Report on determination of spin Hamiltonian parameters from electron paramagnetic resonance spectra using numerical non-linear optimisation methods

Craig Eldershaw
Department of Mathematics,
The University of Queensland,
Queensland 4072, Australia.
ce@maths.uq.oz.au

Semester I, 1996

1Copyright © 1996, Craig Eldershaw
Abstract

In physical-chemistry, Electron Paramagnetic Resonance (EPR) is used to determine information about the structure of paramagnetic substances. To aid in this process, computer simulations can be performed to confirm that the physical characteristics of the system have been correctly determined. In the past the characteristics of substances have been arrived at by a combination of intuition and human trial-and-error. However with recent progress in simulation speed, it has now become feasible to use computer-based optimisation to find these them.

This report details how, when the characteristics of a substance are represented by a set of parameters to a Hamiltonian equation, the extraction of these parameters can then be expressed as a non-linear optimisation problem. Having done this, the choice and implementation details of suitable algorithms is addressed. Several difficulties were encountered when applying the resulting methods to real problems. A number of solutions to these difficulties are investigated.
3.1.1 Non-linear least squares method (LSQ)  .............. 13
3.1.2 Perturbation methods  ................................. 13

3.2 Chosen methods  ........................................... 14
3.2.1 Hooke and Jeeve’s method  ............................. 15
3.2.2 Spendley, Hext and Himsworth’s method ............ 16
3.2.3 Simulated annealing  ................................... 19
3.2.4 Simplex combined with simulated annealing ...... 21

3.3 Scaling/sensitivity—the $h_i$s .................................... 22

3.4 Rejecting extreme/invalid points  ............................ 22

4 Implementation details and Results 24
4.1 SOPHE and parallelism  ..................................... 26
4.2 Merging resonances  ......................................... 28
4.2.1 Overweighting of peaks ................................... 31
4.2.2 Peak/trough detection  ................................. 33
4.2.3 Fourier smoothing  ....................................... 34
4.2.4 Shortcomings in the peak detection method ...... 35
4.2.5 Results of overweighting  ............................... 36
4.3 Comparison of Fourier transforms  ......................... 36

5 Future directions 40

6 Conclusion 42

7 Acknowledgments 43
Chapter 1

Electron paramagnetic resonance spectroscopy

1.1 Electron paramagnetic resonance

Electron paramagnetic resonance (EPR) and electron spin resonance (ESR) are alternative names for the same phenomenon. [6, 22] EPR deals with the resonant absorption of microwave radiation by a paramagnetic substance within a static magnetic field.

All chemical structures consist of atoms which are bound into molecules and have electrons in some kind of “orbit” around them. Now all electrons have a magnetic moment of inertia associated with them. This can have one of two states: “up” or “down”. Most electrons in molecules are paired. Due to the Pauli exclusion principle, each electron in a pair must have opposite spin, giving a net magnetic moment of zero. However in a subset of substances, there exist unpaired electrons. These substances are known as paramagnetic substances.

Each of the unpaired electrons has an energy level associated with it. In the absence of a magnetic field, all unpaired electrons have the same energy—this is known as degeneracy. However when subjected to a stationary magnetic field, unpaired electrons become separated into two different energy levels—one for the up state, and one for the down. The stronger the
magnetic field strength, the greater the energy difference between the two levels.

If the paramagnetic substance being analysed is also irradiated with electromagnetic radiation (usually in the form of microwaves), then, under some conditions, some of this radiation may be absorbed. The absorption occurs because one of the unpaired electrons in the lower energy state can absorb a quantum (or packet) of light to allow it to be raised to the higher energy state. However in accordance with quantum mechanics, this transition can only take place when the energy of the incoming quantum of electromagnetic radiation exactly matches the difference between the two energy states.

Magnetic moments are not restricted to just the behavioural properties of electrons. The nuclei within the substances have magnetic moments of their own. These interact with those of the unpaired electrons causing the two energy levels to split into several more. Thus not just one, but several transitions exist, giving rise to a complicated absorption spectrum.

In the experimental situation, a continuous wave EPR spectrometer irradiates a chemical sample with microwave radiation of a fixed frequency. It uses a detector to record how much of the incident radiation is in fact absorbed by the substance. The measurement is repeated over a range of different magnetic field strengths. What is actually emitted by the spectrometer is the first derivative of the absorption as a function of magnetic field strength.

1.2 Uses of EPR

So it can be seen that the EPR spectrum generated by an experiment reflects the physical characteristics of the sample substance. If these characteristics could be extracted from the spectrum then they could be put to use. Firstly, it can allow physical-chemists to determine more information about the magnetic properties/structure of a known substance. Secondly, it can be used as an aid in the identification of unknown substances. Thus the information has both academic and commercial benefits.
Figure 1.1: This figure shows the increase in the separation of the energy levels as the magnetic field intensity is increased. Only when this difference exactly matches the energy of the incoming photons can the electrons change state. For this transition to take place, some of the incident radiation is absorbed, as shown on the first of the two graphs. What is actually seen in the experiment is the derivative of this, shown in the second graph.

1.3 Computer simulation of spectra

The Hamiltonian mentioned in the title is a mathematical operator which also reflects the magnetic characteristics of the substance. For an isolated paramagnetic centre a general spin Hamiltonian is

$$\mathcal{H} = \hat{S}D\hat{S} + \beta \vec{B}g\hat{S} + \hat{S}A\vec{I} + \vec{I}\vec{Q}\vec{I} - \gamma \vec{B}\vec{I}$$
where $\vec{S}$ and $\vec{I}$ are the electron and nuclear spin operators, $D$ the zero field splitting tensor, $g$ and $A$ the electron Zeeman and hyperfine coupling matrices, $Q$ the quadrupole tensor, $\gamma$ the nuclear gyromagnetic ratio, $\beta$ the Bohr magnetron and $B$ the applied magnetic field. [15] However when two paramagnetic centres are coupled, the EPR spectrum is described by a total spin Hamiltonian which is the sum of the individual Hamiltonians ($\mathcal{H}_A$ and $\mathcal{H}_B$ defined as per the previous equation) for the isolated centres (A and B) and the interaction Hamiltonian ($\mathcal{H}_{int}$) which accounts for the exchange and dipole-dipole interactions between the pair of paramagnetic centres. Thus

$$\mathcal{H}_{Total} = \mathcal{H}_A + \mathcal{H}_B + \mathcal{H}_{int}$$

$$\mathcal{H}_{int} = J_{AB} S_A S_B + d_{AB} S_A \times S_B + S_A D_{AB} S_B.$$  

The three terms in $\mathcal{H}_{int}$ are the isotropic exchange, antisymmetric exchange and the anisotropic spin-spin interactions. The equation can be extended to an arbitrary number of interacting spins by adding the individual spin Hamiltonians and summing $\mathcal{H}_{int}$ over all possible interacting pairs.

It is the various numerical components of this operator which convey meaningful information to the experimentalist. Unfortunately, the numbers that the experimentalist desires cannot be directly extracted from the spectra.

Instead, what happens, is that an estimate of the values is made and these values are used, with the help of a computer simulation, to produce a simulated spectrum. If the experimental spectrum closely resembles the simulated one, then the Hamiltonian values chosen are accurate and can be reliably used. If they do not match, then the need for further refinement of the values is indicated. Thus computer simulations are a very useful tool for the experimentalist.

The actual calculation performed during the simulation is

$$S(B, \nu_c) = \sum_{i=0}^{M} \sum_{j=i+1}^{M} C \int_0^\pi \int_0^\pi |\mu_{ij}|^2 f(\nu_c - \nu_0(B), \sigma_\nu) \, d\cos \theta \, d\phi$$

where $|\mu_{ij}|^2$ is the transition probability, $\nu_c$ the microwave frequency, $\nu_0(B)$ the resonant frequency, $\sigma_\nu$ the spectral linewidth, $f(\nu_c - \nu_0(B), \sigma_\nu)$ a spectral lineshape function and $C$ a constant which takes care of all the other experimental parameters.
1.3.1 Sydney OPera HousE (SOPHE)

Regrettably, most simulations are very expensive (in calculation time) for any of the higher order (more complicated) systems of molecules. Such calculations are in the order of hours. However Hanson and Wang [20, 21] have developed a far more efficient algorithm which they term “SOPHE”. This SOPHE simulation software uses a variety of interpolation methods to reduce the number of expensive matrix diagonalisations required.

An example gives an indication of the speed-up: a good quality simulation of Cr(III) requires approximately 48 hours of cpu time on a Sun SPARCstation 10/30 using the old algorithms. To simulate the same spectra to the same accuracy takes about 40 minutes using SOPHE. It is due to this significant speed increase in simulation technology that computer-based optimisation methods have now become practical.

1.4 The actual problem

The aim of this project is to formulate the problem of extracting the spin Hamiltonian parameters from EPR spectra as an optimisation problem. Once this is achieved, one or more promising methods should be chosen and implemented in an attempt to automate the extraction.
Chapter 2

Mathematical statement of the problem

In its simplest form (from a mathematical perspective), there are two sets of discrete data points.
Let $\mathbf{Y} = (Y_1, \ldots, Y_N)$ denote the $N$ experimental values.
Let $\mathbf{y} = (y_1, \ldots, y_N)$ denote the $N$ calculated values.
Let $\mathbf{x} = (x_1, \ldots, x_M)$ be the $M$ input parameters.
Let $f(x_1, \ldots, x_M) = f(\mathbf{x})$ be a function from $\mathbb{R}^M \to \mathbb{R}^N$ which denotes the SOPHE model. That is, if the parameters $\mathbf{x}$ are passed to the SOPHE model, then the output generated would be $\mathbf{y} = f(\mathbf{x})$.

The problem therefore, is to choose $\mathbf{x}$, so as to make $\mathbf{y} = f(\mathbf{x})$ as close as possible to $\mathbf{Y}$.

2.1 Scaling

Unfortunately, because the $\mathbf{y}$ and $\mathbf{Y}$ values have no units, then they are (separately) arbitrarily scaled. So before any meaningful comparison can be made, the two sets must be scaled to match each other. The method selected to do this was to create a function $S : \mathbb{R}^N \to \mathbb{R}$ which would give some measure of the scale of each of the sets. Using this, a scaling factor
\( \alpha = S(Y)/S(y) \) can be formed. Thus the comparison between \( Y \) and \( y \) becomes a comparison between \( Y \) and \( \alpha y \). The difficulty lies in the actual choice of the function \( S \).

### 2.1.1 Double integration method

One method which has been used in the past is to doubly integrate over the \( N \) data points. This, physically speaking, corresponds to the total transition probability. The discrete approximation to \( S(u) = \int_a^b u \, dz \, dz \) becomes \( S(u) = \sum_{i=1}^N \sum_{j=1}^i u_j \). This should in general be fairly successful, but it has one notable disadvantage: mis-scaling of graphs in the magnetic field intensity axis can lead to erroneous results. Take for instance two near-identical graphs where one is more “stretched out” than the other (ie. the line widths are correct, but the combination of the \( A \) and \( g \) values differ significantly). The double integral of these two spectra will generate different values; thus suggesting that the two spectra should be scaled differently when they are in fact correct.

### 2.1.2 Extreme points method

An alternative approach which can be used, is based upon the highest peaks/lowest valleys. The function is defined to be: \( S(u) = \max_i u_i + |\min_i u_i| \). This method is independent of the spread of the spectrum across the magnetic field intensity axis.

It should be quite effective as it matches the extreme peaks and troughs very well, and the rest will frequently be fairly close. However, if the extreme peaks are not correct, then force-matching them will tend to cause significant errors over the rest of the spectrum. These errors, being cumulative, will cause the optimisation algorithms to concentrate upon matching these extreme peaks accurately before any others. Such an imbalance is probably undesirable.

Matching the extremes rather than the double integral has the additional advantage of reducing the computational effort required (ie. \( 2N \) comparisons rather than \( N^2/2 \) additions). However the time to evaluate \( S \) is virtually insignificant in comparison with a typical evaluation of \( f \).
2.2 Error calculation

Having scaled the y-axis of the calculated data set to (hopefully) match that of the experimental one, the question of how to actually perform the comparison still remains. To be able to apply standard optimisation techniques, it is necessary to reduce the comparison to a single deterministic value. This could be denoted by the function $E : \mathbb{R}^N \times \mathbb{R}^N \to \mathbb{R}$. In addition, it would be useful if the function had a finite limit, ie. $\lim_{y \to Y} E(y, Y) \to 0$.

The simplest formula to achieve this would be $E(u, v) = \sum_i (u_i - v_i)$, which can be expressed as $(\sum_i u_i - \sum_i v_i)$. So (to take a pathological case), if $u_i = v_{N-i+1}$ (ie. a mirror image about a line parallel to the y-axis), then $E(u, v) = 0$. This is clearly undesirable as the optimisation algorithm cannot tell from the $E$ value whether a good fit has been achieved or not. What is wanted is a function $E$ which approximates to 0 if and only if $y \approx Y$.

One function which satisfies this criterion is $\sum_i |u_i - v_i|$. However this is undesirable as it has mathematical properties which make some analysis difficult (eg. differentiability is destroyed). An alternative is to take the sum of the squares, ie. $\sum_i (u_i - v_i)^2$. This is just the standard “least squares” fit (ie. the sum of the squares of the differences). Theoretically at least, this conforms to the requirements stated above in that $\sum_i (y_i - Y_i)^2 = 0$ implies that $y_i - Y_i = 0$ for all $i$. The least squares fit has the additional property that the differences with more extreme positive or negative values are exaggerated. This is desirable as it emphasises genuine peak mis-matching whilst at the same time tending to reduce the impact of the (relatively small) noise.

2.3 Summary

So the problem can now be defined mathematically as finding $x$ such that $F(x) = E(f(x), Y)$ is a minimum. This in full is:

$$\min_x \sum_{i=1}^N \left( Y_i - \frac{S(Y)}{S(y)} y_i \right)^2$$

where $y = f(x)$. 

10
Chapter 3

Optimisation methods

There are many different, well-known, optimisation algorithms. However for any given problem, some will be more effective than others. There are several important characteristics which must be borne in mind when selecting suitable algorithms for this problem.

The first major restriction is that (due to the exceedingly complicated nature of the model), no analytic derivative information is available. This is not an uncommon situation (see section 1.3.1), and can often be solved by the use of numerical approximations. That is:

\[ F_{x_i}(x_1, \ldots, x_M) = \frac{F(x_1, \ldots, x_i + \sigma, \ldots, x_M) - F(x_1, \ldots, x_i, \ldots, x_M)}{\sigma} \]

So obtaining \( \nabla F(x) = (F_{x_1}, \ldots, F_{x_M}) \) will require a total of \( M \) evaluations of \( F \) (which in this case leads to \( M \) evaluations of \( f \)). However as previously mentioned, for some data inputs, function evaluations can be very time consuming (possibly of the order of minutes per evaluation). So performing \( M \) evaluations would be very costly. If second order derivative information had to be approximated, then the number of evaluations of \( F \) would rise to \( M^2/2 + M \) which is a prohibitively large number for the slower function evaluations.

There is a further difficulty with derivatives. One of the integer parameters passed to SOPHE is the number of transition states. The user of the system may not know the correct value for this variable (which they indicate
to SOPHE by a zero value) in which case, the model will determine for itself the correct number, based upon the other inputs. This is very convenient for the user, but it means that a small change in one of the continuously adjustable variables (for which the true value is sought) may lead to the SOPHE model changing the number of transition states. This discrete alteration in the number of transition states will lead to a “sudden” change in the calculated spectrum. So while each of the $y_i$s of $f(x)$ is continuous for some regions of the $M$-space to be explored, this continuity is only piecewise in nature. As a result, the error function, $E$ is not in general continuous.

Therefore, the appropriate optimisation algorithm which needs to be chosen, should have the following characteristics:

- certainly not require any derivative information;
- (ideally) not assume continuity (although an algorithm which assumes continuity may work);
- be very sparing in its use of function evaluations (which are particularly expensive in this problem);
- have a history of effectiveness in curve-fitting problems;
- be one that is resistant to local minima;
- be effective on very non-linear problems;
- not be effected adversely when $M$ is large.

3.1 Historically used methods

Historically, a number of different methods have been used in solving related problems. Two in particular seem to be quite common. One is the well known general non-linear least squares method. [12, 11, 9, 10] The other is known as a Perturbation method. [12] Both of these methods have been rejected for this problem for reasons which are discussed below.
3.1.1 Non-linear least squares method (LSQ)

The non-linear least squares method uses a quadratic approximation to $F$. If the first derivative of $F$ is taken to be $f$, then this has a Taylor expansion about $x$ of:

$$F'(x + \delta) = f(x + \delta)$$
$$\approx f(x) + f'(x)^T \delta$$
$$\approx g + G\delta$$

where $g = \nabla F$ and $G = \nabla^2 F$. What is desired is a value for $\delta$ which will allow a move from the current location $x^k$ to $x^{(k+1)} = x^k + \delta$ where $F'(x^{(k+1)})$ is 0. This implies that $g + G\delta = 0$ which then leads to the iterative updating formula $x^{(k+1)} = x^{(k)} - G^{-1}g$.

This method, a traditional curve-fitting algorithm, has been quite popular historically. Analytic derivative information was available in the past as the models used in the simulation were considerably simpler—these early models did not take into account significant off-diagonal elements in the matrix which arise due to hyperfine coupling. In the absence of analytic derivative information, the first and second order derivative requirements make this an undesirable method for use with the more general SOPHE model.

3.1.2 Perturbation methods

Perturbation techniques are designed to be used only with a simplified version of the Hamiltonian expression. They assume that only one of the possible interactions is dominant, and so disregard the others. While this is a perfectly valid assumption for the simpler spin systems modeled by the older methods, it does not hold, in general, for the more complicated systems which SOPHE was specifically designed to model. Thus the perturbation method is inappropriate for this problem and has been rejected.
3.2 Chosen methods

In matching the many different non-linear optimisation methods against the criteria listed above (see section 3), two particular classes of methods stand out as “scoring highly”: direct-search and Monte-Carlo.

There is a class of non-linear optimisation methods known as “direct methods”, also sometimes known as “ad hoc methods”. Historically, these algorithms were among the first to be developed. They are characterised by evaluating the function at several points within the $M$-space and then use knowledge gained during the last few evaluations in an attempt to choose a more promising point. These methods have lost popularity over time, and have been largely superseded by methods using derivative information, although they are still used in places where noise is prevalent. They suffer only two drawbacks: the algorithms are particularly susceptible to becoming trapped in local minima; and they tend to be fairly inefficient in their use of function evaluations (at least in comparison with derivative-based methods on functions where derivatives are available). Two of these direct methods were considered particularly promising, the Hooke and Jeeve’s method, and the Simplex method.

The other class of optimisation methods which match most of the criteria, is the one into which all Monte-Carlo methods are placed. Monte-Carlo methods are also fairly old, but have only really become practical with the advent of computers. The classic Monte-Carlo method is simply a random walk about the $M$-space. Usually, steps which lead to a function decrease are accepted, and those which do not are rejected (i.e. another point is chosen within a given radius of the current point). These methods are considerably more wasteful of function evaluations than even the ad-hoc methods. However one of the Monte-Carlo methods, Simulated Annealing, has the attraction of being a global optimisation strategy. Global minimisers are those which are not defeated by local minima, but rather find the smallest minimum, or “global” minimum.

In all, four algorithms were chosen to be trialed. This was because it was unclear which would perform the best. While all four were implemented, only a small amount of comparative testing has been performed. The initial tests favored the Hooke and Jeeve’s method, and so testing of the other three methods was suspended. Future research should be done to re-examine the
others in more detail before they are completely rejected.

3.2.1 Hooke and Jeeve’s method

This method [8] was first publicised in 1961 and has been widely used for some time. Of particular note is that it has been used successfully in a number of curve fitting problems which have defeated some other methods. The method starts with an “exploratory” move. If this succeeds, then it is followed by one or more “pattern” moves in an attempt to speed location of a minimum. The pattern moves continue until no further progress is made, at which stage another exploratory move is made.

An exploratory move about $x^{(k-1)}$ consists of a series of $M$ function evaluations. First $x^{(k)}$ is set to $x^{(k-1)}$ then $F$ is evaluated at this point. What follows is a series of function evaluations at different points along each axis in turn. The first is $F(x^{(k)} + h_1 e_1)$. If this is successful (ie. an improvement upon $F(x_i^{(k)})$), then $x_i^{(k)}$ is becomes $x_i^{(k)} + h_1 e_1$. If it is not successful, then $F(x_i^{(k)} - h_1 e_1)$ is tried—if this is successful, then $x_i^{(k)}$ is set to $x^{(k)} - h_1 e_1$.

Having completed these evaluation(s), the process is repeated for $x^{(k)} \pm h_i e_i$ where $i = 2, \ldots, M$. The final resultant $x^{(k)}$ is the result of the exploratory move. This final function evaluation is compared against the evaluation at the start of the move. If it has improved, then the exploration is deemed successful and a pattern move is performed. If it is not successful, then all the $h_i$s are halved and another exploratory move performed.

The aim of the pattern move is to speed up the search for a minimum. It attempts to do this by reasoning that if moving from $x^{k-1}$ to $x^{(k)}$ caused a drop in $F$, then an extra step in this direction might do even better (ie. accelerate the process). So a new point $x'^{(k)} = x^{(k)} + (x^{(k)} - x^{(k-1)})$ is formed. Another exploration is performed, this time about $x'^{(k)}$. That is, evaluating $F$ at some of $x'^{(k)} \pm h_i e_i$ and adjusting $x'^{(k)}$ after each successful step. Success is this time measured against reduction in $F(x^{(k)})$—not $F(x'^{(k)})$ which is never in fact evaluated. If, at the conclusion of the pattern move, $F(x'^{(k)})$ is less than $F(x^{(k)})$, then $x^{(k+1)}$ is set to $x'^{(k)}$ and another pattern move is performed. However if $F$ has not been further reduced, then the pattern move is said to have failed, and another exploratory move
is performed about the last accepted point.

A number of different termination conditions can be used. The method used here is to stop when the $h_i$s have been halved more than a certain number of times (say $p$). This hopefully means that the algorithm will only terminate when a minimum has been located to within $h/2^p$ of the true minimum.

As stated above (see section 3.2), in the initial testing, this method proved to be the most reliable. Due to lack of time to perform further testing, this algorithm has been chosen to be the focus of more extensive testing.

### 3.2.2 Spendley, Hext and Himsworth’s method

A second direct method, which was once very popular, is that of Spendley, Hext and Himsworth [17]. It commonly goes by the name of the “simplex algorithm”—while this is an appropriate name (as will soon be seen), it is sometimes confused with the (unrelated) linear programming method. The simplex method is a more geometrical/topological approach to solving minimisation problems. It works by taking an $M + 1$ vertex simplex (ie. a regular $M + 1$ sided figure in $M$-space) and evaluating the $F$ at each of the vertices. It then takes the point which achieves the highest function evaluation (subject to a few restrictions) and replaces it with its reflection in the hyper-plane formed by the other $M$ points.

The elegant nature of this algorithm is apparent in considering the case where $M = 2$. The problem can now be visualized by a three dimensional landscape (ie. $z = f(x, y)$) whose lowest point (valley/well) we aim to find. The simplex in this case is an equilateral triangle which “rests” upon the three dimensional surface. The algorithm, simply dictates that the highest of the three points will lift over the other two, causing the triangle to “flip” over (so the highest point now points down hill). This process is repeated, and the triangle “flips” its way down to bottom of a nearby minimum.

More formally, the simplex is constructed of $M + 1$ vertices: $x_i$ where $i = 1 \ldots M + 1$. Given the initial point $x_1$, the other points ($1 < i \leq M + 1$)
are initially:

\[ x_i = x_1 + h_{i-1} p e_{i-1} + q \sum_{j \neq i-1} h_j e_j \]

where

\[ p = \frac{1}{M \sqrt{2}} (M - 1 + \sqrt{M+1}) \]

and

\[ q = \frac{1}{M \sqrt{2}} (\sqrt{M+1} - 1) \]

\( F \) is evaluated at each of these points. For convenience, suppose the points are reordered into ascending order, i.e. \( F(x_1) < \cdots < F(x_{M+1}) \). Then the point with the highest function evaluation, namely \( x_{M+1} \), is reflected in the hyper-plane defined by points \( x_1, \ldots, x_M \). So \( x'_{M+1} \) is set to be

\[ \frac{2}{M} \sum_{i=1}^{M} x_i - x_{M+1}. \]

\( F \) is evaluated at this new point. If it is an improvement on \( F(x_{M+1}) \), then \( x_{M+1} \) is replaced by \( x'_{M+1} \) and all the points reordered. If it is not an improvement, then the second highest point \( x_M \) is reflected to

\[ \frac{2}{M} \left( \sum_{i=1}^{M-1} x_i + x_{M+1} \right) - x_{M+1}. \]

If this an improvement upon \( x_M \), then they are swapped.

If both of these reflections fail to give an improvement upon their respective function evaluations, then the simplex is contracted about the point with the lowest function evaluation (i.e. all sides are halved, only \( x_1 \) remains unchanged). The contraction is performed by replacing \( x_i \) by \( x_1 + \frac{1}{2}(x_i - x_1) \) for all \( 1 < i \leq M + 1 \). \( F \) is re-evaluated at all points except \( x_1 \), and the process starts over again.

However, if one of the reflections did result in an improvement, then no contraction is performed. On the basis of their experimentation, Spendley et al. suggested that if one of the vertices has persisted (i.e. has not changed) for more than \( (1.65M + 0.05M^2) \) iterations, then a contraction should be performed about that persistent point anyway.
As with the Hooke and Jeeve’s method, there are numerous ways to determine termination. The one employed in this implementation is stopping when the number of contractions reaches a set limit.

Nelder and Mead [13] proposed a modified version of the simplex method. In this method, the simplex does not remain regular, but rather changes shape dynamically. This shape changing is similar to the Hooke and Jeeve’s method in that some successful moves cause the triangle to stretch twice as far in a promising direction. While this method has been shown to produce increased efficiencies for small $M$, it is not so marked when $M$ exceeds about five, and so would not be appropriate for this application.

The simplex method has two advantages over the Hooke and Jeeve’s method. One advantage is its use of $M + 1$ function evaluations in determining the next point at which to evaluate the function—the Hook and Jeeve’s method uses at most two points (and that only during the pattern step, it normally uses only one). Thus the (expensive in this case) function evaluations are “made the most of”. The second advantage is that the method is somewhat more resilient to local minima. This is because even if one of the points is very close to a local minimum, then, if the simplex is still large enough, the simplex may well “flip out” of the well.

Unfortunately, in some circumstances, the simplex method can fail. To take a two-dimensional case, it is susceptible to becoming trapped in a deep, narrow “valley” if one of the points lands very close the bottom and the other two points are on one wall each. Having reached this stage, no progress can be made except along the valley (the sides are too steep to allow any climbing). However if the point near the bottom is on the down-hill side of the valley, then no movement can be made as the triangle cannot flip over a point, only a line. So no reflections lead to an improvement, only contractions are possible. However as these contractions take place about the lowest point, then the situation remains effectively unchanged. Contractions will continue to occur until the termination conditions finally stop the process—possibly far from even a local minimum.

This irritating situation tends to arise when one of the search dimensions is too sensitive, or with a very non-linear function. The initial tests which were conducted suggested that the simplex algorithm, as applied to this problem, did indeed seem to experience this difficulty. However it should be stressed that more exhaustive testing needs to be performed before finally
rejecting this algorithm.

A possible tactic to overcome this difficulty, is to combine the standard simplex algorithm described above with a stochastic process (such as simulated annealing). This method would hopefully “break-out” of such traps. A brief discussion of how this could be implemented is given after the simulated annealing section.

3.2.3 Simulated annealing

As previously stated, simulated annealing is one of a class of methods known as Monte-Carlo methods. The simplest Monte-Carlo method starts at a point \( x^{(0)} \) and makes a random step of length \( l \) in \( M \)-space to the new point \( x^{(0)} \). If \( F(x^{(0)}) \) is less than \( F(x^{(0)}) \), then the step is accepted and \( x^{(1)} \) is set to \( x^{(0)} \); at which point the process is repeated by taking a step from \( x^{(1)} \). However if the function actually increases, then the step is rejected and another step is taken from \( x^{(0)} \). This process gradually forms a sequence of \( x \)s with decreasing function evaluations. The algorithm terminates with answer \( x^{(k)} \) when no more progress is being made. At this point the algorithm may be restarted from this answer with a decreased step size.

The concept of simulated annealing comes (somewhat loosely) from the annealing (solidifying) of liquids (especially molten metals). The molecules in their liquid state have a high kinetic energy which allows (indeed, forces) them to change their location and orientation. By cooling, their energy is depleted and so they gradually “settle down” into what eventually becomes their fixed location. The structure which results differs significantly depending upon the rate of cooling. A slowly cooled liquid will form a more stable solid with a lower net structural energy. A liquid cooled more quickly will have a higher net energy state and is likely to have flaws in its crystalline structure. This physical property of cooling is employed at metal smelters where metals are allowed to cool slowly, thus increasing their probable strength.

Simulated annealing attempts to emulate the slow cooling process by using one very significant change from the standard Monte-Carlo method. [4, 2, 14, 18, 7] Rather than rejecting all function-increasing steps, it accepts the “uphill” \( k \)th step with probability \( P(F(x^{(k)}), T) \). Where \( P \) satisfies
\[ \lim_{k \to \infty} P = 0. \] The value \( T \) is known as the temperature of the system. This decreases as \( k \) increases in accordance with some cooling schedule. The hope is that the acceptance of a detrimental step will have long-term benefits by allowing the sequence of \( x \)s to “escape” from a local minimum.

There are several, quite distinct versions of the algorithm; two of which have been implemented for this project and are described below.

**Varying several parameters at a time**

This version performs the random jump by choosing a random point on a hypersphere of radius \( r \) about the current point. Both the temperature and the radius are decreased periodically throughout the search. The decreasing temperature is an essential part of the simulated annealing method. The decreasing radius is based upon the premise that towards the later part of the optimisation, the current location will be close to the optimum, and hence the step size should be reduced to allow finer adjustments to be made.

One of the “benefits” of this method, which is emphasised by its supporters, is that with a single iteration, the algorithm is actually searching in all \( M \) dimensions simultaneously. The opposing viewpoint states that this is actually detrimental since if movement in one of the dimensions causes a sufficient increase in the function evaluation, it causes the rejection of the entire step. This is despite the fact that the movement in the other \( M - 1 \) dimensions may have been favorable.

**Varying one parameter at a time**

This version of the simulated annealing algorithm is more complicated. Rather than searching several dimensions simultaneously, it searches along each of the \( M \) axes in turn. For each axis, it takes a step of random length with a variable upper bound from the current position. At this point the function is evaluated and the step accepted or rejected. The step now taken along the next axis is taken from the new point.

An important part of this algorithm is the variable upper limit on the step size in each direction. As the algorithm progresses, a record is kept of
how many steps were accepted (either because they were improvements or because they were randomly accepted) along each of the axes. The theoretical optimum ratio of accepted steps to rejected ones should be about 1:1. If the actual ratio for a given axis does not fall within a certain tolerance of this ratio, then the step length for that axis is increased (if too many being accepted) or decreased (if too many being rejected). This choice is based on the premise that large steps are more likely to move out of the current, relatively good, position. Hence an increased step is more likely to cause a function increase leading to step rejection.

Initial tests with both of the simulated annealing algorithms as applied to this problem were disappointing. While they should theoretically converge upon the global optimum, in most of the tests performed, they failed to even locate a local minimum. While it is possible that better performance could be elicited by more careful tuning (of cooling schedules etc), it does not appear promising.

It must also be noted that the simulated annealing method is extremely wasteful of function evaluations. In particular, no history is kept, and so it is quite possible for the algorithm to explore the same area of the search-space on a number of occasions.

### 3.2.4 Simplex combined with simulated annealing

One method that has not been implemented for this problem, but may be worth trialing at some future date, is a stochastic variation of the simplex algorithm. As a means of avoiding the difficulties with the simplex method (ie. getting trapped in narrow valleys), a stochastic modification is possible. In particular, it has been proposed that aspects of the simulated annealing method be employed. [16]

One suggestion as to how this could be achieved is to add a small random amount to each of the function evaluations. Since the normal algorithm only looks at reflecting the two points with the highest function evaluation, then by artificially increasing the evaluations by differing amounts, other reflections might now be considered. To add a further stochastic element to the algorithm, the function evaluations made at each of the trial reflections could have a random value subtracted. This would have the effect of coaxing
the algorithm into accepting a reflection that it might otherwise have re-
jected. The upper limits on the random additions and subtractions would,
in line with the simulated annealing, be gradually reduced in magnitude.

3.3 Scaling/sensitivity—the $h_i$s

An important part of practical multi-variable optimisation is scaling. All three of the algorithms take finite steps along the various co-ordinate axes. However for the algorithms to work effectively, there must be some uniform-
ity in the rate of change in $F$ in each of these directions. Since a change of $\delta$ in one of the linewidth parameters will make considerably less change than a step of $\delta$ in one of the $g$ values, then, without scaling, for any given step size, only some of the variables will be usefully optimised. To solve this problem, the $M$-space formed by the parameters to be varied is transformed to one where a given step length will have a more uniform effect on $F$ in each of the $M$ directions.

Since the function $F$ is non-linear, then an ideal transformation for such a purpose would need to be non-linear also. This could, perhaps, be ap-
proximated by multiplying each of the $x_i$ by the partial derivative of $F$ with respect to $x_i$. However given previously mentioned difficulties in derivative calculation, this method is not really practical. The alternative chosen is to simply perform a linear transformation by multiplying each of the $x_i$s by $h_i$. The $h_i$s are manually chosen to keep fluctuations in $F$ to approximately the same order of magnitude.

3.4 Rejecting extreme/invalid points

In the early exploration of the $M$-space, the above algorithms sometimes chose fairly extreme values for some of the $x_i$s. This caused a problem when the unusually large (or unusually small) values in question were the $g$ and $A$ values. The values were not actually incorrect, but caused the non-zero part of the simulated spectrum to move completely out of the range in which the SOPHE simulation was working. This caused problems since once the spectrum had “disappeared”, then all function evaluations became identical.
(ie. just $\sum Y_i^2$).

To solve this, a number of tests were devised to determine when these values became too extreme. The restraining rules were as follows:

$$g_{x,y,z} + \frac{\beta}{ch} \leq A_{x,y,z} \leq g_{x,y,z} + width/(isol - 1) + \frac{\beta}{ch}$$

where $g_x$, $g_y$, and $g_z$ are subject to the constraint

$$\frac{10^9 hfreq}{\beta(centre - width/2)} \leq g_{x,y,z} \leq \frac{10^9 hfreq}{\beta(centre + width/2)}.$$

The addition of these constraints may seem trivial, but in fact transforms the problem from one of non-linear optimisation, to one of constrained non-linear optimisation. This is, in general, a very significant change. Many algorithms which work effectively for un-constrained problems require major changes for them to even work for constrained problems.

Fortunately, the nature of the four algorithms implemented here (such as not using derivative information) is such that the change was relatively straightforward. At each occasion where $F$ would normally be evaluated (ie. a complete simulation is performed) the constraints were first checked. If one or more of the constraints were violated, the simulation was not in fact performed and instead, a value of infinity was returned as $F$. This value (in practice just a large number which far exceeds any possible genuine $F$ evaluation) will always compare very unfavorably with the evaluation at any valid point. Such an unfavorable comparison will always lead to a rejection of this proposed new point.
Chapter 4

Implementation details and Results

Significant progress has been made in the accurate determination of Hamiltonian parameters. As an example, Figure 4.1 shows the experimental output from the analysis of \([\text{Vo(nopsal)}]\) in chloroform (with \(\nu = 9.4690 \text{ GHz}, T = 120 \text{ K}\)) along with the simulation of it. The simulation shown is based upon some very inaccurate Hamiltonian parameters. The Hooke and Jeeve’s algorithm was then applied to the problem giving the results shown in Figure 4.2. In this example, there were a total of six variables to be adjusted (ie. \(M = 6\)) and the number of points \((N)\) was 4096 (which is fairly typical). Note that the scale shown on the magnetic field intensity axis is actually the sample number (ie. 1 to \(N\)). As can be seen, the simulation using the parameters found by the optimisation routine gives a very good match to the experimental spectrum.

It is interesting to see how the error changes at each iteration of the Hooke and Jeeve’s method. Here one iteration is defined to be a single exploratory or pattern move. Figure 4.3 shows a plot of the error (which is scaled in a completely arbitrary fashion) against the number of iterations. The first few five iterations have been deliberately omitted as they are very large. The error in the first iteration is 305766, but within 5 iterations this has dropped to 2842. This indicates that the the initial parameters are a fair distance away from the optimum, and so significant improvement can be
made rapidly by taking relatively large steps. The error function plateaus for some time, the algorithm only making small improvements. By iteration 27, an exploratory move finally fails, and the step sizes are all halved. The more accurate adjustments which follow allow for a more rapid decrease in error.
4.1 SOPHE and parallelism

This kind of automated optimisation has only become possible through the speed improvements of SOPHE. However the time for a single function evaluation is still critical. One method to reduce the real time required for a
calculation is to parallelise the simulation code and run it on several processors concurrently. This has been addressed in a fairly superficial manner by simply perusing some sections of the code (ideally one should parallelise an algorithm—not the code directly).
At the cost of a small increase in memory consumption, some speed improvement was obtained. This was performed on a Silicon Graphics PowerChallenge which has a shared memory architecture. By spreading a fairly high level loop across several processors a speed improvement was obtained. This required some re-structuring of the SOPHE code, in particular, more memory needed to be allocated. This was necessary to store multiple copies of the working variables for each concurrent iteration.

The following table contains the timing results of running the parallelised SOPHE on several test problems of varying computational difficulty. The varying difficulty levels were achieved by increasing the accuracy to which SOPHE was requested to calculate the simulation. All times are in seconds.

<table>
<thead>
<tr>
<th>Computational difficulty</th>
<th>Number of processors</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>1</td>
</tr>
<tr>
<td>Test 1</td>
<td>101</td>
</tr>
<tr>
<td>Test 2</td>
<td>390</td>
</tr>
<tr>
<td>Test 3</td>
<td>867</td>
</tr>
<tr>
<td>Test 4</td>
<td>1531</td>
</tr>
</tbody>
</table>

Figure 4.4 shows a plot with logarithmic axis of the above data. The solid lines show the speed-ups actually obtained by running the simulations on different numbers of processors. The dashed lines indicate the ideal and are included for comparison. They indicate what a linear speed-up would look like.

The overall speed increases relative to increases in the number of processors were useful, but can probably be improved upon. A rough approximation of the parallel efficiency would be about 70%. That is, doubling the number of processors should lead to a time decrease of 30-40%.

### 4.2 Merging resonances

Whilst applying the Hooke and Jeeve’s algorithm to the error function as already stated is very effective at matching evenly sized resonances (as demonstrated in Figures 4.1 and 4.2), this method is not so successful with
Figure 4.4: Plot of number of processors against simulation time

problems in which resonances which differ significantly in magnitude, are adjacent.
With some EPR spectra (especially those with hyperfine coupling), several resonances merge to form a single “macro-peak”. This is not a problem in itself. However if one of these resonances is sufficiently offset from the others, then a small trough followed by another peak can occur. This means that a simulation with only one apparent stationary point (ie. formed with all of the resonances sufficiently close together) may receive a very low reported error. However, the accurate location of any such isolated resonance is in fact very significant in EPR spectroscopy (as a different location will indicate an entirely different compound). Therefore, these small resonances should be reflected in the optimum Hamiltonian parameters.

An example of this can been seen in Figures 4.5 and 4.6. Figure 4.5 shows the experimental spectrum for Bis(maltolato)oxovanadium(IV) in methanol (with $\nu = 9.593$ GHz, $T = 120$ K). The simulated spectrum also plotted is based upon parameters chosen by the application of the Hooke and Jeeve’s algorithm to the least squares error formula. For most of the spectrum, the two match quite well, but looking at Figure 4.6, it can be seen that the fine detail has been lost. This is because the resonances in the area have been spread out until they merge to form just a few, large, cumulative peaks.

Unfortunately, while this loss of information is quite apparent to the eye, it is not reflected well in the error formula. The curve shown in Figure 4.6 is in fact a very good fit, if one uses the least squares error function. The places where the two curves differ are small in extent (ie. they do not extend for a large region of the magnetic field intensity axis), and they are not of great magnitude. So the algorithm has “done its best” to fit the large scale features of the spectra, but in the process has irrevocably lost all the fine detail.

In an attempt to overcome this difficulty, a number of strategies can be used. One method is to use some kind of weighted error function to give a bias towards these areas. An alternative is to transform the spectrum and compare the results. Both of these approaches, along with a combination of the two, are discussed below.
4.2.1 Overweighting of peaks

The error function given earlier is to simply take the sum of the squares of the differences between the two spectra. However this tends to give considerably more emphasis to the large resonances (either wide or high) rather
than the small. Looking carefully at Figure 4.6, it can be seen that any coarse approximation to a series of fine resonances will usually involve a noticeable discrepancy for at least half of the peaks or troughs. It would seem, therefore, that if the error at each of these peaks and troughs were emphasised, then an improvement might result.
This new error function can be defined as

\[ E'(u, v) = E(u, v) + E_{\text{Peak}}(u, v) \]

where \( E_{\text{Peak}} \) is a strongly weighted function which ensures that distinct peaks (and troughs) are preserved. Say there are \( s \) distinct peaks and troughs in the experimental spectrum occurring at the points \( p_i \), which are integers between 1 and \( N \). The heights of these peaks are obviously \( Y_{p_1}, \ldots, Y_{p_s} \). Then \( E_{\text{Peak}} \) can be defined as \( k \sum_{i=1}^{s} (Y_{p_i} - y_{p_i})^2 \) (the arguments given previously for using the sum of the squares remain unchanged), where \( k \) is a constant. Now if \( k \) is sufficiently large, then hopefully no optimisation algorithm will try improving the general fit of the curve at the expense of amalgamating any of the discrete peaks.

### 4.2.2 Peak/trough detection

The actual detection of peaks and troughs itself is not a trivial exercise. The simplest method, is to assume that if \( Y_{i-1} < Y_i \) and \( Y_i > Y_{i+1} \) for some \( i \), then the point \( i \) is a maximum (peak); and if \( Y_{i-1} > Y_i \) and \( Y_i < Y_{i+1} \) for some \( i \), that \( i \) is a minimum (trough). However this definition will find many spurious maxima/minima due the noise in the system. To avoid this, an alternative is to look at \( 2t + 1 \) points at once. For the case of the maximum, we can strengthen the previous condition to requiring \( Y_{i-j} < Y_i \) and \( Y_i > Y_{i+j} \) for all \( 1 \leq j \leq t \), where \( t \) is some integer. In effect, this is forcing the point \( Y_i \) to be a local maximum throughout the \( 2t \) points. \( t \) must be made large enough to prevent noise being counted as a peak, but small enough so that close peaks are not omitted.

However this may still produce false minima/maxima on near straight segments where noise exists. To overcome this, a further condition can be imposed, namely that both \( Y_{i-t} - Y_i \) and \( Y_{i+j} - Y_i \) are greater than \( q \) for \( i \) to be considered a peak. \( q \) is some positive real number which is large enough to prevent such false identifications. This forces the peak to be of at least a minimum height. Appropriately strengthened versions for detecting troughs can be deduced along similar lines.
4.2.3 Fourier smoothing

To further aid in the detection of peaks and troughs, some kind of smoothing is beneficial. The raw experimental data sets sometimes contain a considerable amount of noise; while it is not enough to cause problems in the accuracy of the fit of the spectrum, it does cause difficulties in detecting the peaks/troughs.

The Fourier transform of a function treats it as a cyclic function consisting of the sum of a number of sine/cosine curves with varying frequencies and amplitudes. Formally, this is

\[ h(t) = a_0 \sum_{i=1}^{\infty} [a_i \sin(i\pi) + b_i \cos(i\pi)] \]

Since the spectrum in this case is centred about the magnetic field intensity axis, then \( a_0 \) will always be zero, and hence can be disregarded. The noise present in the experimental spectrum can be considered as the sum of a series of (relatively) very high frequency sine/cosine waves. Since the underlying “real” curve of the spectra (ie. what it theoretically should be without the noise) is generally low-frequency in nature, then reducing or removing some of the high frequency components of the spectrum should preserve the overall shape/proportions of the results, whilst reducing the noise level.

Since the experimental spectrum is not continuous, then a discrete Fourier transform must be used. This is \( \text{fft}(u) = (H_1, \ldots, H_N)^T \) where

\[ H_n = \sum_{k=0}^{N-1} u_k e^{2\pi i kn/N}. \]

The inverse transform (ie. that which converts a Fourier transformed spectrum back into a normal curve is then \( \text{fft}^{-1}(u) = (h_1, \ldots, h_N)^T \) where

\[ h_k = \frac{1}{N} \sum_{n=0}^{N-1} u_ne^{-2\pi i kn/N}. \]

The process of filtering out the noise can be achieved by setting \( u' = \text{fft}^{-1}(\text{clean}(\text{fft}(u))) \) where \text{clean} is some function which reduces the high frequency portions of the spectrum whilst leaving the low frequency ones unchanged. A simple method is to set \( \text{clean}(u) = c.u \) where \( c = (c_1, \ldots, c_N)^T \) and all \( c_i \in [0, 1] \). Bearing in mind that the Fourier transform of a real-valued spectrum (which this is) is symmetric, and so too should \( c \). One successful
version is
\[ c_i = \begin{cases} 
  1, & 1 \leq i \leq a \\
  \frac{i-b}{a-b}, & a < i < b \\
  0, & b \leq i \leq N - b \\
  \frac{i-N+b}{b-a}, & N - b < i < N - a \\
  1, & N - a \leq i \leq N 
\end{cases} \]

4.2.4 Shortcomings in the peak detection method

While the detection method described in section 4.2.2 is fine in theory, in practice it is difficult (if not impossible) to choose a \( q \) and \( t \) pair which will select all the “true” resonances, but no noise-induced spurious ones. Trials suggest that Fourier “smoothing” results in a very significant improvement in accuracy. However even then it is not as effective as one might expect. The difficulty arises due to the variation in shape of the “true” resonances. To be sure of correctly locating narrow peaks and troughs which are close together (frequently occurring if hyperfine coupling exists in the test substance), a fairly small value of \( t \) must be chosen. However to prevent false-identification of noise, this small value of \( t \) requires a relatively large \( q \). In many cases, the larger peaks have fairly gradual curves (ie. their derivative is relatively low in magnitude) which do not fall away as rapidly as a small \( t \)/large \( q \) combination requires. A possible solution for this difficulty is to perform a two stage search: once with a small \( t \)/large \( q \) combination, once with large \( t \)/small \( q \), and then take the union of the two results.

An alternative method (which has not been tried) is based upon the Savitzky-Golay smoothing technique. [16] The SG method also looks at each set of \( 2t + 1 \) consecutive points. For each such set of points, the SG method usually fits a polynomial function to the section and takes the evaluation of the polynomial at the \( i \)th position as the new smoothed point. A variation upon this for finding peaks and troughs would be to always fit a quadratic, and then determine the turning point of that fitted quadratic (which occurs at \(-b/2a\)). If the turning point lies within the current window and it is sufficiently higher (or lower in the case of troughs) than the curve at the two ends of the window, then it can be flagged as a peak (or trough).
4.2.5 Results of overweighting

The overweighting of peaks (as described in section 4.2.1) does solve the loss of fine detail to a certain extent. During the first part of the optimisation process, it is successful. However as the parameters yield a simulation which better approximates to the experimental spectrum, the same phenomenon re-occurs. Even removing the first term of the error function, leaving only the $E_{Peak}$ term in place does not solve the problem entirely, even though this extreme version was the most successful, as can be seen from Figure 4.7.

This enlargement showing the problem area has its simulation made from values part way through the optimisation process. Unfortunately, as can be seen in Figure 4.8, the optimisation algorithm has matched the larger resonances very poorly, simply by moving them slightly so that the simulated spectrum crosses the experimental one exactly at the stationary point—thus giving a zero error reading. If the algorithm is allowed to run to completion, then either the large peaks will never match, or else the fine resonances will again be lost.

4.3 Comparison of Fourier transforms

Rather than just comparing the two raw spectra, it is possible that comparing some transformed version of them will prevent the problem of losing the fine detail (described in 4.2).

Define $v = G(u)$ to be the transform of the the spectrum $u$ to a different basis. The “standard” basis is $(e_1, \ldots, e_N)^T$ with the coefficients simply the $(Y_1, \ldots, Y_N)^T$. Irrespective of the basis, the (possibly weighted) least squares comparison can be used. That is, rather than taking the sum of the squares of the differences in the original spectrum, one can take the sum of the squares of the differences in the new basis. Experimentation with comparing the plain Fourier series in this manner showed that it was subject to the same difficulties of losing detail as discussed previously.

Fourier analysis has been applied to EPR spectra in the past with useful results, so it was hoped that it could usefully be used here. [5, 1] Now
Fourier analysis is really mapping these coefficients across to a new basis which consists of sines and cosines. This is useful because it separates out the high frequency components from the low frequency ones. This fact was exploited in the discussion on Fourier smoothing. In this case, the region of
interest is the “medium-high” frequency range; that is, lower in frequency than the pure noise components of the signal, but (hopefully) at about the frequency of the relatively narrow fine resonances.
It was hoped that exaggerating the medium-high frequencies might overcome the difficulty. Several different ways of doing this were trialed. The simplest was a piece-wise linear function which left the low frequency portion of the spectrum unchanged, boosted the medium-high range, and removed the very high range. This weighting was used in two different ways. One was directly comparing the weighted Fourier series. The second was to transform the exaggerated Fourier series back into its original basis and perform the comparison there. Both of these were unsuccessful in the long term—the now familiar difficulty of losing fine details re-occurred.

A second attempt was made with a more sophisticated weighting function. This time a Gaussian curve (ie. $e^{-u^2}$) was used to exaggerate the Fourier sequence. Again, it was applied to emphasise the errors in the medium-high frequency range tailing off to unexaggerated low and very high ranges. This more sophisticated method fared no better than the piece-wise linear version. It could be that some other, more appropriate basis exists which would allow a more successful comparison or exaggeration prior to comparison.
Chapter 5

Future directions

While progress has been made, further improvement is certainly possible. The main objective of any research now would be in an attempt to overcome the loss of fine detail in some spectra. A secondary aim might be to improve the speed/efficiency of the optimisation. There are a number of avenues which have either not been investigated at all, or have not been adequately explored to determine their usefulness. These various paths can be grouped into two areas: those that deal with the actual minimisation algorithm, and those pertaining to the error/comparison function used.

As mentioned earlier, initial tests of the four algorithms implemented suggest that the Hooke and Jeeve’s method is the most reliable. However more thorough testing should be performed to verify this. There is also the possibility of trialing yet another algorithm. One potential candidate is the combined simplex/simulated annealing method outlined earlier. This may combine the efficiencies of the simplex method with the (theoretically) globally convergent behavior of the simulated annealing. If these desirable characteristics were preserved when applied to this problem, then this would make for a most competitive method.

With regards to the error/comparison function, several directions for investigation are apparent. Further attempts could be made at the weighting of the least squares function at the peaks and troughs of a spectrum. Perhaps, rather than just emphasising the stationary point itself, it might be advantageous to overweight the error over a finite region of each peak/trough.
Another alternative might be to determine which peaks are most susceptible to being lost, and only emphasise the error around them.

Rather than working with the raw experimental and simulated spectra, a (possibly weighted) comparison could more easily be performed in a different basis. In a manner similar to the Fourier analysis that has already been tried, the spectra could be decomposed into a series of Gaussian peaks (or their derivatives) in the hope that this might form a more natural set of basis functions. Comparisons could be made either of the (possibly weighted) transformations, or alternatively, the transformation could have the relevant features exaggerated before mapped back to its original basis to for comparison there.

Finally, if the loss of detail by a coarse approximation can be considered to correspond to a local minimum in the search space, then the necessary goal is not just minimisation, but global minimisation. Perhaps by borrowing some of the characteristics of a niche search, a method could be devised which would find the best of the various local minima present.
Chapter 6

Conclusion

This project has accomplished a number of things. It has formalised the problem, of finding Hamiltonian parameters from EPR spectra, mathematically. After determining the special characteristics of this problem, careful consideration was given to choosing a range of suitable algorithms. Four such algorithms were chosen and implemented, one of which in particular has proven to be quite robust in finding at least a good local match of at least the broad features in the spectra.

A number of difficulties which were encountered have been addressed and solved. The problem of the algorithm losing the fine detail of the spectrum in some circumstances remains, but some improvement has been made and a number of possible solutions have been outlined in the Future Directions section. Finally, an improvement in the computational speed of the SOPHE simulation model has been achieved through the use of parallelisation.
Chapter 7

Acknowledgments

Thanks should go to many people who assisted me in this project. In particular Kevin Burrage and Graeme Hanson for being (respectively) my official and unofficial supervisors. Roger Sidje and Kevin Gates were always very generous of their time in assisting me with technical problems on the PowerChallenge. In addition, I would like to thank my parents and colleagues for their encouragement and many valuable suggestions.

Thanks is also due to the Centre for Magnetic Resonance at The University of Queensland for their generous provision of a scholarship whilst carrying out this work.
Bibliography

[1] Basosi, Riccardo; Della Lunga, Giovanni and Pogni, Rebecca Resolution enhancement of nitrogen hyperfine patterns in the EPR spectra of Cu(II) complexes: FT analysis of Cu(II)(His-Gly)$_2$


