitable for nonlinear constrained optimization with clarity.

5.1.1 *Elementary concepts*

The standard form of a (continuous) optimization problem is

$$\begin{aligned} &\text{minimize} && f_0(x) \\ &\text{subject to} && f_i(x) \leq 0,\ i \in \{1,\dots,m\} \\ &&& h_i(x) = 0,\ i \in \{1,\dots,p\} \end{aligned} \tag{5.1}$$

where $f_0(x) : \mathbb{R}^n \to \mathbb{R}$ is the *objective function* to be minimized over the variable $x \in \mathbb{R}^n$, $f_i(x) \leq 0$ are called the *inequality constraints*, and $h_i(x) = 0$ are called the *equality constraints*. The notation $f : A \to B$ denotes that f is a *function* (or *mapping*) with *domain* A, that is A is its set of acceptable inputs, and *codomain* B, that is B is its set of acceptable outputs. We say that f maps values from A to B. $\mathbb{R}$ denotes the set of real numbers, and $\mathbb{R}^n$ denotes the n-dimensional real, or Euclidean, vector space, that is the set of n-dimensional vectors with real components. The notation $x \in S$ denotes that x is an element of the set S.

We denote the domain of the optimization problem (5.1)by

$$\mathcal{D} = \bigcap_{i=1}^{m} \operatorname{dom} f_i \cap \bigcap_{i=1}^{p} \operatorname{dom} h_i \neq \emptyset \tag{5.2}$$

and assume it is nonempty. The $\cap$ operator denotes the intersection operation and the $\emptyset$ denotes the empty set, both from set theory (Halmos, 2017). The operator $\operatorname{dom} f$ denotes the domain of the function f. In effect, an optimization problem is only defined for regions where

the objective function and all constraints are defined. An optimization problem is solved once an optimal solution, usually denoted $x^\star$, is found that minimizes the objective function $f_0(x)$ such that $f_0(x^\star) \leq f_0(x)$ for all $x \in \mathcal{D}$.

Taking our geometry reconstruction problem as an example, we can describe our objective function as

$$f_0(x) = |p(x) - p_{\text{measured}}|^2 \tag{5.3}$$

where $p(x)$ is the momentum vectors produced following Coulomb explosion of a molecular with structure x, and the inequality constraints $f_i(x) \leq 0$ encapsulate the box constraints that limit the geometries recovered to physically reasonable values. For a triatomic molecule, $x = (r_{12}, r_{23}, \theta) \in \mathbb{R}^3$ may be used and $p(x) \in \mathbb{R}^9$ although it may be reduced to contain only 5 or even 3 nonzero components (see section 3.4). For a general molecule with N atoms, $x \in \mathbb{R}^{3N-6}$ and $p(x) \in \mathbb{R}^{3N-3}$ at most. We may wish to limit the reconstructed C−O bond length to lie between 100 pm to 500 pm in which case we would employ the inequality constraints $f_1(r_{\text{CO}}) = 100 - r_{\text{CO}} \leq 0$ and $f_2(r_{\text{CO}}) = r_{\text{CO}} - 500\,\text{pm} \leq 0$. Box constraints for the C−S bond length and bond angle can be placed in a similar manner. Other contraints may be employed limiting, for example, the bond length ratio $r_{\text{CO}}/r_{\text{CS}}$ to lie between certain values pertaining to physically realizable geometries. We employ no equality constraints. Generally, $p(x)$ and p_{measured} should be described in the same rotation convention otherwise they cannot be compared.

Optimization problems can be classified based on the nature of the objective function f_0 and the constraints f_i and h_j, with each class having their own algorithms. Perhaps the simplest commonly encountered class is the class of *linear programs* where the objective function and constraints are linear, that is $f_0, \ldots, f_m, h_1, \ldots, h_p$ all satisfy the linearity property

$$f_i(\alpha x + \beta y) = \alpha f_i(x) + \beta f_i(y) \tag{5.4}$$

for all $x, y \in \mathbb{R}^n$ and $\alpha, \beta \in \mathbb{R}$. Although no analytical solution exists to solve an arbitrary linear program, efficient algorithms with computational run time $\mathcal{O}(n^2 m)$ exist to find solutions, such as George Dantzig's simplex method.

Convex optimization problems are a superset of linear programs and are characterized by having an objective function and constraint functions that all satisfy the convexity property

$$f_i(\alpha x + \beta y) \leq \alpha f_i(x) + \beta f_i(y) \tag{5.5}$$

for all $x, y \in \mathbb{R}^n$ and all $\alpha, \beta \in \mathbb{R}$ with $\alpha, \beta \geq 0$ and $\alpha + \beta = 1$. Commonly encountered convex function include the exponential function e^x and the quadratic function x^2. In general, very mature and effective

algorithms exist to solve convex optimization problems. If a problem can be transformed into convex form, then it becomes rather easy to solve, however this process can be very difficult and many tricks exist. Linear programs and the linear least squares problems are special case of convex optimization problems.

Nonlinear optimization describes the class of problems where the objective or constraint functions are not linear, but not known to be convex. Unfortunately, there are no effective algorithms for solving nonlinear problems in general but there are a number of approaches that may prove fruitful. These include the interior-point method we use and sequential quadratic programming. Sun, Qu, and Wright (2015) provide an expository article on "when nonconvex problems are not scary".

Unfortunately, the problem of geometry reconstruction falls under the category of nonlinear problems. While our objective function (5.3) seems to mimic a least-squares minimization problem, it is certainly nonlinear in nature. There are two contributors to the nonlinear nature of our objective function. One is the existence of multiple solutions as evidenced by the existence of degenerate geometries we encountered in section 4.5. The other is the nonlinear nature of the objective function. A one-dimensional convex function must have a nonnegative second-derivative, and a multi-dimensional convex function must be twice-differentiable and have a positive semidefinite Hessian matrix over its domain (Boyd and Vandenberghe, 2004, p. 71). Nothing guarantees that this is true for our objective function.

Algorithms do exist for the solution of nonlinear least-squares problems such as NL2SOL (Dennis Jr, Gay, and Welsch, 1981) and the Levenberg–Marquardt algorithm (Pujol, 2007), however, modifying them to account for constraints usually introduces penalty functions and trust regions and they begin to appear quite similar to the interior-point methods described in section 5.1.5. Examples include the open-source Interior Point OPTimizer (IPOPT) (Branch, Coleman, and Li, 1999).

5.1.2 *Duality*

In order to describe and understand the interior-point method we use, it is necessary that we look at the concept of duality. Every optimization problem may be viewed from two different perspectives, that of the original, or *primal*, problem and the *dual* problem. Solving the dual problem provides a lower bound to the primal problem as we will show. Additional, by attempting to solve both problems at the same time, as interior-point methods do, an optimal solution may be found more efficiently.

We begin by defining the *Lagrangian* associated with the optimization problem (5.1) as

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x) \tag{5.6}$$

where $L : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$ and $\operatorname{dom} L = \mathcal{D} \times \mathbb{R}^m \times \mathbb{R}^p$. λ_i is the Lagrange multiplier associated with the inequality constraint $f_i(x) \leq 0$ and ν_i is the Langrange multiplier associated with the equality constraint $h_i(x) = 0$. Together, $\lambda \in \mathbb{R}^m$, and $\nu \in \mathbb{R}^p$, are called the *dual variables* or *Langrange multiplier vectors*. The basic idea is that we're accounting for the constraint functions by adjusting the objective function to include a weighted sum of the constraint functions.

The *Lagrange dual function* is defined as the minimum value of the Lagrangian L over x

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) = \inf_{x \in \mathcal{D}} \left[f_0(x) + \sum_{i=1}^{m} \lambda_i f_i(x) + \sum_{i=1}^{p} \nu_i h_i(x) \right] \tag{5.7}$$

where $g : \mathbb{R}^m \times \mathbb{R}^p \to \mathbb{R}$. The inf operator refers to the *infimum* operator, which may also be called the *greatest lower bound* operator. An important property of the dual function is that it is concave even when the problem is not convex, as it is the pointwise infimum of a family of affine functions of (λ, ν).

Theorem 5.1. *The Lagrange dual function yields a lower bound on the optimal value of the problem* (5.1) *for* $\lambda \succeq 0$ *and any* ν.

Proof. Denote the optimal value of the dual function by $p^\star$ and let x' denote a feasible input of the Lagrangian, that is it satisfies the constraints $f_i(x') \leq 0$ and $h_i(x') = 0$. Then for $\lambda \succeq 0$ and any ν we have that

$$\sum_{i=1}^{m} \lambda_i f_i(x') + \sum_{i=1}^{p} \nu_i h_i(x') \leq 0$$

so that

$$L(x', \lambda, \nu) = f_0(x') + \sum_{i=1}^{m} \lambda_i f_i(x') + \sum_{i=1}^{p} \nu_i h_i(x') \leq f_0(x')$$

and

$$g(\lambda, \nu) = \inf_{x \in \mathcal{D}} L(x, \lambda, \nu) \leq L(x', \lambda, \nu) \leq f_0(x')$$

which must hold for every feasible point x' including the optimal solution $x^\star$ and thus

$$g(\lambda, \nu) \leq p^\star = f_0(x^\star)$$

□

As the lagrange dual function provides a lower bound on the optimal value $p^\star$ that depends on (λ, ν), we may be interested in finding the best lower bound. This leads to the *Lagrange dual problem* associated with (5.1) which can be stated as

$$\begin{aligned} \text{maximize} \quad & g(\lambda, \nu) \\ \text{subject to} \quad & \lambda \succeq 0 \end{aligned} \tag{5.8}$$

and is always a convex problem as the dual function $g(\lambda, \nu)$ is always convex as mentioned when we introduced it. We can then talk about *dual feasible* pairs (λ, ν) with $\lambda \succeq 0$ and $g(\lambda, \nu) > -\infty$, *optimal Lagrange multipliers* or the *dual optimal* pair $(\lambda^\star, \nu^\star)$, and the optimal value of the dual problem, denoted $d^\star$. In some contexts involving both the dual problem (5.8) and the original problem (5.1), the original problem is called the *primal problem*.

If the optimal value of the dual problem $d^\star$ and of the primal problem $p^\star$ are equal, $d^\star = p^\star$, then we say that *strong duality* holds and the *optimal duality gap* is zero, $d^\star - p^\star = 0$. Otherwise $d^\star \leq p^\star$ and we say that *weak duality* holds.

5.1.3 *Optimality conditions*

It will be quite useful to impose conditions on what makes a feasible solution an optimal solution for both the primal and dual problems. This will provide a means of checking whether a proposed solution is optimal, as every optimal solution must satisfy the optimality conditions. Denoting the optimal primal solution by $x^\star$ and the optimal value by $p^\star = f_0(x^\star)$, we already know that it must satisfy the inequality and equality constraints,

$$f_i(x^\star) \geq 0 \quad \text{and} \quad h_i(x^\star) = 0 \tag{5.9}$$

giving us two optimality conditions so far. Denoting the dual optimal by $(\lambda^\star, \nu^\star)$ we would like for

$$\lambda_i^\star \geq 0 \tag{5.10}$$

so that the dual function provides a lower bound on $p^\star$ by theorem 5.1, giving us a third condition.

For the fourth condition, we look to the dual function and assume that strong duality holds, that is that $f_0(x^\star) = g(\lambda^\star, \nu^\star)$. Then we can write

$$\begin{aligned} f_0(x^\star) &= g(\lambda^\star, \nu^\star) \\ &= \inf_x \left[f_0(x) + \sum_{i=1}^{m} \lambda_i^\star f_i(x) + \sum_{i=1}^{p} \nu_i^\star h_i(x) \right] \\ &\leq f_0(x^\star) + \underbrace{\sum_{i=1}^{m} \lambda_i^\star f_i(x^\star)}_{\leq 0} + \underbrace{\sum_{i=1}^{p} \nu_i^\star h_i(x^\star)}_{=0} \\ &\leq f_0(x^\star) \end{aligned} \tag{5.11}$$

where we invoked the definition of the dual function on the second line, and the third line follows from the fact that the Lagrangian evaluated at the optimal primal point $x^\star$ provides a lower bound. Then on the fourth line we realize that the equality constraint functions $h_i(x) = 0$ cause the third term to vanish, and we assumed that the Lagrange multipliers obeyed $\lambda_i^\star \geq 0$ while the inequality constraints satisfy $f_i(x) \leq 0$ thus causing the second term to be nonpositive. However we require that equality hold on the fourth line as $f_0(x^\star)$ must equal itself. For equality to hold, we thus require that the second term vanish just like the third term,

$$\sum_{i=1}^{m} \lambda_i f_i(x^\star) = 0 \tag{5.12}$$

however each term is nonpositive so we can recast this condition as

$$\lambda_i f_i(x^\star) = 0, i = 1, 2, \ldots, m \tag{5.13}$$

This provides a fourth optimality condition, known as *complementary slackness*.

For the fifth condition, we realize that a function's first-derivative must vanish at a minima or maxima. Since $x^\star$ minimizes the Lagrangian $L(x, \lambda^\star, \nu^\star)$ over x, its gradient must be zero at the minima or maximum $x^\star$, giving us

$$\nabla f_0(x^\star) + \sum_{i=1}^{m} \lambda_i^\star \nabla f_i(x^\star) + \sum_{i=1}^{p} \nu_i^\star \nabla h_i(x^\star) = 0 \tag{5.14}$$

Together, we can summarize the five conditions we obtained

$$\begin{aligned} f_i(x^\star) &\geq 0, \quad i \in 1, \ldots, m \\ h_i(x^\star) &= 0, \quad i \in 1, \ldots, p \\ \lambda_i^\star &\geq 0, \quad i \in 1, \ldots, m \\ \lambda_i^\star f_i(x^\star) &= 0, \quad i \in 1, \ldots, m \\ \nabla f_0(x^\star) + \sum_{i=1}^{m} \lambda_i^\star \nabla f_i(x^\star) + \sum_{i=1}^{p} \nu_i^\star \nabla h_i(x^\star) &= 0, \quad i \in 1, \ldots, m \end{aligned} \tag{5.15}$$

which together are called the *Karush-Kuhn-Tucker (KKT) conditions*. They are sometimes referred to as the first-order optimality conditions, as second-order conditions do exist (Nocedal and Wright, 2006, §12.5).

5.1.4 *Trust regions*

In general, when searching for an optimal solution an optimization algorithm begins from an initial guess x_0. Then successive guesses, or iterates, denoted by x_k for the k^{th} guess or iterate are made with the aim of converging on the optimal solution $x^\star$.

Switching gears a little bit in this subsection, we'll look at a general strategy of solving optimization problem using the concept of a *trust region* which may be used to determine the next iterate x_{k+1}. The idea is to create and solve an approximate optimization problem at each iterate x_k with the hope that the approximation is easier to solve yet locally accurate enough to help locate the true optimal solution. The approximated is *trusted* only so much, up to some radius or region boundary. A circular or spherical trust region may be used, but so can box and elliptical regions. If a sufficiently better iterate x_{k+1} is not found within the trust region then the region may be shrunk in case the approximation becomes grossly invalid for points far way from the iterate x_k.

The approximation employed may be termed the *model function* so that the approximate problem at iterate x_k becomes

$$\underset{p}{\text{minimize}}\, m_k(x_k + p) \tag{5.16}$$

where p is the candidate step so that $x_k + p$ lies within the trust region. A very popular model function takes the form of a quadratic approximation using the first two terms of a Taylor approximation of the objective function at the iterate point

$$m_k(x_k + p) = f(x_k) + p^T \nabla f(x_k) + \frac{1}{2} p^T H[f(x_k)] p \tag{5.17}$$

where $\nabla f(x_k)$ and $H[f(x_k)]$ are the gradient and Hessian respectively, of the objective function f at the point x_k (Moré and Sorensen, 1983). Recall that the Hessian is a square matrix of second-order partial derivatives with entries $H[f(x)]_{i,j} = \partial^2 f / \partial x_i \partial x_j$. A quadratic function is convex and thus such a convex "subproblem" that locally approximates the optimization problem can be solved efficiently.

Trust regions see a great deal of use in nonlinear optimization methods, and can be modified for constrained optimization. Beyond the choice of approximation and trust region type, choosing the region size and shape, the step size, and the method used to solve even the trust region subproblem are important (Nocedal and Wright, 2006, ch. 4). Another class of methods serving a similar purpose are line search methods where a direction is first chosen to search for the next iterate, so that the step size is chosen second. Line search methods are in a sense the dual of trust region methods, where the step size (trust region radius or boundary) is chosen first, then a direction is chosen (Nocedal and Wright, 2006, ch. 1).

5.1.5 *Primal-dual interior point methods*

Having discussed some important ideas and concepts from mathematical optimization theory, we can now begin to discuss the interior-point optimization method we rely on for geometry reconstruction in this chapter.

Introduction of a logarithmic barrier function

The basic idea behind interior-point methods is to modify the original optimization problem to take into account the constraint functions by modifying the objective function to penalize guesses that leave the feasible region, that is guesses that break the inequality and equality constraints. As this may modify the optimal solution, the approximation is parameterized by a real parameter $\mu > 0$ that is relaxed as a local minimum is approached, so you are essentially solving a series of approximate optimization "subproblems" that converges to the original problem as $\mu \to 0$. A very popular approximation is to use a logarithmic barrier that induces a penalty that approaches ∞ as you approach any barrier defined by a constraint, that is the penalty increases exponentially as a constraint is close to being violated. This approach is derived and described in detail by Byrd, Gilbert, and Nocedal (2000), Byrd, Hribar, and Nocedal (1999), and Waltz et al. (2006) as well as by Nocedal and Wright (2006, ch. 19).

To begin we will rewrite the original optimization problem (5.1) as

$$\begin{aligned} \text{minimize} \quad & f_0(x) \\ \text{subject to} \quad & f_i(x) + s_i = 0,\ i \in \{1, \ldots, m\} \\ & h_i(x) = 0,\ i \in \{1, \ldots, p\} \end{aligned} \tag{5.18}$$

where we have transformed the inequalities $f_i(x) \leq 0$ into equalities by the introduction of so-called *slack variables* $s_i \geq 0$. If the slack variable s_i associated with the constraint $f_i(x) \leq 0$ is positive ($s_i > 0$) for a given solution, then the constraint is being obeyed and does not restrict the possible steps the algorithm may take. Points with $s_i < 0$ represent unfeasible solutions that do not obey the constraint $f_i(x) \leq 0$. If $s_i = 0$ the constraint $f_i(x) \leq 0$ is said to be *binding* in that the possible steps to be taken are restricted but the constraint is still obeyed. The logarithmic barrier will penalize solutions exponentially as any of the slack variables s_i approach zero.

Now that we have introduced slack variables that allow us to penalize solutions that are very close to breaking a constraint, we can now modify the objective function by appending to it the logarithmic barrier penalty function,

$$\begin{aligned} \underset{x,s}{\text{minimize}} \quad & f_0(x) - \mu \sum_{i=1}^{m} \ln s_i \\ \text{subject to} \quad & f_i(x) + s_i = 0,\ i \in \{1, \ldots, m\} \\ & h_i(x) = 0,\ i \in \{1, \ldots, p\} \end{aligned} \tag{5.19}$$

where $\mu > 0$ is a real positive parameter and ln denotes the natural logarithm function. The interior-point approach involves solving the barrier subproblem (5.19) for a sequence of positive barrier parameters μ_k that converges to zero, thus reproducing and hopefully solving the original optimization problem while accounting for the constraints. Thus interior-point methods get their name from the fact that iterates are prevented from leaving the region of feasible solutions, however most modern interior-point methods can start from any initial guess x_0.

At each iteration, if the problem is locally convex then a trust region subproblem is created and solved to take the next step. Otherwise, a direct step is taken that attempts to converge on an optimal solution by solving the KKT optimality conditions (5.15). Thus such methods are sometimes referred to as *primal-dual interior-point methods* as attempting to solve the KKT equations involves attempting to solve both the primal and dual problem simultaneously.

Direct step by solving the KKT optimality conditions

Applying Newton's method to the KKT optimality conditions (5.15) for the barrier subproblem (5.19) yields a system of equations for the step size and direction to be taken, Δx and Δs, which may be written in matrix form as (Nocedal and Wright, 2006, p. 566)

$$\begin{pmatrix} H & 0 & -J_h^T & -J_f^T \\ 0 & \Lambda & 0 & S \\ J_h & 0 & 0 & 0 \\ J_f & -I & 0 & 0 \end{pmatrix} \begin{pmatrix} \Delta x \\ \Delta s \\ \Delta \nu \\ \Delta \lambda \end{pmatrix} = - \begin{pmatrix} \nabla f_0(x) - J_f^T \lambda - J_h^T \nu \\ S\Lambda - \mu e \\ h \\ f + s \end{pmatrix} \tag{5.20}$$

where $f = (f_1, f_2, \ldots, f_m)$, $h = (h_1, h_2, \ldots, h_p)$, and $s = (s_1, s_2, \ldots, s_m)$ are vectors containing the inequality constraint functions, equality constraint functions, and slack variables, respectively. $\lambda \in \mathbb{R}^m$ and $\nu \in \mathbb{R}^p$ are the Lagrange multipliers associated with the constraint functions f_i and h_i respectively. H is the Hessian of the Lagrangian of the objective function f_0,

$$H = H[L(x, \lambda, \nu)] = H[f(x)] + \sum_{i=1}^{m} \lambda_i H[f_i(x)] + \sum_{i=1}^{p} \nu_i H[h_i(x)], \tag{5.21}$$

and J_f and J_h are the Jacobian matrices of the constraint functions $f_i(x) \leq 0$ and $h_i(x) = 0$ respectively. Recall that the Jacobian matrix is a square matrix containing the first-order partial derivatives of a vector-valued function with entries $J[f(x)]_{i,j} = \partial f_i / \partial x_j$. In this case the i^{th} row of J_f is the partial derivatives of the constraint function $f_i(x)$, $(\partial f_i / \partial x_1, \ldots, \partial f_i / \partial x_n)$. $S = \text{diag}(s)$ and $\Lambda = \text{diag}(\lambda)$ are diagonal matrices with diagonal entries given by the vectors s and λ. That is, $S = \text{diag}(s) = \text{diag}(s_1, s_2, \ldots, s_m)$ is a matrix with diagonal entries $s_1, s_2, \ldots, s_m$, for example. e is a vector of ones with the same size as h.

In order to solve for the steps Δx and Δs, MATLAB factorizes the matrix into its LDL (or Cholesky) decomposition which constitutes the most computationally expensive step of using an interior-point method. This step also allows us to check whether we are at an optimal solution given by a local minima or maxima as the Hessian is positive-definite at an optimal point (also called the second derivative test).

Trust region step

If the Hessian is not positive-definite (that is, we have not arrived at an optimal point), the algorithm may attempt a trust region step

within a trust region of some radius R if the objective function is locally convex. The trust region subproblem is given by

$$\min_{\Delta x, \Delta s} \nabla f_0(x_k)^T \Delta x + \frac{1}{2}\Delta x^T H[L]\Delta x + \mu e^T S^{-1}\Delta s + \frac{1}{2}\Delta s^T S^{-1}\Lambda\Delta s \quad (5.22)$$

subject to the approximate constraints $f + J_f\Delta x + \Delta s = 0$ and $h + J_h\Delta x = 0$, where all the variables are as defined for the Newton step described above. The steps Δx and Δs that minimize the quadratic function are taken by the algorithm. $H[L]$ denotes the Hessian of the Lagrangian of the objective function f_0. It is worth mentioning that the calculation of the Hessian is a computationally intensive task and thus it is approximated by one of a number of algorithms.

5.1.6 *Curse of dimensionality and possible solutions*

The curse of dimensionality, a term first introduced by Bellman (1957) when considering problems in dynamic optimization, refers to the exponential increase in volume when adding extra dimensions to Euclidean space (Keogh and Mueen, 2010). It manifests itself in two ways when tackling the geometry reconstruction problem for larger and larger molecules, as we need $3N - 6$ parameters to describe the geometry of an molecule with $N \geq 3$ atoms. Firstly, the parameter space or phase space to be searched increases exponentially with N, and with this increase may come an increase in local minima, and possible an increase in the number of degenerate geometries. While interior-point methods may still be feasible for polyatomic molecules with several atoms, convergence will definitely take longer and multiple runs may be required before finding a feasible geometry or any degenerate geometries, possibly necessitating the use of a supercomputer cluster.

The second manifestation, which seems more severe from preliminary investigations of reconstructing acetylene (C_2H_2) molecular geometries, is the proliferation of saddle points in high-dimensional spaces, termed the *saddle-point problem* as argued by Pascanu et al. (2014) using evidence from statistical physics, random matrix theory, and neural network theory. Fortunately, this is a very active area of research due to the recent surge and revival of interest in artificial intelligence (Bengio, 2016; LeCun, Bengio, and Hinton, 2015) and the development of new algorithms may be helpful in reconstructing larger molecules. One recent example worth looking into for future improvements include the saddle-free Newton method proposed by Dauphin et al. (2014) which uses second-curvature information to rapidly escape from high-dimensional saddle points.

One easy method of tackling this problem when attempting to reconstruct larger molecules is to fix certain parameters of the molecule's

geometry, ones which may exhibit very low variability. An example may be the triple C≡C bond in acetylene.

5.2 IMPLEMENTATION

As mentioned in the previous section, there are many complications involved with implementing advanced optimization algorithms such as the interior-point method we wish to use, and so we turn to the readily-available and mature implementation in MATLAB's Optimization Toolbox in conjunction with the Global Optimization and Parallel Processing Toolboxes. In the spirit of open science and reproducibility, we strongly feel that we should have chosen an open-source implementation however this was not a consideration at the beginning of this project and the MATLAB implementation seems to be superior to most of the available alternatives we inspected, proprietary and open-source, and is well-documented and easy to use as opposed to specialized mathematical optimization software developed for research purposes. So we see the use of MATLAB as a necessary evil at this point in time to test out the effectiveness of interior-point optimization methods for geometry reconstruction.

The concern behind relying on proprietary software is mainly to do with scientific reproducibility in computational studies (Easterbrook, 2014) for which Millman and Pérez (2014) and Wilson et al. (2014) provide excellent advice. MATLAB is popular enough and a standard piece of software in many fields that one may quite easily find a usable instance in an academic setting to run the code in appendix B and replicate the results presented here. However, in order to reproduce our results from scratch and verify their correctness, the optimization code would need to be inspected, which is impossible in this case due to the proprietary nature of MATLAB. This was not as big a concern with the lookup table as it did not rely on any MATLAB-specific library functions that do not have direct analogues in other programming environments.

For the implementation, the MATLAB Optimization Toolbox provides a general-purpose nonlinear programming solver through the `fmincon` function that attempts to find the minimum of a constrained nonlinear multivariable objective function. Among the algorithms `fmincon` can employ is the interior-point method we described in the previous section. We use it to find geometries whose post-explosion momentum vectors most precisely match the measurements. We constrain the solution using *box constraints* to ensure we recover triatomic geometries with the constraints that $100\,\mathrm{pm} \leq r_{CO}, r_{CS} \leq 500\,\mathrm{pm}$ and $140° \leq \theta \leq 180°$ for the OCS molecule.

We use the objective function

$$\log_{10} |p(x) - p_{\text{measured}}|^2 \tag{5.23}$$

where $p(x)$ is the momentum vectors produced following Coulomb explosion of a molecule with structure $x = (r_{12}, r_{23}, \theta)$. The optimization routine seemed to perform slightly better after taking the base-10 logarithm so that the absolute error is quantified on the order of -10^{-2} as opposed to 10^{-50} as most optimization problems tend to have a default error tolerance of 10^{-6} below which a solution is taken to be optimal. However, introducing the logarithm also introduces singularities in the objective function since it diverges to $-\infty$ as the optimal solution is approached. A better choice may have been a scaled ℓ_2-norm such as $10^{50}|p(x) - p_{\text{measured}}|^2$ so that absolute errors below 1 correspond to good geometries and so the objective functions tends to 0 as the optimal solution is approached. For the purposes of this thesis, the choice of objective function did not seem to affect the performance of the optimization routine as long as the error tolerance is adjusted accordingly, however, it may for the reconstruction of more complicated molecular structures.

While developing the lookup table we described the bond lengths in units of angstroms (Å) but we will now describe them in picotmeters (10^{-12} m), while the bond angles will still be described in units of degrees to keep all the molecular parameters within the same order of magnitude (as opposed to 12 orders of magnitude apart if we used meters and degrees). This is especially important for optimization algorithms which compute step sizes and trust region radii using the values of the Hessian and Jacobian which may take on extreme values when the derivatives of the objective function are very small or very large, as may be the case when the parameters vary in magnitude by 10^{12}. A more visual description would be to imagine finding an optimal point within a region described by a cuboid in phase space when using picometers and degrees, and finding an optimal point within an almost infinitesimally thin sheet in phase space when using meters and degrees. As expected, this turns out to be important for `fmincon` as it performed much better, converging on the correct solution more often and in fewer iterations when the parameters were all numerically within the same order of magnitude.

To ensure that we have found geometries corresponding to global minima and not just local minima, and to find degenerate geometries, we run `fmincon` multiple times for each set of measured momentum vectors, each time using a different initial starting point. This is done using the `MultiStart` class from the Global Optimization Toolbox which runs multiple instances of `fmincon` in parallel (requiring the use of the Parallel Processing Toolbox) using a uniformly distributed set of starting points in an attempt to find multiple solutions. Typically, only a single run is required to find a solution, especially when using simulated data, but at least several runs may be needed before finding a second degenerate geometry.

As we may have many measurements to reconstruct, we would like to make use of all available processor cores when running on a personal computer and we especially want to make full use of each core when running on a supercomputer cluster, so the measurements are iterated over using a *parallel for loop* or a `parfor` loop which executes each loop iteration on a different core. When the number of cores exceeds the number of different starting points used by `MultiStart`, this will ensure that the other cores are reconstructing other geometries thus keeping CPU utilization at 100%.

5.3 RECONSTRUCTIONS OF EXPERIMENTAL DATA

Now that we have a more sophisticated method for reconstructing geometries, we should attempt to reconstruct the same geometries we saw in section 4.5 for the OCS molecule following Coulomb explosion by a 7 fs laser pulse for the $(2, 2, 2)$ fragmentation channel, the results of which are shown in figure 25. The modal geometry is calculated to be $r_{CO} = 170\,\text{pm}$, $r_{CS} = 159\,\text{pm}$, $\theta_{OCS} = 173°$ while the average geometry of $r_{CO} = 193\,\text{pm}$, $r_{CS} = 168\,\text{pm}$, $\theta_{OCS} = 171°$ is slightly larger due to outliers.

Let us look at the process of analyzing and plotting the geometries in detail that is involved in the production of figure 25 as we now possess more information about the recovered geometries (e. g. whether the solution is optimal and whether degenerate geometries exist) than the lookup table provided, allowing for some pre-processing and filtering.

Before plotting, badly reconstructed geometries and duplicate geometries are filtered out. A badly reconstructed geometry satisfy at least one of three criteria. It (1) does not satisfy the KKT optimality conditions from section 5.1.3 in which case the optimization algorithm is said to not have converged and `MultiStart` assigns such a geometry with an exit flag of 1 making it easy to filter out such geometries. Or (2) it produces momentum vectors with a high absolute error when compared to the measured momentum vectors. We choose a threshold of 10^{-50} above which we say that the geometry is not precise enough to be a good reconstruction. The vast majority of reconstructions have significantly lower error (10^{-59} to 10^{-54}) and in general, geometries with a high absolute error do not satisfy (1) either. Or finally, (3) it lies very close to the box constraints (within 0.001) that form our cuboid of physically realistic geometries in phase space. This usually indicates that the optimal geometry lies outside the constraints and that the solver asymptotically approached the boundary (due to the logarithmic barrier) in an attempt to converge on an optimal solution. Such bad geometries tend to also satisfy (1) and (2), but some redundancy is desirable to find all badly reconstructed geometries.

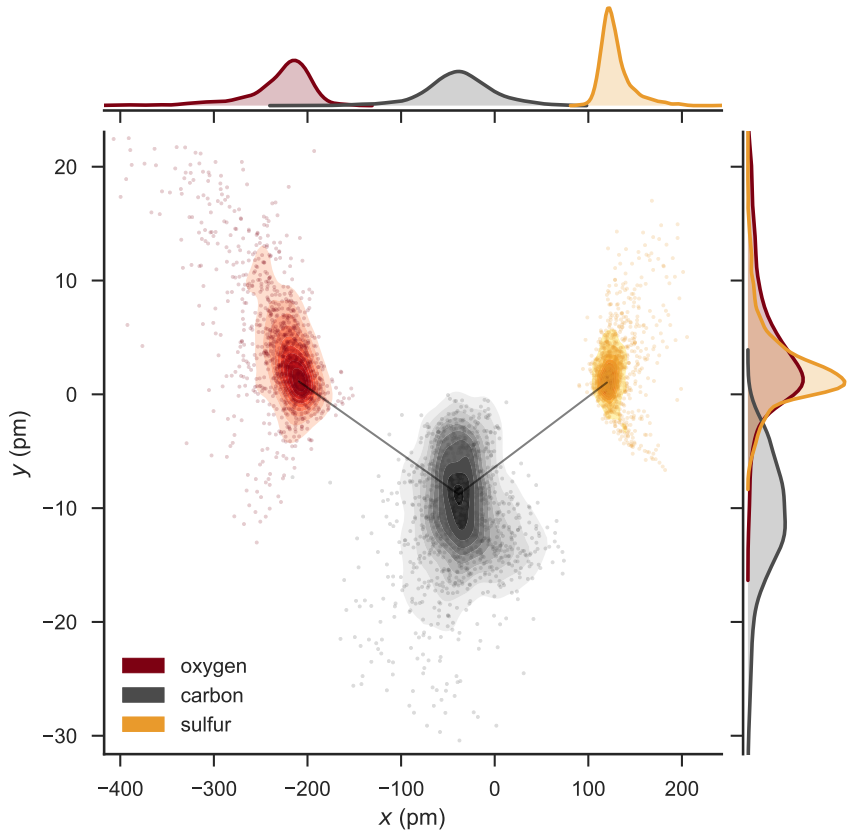

Figure 25: Scatter plot showing a reconstruction of the molecular geometry of OCS following Coulomb explosion by a 7 fs laser pulse for the $(2, 2, 2)$ fragmentation channel. Each geometry is represented by three colored points, one for each atomic fragment; red for oxygen on the left, black for carbon in the center, and yellow for sulfur on the right. The colors were chosen to imitate the CPK coloring convention. Geometries are plotted such that the molecule's center of mass is at the origin to showcase the variance in each atomic fragment's position, and are rotated such that a vertical line bisects the O−C−S bond angle. Bivariate kernel density estimates (KDE) with a Gaussian kernel, plotted as shaded-in contours, are used to estimate the probability density of each atomic fragment's position (see section 3.2.3 for a discussion of KDE's). Solid black lines are drawn between the peaks of each atomic fragment's kernel density estimate to illustrate the *modal geometry* or most likely geometry. Along the top of the plot, univariate KDE's show the probability density of each atomic fragment's position along the x-axis, and the same is done for the y-axis along the right. The molecule is almost straight but an aspect ratio of approximately $10 : 1$ is employed to showcase variability in the y-axis.

For our reconstruction, we configured `MultiStart` to use 50 uniformly distributed starting points, which in hindsight was highly excessive. Out of 1,285 sets of measured momentum vectors, we attempted to reconstruct each 50 times, recovering 53,648 geometries in total. 50,615 represented duplicate geometries. Due to rounding errors involved with the comparison of floating-point numbers, we defined two triatomic geometries i and j to be duplicates if $\Delta_{ij} < 0.1$ where $\Delta_{ij} = |r^i_{12} - r^j_{12}| + |r^i_{23} - r^j_{23}| + |\theta^i - \theta^j|$ with bond lengths and angles being numerically described in picometers and degrees. After filtering out duplicate geometries, 3,033 unique reconstructed geometries remain. 1,816 had an exit flag of 1 indicating a bad geometry (a non-optimal solution). No geometries with high error ($> 10^{-50}$) were found, having all been previously found with an exit flag of 1. Then a further 91 geometries were found very close to the box constraints. Filtering out these $1,816 + 91 = 1,907$ bad geometries, we are left with 1,126 "good" geometry reconstructions. 1,072 were mapped to a single geometry, 18 measurements to two distinct degenerate geometries, and 6 to three distinct degenerate geometries. No measurement mapped to 4 or more degenerate geometries, and 189 measurements could not be reconstructed to satisfy our box constraints. In total, 2% of measurements mapped to multiple degenerate geometries and 1,096 measurements were successfully reconstructed, giving an 85% success rate.

The recovered geometries are plotted in figure 25 and the geometries recovered for other laser pulse lengths (30 fs, 60 fs, and 100 fs) are plotted in appendix A (figures 43–45). The 200 fs data was not analyzed as the molecule seemed to have stretched too much for the lookup table reconstruction to seem trustworthy. Reconstruction statistics including success rate and number of degenerate geometries found for each pulse length is tabulated in table 4. The modal and average geometries of for each pulse length are tabulated in table 5.

5.3.1 *Comparision with the lookup table*

Comparing figures 22 and 25 we do not see much qualitative difference between the two. Figure 25 does have a greater number of reconstructions and data points. This is due to both the higher success rate (85% versus 73%) and since 500 more measurements were found since the lookup table was tested. Degenerate geometries contribute a small amount as well. The atomic positions exhibit greater variability in their positions, which may be because the optimization routine is able to reconstruct more extreme geometries than the lookup table could and with greater precision. The modal and average geometries calculated from both reconstructions are extremely similar.

While the overall geometry is not very different, the absolute errors on each geometry are much smaller. The lookup table produces ge-

Pulse length (fs)	Geometries	Reconstructions	Degenerate
7	1285	1096 (85%)	18+6 (2.2%)
30	1501	1164 (76%)	62+30 (7.9%)
60	358	249 (70%)	15+6 (8.4%)
100	1056	694 (66%)	39+13 (7.5%)

Table 4: Statistics for geometry reconstruction using constrained nonlinear optimization. The geometries column lists the number of experimental measurements (sets of momentum vectors) obtained, the reconstructions column lists the number and percentage of these measurements that were successfully reconstructed, and the degenerate column lists the number of measurements for which 2 and 3 degenerate geometries were found, respectively, and the percentage of successful reconstructions that yielded degenerate geometries. No measurement ever yielded more than 3 degenerate geometries.

Pulse	Modal geometry			Average geometry		
length (fs)	r_{CO}	r_{CS}	θ_{OCS}	r_{CO}	r_{CS}	θ_{OCS}
(Equilibrium)	116	156	172	116	156	172
7	170	159	173	193	168	172
30	231	177	172	243	215	168
60	238	196	172	242	232	166
100	249	206	171	256	254	165

Table 5: Average and modal geometries calculated from the geometries reconstructed using constrained nonlinear optimization as a function of pulse length. The bond lengths are given in picometers (10^{-12} m) and the bond angles in degrees.

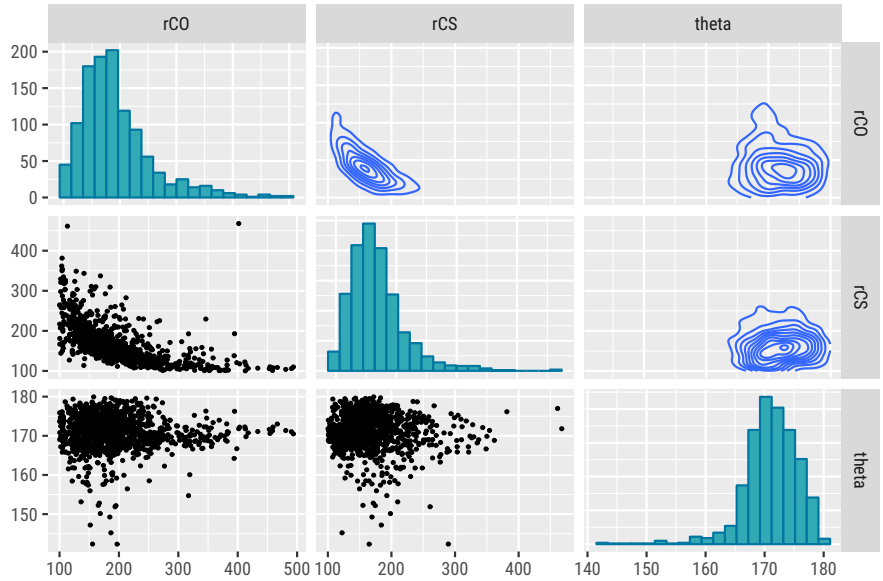

Figure 26: Scatter plot matrix showing the bivariate relationship between the parameters (r_{CO}, r_{CS}, θ) for the reconstructions of the molecular geometry of OCS following Coulomb explosion by a 7 fs laser pulse for the $(2, 2, 2)$ fragmentation channel. On the diagonal, histograms show the distribution of bond lengths and bond angle for the reconstructed geometries (tick marks at the bottom of each column). Below the diagonal, scatter plots show the bivariate relationship between each molecular parameter. Above the diagonal, the same relationship is given using a contour plot instead.

ometries with absolute errors on the order of 10^{-48} at best due to its low resolution but the optimization routine finds geometries with errors on the order of 10^{-59} to 10^{-54}, corresponding to an additional 3-5 decimal places of precision on the numerical values of the molecular parameters.

The optimization routine is also able to precisely find degenerate geometries by using multiple starting points, a task which would have been non-trivial for the lookup table.

5.3.2 *Investigating bond length correlations*

Figure 25 provides an intuitive image of what a molecular geometry looks like, but it would be interesting to also see a distribution of bond lengths and bond angles to quantify the variability in each of these parameters. Correlations between the bond lengths and bond angles may also be looked at to study, for example, if longer C−O bond lengths correspond to longer C−S bond lengths, or if more bent geometries tend to have longer bond lengths. To visualize this relationship in figure 26 we utilize a scatterplot matrix.

The bond length distributions on the diagonal of figure 12 are somewhat expected; they peak slightly above the equilibrium values indicating some molecular rearrangement and bond lengthening due to

the molecule's interaction with the laser field. The bond angle distribution shows less variability suggesting that the molecule may have not had much time to bend yet, or that the molecule stretches due to ion motion during the ionization process.

Comparing the C−O bond length between equilibrium and after Coulomb explosion by a 7 fs laser pulse we see a stretch of about 45% which is larger than the stretch exhibited by CO_2 Bocharova et al., 2011 after Coulomb explosion by a 7 fs laser pulse. However, in the case of OCS, the C−S bond remains unchanged. Comparing the bond angle distribution from figure 26 with theoretical simulations of the bond angle distribution for OCS $(2, 2, 2)$ (Wales et al., 2012b), we see they are quite similar hinting that the bond angle, at least for 7 fs remains very close to equilibrium and that we are reconstructing geometries with a reasonably accurate bond angle distribution.

What is very interesting, however, is the relationship between the two bond lengths, r_{CO} and r_{CS}. We may intuitively expect both bond lengths to lengthen as the three atoms should repel each other in the $(2, 2, 2)$ charge state, or possibly to have one bond stretch while the other remains unchanged. However, the reconstructions actually indicate the complete opposite—that while one bond stretches the other shrinks, almost following a reciprocal relationship. This effect becomes even more pronounced for reconstructions of molecular geometries exposed to longer pulse lengths (figures 46 – 48).

Unfortunately, to our knowledge no other studies performing geometry reconstruction using Coulomb explosion imaging have reported the correlations between their bond lengths and bond angles, so no comparisons or references can be made. When plotted, as in figure 25, the geometries seem to be physically reasonable and so do the bond length and bond angle distributions on the diagonal of figure 26, it is only the correlations that do not.

At this juncture, such worrying results force us to distrust the geometry reconstructions we have produced thus far. While the average geometries and overall dataset seems to be physically reasonable, the individual geometries seem to not make physical sense. Additionally, the same unusual relationship emerges out of at least two different reconstruction methods. For now we will move on to utilize the optimization routine to further investigate the nature of the degenerate geometries we are recovering, however we will come back and resolve this issue in the next chapter. Analyzing the reconstructed geometries produced by the lookup table results in a very similar relationship between the bond lengths of reconstructed geometries.

5.4 INVESTIGATING DEGENERATE GEOMETRIES

The optimization approach should also be tested to study whether it can accurately reconstruct simulated geometries in the same man-

ner we tested the Nelder-Mead simplex method (section 4.1.2) and the lookup table (section 4.3.3). We will also use this information to further investigate the nature of the degenerate geometries we are recovering.

The accuracy testing of the Nelder-Mead simplex method and the lookup table was relatively cursory, and could be made more thorough. We varied one molecular parameter at a time, which was however, sufficient to show the inadequacy of the simplex method. For a more thorough test, we will substantially vary all the molecular parameters by simulating the Coulomb explosion of geometries within a box in phase space described by $100\,\text{pm} \leq r_{CO}, r_{CS} \leq 500\,\text{pm}$ and $140° \leq \theta \leq 180°$. For a low-resolution test, we pick geometries $r_{CO} \times r_{CS} \times \theta_{OCS}$ where r_{CO} and r_{CS} are sets containing 10 uniformly spaced bond lengths between 100 pm and 500 pm, θ_{OCS} is a set containing 10 uniformly spaced bond angles between 140 deg and 180 deg, and $X \times Y = \{(x, y) | x \in X \text{ and } y \in Y\}$ denotes the cartesian product as expressed in set-builder notation (Warner, 1990, p. 6). Thus we are reconstructing $10^3 = 1,000$ geometries from simulated Coulomb explosions. We also perform a higher-resolution test using $20^3 = 8,000$ uniformly distributed geometries within the same box.

In both tests, we find that the optimization routine can reconstruct the simulated geometries in 98.5% of cases using only a single starting guess. Each reconstruction takes approximately one second. In each of these reconstructions, the parameters of the recovered geometry numerically matched the original geometry up to several decimal places. The absolute error between the recovered and simulated momentum vectors was below 10^{-55} for 95% of reconstructions, with the mean error being approximately 10^{-57}, is 10 orders of magnitude lower than the absolute errors on momentum vectors retrieved using the lookup table. Since we use the square of the ℓ_2-norm to quantify the absolute error, this actually represents an improvement of approximately 5 orders of magnitude in accuracy over the lookup table. Using multiple starting points recovers geometries for the other 1.2% of cases.

While the optimization routine recovered very precise geometries whose post-explosion momentum vectors very closely match the expected vectors, the geometries were not always the originally generated geometries which we expected. It seems that in about 5% of cases, the routine returned a degenerate geometry. It is important to make the distinction that when the Nelder-Mead simplex method and the lookup table returned a different geometry than expected, it was a failure in recovering the exact geometry as the momentum vectors did not match closely resulting in a large absolute error (roughly $> 10^{-46}$) (except for a small number of cases where it had found a degenerate geometry). However, in this case we are finding very precise reconstructions and each different geometry represents a degenerate

geometry. To showcase the nature of these degenerate geometry, we plot arrows between the original (expected) geometry and the recovered degenerate geometry in figure 27, which we will refer to as a *degeneracy map*. If the recovered geometry was the original geometry as expected, then no arrow is plotted.

We see some patterns in the degenerate geometries recovered. Degenerate geometries exist only for molecules with bond angles between approximately 150 deg and 170 deg which includes a sizable minority of the molecules reconstructed as indicated by the bond angle distribution in figure 26. They also exist mainly for molecules exhibiting significant bond asymmetry and tend to be degenerate with another molecule that is more bent and has a smaller degree of bond length asymmetry. They also appear in two regions, one where the C−O bond length is longer, and another where the C−S bond length is longer. The discrete number of arrows figure 27(a) may suggest the existence of a discrete number of these degenerate geometries, however, repeating the test with a greater number of geometries produces very similar results except for a correspondingly higher density of lines, suggesting that regions exist in phase space where every geometry is degenerate with another, i. e. that an uncountably infinite[1] number of degenerate geometries exist.

In these simulations we knew *a priori* which geometry we expected to reconstruct, so we can assign a direction to each arrow. However, when reconstructing experimental data, we have no prior knowledge of what geometry we expected to recover, so the arrow could point in either direction. In some cases it might be possible to choose one degenerate geometry over the other(s) if one is physically unrealistic, e. g. if it is very highly bent or exhibits an extreme bond length asymmetry.

To further visualize the set of geometries we recovered, we utilize a scatterplot matrix in figure 28 showing the bond length and angle distributions for the 8,000 geometries used for accuracy testing as well as their bivariate relationships. It is interesting to note that whenever degenerate geometries exist, the more bent geometry is found. Perhaps these more bent solutions posess a larger *basin of attraction*, which may or may not be strongly dependent on the optimization algorithm employed. It might be interesting to visualize them, possibly taking a similar approach as Asenjo et al. (2013) who actually employ mathematical optimization methods to determine energy minimizing molecular structures. They observe basins with rather complicated boundaries, reminiscent of the beautiful and spatially chaotic patterns produced by the magnetic pendulum.

For the feasibility of geometry reconstruction, however, this result raises additional issues past the unusual bond length correlations

1 A countably finite set may refer to the set of integers $\mathbb{Z} = \{0, \pm 1, \pm 2, \dots\}$, for example, which may be enumerated. An uncountably infinite set has the same cardinality as the set of real numbers $\mathbb{R}$ which cannot be enumerated (Halmos, 2017).

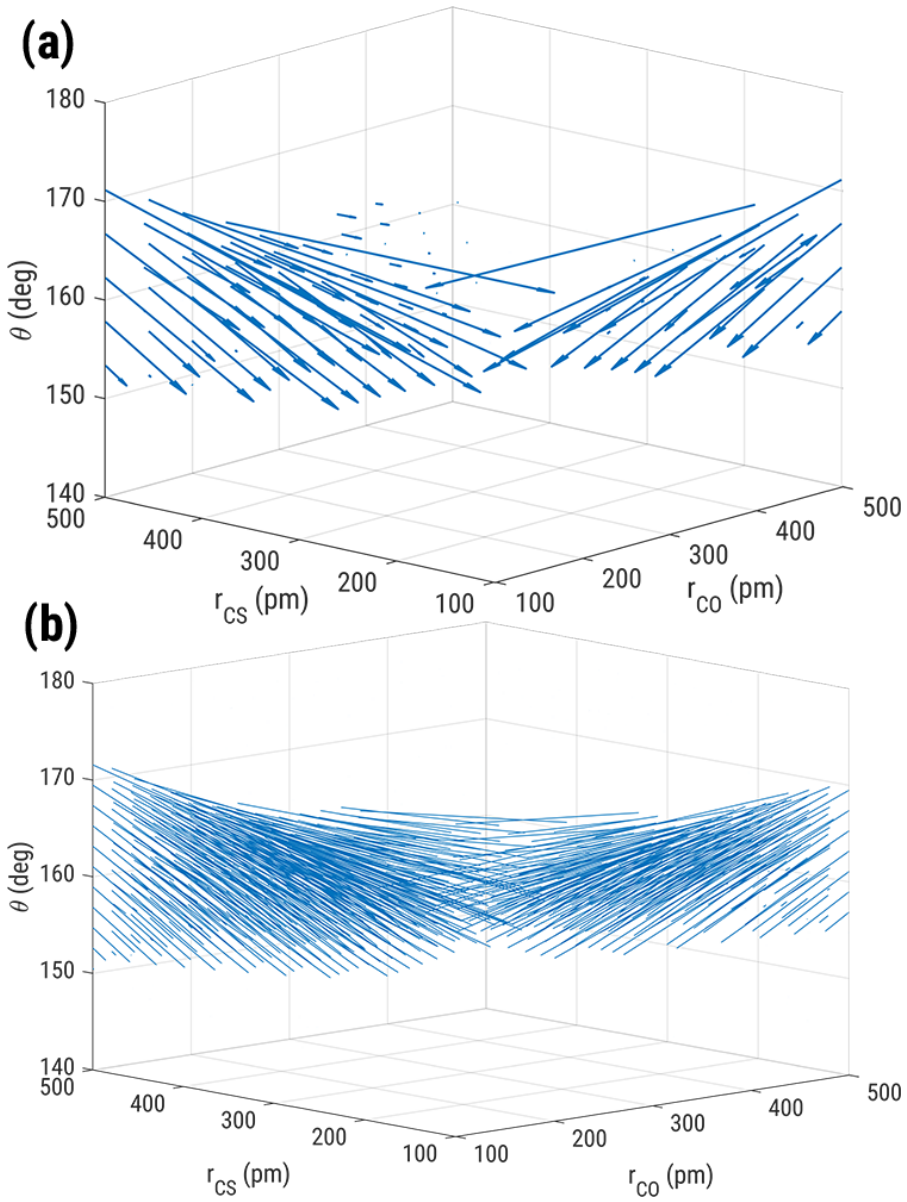

Figure 27: Mapping between degenerate geometries for the OCS $(2,2,2)$ molecule. The Coulomb explosion of (a) $1,000$ (10^3) and (b) $8,000$ (20^3) uniformly distributed geometries within a box in phase space described by $100\,\text{pm} \leq r_{CO}, r_{CS} \leq 500\,\text{pm}$ and $140° \leq \theta \leq 180°$ was simulated and then reconstructed using the momentum vectors of the atomic fragments that resulted from the simulations. If the reconstructed geometry matched the original geometry exactly, then no arrow is plotted. However, an (a) arrow or (b) line is plotted from the original expected geometry to the recovered geometry if it represented a degenerate geometry. Some arrows and lines appear small enough that they resemble points, they are actually slight numerical difference between the expected geometry and the reconstructed geometry, and do not necessarily represent a degenerate geometry.

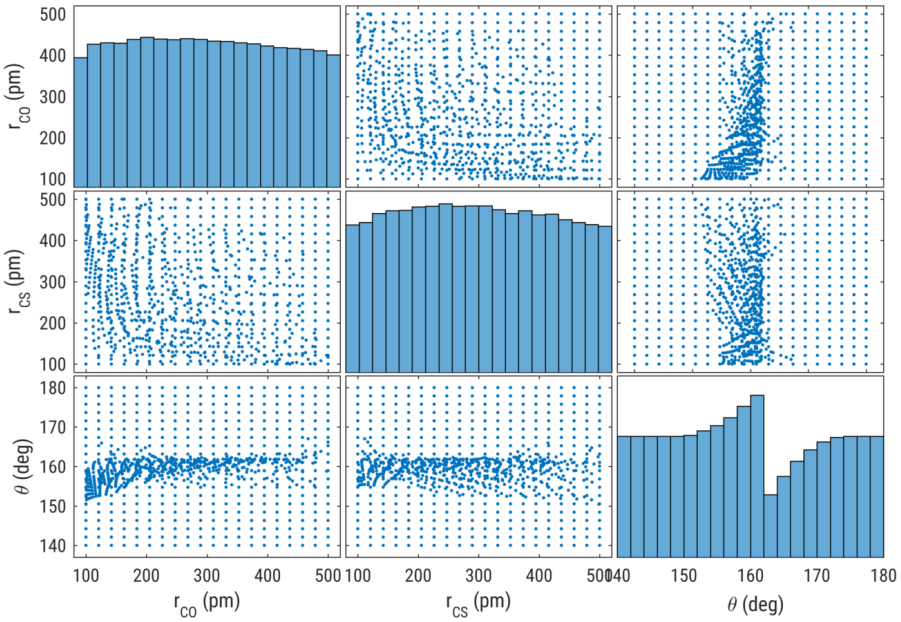

Figure 28: Scatter plot matrix showing the bivariate relationship between the parameters (r_{CO}, r_{CS}, θ) for the reconstructions of the molecular geometry of OCS from momentum vectors obtained from simulated Coulomb explosions of $8,000$ (20^3) uniformly distributed geometries within a cuboid in phase space described by $100\,\text{pm} \leq r_{CO}, r_{CS} \leq 500\,\text{pm}$ and $140° \leq \theta \leq 180°$. On the diagonal, histograms show the distribution of bond lengths and bond angle for the reconstructed geometries with tick marks at the bottom of each column (no vertical scale is given for the histograms). Below the diagonal, scatter plots show the bivariate relationship between each molecular parameter. Above the diagonal, the same scatter plots are shown with the x and y-axes interchanged to correspond with the axes labels and tick marks.

found in the previous section. Even when the measured momentum vectors can be mapped to a molecular geometry very precisely, the existence of multiple solutions corresponding to very different geometries may make it impossible to perform accurate geometry reconstructions, especially when multiple degenerate geometries represent physically realizable geometries. Table 2 also suggests the existence of triply degenerate geometries, which may further complicate the task of geometry reconstruction.

Interestingly, we find that we recover degenerate geometries approximately 5% of the time while Kunitski et al. (2015, supplementary information) report finding degenerate geometries approximately 10% of the time, although they chose to disregard them for their analyses.

We considered whether these degeneracies could arise due to our choice of momentum convention in section 3.4 however our convention simply rotates the three momentum vectors such that the carbon's momentum vector lies along the $+x$-axis and all three momentum vectors lie in a plane. The length of the momentum vectors and the relative angles between them remain unchanged, and so two different geometries producing different momentum vectors cannot result in the same set of vectors after rotation into our convention, unless they represent degenerate geometries.

5.5 CONCLUSIONS

Geometries may be reconstructed quickly and precisely using constrained nonlinear optimization—This approach represents a significant improvement over the geometry reconstructions provided by the lookup table. Reconstructing a single geometry can be done in approximately one second, which is faster than the lookup table if enhanced precision is desired. Furthermore, the recovered molecular parameters are orders of magnitude more precise than those provided by the lookup table, and degenerate geometries may be found precisely.

6

UNCERTAINTY QUANTIFICATION IN GEOMETRY RECONSTRUCTION

In the previous two chapters, we approached the task of geometry reconstruction using two approaches, a lookup table which was only feasible for the reconstruction of triatomic molecules, and the more sophisticated optimization approach. We also ended on a rather troubling note regarding the feasibility of geometry reconstruct, namely that the reconstructed geometries exhibited unusual bond length correlations (section 5.3.2) and that degenerate geometries can be found in large region of phase space (section 5.4).

In this chapter we will begin by tackling the important task of quantifying the uncertainty on our geometry reconstructions, which surprisingly has not been performed by any previous study. We will take a heuristic approach, which provides some valuable estimates on the amount of uncertainty to expect and will help resolve the issue of unusual bond length correlations. A more rigorous and sophisticated approach of uncertainty quantification in the Bayesian inference framework has been considered and for which the motivation and methodology of this approach will be discussed.

6.1 UNCERTAINTY ON A RECONSTRUCTED GEOMETRY

The question of interest in this section is, *"how does uncertainty in the measured momentum vectors affect the uncertainty of the reconstructed geometry?"* We have already calculated the uncertainty on the momentum vectors in section 3.1.3 but we cannot derive an analytic formula for the uncertainty on the molecular parameters or the atomic positions.

6.1.1 *A heuristic approach*

We will take a very basic approach here by attempting to generalize a simple result for the propagation of error through a monotonically increasing function, that is a function which is always increasing and always has a nonnegative first derivative.

If a particular measurement $\bar{x}$ of a variable x carries some uncertainty ϵ such that the true value of $\bar{x}$ lies within some interval $\bar{x} - \epsilon \leq \bar{x} \leq \bar{x} + \epsilon$ then the true value of some arbitrary monotonically increasing scalar function $f(x)$ that is dependent on the value of the measurement will lie within some interval

$$f(\bar{x} - \epsilon) \leq f(\bar{x}) \leq f(\bar{x} + \epsilon) \tag{6.1}$$

Now if we relax the condition that f be a monotonically increasing function, then the true value of $f(\bar{x})$ can take on any value that f attains within the interval $\bar{x} - \epsilon \leq \bar{x} \leq \bar{x} + \epsilon$, and we can say very generally that it lies between

$$\min_{\bar{x} - \epsilon \leq x \leq \bar{x} + \epsilon} f(x) \leq f(\bar{x}) \leq \max_{\bar{x} - \epsilon \leq x \leq \bar{x} + \epsilon} f(x) \tag{6.2}$$

which however, may not yield a useful interval, especially in the presence of discontinuities or divergences. However, for finite-valued and well-behaved functions that do not change rapidly within the interval $\bar{x} - \epsilon \leq \bar{x} \leq \bar{x} + \epsilon$, (6.2) may provide a useful upper bound on the uncertainty in $f(\bar{x})$. This could be particularly accurate for small neighbourhoods about $\bar{x}$. As geometry reconstruction produces physically reasonable values for the molecular parameters without any discontinuities or divergences, we will attempt to use this idea to quantify the uncertainty on a geometry's reconstruction.

Before generalizing these two ideas for multivariable measurements and functions, it will be helpful to look at this idea for the case of measurement of a vector of two variables $\bar{\mathbf{x}} = (\bar{x}_1, \bar{x}_2)$ with an uncertainty described by the vector $\boldsymbol{\epsilon} = (\epsilon_1, \epsilon_2)$ and error propagation through a vector-valued function of two variables $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}))$. We will treat $\mathbf{f}(\mathbf{x})$ as non-parametric and assume that it has no analytic form, i. e. as a *black box* function as the mapping from momentum vector measurements to geometries cannot be parameterized or given in any analytical form.

In this case the true value of $\bar{\mathbf{x}}$ lies within a box described by $\bar{x}_1 - \epsilon_1 \leq \bar{x}_1 \leq \bar{x}_1 + \epsilon_1$ and $\bar{x}_2 - \epsilon_2 \leq \bar{x}_2 \leq \bar{x}_2 + \epsilon_2$ in the $x_1 x_2$ plane. If both $f_1(\mathbf{x})$ and $f_2(\mathbf{x})$ depend monotonically on x_1 and x_2 then determining the range of possible values of $\mathbf{f}(\bar{x})$ would be simple. Evaluating $\mathbf{f}(\mathbf{x})$ for

$$\mathbf{x} \in \mathbf{x}_{\text{ep}} = \left\{ \begin{pmatrix} x_1 - \epsilon_1 \\ x_2 - \epsilon_2 \end{pmatrix}, \begin{pmatrix} x_1 - \epsilon_1 \\ x_2 + \epsilon_2 \end{pmatrix}, \begin{pmatrix} x_1 + \epsilon_1 \\ x_2 - \epsilon_2 \end{pmatrix}, \begin{pmatrix} x_1 + \epsilon_1 \\ x_2 + \epsilon_2 \end{pmatrix} \right\} \tag{6.3}$$

where ep is an abbreviation for "endpoints" (as $\mathbf{x}_{\text{ep}}$ denotes the set of endpoints for the box containing feasible values of $\mathbf{x}$ in the x_1x_2 plane) would produce 4 point in the f_1f_2 plane, whose set we denote by $\mathbf{f}_{\text{ep}}$ and whose rectangular boundary encloses the possible values of $f(\bar{x})$. Thus the propagation of uncertainty in this case can be thought of a mapping from a rectangle in the x_1x_2 plane to a rectangle in the f_1f_2 plane.

If f_1 and f_2 do not depend monotonically on x_1 and x_2, then values of x_1, x_2 between the endpoints may produce values of f_1,f_2 that lie outside the rectangular boundary. A further heuristic would be to not only look at the endpoints, but also a set of uniformally distributed points in the x_1x_2 plane within the box of possible values for $\mathbf{x}$. In this case, the propagation of uncertainty can be thought of as a mapping from a rectangle in the x_1x_2 plane to an arbitrary region in the f_1f_2 plane whose boundary may not be rectangular anymore, or even a single region. In the case of a more complicated boundary, we will generalize (6.2) by describing the boundary using a *convex hull*. The convex hull C of a set of points $S = \{p_1, p_2, \ldots, p_n\}$ where $p_i \in \mathbb{R}^m$ for all i can be expressd mathematically as

$$C = \left\{ \sum_{i=1}^{n} \lambda_i p_i \,\middle|\, \lambda_i \geq 0 \text{ and } \sum_{i=1}^{n} \lambda_i = 1 \right\} \tag{6.4}$$

An analogy in two dimensions would be to stretch a rubber band around the set of points and let it rest, its final shape being the convex hull.

Extending this idea to our problem of geometry reconstruction, we have 9 measurements $\bar{\mathbf{p}} = (\bar{p}_1, \ldots, \bar{p}_9)$ with uncertainty $\boldsymbol{\epsilon} = (\epsilon_1, \ldots, \epsilon_9)$ and we are interested in the range of possible geometries as produced by $\mathbf{g}(\mathbf{p}) = (r_{12}(\mathbf{p}), r_{23}(\mathbf{p}), \theta(\mathbf{p}))$. The possible values of the momentum components are contained within a 9-dimensional hyperrectangle or box in momentum space, and we would like to obtain a 3-dimensional region in phase space describing the set of possible geometries that the measurement could correspond to. Generating a set of N uniformly distributed points within the 9-dimensional box would produce N^9 sets of momentum vectors, each of which must be reconstructed. This represents a rather unfeasible number of reconstructions to perform, thus we will start by looking at the set of endpoints only, which contains 512 (2^9) sets of momentum vectors. Attempting to reconstruct a geometry for each set of momentum vectors will ideally provide us with 512 geometries that together give us some idea into the range of possible geometries the measurement could possibly belong to. By only reconstructing the endpoints of our box in momentum space, we may be underestimating the range of possible geometries as points within the box may produce more extreme geometries when reconstructed.

To carry out this idea for geometry reconstruction, we randomly chose a representative geometry, $(r_{CO}, r_{CS}, \theta) = (130\,\text{pm}, 190\,\text{pm}, 169°)$ that does not lie in any of the degenerate regions discussed in figure 27 to avoid having to account for degenerate geometries. Simulating a Coulomb explosion using this geometry as the intial condition yields a set of momentum vectors, upon which we artificially placed an uncertainty of 5% for each momentum component such that $\boldsymbol{\epsilon} = 0.05\mathbf{p}$, producing 512 sets of momentum vectors corresponding to the corners of the 9-dimensional box in momentum space, that is, a 9-dimensional version of (6.3). Reconstructing a geometry for each set of momentum vectors, we obtain 512 geometries and all 512 sets of momentum vectors were successfully mapped to a unique geometry. Histograms showcasing the bond length and bond angle distributions of these reconstructed geometries are plotted in figure 29 along with scatter plots showcasing the bivariate relationships between the three molecular parameters.

We immediately see that a strikingly wide range of geometries are reconstructed assuming the momentum components carry an uncertainty of only 5%. Inspection suggests that the bond length and bond angle distributions roughly form a Gaussian distribution about the true parameters. The distributions encompass a wide range of possible bond lengths, 80 pm for the C−O bond and 175 pm for the C−S bond. The variability in the bond angle is not as extreme, encompassing a 6° range of possible bond angles.

The bivariate relationships shown in figures 29(d)-(f) suggest the range of possible geometries. Interestingly, the bond length correlation in figure 29(d) seems to follow a reciprocal relationship, a strikingly similar one to the unusual one observed in the reconstructions of experimental data. This suggests that uncertainty in the momentum measurements may lead to the reconstruction of a geometry with asymmetrically stretched bonds, and that the unusual relationship we observed in section 5.3.2 may have been a manifestation of measurement uncertainty and due to the highly sensitive nature of geometry reconstruction.

Placing a larger uncertainty on the momentum vector components further stretches the shape of the set of points in figure 29(d), modifying it to further resemble the more extreme reciprocal relationship between r_{CO} and r_{CS} found for longer pulse lengths (see figures 46 – 48).

Exposing the molecule to longer laser pulses provides a longer amount of time for the molecule to rearrange and may impart the atomic fragments with a larger initial momentum, thus decreasing the certainty with which the measured momentum vectors correspond to the true geometry. If we take a reciprocal relationship to be a signature indicating that too much measurement uncertainty is present for trustworthy geometry reconstructions, then we must conclude that

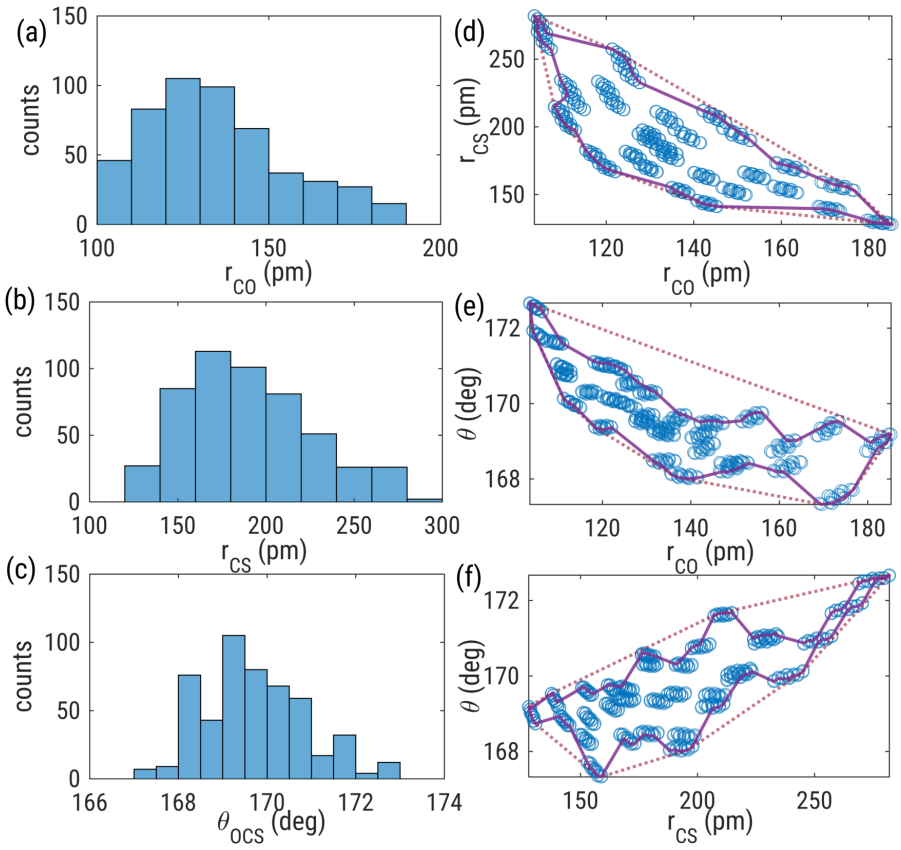

Figure 29: A heuristic estimate on the range of possible OCS $(2,2,2)$ geometries that may be reconstructed assuming a true geometry of $(r_{CO}, r_{CS}, \theta) = (130\,\text{pm}, 190\,\text{pm}, 169°)$ and 5% uncertainty on the measured momentum vectors. Histograms showcase the (a) C−O bond length, (b) C−S bond length, and (c) bond angle distributions of the reconstructed geometries. Scatter plots showcase the bivariate relationships between (d) r_{CO} and r_{CS}, (e) between r_{CO} and θ, and (f) between r_{CS} and θ. Each reconstructed geometry is plotted as an open blue circle. The boundary of the set of reconstructed geometries is calculated using two methods and plotted; The dotted red line denotes the convex hull and the solid purple line denotes the alpha shape (using (d) $\alpha = 15$, (e) $\alpha = 22$, and (f) $\alpha = 30$) of the set of points.

even an uncertainty as low as a few parts per hundred in the momentum vector components is "too much" and that accurate geometry reconstruction using Coulomb explosion imaging is unfeasible under such conditions.

We may use the area formed by the set of points in figures 29(d)-(f), plotted as red dotted lines by the use of a convex hull, as a quantitative measure of uncertainty.

An interesting observation resulting from the close inspection of the open blue circles in figures 29(d)-(f) is the clustering of geometries in phase space as geometries seem to cluster in little groups. Each geometry is actually paired with one other geometry (close inspection of the open blue circles should reveal that they appear in pairs, sometimes appearing as a single blurred circle) so that 256 points are visible unless the scatter plots are very closely inspected. As we picked one of two extreme values, $\bar{p} - \epsilon$ and $\bar{p} + \epsilon$ for each momentum component, each component may be responsible for the "splitting of the geometries" into pairs or clusters. This suggests that some uncertainty in certain momentum components may have a greater effect on the uncertainty of reconstructed geometries, and it may be worth quantifying the relative effects of each component (e. g. the components parallel and perpendicular to the bond lengths and the component out of the plane of the molecule). Repeating the analysis but with 1% uncertainty instead of 5% results in reconstructed geometries with $173\,\text{pm} < r_{CO} < 186\,\text{pm}$, $144\,\text{pm} < r_{CS} < 158\,\text{pm}$, and $173.4° < \theta < 173.4°$.

As we saw in section 3.2.2, the discrepancy in the OCS 7 fs $(2, 2, 2)$ momentum vectors is less than 2% and so while it will be a contributing factor to the uncertainty in the reconstructed geometries, most of the error must be due to our assumption that the molecule starts exploding from its equilibrium geometry and that each atomic fragment starts with zero initial momentum.

6.1.2 *Convex hulls and alpha shapes*

To quantify the uncertainty in the geometries, we will use two useful concepts from computational geometry, namely convex hulls and alpha shapes, which allow us to assign a shape and a volume to a set of points, and thus provide an additional heuristic quantitative measure of uncertainty. The convex hull of a set of points, introduced earlier and described mathematically by (6.4), is used to assign boundaries in phase space for figures 29(d)-(f), plotted as a dotted red line. Many algorithms exist to calculate the convex hull, especially in 2 or 3 dimensions (De Berg et al., 2008, ch. 11). However, the convex hull may grossly overestimate the area, as in figure 29(e) in comparison to (d).

Another boundary may be provided by the *alpha shape* of the set of points, plotted as solid purple lines. The concept of an alpha shape is

a generalization of the convex hull used to assign shapes and volumes to a set of points, parameterized by a real number $\alpha \geq 0$. The parameter α may be varied until a desired shape is produced. It may be thought of as a linear approximation to the shape of the set of points. They were first introduced by Edelsbrunner, Kirkpatrick, and Seidel (1983) for two-dimensional shapes, then for three-dimensional shapes (Edelsbrunner and Mücke, 1994) with applications in fields such as computer graphics. Interestingly, alpha shapes have been used to analytically compute shapes for macromolecules such proteins and estimate their molecular areas and volume (Liang et al., 1998). The three-dimensional alpha shape can be thought of intuitively as using an ice-cream scooper to scoop out chocolate chip ice-cream. (Da, 2017) describes this analogy better than any paraphrase I could come up with. He writes, "imagine a huge mass of ice-cream making up the space $\mathbb{R}^3$ and containing the points as "hard" chocolate pieces. Using one of these sphere-formed ice-cream spoons we carve out all parts of the ice-cream block we can reach without bumping into chocolate pieces, thereby even carving out holes in the inside (e.g. parts not reachable by simply moving the spoon from the outside). We will eventually end up with a (not necessarily convex) object bounded by caps, arcs and points. If we now straighten all "round" faces to triangles and line segments, we have an intuitive description of what is called the alpha-shape." The radius of the carving spoon is $\sqrt{\alpha}$.

A desirable feature of convex hulls is that they are unique for each set of points, while multiple distinct alpha shapes exist. This is a desirable property of alpha shapes as there is no formal concept of shape so no algorithm can determine the correct shape for a set of points, and indeed the alpha-shape is just one possibly family of shapes. However, the concept allows for an α to be picked that produces the most desirable shape [see Edelsbrunner and Mücke (1994) for concrete examples in three dimensions].

While a very haphazard measure of uncertainty, they are much easier to employ than the sophisticated uncertainty quantification framework of Bayesian inference and satisfy our basic needs for now. Figure 30 shows the three-dimensional convex hull enclosing all the reconstructed geometries from figure 29.

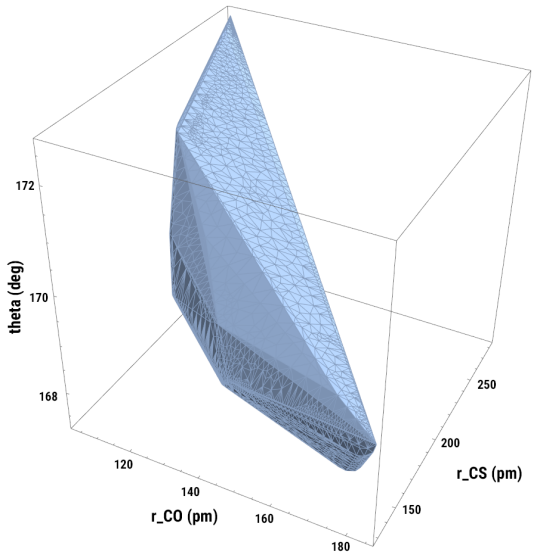

Figure 30: Three dimensional convex hull enclosing the possible OCS $(2,2,2)$ geometries that may be reconstructed assuming a true geometry of $(r_{CO}, r_{CS}, \theta) = (130\,\text{pm}, 190\,\text{pm}, 169°)$ and 5% uncertainty on the measured momentum vectors.

6.2 UNCERTAINTY QUANTIFICATION USING BAYESIAN INFERENCE

The heuristic uncertainty quantification performed in the previous section does provide some insight on the process of geometry reconstruction, and emphasized the large effect that measurement uncertainty must have played in our reconstruction of experimental data. This further highlights the importance of tackling the task of uncertainty quantification in a rigorous and sophisticated manner, in which case the Bayesian inference framework provides a mature set of methods, software, and learning material to do this.

In this section we will look at why a Bayesian approach makes sense for the geometry reconstruction problem and for inference problems in science more generally. The aim is to be expository and to that effect, many references are provided for further reading.

6.2.1 *Why take a Bayesian approach?*

The Bayesian point of view provides a more natural and intuitive way of thinking about uncertainty in the physical sciences. While virtually any analysis may be done in either the classical or Bayesian framework, the Bayesian approach tends to make more sense and can be philosophically superior (Gelman and Shalizi, 2013). Frequentist statistics defines the probability as a limiting case of repeated measurements and so after many repeated measurements, the true value of a parameter of interest is typically given as being within some *confidence interval* with 95% probability. The Bayesian approach allows us to write down the exact probability distribution we wish to find in terms of the measurements we have made and our prior beliefs and solve the resulting equations for it using Bayes' theorem, such that our prior beliefs are updated based on the measurements we

have made. Bayesian inference is also suitable for inferring unknown parameters in inverse problems.

Bayesian statistics actually predates the more commonly taught frequentist statistical methods (e. g. p-values, confidence intervals, etc.) but has made a strong resurgence in recent years due to rising computational abilities and more recently, available software for parameter estimation in statistical models using Markov chain Monte Carlo methods (Brooks, 2003). It is important to note that frequentist methods such as *t*-tests still have immense utility in the sciences, however, Bayesian inference methods provide the means of tackling more complex problems that are intractable otherwise. Bayesian methods have had large impacts in many scientific fields such as genetics (Beaumont and Rannala, 2004) and more recently, machine learning (Murphy, 2012), and an interpretation of quantum mechanics, termed *Quantum Bayesianism*, exists as well (Timpson, 2008).

Many have argued for philosophical superiority of the Bayesian approach, most famously by Efron (1986) in his article titled "*Why isn't everyone a Bayesian?*". Cousins (1995) authored a similarly titled article, "*Why isn't every physicist a Bayesian?*" comparing both the classical and Bayesian frameworks in the context of particle physics. A more recent article on the subject authored by Lyons (2012) appeared in *Physics Today*. Extensive review articles have been written on the general subject of Bayesian inference in physics and include numerous examples and case studies D'Agostini (2003), Dose (2003), and Toussaint (2011).

Numerous textbooks exist on the topic. Gelman et al. (2014) provide an excellent introduction to Bayesian data analysis, especially the theoretical foundations of Bayesian statistics, and Kruschke (2014) provides a practical and easy-to-follow introduction accompanied with many examples. Although not particularly writing about just Bayesian inference, Hogg, Bovy, and Lang (2010) provide a rather excellent and extensive tutorial in fitting data to models in the form of an article, and include numerous exercises relying on the use of Bayesian inference. VanderPlas (2014) also provides a primer comparing the frequentist and Bayesian approaches with several examples implemented in Python.

6.2.2 *Bayes' Theorem*

At the center of Bayesian statistics lies Bayes' theorem or Bayes' rule which describes the probability of an event based on prior knowledge of conditions that may be related to the event. It was first introduced by Reverend Thomas Bayes and published two years after his death with revisions by Ronald Price in a work titled "*An Essay towards solving a Problem in the Doctrine of Chances*" (Bayes and Price, 1763) although evidence suggests that the title was modified by Price

to support a theological argument, and that Bayes intended it to be "*A Method of Calculating the Exact Probability of All Conclusions founded on Induction*" (Stigler, 2013). Pierre-Simon Laplace then developed it further and published it in its modern form along with many fundamental results in statistics (Laplace, 1820). Dale (1999) discusses the history in more detail, and McGrayne (2011) traces the history of Bayes' rule in her popular science book and its applications from World War II to bioinformatics.

Bayes' theorem in its simplest form states that probability of observing an event A given that B is true is

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)} \tag{6.5}$$

where $P(B|A)$ is the probability that event B occurs given that A is true, and $P(A)$ and $P(B)$ are the probabilities that events A and B occur respectively. $P(A|B)$ denotes a conditional probability and can be expressed as $P(A|B) = P(A \cap B)/P(B)$ where $P(A \cap B)$ is the probability of both A and B occurring and $\cap$ denotes the intersection operation from set theory.[1]

For Bayesian inference of model parameters given that certain measurements or observations, the basic idea can be extracted from (6.5). Let us use θ to denote the parameters of a model, for example $\theta = (r_{12}, r_{23}, \phi)$ for a model of a triatomic molecule, and y to denote data that has been measured or observed. We are interested in finding the *posterior probability density* $P(\theta|y)$, the probability that our model is described by the parameters θ given that certain measurements y have been made. Using (6.5), we can write

$$P(\theta|y) = \frac{P(\theta)P(y|\theta)}{P(y)} \tag{6.6}$$

where $P(\theta)$ denotes the *prior distribution*, that is the probability distribution encapsulating our belief regarding the model parameters prior to making any measurements, and $P(y|\theta)$ is the *likelihood function* is the distribution of observed data conditional on the model parameters. $P(y)$ is the *marginal likelihood*,

$$P(y) = \int P(\theta)P(y|\theta)\, d\theta \tag{6.7}$$

or the distribution of measurements marginalized over the model parameters. It can be difficult to compute as a high-dimensional integral. As it does not depend on θ it is a constant when considering a fixed

1 The concepts of a *probability measure* and an *event*, which is a subset of a *sample space*, can all be defined precisely in a modern framework. See Kallenberg (2002) for an introduction to modern probability theory.

set of measurements y, and thus we may instead be interested in computing the *unormalized posterior density*

$$\underbrace{P(\theta|y)}_{\text{posterior}} \propto \underbrace{P(\theta)}_{\text{prior}} \times \underbrace{P(y|\theta)}_{\text{likelihood}} \tag{6.8}$$

Using our geometry reconstruction problem for a triatomic molecule as an example, we are modeling a molecule using the parameters $\theta = (r_{12}, r_{23}, \phi)$ where ϕ now denotes the bond angle, and the measurements y constitute a set of three momentum vectors $y = (\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$ so (6.5) gives

$$P(r_{12}, r_{23}, \phi|\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3) = \frac{P(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3|r_{12}, r_{23}, \phi)P(r_{12}, r_{23}, \phi)}{P(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)} \tag{6.9}$$

where $P(r_{12}, r_{23}, \phi)$ is the prior distribution and $P(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3|r_{12}, r_{23}, \phi)$ is the likelihood, that is the probability that the momentum vectors $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$ are observed given that the molecule is described by the parameters (r_{12}, r_{23}, ϕ).

In our case, a uniform distribution or a guess distribution may be used for the prior, or it may be obtained from quantum chemistry simulations for greater accuracy. Generally the posterior only depends strongly on the prior in the absence of sufficient data. Gelman et al. (2014) discusses the choice of prior distribution in great detail. The likelihood function $P(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3|r_{12}, r_{23}, \phi)$ is governed by a deterministic simulation as exploding a certain molecule will produce exactly one set of momentum vectors, so in reality it is a Dirac-delta function. However, in the presence of measurement uncertainty it will form a proper probability distribution. I have thought of two methods by which measurement uncertainty may be introduced into the deterministic forward simulation of the Coulomb explosion. One idea is to place Gaussian errors on the momentum vectors such that $\mathbf{p}_i \sim \mathbf{w}_i + \boldsymbol{\epsilon}_i$ where $\mathbf{w}_i$ is the measured momentum vector and $\boldsymbol{\epsilon}_i$ models the measurement uncertainty on the momentum vector, which may take the form of a multivariate Gaussian distribution, for example. This in effect employs a hierarchical statistical model and adds a second layer of variables [Kruschke (2014, ch. 9) and Gelman et al. (2014, ch. 5) discuss hierarchical statistical models from a Bayesian perspective]. A second idea is to allow the initial conditions of the Coulomb explosion simulation (the molecular parameters and initial momenta) to be probability distributions themselves such that a probabilistic solution of the differential equations is computed using Bayesian uncertainty quantification and inference. Such an approach has been recently developed, especially for the inference of parameters in dynamical systems, by Chkrebtii et al. (2016) and further detailed in her thesis (Chkrebtii, 2013).

The denominator of (6.9),

$$P(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3) = \int P(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3 | r_{12}, r_{23}, \phi)\, dr_{12}\, dr_{23}\, d\phi \tag{6.10}$$

will be extremely difficult to calculate in general as it is the probability that the momentum vectors $(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3)$ are measured, regardless of the molecular geometry that produced them. As the marginal likelihood is simply a scaling factor, we will instead be interested in computing the unnormalized posterior density

$$P(r_{12}, r_{23}, \phi | \mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3) \propto P(\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3 | r_{12}, r_{23}, \phi) P(r_{12}, r_{23}, \phi) \tag{6.11}$$

Evaluating this expression can still be difficult, and the usual approach is to randomly sample from it, forming a posterior probability density in the process. *Markov chain Monte Carlo* methods provide the means of doing this. We can also see why Bayesian inference is suitable for inferring latent variables in inverse problems as the probability density of interest for the inverse problem is directly related to the probability density of interest for the forward problem, which is generally known [see Stuart (2010) for a Bayesian perspective on inverse problems].

6.2.3 *Markov chain Monte Carlo*

Markov chain Monte Carlo (MCMC) methods are a class of algorithms for sampling from a probability distribution by constructing a *Markov chain* whose equilibrium distribution is the same as that of the probability distribution we are sampling from. Extensive literature exists on the theory of Markov chains (Levin, Peres, and Wilmer, 2009) and their mixing times, that is, the time until they come close to their equilibrium distribution. In practice, mature MCMC methods packaged as well-documented software, such as JAGS (Plummer, 2003) or Stan (Carpenter et al., 2017), do not require intricate knowledge of Markov chains or MCMC sampling algorithms to use, however, it is always useful for any practitioner to have some idea of their inner workings.

Diaconis (2009) gives an introduction to MCMC methods with a focus on their impact on the types of scientific and mathematical problems they have allowed us to solve, opening with a fascinating example involving deciphering coded messages between prison inmates. Richey (2010) and Robert and Casella (2011) give a short history of MCMC methods and their development.

An extensive handbook on MCMC methods does exist (Brooks et al., 2011) and includes a practical introduction to MCMC methods in its first chapter, authored by Geyer (2011), who discusses some important aspects of using Markov chains such as burn-in and convergence checking, and provides an overview of some important sam-

pling methods. Standard textbooks and references on MCMC methods include Gilks, Richardson, and Spiegelhalter (1995) and Christian and Casella (1999).

Two popular random walk MCMC methods include the Metropolis-Hastings algorithm and Gibbs sampling. The Metropolis-Hastings algorithm, first proposed by Metropolis et al. (1953) performs a random walk using some proposed probability density with some rejection criterion, and can be quite inefficient especially for high-dimensional integrals. Gibbs sampling uses the Metropolis-Hastings algorithm, however, it samples from parameterized probability distributions thus requiring no tuning which can be highly desirable when direct sampling is difficult. Casella and George (1992) gives an introduction to the Gibbs Sampler and it is discussed extensively throughout the handbook by Brooks et al. (2011). Just another Gibbs Samples (JAGS) (Plummer, 2003) along with its R package (RJAGS) are popular implementations. Brooks et al. (2011) discusses other random walk MCMC algorithms, such as slice sampling and reversible jump MCMC, extensively.

One popular algorithm not based on random walks, Hamiltonian MCMC, has risen in popularity in recent years due to its utility in sampling from high-dimensional distributions more effectively than random walk based algorithms. Inspired by a hybrid Monte Carlo method first proposed by Duane et al. (1987) for the numerical simulation of lattice field theories, Hamiltonian MCMC takes a Hamiltonian mechanics approach to sampling by subjecting a fictional particle to a potential energy function described by the target probability density distribution. Random sampling is performed by solving Hamilton's equations for the fictional particle. While seemingly bizarre, Hamiltonian MCMC does indeed work quite well in many cases. Neal (2011) provides an excellent and practical introduction. Stan (along with it's R wrapper, RStan, and Python wrapper, PyStan) is the most popular and mature implementation (Carpenter et al., 2017), finding use in many scientific fields and in the humanities as well.

If a method or framework for geometry reconstruction using Bayesian inference is developed and implemented, we humbly suggest naming it MMMGRUBS[2] (Molecular Motion Movies and Geometry Reconstruction Using Bayesian Statistics).

2 Grubs are beetle larvae, as enjoyed by Simba in the 1994 animated film *The Lion King*, mmm!

7 CONCLUSION

We have shown that geometry reconstruction can be performed quickly using two approaches, a heuristic lookup table discussed in chapter 4 and the more precise optimization routine relying on constrained nonlinear optimization methods in chapter 5. We used both methods to investigate the existence of multiple solutions, or degenerate geometries, and used different analyses to study their nature which should be performed before attempting to reconstruct any molecular geometry. We also investigated the effect of uncertainty in the momentum vectors on the reconstructed geometries in chapter 6, highlighting the sensitive nature of the task.

7.1 INFEASIBILITY OF GEOMETRY RECONSTRUCTION

We have found two major barriers to accurate geometry reconstruction. Firstly, it is highly sensitive to uncertainties in the momentum vectors, however, even if they are measured accurately enough that measurement uncertainty is no longer a concern, the initial momentum carried by each atomic fragment would need to be measured accurately as well since they introduce additional uncertainty that cannot be inferred otherwise. Secondly, the existence of degenerate geometries for large regions of phase space may make reconstruction difficult, especially if multiple degeneracies represent physically realizable structures. The problem is exasperated in the presence of uncertainty as the true geometry may now be contained in multiple unconnected regions in phase space.

We only investigated the simplest case possible by assuming the atomic fragments evolve on a purely classical Coulomb potential and that they possess zero initial momentum, and yet the reconstruction problem already seems to exhibit pathological behavior. Accounting for additional complexity through the use of a more accurate non-Coulombic potential, the measurement or estimation of initial momentum for the atomic fragments, and modeling the effect of the laser's electric field with the molecule are likely to increase the problem's sensitivity to uncertainty in the momentum vectors, and thus the entire task of geometry reconstruction appears to be quite complex and only possible under certain conditions, and becomes even more complex for larger molecules which require at least twice the number of degrees of freedom to describe. Additional degrees of freedom, introduced due to the modeling of additional phenomena or the study of larger molecules, would increase the dimensionality of

the optimization problem, possibly resulting in a problem that may be difficult to solve with the already advanced optimization methods employed due to the curse of dimensionality. It is also possible that accounting for additional complexity may introduce additional degeneracies through the introduction of extraneous degrees of freedom, which would make the accurate reconstruction of molecular geometries even more difficult.

7.2 A FRAMEWORK FOR GEOMETRY RECONSTRUCTION USING CEI

While geometry reconstruction using CEI can be difficult, it may be performed under certain conditions:

1. The momentum vectors are measured very accurately, with an error less than 1%.

2. The atomic fragments carry very little initial momentum. Or in the case that they do not, then the initial momenta of the atomic fragments must be measured experimentally, or estimated accurately (which would add further complexity to the geometry reconstruction problem).

3. Negligible molecular rearrangement occurs during the ionization process. This would require at least, the use of few-cycle laser pulses and depends on the molecule as well, as some may have a tendency to rearrange significantly on a time scale shorter than the shortest intense laser pulses.

4. The reconstructed geometries are not degenerate with other geometries. Thus regions of phase space containing degenerate geomtries must be mapped out.

5. The Coulomb potential very closely approximates the potential experienced by the atomic fragments during the Coulomb explosion.

If these conditions above hold, then we propose the following framework for geometry reconstruction:

1. Choose conventions for describing the molecule's structure and the orientation of the asymptotic momentum vectors following a Coulomb explosion, for example, as described in section 3.4..

2. Map out the degenerate regions of your molecular system, that is, find the regions of phase space which contain degenerate geometries (as done in section 5.4). You may choose to completely ignore geometries from these regions if multiple degeneracies correspond to physically realizable geometries.

3. Filter out momentum vector measurements that exhibit a large error, such as by not summing to zero, or that correspond to an explosion that is highly non-Coulombic, that is the kinetic energy of the atomic fragments significantly differs from the kinetic energy expected by a purely Coulombic explosion. If not using the Coulomb potential, then compare with simulations using the more appropriate potential.

4. Set up the optimization problem (section 5.2) or the statistical model (section 6.2) for geometry reconstruction. Inequality constraints may be used to constrain the reconstructed geometries to have feasible parameters. If using Bayesian inference is used, quantum chemistry simulations of the ground or excited state may provide prior distributions, otherwise uninformative priors may be used.

5. If using Bayesian inference, then the calculated posterior distributions should already have taken into account the effect of uncertainty. If not, the uncertainty in the reconstructed geometries must be quantified.

6. Plot the final geometry. Bond length and bond angle (as well as dihedral angles) distributions should be reported, as well as the correlations between each of these parameters (section 5.3.2). This may constitute a single frame of a molecular movie.

7. Repeat this process for each frame, which may be for each pulse length in a pump-probe CEI experiment, thus producing a molecular movie.

REFERENCES

Abdulaziz, A. A. (2008). On the Egyptian method of decomposing $2/n$ into unit fractions. *Historia Mathematica* **35**(1), 1–18 (cit. on p. 42).

Ablikim, U. et al. (2016). Identification of absolute geometries of cis and trans molecular isomers by Coulomb Explosion Imaging. *Scientific Reports* **6**(1) (cit. on p. 4).

Arnold, J. B. (2017). *ggthemes: Extra Themes, Scales and Geoms for 'ggplot2'* (cit. on p. 145).

Asenjo, D., Stevenson, J. D., Wales, D. J., and Frenkel, D. (2013). Visualizing basins of attraction for different minimization algorithms. *The Journal of Physical Chemistry B* **117**(42), 12717–12723 (cit. on p. 78).

Baker, J. and Chan, F. (1996). The location of transition states: A comparison of Cartesian, Z-matrix, and natural internal coordinates. *Journal of computational chemistry* **17**(7), 888–904 (cit. on p. 34).

Baker, J. and Hehre, W. J. (1991). Geometry optimization in cartesian coordinates: The end of the Z-matrix? *Journal of computational chemistry* **12**(5), 606–610 (cit. on p. 34).

Bandrauk, A. D. and Chelkowski, S. (2001). Dynamic imaging of nuclear wave functions with ultrashort UV laser pulses. *Physical Review Letters* **87**(27), 273004 (cit. on p. 14).

Bayes, T. and Price, R. (1763). An Essay towards Solving a Problem in the Doctrine of Chances. By the Late Rev. Mr. Bayes, F. R. S. Communicated by Mr. Price, in a Letter to John Canton, A. M. F. R. S. *Philosophical Transactions (1683-1775)* **53**, 370–418 (cit. on p. 90).

Beaumont, M. A. and Rannala, B. (2004). The Bayesian revolution in genetics. *Nature Reviews Genetics* **5**(4), 251–261 (cit. on p. 90).

Beck, W., Evers, J., Göbel, M., Oehlinger, G., and Klapötke, T. M. (2007). The Crystal and Molecular Structure of Mercury Fulminate (Knallquecksilber). *Zeitschrift für anorganische und allgemeine Chemie* **633**(9), 1417–1422 (cit. on p. 4).

Bellman, R. E. (1957). *Dynamic programming*. Princeton, New Jersey: Princeton University Press. 340 pp. (cit. on p. 68).

Bengio, Y. (2016). Machines who learn. *Scientific American* **314**(6), 46–51 (cit. on p. 68).

Birge, R. T. (1939). The propagation of errors. *American Journal of Physics* **7**(6), 351–357 (cit. on p. 26).

Bocharova, I. et al. (2011). Charge Resonance Enhanced Ionization of CO_2 Probed by Laser Coulomb Explosion Imaging. *Physical Review Letters* **107**(6), 063201 (cit. on pp. 36, 37, 41, 49, 76).

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge University Press. 727 pp. (cit. on pp. 58, 60).

Branch, M. A., Coleman, T. F., and Li, Y. (1999). A subspace, interior, and conjugate gradient method for large-scale bound-constrained minimization problems. *SIAM Journal on Scientific Computing* **21**(1), 1–23 (cit. on p. 60).

Brichta, J. P., Walker, S. J., Helsten, R, and Sanderson, J. H. (2007). Ultrafast imaging of multielectronic dissociative ionization of CO_2 in an intense laser field. *Journal of Physics B: Atomic, Molecular and Optical Physics* **40**(1), 117–129 (cit. on pp. 36, 37, 41, 49).

Brichta, J.-P., Seaman, A. N., and Sanderson, J. H. (2009). Ultrafast imaging of polyatomic molecules with simplex algorithm. *Computer Physics Communications* **180**(2), 197–200 (cit. on pp. 35, 36, 38, 55).

Brooks, S. P. (2003). Bayesian computation: a statistical revolution. *Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences* **361**(1813), 2681–2697 (cit. on p. 90).

Brooks, S., Gelman, A., Jones, G., and Meng, X.-L. (2011). *Handbook of Markov chain Monte Carlo*. Boca Raton, Florida: CRC press. 619 pp. (cit. on pp. 93, 94).

Byrd, R. H., Gilbert, J. C., and Nocedal, J. (2000). A trust region method based on interior point techniques for nonlinear programming. *Mathematical Programming* **89**(1), 149–185 (cit. on p. 65).

Byrd, R. H., Hribar, M. E., and Nocedal, J. (1999). An interior point algorithm for large-scale nonlinear programming. *SIAM Journal on Optimization* **9**(4), 877–900 (cit. on p. 65).

Carpenter, B., Gelman, A., Hoffman, M., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. (2017). Stan: A Probabilistic Programming Language. *Journal of Statistical Software* **76**(1), 1–32 (cit. on pp. 93, 94).

Casella, G. and George, E. I. (1992). Explaining the Gibbs sampler. *The American Statistician* **46**(3), 167–174 (cit. on p. 94).

Chang, W. (2012). *R Graphics Cookbook: Practical Recipes for Visualizing Data*. Sebastopol, California: O'Reilly Media, Inc. 416 pp. (cit. on p. 145).

Chelkowski, S. and Bandrauk, A. D. (2002). Measuring moving nuclear wave packets using laser Coulomb-explosion imaging. *Physical Review A* **65**(2), 023403 (cit. on p. 14).

Chkrebtii, O. A. (2013). Probabilistic solution of differential equations for Bayesian uncertainty quantification and inference. PhD thesis. Simon Fraser University (cit. on p. 92).

Chkrebtii, O. A., Campbell, D. A., Calderhead, B., and Girolami, M. A. (2016). Bayesian solution uncertainty quantification for differential equations. *Bayesian Analysis* **11**(4), 1239–1267 (cit. on p. 92).

Christian, P. R. and Casella, G. (1999). *Monte Carlo statistical methods*. 2nd ed. New York: Springer. 649 pp. (cit. on p. 94).

Conn, A. R., Scheinberg, K., and Vicente, L. N. (2009). *Introduction to Derivative-free Optimization*. Philadelphia: Society for Industrial and Applied Mathematics. 289 pp. (cit. on p. 16).

Cornford, F. M. (2014). *Plato's cosmology: the Timaeus of Plato*. Routledge. 398 pp. (cit. on p. 4).

Cousins, R. D. (1995). Why isn't every physicist a Bayesian? en. *American Journal of Physics* **63**(5), 398 (cit. on p. 90).

D'Agostini, G. (2003). Bayesian inference in processing experimental data: principles and basic applications. *Reports on Progress in Physics* **66**(9), 1383 (cit. on p. 90).

Da, T. K. F. (2017). 2D Alpha Shapes. *CGAL User and Reference Manual*. 4.10. CGAL Editorial Board (cit. on p. 88).

Dale, A. I. (1999). *A history of inverse probability: From Thomas Bayes to Karl Pearson*. 2nd ed. New York: Springer Science & Business Media. 671 pp. (cit. on p. 91).

Dalitz, R. (1953). CXII. On the analysis of τ-meson data and the nature of the τ-meson. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **44**(357), 1068–1080 (cit. on p. 17).

Dauphin, Y. N., Pascanu, R., Gulcehre, C., Cho, K., Ganguli, S., and Bengio, Y. (2014). Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Advances in Neural Information Processing Systems 27 (NIPS 2014)*, 2933–2941 (cit. on p. 68).

De Berg, M., Cheong, O., Van Kreveld, M., and Overmars, M. (2008). *Computational Geometry: Introduction*. Springer. 386 pp. (cit. on p. 87).

Dennis Jr, J. E., Gay, D. M., and Welsch, R. E. (1981). Algorithm 573: NL2SOL—an adaptive nonlinear least-squares algorithm [E4]. *ACM Transactions on Mathematical Software (TOMS)* **7**(3), 369–383 (cit. on p. 60).

Diaconis, P. (2009). The Markov chain Monte Carlo revolution. *Bulletin of the American Mathematical Society* **46**(2), 179–205 (cit. on p. 93).

Dörner, R., Mergel, V., Jagutzki, O., Spielberger, L., Ullrich, J., Moshammer, R., and Schmidt-Böcking, H. (2000). Cold Target Recoil Ion Momentum Spectroscopy: a 'momentum microscope' to view atomic collision dynamics. *Physics Reports* **330**(2), 95–192 (cit. on p. 6).

Dose, V (2003). Bayesian inference in physics: case studies. *Reports on Progress in Physics* **66**(9), 1421–1461 (cit. on p. 90).

Duane, S., Kennedy, A. D., Pendleton, B. J., and Roweth, D. (1987). Hybrid Monte Carlo. *Physics letters B* **195**(2), 216–222 (cit. on p. 94).

Duong, T. and Hazelton, M. (2003). Plug-in bandwidth matrices for bivariate kernel density estimation. *Journal of Nonparametric Statistics* **15**(1), 17–30 (cit. on p. 32).

Easterbrook, S. M. (2014). Open code for open science? *Nature Geoscience* **7**(11), 779–781 (cit. on p. 69).

Edelsbrunner, H., Kirkpatrick, D., and Seidel, R. (1983). On the shape of a set of points in the plane. *IEEE Transactions on information theory* **29**(4), 551–559 (cit. on p. 88).

Edelsbrunner, H. and Mücke, E. P. (1994). Three-dimensional alpha shapes. *ACM Transactions on Graphics (TOG)* **13**(1), 43–72 (cit. on p. 88).

Efron, B. (1986). Why isn't everyone a Bayesian? *The American Statistician* **40**(1), 1–5 (cit. on p. 90).

Emerson, J. W., Green, W. A., Schloerke, B., Crowley, J., Cook, D., Hofmann, H., and Wickham, H. (2013). The Generalized Pairs Plot. *Journal of Computational and Graphical Statistics* **22**(1), 79–91 (cit. on pp. 29, 145).

Frasinski, L. J., Codling, K., and Hatherly, P. A. (1989). Covariance Mapping: A Correlation Method Applied to Multiphoton Multiple Ionization. *Science* **246**(4933), 1029–1031 (cit. on p. 8).

Frasinski, L. J., Codling, K., Hatherly, P., Barr, J., Ross, I. N., and Toner, W. T. (1987). Femtosecond dynamics of multielectron dissociative ionization by use of a picosecond laser. *Physical Review Letters* **58**(23), 2424–2427 (cit. on p. 8).

Gagnon, J, Lee, K. F., Rayner, D. M., Corkum, P. B., and Bhardwaj, V. R. (2008). Coincidence imaging of polyatomic molecules via laser-induced Coulomb explosion. *Journal of Physics B: Atomic, Molecular and Optical Physics* **41**(21), 215104 (cit. on pp. 16, 17).

Gagnon, J. (2006). Laser Coulomb Explosion Imaging of Polyatomic Molecules. MA thesis. University of Ottawa (cit. on p. 16).

Gaillard, M. J., Gemmell, D. S., Goldring, G., Levine, I., Pietsch, W. J., Poizat, J. C., Ratkowski, A. J., Remillieux, J., Vager, Z., and Zabransky, B. J. (1978). Experimental determination of the structure of H_3^+. *Physical Review A* **17**(6), 1797–1803 (cit. on pp. 10, 11, 17).

Gardner, M. (2005). *The New Ambidextrous Universe: Symmetry and Asymmetry from Mirror Reflections to Superstrings*. 3rd ed. Dover Publications. 416 pp. (cit. on p. 4).

Gelman, A. and Shalizi, C. R. (2013). Philosophy and the practice of Bayesian statistics. *British Journal of Mathematical and Statistical Psychology* **66**(1), 8–38 (cit. on p. 89).

Gelman, A., Carlin, J. B., Stern, H. S., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2014). *Bayesian data analysis*. 3rd ed. Boca Raton, Florida: CRC press. 675 pp. (cit. on pp. 90, 92).

Gemmell, D. S., Remillieux, J., Poizat, J. C., Gaillard, M. J., Holland, R. E., and Vager, Z. (1975). Evidence for an Alignment

Effect in the Motion of Swift Ion Clusters through Solids. *Physical Review Letters* **34**(23), 1420–1424 (cit. on p. 10).

Geyer, C. (2011). Introduction to Markov chain Monte Carlo. *Handbook of Markov Chain Monte Carlo*. Ed. by S. Brooks, A. Gelman, G. Jones, and X.-L. Meng. Boca Raton, Florida: CRC Press, pp. 3–48 (cit. on p. 93).

Gilks, W. R., Richardson, S., and Spiegelhalter, D. (1995). *Markov chain Monte Carlo in practice*. Boca Raton, Florida: CRC press. 512 pp. (cit. on p. 94).

Gillings, R. J. (1974). The Recto of the Rhind mathematical Papyrus How Did the Ancient Egyptian Scribe Prepare it? *Archive for History of Exact Sciences* **12**(4), 291–298 (cit. on p. 42).

Gillings, R. J. (1982). *Mathematics in the time of the pharaohs*. New York: Dover Publications. 288 pp. (cit. on p. 42).

Glanville, S. R. K. (1927). The Mathematical Leather Roll in the British Museum. *The Journal of Egyptian Archaeology* **13**(3/4), 232–239 (cit. on pp. 42–44).

Greene, C. H. (2010). Universal insights from few-body land. *Physics Today* **63**(3), 40–45 (cit. on p. 17).

Greenwood, N. N. and Earnshaw, A. (1997). *Chemistry of the Elements*. 2nd ed. Oxford: Butterworth-Heinemann. 1600 pp. (cit. on pp. 36, 38).

Halmos, P. R. (2017). *Naive Set Theory*. Mineola, New York: Dover Publications. 112 pp. (cit. on pp. 58, 78).

Hartigan, J. (1975). Printer graphics for clustering. *Journal of Statistical Computation and Simulation* **4**(3), 187–213 (cit. on p. 29).

Hayashi, T. (1997). Āryabhaṭa's Rule and Table for Sine-Differences. *Historia Mathematica* **24**(4), 396–406 (cit. on p. 42).

Hendrickson, W. A. (1995). X Rays in Molecular Biophysics. *Physics Today* **48**(11), 42–50 (cit. on p. 4).

Herwig, P., Schwalm, D., Čížek, M., Golser, R., Grieser, M., Heber, O., Repnow, R., Wolf, A., and Kreckel, H. (2013a). Metastable states of D_2^- observed by foil-induced Coulomb explosion imaging. *Physical Review A* **87**(6) (cit. on p. 13).

Herwig, P. et al. (2013b). Imaging the absolute configuration of a chiral epoxide in the gas phase. *Science* **342**(6162), 1084–1086 (cit. on pp. 10, 13).

Hogg, D. W., Bovy, J., and Lang, D. (2010). Data analysis recipes: Fitting a model to data. *arXiv preprint arXiv:1008.4686* (cit. on p. 90).

Hunter, J. D. (2007). Matplotlib: A 2D graphics environment. *Computing In Science & Engineering* **9**(3), 90–95 (cit. on p. 145).

Ibrahim, H. et al. (2014). Tabletop imaging of structural evolutions in chemical reactions demonstrated for the acetylene cation. *Nature Communications* **5**, 4422 (cit. on pp. 4, 6).

Jones, E., Oliphant, T., Peterson, P., et al. (2001–). *SciPy: Open source scientific tools for Python* (cit. on p. 145).

Jones, M. C., Marron, J. S., and Sheather, S. J. (1996). A brief survey of bandwidth selection for density estimation. *Journal of the American Statistical Association* **91**(433), 401–407 (cit. on p. 32).

Jordon-Thaden, B. et al. (2011). Structure and Stability of the Negative Hydrogen Molecular Ion. *Physical Review Letters* **107**(19) (cit. on p. 13).

Kallenberg, O. (2002). *Foundations of modern probability*. 2nd ed. New York: Springer Science & Business Media. 638 pp. (cit. on p. 91).

Kanter, E. P., Cooney, P. J., Gemmell, D. S., Groeneveld, K.-O., Pietsch, W. J., Ratkowski, A. J., Vager, Z., and Zabransky, B. J. (1979). Role of excited electronic states in the interactions of fast (MeV) molecular ions with solids and gases. *Physical Review A* **20**(3), 834–854 (cit. on p. 8).

Karimi, R., Liu, W.-K., and Sanderson, J. (2016). Femtosecond Laser-Induced Coulomb Explosion Imaging. *Advances in Multi-Photon Processes and Spectroscopy (Volume 23)*. Ed. by S. H. Lin, A. A. Villaeys, and Y. Fujimura. Singapore: World Scientific Publishing. Chap. 3, pp. 97–153 (cit. on p. 8).

Keogh, E. and Mueen, A. (2010). Curse of Dimensionality. *Encyclopedia of Machine Learning*. Ed. by C. Sammut and G. I. Webb. Boston: Springer, 257–258 (cit. on p. 68).

Kepler, J., Aiton, E., Duncan, A., and Field, J. (1997). *The Harmony of the World*. Memoirs of the American Philosophical Society, Book 209. Philadelphia: American Philosophical Society. 549 pp. (cit. on p. 5).

Kepler, J. (1619). *Ioannis Keppleri Harmonices mundi libri V [The Five Books of Johannes Kepler's The Harmony of the World]*. Linz: Johannes Planck. 255 pp. Published in Latin. The five books are bound and published as a single volume. (Cit. on p. 5).

Kitamura, T., Nishide, T., Shiromaru, H., Achiba, Y., and Kobayashi, N. (2001). Direct observation of "dynamic" chirality by Coulomb explosion imaging. *The Journal of Chemical Physics* **115**(1), 5–6 (cit. on p. 4).

Kruschke, J. (2014). *Doing Bayesian data analysis: A tutorial with R, JAGS, and Stan*. 2nd ed. Academic Press. 776 pp. (cit. on pp. 90, 92).

Ku, H. H. (1966). Notes on the use of propagation of error formulas. *Journal of Research of the National Bureau of Standards* **70**(4) (cit. on pp. 25, 26).

Kunitski, M. et al. (2015). Observation of the Efimov state of the helium trimer. *Science* **348**(6234), 551–555 (cit. on pp. 17, 18, 81).

Lagarias, J. C., Reeds, J. A., Wright, M. H., and Wright, P. E. (1998). Convergence properties of the Nelder-Mead simplex method in low dimensions. *SIAM Journal on optimization* **9**(1), 112–147 (cit. on p. 41).

Laplace, P. S. de (1820). *Théorie analytique des probabilités*. Vol. 7. Courcier (cit. on p. 91).

LeCun, Y., Bengio, Y., and Hinton, G. (2015). Deep learning. *Nature* **521**(7553), 436–444 (cit. on p. 68).

Levin, D. A., Peres, Y., and Wilmer, E. L. (2009). *Markov chains and mixing times*. Providence, Rhode Island: American Mathematical Society. 371 pp. (cit. on p. 93).

Légaré, F., Lee, K., Litvinyuk, I., Dooley, P., Bandrauk, a., Villeneuve, D., and Corkum, P. (2005a). Imaging the time-dependent structure of a molecule as it undergoes dynamics. *Physical Review A* **72**(5), 052717 (cit. on pp. 15, 49).

Légaré, F. et al. (2005b). Laser Coulomb-explosion imaging of small molecules. *Physical Review A* **71**(1), 013415 (cit. on pp. 1, 15, 16, 49).

Liang, J., Edelsbrunner, H., Fu, P., Sudhakar, P. V., and Subramaniam, S. (1998). Analytical shape computation of macromolecules: I. Molecular area and volume through alpha shape. *Proteins: Structure, Function, and Bioinformatics* **33**(1), 1–17 (cit. on p. 88).

Lide, D. R. (2007). *CRC Handbook of Chemistry and Physics*. 88th ed. CRC Press. 2640 pp. (cit. on p. 39).

Liekhus-Schmaltz, C. E. et al. (2015). Ultrafast isomerization initiated by X-ray core ionization. *Nature Communications* **6**, 8199 (cit. on pp. 4, 6).

Lyons, L. (2012). Discovery or fluke: statistics in particle physics. en. *Physics Today* **65**(7), 45–51 (cit. on p. 90).

Maher, D. W. and Makowski, J. F. (2001). Literary evidence for Roman arithmetic with fractions. *Classical Philology* **96**(4), 376–399 (cit. on p. 42).

Maine, P, Strickland, D, Bado, P, Pessot, M, and Mourou, G (1988). Generation of ultrahigh peak power pulses by chirped pulse amplification. *IEEE Journal of Quantum Electronics* **24**(2), 398–403 (cit. on p. 14).

Mason, P. E., Uhlig, F., Vaněk, V., Buttersack, T., Bauerecker, S., and Jungwirth, P. (2015). Coulomb explosion during the early stages of the reaction of alkali metals with water. *Nature Chemistry* **7**(3), 250–254 (cit. on p. 3).

Mathur, D, Krishnakumar, E, Rajgara, F. A., Raheja, U. T., and Krishnamurthi, V (1992). Energy distributions of recoil ions produced in 100 MeV collisions of Si^{8+} with CO_2 and CS_2 molecules. *Journal of Physics B: Atomic, Molecular and Optical Physics* **25**(13), 2997–3008 (cit. on pp. 36, 38).

Matsuda, A., Takahashi, E. J., and Hishikawa, A. (2014). Time-resolved laser Coulomb explosion imaging using few-cycle intense laser pulses: Application to exploding CS2 in highly charged states. *Journal of Electron Spectroscopy and Related Phenomena* **195**(2014), 327–331 (cit. on pp. 1, 19).

Maxwell, J. C. (1873). Molecules. *Nature* **8**(204), 437–441 (cit. on p. 4).
McGrayne, S. B. (2011). *The theory that would not die: how Bayes' rule cracked the enigma code, hunted down Russian submarines, & emerged triumphant from two centuries of controversy*. New Haven, CT: Yale University Press. 360 pp. (cit. on p. 91).
McKinnon, K. I. (1998). Convergence of the Nelder-Mead Simplex Method to a Nonstationary Point. *SIAM Journal on Optimization* **9**(1), 148–158 (cit. on p. 41).
Metropolis, N., Rosenbluth, A. W., Rosenbluth, M. N., Teller, A. H., and Teller, E. (1953). Equation of State Calculations by Fast Computing Machines. *The Journal of Chemical Physics* **21**(6), 1087–1092 (cit. on p. 94).
Miller, R. J. D., Ernstorfer, R., Harb, M., Gao, M., Hebeisen, C. T., Jean-Ruel, H., Lu, C., Moriena, G., and Sciaini, G. (2010). 'Making the molecular movie': first frames. *Acta Crystallographica Section A: Foundations of Crystallography* **66**(2), 137–156 (cit. on p. 3).
Millman, K. J. and Pérez, F. (2014). Developing open-source scientific practice. *Implementing reproducible research*. Ed. by V. Stodden, F. Leisch, and R. D. Peng. Boca Raton, Florida: CRC Press, pp. 149–183 (cit. on p. 69).
Mizogawa, T., Awaya, Y., Isozumi, Y., Katano, R., Ito, S., and Maeda, N. (1992). New readout technique for two-dimensional position-sensitive detectors. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* **312**(3), 547–552 (cit. on p. 23).
Mizogawa, T., Shiromaru, H., Sato, M., and Ito, Y. (2002). A two-dimensional position-sensitive ion detector based on modified backgammon method with weighted-coupling capacitors. *International Journal of Mass Spectrometry* **215**(1), 141–149 (cit. on p. 22).
Mizuse, K., Kitano, K., Hasegawa, H., and Ohshima, Y. (2015). Quantum unidirectional rotation directly imaged with molecules. *Science Advances* **1**(6), e1400185 (cit. on p. 4).
Moré, J. J. and Sorensen, D. C. (1983). Computing a trust region step. *SIAM Journal on Scientific and Statistical Computing* **4**(3), 553–572 (cit. on p. 65).
Moulton, P. F. (1986). Spectroscopic and laser characteristics of Ti:Al_2O_3. *Journal of the Optical Society of America B* **3**(1), 125–133 (cit. on p. 13).
Murphy, K. P. (2012). *Machine learning: a probabilistic perspective*. Cambridge, Massachusetts: MIT press. 1104 pp. (cit. on p. 90).
Nagaya, K. and Bandrauk, A. D. (2004). Laser Coulomb explosion imaging of linear triatomic molecules. *Journal of Physics B: Atomic, Molecular and Optical Physics* **37**(14), 2829–2853 (cit. on pp. 14, 15).
Neal, R. M. (2011). MCMC using Hamiltonian dynamics. *Handbook of Markov Chain Monte Carlo*. Ed. by S. Brooks, A. Gelman, G.

Jones, and X.-L. Meng. Boca Raton, Florida: CRC Press, pp. 113–162 (cit. on p. 94).

Nelder, J. A. and Mead, R. (1965). A Simplex Method for Function Minimization. *The Computer Journal* **7**(4), 308–313 (cit. on p. 41).

Neugebauer, O., Sachs, A. J., and Götze, A. (1945). *Mathematical Cuneiform Texts*. Vol. 29. New Haven: American Oriental Society and the American Schools of Oriental Research. 177 pp. (cit. on p. 42).

Nocedal, J. and Wright, S. (2006). *Numerical Optimization*. New York: Springer Science & Business Media. 664 pp. (cit. on pp. 58, 64, 65, 67).

Novick, R., Lipworth, E., and Yergin, P. F. (1955). Fine structure of singly ionized helium. *Physical Review* **100**(4), 1153–1173 (cit. on p. 9).

Oliphant, T. E. (2007). Python for scientific computing. *Computing in Science & Engineering* **9**(3) (cit. on p. 145).

Parsons, J., Holmes, J. B., Rojas, J. M., Tsai, J., and Strauss, C. E. M. (2005). Practical conversion from torsion space to Cartesian space for In Silico protein synthesis. *Journal of Computational Chemistry* **26**(10), 1063–1068 (cit. on p. 34).

Pascanu, R., Dauphin, Y. N., Ganguli, S., and Bengio, Y. (2014). On the saddle point problem for non-convex optimization. *arXiv preprint arXiv:1405.4604* (cit. on p. 68).

Peng, C., Ayala, P. Y., Schlegel, H. B., and Frisch, M. J. (1996). Using redundant internal coordinates to optimize equilibrium geometries and transition states. *Journal of Computational Chemistry* **17**(1), 49–56 (cit. on p. 34).

Pitzer, M. et al. (2013). Direct Determination of Absolute Molecular Stereochemistry in Gas Phase by Coulomb Explosion Imaging. *Science* **341**(6150), 1096–1100 (cit. on p. 4).

Plummer, M. (2003). JAGS: A program for analysis of Bayesian graphical models using Gibbs sampling. *Proceedings of the 3rd international workshop on distributed statistical computing* **124**, 125 (cit. on pp. 93, 94).

Posthumus, J. H. (2004). The dynamics of small molecules in intense laser fields. *Reports on Progress in Physics* **67**(5), 623–665 (cit. on p. 8).

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. (2007). *Numerical Recipes: The Art of Scientific Computing*. 3rd ed. Cambridge: Cambridge University Press. 1256 pp. (cit. on p. 41).

Pujol, J. (2007). The solution of nonlinear inverse problems and the Levenberg-Marquardt method. *Geophysics* **72**(4), W1–16 (cit. on p. 60).

R Core Team (2017). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing. Vienna, Austria (cit. on p. 145).

Ramadhan, A., Wales, B., Karimi, R., Gauthier, I., MacDonald, M., Zuin, L., and Sanderson, J. (2016). Ultrafast molecular dynamics of dissociative ionization in OCS probed by soft x-ray synchrotron radiation. *Journal of Physics B: Atomic, Molecular and Optical Physics* **49**(21), 215602 (cit. on pp. 1, 6, 7, 18, 19, 22, 24).

Richey, M. (2010). The evolution of Markov chain Monte Carlo methods. *American Mathematical Monthly* **117**(5), 383–413 (cit. on p. 93).

Robert, C. and Casella, G. (2011). A short history of Markov chain Monte Carlo: Subjective recollections from incomplete data. *Statistical Science*, 102–115 (cit. on p. 93).

Schloerke, B. et al. (2017). *ggobi/ggally: v1.3.1* (cit. on p. 145).

Schulz, M, Fischer, D, Ferger, T, Moshammer, R, and Ullrich, J (2007). Four-particle Dalitz plots to visualize atomic breakup processes. *Journal of Physics B: Atomic, Molecular and Optical Physics* **40**(15), 3091–3099 (cit. on p. 19).

Scott, A. and Hall, H. R. (1927). Laboratory Notes: Egyptian Leather Roll of the Seventeenth Century B.C. *The British Museum Quarterly* **2**(2), 56–57 (cit. on p. 42).

Scott, D. W. (2015). *Multivariate density estimation: theory, practice, and visualization*. 2nd ed. Hoboken, New Jersey: John Wiley & Sons. 384 pp. (cit. on pp. 31, 32).

Siegmann, B, Werner, U, Lutz, H. O., and Mann, R (2002). Complete Coulomb fragmentation of CO_2 in collisions with 5.9 MeV u^{-1} Xe^{18+} and Xe^{43+}. *Journal of Physics B: Atomic, Molecular and Optical Physics* **35**(17), 3755–3765 (cit. on pp. 36, 38).

Silverman, B. W. (1986). *Density estimation for statistics and data analysis*. Chapman and Hall. 176 pp. (cit. on p. 32).

Slater, C. S., Blake, S., Brouard, M., Lauer, A., Vallance, C., Bohun, C. S., Christensen, L., Nielsen, J. H., Johansson, M. P., and Stapelfeldt, H. (2015). Coulomb-explosion imaging using a pixel-imaging mass-spectrometry camera. *Physical Review A* **91**(5) (cit. on p. 33).

Spence, D. E., Kean, P. N., and Sibbett, W. (1991). 60-fsec pulse generation from a self-mode-locked Ti: sapphire laser. *Optics letters* **16**(1), 42–44 (cit. on p. 14).

Spendley, W., Hext, G. R., and Himsworth, F. R. (1962). Sequential Application of Simplex Designs in Optimisation and Evolutionary Operation. *Technometrics* **4**(4), 441–461 (cit. on p. 41).

Stigler, S. M. (2013). The true title of Bayes's essay. *Statistical Science* **28**(3), 283–288 (cit. on p. 91).

Strickland, D. and Mourou, G. (1985). Compression of amplified chirped optical pulses. *Optics Communications* **55**(6), 447–449 (cit. on p. 14).

Stuart, A., Ord, J. K., and Arnold, S. (2010). Kendall's advanced theory of statistics, Volume 2A, Classical inference and the linear model (cit. on p. 31).

Stuart, A. M. (2010). Inverse problems: a Bayesian perspective. *Acta Numerica* **19**, 451–559 (cit. on p. 93).

Sun, J., Qu, Q., and Wright, J. (2015). When are nonconvex problems not scary? *arXiv preprint arXiv:1510.06096* (cit. on p. 60).

Taylor, C. C. W. (2010). *The Atomists: Leucippus and Democritus: Fragments: A Text and Translation with a Commentary*. University of Toronto Press. 288 pp. (cit. on p. 4).

Timpson, C. G. (2008). Quantum Bayesianism: a study. *Studies in History and Philosophy of Science Part B: Studies in History and Philosophy of Modern Physics* **39**(3), 579–609 (cit. on p. 90).

Toussaint, U. von (2011). Bayesian inference in physics. *Reviews of Modern Physics* **83**(3), 943 (cit. on p. 90).

Tukey, J. W. (1977). *Exploratory data analysis*. Phillipines: Addison-Wesley Publishing Company. 688 pp. (cit. on p. 27).

Vager, Z. (2001). Coulomb explosion imaging of molecules. *Advances In Atomic, Molecular, and Optical Physics, Volume 45*. Ed. by B. Bederson and H. Walther. San Diego: Academic Press, pp. 203–239 (cit. on pp. 1, 13).

Vager, Z., Naaman, R., and Kanter, E. P. (1989). Coulomb explosion imaging of small molecules. *Science* **244**(4903), 426–431. The authors provide an engaging and extensive introduction to Coulomb explosion imaging and motivate the need to image non-classical molecular structures. A highly recommended read. (Cit. on pp. 8–12, 16, 17).

Vager, Z. and Gemmell, D. S. (1976). Polarization induced in a solid by the passage of fast charged particles. *Physical Review Letters* **37**(20), 1352–1354 (cit. on p. 10).

Vager, Z., Gemmell, D. S., and Zabransky, B. J. (1976). Dissociation of fast HeH^+ ions traversing thin foils. *Physical Review A* **14**(2), 638–641 (cit. on p. 10).

VanderPlas, J. (2014). Frequentism and bayesianism: a python-driven primer. *arXiv preprint arXiv:1411.5018* (cit. on p. 90).

Veshapidze, G., Nishide, T., Shiromaru, H., Kobayashi, N., and Mizogawa, T. (2002). A Time- and Position-Sensitive Detector using a Resistive Film Anode Combined with a "Modified Backgammon with Weighted Capacitors" Readout Pad. *Japanese Journal of Applied Physics* **41**(Part 1, No. 2A), 871–875 (cit. on p. 22).

Voigtsberger, J. et al. (2014). Imaging the structure of the trimer systems 4He3 and 3He4He2. *Nature Communications* **5**, 5765 (cit. on p. 4).

Wales, B., Bisson, E., Karimi, R., Kieffer, J.-C., Légaré, F., and Sanderson, J. (2012a). A coincidence detection algorithm for improving detection rates in coulomb explosion imaging. *Nuclear*

Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment **667**, 11–15 (cit. on p. 23).

Wales, B., Motojima, T., Matsumoto, J., Long, Z., Liu, W.-K., Shiromaru, H., and Sanderson, J. (2012b). Multiple ionization and complete fragmentation of OCS by impact with highly charged ions Ar4+ and Ar8+ at 15 keV q^{-1}. *Journal of Physics B: Atomic, Molecular and Optical Physics* **45**(4), 045205 (cit. on pp. 39, 76).

Wales, B. et al. (2014). Coulomb imaging of the concerted and stepwise break up processes of OCS ions in intense femtosecond laser radiation. *Journal of Electron Spectroscopy and Related Phenomena* **195**(2014), 332–336 (cit. on p. 19).

Waltz, R. A., Morales, J. L., Nocedal, J., and Orban, D. (2006). An interior algorithm for nonlinear optimization that combines line search and trust region steps. *Mathematical programming* **107**(3), 391–408 (cit. on p. 65).

Wand, M. P. and Jones, M. C. (1993). Comparison of smoothing parameterizations in bivariate kernel density estimation. *Journal of the American Statistical Association* **88**(422), 520–528 (cit. on p. 32).

Warner, S. (1990). *Modern algebra*. Dover Publications. 832 pp. (cit. on p. 77).

Waskom, M. et al. (2017). *mwaskom/seaborn: v0.8.0 (July 2017)* (cit. on p. 145).

Weinstein, J. A. and Hunt, N. T. (2012). Ultrafast chemical physics: In search of molecular movies. *Nature chemistry* **4**(3), 157–158 (cit. on p. 4).

Wickham, H. (2009). *ggplot2: Elegant Graphics for Data Analysis*. New York: Springer-Verlag. 213 pp. (cit. on p. 145).

Wilson, G. et al. (2014). Best Practices for Scientific Computing. *PLoS Biology* **12**(1). Ed. by J. A. Eisen, e1001745 (cit. on p. 69).

Wright, M. H. (2010). Nelder, Mead, and the other simplex method. *Documenta Mathematica* **7**, 271–276 (cit. on p. 41).

Xu, J., Blaga, C. I., Agostini, P., and DiMauro, L. F. (2016). Time-resolved molecular imaging. *Journal of Physics B: Atomic, Molecular and Optical Physics* **49**(11), 112001 (cit. on p. 1).

Zeller, S. et al. (2016). Imaging the He_2 quantum halo state using a free electron laser. *Proceedings of the National Academy of Sciences* **113**(51), 14651–14655 (cit. on p. 4).

Zewail, A. H. (2000). Femtochemistry: Atomic-scale dynamics of the chemical bond. *The Journal of Physical Chemistry A* **104**(24), 5660–5694 (cit. on p. 4).

A

SUPPLEMENTARY FIGURES

This appendix contains supplementary figures, mainly pertaining to measurements and geometry reconstructions made for pulse lengths longer than 7 fs that seemed rather obstructive placed within chapters 3 to 5 yet may be referenced throughout this thesis.

A.1 MOMENTUM DATA MEASUREMENTS (30 fs TO 200 fs)

A.1.1 *Momentum distributions (30 fs to 200 fs)*

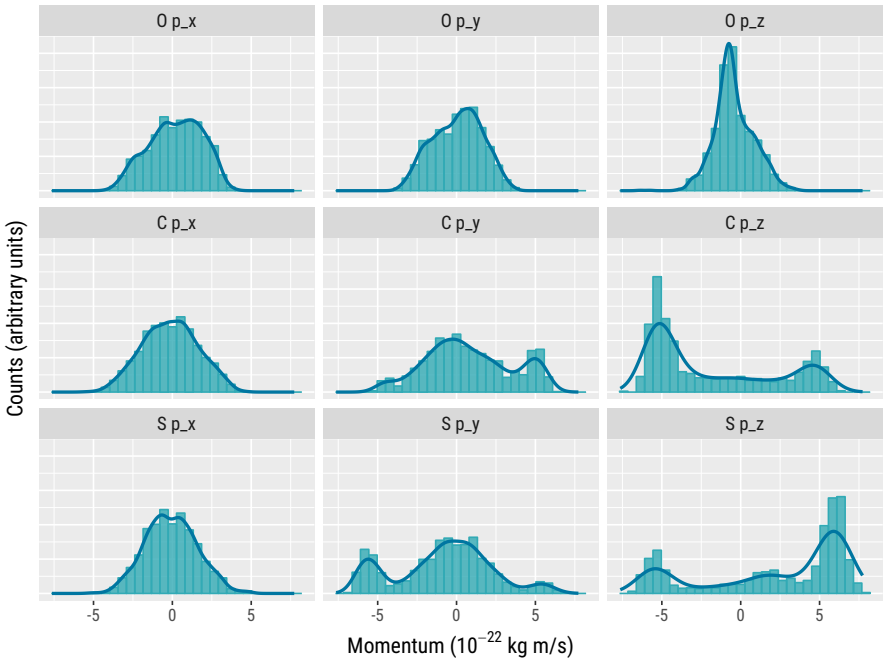

Figure 31: OCS (2,2,2) 30 fs momentum.

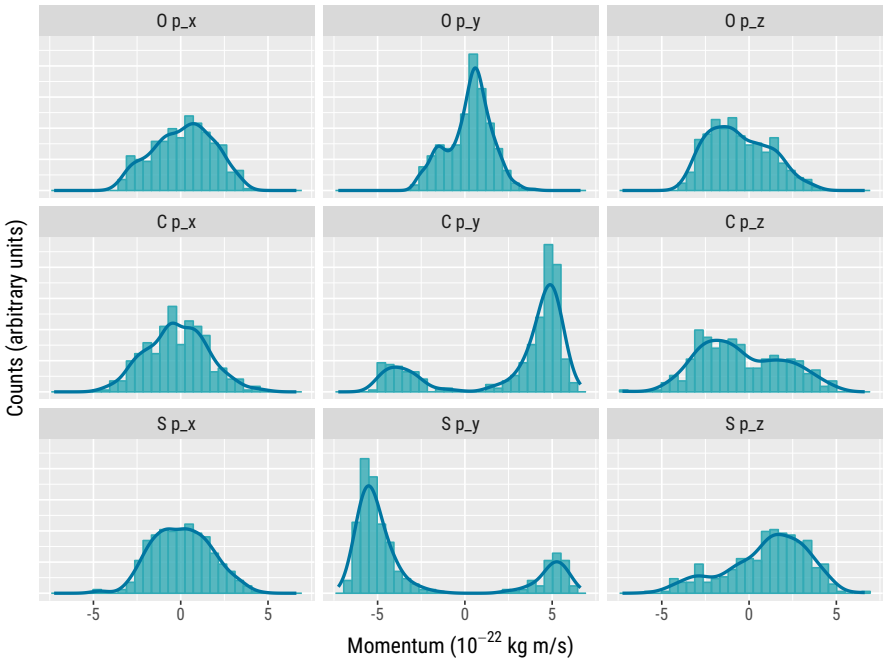

Figure 32: OCS (2,2,2) 60 fs momentum.

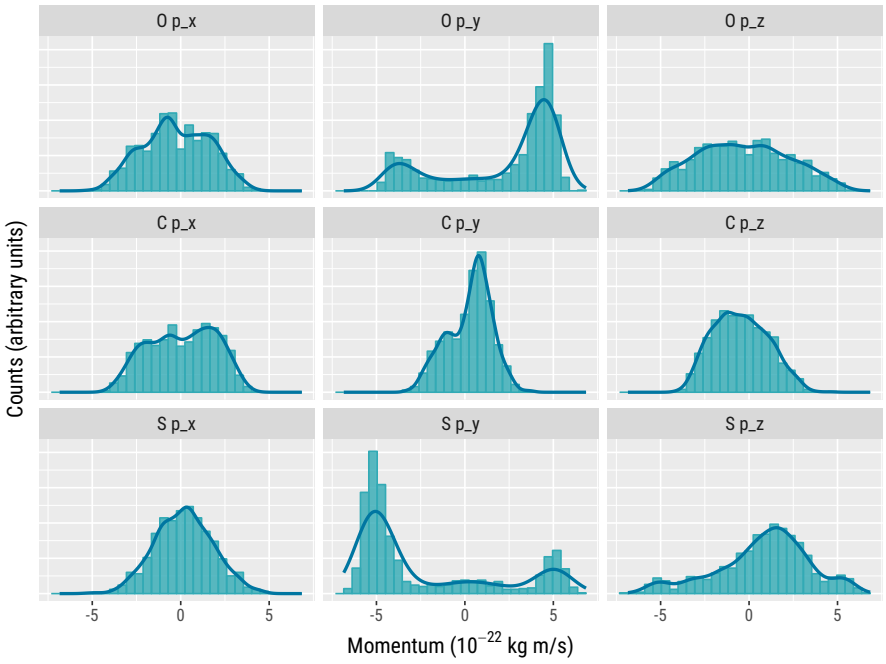

Figure 33: OCS (2,2,2) 100 fs momentum.

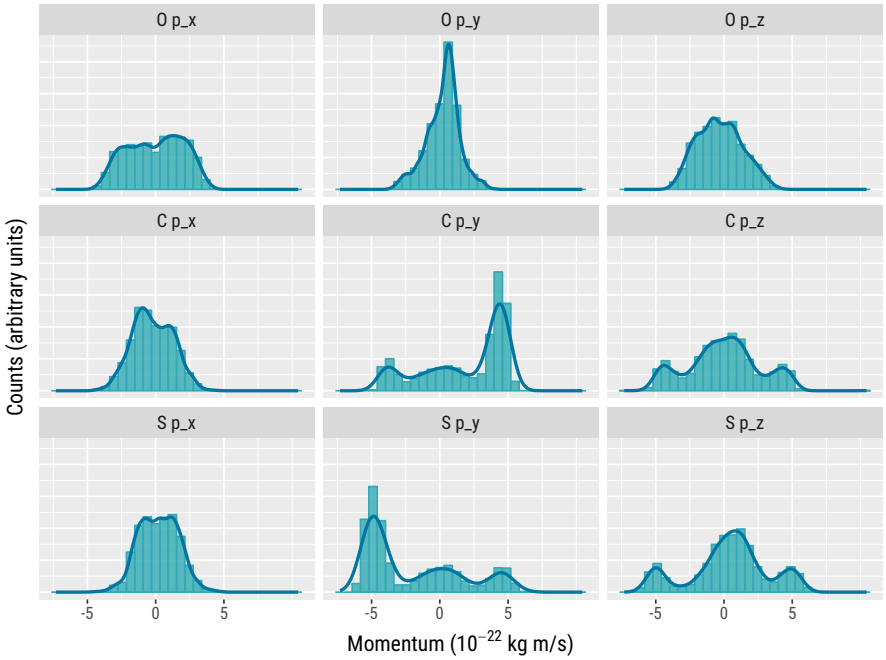

Figure 34: OCS (2,2,2) 200 fs momentum.

A.1.2 *Momentum measurement scatter plot matrices (30 fs to 200 fs)*

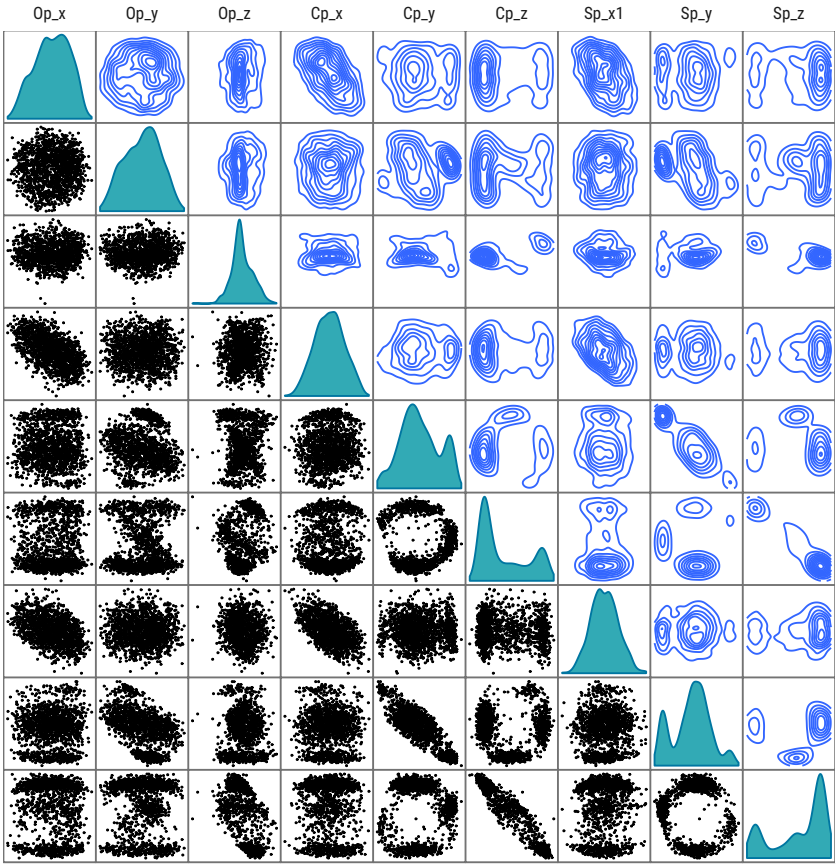

Figure 35: OCS (2,2,2) 30 fs momentum pair plots.

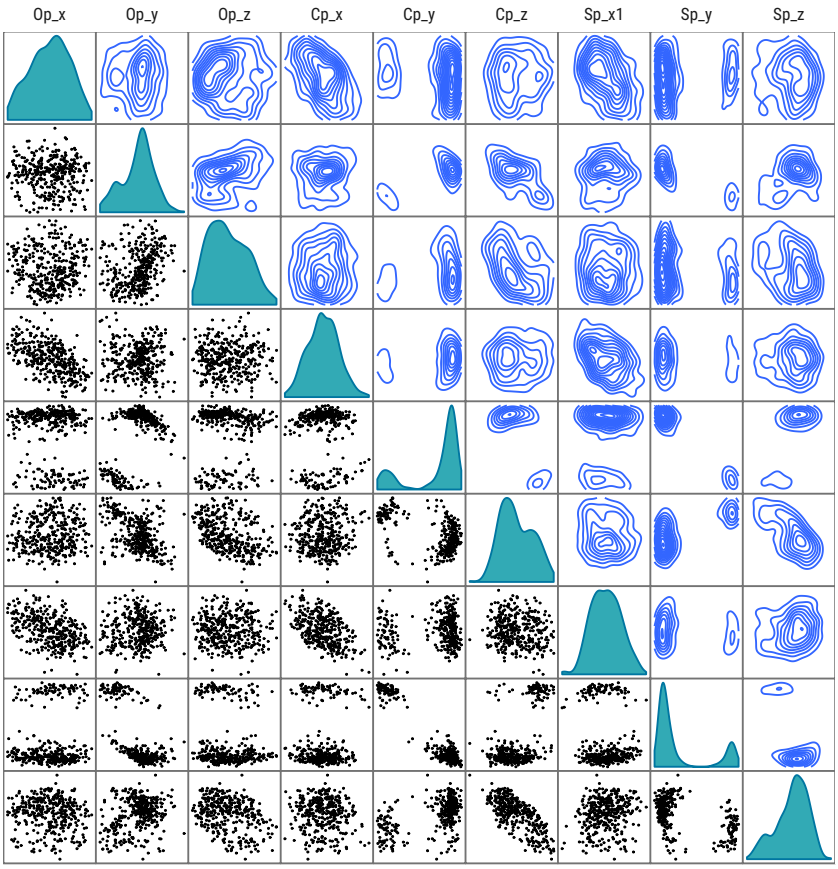

Figure 36: OCS (2,2,2) 60 fs momentum pair plots.

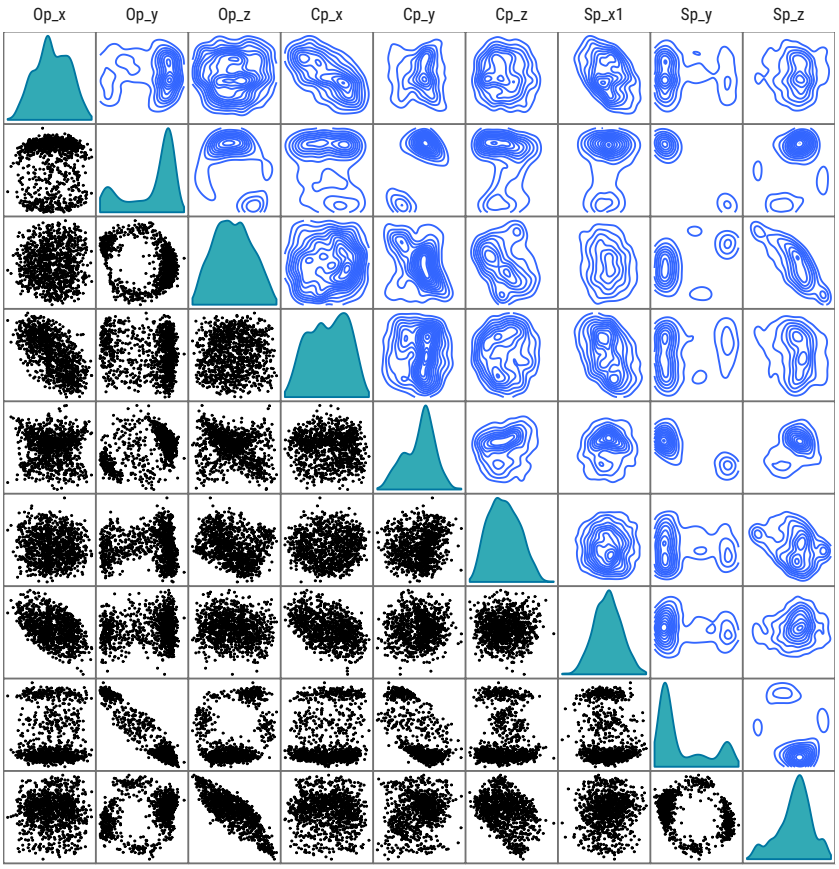

Figure 37: OCS (2,2,2) 100 fs momentum pair plots.

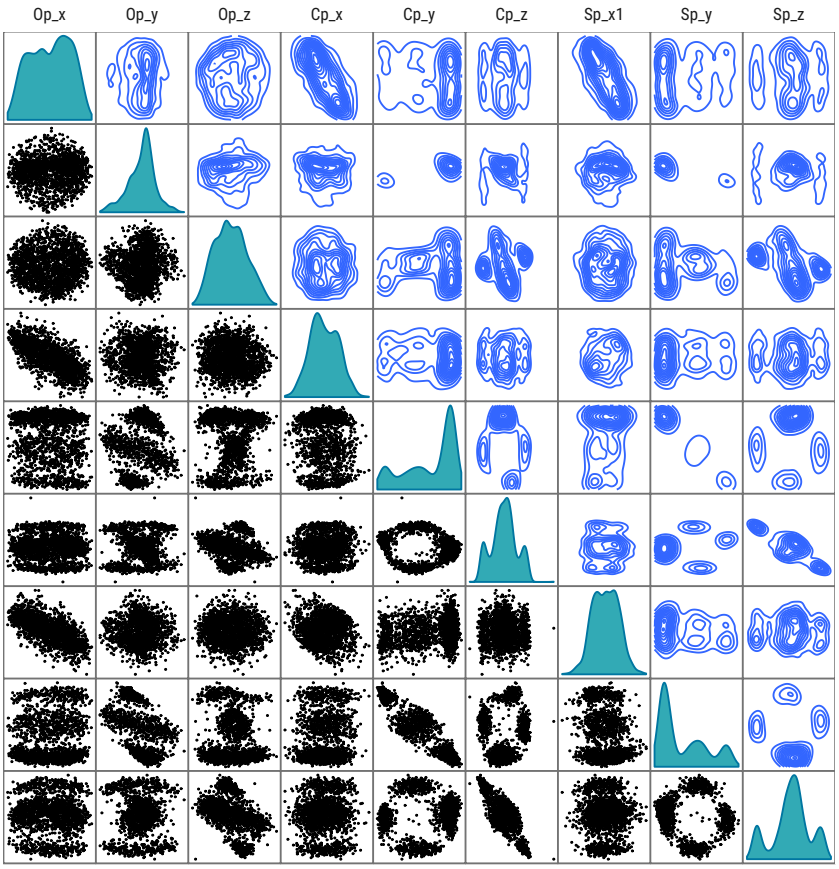

Figure 38: OCS (2,2,2) 200 fs momentum pair plots.

A.2 LOOKUP TABLE GEOMETRY RECONSTRUCTIONS

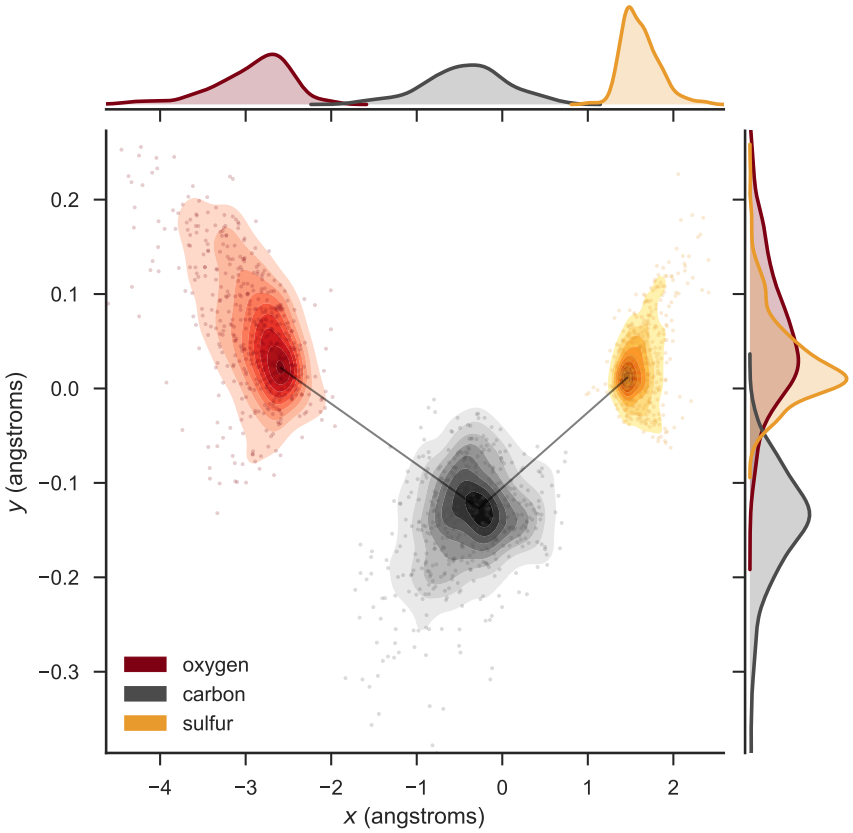

Figure 39: OCS (2,2,2) 30 fs lookup table geometry reconstruction.

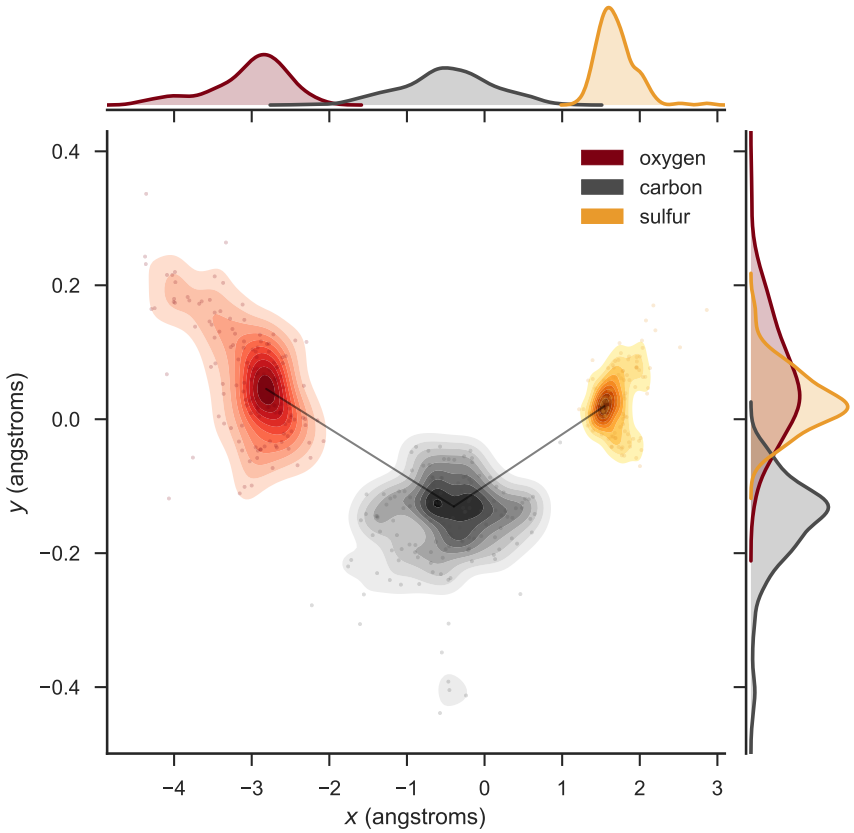

Figure 40: OCS (2,2,2) 60 fs lookup table geometry reconstruction.

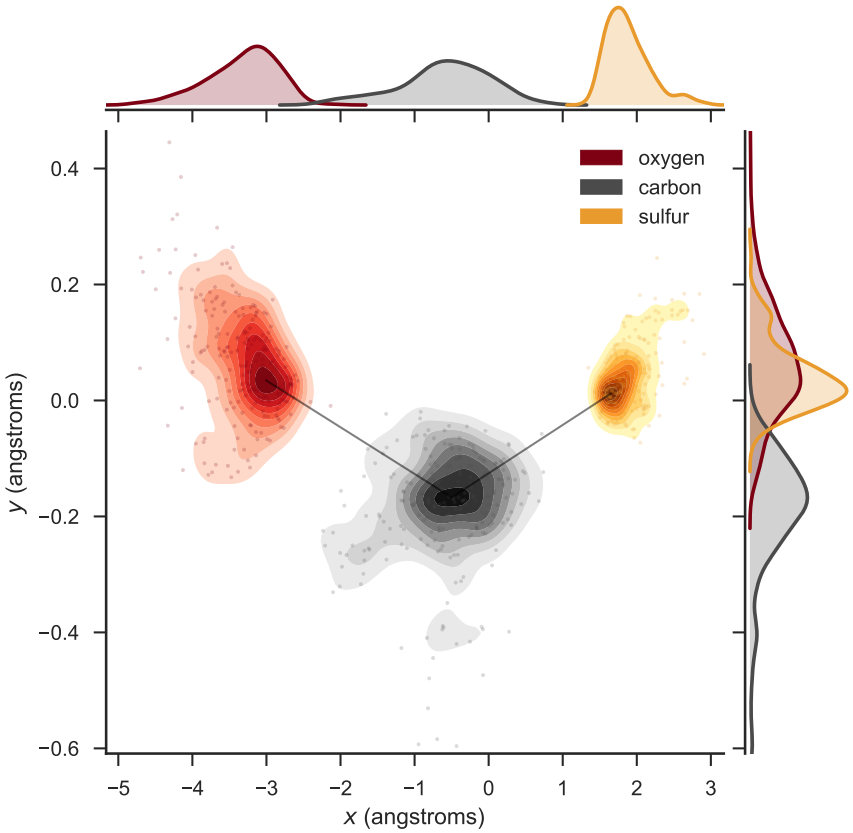

Figure 41: OCS (2,2,2) 100 fs lookup table geometry reconstruction.

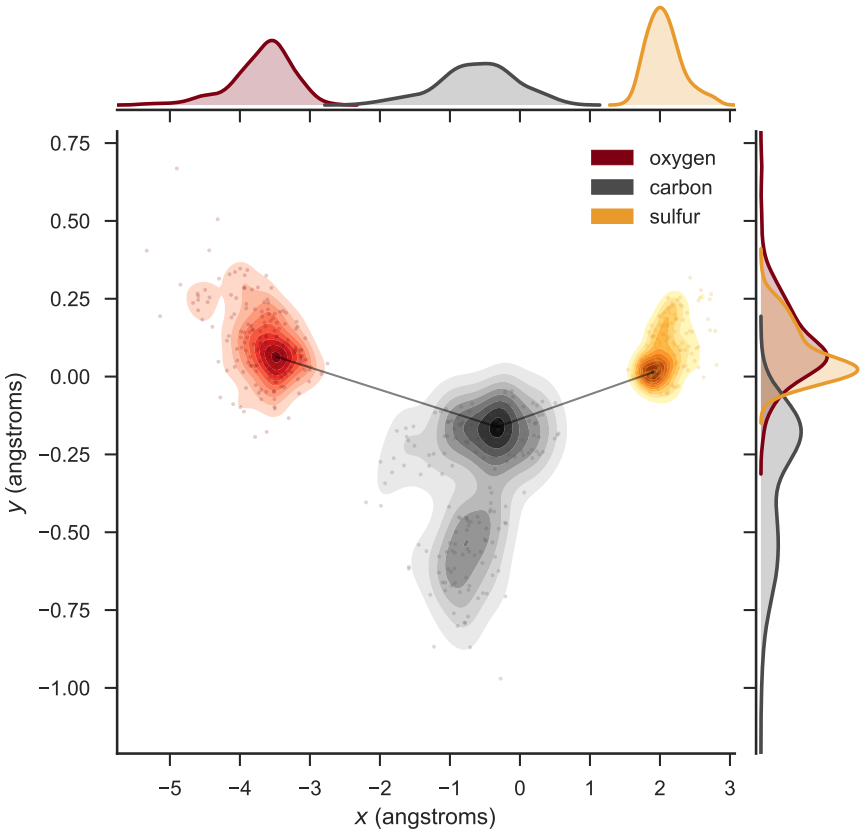

Figure 42: OCS (2,2,2) 200 fs lookup table geometry reconstruction.

A.3 MATHEMATICAL OPTIMIZATION GEOMETRY RECONSTRUCTIONS

A.3.1 *Geometry plots*

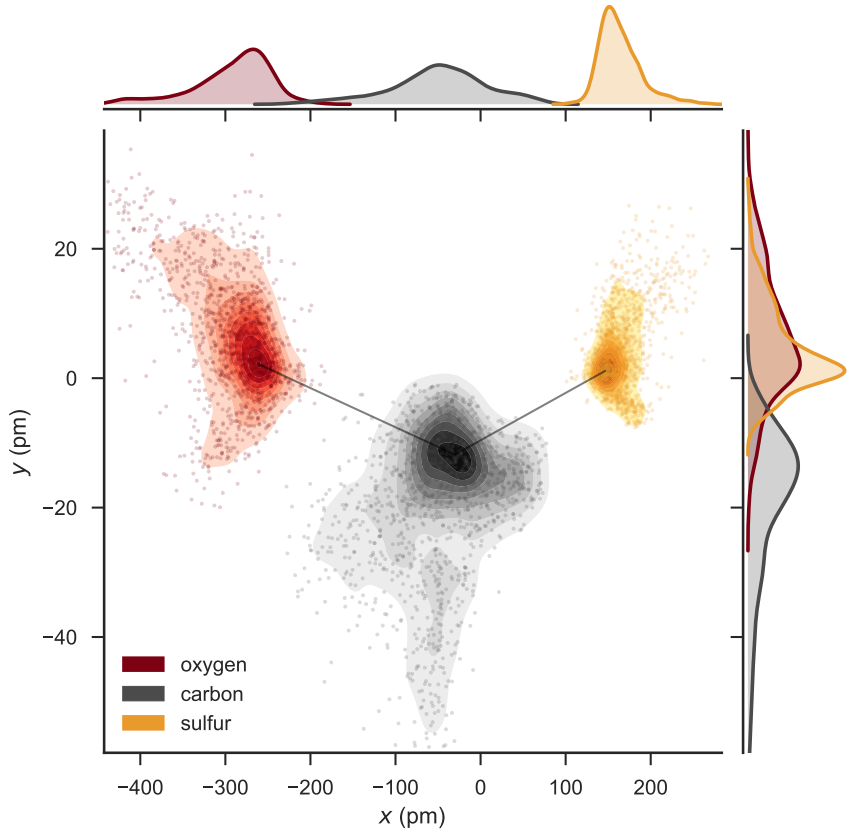

Figure 43: OCS (2,2,2) 30 fs geometry reconstructions using nonlinear constrained optimization.

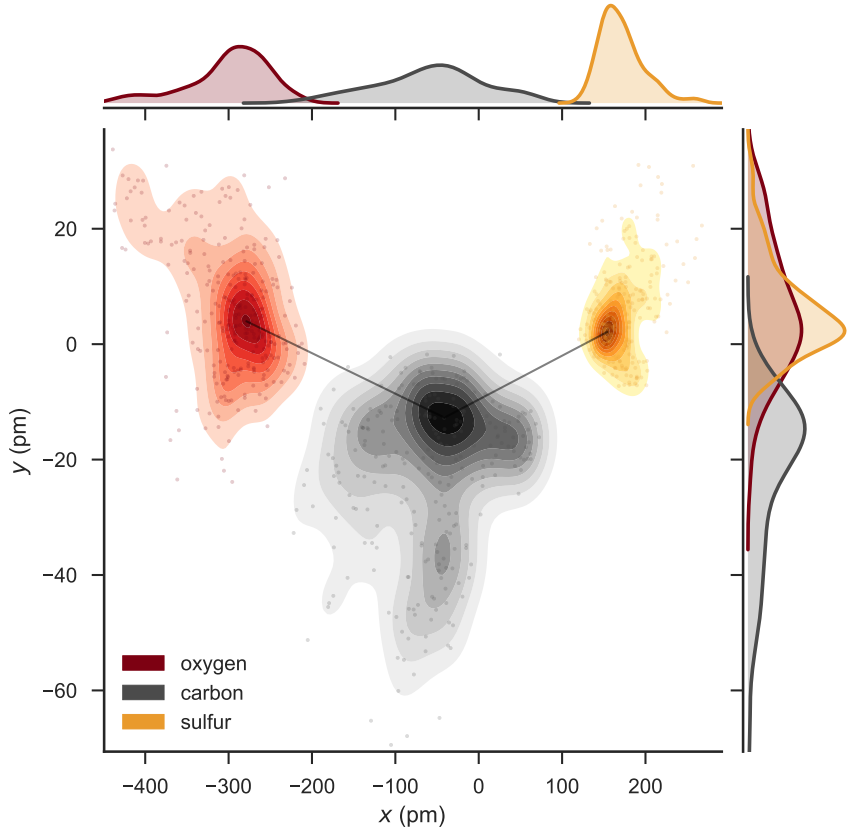

Figure 44: OCS (2,2,2) 60 fs geometry reconstructions using nonlinear constrained optimization.

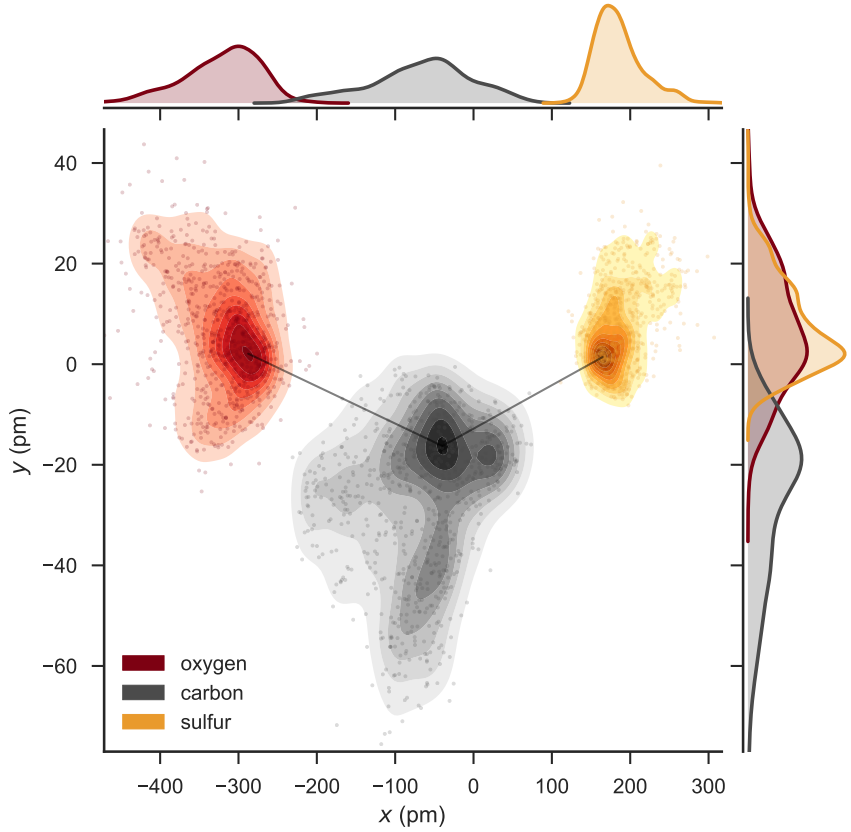

Figure 45: OCS (2,2,2) 100 fs geometry reconstructions using nonlinear constrained optimization.

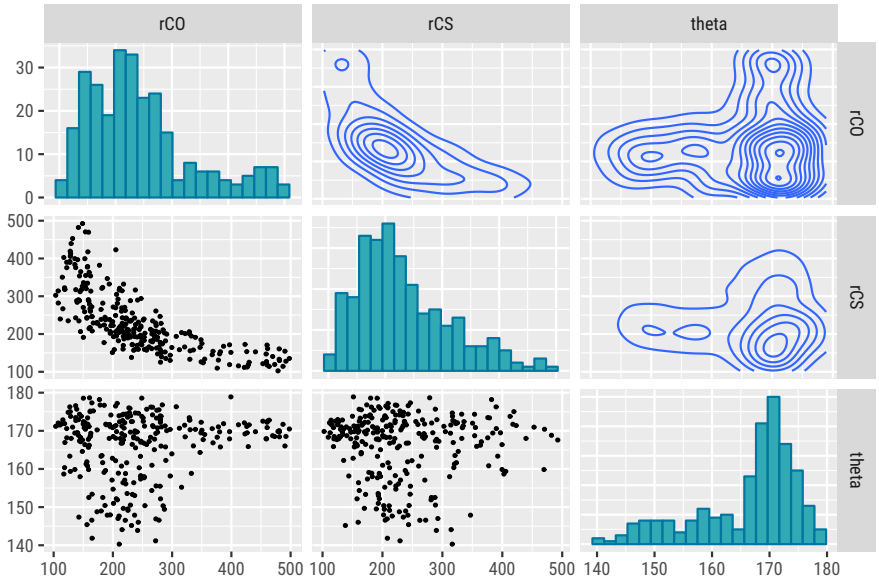

Figure 47: Scatter plot matrices for the OCS (2,2,2) 60 fs geometry reconstructions using nonlinear constrained optimization.

A.3.2 *Scatter plot matrices*

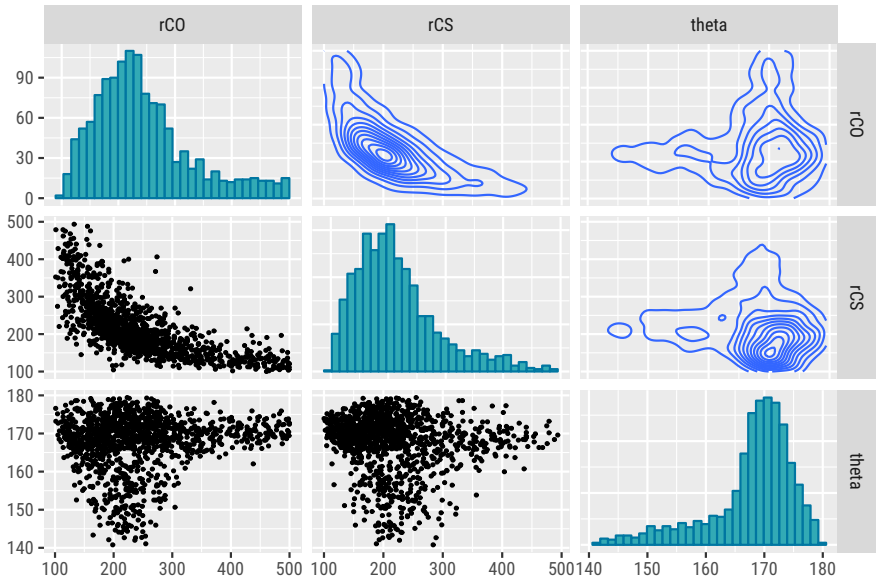

Figure 46: Scatter plot matrices for the OCS (2,2,2) 30 fs geometry reconstructions using nonlinear constrained optimization.

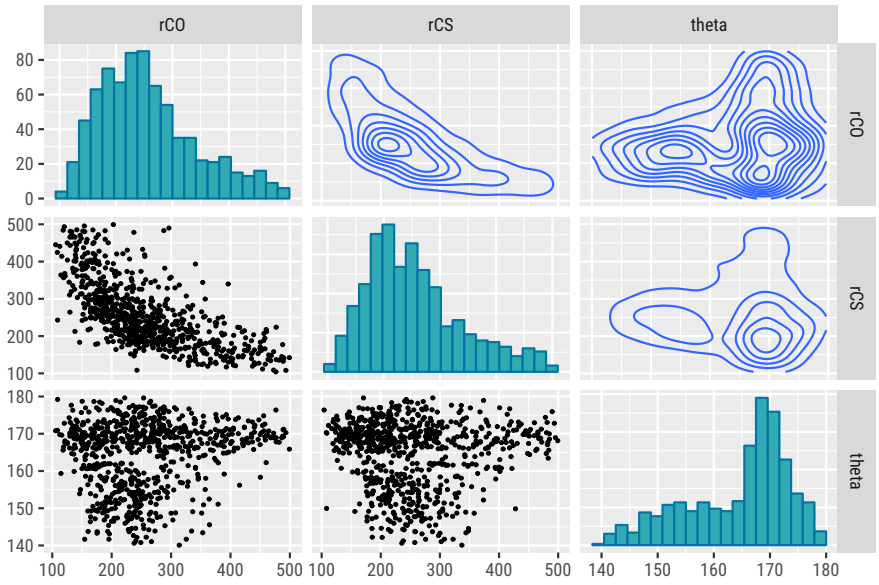

Figure 48: Scatter plot matrices for the OCS (2,2,2) 100 fs geometry reconstructions using nonlinear constrained optimization.

B

ESSENTIAL CODE LISTINGS

This appendix includes the most essential code required to replicate the results presented in this thesis. The rest of the code, including undiscussed extensions and scripts to reproduce the plots, is available on the `ali-ramadhan/msc-thesis-code` GitHub repository. In the case that the code is not available from GitHub for any reason, I should be able to provide it via request by email or otherwise.

```
% hamiltonianDerivative.m:
% Computes the derivatives of the Hamiltonian (Hamilton's equations)
% for a triatomic molecule.
%
% Inputs:
% * time: a 1x2 row vector containing [InitialTime, FinalTime]
% * p   : a 1x18 row vector containing position and momemtum parameters
%         of the three particles, given as
%         [x1 y1 z1 x2 y2 z2 x3 y3 z3 px1 px2 ... pz3].
%         p stands for parameters.
%
% Output:
% * out : an nx19 matrix where each row contains [Time, Position[1x9],
%         Momentum[1x9]] in the same format as p. In practice, only
%         the final row is utilized to evaluate the final conditions
%         of the system, i.e. we are only interested in the asymptotic
%         momentum vectors.
%
% Notes: * All units are SI.
%        * This function is written in the form of a system of
%          first-order ordinary differential equations (ODE's),
%          y' = f(t, y), so that it may be solved using MATLAB's
%          4/5th-order Runge-Kutta ODE solver, ode45. Pretty much all
%          numerical ODE solvers expect the system of ODE's to be
%          described in this form.

function out = hamiltonianDerivative(time, p)

% Physical constants
amu = 1.66053886e-27; % [kg], 1 atomic mass unit
e   = 1.60217646e-19; % [C], 1 elementary charge
k   = 8.987551e9;     % [N m^2 C^-2], electrostatic constant

% Atomic masses and charges
m1 = amu*p(19); m2 = amu*p(20); m3 = amu*p(21);
q1 = e*p(22);   q2 = e*p(23);   q3 = e*p(24);

% Calculate the distance between ions. Note that this quantity does
% not preserve vector direction.
r12 = ((p(1)-p(4))^2 + (p(2)-p(5))^2 + (p(3)-p(6))^2)^0.5; % [m]
r13 = ((p(1)-p(7))^2 + (p(2)-p(8))^2 + (p(3)-p(9))^2)^0.5; % [m]
```

```
r23 = ((p(4)-p(7))^2 + (p(5)-p(8))^2 + (p(6)-p(9))^2)^0.5; % [m]

% pDot is a column vector with components [vx1; vy1; vz1; vx2; ... vz3;
% p'x1; ... p'z3]. These quantities are produced by taking the first
% derivative of the Hamiltonian with respect to the appropriate variable.

pDot = [p(10)./m1; p(11)./m1; p(12)./m1; ...
        p(13)./m2; p(14)./m2; p(15)./m2; ...
        p(16)./m3; p(17)./m3; p(18)./m3; ...

        k*q1*q2*(p(1)-p(4))/r12^3 + k*q1*q3*(p(1)-p(7))/r13^3; ...
        k*q1*q2*(p(2)-p(5))/r12^3 + k*q1*q3*(p(2)-p(8))/r13^3; ...
        k*q1*q2*(p(3)-p(6))/r12^3 + k*q1*q3*(p(3)-p(9))/r13^3; ...

        k*q2*q1*(p(4)-p(1))/r12^3 + k*q2*q3*(p(4)-p(7))/r23^3; ...
        k*q2*q1*(p(5)-p(2))/r12^3 + k*q2*q3*(p(5)-p(8))/r23^3; ...
        k*q2*q1*(p(6)-p(3))/r12^3 + k*q2*q3*(p(6)-p(9))/r23^3; ...

        k*q3*q1*(p(7)-p(1))/r13^3 + k*q3*q2*(p(7)-p(4))/r23^3; ...
        k*q3*q1*(p(8)-p(2))/r13^3 + k*q3*q2*(p(8)-p(5))/r23^3; ...
        k*q3*q1*(p(9)-p(3))/r13^3 + k*q3*q2*(p(9)-p(6))/r23^3];

out = [pDot; p(19:24)];
end
```

```
% simulateMomenta.m:
% Given a list of geometries, simulate a Coulomb explosion for each and
% for each geometry, return the asymptotic momentum vectors for the
% atomic fragments.
%
% Inputs:
% * geometries: nx3 matrix where each row is of the form
%               [r_12 r_23 theta]. r_12 and r_23 should be given in SI
%               units [m] and theta in [deg].
% * masses:     row vector [m1 m2 m3] with the atomic masses in amu.
% * charges:    row vector [q1 q2 q3] with the atomic charges in units
%               of the elementary charge e. So they should be integers.
%
% Output:
% * out: nx12 matrix where each row contains the molecular parameters
%        r_12, r_23, and theta as well as the asymptotic momentum vectors
%        of the three atomic fragments after the Coulomb explosion.
%        [r_12 r_23 theta p_1 p_2 p_3]
%
% Notes: * All units are SI.
%        * parfor loop requires the Parallel Processing Toolbox.

function out = simulateMomenta(geometries, masses, charges, debug)
    nGeometries = size(geometries, 1);

    if debug
    fprintf('# Simulating asymptotic momenta for %d geometries...\n', ...
        nGeometries);
    fprintf('#     Masses = (%.2f, %.2f, %.2f) [amu]\n', ...
        masses(1), masses(2), masses(3));
    fprintf('#     Charges = (%d, %d, %d) [e]\n', ...
```

```
32          charges(1), charges(2), charges(3));
33      end
34
35      out = zeros(nGeometries, 12);
36
37      r_12  = geometries(:, 1);
38      r_23  = geometries(:, 2);
39      theta = geometries(:, 3);
40
41      % Place each first atom to the left of central atom, along the
42      % y-axis.
43      x_1 = -r_12;
44      y_1 = zeros(nGeometries, 1);
45      z_1 = zeros(nGeometries, 1);
46
47      % Place each central atom at the origin.
48      x_2 = zeros(nGeometries, 1);
49      y_2 = zeros(nGeometries, 1);
50      z_2 = zeros(nGeometries, 1);
51
52      % Place each third atom to the right of the central in the +x/+y
53      % quadrant, forming an angle (180-theta) with the x-axis.
54      x_3 = r_23 .* cosd(180 - theta);
55      y_3 = r_23 .* sind(180 - theta);
56      z_3 = zeros(nGeometries, 1);
57
58      % For each geometry, calculate the asymptotic momentum vectors it
59      % would produce in a Coulomb explosion.
60      parfor i = 1:nGeometries
61          g = [x_1(i) y_1(i) z_1(i) x_2(i) y_2(i) z_2(i) x_3(i) y_3(i) ...
62               z_3(i)]; % initial positions
63          p_0 = zeros(1, 9); % initial momentum
64          p = coulombExplode([g p_0], masses, charges);
65          out(i,:) = [r_12(i) r_23(i) theta(i) p(1:3) p(4:6) p(7:9)];
66
67          % Write progress report to console every 100 simulations.
68          % Note: Simulations may not be done in order since loop
69          % iterations are executed in parallel in a nondeterministic order
70          % (parfor loop).
71          if debug && (rem(i, 100) == 0)
72              fprintf('Simulated geometry #%d/%d.\n', i, nGeometries);
73              drawnow('update');
74          end
75      end
76
77      % Extract each atom's momentum into 2D vectors in preparation to
78      % rotate into our convention.
79      p_1 = out(:, 4:5);
80      p_2 = out(:, 7:8);
81      p_3 = out(:, 10:11);
82
83      % Put each momentum into column vector form so we can use matrix
84      % multiplication.
85      p_1 = p_1'; p_2 = p_2'; p_3 = p_3';
86
87      % Calculate the angle between the central atom and the +x-axis then
88      % rotate the three momentum vectors back towards the origin by that
```

```
    % much so that the central atom's momentum vector points along the
    % +x-axis.
    for i = 1:nGeometries
        theta_2x = atan2(p_2(2, i), p_2(1, i));

        % Rotation matrix for rotating points in the xy-plane
        % counterclockwise through an angle -theta_2x about the origin.
        R = [cos(-theta_2x) -sin(-theta_2x); ...
             sin(-theta_2x) cos(-theta_2x);];

        % Rotate each vector.
        p_1(:, i) = R*p_1(:, i);
        p_2(:, i) = R*p_2(:, i);
        p_3(:, i) = R*p_3(:, i);
    end

    % Put everything back into row vector form.
    p_1 = p_1'; p_2 = p_2'; p_3 = p_3';

    % Set the z components (and also y in case of carbon) to 0 so they
    % all have exactly the exact same numerical value rather than 0.0000
    % and -0.0000, etc. This makes data analysis and filtering more
    % convenient as we can just check for equality with 0.
    % Note: In our momentum vector convention for triatomic molecules,
    % these components should all be zero anyways.
    p_1(:, 3) = 0;
    p_2(:, 2) = 0; p_2(:, 3) = 0;
    p_3(:, 3) = 0;

    out = [r_12 r_23 theta p_1 p_2 p_3];
end

% coulombExplode:
% Given the initial positions and momentum vectors of three ions forming
% a triatomic molecule, simulate their Coulomb explosion and return the
% asymptotic momentum vectors of the three ions.
%
% Inputs:
%  * initialConditions: 18-element row vector containing the initial
%        positions and momentum of the atoms, in the form
%        [x1_0 y1_0 ... z3_0 px1_0 py1_0 ... pz3_0].
% * masses:  row vector [m1 m2 m3] with the atomic masses in amu.
% * charges: row vector [q1 q2 q3] with the atomic charges in units
%            of the elementary charge e. So they should be integers.
%
% Output:
%  * out: A 9-element row vector containing the asymptotic momentum
%         vector components for each ion in the form
%         [px1 py1 pz1 ... pz3]

function out = coulombExplode(initialConditions, masses, charges)
    % Set some error tolerances and initial step sizes for the ODE solver
    % to use. See: https://www.mathworks.com/help/matlab/ref/odeset.html
    options = odeset('AbsTol', 1e-27, ...   % absolute error tolerance
                     'RelTol', 1e-6, ...    % relative //
                     'InitialStep', 1e-18); % initial step size

```

```
    % Solve the first-order ODE system described in
    % hamiltonianDerivative.m from t=0 to t=1e-11, by which time the
    % atoms have long attained their asymptotic values. od45 uses
    % adaptive time steps so the intermediate times won't be uniformly
    % distributed.
    [t,y] = ode45('hamiltonianDerivative', [0 1e-11], ...
                  [initialConditions masses charges], options);

    % We have all the intermediate positions and momentum values but we
    % only care about the asymptotic momentum components, so we return
    % them and ignore everything else.
    out = y(size(t, 1), 10:18);
end
```

```
% removeCOMMOtion.m:
% Takes a momentum triple [p_1 p_2 p_3] and returns it in the same order
% but with the center of mass motion removed. That is, we are converting
% the momentum vectors from the lab frame to the center-of-momentum (COM)
% frame (sometimes called the molecular frame).
%
% Inputs:
% * momentum: 9-element row vector containing the momentum triple in
%             the form [px1 py1 ... pz3].
% * masses: 3-element row vector [m1 m2 m3] containing the mass of each
%           atom in atomic mass units [amu].
%
% Output:
% * out: 9-element row vector containing the momentum triple in the
%        form [px1 py1 ... pz3] in the COM frame.
%
% Notes: * This should only need to be done for experimentally measured
%          momentum vectors. Simulated momentum vectors (e.g. from
%          simulateMomenta.m) are already in the COM frame so this should
%          do nothing to them.
% TODO: * Vectorize this function so it converts an entire set of
%         momentum triples. That way the same function can be used for 1
%         or 100 triples.

function out = removeCOMMotion(momentum, masses)
      % Split the momentum triple into X, Y, Z system components.
      p_X = momentum(1:3:7);
      p_Y = momentum(2:3:8);
      p_Z = momentum(3:3:9);

      massSum = sum(masses);

      % Eliminate center of mass motion.
      p_X = p_X - sum(p_X) .* (masses)/massSum;
      p_Y = p_Y - sum(p_Y) .* (masses)/massSum;
      p_Z = p_Z - sum(p_Z) .* (masses)/massSum;

      % Put the vectors back into the original [p1 p2 p3] form.
      momentum(1:9) = reshape([p_X; p_Y; p_Z], 1, 9);

      out = momentum;
end
```

```
% rotateMomentum.m:
% Rotate the asymptotic momentum vectors produced by a Coulomb explosion
% such they lie in the xy-plane with the middle atom's momentum vector
% pointing along the +x-axis.
%
% Input:
% * momenta: 1x9 row vector of momentum components in the form
%            [p2x p2y p2z p1x p1y p1z p3x p3y p3z]
%            WARNING: MIDDLE ATOM FIRST, THEN FIRST, THEN THIRD ATOM!
%
% Output:
% * out: 1x11 row vector containing the momentum triples as well as the
%        theta_v and chi angles in the form
%        [p2x p2y p2z p1x p1y p1z p3x p3y p3z theta_v chi]
%
% Notes: * THIS FUNCTION REQUIRES THE MIDDLE ATOM'S MOMENTUM FIRST, THEN
%          THE FIRST, THEN THIRD! I have no idea why I haven't changed
%          it yet...
%
% TODO: * Vectorize this function so it rotates an entire set of momentum
%         triples. That way the same function can be used for 1 or 100
%         triples.

function out = rotateMomentum (momenta)
    % We want to put the momentum components in the form
    % [px1 py1 pz1 px2 py2 pz2 px3 py3 pz3].
    momenta = reshape(momenta, 3, 3)';

    % If all the vectors are already in the xy-plane, just return them.
    z_components = momenta(:,3);
    if z_components == [0; 0; 0]
        % MATLAB Editor may complain about this equality above but it
        % works for momentum vectors produced by simulateMomenta.m or
        % vectors already rotated using this function.
        % TODO: Assume the zeros may be floating-point and very close to
        % zero.

        theta_v = acos(dot(momenta(2,:), momenta(3,:)) / ...
                       norm(momenta(2,:)) / norm(momenta(3,:)));
        chi = acos(dot(momenta(1,:), momenta(2,:)-momenta(3,:))/...
                  norm(momenta(1,:))/norm(momenta(2,:)-momenta(3,:)));
        out =  [momenta(2,:), momenta(1,:), momenta(3,:), theta_v, chi];
        return
    end

    % First, check that all three momentum vectors form a plane.
    % If this is the case, the determinant of the momentum vectors will
    % be zero (or very close).

    if det(momenta) < 1e-50
        % The normal vector of the plane is the cross product of two of
        % the vectors.
        normal = cross(momenta(1, :), momenta(2, :));

        % Normalise the normal vector!
        normal = normal / norm(normal);
```

```
57
58          % The components of the normal vector (A,B,C) define the plane.
59          % By construction, the plane goes through the origin.  We must
60          % now find the angle this plane makes with the xy-plane, which is
61          % defined by the normal vector normal_xy = [0 0 1]. This angle is
62          % called the dihedral angle.
63          normal_xy = [0 0 1];
64          dihedral = acos(dot(normal, normal_xy)); % [rad]
65
66          % The two planes intersect in a line defined by crossing the two
67          % normal vectors.
68          intersection = cross(normal, normal_xy);
69
70          % if intersection - zeros(1,3) ~= 0 % this is only the case if
71          intersection = intersection / norm(intersection);
72
73          % Insert all of the relevant information into the matrix which
74          % performs the plane rotation.
75          momenta = rotatePlane(momenta, dihedral, intersection);
76
77          theta_v = acos(dot(momenta(2,:), momenta(3,:))/...
78                        norm(momenta(2,:))/norm(momenta(3,:)));
79
80          % Determine phi, the minimum angle between one of the terminal
81          % atoms and the x-axis.
82          % phi = abs(atan2(momenta(2,2), momenta(2,1)));
83
84          % For some reason I couldn't get atan2 to work, so let's go with
85          % this.
86          phi = abs(atan(momenta(2,2)/momenta(2,1)));
87
88          if momenta(2,1) >= 0 && momenta(2,2) >= 0    % first quadrant
89              phi = phi;
90          elseif momenta(2,1) < 0 && momenta(2,2) >= 0 % second quadrant
91              phi = pi - phi;
92          elseif momenta(2,1) < 0 && momenta(2,2) < 0  % third quadrant
93              phi = pi + phi;
94          else % fourth quadrant
95              phi = 2*pi - phi;
96          end
97
98          % Rotate everything (clockwise) through the angle phi
99          M = [cos(-phi), -sin(-phi), 0; ...
100              sin(-phi), cos(-phi),  0; ...
101              0, 0, 1];
102
103         momenta(1,:) = (M*(momenta(1,:)'))';
104         momenta(2,:) = (M*(momenta(2,:)'))';
105         momenta(3,:) = (M*(momenta(3,:)'))';
106
107         % Flip in the y-axis (if necessary) such that the second end
108         % atom sits in the +x half plane.
109         if momenta(3,2) < 0
110             momenta(1:2:3,2) = -momenta(1:2:3,2);
111         end
112
113         if rand(1) > 0.5
```

```
            chi = acos(dot(momenta(1,:), momenta(2,:)-momenta(3,:))/...
                  norm(momenta(1,:))/norm(momenta(2,:)-momenta(3,:)));
        else
            chi = acos(dot(momenta(1,:), momenta(3,:)-momenta(2,:))/...
                  norm(momenta(1,:))/norm(momenta(3,:)-momenta(2,:)));
        end

        out = [momenta(2,:), momenta(1,:), momenta(3,:), theta_v, chi];

    else
        error('Momentum vectors cannot form a plane!')
    end
end

% rotatePlane:
% Rotate the momentum vectors which define the plane Ax + By + Cz = 0 by
% the angle dihedral in the line of intersection defined by the direction
% cosines a, b, and c made with the plane z=0. This makes the momentum
% vectors exist only in the x-y plane while retaining their configuration
% and magnitude. The dihedral angle is in radians.
function out = rotatePlane (momenta, dihedral, intersection)
  lineMagnitude = norm(intersection);
  a = intersection(1) / lineMagnitude;
  b = intersection(2) / lineMagnitude;
  c = intersection(3) / lineMagnitude;

  M = [a^2*(1-cos(dihedral))+cos(dihedral), ...
       a*b*(1-cos(dihedral))-c*sin(dihedral), ...
       a*c*(1-cos(dihedral))+b*sin(dihedral); ...
       a*b*(1-cos(dihedral))+c*sin(dihedral), ...
       b^2*(1-cos(dihedral))+cos(dihedral), ...
       b*c*(1-cos(dihedral))-a*sin(dihedral); ...
       a*c*(1-cos(dihedral))-b*sin(dihedral), ...
       b*c*(1-cos(dihedral))+a*sin(dihedral), ...
       c^2*(1-cos(dihedral))+cos(dihedral)];

  for i = 1:3
    newVector(i,:) = (M*(momenta(i,:)'))';
  end

  out = newVector;
end
```

```
% lookupGeometries.m:
% Takes in a 12-column momentum lookup table `table`, an auxillary lookup
% table `auxtable` for storing results from extra simulations, a list of
% momentum triples `momenta` that you want to find geometries for and an
% error tolerance to aim for.
%
% It will return a matrix bestGeometries with the best matching
% geometries for each momentum triple in momenta. If one could not be
% found, it will return a row of zeros. It will also return an updated
% auxtable with extra entries computed when running extra simulations.
%
% Outline of the lookup table implementation:
% 1. Check the momentum triple against all entries in the coarse lookup
```

```
14  %    table and auxillary lookup table, and find the best matching
15  %    geometry by finding the geometry and momentum triple that minimizes
16  %    the norm squared of the difference of the two momentum vectors.
17  % 2. If the geometry we found isn't good enough, simulate the experiment
18  %    for geometries close to the best one we found from the lookup table
19  %    and check if the best simulated geometry is good enough. If not,
20  %    keep doing this iteratively for smaller regions around the best
21  %    geometry so far until a good enough geometry has been found.
22  % 3. If successive iterations of step 2 don't yield a good enough
23  %    geometry, we look at the top 100 geometries we found from step 1
24  %    and try a different geometry. We try to pick a geometry from a
25  %    different region in the parameter space in case there is more than
26  %    one converging region.
27  % 4. If we have tried converging on three different geometries with no
28  %    luck, just return zeros. We give up.
29  %
30  % Inputs:
31  % * table: nx12 matrix with each row containing geometries and their post
32  %          Coulomb explosion momentum vectors in the form
33  %          [r12 r23 theta p1x p1y ... p3z]
34  % * auxtable: optional mx12 matrix. You could put in an empty matrix or
35  %             an existing momentum lookup table with the same format as
36  %             `table`. lookupGeometries will use this table in addition
37  %             to the coarse lookup table but whenever lookupGeometries
38  %             simulates extra geometries, it will add the simulation to
39  %             the table so it can be used in future calculations. I.e.
40  %             memoization, or dynamically updating the table. It's mostly
41  %             to make sure we don't run the same simulation twice as it's
42  %             always faster to check against a lookup table than it is to
43  %             simulate more data.
44  % * momenta: nx9 matrix
45  % * tolerance: error threshold below which a geometry is considered to be
46  %              a good, or optimal, solution. Generally, using just the
47  %              coarse lookup table, you can get down to errors of ~5e-48,
48  %              which is barely 2 significant figures of accuracy if
49  %              you're looking at angstroms and degrees. 1e-50 is about
50  %              2-3 significant figures, and 1e-52 is probably 3-ish. Of
51  %              course, the lower the desired error tolerance, the longer
52  %              it takes to converge and sometimes it might not be able
53  %              to if it gets stuck in a local minimum, in which case a
54  %              row of zeros is returned.
55  %
56  % Outputs:
57  % * bestGeometries: nx12 matrix
58  % * auxtable: nx12 matrix
59  %
60  % Notes: * All momentum triples should be ordered as OCS in table,
61  %          auxtable and momenta matrices!
62  %        * Lookup table can be generated using simulateMomenta.m. You
63  %          must be careful that the momentum vectors are all in some sort
64  %          of standard configuration or convention. For example, all OCS
65  %          momentum vectors have the p_C vector pointing along the
66  %          +x-axis, the p_S vector always points in the -x,+y direction
67  %          and the p_O vector points in the -x,-y or +x,-y direction,
68  %          it's on either side of the -y-axis. All momentum vectors have
69  %          no z-components in the case of OCS and CO2 as they are
70  %          triatomic and we can rotate the momentum vectors to lie in a
```

```
71  %           plane. simulateMomenta.m will produce momentum vectors with
72  %           this configuration and rotateMomentum will put momentum
73  %           vectors in that configuration if you're processing
74  %           experimentally measure data.
75  %
76  % TODO: * Have the auxillary table update during simulations.
77  %       * If it doesn't converge, maybe include the best geometry it
78  %         found anyways then filter based on fitness?
79  %       * Multiple debug message levels. Does MATLAB have a logger class?
80
81  function [bestGeometries, auxtable] = ...
82      lookupGeometries(table, auxtable, momenta, tolerance)
83
84      simulationSteps = 5;
85      geometriesToTry = 3;
86      coarseR_12Step = 0.05e-10; % inital rStep?
87      coarseR_23Step = 0.05e-10; % initial thetaStep?
88      coarseThetaStep = 0.25;
89      separationThreshold = 10;
90      successiveIterationThreshold = 0.05;
91
92      % For each momentum triple you want to find a geometry for...
93      for i = 1:size(momenta, 1)
94          momentum = momenta(i,:);
95
96          % If we're dealing with real data, we want to remove COM (center
97          % of mass) motion and rotate the momentum into a standard
98          % configuration. If you're dealing with simulated data, that's
99          % already been done so you should comment these lines out.
100
101         % TODO: only call removeCOMMotion is sum of p is not zero and
102         % only call rotateMomentum2 if p vectors aren't in a standard
103         % configuration. Easy to check and no need to comment.
104
105         %momentum = removeCOMMotion(momentum);
106         %momentum = rotateMomentum2(momentum);
107
108         % Search through the coarse and auxillary momentum tables for a
109         % best matching geometry. Don't expect an error of lower than
110         % ~5e-48 from just the coarse lookup table.
111         [bestGeometryCoarse, bestErrorCoarse] = ...
112             lookupGeometry(table, momentum);
113         [bestGeometryAux, bestErrorAux] = ...
114             lookupGeometry(auxtable, momentum);
115
116         % Save the best geometry we found from the two lookup tables.
117         if bestErrorCoarse < bestErrorAux
118             bestGeometry = bestGeometryCoarse;
119             bestError = bestErrorCoarse;
120             fprintf(['[%d] Found coarse geometry (%e, %e, %f) with' ...
121                 'e = %.2e.\n'], i, bestGeometry(1), bestGeometry(2), ...
122                 bestGeometry(3), bestErrorCoarse);
123         else
124             bestGeometry = bestGeometryAux;
125             bestError = bestErrorAux;
126             fprintf(['[%d] Found aux geometry (%e, %e, %f) with' ...
127                 'e = %.2e.\n'], i, bestGeometry(1), bestGeometry(2), ...
```

```
128                 bestGeometry(3), bestErrorAux);
129         end
130
131         % The geometry number we're looking at. It will only go up if we
132         % fail to converge on the first coarse geometry we find.
133         geometryNum = 1;
134
135         topGeometries = top100(table, momentum);
136         for g = geometryNum:size(topGeometries,1)
137             g2 = topGeometries(g,1:3);
138
139             if (g2(1) < 0.95e-10) || (g2(1) > 4.50e-10) || ...
140                     (g2(2) < 0.95e-10) || (g2(2) > 4.50e-10)
141                 continue;
142             end
143
144             bestGeometry = topGeometries(g,:);
145             bestError = norm(bestGeometry(4:12) - momentum)^2;
146             fprintf(['[%d] Starting with geometry %d, (%e, %e, %f)' ...
147                      'with e = %.2e.\n'], i, g, bestGeometry(1), ...
148                      bestGeometry(2), bestGeometry(3), bestError);
149                 break;
150         end
151
152         % If we haven't found a good enough geometry, we'll have to run
153         % simulations for geometries close to what we converged on to see
154         % if we can find a better one. Initially, the lookup table has a
155         % resolution of 0.05 A and 0.25 degrees so we start with those
156         % then the algorithm will make them smaller as it goes.
157         rStep = 0.05e-10;
158         thetaStep = 0.25;
159
160         originalGeometry = bestGeometry;
161         originalError = bestError;
162
163         while bestError > tolerance
164             rStep = rStep / 5;
165             thetaStep = thetaStep / 5;
166
167             simulatedTable = simulateRange(bestGeometry, i, rStep, ...
168                 thetaStep, simulationSteps);
169             [bestGeometrySim, bestErrorSim] = ...
170                 lookupGeometry(simulatedTable, momentum);
171
172             if abs(bestErrorSim - bestError)/bestError < 0.05
173                 % This will happens if we fail to converge onto a
174                 % geometry so even after simulating a deeper region, we
175                 % still haven't found anything better. We look at another
176                 % region if the error is less than 1% better for
177                 % consecutive geometries. We usually expect the error to
178                 % be ~1000% better per iteration for a nice convergence
179                 % point.
180
181                 if simulationSteps == 5 && geometryNum >= 2
182                     fprintf(['[%d] Enhancing momentum lookup table' ...
183                         'simulations from %d to %d!\n'], i, 10^3, 20^3);
184
```

```
185                     bestGeometry = originalGeometry;
186                     bestError = originalError;
187                     simulationSteps = 10;
188                     rStep = 0.05e-10;
189                     thetaStep = 0.25;
190                 else
191                     simulationSteps = 5;
192
193                     if geometryNum == 3
194                         fprintf(['[%d] We have looked at %d '...
195                             'geometries. Giving up now.\n'], ...
196                             i, geometryNum);
197                         bestGeometry = zeros(1,12);
198                         break;
199                     end
200
201                     geometryNum = geometryNum+1;
202                     fprintf(['[%d] Cannot converge to a geometry.' ...
203                         'Attempting to switch to a better' ...
204                         'geometry...\n'], i);
205
206                     topGeometries = top100(table, momentum);
207
208                     space = [topGeometries(:,1:3) ...
209                              kron(topGeometries(:,1), false)];
210                     g1 = bestGeometrySim(1:3);
211
212                     switched = false;
213                     for g = geometryNum:size(topGeometries,1)
214                         g2 = topGeometries(g,1:3);
215
216                         if g2(1) < 0.95e-10 || g2(1) > 4.50e-10 || ...
217                                 g2(2) < 0.95e-10 || g2(2) > 4.50e-10
218                             continue;
219                         end
220
221                         if ~isConnected(space, [g1 false], [g2 false])...
222                                 && separation(g1, g2) >= 10
223                             bestGeometry = topGeometries(g,:);
224                             bestError = ...
225                                 norm(bestGeometry(4:12) - momentum)^2;
226                             fprintf(['[%d] Switched to disconnected' ...
227                                 'geometry %d, (%e, %e, %f) with' ...
228                                 'e = %.2e.\n'], i, g, bestGeometry(1),...
229                                 bestGeometry(2), bestGeometry(3), ...
230                                 bestErrorSim);
231                             switched = true;
232                             break;
233                         end
234                     end
235
236                 if ~switched
237                     for g = geometryNum:size(topGeometries,1)
238                         g2 = topGeometries(g,1:3);
239                         g2(1:2)
240
241                         if g2(1) < 0.95e-10 || g2(1) > 4.50e-10 || ...
```

```
                                g2(2) < 0.95e-10 || g2(2) > 4.50e-10
                            continue;
                        end

                        bestGeometry = topGeometries(g,:);
                        bestError = ...
                            norm(bestGeometry(4:12) - momentum)^2;
                        fprintf(['[%d] Switched to connected geometry'...
                            '%d, (%e, %e, %f) with e = %.2e.\n'], ...
                            i, geometryNum, bestGeometry(1), ...
                            bestGeometry(2), bestGeometry(3), ...
                            bestErrorSim);
                        break;
                    end
                end

                originalGeometry = bestGeometry;
                originalError = bestError;
                rStep = 0.05e-10;
                thetaStep = 0.25;
                end
            elseif bestErrorSim < bestError
                bestGeometry = bestGeometrySim;
                bestError = bestErrorSim;
                fprintf(['[%d] Found simulated geometry (%e, %e, %f)' ...
                    'with e = %.2e.\n'], i, bestGeometry(1), ...
                    bestGeometry(2), bestGeometry(3), bestErrorSim);
            end
        end
        simulationSteps = 5;

        % This will fail if no simulation was needed.
        fprintf(['[%d] Converged to geometry (%e, %e, %f) with' ...
            'e = %.2e.\n\n'], i, bestGeometry(1), bestGeometry(2), ...
            bestGeometry(3), bestErrorSim);

        % Doing 1:12 because sometimes (after enhancing), we end up with
        % a 13th fitness/error column and I'm not sure why.
        bestGeometries(i,:) = bestGeometry(1:12);
    end
end

function [bestGeometry, error_min]  = lookupGeometry(table, momentum)
    error_min = 100;
    j_min = -1;

    for j = 1:size(table,1)
        error = norm(momentum - table(j, 4:12))^2;

        if error < error_min
            error_min = error;
            j_min = j;
        end
    end

    if j_min == -1
        bestGeometry = zeros(12);
```

```
299      else
300          bestGeometry = table(j_min,:);
301      end
302  end
303
304  function out = simulateRange(geometry, i, rStep, thetaStep, ...
305                               simulationSteps)
306      midR12   = geometry(1);
307      midR23   = geometry(2);
308      midTheta = geometry(3);
309
310      nStep = simulationSteps;
311
312      fprintf(['[%d] Simulating data for r_12 = [%e,%e], ' ...
313               'r_23 = [%e,%e], theta = [%f,%f].\n'], ...
314               i, midR12 - nStep*rStep, midR12 + nStep*rStep, ...
315               midR23 - nStep*rStep, midR23 + nStep*rStep, ...
316               midTheta - nStep*thetaStep, midTheta + nStep*thetaStep);
317
318      j = 1;
319      for r_12 = midR12 - nStep*rStep : rStep : midR12 + nStep*rStep
320          for r_23 = midR23 - nStep*rStep : rStep : midR23 + nStep*rStep
321              for theta = midTheta - nStep*thetaStep : thetaStep : midTheta + nStep*thetaStep
322                  out(j,:) = simulateMomentum(r_12, r_23, theta);
323                  j = j+1;
324              end
325          end
326      end
327  end
328
329  function out = top100(table, momentum)
330      % Keeping commented because if you use this function, you already
331      % have COM removed and momentum vectors rotated.
332
333      %momentum = removeCOMMotion(momentum);
334      %momentum = rotateMomentum2(momentum);
335
336      out = zeros(size(table,1), 13);
337      for i = 1:size(table,1)
338          out(i,:) = [table(i,:) norm(momentum - table(i,4:12))^2];
339      end
340
341      out = sortrows(out, 13); % Sort rows by error.
342      out = out(1:100,:);
343  end
344
345  % space is an nx4 matrix with (r_12, r_23, theta, false) data points
346  % corresponding to possible geometries from the lookup table. p1 and p2
347  % are possible geometries. isConnected returns true if p1 and p2 are
348  % within the same region of convergence, that is, they exists a path
349  % between p1 and p2 in the lookup table. Otherwise, it returns false.
350  % isConnected is usually used to check if two points in the top 100
351  % geometries are in the same region or not. If they are not within the
352  % same region, then we can use that point to check if it represents the
353  % geometry we're looking for to avoid searching for the right geometry
354  % simply locally.
355  function out = isConnected(space, p1, p2)
```

```
    [contained,idx] = ...
        ismember(single(p1(1:3)), single(space(:,1:3)), 'rows');

    % If p1 is not in space, return false. If p1 is in space and we've
    % already checked it, then return false. Only continue if p1 is in
    % space and we haven't already checked it.
    if idx == 0
        % disp('not in table');
        out = false;
        return;
    else
        if space(idx,4)
            % disp('already visited');
            out = false;
            return;
        else
            % disp('new')
            space(idx,4) = true;
        end
    end

    if all(single(p1) == single(p2))
        %disp('there!');
        out = true;
        return;
    end

    r_12Step  = 0.05e-10;
    r_23Step  = 0.05e-10;
    thetaStep = 0.25;

    P = p2 - p1; % Direction from geometry p1 to p2.

    possibleSteps = ...
        [dir(P,1)*r_12Step 0                 0;
         0                 dir(P,2)*r_23Step 0;
         0                 0                 dir(P,3)*thetaStep;
         dir(P,1)*r_12Step dir(P,2)*r_23Step 0;
         dir(P,1)*r_12Step 0                 dir(P,3)*thetaStep;
         0                 dir(P,2)*r_23Step dir(P,3)*thetaStep;
         dir(P,1)*r_12Step dir(P,2)*r_23Step dir(P,3)*thetaStep;];

    possibleSteps = unique(possibleSteps, 'rows');
    possibleSteps( ~any(possibleSteps,2), : ) = [];

    for i = 1:size(possibleSteps,1)
        if isConnected(space, p1 + [possibleSteps(i,:) false], p2)
            out = true;
            return;
        end
    end
    out = false;
end

function out = dir(v, d)
    if v(d) == 0
        out = 0;
```

```
    elseif v(d) > 0
        out = 1;
    else
        out = -1;
    end
end

function out = separation(g1, g2)
    r_12Step = 0.05e-10;
    r_23Step = 0.05e-10;
    thetaStep = 0.25;

    delta_g = abs(g2 - g1);
    sep = delta_g(1)/r_12Step + delta_g(2)/r_23Step + ...
          delta_g(3)/thetaStep;
    out = sep;
end
```

```
% multiStartTriatomic.m:
% Given a list of momentum triples `momenta` for a triatomic molecule
% whose atoms have masses `masses` and charges `charges`, reconstruct the
% geometry corresponding to each momentum triple using nonlinear
% constrained optimization, namely through the primal-dual interior-point
% methods provided by the fmincon function. The MultiStart class is used
% to run fmincon using different initial starting points in parallel.
%
% Inputs:
% * momenta: nx9 matrix [p1x p1y ... p3z]
% * masses:  row vector [m1 m2 m3] with the atomic masses in amu.
% * charges: row vector [q1 q2 q3] with the atomic charges in units
%            of the elementary charge e. So they should be integers.
% * fOutFilenamePrefix: string prefix for the files containing the
%       reconstructed geometries. See note below for more information.
% * startingIndex: Not relevant any more, just set to 1.
% * runs: integer number of times to run fmincon for each measured
%         momentum triple.
%
% Output: None.
%
% Notes: * Requires the Optimization, Global Optimization, and Parallel
%          Processing Toolboxes.
%        * This function does not return anything as the reconstructions
%          are done in parallel in a parfor loop. So to avoid multiple
%          threads writing to the same out variable, the results of each
%          reconstruction are saved to a separate file called
%          `fOutFilenamePrefix_Gxxxxx.log` where xxxxx is the index of
%          the geometry in the `momenta` matrix. Yes, there will be
%          problems if you're trying to reconstruct more than 99,999
%          geometries...

function multiStartTriatomic(momenta, masses, charges, ...
                             fOutFilenamePrefix, startingIndex, runs)
    nMomenta = size(momenta, 1);

    parfor i = startingIndex:nMomenta
        fOutFilename = strcat(fOutFilenamePrefix, '_G', ...
```

```
                        sprintf('%05d',i), '.log');
    fOut = fopen(fOutFilename, 'a');

    p = momenta(i, :);
    p = removeCOMMotion(p);
    p = rotateMomentum2(p);

    pGoal = p;
    residualNormObjective = @(g)residualNorm(g, pGoal, masses, ...
                                             charges);

    % These will be our bounds for the multi start algorithm. The
    % algorithm will not search outside of these bounds. They include
    % a wide variety of possible geometries but nothing super
    % unrealistic (e.g. super compressed bonds). Lengths are in [pm]
    % and angles in [deg] because computers don't like numbers that
    % are too small (it seems to be harder to converge when one
    % parameter is ~1e-12 and another ~1e2.) and I thought it would
    % be nice to keep all numbers in the same order of magnitude.
    r_12LowerBound  = 100;
    r_12UpperBound  = 500;
    r_23LowerBound  = 100;
    r_23UpperBound  = 500;
    thetaLowerBound = 140;
    thetaUpperBound = 180;
    lowerBounds = [r_12LowerBound r_23LowerBound thetaLowerBound];
    upperBounds = [r_12UpperBound r_23UpperBound thetaUpperBound];

    % You have to give the multi start algorithm a starting point so
    % I thought might as well give it some middle point. It's not
    % sensitive to the starting point at all which is great. It's
    % going to guess different points anyways.
    % Note: r_12 = 115 pm, r_23 = 156 pm, theta = 175 deg is ground
    % state equilibrium.
    r_12Initial  = 250;
    r_23Initial  = 250;
    thetaInitial = 170;
    initialGeometry = [r_12Initial r_23Initial thetaInitial];

    options = optimoptions('fmincon', ...
                           'Algorithm', 'interior-point', ...
                           'Display', 'off', ...
                           'MaxFunEvals', 3000);

    problem = createOptimProblem('fmincon', ...
        'objective', residualNormObjective, ...
        'lb', lowerBounds, 'ub', upperBounds, ...
        'x0', initialGeometry, ...
        'options', options);

    ms = MultiStart('UseParallel', 'always', ...
                    'Display', 'off', ...
                    'StartPointsToRun', 'bounds');

    [g, fval, exitflag, output, solutions] = run(ms, problem, runs);

    fprintf('G%05d DONE @ %s.\n', i, datestr(now));
```

```
96          fprintf(['G%05d Best geometry found: ' ...
97                   '(%.2f pm, %.2f pm, %.2f deg)' ...
98                   'with log residual norm %.2f and exit flag %d.\n'], ...
99                   i, g(1), g(2), g(3), fval, exitflag);
100         fprintf('G%05d Solver: funcCountNumber:       %d\n', ...
101                 i, output.funcCount);
102         fprintf('G%05d         localSolverIncomplete: %d\n', ...
103                 i, output.localSolverIncomplete);
104         fprintf('G%05d         localSolverNoSolution: %d\n', ...
105                 i, output.localSolverNoSolution);
106         fprintf('G%05d         localSolverSuccess:    %d\n', ...
107                 i, output.localSolverSuccess);
108         fprintf('G%05d         localSolverTotal:      %d\n', ...
109                 i, output.localSolverTotal);
110
111         % We don't always get back 'runs' solutions so we count how many
112         % we found.
113         numSolutionsFound = size([solutions.Fval], 2);
114
115         % We put each distinct solution we found into a row vector and
116         % print them all.
117         mostLikelyGeometries = ...
118             [i*ones(numSolutionsFound, 1), ...
119              reshape([solutions.X], 3, numSolutionsFound)', ...
120              [solutions.Fval]', ...
121              [solutions.Exitflag]'];
122
123         fprintf('G%05d Writing %s.\n', i, fOutFilename);
124         for j = 1:numSolutionsFound
125             fprintf('G%05d G %d\t%3.6f\t%3.6f\t%3.6f\t%2.2f\t%d\n', ...
126                     i, mostLikelyGeometries(j,:));
127             fprintf(fOut, '%d\t%3.6f\t%3.6f\t%3.6f\t%2.2f\t%d\n', ...
128                     mostLikelyGeometries(j,:));
129         end
130
131         fprintf('\n');
132         fprintf(fOut,'\n');
133         fclose(fOut);
134     end
135 end
136
137 % This is our objective or fitness function for the multi start
138 % algorithm. It takes a vector g with our molecule's configuration
139 % (r_12, r_23, theta) and simulates a Coulomb explosion for it. It then
140 % compares the asymptotic momentum produced with the goal momentum we are
141 % attempting to recreate and returns the log10 of the norm of the
142 % difference between the two momentum squared.
143 function rn = residualNorm(g, pGoal, masses, charges)
144     g = [1e-12*g(1) 1e-12*g(2) g(3)];
145     p = simulateMomentum(g, masses, charges);
146     p = p(4:12);
147     rn = log10(norm(pGoal - p)^2);
148 end
149
150 % rotateMomentum2 rotates the momentum to eliminate the z-component. We
151 % change format 123->213 (e.g. OCS->COS) because rotateMomentum expects
152 % 213/COS.
```

```
153 function out = rotateMomentum2(momentum)
154     p_1 = momentum(1:3);
155     p_2 = momentum(4:6);
156     p_3 = momentum(7:9);
157 
158     momentum = [p_1 p_2 p_3];
159     momentum = rotateMomentum(momentum);
160     momentum = [momentum(4:6) momentum(1:3) momentum(7:9)];
161 
162     out = momentum(1:9);
163 end
```

C

COLOPHON

This document was typeset using the typographical look-and-feel `classicthesis` developed by André Miede. The style was inspired by Robert Bringhurst's seminal book on typography "*The Elements of Typographic Style*".

Figures 22, 25, 43–42, and 43–45 were plotted using the Python seaborn package (Waskom et al., 2017) which is built upon the MatplotLib package (Hunter, 2007), and uses SciPy (Jones, Oliphant, and Peterson, 2001–; Oliphant, 2007) to perform the kernel density estimation.

Figures 17, 18, and 21 were produced in R (R Core Team, 2017) using the `multiplot` function by Chang (2012).

Figures 12 and 31–34 were produced in R using the `ggplot2` package (Wickham, 2009) and the `ggthemes` package (Arnold, 2017).

Figures 26, 26 , 35–38, and 46–48 were produced in R using the `ggally` package (Emerson et al., 2013; Schloerke et al., 2017).

Figures 14, 23, 27, 28, 29 were produced in MATLAB. Figures 11 and 24 were produced in OriginPro. Figure 2 was very shamefully created in Microsoft Powerpoint.