Development of Frequency Domain Multidimensional Spectroscopy with Applications in Semiconductor Photophysics

By
Blaise Jonathan Thompson

A dissertation submitted in partial fulfillment of
the requirements for the degree of

Doctor of Philosophy
(Chemistry)

at the
UNIVERSITY OF WISCONSIN - MADISON
2018

Date of final oral examination: April 23, 2018
This dissertation is approved by the following members of the Final Oral Committee:
John C. Wright, Professor, Analytical Chemistry
Randall Goldsmith, Professor, Analytical Chemistry
Tim Bertram, Professor, Analytical Chemistry
Kyoung-Shin Choi, Professor, Materials Chemistry

Contents

List of Figures

List of Tables

LaTeX Font Warning: Font shape ‘OT1/cmss/m/it’ in size ¡10.95¿ not available (Font) Font shape ‘OT1/cmss/m/sl’ tried instead on input line 29.

Chapter 1. Chapter 2.

LaTeX Warning: Citation 'LeeDuckhwan1985a' on page 8 undefined on input line 68.

LaTeX Warning: Citation 'PankoveJacques1975a' on page 9 undefined on input line 91.

LaTeX Warning: Citation 'KumarNardeep2013a' on page 9 undefined on input line 98.

LaTeX Font Warning: Font shape ‘OMS/cmss/m/n’ undefined (Font) using ‘OMS/cmsy/m/n’ instead (Font) for symbol ‘textbullet’ on input line 145.

LaTeX Warning: Citation 'ChengJixin2001a' on page 11 undefined on input line 150.

LaTeX Warning: Citation 'SpencerAustinP2015a' on page 11 undefined on input line 153.

Chapter 3. Chapter 4.

LaTeX Warning: Citation 'CarlsonRogerJ1988a' on page 21 undefined on input line 19.

Chapter 5.

LaTeX Warning: Reference ‘sec:processing$_d$*istbribution*′*onpage*26*undefinedoninputline*30.

Chapter 6. No file dissertation.gls. No file dissertation.acr. No file dissertation.syi.

LaTeX Font Warning: Font shape ‘OT1/cmss/m/sc’ in size ¡10.95¿ not available (Font) Font shape ‘OT1/cmr/m/sc’ tried instead on input line 105.

Package etoc Warning: Please load ‘tocloft’ /before/ ‘etoc’! on input line 107.

LaTeX Font Warning: Some font shapes were not available, defaults substituted.

LaTeX Warning: There were undefined references.

Package biblatex Warning: Please (re)run Biber on the file: (biblatex) dissertation (biblatex) and rerun LaTeX afterwards.

Acknowledgments

To John...

To my colleagues...

To Tyler....

To Claire...

To Sam...

To my parents...

Finally, thank you to all humans who have and continue to undertake the ongoing free and responsible search for truth and meaning. Thanks to free software / free culture / open science advocates who have worked to create and share foundational tools and ideas, often at great personal opportunity cost. Thanks to thought leaders who have shown me what it means to have a good life without fully abandoning moral principles. And thank you to those who bravely speak truth to power. This universe is stranger, more terrible, and more fantastic than we want to believe. We must find ways to describe it's complexity without falling victim to the sometimes-overwhelming power of simple, "useful" narratives.

The explanatory stories that people find compelling are simple; are concrete rather than abstract; assign a larger role to talent, stupidity and intentions than to luck; and focus on a few striking events that happened rather than on the countless events that failed to happen.

The ultimate test of an explanation is whether it would have made the event predictable in advance.

Paradoxically, it is easier to construct a coherent story when you know little, when there are fewer pieces to fit into the puzzle. Our comforting conviction that the world makes sense rests on a secure foundation: our almost unlimited ability to ignore our ignorance.

– Daniel Kahneman [1]

Abstract

Chapter 1

Introduction

1.1 Coherent Multidimensional Spectroscopy

CMDS, coherent multidimensional spectroscopy

1.2 The CMDS Instrument

From an instrumental perspective, MR-CMDS is a problem of calibration and coordination. Within the Wright Group, each of our two main instruments are composed of roughly ten actively moving component hardwares. Many of these components are purchased directly from vendors such as SpectraPhysics, National Instruments, Horiba, Thorlabs, and Newport. Others are created or heavily modified by graduate students. The Wright Group has always maintained custom acquisition software packages which control the complex, many-stepped dance that these components must perform to acquire MR-CMDS spectra.

1.3 Scientific Software

When I joined the Wright Group, I saw that acquisition software was a real barrier to experimental progress and flexibility. Graduate students had ideas for instrumental enhancements that were infeasible because of the challenge of incorporating the new components into the existing software ecosystem. At the same time, students were spending much of their time in lab repeatedly calibrating optical parametric amplifiers by hand, a process that sometimes took days. I chose to spend a significant portion of my graduate career focusing on solving these problems through software development. At first, I focused on improving the existing LabVIEW code. Eventually, I developed a vision for a deeply modular acquisition software that could not be practically created with LabVIEW. Using Python and Qt, I created a brand new acquisition software PyCMDS: built from the ground up to fundamentally solve historical challenges in the Group. PyCMDS offers a modular hardware model that can "re-configure" itself to flexibly control a variety of component hardware configurations. This has enabled graduate students to add and remove hardware whenever necessary, without worrying about a heavy additional programming burden. PyCMDS is now used to drive both MR-CMDS instruments in the Group, allowing for easy sharing of component hardware and lessening the total amount of software that the Group needs to maintain. Besides being more flexible, PyCMDS solves a number of other problems. It offers fully automated strategies for calibrating component hardwares, making calibration less arduous and more reproducible. It offers more fine-grained control of data acquisition and timing, enabling more complex algorithms to quickly acquire artifact-free results. In conjunction with other algorithmic and hardware improvements that I have made, PyCMDS has decreased acquisition times by up to two orders of magnitude. A companion software, WrightTools (which I also created), solves some of the processing and representation challenges of multidimensional data.

Part I

Background

Chapter 2

Spectroscopy

A hundred years ago, Auguste Comte, . . . a great philosopher, said that humans will never be able to visit the stars, that we will never know what stars are made out of, that that's the one thing that science will never ever understand, because they're so far away. And then, just a few years later, scientists took starlight, ran it through a prism, looked at the rainbow coming from the starlight, and said: "Hydrogen!" Just a few years after this very rational, very reasonable, very scientific prediction was made, that we'll never know what stars are made of.

– Michio Kaku

In this chapter I lay out the foundations of spectroscopy.

2.1 Light

2.2 Light-Matter Interaction

Spectroscopic experiments all derive from the interaction of light and matter. Many material properties can be deduced by measuring the nature of this interaction.

Nonlinear spectroscopy relies upon higher-order terms in the light-matter interaction. In a generic system, each term is roughly ten times smaller than the last.

2.2.1 Representations

Many strategies have been introduced for diagrammatically representing the interaction of multiple electric fields in an experiment.

Circle Diagrams

Double-sided Feynman Diagrams

WMEL Diagrams

So-called wave mixing energy level (WMEL) diagrams are the most familiar way of representing spectroscopy for Wright group members. WMEL diagrams were first proposed by Lee and Albrecht in an appendix to their seminal work *A Unified View of Raman, Resonance Raman, and Fluorescence Spectroscopy* [**LeeDuckhwan1985a**]. WMEL diagrams are drawn using the following rules.

1. The energy ladder is represented with horizontal lines - solid for real states and dashed for virtual states.

2. Individual electric field interactions are represented as vertical arrows. The arrows span the distance between the initial and final state in the energy ladder.
3. The time ordering of the interactions is represented by the ordering of arrows, from left to right.
4. Ket-side interactions are represented with solid arrows.
5. Bra-side interactions are represented with dashed arrows.
6. Output is represented as a solid wavy line.

Mukamel Diagrams

2.3 Linear Spectroscopy

2.3.1 Reflectivity

This derivation adapted from *Optical Processes in Semiconductors* by Jacques I. Pankove [**PankoveJacques1975a**]. For normal incidence, the reflection coefficient is

$$R = \frac{(n-1)^2 + k^2}{(n+1)^2 + k^2} \tag{2.1}$$

Further derivation adapted from [**KumarNardeep2013a**]. To extend reflectivity to a differential measurement

2.4 Coherent Multidimensional Spectroscopy

multiresonant coherent multidimensional spectroscopy

2.4.1 Three Wave

2.4.2 Four Wave

Fluorescence

Raman

2.4.3 Five Wave

2.4.4 Six Wave

multiple population-period transient spectroscopy (MUPPETS)

2.5 Strategies for CMDS

2.5.1 Homodyne vs. Heterodyne Detection

Two kinds of spectroscopies: 1) heterodyne 2) homodyne. Heterodyne techniques may be self heterodyne or explicitly heterodyned with a local oscillator.

In all heterodyne spectroscopies, signal goes as N. In all homodyne spectroscopies, signal goes as N^2. This literally means that homodyne signals go as the square of heterodyne signals, which is what we mean when we say that homodyne signals are intensity level and heterodyne signals are amplitude level.

Transient absorption, TA

2.5.2 Frequency vs. Time Domain

Time domain techniques become more and more difficult when large frequency bandwidths are needed. With very short, broad pulses:

- Non-resonant signal becomes brighter relative to resonant signal
- Pulse distortions become important.

This epi-CARS paper might have some useful discussion of non-resonant vs resonant for shorter and shorter pulses [**ChengJixin2001a**].

An excellent discussion of pulse distortion phenomena in broadband time-domain experiments was published by **SpencerAustinP2015a**.

Another idea in defense of frequency domain is for the case of power studies. Since time-domain pulses in-fact possess all colors in them they cannot be trusted as much at perturbative fluence. See that paper that Natalia presented...

2.5.3 Triply Electronically Enhanced Spectroscopy

Triply Electronically Enhanced (TrEE) spectroscopy has become the workhorse homodyne-detected 4WM experiment in the Wright Group.

2.5.4 Transient Absorbance Spectroscopy

Transient absorption (TA)

Quantitative TA

Transient absorbance (TA) spectroscopy is a self-heterodyned technique. Through chopping you can measure nonlinearities quantitatively much easier than with homodyne detected (or explicitly heterodyned) experiments.

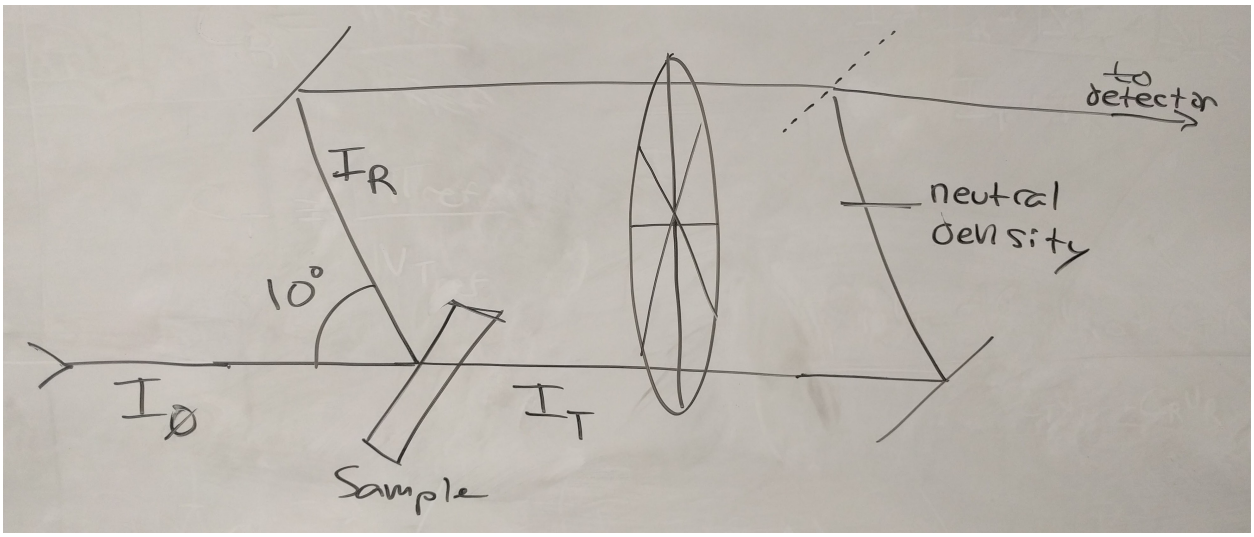

Figure 2.1: CAPTION TODO

Figure 2.1 diagrams the TA measurement for a generic sample. Here I show measurement of both the reflected and transmitted probe beam . . . not important in opaque (pyrite) or non-reflective (quantum dot) samples . . .

Typically one attempts to calculate the change in absorbance ΔA . . .

$$\begin{aligned}
\Delta A &= A_{\text{on}} - A_{\text{off}} && (2.2)\\
&= -\log_{10}\left(\frac{I_{\text{T}} + I_{\text{R}} + I_{\Delta\text{T}} + I_{\Delta\text{R}}}{I_0}\right) + \log\left(\frac{I_{\text{T}} + I_{\text{R}}}{I_0}\right) && (2.3)\\
&= -\left(\log_{10}(I_{\text{T}} + I_{\text{R}} + I_{\Delta\text{T}} + I_{\Delta\text{R}}) - \log_{10}(I_0)\right) + \left(\log_{10}(I_{\text{T}} + I_{\text{R}}) - \log_{10}(I_0)\right) && (2.4)\\
&= -\left(\log_{10}(I_{\text{T}} + I_{\text{R}} + I_{\Delta\text{T}} + I_{\Delta\text{R}}) - \log_{10}(I_{\text{T}} + I_{\text{R}})\right) && (2.5)\\
&= -\log_{10}\left(\frac{I_{\text{T}} + I_{\text{R}} + I_{\Delta\text{T}} + I_{\Delta\text{R}}}{I_{\text{T}} + I_{\text{R}}}\right) && (2.6)
\end{aligned}$$

Equation 2.6 simplifies beautifully if reflectivity is negligible . . .

Now I define a variable for each experimental measurable:

V_{T}	voltage recorded from transmitted beam, without pump
V_{R}	voltage recorded from reflected beam, without pump
$V_{\Delta\text{T}}$	change in voltage recorded from transmitted beam due to pump
$V_{\Delta\text{R}}$	change in voltage recorded from reflected beam due to pump

We will need to calibrate using a sample with a known transmisivity and reflectivity constant:

$V_{\text{T, ref}}$	voltage recorded from transmitted beam, without pump
$V_{\text{R, ref}}$	voltage recorded from reflected beam, without pump
$\mathcal{T}_{\text{ref}}$	transmissivity
$\mathcal{R}_{\text{ref}}$	reflectivity

Define two new proportionality constants...

$$C_{\mathrm{T}} \equiv \frac{\mathcal{T}}{V_{\mathrm{T}}} \tag{2.7}$$

$$C_{\mathrm{R}} \equiv \frac{\mathcal{R}}{V_{\mathrm{R}}} \tag{2.8}$$

These are explicitly calibrated (as a function of probe color) prior to the experiment using the calibration sample.

Given the eight experimental measurables (V_{T}, V_{R}, $V_{\Delta\mathrm{T}}$, $V_{\Delta\mathrm{R}}$, $V_{\mathrm{T,\,ref}}$, $V_{\mathrm{R,\,ref}}$, $\mathcal{T}_{\mathrm{ref}}$, $\mathcal{R}_{\mathrm{ref}}$) I can express all of the intensities in Equation 2.6 in terms of I_0.

$$C_{\mathrm{T}} = \frac{\mathcal{T}_{\mathrm{ref}}}{V_{\mathrm{T,\,ref}}} \tag{2.9}$$

$$C_{\mathrm{R}} = \frac{\mathcal{R}_{\mathrm{ref}}}{V_{\mathrm{R,\,ref}}} \tag{2.10}$$

$$I_{\mathrm{T}} = I_0 C_{\mathrm{T}} V_{\mathrm{T}} \tag{2.11}$$

$$I_{\mathrm{R}} = I_0 C_{\mathrm{R}} V_{\mathrm{R}} \tag{2.12}$$

$$I_{\Delta\mathrm{T}} = I_0 C_{\mathrm{T}} V_{\Delta\mathrm{T}} \tag{2.13}$$

$$I_{\Delta\mathrm{R}} = I_0 C_{\mathrm{R}} V_{\Delta\mathrm{R}} \tag{2.14}$$

Wonderfully, the I_0 cancels when plugged back in to Equation 2.6, leaving a final expression for ΔA that only depends on my eight measurables.

$$\Delta A = -\log_{10}\left(\frac{C_{\mathrm{T}}(V_{\mathrm{T}} + V_{\Delta\mathrm{T}}) + C_{\mathrm{R}}(V_{\mathrm{R}} + V_{\Delta\mathrm{R}})}{C_{\mathrm{T}} V_{\mathrm{T}} + C_{\mathrm{R}} V_{\mathrm{R}}}\right) \tag{2.15}$$

2.5.5 Cross Polarized TrEE

2.5.6 Pump-TrEE-Probe

Pump TrEE probe (PTP).

2.6 Instrumental Response Function

The instrumental response function (IRF) is a classic concept in analytical science. Defining IRF becomes complex with instruments as complex as these, but it is still useful to attempt.

It is particularly useful to define bandwidth.

2.6.1 Time Domain

I will use four wave mixing to extract the time-domain pulse-width. I use a driven signal *e.g.* near infrared carbon tetrachloride response. I'll homodyne-detect the output. In my experiment I'm moving pulse 1 against pulses 2 and 3 (which are coincident).

The driven polarization, P, goes as the product of my input pulse *intensities*:

$$P(T) = I_1(t - T) \times I_2(t) \times I_3(t) \tag{2.16}$$

In our experiment we are convolving I_1 with $I_2 \times I_3$. Each pulse has an *intensity-level* width, σ_1, σ_2, and σ_3. $I_2 \times I_3$ is itself a Gaussian, and

$$\sigma_{I_2 I_3} = \ldots \tag{2.17}$$

$$= \sqrt{\frac{\sigma_2^2 \sigma_3^2}{\sigma_2^2 + \sigma_3^2}}. \tag{2.18}$$

The width of the polarization (across T) is therefore

$$\begin{aligned} \sigma_P &= \sqrt{\sigma_1^2 + \sigma_{I_2 I_3}^2} && (2.19) \\ &= \ldots && (2.20) \\ &= \sqrt{\frac{\sigma_1^2 + \sigma_2^2 \sigma_3^2}{\sigma_1^2 + \sigma_2^2}}. && (2.21) \end{aligned}$$

I assume that all of the pulses have the same width. I_1, I_2, and I_3 are identical Gaussian functions with FWHM σ. In this case, Equation 2.21 simplifies to

$$\begin{aligned} \sigma_P &= \sqrt{\frac{\sigma^2 + \sigma^2 \sigma^2}{\sigma^2 + \sigma^2}} && (2.22) \\ &= \ldots && (2.23) \\ &= \sigma \sqrt{\frac{3}{2}} && (2.24) \end{aligned}$$

Finally, since we measure σ_P and wish to extract σ:

$$\sigma = \sigma_P \sqrt{\frac{2}{3}} \quad (2.25)$$

Again, all of these widths are on the *intensity* level.

2.6.2 Frequency Domain

We can directly measure σ (the width on the intensity-level) in the frequency domain using a spectrometer. A tune test contains this information.

2.6.3 Time-Bandwidth Product

For a Gaussian, approximately 0.441

Chapter 3

Materials

"Kroemer's Lemma of Proven Ignorance": If, in discussing a semiconductor problem, you cannot draw an Energy Band Diagram, this shows that you don't know what you are talking about, If you can draw one, but don't, then your audience won't know what you are talking about.

Chapter 4

Software

The following guidelines are to be used in the documentation of all software developed in the Wright group for the IBM 9000 computer. These rules have arisen as a necessary consequence of the group's programming philosophy of writing software in the form of units which can be readily shared among a number of programmers. The approach outlined here should help to avoid some of the confusion otherwise produced by several persons simultaniously developing and modifying shared software.

– Roger Carlson, "Software Development Guidelines" (1988) [**CarlsonRogerJ1988a**]

Cutting-edge science increasingly relies on custom software. In their 2008 survey, Hannay et al. [2] demonstrated just how important software is to the modern scientist.

1. 84.3% of surveyed scientists state that developing scientific software is important or very important for their own research.
2. 91.2% of surveyed scientists state that using scientific software is important or very important for their own research.
3. On average, scientists spend approximately 40% of their work time using scientific software.
4. On average, scientists spend approximately 30% of their work time developing scientific software.

Despite the importance of software to science and scientists, most scientists are not familiar with basic software engineering concepts. This is in part due to the their general lack of formal training in programming and software development. Hannay et al. [2] found that over 90% of scientists learn software development through 'informal self study'. Indeed, I myself have never been formally trained in software development.

Software development in a scientific context poses unique challenges. Many traditional software development paradigms demand an upfront articulation of goals and requirements. This allows the developers to carefully design their software, even before a single line of code is written. In her seminal 2005 case study Segal [3] describes a collaboration between a team of researchers and a contracted team of software engineers. Ultimately

Part II

Development

Chapter 5

Processing

From a data science perspective, CMDS has several unique challenges:

1. Dimensionality of datasets can typically be greater than two, complicating **representation**.
2. Shape and dimensionality change...
3. Data can be large (over one million points).

I have designed a software package that directly addresses these issues.

WrightTools is a software package at the heart of all work in the Wright Group.

WrightTools is written in Python, and endeavors to have a "pythonic", explicit and "natural" application programming interface (API). To use WrightTools, simply import:

```
>>> import WrightTools as wt
>>> wt.__version__
3.0.0
```
(5.1)

I'll discuss more about how exactly WrightTools packaging, distribution, and instillation works in **??**.

We can use the builtin Python function `dir` to interrogate the contents of the WrightTools package.

```
>>> dir(wt)
['Collection',
 'Data',
 '__branch__',
 '__builtins__',
 '__cached__',
 '__doc__',
 '__file__',
 '__loader__',
 '__name__',
 '__package__',
 '__path__',
 '__spec__',
 '__version__',
 '__wt5_version__',
 '_dataset',
 '_group',
 '_open',
 '_sys',
 'artists',
 'collection',
 'data',
 'diagrams',
 'exceptions',
 'kit',
 'open',
 'units']
```

(5.2)

Many of these are dunder (double underscore) attributes—Python internals that are not normally used directly. The ten attributes that do not start with underscore are the public API that users of WrightTools typically use. Within the public API are two classes, `Collection` & `Data`, which are the two main classes in the WrightTools object model. `Data` stores spectra directly as multidimensional arrays, and `Collection` stores *groups* of data objects (and other collection objects) in a hierarchical way for internal organization purposes.

5.1 Data object model

WrightTools uses a programming strategy called object oriented programming (OOP).

It contains a central data "container" that is capable of storing all of the information about each multidimensional (or one-dimensional) spectra: the `Data` class. It also defines a `Collection` class

that contains data objects, collection objects, and other pieces of metadata in a hierarchical structure. Let's first discuss pythonData.

All spectra are stored within WrightTools as multidimensional arrays. Arrays are containers that store many instances of the same data type, typically numerical datatypes. These arrays have some `shape`, `size`, and `size`. In the context of WrightTools, they can contain floats, integers, complex numbers and NaNs.

The `Data` class contains everything that is needed to define a single spectra from a single experiment (or simulation). To do this, each data object contains several multidimensional arrays (typically 2 to 50 arrays, depending on the kind of data). There are two kinds of arrays, instances of `Variable` and `Channel`. Variables are coordinate arrays that define the position of each pixel in the multidimensional spectrum, and channels are each a particular kind of signal within that spectrum. Typical variables might be `[w1, w2, w3, d1, d2]`, and typical channels `[pmt, pyro1, pyro2, pyro3]`.

5.2 Artists

5.2.1 Colormaps

5.2.2 Interpolation

5.3 Fitting

5.4 Distribution and licensing

5.5 Future directions

Chapter 6

Acquistion

In the Wright Group, PyCMDS replaces the old acquisition softwares 'ps control', written by Kent Meyer and 'Control for Lots of Research in Spectroscopy' written by Schuyler Kain.

PyCMDS directly addresses the hardware during experiments.

6.1 Overview

PyCMDS has, through software improvements alone, dramatically lessened scan times...

- → simultaneous motor motion
- → digital signal processing
- → ideal axis positions 6.2.1

6.2 Future directions

6.2.1 Ideal Axis Positions

Frequency domain multidimensional spectroscopy is a time-intensive process. A typical pixel takes between one-half second and three seconds to acquire. Depending on the exact hardware being scanned and signal being detected, this time may be mostly due to hardware motion or signal collection. Due to the curse of dimensionality, a typical three-dimensional CMDS experiment contains roughly 100,000 pixels. CMDS hardware is transiently-reliable, so speeding up experiments is a crucial component of unlocking ever larger dimensionalities and higher resolutions.

One obvious way to decrease the scan-time is to take fewer pixels. Traditionally, multidimensional scans are done with linearly arranged points in each axis—this is the simplest configuration to program into the acquisition software. Because signal features are often sparse or slowly varying (especially so in high-dimensional scans) linear stepping means that *most of the collected pixels* are duplicates or simply noise. A more intelligent choice of axis points can capture the same nonlinear spectrum in a fraction of the total pixel count.

An ideal distribution of pixels is linearized in *signal*, not coordinate. This means that every signal level (think of a contour in the N-dimensional case) has roughly the same number of pixels defining it. If some generic multidimensional signal goes between 0 and 1, one would want roughly 10% of the pixels to be between 0.9 and 1.0, 10% between 0.8 and 0.9 and so on. If the signal is sparse in the space explored (imagine a narrow two-dimensional Lorentzian in the center of a large 2D-Frequency scan) this would place the majority of the pixels near the narrow peak feature(s), with only a few of them defining the large (in axis space) low-signal floor. In contrast linear stepping would allocate the vast majority of the pixels in the low-signal 0.0 to 0.1 region, with only a few being used to capture the narrow peak feature. Of course, linearizing pixels in signal requires prior expectations about the shape of the multidimensional signal—linear stepping is still an appropriate choice for low-resolution "survey" scans.

CMDS scans often posses correlated features in the multidimensional space. In order to capture such features as cheaply as possible, one would want to define regions of increased pixel density along the correlated (diagonal) lineshape. As a concession to reasonable simplicity, our acquisition software (PyCMDS) assumes that all scans constitute a regular array with-respect-to the scanned axes. We can acquire arbitrary points along each axis, but not for the multidimensional scan. This means that we cannot achieve strictly ideal pixel distributions for arbitrary datasets. Still, we can do much better than linear spacing.

Almost all CMDS lineshapes (in frequency and delay) can be described using just a few lineshape functions:

1. exponential
2. Gaussian
3. Lorentzian
4. bimolecular

Exponential and bimolecular dynamics fall out of simple first and second-order kinetics (I will ignore higher-order kinetics here). Gaussians come from our Gaussian pulse envelopes or from normally-distributed inhomogeneous broadening. The measured line-shapes are actually convolutions of the above. I will ignore the convolution except for a few illustrative special cases. More exotic lineshapes are possible in CMDS—quantum beating and breathing modes, for example—I will also ignore these.

Derivations of the ideal pixel positions for each of these lineshapes appear below.

6.2.2 Exponential

Simple exponential decays are typically used to describe population and coherence-level dynamics in CMDS. For some generic exponential signal S with time constant τ,

$$S(t) = \mathrm{e}^{-\frac{t}{\tau}}. \tag{6.1}$$

We can write the conjugate equation to 6.1, asking "what t do I need to get a cerain signal level?":

$$\log(S) = -\frac{t}{\tau} \tag{6.2}$$

$$t = -(S). \tag{6.3}$$

So to step linearly in t, my step size has to go as $-\tau \log(S)$.

We want to go linearly in signal, meaning that we want to divide S into even sections. If S goes from 0 to 1 and we choose to acquire N points,

$$t_n = -\tau \log\left(\frac{n}{N}\right). \tag{6.4}$$

Note that t_n starts at long times and approaches zero delay. So the first t_1 is the smallest signal and t_N is the largest.

Now we can start to consider realistic cases, like where τ is not quite known and where some other longer dynamics persist (manifested as a static offset). Since these values are not separable in a general

system, I'll keep S normalized between 0 and 1.

$$S = (1-c)\,\mathrm{e}^{-\frac{t}{\tau_{\text{actual}}}} + c \tag{6.5}$$

$$S_n = (1-c)\,\mathrm{e}^{-\frac{-\tau_{\text{step}} \log\left(\frac{n}{N}\right)}{\tau_{\text{actual}}}} + c \tag{6.6}$$

$$S_n = (1-c)\,\mathrm{e}^{-\frac{\tau_{\text{step}}}{\tau_{\text{actual}}} \log\left(\frac{N}{n}\right)} + c \tag{6.7}$$

$$S_n = (1-c)\left(\frac{N}{n}\right)^{-\frac{\tau_{\text{step}}}{\tau_{\text{actual}}}} + c \tag{6.8}$$

$$S_n = (1-c)\left(\frac{n}{N}\right)^{\frac{\tau_{\text{step}}}{\tau_{\text{actual}}}} + c \tag{6.9}$$

[p!]

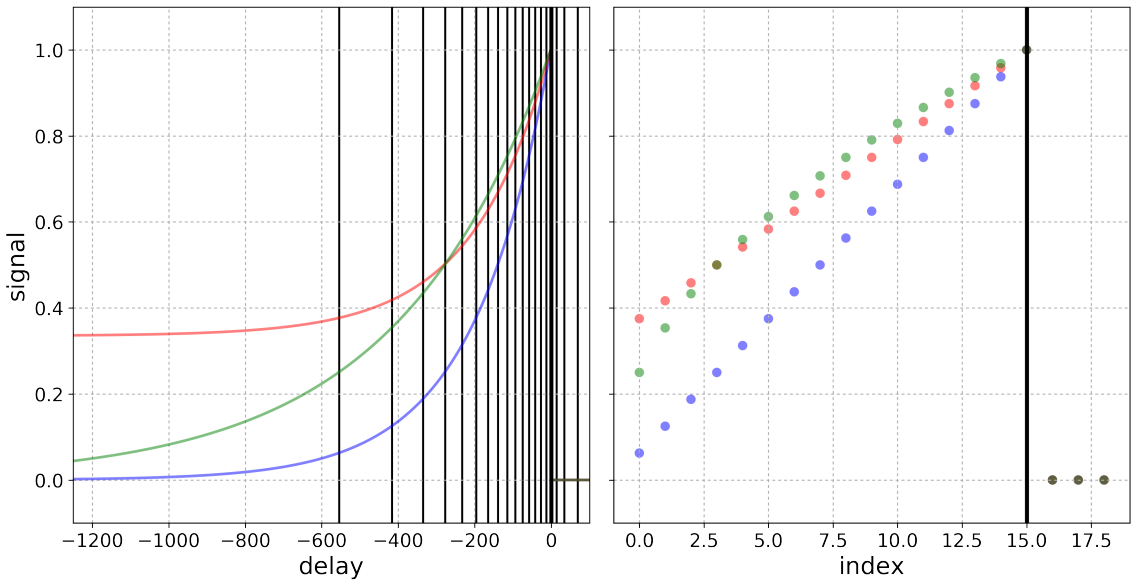

Figure 6.1: TODO

Gaussian

Lorentzian

Bimolecular

Part III

Applications

Part IV

Appendix

Bibliography

[1] Daniel Kahneman. *Thinking, Fast and Slow*. Farrar, Straus and Giroux, 2013.

[2] Jo Erskine Hannay, Carolyn MacLeod, Janice Singer, Hans Petter Langtangen, Dietmar Pfahl, and Greg Wilson. "How do scientists develop and use scientific software?" In: *2009 ICSE Workshop on Software Engineering for Computational Science and Engineering*. Institute of Electrical and Electronics Engineers (IEEE), May 2009. DOI: 10.1109/secse.2009.5069155.

[3] Judith Segal. "When Software Engineers Met Research Scientists: A Case Study". In: *Empirical Software Engineering* 10.4 (Oct. 2005), pp. 517–536. DOI: 10.1007/s10664-005-3865-y.