You can never solve a problem on the level on which it was created.

–Albert Einstein
A STOCHASTIC OPTIMIZATION METHOD FOR PARTIALLY DECOMPOSABLE PROBLEMS, WITH APPLICATION TO ANALYSIS OF NMR SPECTRA

by

Mohsen Ravanbakhsh

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of

Master of Science

Department of Computing Science

©Mohsen Ravanbakhsh
Fall 2009
Edmonton, Alberta

Permission is hereby granted to the University of Alberta Libraries to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only. Where the thesis is converted to, or otherwise made available in digital form, the University of Alberta will advise potential users of the thesis of these terms.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as herein before provided, neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author’s prior written permission.
Examiner Committee

Professor Russell Greiner, Department of Computing Science

Dr. Barnabás Póczos, Department of Computing Science

Professor Douglas Wiens, Department of Mathematical and Statistical Sciences

Professor David Wishart, Department of Computing Science
To my parents
Many real-life optimization problems have structures that one may be able to exploit to solve the problem efficiently. A common structure is *partial decomposability*, which decomposes the original problem into a set of weakly coupled sub-problems that are each relatively easy to solve. However, due to the coupling, the solution to each sub-problem must also incorporate information from the other sub-problems. We designed the Cross Entropy method Exploiting (partial) Decomposability (CEED) to address such problems, especially analyzing $^1$H Nuclear Magnetic Resonance (NMR) spectra of mixtures. This analysis involves finding the concentration of chemicals in a complex liquid mixture. Each chemical’s signature is a set of clusters of peaks, that shift horizontally in a bounded domain and scale vertically proportional to the chemical’s concentration. Therefore when each chemical’s signature is known, the task is to best match a nonlinear model, by finding appropriate shift and concentration values. This could be formulated as a nonlinear optimization problem involving hundreds of variables. Since chemical signatures have ‘local’ effects on each other, the problem is decomposable to loosely coupled sub-problems that are easier to solve. Experimental results show superior performance of CEED to other common optimization methods as well as a state-of-the-art systems for analysis of NMR. We also apply CEED to the SAT and Sudoku combinatorial optimization problems to show the generality of our method and its superiority to the original Cross Entropy method.
Acknowledgements

I wish to thank my co-supervisors; Prof. Greiner for his invaluable support and excellent guidance and Dr. Barnabás Póczos for his insightful and expert assistance. I want to express my gratitude to Jack Newton and especially Pascal Mercier from ChenomX Inc. for providing us with technical information, real-world data and their constructive feedback. I also wish to thank my other committee members Prof. Wiens and especially Prof. Wishart for his detailed feedback and finally Maria and Stuart for their kind support.
# Table of Contents

1 Introduction
   1.1 Stochastic Optimization Methods ................................................ 2
   1.2 The Cross-Entropy Method ......................................................... 3

2 Partially Decomposable Optimization
   2.1 Definition and Notation .............................................................. 9
   2.2 Cross Entropy Exploiting partial Decomposability (CEED) method .......... 10
   2.3 Combining Several Maximum Likelihood Estimations ......................... 13
      2.3.1 Combining Estimators Using Their Fisher Information ............... 14
      2.3.2 Relation of different combination methods .......................... 15
   2.4 Examples ................................................................................. 17
      2.4.1 SAT Problem .................................................................. 17
      2.4.2 Sudoku Problem ............................................................... 21

3 Analysis of Nuclear Magnetic Resonance Spectra
   3.1 Background, Problem Formulation and Application of CEED .................. 23
   3.2 Experimental Results ................................................................... 26
      3.2.1 Simulated Spectra ............................................................... 27
      3.2.2 Real Spectra .................................................................. 35

4 Conclusion ................................................................. 40

Bibliography .............................................................. 41

Appendices .................................................................... 45

A Cross-Entropy Method from Importance Sampling Perspective .............. 45
   A.1 Iterative Method .................................................................. 47

B A More Technical Background on NMR Spectroscopy ................................ 49

C Details of CEED for NMR
   C.1 Preprocessing .................................................................. 52
   C.2 Defining Distributions ............................................................... 55
   C.3 The Loss Measure ................................................................. 55

D Quasi-Convexity of $L_2$ Loss ..................................................... 57
# List of Tables

<table>
<thead>
<tr>
<th>Table</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.1</td>
<td>A Summary of notation.</td>
<td>10</td>
</tr>
<tr>
<td>3.1</td>
<td>Library configuration for all 5 scenarios.</td>
<td>28</td>
</tr>
<tr>
<td>3.3</td>
<td>Concentration error of CEED for 5 scenarios of simulated spectra.</td>
<td>29</td>
</tr>
<tr>
<td>3.2</td>
<td>Detection error of CEED for 5 scenarios of simulated spectra.</td>
<td>29</td>
</tr>
<tr>
<td>3.4</td>
<td>Comparison of CEED and SAGD for presence detection task on real spectra.</td>
<td>36</td>
</tr>
<tr>
<td>3.5</td>
<td>Comparison of CEED and SAGD concentration errors for real spectra.</td>
<td>36</td>
</tr>
<tr>
<td>C.1</td>
<td>Comparison of performance when using different loss measures.</td>
<td>56</td>
</tr>
</tbody>
</table>
# List of Figures

1.1 A simple illustration of CE’s evolution in iterations. ....................... 4

2.1 A Sudoku puzzle as a partially decomposable problem ....................  8
2.2 Coupling matrices for different problems. ................................. 11
2.3 Combination of two beta distributions using Fisher Information ........... 17
2.4 Evolution of CEED and OCE for SAT problem. .............................. 21
2.5 Comparison of convergence rate of CE and OCE for SAT problem .......... 21
2.6 Evolution of joint categorical distribution for a sample Sudoku puzzle.  22

3.1 A single lorentzian peak as a function of three parameters. .............. 23
3.2 Two clusters of a metabolite, $i_1, i_2 \in \Gamma(j)$ .......................... 24
3.3 Comparison of the convergence rate on a simulated spectra. .............. 27
3.4 Sample CEED fits for scenarios 1 and 2 of simulated spectra .............. 30
3.5 Sample CEED fits for scenarios 3 and 4 of simulated spectra .......... 31
3.6 Sample CEED fits for scenarios 5 of simulated spectra (incomplete library) 32
3.7 Sample CEED reported concentrations for scenarios 1, 2 and 4 of simulated spectra. 33
3.8 $L_2$ error for different metabolites of analyzing library of 4th scenario. 35
3.9 The effect of sample-size on convergence rate and the resulting fit. .... 36
3.10 CEED’s fit for a typical real 600MHz (a) and 800 MHz (b) spectra ....... 37
3.11 Comparison of CEED’s reported concentration with expert estimate. .... 38

B.1 Effect of different parameters on a single harmonic component. ........ 50

C.1 Cluster center as a function of pH. ........................................ 53
C.2 The border of effect and shift bounds for different clusters of library.  54

D.1 Loss function and its derivatives for a single-peak curve-fitting problem. 58
D.2 The $L_2$ loss as a function of cluster center. .............................. 59
Chapter 1

Introduction

Nuclear Magnetic Resonance (NMR) spectroscopy is a technique that exploits the magnetic properties of certain nuclei to measure the radio frequencies that the nuclei in a chemical or mixture of chemicals absorb. These absorption frequencies are measured to produce an NMR spectrum that consists of many peaks—each having a characteristic chemical shift (absorption frequency) and intensity. The versatility of NMR spectroscopy has made it a widely used tool in many fields of natural science, including chemistry and biology with applications ranging from determining the molecular structure of chemicals to verifying chemical content to analyzing cellular metabolism.

Many applications of NMR fall within the field of metabolomics. Metabolomics is a newly emerging field of science that studies the chemical fingerprints associated with cellular processes [13]. Beside applications in biology and chemistry, our core motivation to study NMR spectra comes from the fact that it is both reliable and rapid. Indeed NMR-based metabolomics could be part of a new revolution in health-care, as it will enable researchers and physicians to learn which metabolic profiles are associated with a particular disease or the success of a treatment. In other words NMR-based metabolomics could provide better detection, prognosis and treatment of diseases.

NMR spectroscopy exploits the magnetic properties of nuclei belonging to a specific element (usually $^1$H or $^{13}$C). Since the resonance condition at which a given nucleus absorbs a given magnetic field depends on the structure of the compound to which it belongs, the resulting spectrum contains a specific NMR spectral signature (which is a collection of peaks) for each compound present in the mixture. The total area under these peaks is proportional to the compound’s concentration [16]. See Appendix B for a more technical background on NMR spectroscopy.

Given an NMR spectrum (e.g., from a urine or blood sample) and a library that characterizes the compounds (in terms of locations and relative heights of each compound’s peaks), our task is to compute the concentrations of the compounds, seeking the reconstruction that best resembles the original spectrum [44]. This task would be easy if each peak was at a fixed location but unfortunately, these peaks can shift their locations, based on factors such as temperature, pH and the effect of different compounds on each other (a.k.a. the matrix effect). Moreover, these shifts are not predictable over all factors—i.e., we cannot predict the shifts, which means we have to find the shift of
(the peaks of) each compound, as well as the concentration. Since unknown shift values make our model of the spectra non-linear in parameters, our problem is significantly more difficult. Depending on the number of potentially present compounds, the process of fitting could be extremely hard and cumbersome.

We provide an automated procedure for analyzing NMR spectra, viewing it as a constraint non-linear optimization problem that involves hundreds of bounded variables. As the data is noisy, and the loss function is not convex, the best candidates for solving this problem seem to be global, stochastic optimization methods. However like some other optimization problems, we can decompose our problem into a set of coupled sub-problems that are easier to solve. Therefore, motivated by the structure of this problem, we devised an optimization method that can exploit the structure of partially decomposable problems, by first considering each sub-problem separately and then combining the updated distributions over each variable domain iteratively.

This is done by extending the Cross Entropy (CE) method [41] — a stochastic optimization method that resorts to iterative adaptive sampling to find the global minimum of the loss function. This method has been successfully applied to many different domains, including a variety of continuous and combinatorial optimizations [41, 23], clustering and vector quantization [24], policy optimization [29], and buffer allocation [2].

Overall this thesis introduces a new method for optimization of partially decomposable problems and demonstrates its successful applicability to the real world problem of mixture analysis by NMR spectra as well as some related combinatorial problems. In the remaining parts of this chapter we will provide an overview of stochastic optimization methods (Section 1.1), focusing on the Cross-Entropy method in Section 1.2. In Chapter 2 we introduce our iterative stochastic method for partially decomposable problems (CEED) and give some examples of its application to the combinatorial SAT problem and the Sudoku puzzle. Finally in Chapter 3 we use our method to analyze the NMR spectra, and present experimental results to support CEED’s superior performance.

1.1 Stochastic Optimization Methods

A stochastic optimization method involves randomly sampling from the compact domain $\mathcal{X}$, seeking the global minimum (maximum) of the loss (objective) function. Pardalos [31] divides the stochastic optimization methods into three classes:

1. A Two Phase method has a phase of global sampling followed by a local search; see multi-start variations like GRASP (Greedy Randomized Adaptive Search procedure) [15] and Random Linkage [28] methods. These methods are most successful when efficient local search is possible.

2. In Random Function methods, the objective function is modelled as a sample path of some stochastic process, which gives a distribution over a class of functions. When it is expensive
to evaluate the objective function itself, the probabilistic model of the function, constructed by active sampling, could be used for optimization (see for example [21])

3. Finally, *Random Search* methods generate sequences of sample points in the feasible region, according to some (potentially) adaptive distribution. A most basic approach is *Pure Random Search*, which simply generates samples from a predefined distribution over the domain $X$, and reports the best sample as the answer.

*Pure Adaptive Search* (PAS) is a random search method in which a sequence of points are uniformly generated from a level-set of samples from a previous iteration. A level-set $\tau$, refers to the set of all parameters (regions in a continuous domain) for which the value of the loss function is less than or equal to $\tau$.

Smith and Zabinsky show that using PAS for a Lipschitz loss function in a continuous domain, the number of function evaluations required to find the global minimum increases linearly with dimensions of the problem [43]. This is a bound expected for a convex program [8]. This result was later generalized to finite, discrete domains [50].

This method inspired algorithms like *Improving Hit and Run* [49] and *Hide and Seek* [35]. *Hesitant Adaptive Search* (HAS) generalizes PAS by allowing non-improving samples from the current level-set with some probability [9]. *Adaptive Search* (AS) is a theoretical algorithm modeling the behaviour of Simulated Annealing (SA) [35]. Instead of sequential sampling over improving level-sets, AS samples from an improving *Boltzmann* distribution over the search domain.

Given properly adjusted parameters, HAS and AS are known to have linear convergence properties similar PAS [48, 42]. Markov Chain Monte Carlo (MCMC) methods for implementation of this algorithm have been devised (see [42]).

One could view the *Cross Entropy* (CE) method as a random search method that, rather than taking sequences of one sample at a time, instead generates a set of samples in each iteration. This allows the algorithm to adaptively adjust some parameters (e.g., the counterpart of cooling parameter in Simulated Annealing) that are crucial factors in the performance of the method. Furthermore, in contrast to AS, the CE method does not constrain the sampling distribution to the Boltzmann family.

### 1.2 The Cross-Entropy Method

The basic idea of the CE method is to produce improving distributions over the problem domain. This is performed by batch sampling from a prior in each iteration and using the information from the samples to produce a better distribution as the prior for the next iteration. Figure 1.1 shows how a Gaussian distribution converges to the global optimum of the given loss function in successive iterations.

---

1AS is theoretical in the sense that it does not specify the sampling method.
Figure 1.1: A simple illustration of evolution of distributions in CE and its convergence to global optimum.

Now we introduce the CE method starting from the Pincus theorem. In this way we can better show its relation to other stochastic optimization methods like AS and CE’s recent variations [39, 37, 38]. (Appendix A introduces CE method as an iterative importance sampling approach. This is historically the context in which CE was developed.)

Let $\mathcal{X}$ be a bounded convex domain, either continuous or discrete, over which the loss function $L(\cdot) : \mathcal{X} \to \mathbb{R}$ is defined. Our goal is to find an approximation to $x^* \in \mathcal{X}$, the global minimizer of $L(\cdot)$. When $\mathcal{X}$ is a continuous domain, the problem is known as continuous multi-extremal optimization. When $\mathcal{X}$ is a discrete domain we are dealing with combinatorial optimization problems.

Pincus’ theorem [32] states that, if $L(x)$ has a unique global minima, it is given by:

$$
x^* = \lim_{\lambda \to 0} \frac{\int_{\mathcal{X}} xe^{-\frac{L(x)}{\lambda}} dx}{\int_{\mathcal{X}} e^{-\frac{L(x)}{\lambda}} dx}
$$

(1.1)

To understand the statement intuitively, one should notice that for a small $\lambda$, the main contribution to the nominator comes from the small neighborhood of global minima. The denominator serves as a normalization factor.

Eq (1.1) is basically $E_{B_{\lambda,L}}[X]$ (i.e., $E[X]$ when $X \sim B_{\lambda,L}$), where $B_{\lambda,L}(x) = \frac{e^{-\frac{L(x)}{\lambda}} x}{\int_{\mathcal{X}} e^{-\frac{L(x)}{\lambda}} dx}$ is a Boltzmann distribution with $L(\cdot)$ as the energy function and $\lambda$ as the temperature parameter. The Boltzmann distribution is the distribution with maximum entropy with some expected energy [45].

$$
E_{B_{\lambda,L}}[L(x)] = U(\lambda)
$$

(1.2)

Here $U(\lambda) = \tau$, the expected energy, is an increasing function of temperature and may naturally refer to some level-set. This equation implies that one could sample from $B_{\lambda,L}$ for an small $\lambda$, and use the sample mean as an approximation to $x^*$. Pincus used the Metropolis-Ulam [30] sampling...
method for this purpose [33]. Alternatively, for difficult problems we could decrease the temperature parameter as we select samples, with smaller expected values in each iteration. This gives us samples around improving level-sets, which is basically the Annealing Adaptive Search method. Here although all samples are not within a level-set, their expectation $E[L(X)]$ represents a level-set that is getting closer to the ideal level-set, $L(x^*).$ When using MCMC methods for sampling from $B_\lambda,L,$ gradual reduction of temperature gives sequential sampling a higher chance of reaching the global minima. This is the reason for the good performance of the celebrated Simulated Annealing (SA) method.

Let $I_{\{\text{cond}\}}(x)$ be equal to one if the Boolean condition $\text{cond}(x)$ is true and zero otherwise. While AS takes its samples from a maximum entropy distribution around improving level-sets, the Cross-Entropy method samples from the distribution (of specific family) with the minimum cross entropy distance to the uniform distribution within level-set, $L(x) < U(\lambda) = \tau.$ That is:

$$f(\cdot) \doteq \arg\min_{g(\cdot)} \left\{ \int_X h_\tau(x) \ln \frac{h_\tau(x)}{g(x)} dx \right\} = \arg\max_{g(\cdot)} \left\{ \int_X h_\tau(x) \ln(g(x)) dx \right\}$$  \hspace{1cm} (1.3)

where

$$h_\tau(x) \doteq \frac{\mathbbm{1}_{\{L(x) < \tau\}}}{\int_X \mathbbm{1}_{\{L(x) < \tau\}} dx},$$

is the uniform level-set distribution.

\textbf{Eq} (1.3) represents the main idea of the Original Cross Entropy (OCE) method for optimization, but in a different perspective from its original importance sampling approach [36, 41].

This calculation (of Eq (1.3) ), however is performed iteratively while updating of $\tau$ and $f,$ which we will denote by $\tau^t$ and $f^t.$ In iteration $t,$ OCE first draws $N$ instances from $f^{t-1}$: $X = \{X_i \sim f^{t-1}(x)\}_{1 \leq i \leq N},$ renumbered such that $L(X_1) \leq \ldots \leq L(X_N).$ It then uses the elite samples $X_{\text{elite}} = \{X_i\}_{1 \leq i \leq \rho N},$ which are the top $\rho$ percentile of the instances. At the next time-step, OCE sets $\tau^t = L(X_{\lfloor \rho N \rfloor})$ to be the smallest level-set that includes top $\rho$ percentile of samples, and then use $X_{\text{elite}}$ to represent $h_{\tau^t}(x).$ OCE then uses Eq (1.3) to calculate $f^t.$

We use a parametric approach by restricting $f$ to a parametric family $\mathcal{F} = \{f_v(x)\}_v.$ For combinatorial optimization, this family represents a probability mass function (pmf). In this parametric representation, the empirical counterpart of Eq (1.3) at time-step $t$ simplifies to...

$$\hat{v}^t \doteq \arg\max_v \left\{ \frac{1}{N} \sum_{i=1}^N \mathbbm{1}_{\{L(X_i) < \tau^t\}} \ln f_v(X_i) \right\} = \arg\max_v \left\{ \sum_{X_i \in X_{\text{elite}}} \ln f_v(X_i) \right\},$$

where $X_i \sim f_{\hat{v}^{t-1}}(x)$ are the instances generated from the distribution with minimum cross entropy to the previous level-set $\tau^{t-1}.$ The solution to Eq (1.5) is the maximum likelihood estimate of

$^3$ Notice that SA differs from CE in bounding the distribution strictly to the loss function, which appears in the Boltzmann distribution that it is sampling from. This is not the case with CE, as the error function does not appear as a parameter of any distribution that CE defines over domain $\mathcal{X}.$

$^4$ Botev et al. introduce a generalized CE method with a non-parametric approach [7].

$^5$ When $\mathcal{X} = \{1, \ldots, q\}$ is discrete, $v = \{v(1), \ldots, v(q)\}$ (when $\sum_i v(i) = 1$) defines a pmf $f_v(X = i) = v(i)$ over $\mathcal{X}.$
A generalization of the maximum entropy (a.k.a. MaxEnt) framework [20] is the MinxEnt, minimum cross entropy framework [25]. In MinxEnt we are interested in a distribution \( f(\cdot) \) with minimum cross entropy to a given distribution \( h(\cdot) \) that at the same time satisfies an expectation. This is given by the following program:

\[
\begin{align*}
\min_{g} & \quad \int_{X} g(x) \ln g(x) \, dx \\
\text{s.t.} & \quad \mathbb{E}_{g}[L(x)] = \tau
\end{align*}
\]

(1.6)

when \( f(\cdot) \) is also forced to be a distribution. For a uniform prior \( h(\cdot) \), MinxEnt and MaxEnt are the same. In general, the solution to the MinxEnt program of Eq (1.6) is [5]:

\[
\begin{align*}
f(x) &= \frac{h(x) e^{-\frac{L(x)}{\lambda}}}{\mathbb{E}_{h} e^{-\frac{L(x)}{\lambda}}} \\
\end{align*}
\]

(1.7)

where \( \lambda \) is the temperature parameter given by 6:

\[
\mathbb{E}_{h} \left[ L(x) e^{-\frac{L(x)}{\lambda}} \right] = \tau
\]

(1.8)

Inspired by batch sampling of the OCE method, and in contrast to sequential sampling methods like SA, one could use this kind of sampling to solve the optimization problem suggested by MinxEnt. Starting from Eq (1.6) one may update the level-set and the prior adaptively, using

\[
\nu^{t} = \arg\min_{\nu} \left\{ \int_{X} f_{w}(x) \ln \frac{f_{w}(x)}{f_{w-1}(x)} \right\} \\
\text{s.t.} \quad \mathbb{E}_{\nu^{t}}[L(x)] = \tau^{t}
\]

(1.9)

This method, called Minimum Cross Entropy (MCE), was introduced by Rubinstein [37] for combinatorial optimization. According to Rubinstein, the stochastic counterpart of program Eq (1.9) after

---

6Here substituting \( h(\cdot) \) by a uniform distribution reasonably gives the answer to MaxEnt program.
simplification is:

\[
v^t = \arg\max_w \left\{ \sum_{i=1}^{N} f_w(X_i) \ln f_w(X_i) \right\}
\]

(1.10)

s.t. \[ \sum_{i=1}^{N} f_{v^t}(X_i) L(X_i) = \tau^t \]

when \( X_i \sim f_{v^t-1} \) and \( \tau^t \) is adaptively set to the mean of the loss for elite samples in each iteration – i.e., in contrast to CE for which \( \tau^t \) refers to the best level-set that includes all the elite-samples here \( \tau^t \) is referring to the level-set of the average loss of elite samples. The optimization Eq (1.10) has an analytical solution for combinatorial problems, when the domain \( \mathcal{X} \) is discrete and \( f_v(\cdot) \) is a probability mass function. This method differs from SA by using batch sampling, adaptive selection of temperature, and minimization of cross-entropy to the previous estimate rather than a uniform distribution.

Rubinstein also introduced an alternative for MCE, called iterative Indicator-based MinxEnt (iterative IME) [40, 39]. In this method the constraint on the expected value of loss is generalized to the expected value (denoted by \( \gamma \)) of an arbitrary indicator or sum of indicators. For a single indicator, the new constraint reads as

\[
\mathbb{E}_{v^t}[\mathbb{I}(L(X) \leq \tau^t)] = \gamma^t
\]

(1.11)

where \( \gamma^t \) is basically the probability of samples belonging to a desired level-set. Furthermore while MCE minimizes the cross-entropy distance with respect to \( f_{v^t} \), iterative IME minimizes the cross-entropy distance to the original (flat) prior \( f_{v^0} \), which is the same as maximizing the entropy, performed by Simulated Annealing.

Generally we see a similar approach in all of these stochastic optimization methods: They all involve iterative minimization of the divergence of a distribution to a better distribution that is empirically represented by elite samples. In this thesis, we carry this idea to address partially decomposable problems, choosing Ordinary Cross Entropy (OCE) as our base. However it seems possible to use our idea (presented in the next chapter) with any of the stochastic optimization methods introduced in this section.
Chapter 2

Partially Decomposable Optimization

Many real-world optimization problems are (partially) decomposable to a set of ‘simpler’ subproblems, typically each with fewer variables. In convex optimization this is addressed by decomposition methods [8]. Alternatively, this simplification may be possible as a result of decomposition to sub-problems of simpler nature defined on the whole set of variables. An example of this for deterministic case, is when the loss function is a linear combination of some convex functions. This is addressed by D.C. (Difference of Convex) optimization [19].

Here we concentrate on the case when simplification is a result of reducing the number of variables in the sub-problems. As problem domains grow exponentially in the number of variables, this reduction in the number of variables can be very advantageous.

As an example, consider the Sudoku puzzle. Sudoku is a well-known 21st century puzzle, which is also known to be NP-Complete [22]. The problem is defined as arranging the numbers from 1 to \( r^2 \) in each row, each column and each of a set of a specified disjoint small \( r \times r \) square, inside the partially occupied \( r^2 \times r^2 \) square.

![A solved Sudoku puzzle for \( r = 3 \). Red figures show the solution. Each dotted oval is an example of a sub-problems.](image)

Figure 2.1: A solved Sudoku puzzle for \( r = 3 \). Red figures show the solution. Each dotted oval is an example of a sub-problems.
We define the objective to be maximized as the distinct numbers in each row, column and small square. The size of the problem domain is \((r^2)^{(r^2-c)}\) (when \(c\) is the number of squares occupied when the problem is given) However if instead we consider the number of distinct elements in each row, column or small square as a sub-problem, each optimization domain has a size of \((r^2)^{(r^2-c)}\). Figure 2.1 shows a solved example for \(r = 3\). If we could solve each sub-problem independently we could have in the order of a \(\sim 1.87 \times 10^{67}\) fold reduction of the problem size, for \(r = 3\).

However here we cannot consider each sub-problem independently as they have variables in common. We call such problems partially decomposable. The fact that sub-problems are entangled with each other means it is not sufficient to address each sub-problem separately.

Our approach is to allow each sub-problem to provide a distribution over the values of variables involved in that specific sub-problem. We suggest a method to combine several distributions over each variable. We will perform this in iterations and use the final distribution to determine the optimal value of each variable.

## 2.1 Definition and Notation

Let \(\mathcal{X}^d\) be a bounded \(d\)-dimensional domain and for each \(i \in \{1, \ldots, d\}\),

\[
v_i = [v_i(1), \ldots, v_i(q)] \in \mathbb{R}^q
\]  

(2.1)

is a row vector.

Let \(\mathcal{F}\) be a joint distribution of a specific parametric family defined over dimensions of \(\mathcal{X}^d\):

\[
f_v(x) = \prod_{i=1}^{d} f_{v_i}(x_i),
\]

where \(v_i\) is the parameter vector of the distribution \(f_{v_i} : \mathcal{X} \to \mathbb{R}\), and \(v = [v_1; \ldots; v_d] \in \mathbb{R}^{dq}\) is the parameter vector of \(f_v : \mathcal{X}^d \to \mathbb{R}\).

For combinatorial optimizations, that is when \(\mathcal{X} = \{1, \ldots, q\}\) is discrete, \(f_v\) is a probability mass function (pmf) over \(\mathcal{X}\). In this case \(Pr(x_i = j) = f_{v_i}(x_i) = v_i(j)\), where \(v_i\) is located on a simplex — i.e., \(\sum_{j=1}^{q} v_i(j) = 1\) and \(v_i(\cdot) \geq 0\).

In the 9 Sudoku domain \(f_{v_i}\) is a pmf over the numbers 1 to 9 in each square. Here \(x \in \{1, \ldots, 9\}^{81}\) and for example \(v_{7,5}(2) = 0.4\) means that the probability that the square at location \((7, 5)\) has the value 2, in 0.4. (In general, we consider \(v_{i,j}(k)\) where each of \(i, j, k \in \{1, 2, \ldots, 9\}\), with the obvious constraint that \(\sum_k v_{i,j}(k) = 1\). (We will typically write \(v_{i,j}(\cdot)\) with a single subscript \(v_{a}(\cdot)\), where of course “\(a\)” refers to the “\((i, j)\)” location.)

Let \(x_{A}\) be the restriction of \(x\) to the coordinates in \(A \subseteq \{1, \ldots, d\}\) and let \(v_A\) show the same restriction on \(v\). – perhaps \(A = 1, 2, 3, \ldots, 9\) referring to the first row, or \(A = 1, 10, 19, \ldots 73\) referring to the first column – Using this notation, \(f_{v_A}(x_{A}) = \prod_{i \in A} f_{v_i}(x_i)\) is a joint product distribution defined over \(\mathcal{X}_{A}^d\), the restriction of \(\mathcal{X}^d\) to \(A\). This is basically the marginal distribution of \(f_v\) over variables in \(A\).
Table 2.1: A Summary of notation.

<table>
<thead>
<tr>
<th>Notation</th>
<th>meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f^t$, $τ^i$, $v^t$, $v^j$</td>
<td>superscript $t$: a variable or function at time-step $t$</td>
</tr>
<tr>
<td>$\hat{v}_i$, $\hat{v}_i$, $x_i$</td>
<td>subscript $i$: $i^{th}$ component of a vector</td>
</tr>
<tr>
<td>$M(k)$</td>
<td>indexes the variables in $k^{th}$ sub-problem; $M(k) \subseteq {1, \ldots, d}$</td>
</tr>
<tr>
<td>$A(i)$</td>
<td>indexes the sub-problems involving $i^{th}$ variable; $A(i) \subseteq {1, \ldots, p}$</td>
</tr>
<tr>
<td>$\hat{v}_{A,i}$, $x_A$, $x_A$</td>
<td>subscript $A$: restriction of a vector to components indexed by the set $A$</td>
</tr>
<tr>
<td>$τ_{ik}$</td>
<td>second level-set for sub-problem $k$ at time $t$</td>
</tr>
<tr>
<td>$\hat{v}_{i,k}$</td>
<td>second subscript $k$: estimate of $v_i$ given by $k^{th}$ sub-problem, when $i \in M(k)$</td>
</tr>
<tr>
<td>$\hat{v}_{A,k}$</td>
<td>second subscript $k$: estimate of $v_A$ given by $k^{th}$ sub-problem, when $A \subseteq M(k)$</td>
</tr>
</tbody>
</table>

For each $k \in \{1, \ldots, p\}$, let $M(k) \subseteq \{1, \ldots, d\}$ index the set of variables in the $k^{th}$ group (sub-problem)– (e.g., this could be the variables involved in third row of Sudoku puzzle, second column or first small square). For any loss function of the form $L(x) = \sum_{i=1}^{p} L_i(x_{M(i)})$ and for any value $τ$, we necessarily have $\|L(x) < τ\| = \prod_{i=1}^{p} \|L_i(x_{M(i)}) < τ_i\|$ for some $\{τ_i\}$, (here, we need $\sum_i τ_i \leq τ$). We can therefore try to solve the optimization Eq (1.5) by finding assignments that produce good results for all sub-problems at the same time.

Given this notion of a sub-problem, we can show the interaction of variables in sub-problems using the coupling matrix $C \in \{0, 1\}^{p \times q}$, whose rows correspond to sub-problems and columns to variables, where the element $C_{j,k}$ is non-zero if the sub-problem $j$ depends on variable $x_k$. We then define $M(k) = \{i : C_{i,k} \neq 0\}$ to index the variables involved in sub-problem $k$ and $A(i) = \{k : C_{i,k} \neq 0\}$ to index sub-problems that involve variable $x_i$. Table 2.1 summarizes the notation.

So for the Sudoku puzzle, each variable belongs to three sub-problems– i.e., the corresponding row, column and small square, and so $A(\{3, 2\}) = \{3^{rd}$ row, $2^{nd}$ column, $1^{st}$ small square $\}$. The 3rd row includes many variables – $M(3^{rd}$ row $) = \{x_{3,1}, x_{3,2}, \ldots, x_{3,9}\}$.

Figure 2.2a shows the coupling matrix for a $9 \times 9$ Sudoku problem (see Section 2.4.2). Figures 2.2b and 2.2c show the coupling matrix for a random SAT and a SAT formulated graph coloring problem (see Section 2.4.1). Finally, Figure 2.2d shows the coupling matrix for our NMR Spectra interpretation task; see Chapter 3.

2.2 Cross Entropy Exploiting partial Decomposability (CEED) method

The basic idea underlying the CEED method is to first obtain several estimates of each variable using interrelated loss functions, using an approach similar to OCE, and then combine these estimates, and use the combined distribution for sampling in the next iteration. We can summarize CEED algorithm in the following steps:

**Require:** A coupling matrix $C$, a set of loss functions $\{L_i(\cdot) : \mathcal{X}^{|M(i)|} \rightarrow \mathbb{R}\}$ and a family of
Figure 2.2: Coupling matrices for (a) Sudoku, (b) Random SAT, (c) Structured SAT, (d) NMR problem
probability distributions \( \mathcal{F} = \{ f_v \} \).

1. Set \( t = 1 \) and specify a (uniform) prior \( f_{\hat{v}^0} \).

2. At time-step \( t \), sample the joint distribution \( f_{\hat{v}^{t-1}_{\mathcal{M}(k)}} \) for each sub-problem \( k \). (Recall \( \hat{v}_{\mathcal{M}(k)} \) is the restriction of \( \hat{v} \) to the variables in sub-problem \( k \) and \( t \) refers to the iteration of the algorithm.) When sampling, choose the number of instances \( N_k \) based on the difficulty of sub-problem \( k \).

3. For each sub-problem \( k \), calculate the loss for the instances, and find a joint distribution with the least cross entropy to the elite samples. More formally, similar to Eq (1.5), define, for each \( k \in \{1, \ldots, p\} \),

\[
\hat{v}^t_{\mathcal{M}(k), k} = \arg \max_v \left\{ \frac{1}{N_k} \sum_{i=1}^{N_k} I_{\{L_k(X_i) < \tau_t^k\}} \ln f_{v_{\mathcal{M}(k)}}(X_i) \right\} \quad \text{where} \quad X_i \sim f_{\hat{v}^{t-1}_{\mathcal{M}(k)}} \tag{2.2}
\]

Here, \( \hat{v}^t_{\mathcal{M}(k), k} \) is the estimate of \( v^*_{\mathcal{M}(k)} \) for all of the variables in \( \mathcal{M}(k) \), given by sub-problem \( k \) at time-step \( t \).

4. Each variable \( v_i \) appears in each of the sub-problems in \( \mathcal{A}(i) \), and so is estimated by multiple parameter values \( \{ \hat{v}^t_{i, k} \mid k \in \mathcal{A}(i) \} \) — one for each sub-problem \( k \in \mathcal{A}(i) \). We therefore “combine” the associated distributions \( \{ f_{\hat{v}^t_{i, k}} \mid k \in \mathcal{A}(i) \} \) to produce a combined parameter value \( \hat{v}^t_i \) (i.e., combined distribution \( f_{\hat{v}^t_i} \)). We then consider the joint distribution over all variables given by \( f_{\hat{v}^t} \), when \( \hat{v}^t = [\hat{v}^t_1, \ldots, \hat{v}^t_d] \).

5. If not converged, increase \( t \) and return to step 2. Otherwise stop.

Return: Mode of \( f_{\hat{v}^t} \) as an approximation to \( x^* \), the minimizer of the sum of the loss functions.

For our Sudoku example, after sampling a distribution, and assigning a distribution to the variables in each sub-problem, we get several distributions over each variable \( x_i \) (each square of the puzzle), each of which is basically the marginal of the joint distribution over all variables for one sub-problem (So for the \((3, 2)\) variable \( x_{3,2} \), we have one distribution based on the third row, another distribution based on the second column, and a third based on the top-left \(3 \times 3\) square.)

We then combine these marginal distributions to get a single distribution over each variable, the joint distribution of which we then use in the next iteration.

In terms of the general coupling matrix \( C \), each CEED iteration first estimates the parameters in each row \( \{ \hat{v}^t_{\mathcal{M}(k), k} \}_{1 \leq k \leq p} \) (each corresponding to a single sub-problem — e.g., 3rd row), then combines the estimates in each column, \( \{ \{ \hat{v}^t_{i, k} \}_{k \in \mathcal{A}(i)} \}_{1 \leq i \leq d} \) (each corresponding to a single variable — e.g., \( x_{3,2} \)) to get \( \hat{v}_i \). This is repeated in each iteration, using the previous estimate as the sampling distribution.

Step 4 needs to combine several estimates. In the next section we elaborate on this task.
2.3 Combining Several Maximum Likelihood Estimations

Once we have decomposed the problem, one could think of calculating a combined distribution directly from elite samples of related sub-problems. That is for each $i$, we could look for a distribution $f_{\hat{v}_t^i}(x)$ that minimizes the KL-divergence to elite samples from all relevant sub-problems, $j \in \mathcal{A}(i)$:

$$
\hat{v}_t^i \doteq \arg \max_\omega \left\{ \sum_{j \in \mathcal{A}(i)} \sum_{k=1}^{N_j} \mathbb{I}(L_j(M(j)) < \tau_j) (X^{j,k}) \ln (f_\omega(X^{j,k})) \right\} \text{ where } X^{j,k} \sim f_{\hat{v}_t^{i-1}}^{M(j)}
$$

(2.3)

As a reminder, $i$ in $X^{j,k}_i$ is a sub-coordinate notation and it is refering to the element of sample vector $X^d$, sampled from $i^{th}$ coordinate of $X^d$. So for Sudoku puzzle, sample vectors $X^{2,k}$ (of length 9 for each $k$), are from joint distribution over the variables in the second row (as $2^{nd}$ sub-problem) and $X^{2,k}_2$ (or $X^{2,k}_3$, as a compressed notation) is the third element of this vector.

However this method of combination does not work well in practice. This is because it does not properly take into the account the certainty of different sub-problems in the estimates of one variable. This happens because all the samples from the different sub-problems would appear with the same weight in the optimization of Eq (2.3).

We now consider the case when have several estimates $\{\hat{v}_{t,k}^i\}_{k \in 1,\ldots,p}$, of variable $i$ that we need to combine. We may aim for a combined estimate that minimizes the sum of some distance to the given set of estimates. Different choices of this distance however may drastically change the final result. For example, if we consider the Euclidean distance between the estimates, the value that minimizes that distance is the mean of estimates:

$$
\hat{v}_t^i \doteq \arg \min_\omega \left\{ |\mathcal{A}(i)|^{-1} \sum_{k \in \mathcal{M}(i)} \hat{v}_{t,k}^i \right\} = \mathbb{I}(\omega)^{-1} \sum_{k \in \mathcal{A}(i)} \hat{v}_{t,k}^i
$$

Alternatively, the weighted distance minimization,

$$
\hat{v}_t^i \doteq \arg \min_\omega \left\{ \mathbb{I}(\omega)^{-1} \sum_{k \in \mathcal{A}(i)} (\omega - \hat{v}_{t,k}^i) ((\hat{\sigma}_{t,k}^i)^2)^{-1} (\omega - \hat{v}_{t,k}^i)^T \right\}
$$

(2.4)

where $(\hat{\sigma}_{t,k}^i)^2$ is the estimated variance (or variance-covariance matrix) of $\hat{v}_{t,k}^i$ is a well-known method, and it is known to give the combination with lowest variance in linear regression models with uncorrelated errors [46]. Using this, the answer to this optimization is:

$$
\hat{v}_t^i \doteq \left( \sum_{k \in \mathcal{A}(i)} \hat{v}_{t,k}^i ((\hat{\sigma}_{t,k}^i)^2)^{-1} \right) \left( \sum_{k \in \mathcal{A}(i)} ((\hat{\sigma}_{t,k}^i)^2)^{-1} \right)^{-1}
$$

(2.5)

This method, known as Mixture Using Variance (MUV), needs an estimate of each estimator’s variance, which could be costly or inaccurate in practice.
Alternatively we may consider a measure of divergence defined for corresponding distributions, rather than the parameters. CEED uses the distribution that minimizes the sum of the KL-divergences:

\[
\hat{v}_t^i = \arg \min_{\omega} \left\{ \sum_{k \in A(i)} D_{KL}(f_\omega(x), f_{\hat{v}_t^i,k}(x)) \right\} = \arg \min_{\omega} \left\{ \sum_{k \in A(i)} \left( \int_{x \in X} f_\omega(x) \ln \left( \frac{f_\omega(x)}{f_{\hat{v}_t^i,k}(x)} \right) dx \right) \right\}
\]

(2.6)

Although the optimization of Eq (2.6) is convex, in general it is difficult to obtain an analytic solution here. The next section therefore provides a linear combination method to approximate this solution to Eq (2.6). We believe this is a novel method for combining a set of maximum likelihood estimators.

### 2.3.1 Combining Estimators Using Their Fisher Information

Given a set of random independent and identically distributed samples \( X = \{X^1, ..., X^N\} \), the score function is defined as the gradient of the log-likelihood function:

\[
U(v, X) = \frac{\partial \log(L(v; X))}{\partial v},
\]

where \( L(v, X) = \prod_{i=1}^{N} f_i(X^i) \) is the likelihood function. This is the zero vector for the maximum-likelihood (ML) parameters. Since for ML estimates \( \mathbb{E}\{U(v, X)|v\} = 0 \), the variance of the score function is the quantity of our interest. Here it is score functions’ second moment, a.k.a. its Fisher information [27]:

\[
\mathcal{I}(v) = \mathbb{E}\{U(v, X)^2|v\} = \mathbb{E}\left\{ \left[ \frac{\partial}{\partial v} \log(f_\omega(X)) \right]^2 | v \right\}
\]

If the distribution family \( \mathcal{F} \) satisfies the regularity condition \( \int \frac{\partial^2}{\partial v^2} f_\omega(x) dx = 0 \), then we can also write the Fisher information as:

\[
\mathcal{I}(v) = -\mathbb{E} \left\{ \frac{\partial}{\partial v} U(v, X) | v \right\} = -\mathbb{E} \left\{ \frac{\partial^2}{\partial v^2} \log(f_\omega(X)) | v \right\}
\]

The element \( \mathcal{I}_{i,j}(\hat{v}) \) basically shows the rate of change in the maximum likelihood estimation of parameter \( v(i) \) by changing the parameter \( v(j) \) in the neighborhood of \( \hat{v}(j) \). Therefore higher values suggest greater accuracy in the maximum likelihood estimation. One may understand this score function and Fisher information in analogy with the first and second derivative of a function:

Suppose we are estimating a local optima by taking samples in the neighborhood of that point. Larger values of the second derivative suggest more accurate estimation of local optima by sampling in its neighborhood. This makes the basis of our approach for combining estimations; we combine the estimations, \( \{\hat{v}_{i,j}^t\} \), by weighting the estimates with the Fisher information matrices \( \mathcal{I}(\hat{v}_{i,j}^t) \):

\[
\hat{v}_t^i = \left( \sum_{j \in A(i)} \hat{v}_{i,j}^t \mathcal{I}(\hat{v}_{i,j}^t) \right) \left( \sum_{j \in A(i)} \mathcal{I}(\hat{v}_{i,j}^t) \right)^{-1}
\]

(2.7)

Considering this method of combination, Algorithm (2) summarizes the CEED method.
Algorithm 2 Cross Entropy Exploiting partial Decomposability (CEED) method

Require: Set of loss functions $L_i(\cdot)$, coupling matrix $C$ which provides $\mathcal{A}(i)$ and $\mathcal{M}(k)$, learning rate $\zeta$, the percentage of elite samples $\rho$, and the number of samples for each loss function in each iteration $N(k,t)$.

$t \leftarrow 0$
Start from a prior $v^t$

repeat
  for $k = 1$ to $p$ do
    $t \leftarrow t + 1$
    Draw samples: $\{X_1, \ldots, X_{N(k,t)}\} \sim f_{v^{t-1}}(x)$
    Calculate loss for all samples: $\{L_k(X_j)\}_{1 \leq j \leq N(k,t)}$
    Calculate $\tau^t_k$ as the maximum loss of $\rho\%$ top samples.
    Calculate $\hat{v}^t_{i,k}$ from Eq (2.2).
  end for
  $\forall i$, calculate $\hat{\nu}^t_i$ by combining $\{\hat{\nu}^t_{i,k}\}_{k \in \mathcal{A}(i)}$ e.g., using Eq (2.7) or Eq (2.6).
  Set $\hat{\nu}^t \leftarrow [\hat{v}^t_1, \ldots, \hat{v}^t_d]$ as a joint distribution.
  Update using a learning rate($\zeta$): $\hat{\nu}^t \leftarrow \zeta \hat{\nu}^t + (1 - \zeta)\hat{\nu}^t$
  until convergence
  $\hat{x}^* \leftarrow \text{mode}(f_{v^t}(x))$
return $\hat{x}^*$

2.3.2 Relation of different combination methods

Combining the likelihood estimators using their Fisher information is interestingly connected to both Mixture Using Variance Eq (2.5) and combination by minimization of the sum of KL-divergences Eq (2.6).

We start with the relation to MUV. The Cramér-Rao theorem [11, 34] states that any unbiased estimator $\hat{v}_i(X)$ of parameter $v^*_i$, with variance-covariance matrix $Var(\hat{v}_i)$ satisfies the following inequality:

$$Var(\hat{v}_i) \geq \mathcal{I}^{-1}(v^*_i)$$  \hspace{1cm} (2.8)

when the inequality means that the difference $Var(\hat{v}_i) - \mathcal{I}^{-1}(v^*_i)$, is positive semi-definite. It is known that ML estimators are asymptotically efficient, which means the inequality above becomes equality, when the sample size grows to infinity [14].

Therefore for a relatively large sample size, we can assume that inverse of the Fisher information matrix is a good representative of variance of ML estimates, and therefore by approximating MUV in Eq (2.5) we have:

$$\hat{v}^t_i = \left( \sum_{k \in \mathcal{A}(i)} \hat{v}^t_{i,k} (\hat{v}^t_{i,k})^{-1} \right) \left( \sum_{k \in \mathcal{A}(i)} (\hat{v}^t_{i,k})^{-1} \right)^{-1}$$  \hspace{1cm} (2.9)

$$\approx \left( \sum_{k \in \mathcal{A}(i)} \hat{v}^t_{i,k} \mathcal{I}(\hat{v}^t_{i,k}) \right) \left( \sum_{k \in \mathcal{A}(i)} \mathcal{I}(\hat{v}^t_{i,k}) \right)^{-1}$$  \hspace{1cm} (2.10)

Moreover we know that the inequality Eq (2.8) holds as an equality for the mean estimator — that is, for a set of samples, the estimation of sample mean is efficient. This implies that if the
mean of a distribution appears as a parameter in a parameter vector $v$, the combination suggested by Eq (2.7) gives the same mean as Eq (2.5) in which the variance of estimation of $v$ is known. This is the case when we estimate the mean of a Gaussian distribution.

For example for a single-variate Gaussian family, $f_{[\mu, \sigma^2]}(x) = \frac{e^{-(x-\mu)^2/2\sigma^2}}{\sqrt{2\pi\sigma^2}}$, the the score function is given by

$$U([\mu, \sigma^2], x) = \frac{\partial \ln(f_{[\mu, \sigma^2]}(x))}{\partial [\mu, \sigma^2]} = \left[ \frac{x - \mu}{\sigma^2}, \frac{(x - \mu)^2 - \sigma^2}{2\sigma^4} \right]$$

from which we can calculate the Fisher information matrix:

$$\mathcal{I}([\mu, \sigma^2]) = -\mathbb{E} \left\{ \frac{\partial U([\mu, \sigma^2], x)}{\partial [\mu, \sigma^2]} \right\} = -\mathbb{E} \left\{ \begin{bmatrix} -\frac{x+\mu}{\sigma^4} & -\frac{x+\mu}{\sigma^4} \\ -\frac{x+\mu}{\sigma^4} & -2(x-\mu)^2 + \sigma^2 \end{bmatrix} \right\} = \begin{bmatrix} \frac{1}{\sigma^2} & 0 \\ 0 & \frac{1}{2\sigma^4} \end{bmatrix}$$

which then we use to combine different estimates:

$$[\hat{\mu}_i, (\hat{\sigma}_i^2)] = \left( \sum_{k \in A(i)} \begin{bmatrix} \mu_{i,k}^4 & \sigma_{i,k}^4 \\ \sigma_{i,k}^4 & \frac{1}{2\sigma_{i,k}^4} \end{bmatrix} \right) \left( \sum_{k \in A(i)} \begin{bmatrix} \frac{1}{\sigma_{i,k}^4} & 0 \\ 0 & \frac{1}{2\sigma_{i,k}^4} \end{bmatrix} \right)^{-1}$$

which is basically:

$$\begin{align*}
\hat{\mu}_i &= \frac{\sum_{j \in A(i)} \mu_{i,j}^4 (\sigma_{i,j}^4)^2}{\sum_{j \in A(i)} (\sigma_{i,j}^4)^2} \\
\hat{\sigma}_i^2 &= \frac{\sum_{j \in A(i)} (\sigma_{i,j}^4)^2}{\sum_{j \in A(i)} (\sigma_{i,j}^4)^2} 
\end{align*} \tag{2.11}$$

To see the relation of linear combination using Fisher information with the minimization of Eq (2.6), consider the family $\mathcal{F} = \{ f_v(x) \}$ as a manifold parametrized by $v$. Each point on this manifold represents a probability distribution. Let $T(v)$ denote the tangent space of $\mathcal{F}$ at point $v$, which is the linear space spanned by $\{ \frac{\partial f_v(X)}{\partial v_i} \}$. There is a natural isomorphism between this tangent space and the vector space $T^{(1)}(v)$ spanned by random vectors, $\{ \frac{1}{\sigma_{i,j}} \log(f_v(X)) \}$. The inner product for basis vectors of such space is defined as:

$$\mathcal{I}_{i,j}(v) = \mathbb{E} \left[ \frac{\partial}{\partial v_i} \log(f_v(X)) \frac{\partial}{\partial v_j} \log(f_v(X)) \right]$$

which basically serves as a Riemannian metric on this manifold and is equal to the Fisher information (see [3]). Given the inner product one may define the length of a path on the manifold. The differential distance along a differential vector $dv$, is given by:

$$ds^2 = dv^T \mathcal{I}(v) dv$$

This is related to the KL-divergence by [25]:

$$2\mathcal{D}_{KL}(f_v, f_{v+dv}) = dv^T \mathcal{I}(v) dv$$

1KL-Divergence in general corresponds to integration of the differential distance $ds$, along specific path on the statistical manifold[3, 4].
Figure 2.3: This figure graphically shows how Fisher information combines two Beta distributions. In (a), \([\alpha_1,\beta_1] = [1, 6], [\alpha_2,\beta_2] = [5, 2]\) gives \([\alpha, \beta] = [.54, .74]\) and in (b), \([\alpha_1,\beta_1] = [12, 13], [\alpha_2,\beta_2] = [5, 10]\) resulted in \([\alpha, \beta] = [3.84, 6.08]\).

This provides us with an approximation of KL-divergence, which we use to approximate the optimization of Eq (2.6):

\[
\hat{v}_t^i = \arg \min_\omega \left\{ \sum_{j \in A(i)} D_{KL}(f_\omega(x), f_{\hat{v}_t^j}(x)) \right\} \approx \arg \min_\omega \left\{ \sum_{j \in A(i)} \frac{1}{2}(\hat{v}_{t,j}^i - \omega) I(\hat{v}_{t,j}^i)(\hat{v}_{t,j}^i - \omega)^T \right\}
\]

This minimization has the analytical solution, in the form of a linear combination, given by Eq (2.7).

Returning to our example of combining Gaussian distributions, we compare the answer to the optimization Eq (2.6) and combination Eq (2.7). A simple calculation shows for Gaussian family, the optimization Eq (2.6) for arbitrary number of terms has a nice solution:

\[
\hat{\mu}_t^i = \frac{\sum_{j \in A(i)} \hat{\mu}_{t,j}^i}{\sum_{j \in A(i)} (\hat{\sigma}_{t,j}^i)^2}
\]

\[
\hat{\sigma}_t^i = \frac{|A(i)|}{\sum_{j \in A(i)} (\hat{\sigma}_{t,j}^i)^2}
\]

Comparing Eq (2.12) and Eq (2.11) we can see the reasonable approximation given by Eq (2.7).

Figure 2.3 graphically shows combination of two Beta distributions using their Fisher information.

### 2.4 Examples

In this section we present two applications of the CEED method, using the notation already established in the previous section. These problems help to demonstrate the generality of our approach.

#### 2.4.1 SAT Problem

The SAT (or Boolean Satisfiability) problem is defined as: Given a Conjunctive Normal Form formula (over \(d\) variables with \(p\) clauses), find a Boolean assignment of the variables that satisfied every clause. The following Boolean expression

\[
SAT(x) = (x_1 \lor \neg x_2) \land (x_2 \lor \neg x_3) \land (\neg x_3 \lor \neg x_1) \land (x_1 \lor x_2)
\]
corresponds to a simple SAT problem. Here, the assignment \( \{ x_1 = true, x_2 = true, x_3 = false \} \) is a solution, as this makes each of the 4 clauses true — as \( x_1 = true \), the first clause is satisfied, as \( x_2 = true \), the second and last clauses are satisfied, and \( x_3 = false \) means the 3rd clause is satisfied.

This SAT decision problem — of finding an assignment that satisfies all \( p \) of the \( p \) clauses — is known to be NP-hard \([17]\). The MaxSAT problem is the related optimization problem: find the assignment that satisfies the most clauses. Here, the assignment given above satisfies all \( p = 4 \) clauses. In some problem, there might be no assignment that satisfies all \( p \) clauses.

Many different variations of CE method have been applied to the SAT problem \([39, 37, 6, 38]\). In these applications we are interested in an importance sampling distribution \( f_v^* \) that increases the chance of a rare event (see Appendix A for an introduction to CE from this perspective). For the SAT problem this rare event is the event of an assignment satisfying every clause. Here we confine ourselves to the corresponding optimization problem — i.e., maximizing the number of satisfied clauses a.k.a. MaxSAT.

Let the SAT matrix \( A = (A_{i,j}) \in \{+, -, Null\} \) represent how variable \( j \) participates in clause \( i \). When every clause is considered as a sub-problem, the coupling matrix \( C \) has the same form as \( A \). The domain of the problem is \( \mathcal{X}^d = \{+,-\}^d \), which has a Bernoulli distribution (as a pmf) over each variable, that is

\[
f_v(x) = \begin{cases} v_i(1) & x_i = + \\ v_i(2) = 1 - v_i(1) & x_i = - \end{cases}
\]

or

\[
f_v(x) = (v_i)^{\frac{x_i}{2}}(1 - v_i)^{1 - x_i} \text{ when } \mathcal{X} = \{+1, -1\}
\]

(Notice that here superscripts are exponents and do not refer to the time-step.) The general objective function, to be maximized, is the number of satisfied clauses

\[
L(x) = \sum_{k=1}^{p} L_k(x_{M(k)}) = \sum_{k=1}^{p} I_{\{\bigvee_{i\in M(k)} A_{k,i} x_i \}}(x) \tag{2.15}
\]

Here \( \bigvee_{i\in M(k)} A_{k,i} x_i \) is the evaluation of a single clause, with \( A_{k,i} x_i \) being a literal of the \( k \)th clause (\( A_{k,i} \) is a sign and \( x_i \) a Boolean variable).

When applied to SAT, CEED first finds the maximum likelihood Bernoulli distribution over variables in each clause, given the prior (i.e., sampling distribution) and the fact that the clause was satisfied. Then for each variable CEED combines several maximum likelihood Bernoulli distributions from different clauses to arrive at a new prior for the next iteration. After some iterations the mode of the Bernoulli distribution over each variable is reported as the value of that variable.

An interesting property of CEED here is that, since each sub-problem is very simple, we can perform the update of Eq (1.3) analytically rather than empirically (as in Eq (2.2)). That is for each sub-problem (i.e., clause) given a prior \( v^{t-1}_{M(k)} \), one may calculate \( v^t_{M(k),k} \), the solution to Eq (2.2)
analytically. Since Eq (2.2) and Eq (1.3) are finding the maximum likelihood estimate for elite samples, they correspond to the ML parameter for satisfied clauses in a SAT problem.

Let \( w_{i,k}^{t-1} \) be the probability of literal \( A_{k,i} \) being true in clause \( k \)—i.e., \( w_{i,k}^{t-1} \) is defined for every literal of every clause:

\[
\begin{align*}
\hat{v_1}^{t-1}_{i,k} & = \begin{cases} 
\hat{v_1}^{t-1}_{i,k} & A_{k,i} = + \\
\hat{v_2}^{t-1}_{i,k} & A_{k,i} = - 
\end{cases} 
\end{align*}
\]  

(2.16)

and let \( w_{i,k}^t \) be the probability of the same literal being true, given that clause \( k \) was satisfied. The probability of \( k^{th} \) clause being satisfied is \( 1 - \prod_{j \in \mathcal{M}(k)} (1 - w_{j,k}^{t-1}) \), therefore

\[
\begin{align*}
\hat{v_2}^{t}_{i,k} & = \frac{w_{i,k}^{t-1}}{1 - \prod_{j \in \mathcal{M}(k)} (1 - w_{j,k}^{t-1})} 
\end{align*}
\]  

(2.17)

Then we have:

\[
\hat{v_1}^t_{i,k} = \begin{cases} 
\hat{v_1}^{t-1}_{i,k} & A_{k,i} = + \\
1 - \hat{v_1}^{t-1}_{i,k} & A_{k,i} = - 
\end{cases} \quad \hat{v_2}^t_{i,k} = 1 - \hat{v_1}^t_{i,k} 
\]  

(2.18)

That is, we come up with the Bernoulli distribution with least cross entropy to the ideal level-set, analytically—i.e., without using any elite samples to empirically represent this distribution. The Fisher information for the Bernoulli distribution is

\[
\mathcal{I}(v_1) = \sum_{x \in \{-1, 1\}} \frac{\partial^2 \log \left( (v_1 x + 1 - v_1) x \right)}{\partial v_1^2} = \frac{1}{v_1(1 - v_1)} 
\]  

(2.19)

We use this to combine different estimates of each variable.

As an example consider the problem of Eq (2.14). It has the SAT matrix of the form

\[
A = \begin{pmatrix}
1 & -1 & 0 \\
0 & 1 & -1 \\
-1 & 0 & -1 \\
1 & 1 & 0 \\
\end{pmatrix}
\]

corresponding to the coupling matrix

We start with a Bernoulli distribution over all variables \((x_1, x_2, x_3)\). For example \( v_2^0(1) = v_2^0(2) = .5 \) are parameters of the Bernoulli distribution over \( x_2 \) at time 0. Then we consider each

\[\text{Remember that } v(j)^t_{i,k} \text{ is the } j^{th} \text{ element of the vector } v^t_{i,k}, \text{ which is the estimate of the } i^{th} \text{ variable (vector) given by } k^{th} \text{ sub-problem at time-step } t. \text{ Please refer to the Table 2.1}\]
clause and find the maximum likelihood distribution over each variable given that the clause was satisfied using Eq (2.16), Eq (2.17) and Eq (2.18). This yields the following values for \( \hat{v}(1)_{i,k} \), where \( 1 \leq k \leq 4 \) and \( 1 \leq i \leq 3 \) are indexing clauses and variables respectively:

\[
\hat{v}(1)_{i,k} = \begin{pmatrix}
0.6667 & 0.3333 \\
0.3333 & 0.6667 \\
0.3333 & 0.3333 \\
0.6667 & 0.6667
\end{pmatrix}
\]

Here for example variable \( x_2 \) with distribution defined by \( v_2 = [v_2(1), 1 - v_2(1)] \) has three estimates given by first, second and fourth clauses at this time-step \( (t = 1) \) and the estimate given by the fourth clause is \( \hat{v}(1)_{2,4} = 0.6667 \).

In the next step we combine all the different estimates for each variable using their Fisher Information. For all three different estimates of \( v_2(1) \) Fisher Information using Eq (2.19) is 4.5. Therefore for \( v_2 \) we get:

\[
\hat{v}(1)_{2} = \frac{0.3333 \times 4.5 + 0.3333 \times 4.5 + 0.6667 \times 4.5}{4.5 + 4.5 + 4.5} = 0.5556
\]

Repeating this procedure for all three variables we get the following new estimates: \( \hat{v}(1)_{1} = 0.5556, \hat{v}(1)_{2} = 0.5556, \hat{v}(1)_{3} = 0.3333 \). Even after the first iteration the mode of each Bernoulli distribution gives us the correct variable assignment—i.e., mode(\( \text{Bernoulli}_{i_1} \)) = true, mode(\( \text{Bernoulli}_{i_2} \)) = true and mode(\( \text{Bernoulli}_{i_3} \)) = false. However by each additional iteration the distribution more and more converges to the correct value. The following Table shows how the distribution evolves the 5 iterations.

<table>
<thead>
<tr>
<th>Variable/Time</th>
<th>t = 0</th>
<th>t = 1</th>
<th>t = 2</th>
<th>t = 3</th>
<th>t = 4</th>
<th>t = 5</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \hat{v}_1(1) )</td>
<td>.5000</td>
<td>.5556</td>
<td>.6402</td>
<td>.7343</td>
<td>.8198</td>
<td>.8856</td>
</tr>
<tr>
<td>( \hat{v}_2(1) )</td>
<td>.5000</td>
<td>.5556</td>
<td>.5907</td>
<td>.6109</td>
<td>.6209</td>
<td>.6248</td>
</tr>
<tr>
<td>( \hat{v}_3(1) )</td>
<td>.5000</td>
<td>.3333</td>
<td>.1984</td>
<td>.1002</td>
<td>.0399</td>
<td>.0115</td>
</tr>
</tbody>
</table>

For the sake of comparison, we apply OCE method, using the same objective function Eq (2.15). Figure 2.4 compares the evolution of CEED and OCE distributions for a random SAT problem\(^3\), with \( p = 325 \) and \( d = 75 \). Each column is represents the Bernoulli parameters, \( \{\hat{v}_i(1)\}_{1 \leq i \leq d} \) (black is zero and white is one). The horizontal axis is representing the evolution of these parameters in time.

The similarity of these distributions suggests that CEED, like OCE, could be used for probability estimation and counting. This is motivated by the considerable computational advantage of CEED. As a typical case, for the same SAT problem, CEED *without taking any samples* finds a distribution \( f_v \) such that \( \hat{x}^* = \text{mode}(f_v) \) satisfies all the clauses in 42 iterations while, OCE does not converge to the correct distribution. This is true over variety of conditions: \( N \in \{100, 1000, 10000\} \), elite percentile \( \rho \in \{.1, .05, .01\} \) and different learning rates.

\(^3\)SAT samples are from http://www.satlib.org
5

(a)

(b)

Figure 2.4: These figures show the evolution of CEED (a) and OCE (b) for a SAT problem with \( d = 75 \) and \( p = 325 \). For OCE, \( N = 40000 \) and \( \rho = .05 \). Each square is the value of \( v(1)^i_t \) when the horizontal axis is time \((t)\) and vertical axis is variable, \((1 \leq i \leq d)\)

\[
\text{Iteration} \quad \text{Satisfied Clauses}
\]

<table>
<thead>
<tr>
<th>CEED with no samples</th>
<th>CE, ( N = 10,000/iteration )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>1020</td>
</tr>
<tr>
<td>10</td>
<td>1140</td>
</tr>
<tr>
<td>20</td>
<td>1250</td>
</tr>
<tr>
<td>30</td>
<td>1390</td>
</tr>
<tr>
<td>40</td>
<td>1530</td>
</tr>
<tr>
<td>50</td>
<td>1670</td>
</tr>
<tr>
<td>60</td>
<td>1810</td>
</tr>
<tr>
<td>70</td>
<td>1950</td>
</tr>
<tr>
<td>80</td>
<td>2090</td>
</tr>
<tr>
<td>90</td>
<td>2230</td>
</tr>
<tr>
<td>100</td>
<td>2370</td>
</tr>
</tbody>
</table>

Figure 2.5: Convergence of CEED and OCE (with \( N = 10000 \) and \( \rho = .05 \)). CEED is performing even better than CE, while taking no samples, therefore being significantly faster.

Figure 2.5 shows the convergence of CEED for a SAT problem with \( p = 4250 \) and \( d = 1000 \). In each time-step we show \( L(x^t) \) when \( x^t = \text{mode}(\text{Bernoulli}_v(x)) \) is by far our best estimate of \( x^* \). We see that CEED, using simple analytical calculations, outperforms OCE, which is taking 10000 samples per iteration.

#### 2.4.2 Sudoku Problem

In this section we apply CEED to the Sudoku puzzle, which we already introduced in the beginning of this chapter.

Using our previous notation, let \( q = r^2 \). Since \( d = r^4 = q^2 \) is the dimension of the problem, \( \mathcal{X}^d = \{1, \ldots, q\}^q \) represents the problem domain. Here, for simplicity, we are including the fixed numbers in the problem domain as well. However the distribution over their corresponding variables are fixed (so if \( x_{1,2} \) is fixed at 3, we have \( v_{1,2}(3) = 1.0 \), and \( v_{1,2}(k) = 0 \) for \( k = 1, 2, 4, \ldots, 9 \). Let \( v_i = [v(1), \ldots, v(q)] \) be the parameter of \( f_{u_i}(x_i) = v_i(x_i) \), the pmf defined over \( \mathcal{X} \) (each square). Let \( u(x) : \mathcal{X}^q \to \mathcal{X} \) show the number of unique elements of vector \( x \). For example
Figure 2.6: This figure shows the evolution of CEED for the given Sudoku puzzle. Each column of each square is representing the pmf over the value of that variable in one time-step.

\[ u([1, 1, 2, 3, 4, 5, 3, 8, 9]) = 7 \], because it has 7 distinct elements. Our goal is to maximize \( u(x_{M(k)}) \) over each row, column and small square of the puzzle (all indexed by \( k \)).

To summarize, the basic idea (for example for a 9x9 puzzle) is to define a categorical distribution over the numbers in each square—i.e., a probability to each of the numbers 1 to 9. These distributions are to evolve so that their mode at some point represent the correct solution of each square. This is done by defining the number of unique element of each row, column and sub-square as the objective function (to be maximized).

First we sample from the distribution over variables in each sub-problem and find the maximum likelihood parameters for the best (elite) samples—i.e., the samples that produce the largest number of unique elements. Then different estimates for the distribution over each square (given by the corresponding row, column and sub-square) is combined to produce a new distribution over that variable. This distribution will be used for sampling in the next iteration. Combining distributions using their Fisher Information helps to incorporate the certainty of different sub-problems about the value of each square.

Given this formulation, OCE and CEED are easy to apply. When OCE is maximizing \( L(x) = \sum_{k=1}^{u} u(x_{M(k)}) \), CEED is maximizing each \( L_k(x_{M(k)}) = u(x_{M(k)}) \) for different values of \( k \). The Fisher information for a pmf over \( \mathcal{X} \) is \( I(v_i) = \text{Diagonal}(\frac{1}{v_i(1)}, ..., \frac{1}{v_i(q)}) \). We use this matrix to combine 3 pmf’s from 3 sub-problems (using Eq (2.7)) in which a variable participates—these sub-problems correspond to the row, column and sub-square of each variable’s square (see the coupling matrix Figure 2.2a).

For the 9x9 puzzle of Figure 2.6a, CEED finds a correct solution in 10 iterations using 1000 samples per iteration while OCE needs more than 5 times the number of samples used by CEED to find the solution. Figure 2.6b shows the evolution of \( \hat{v}_i^t(j) \) for the puzzle of Figure 2.6a with \( i \) indexing the small squares, \( t \) and \( j \) are varying along the horizontal and vertical axis respectively.
Chapter 3

Analysis of Nuclear Magnetic Resonance Spectra

3.1 Background, Problem Formulation and Application of CEED

Each pure chemical compound has its unique $^1$H NMR “signature”, which is a 1-dimensional signal composed of a set of clusters, where each cluster has a center and involves one or more peaks, each of which is characterized by 3 parameters, defining the peak’s height $a$, width $w$ and position $d$ relative to the cluster center, within a Lorentzian function (see Figure 3.1) [47, 44, 18]. As these clusters do not move much, biochemists have long used NMR to determine the identity of a pure compound, based on the observed peak locations. Moreover, as the height of the peaks in a compound are essentially proportional to the concentration of that compound, they can also quantify that concentration. See Appendix B for a more technical background on NMR spectroscopy. As the NMR spectrum of a mixture of chemicals $c_i$ (appearing in, say, human blood or urine) is essentially the linear combination of those signatures — $\sum_i \beta_i \text{signature}(c_i)$ where the $\beta_i$ coefficients depends on the concentrations of the $c_i$s — we can often recover those concentrations from a mixture [44]. In fact, once we determine the centers $\alpha = \{\alpha_j\}_j$ for the clusters, we can then find the concentrations...
\( \beta = \{ \beta_i \} \), by the non-negative linear least square methods [26]. The challenge, however, is finding the cluster centers.

To be more precise, the following equation shows the 1-dimensional spectrum (over the line \( Y \)) produced by \( m \) metabolites, where the \( i^{th} \) metabolite has concentration \( \beta_i \geq 0 \) and involves clusters in \( \Gamma(i) \) (as a set), where the \( j^{th} \) cluster is at position \( \alpha_j \) and involves peaks in \( \Upsilon(j) \), each peak associated with its Lorentzian parameters \( \langle w_k, a_k, d_k \rangle \) (which appear in a predefined library for this specific peak).

\[
\forall y \in Y \quad S_{\alpha,\beta}(y) = \sum_{i=1}^{m} \beta_i \sum_{j \in \Gamma(i)} \sum_{k \in \Upsilon(j)} \frac{a_k w_k}{w_k + 4(\alpha_j + d_k - y)^2} + \text{Noise}, \quad (3.1)
\]

Given an observed spectrum \( \tilde{S} \), we want to find the metabolite concentrations, \( \beta = (\beta_i) \) over the set of possible compounds — which corresponds to \( \{ \hat{\alpha}, \hat{\beta} \} = \arg \min_{\alpha,\beta} L(\{ \alpha, \beta \}) \) where

\[
L(\{ \alpha, \beta \}) = \sum_{y \in Y} [S_{\alpha,\beta}(y) - \tilde{S}(y)]^2 \quad (3.2)
\]

is the obvious \( L_2 \) loss; see Figure 3.2. For the effect of using other loss measures, see Appendix C.3.

While we only care about the concentrations \( \beta \), we must also determine the cluster centers \( \alpha \) to find them. If we knew the true concentrations \( \beta^* \), we could measure the loss using \( |\hat{\beta} - \beta^*|_2 \) — i.e., as the \( L_2 \) loss of the concentrations. In the experiments in Section 3.2, we consider \( L_2 \) loss (both over concentration and spectra) as well as the average controlled relative error:

\[
\kappa_1 = m^{-1} \sum_{i=1}^{m} \min \left( 1, \frac{|\hat{\beta}_i - \beta_i^*|}{\beta_i^*} \right) \quad (3.3)
\]

We also consider relative absolute error defined as:

\[
\kappa_2 = \frac{\sum_{i=1}^{m} |\hat{\beta}_i - \beta_i^*|}{\sum_{i=1}^{m} \beta_i^*} \quad (3.4)
\]

Figure 3.2: Two clusters of a metabolite, \( i_1, i_2 \in \Gamma(j) \)
We are not using ordinary relative error as a bad estimate of a low concentration metabolite, will severely affect the average, which does not properly reflect the accuracy of the method.

Optimization of Eq (3.2) is difficult, since it is large (i.e., involves hundreds of variables), and non-linear with respect to the majority of variables — the α’s. Furthermore, gradient based methods are slow, since the neighborhood of local optima in which the loss function is convex is usually small. That is, the Hessian matrix for the loss is typically not positive definite. See Appendix D for details.

As we see in Eq (3.1), the value of the Lorentzian function drops quadratically with the distance \((\alpha_j + d_k - y)\) from the center of peak (i.e., \(\alpha_j + d_k\)). We can therefore assume that each peak has a compact support, which consequently implies that each cluster (small set of close-by peaks) will potentially affect only a small region of the whole spectra. This region includes the bounded amount of shift in the center of the cluster \(\alpha_j\), and the effective width of cluster; call this region \(Y_j \subset \mathcal{Y}\).

We can rewrite the optimization Eq (3.2) as a weighted sum of squared error over all regions.

\[
\{\hat{\alpha}, \hat{\beta}\} = \arg \min_{\alpha, \beta}\sum_{j \in \Gamma(i) \atop 1 \leq i \leq m} L_j(\{\alpha, \beta\}_{M(j)})
\]  

(3.5)

where \(\{\alpha, \beta\}_{M(j)}\) is the set of variables appearing in sub-problem \(j\). For convenience we define the function \(\Lambda(l)\) that identifies the metabolite of cluster \(l\): \(\Lambda(l) = i\) iff \(l \in \Gamma(i)\). Using this notation \(\{\alpha, \beta\}_{M(j)} = \{\alpha_l, \beta_{\Lambda(l)}\}_{l \in M(j)}\) and \(M(j) = \{l \text{ s.t. } Y_j \cap Y_l \neq \emptyset\}\). Now define each sub-problem as:

\[
L_j(\{\alpha, \beta\}_{M(j)}) = \sum_{y_k \in Y_j} \eta_k \left( \tilde{S}(y_k) - \sum_{l \in M(j)} S_{\alpha_l, \beta_{\Lambda(l)}}(y_k) \right)^2,
\]  

(3.6)

where

\[
S_{\alpha_l, \beta_{\Lambda(l)}}(y) = \beta_{\Lambda(l)} \sum_{i \in Y(l)} \frac{a_i w_i}{w_i + 4(d_i + \alpha_l - y)^2}
\]  

(3.7)

is a single cluster that appears in sub-problem \(j\).

(As a small technical issue: For Eq (3.5) to equal Eq (3.2), we need to avoid double-counting. Here, we could set \(\eta_k = |\{Y_i \text{ s.t. } y_k \in Y_i\}|^{-1}\). For example, for each \(y_k \in Y_{j1} \cap Y_{j2} \cap Y_{j3}\), we should set \(\eta_k = \frac{1}{3}\). However we set all the weights \(\eta\) equal to 1 in our experiments. In the ideal case, as all loss functions \(L_i\) are equal to zero in global minima, any choice of weights gives the same zero loss, and therefore this \(\eta \equiv 1\) will not matter. For real spectra, however by removing the weights, we are giving more weight to more critical regions, which results in better analysis in practice.)

Figure 2.2d shows the coupling matrix for this optimization problem. Our CEED implementation uses Gaussian distributions, which means \(v_i = [\mu_i, \sigma_i]\). Therefore \(N_{\nu_{M(j)}} = N_{[\mu_{M(j)}, \sigma_{M(j)}]}\) is a distribution over the variables of \(j^{th}\) sub-problem, including both αs and βs.
In practice, the available metabolite library will not include all-and-only the metabolites in the chemical mixture — i.e., there are some metabolites in the mixture that not present in the library and vice versa. Here, CEED may produce an inferior fit by trying too hard to match the metabolites in its library. To reduce this problem, we use the minimum of the different estimates of each concentration value ($\beta_i$), rather than their linear combination Eq (2.11) (or Eq (2.12)). However we still use Eq (2.11) (or Eq (2.12)) to combine $\alpha$’s. Algorithm 3 shows the general steps of CEED when applied to the problem of analysis of NMR Spectra. Please refer to Appendix C.1 and C.2 for details on preprocessing, calculating the bounds and defining distributions.

Algorithm 3 CEED for NMR

Require: Spectra $S(x)$, Library, $t_{max}$ or $\delta$ for stopping criteria and $\rho$ as the percentile of top samples.
Calculate the bounds for all variables (see Appendix C.1)
Define $v^0 = [\mu^0, (\sigma^0)^2]$ using calculated bounds
$t \leftarrow 1$
repeat
   for $k = 1$ to $p = \sum_{i=1}^{m} \Gamma(i)$ do
      {$X_1, \ldots, X_{N(k,t)}$} $\sim N_{\nu,k}(x)$
      Calculate $L_k(X_j)_{1 \leq j \leq N(k,t)} = \sum_{y \in \mathbb{Y}_k} (S_{X_i}(k,y) - \hat{S}(y))^2$
      Calculate $\tau_k$ as the maximum loss of $\rho\%$ top samples (e.g., $\rho = 5$).
      Calculate $\hat{v}^{t+1}_{i,k}$ from Eq (1.5)
   end for
   Calculate $\hat{v}^{t+1}$ corresponding to $\alpha$’s using Eq (2.11) or Eq (2.12)
   Calculate $\mu^{t+1}$ for $i$ corresponding to $\beta$’s as $\min\{\mu^{t+1}_{i,k}\}_{k \in A(i)}$
   Calculate $(\sigma^{t+1})^2$ for $i$ corresponding to $\beta$’s using Eq (2.11) or Eq (2.12)
   Set $\hat{v}^{t+1} \leftarrow [\mu^{t+1}, (\sigma^{t+1})^2]$ for $i$ corresponding to a $\beta$ value
   Get the joint distribution– $\hat{v}^{t+1} \leftarrow [\hat{v}^{t+1}_1, \ldots, \hat{v}^{t+1}_d]$
   $t \leftarrow t + 1$
until $\max(\sigma^{t+1}) < \delta$ or $t \geq t_{max}$
$\hat{x}^* \leftarrow \text{mode}(N_{\nu}(x))$
return $\hat{x}^*$

3.2 Experimental Results

We measured the performance of our algorithm against Gradient Descent (GD), Simulated Annealing (SA) and Genetic Algorithm (GA) methods, on a typical simulated spectrum, with 90 metabolites and total of 595 variables (for the 90 concentrations and 505 cluster centers). We used the efficient and reliable implementation of these methods provided by the standard Matlab™ toolbox. GD is implemented by constrained nonlinear optimization, which uses active-set and line-search. SA is using fast annealing with exponential temperature update; we report results based on the best reannealing interval. GA used ranking for fitness scaling, stochastic uniform method for parent selection, cross-over fraction of 0.8; we report the result for the best combination of population size and generations. All major choices are made by reasonable effort of trial and error.

---

1 GD is implemented by constrained nonlinear optimization, which uses active-set and line-search. SA is using fast annealing with exponential temperature update; we report results based on the best reannealing interval. GA used ranking for fitness scaling, stochastic uniform method for parent selection, cross-over fraction of 0.8; we report the result for the best combination of population size and generations. All major choices are made by reasonable effort of trial and error.
Figure 3.3: Comparison of the convergence rate on a simulated spectra.

(e.g., sum of squared errors over all spectra) evaluations as the measure of computational resources used by the algorithm. Therefore we count the number of total evaluations of the Lorentzian function (peaks in Eq (3.1)) instead.

In the following sections we analyze the performance of our algorithm using two related criteria. The first task is detecting the presence of metabolites in a sample. The second task is correctly measuring the concentrations of metabolites. We analyze the performance of our algorithm in different scenarios of simulated spectra and real spectra against ground truth, expert manual analysis and a state-of-the-art mixture analysis method.

3.2.1 Simulated Spectra

Ideally, we want a method that works effectively, even if our library does not include signatures for all of the metabolites present, and also if it includes metabolites that are not present. Therefore to better analyze the behaviour of our method, we considered four scenarios. In each case we simulate an NMR spectra using a generating library, and then apply CEED using an analyzing library. We study the effect of overlap between these two libraries in the following cases:

1. The two libraries completely match each other.

2. The analyzing library is a sub-set of the generating library. This happens when the analyzing library is small but includes (only) essentials metabolites.

3. The generating library is a sub-set of analyzing library. This is only evaluating the effect of over-complete basis set which mainly occurs when large libraries are applied to simple mixtures (e.g., serum samples)

4. The two libraries have a sub-set of their signatures in common. This is basically what happens in practice with challenging mixtures (e.g., urine samples)
5. The last scenario is to examine the scalability of the method with large generating and analysing libraries.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Size of generating library</th>
<th>Size of analysing library</th>
<th>Overlap of two libraries</th>
</tr>
</thead>
<tbody>
<tr>
<td>Exact lib.</td>
<td>50</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>Essential lib.</td>
<td>100</td>
<td>50</td>
<td>50</td>
</tr>
<tr>
<td>Over-complete lib.</td>
<td>50</td>
<td>150</td>
<td>50</td>
</tr>
<tr>
<td>Incomplete lib.</td>
<td>70</td>
<td>70</td>
<td>50</td>
</tr>
<tr>
<td>Scaled experiment</td>
<td>120</td>
<td>120</td>
<td>100</td>
</tr>
</tbody>
</table>

Table 3.1 quantifies the library configuration for all four scenarios. We generated all the synthetic (analyzing and generating) libraries from a set of randomly selected metabolites of a 500 MHz, $^1$H NMR spectrum urine library. For each scenario then we generated 20 random spectra. The concentration and shift values are sampled from a truncated Gaussian distribution having a support equal to variable bounds. For distributions of concentrations, the mean value is 2 mMol (See Appendix C.2 for details.) The shift bounds correspond to a change of 0.5 in pH at pH = 7.0, with a minimum shift window of 0.007 ppm. Independent zero mean Gaussian noise with standard deviation of 0.01 was added to $\tilde{S}(y)$ for all $y \in \mathcal{Y}$.

Figures 3.4, 3.5 and 3.6 show sample fits generated by CEED for all the scenarios. Comparison of the estimated and correct concentration values for a sample case is presented in Figures 3.7, 3.7.

First we consider the presence detection task. For this we define a presence threshold. If a metabolite has a concentration larger than this threshold we report it present and otherwise absent. Since the concentrations have an average of 2 mMol (about 10 times our real spectra) we use a threshold of .2 mMol for presence detection. We report the following three measures for this binary classification:

- **Precision**: the ratio of correctly identified metabolites to the total number of metabolites detected.
- **Recall**: the ratio of correctly identified metabolites to the total number of metabolites present (ground truth).
- **f-measure**: combines recall and precision:

$$f\text{-measure} = 2 \frac{\text{Recall} \times \text{precision}}{\text{Recall} + \text{precision}}$$

---

$^2$Libraries used in all the experiments are obtained from ChenomX Inc.

$^3$ppm (part per million) is the unit for the horizontal axis of spectra, which shows the difference of frequency of resonance between to points on the spectra divided by the operating frequency of the magnet of spectrometer. See Appendix B
Table 3.3: Concentration error of CEED for all 5 scenarios. In all cases average value ± one standard deviation is reported.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>$\kappa_1$ (Eq 3.3)</th>
<th>$L_2$ over $\beta$</th>
<th>$\kappa_2$ (Eq 3.4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Exact lib.</td>
<td>.24 ± .02</td>
<td>4.06 ± 1.26</td>
<td>.15 ± .03</td>
</tr>
<tr>
<td>2. Essential lib.</td>
<td>.35 ± .04</td>
<td>10.42 ± 1.98</td>
<td>.35 ± .04</td>
</tr>
<tr>
<td>3. Over-complete lib.</td>
<td>.26 ± .02</td>
<td>5.29 ± .94</td>
<td>.29 ± .03</td>
</tr>
<tr>
<td>4. Incomplete lib.</td>
<td>.24 ± .03</td>
<td>4.84 ± .81</td>
<td>.21 ± .03</td>
</tr>
<tr>
<td>5. Scaled experiment</td>
<td>.39 ± .03</td>
<td>9.47 ± 1.28</td>
<td>.33 ± .03</td>
</tr>
</tbody>
</table>

Table 3.2: Detection error of CEED for all 5 scenarios. In all cases average value ± one standard deviation is reported.

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Precision</th>
<th>Recall</th>
<th>f-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Exact lib.</td>
<td>.98 ± .02</td>
<td>.96 ± .02</td>
<td>.97 ± .02</td>
</tr>
<tr>
<td>2. Essential lib.</td>
<td>.95 ± .03</td>
<td>.96 ± .02</td>
<td>.96 ± .02</td>
</tr>
<tr>
<td>3. Over-complete lib.</td>
<td>.83 ± .03</td>
<td>.95 ± .03</td>
<td>.89 ± .03</td>
</tr>
<tr>
<td>4. Incomplete lib.</td>
<td>.87 ± .03</td>
<td>.95 ± .02</td>
<td>.91 ± .02</td>
</tr>
<tr>
<td>5. Scaled experiment</td>
<td>.84 ± .02</td>
<td>.97 ± .01</td>
<td>.91 ± .01</td>
</tr>
</tbody>
</table>

Table 3.2 shows the error statistics for detection task. In scenarios 3 to 5, because of the sparseness of the model (increased chance of false-positive classification) the precision is still very good. The average higher value of recall tells us that our method does have a tendency to over-fit. This could be addressed, by increasing the value of threshold. However for consistency we select the threshold to be $\sim 10\%$ of the average concentration in all experiments.

Now we analyze the performance of CEED on the second measure– accuracy. Table 3.3 presents the error statistics for different error measures on concentration values.

Considering all these results we may safely conclude that our method reacts very well to over-complete libraries. As an example, Figure 3.7c shows that CEED correctly recognizes the 50 available metabolites in a set of 120. However metabolites that appear in the sample but are not in the library seem to be the main source of inaccuracy in the analysis. We also see that for the 4th scenario which is the most realistic one, our errors are close to the case when we have the exact library.

Of course choosing the minimum of all estimates of a metabolite will increase the accuracy of our CEED in the scenarios 1 and 3. However it will produce more over-fitting in the other cases, when the library is not complete. The method also seems reasonably scalable, considering the large size of the generating library in the final scenario.

Different factors such as crowdedness of corresponding area, number of clusters, etc. affect the accuracy of analysis for individual metabolites. Figure 3.8 shows the average $L_2$ error value for different metabolites in the analysis of 4th scenario. Since all the metabolite concentrations are sampled from the same distribution, the difference of $L_2$ loss is not a result of different concentration but the difficulty in the analysis of individual metabolites.

Of course the number of samples that we take for each variable in each iteration affects the performance of CEED. Figure 3.9a shows the convergence of CEED for different number of samples
Figure 3.4: Sample CEED fits for scenarios 1 and 2 of simulated spectra

(a) Complete Library

Fast convergence of cluster center and/or concentration (under-estimation)

(b) Essential Library

Plenty of unavailable metabolite signatures provides many opportunities for overfitting. This is reasonably avoided by restrictions imposed by different clusters of each metabolite.
Figure 3.5: Sample CEED fits for scenarios 3 and 4 of simulated spectra

(a) Over-complete Library

Under-estimating the concentration as a result of early convergence (this could be avoided by increasing the sample size).

(b) Incomplete Library

Desirable under-fitting of the spectrum and correct recognition of cluster centers, when only a subset of metabolites are available to us.

Unavailable to the library.
Figure 3.6: Sample CEED fits for scenarios 5 of simulated spectra (incomplete library)
Figure 3.7: Comparison of the correct and the CEED reported concentration for scenarios 1, 2 and 4.
Figure 3.7: Comparison of the correct and the CEED reported concentration for scenarios 3 and 5.

(3) over-complete library

(5) scaled experiment

CEED correctly finds close to zero concentration for most of 100 metabolites not in the spectrum.
Figure 3.8: $L_2$ error for different metabolites of analyzing library of $4^{th}$ scenario.

per iteration for the $4^{th}$ scenario and Figure 3.9b shows the the resulting fit. In all the previous experiments we used $N = 100000$ samples per iteration and $\rho = .05$.

### 3.2.2 Real Spectra

We also applied our CEED system to sets of 71 and 99 manually fitted spectra with operating magnet frequency of 600 MHz and 800 MHz respectively. We compared our results to results obtained by the SAGD automated tool, a hybrid of Simulated Annealing and Gradient Descent method provided by Chenomx—a company that is active in the analysis of NMR Spectra. Overall, our CEED achieved a better fit in both tasks. Figure 3.10 shows a CEED fit for each case.

Our observation is that CEEDover-estimates a metabolite’s concentration only if none of its clusters are constrained either directly by the low interce pt of the spectrum or by the presence of other more-reliably-estimated clusters in their region. Under-estimation usually only happens when the method converges very fast so that the cluster centers do not adjust properly and therefore the corresponding concentration is lower than correct value.

Figure 3.11 compares the concentration values for a typical samples with the reported manual fit, in each case.

Here we also consider the median of relative error over all metabolites and spectra as an evaluation measure:

$$\kappa_3 \triangleq \text{median} \left\{ \frac{|\hat{\beta}_{i,j} - \beta^*_{i,j}|}{\beta^*_{i,j}} \right\}_{i,j}$$

where $i$ and $j$ are indexing metabolites and spectra respectively.

In the presence-detection task, our method shows a better performance, however the difference is not significant. Table 3.4 compares the performance of two methods on both data-sets. We used the presence threshold of .02 mMol in all the cases.

Table 3.5 compares the performance of two algorithms in terms of different error measures for concentration values. We see that CEED is performing better than SAGD in both cases. The only measure on which SAGD performs (insignificantly) better is the average $L_2$ loss over concentrations; note this neglects the relative error of analysis for different concentration values.
Figure 3.9: The effect of sample-size on convergence rate (a) and the resulting fit (b).

Table 3.4: Comparison of CEED and SAGD for presence detection task on two data-sets of real spectra.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Precision</th>
<th>Recall</th>
<th>f-measure</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEED (600 MHz)</td>
<td>.71 ± .11</td>
<td>.97 ± .04</td>
<td>.82 ± .09</td>
</tr>
<tr>
<td>SAGD (600 MHz)</td>
<td>.67 ± .13</td>
<td>.97 ± .03</td>
<td>.79 ± .10</td>
</tr>
<tr>
<td>CEED (800 MHz)</td>
<td>.89 ± .06</td>
<td>.98 ± .03</td>
<td>.93 ± .04</td>
</tr>
<tr>
<td>SAGD (800 MHz)</td>
<td>.88 ± .07</td>
<td>.96 ± .04</td>
<td>.92 ± .05</td>
</tr>
</tbody>
</table>

Table 3.5: Comparison of CEED and SAGD concentration errors for real spectra.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>$\kappa_1$ (Eq. 3.3)</th>
<th>$\kappa_3$ (Eq. 3.8)</th>
<th>L2 over $\beta$</th>
<th>$\kappa_2$ (Eq. 3.4)</th>
</tr>
</thead>
<tbody>
<tr>
<td>CEED (600 MHz)</td>
<td>.60 ± .05</td>
<td>.64</td>
<td>2.59 ± 3.38</td>
<td>.43 ± .11</td>
</tr>
<tr>
<td>SAGD (600 MHz)</td>
<td>.77 ± .04</td>
<td>1.83</td>
<td>2.39 ± 1.52</td>
<td>.69 ± .17</td>
</tr>
<tr>
<td>CEED (800 MHz)</td>
<td>.47 ± .05</td>
<td>.36</td>
<td>2.37 ± 1.38</td>
<td>.27 ± .09</td>
</tr>
<tr>
<td>SAGD (800 MHz)</td>
<td>.61 ± .06</td>
<td>.66</td>
<td>2.07 ± 2.16</td>
<td>.32 ± .13</td>
</tr>
</tbody>
</table>
The large number of unavailable metabolites in this crowded region makes the analysis very difficult.

Desirable behaviour of CEED in avoiding over-fitting in empty regions.
Figure 3.11: Comparison of CEED’s reported concentration values (β) with expert’s estimate, for a typical 600 MHz and 800 MHz spectra
Overall in the experiments we found that our method very reliable in identifying metabolites in a sample, and that here it was able to outperform SAGD. By the analysis of simulated spectra we saw that our method is able to reliably identify and measure the concentration of metabolites with a good accuracy, even if our reference library included too many, compounds or when it was missing some metabolites.  

---

Note that we used a preliminary version of the ChenomX software; It seems they have continued to improve their algorithm, and now have better results.
Chapter 4

Conclusion

This thesis has introduced a new stochastic optimization method called CEED that attempts to find good solutions to partially decomposable problems, and demonstrates that it works effectively, in the context of interpreting $^1$H NMR spectra of complex mixtures.

The basic idea was to consider sub-problems separately and combine their estimates of optimal results. This is done by defining joint distributions over the problem domain, and using the corresponding marginals for each subproblem. The thesis then uses a novel method of combining maximum likelihood estimates and it also relates this technique to some basic methods like Mixture Using Variance.

In experimental results, we found that CEED was faster than many existing optimization methods. We explored the robustness of $\text{CEED for NMR}$ with regard to handling incomplete and over-complete libraries for NMR analysis, using simulated spectra, and we found our method capable of producing sparse models— which is desirable when there is a possibility of over-fitting in a model. In analysis of real spectra we reported better results than an state-of-the-art automated tool. In the task of identifying the metabolites in a sample we acheived a very good and reliable performance in all scenarios of simulated spectra as well as for the real data.

We also applied CEED to SAT and Sudoku as two examples in the combinatorial optimization domain, mainly to demonstrate generality of our algorithm. We found that the distributions produced by CEED resemble those of the CE method and hence have the potential of being used as importance sampling distributions for counting problems. In both of these cases CEED proved superior to OCE.

In addition to our theoretical claims, we have also produced a very practical system, one that can effectively analyse complex $^1$H NMR spectra. We believe this will prove valuable in the study of metabolic biomarkers, which may prove very helpful in diagnosing and treating diseases [47].
Bibliography


[38] R Rubinstein. How many needles are in a haystack, or how to solve#p-complete counting problems fast. Methodology and Computing in Applied Probability, Jan 2006.


Appendices
Appendix A

Cross-Entropy Method from Importance Sampling Perspective

In this section we introduce CE method as an iterative importance sampling method\(^1\) (see for example [41, 36]). Given an error function \(L(x)\), an inefficient way to approximate \(x^* = \arg \max_x L(x)\), the minimizer of \(L(x)\), is to sample a uniform distribution over \(\mathcal{X}\), \(f_u(x) = \frac{1}{|\mathcal{X}|}\), and consider \(\hat{x}^*\) to be the empirical mode of samples \((X_i)\) for which \(\mathbb{I}_{\{L(X_i) < \tau\}}\) for small \(\tau\).

However in problems of a combinatorial nature, the probability of event \(\mathbb{I}_{\{L(X) < \tau\}}\) when \(X \sim f_u(x)\), decreases exponentially in the number of variables. Therefore we need a sampling distribution \(f^*(x) \in \mathcal{F}\) that increases the chance of event \(\mathbb{I}_{\{L(X) < \tau\}}\), so that we can use \(\hat{x}^* = \text{mode}(f^*)\) as an approximation to \(x^*\).

For a limited number of samples, we ideally want to sample from:

\[
    f^*(x) = \frac{\mathbb{I}_{\{L(x) < \tau\}}}{\int_{\mathcal{X}} \mathbb{I}_{\{L(x) < \tau\}} dx} \tag{A.1}
\]

Once we have this distribution we can report \(\hat{x}^* = \text{mode}(f^*)\). However estimation of \(f^*\) is a difficult problem. Notice the denominator, \(c = \int_{\mathcal{X}} \mathbb{I}_{\{L(x) < \tau\}} dx\), of right hand side of Eq (A.1), requires to calculate the probability of a rare event, \(P(\mathbb{I}_{\{L(X) < \tau\}} | X \sim f_u) = \frac{\tau}{|\mathcal{X}|}\). Therefore in what follows we think of our original optimization problem as a problem of estimating this rare event. The relation becomes more clear once we see a similar solution for both problems later in this chapter. Now we will look into the ways in which we can best estimate Eq (A.1).

To show the difficulty of the desired approximation, consider the Crude Monte Carlo (CMC) estimator of \(c\):

\[
    \hat{c} = N^{-1} \sum_{i=1}^{N} \mathbb{I}_{\{L(X_i) < \tau\}} \quad , \quad X_i \sim f_u(x) \tag{A.2}
\]

When \(\mathbb{I}_{\{L(X) < \tau\}}\) is rare, such estimation has a high relative error. Let

\[
    \hat{S}^2 = \frac{1}{N-1} \sum_{i=1}^{n} (\mathbb{I}_{\{L(X_i) < \tau\}} - \hat{c})^2 \tag{A.3}
\]

\(^1\)As one may expect, some definitions and keywords from Section 1.2 are re-defined here from another viewpoint.
be the estimation of \( S^2 = \text{Var}[\mathbb{I}_{\{L(X) < \tau\}}] \).

Given \( \hat{S}^2 \), by the central limit theorem, the variance of \( \hat{c} \) is \( \frac{\hat{S}^2}{N} \), which is approximated by \( \frac{\hat{S}^2}{N} \) for large \( N \). For large \( N \), Eq (A.3) simplifies to:

\[
\hat{S}^2 \approx c(1 - \hat{c})^2 + (1 - c)\hat{c}^2 \quad (A.4)
\]

This equation is basically saying that with probability ratio \( c \), the term in summation of Eq (A.3) assumes a value of \( (1 - \hat{c})^2 \) and with probability of \( 1 - c \) it is equal to \( \hat{c}^2 \). For rare events the second term in equation Eq (A.4) becomes negligible, and so:

\[
\hat{S}^2 \approx c(1 - \hat{c})^2 \quad (A.5)
\]

Now consider the relative error as the measure of accuracy of our estimation (i.e., \( \hat{c} \)):

\[
\kappa = \sqrt{\frac{\text{Var}(\hat{c})}{E\{\hat{c}\}}} \approx \frac{\sqrt{S^2}}{c\sqrt{N}} \quad (A.6)
\]

For a rare event, using the substitution from Eq (A.5)

\[
\kappa \approx \sqrt{\frac{1 - c}{cN}} \approx \sqrt{\frac{1}{cN}} \quad (A.7)
\]

which means for \( c = P(\mathbb{I}_{\{L(X) < \tau\}}) = 10^{-6} \), for an accuracy of \( \kappa = 0.01 \), we need \( N \approx 10^{10} \) samples. Therefore we see that Monte Carlo estimation is not efficient in the case of our interest.

Alternatively, we can use Importance Sampling (IS), to increase the probability of sampling the rare event and therefore provide a more reliable estimate. For this, define the Likelihood Ratio as:

\[
W_{f,u,g}(x) = \frac{f(x)}{g(x)} \quad (A.8)
\]

where \( g(\cdot) \) is used instead of \( f(\cdot) \) for sampling. Then rewriting Eq (A.2) we have:

\[
\hat{c} = \sum_{i=1}^{N} \mathbb{I}_{\{L(X_i) < \tau\}} W_{f,u,g}(X_i) \quad , \quad X_i \sim g(x) \quad (A.9)
\]

The only constraint on IS distribution \( g(x) \) is that it should have non-zero probability wherever \( f_u(x)\mathbb{I}_{\{L(X) < \tau\}} > 0 \). More formally if

\[
\text{supp}(f(x)) = \{ x \in \mathcal{X} ; \ f(x) \neq 0 \} \quad (A.10)
\]

then \( g(x) \) should satisfy:

\[
\text{supp}(g(x)) \supseteq \text{supp}(\mathbb{I}_{\{L(X) < \tau\}}f_u(x)) \quad (A.11)
\]

When \( g(x) \) is the same as \( f^*(x) \) in Eq (A.1), one could accurately estimate \( c \), using only one sample! Therefore the variance of the estimation of \( c \) is minimized for \( f^* \) as defined by Eq (A.1):

\[
f^*(x) = \arg\min_{g(x)} \left\{ \text{Var}_g[\mathbb{I}_{\{L(X) < \tau\}}W_{f_u,g}(X)] \right\} \quad (A.12)
\]

\[
= \arg\min_g \left\{ \mathbb{E}_g[\mathbb{I}_{\{L(X) < \tau\}}^2f_u(x)^2/g(x)^2] \right\} \quad (A.13)
\]

\[
= \arg\min_g \left\{ \mathbb{E}_{f_u(x)}[\mathbb{I}_{\{L(X) < \tau\}}f_u(x)/g(x)] \right\} \quad (A.14)
\]
when the change in Eq (A.14) from Eq (A.13) is based on the change of sampling distribution (i.e., multiplication by \( W_{g,f_u} \)), and using the fact that \( \mathbb{I}_{L(X)<\tau}^2 = \mathbb{I}_{L(X)<\tau} \). We changed the sampling distribution because in contrast to \( g(\cdot) \), we can sample from \( f_u(\cdot) \).

The variance of IS, which is minimized by \( f^*(x) \) in Eq (A.14), is basically a special case of general family of distances known as Ali and Silvey distances [1] (a.k.a Csiszár f-divergence [12]) between two distributions \( (f, g) \):

\[
\mathcal{D}(f, g) \equiv \Phi \left( \mathbb{E}_f[\Psi \left( \frac{f(X)}{g(X)} \right)] \right)
\]

(A.15)

where \( \Phi \) is convex on \([0, +\infty]\) and \( \Psi \) is an increasing function.

Here we are trying to estimate \( f^* \) by finding the distribution, \( g \) with the least divergence to the samples taken from \( f^* \) (i.e., \( X_i \) for which \( L(X_i) < \tau \)). We may also consider the minimization Eq (A.14) for another member of f-divergence family, KL-divergence or Cross Entropy:

\[
f^*(x) = \arg \min_g \left\{ \mathbb{E}_{f_u}[|f_u(x)|_{L(X)<\tau} \ln \left( \frac{f_u(x)|_{L(X)<\tau}}{g(x)} \right)] \right\}
\]

(A.16)

When \( f^*(x) \in \mathcal{F} \) belongs to specific parametric family, we denote it by \( f_{v^*}(x) \), where \( v^* \) is the optimal parameter. Then we have:

\[
v^*(x) = \arg \max_v \left\{ \mathbb{E}_{f_u}[|f_u(x)|_{L(X)<\tau} \ln(f_v(x))] \right\}
\]

(A.17)

In practice we consider the empirical counterpart of Eq (A.17):

\[
\hat{v^*}(x) = \arg \max_v \left\{ \sum_{i=1}^N f_u(X_i)|_{L(X_i)<\tau} \ln(f_v(X_i)) \right\}, \quad X_i \sim f_u(X)
\]

(A.18)

By changing the sampling distribution to an arbitrary one and once again using the IS ratio, we get

\[
\hat{v^*}(x) = \arg \max_v \left\{ \sum_{i=1}^N f_u(X_i)|_{L(X_i)<\tau} W_{f_u(x), f_v(x)}(X_i) \ln(f_v(X_i)) \right\}, \quad X_i \sim f_w(x)
\]

(A.19)

Once again the sampling distribution \( f_w \) that maximizing the probability of the rare event and satisfies the condition Eq (A.11) is \( f_{v^*} \). The equation Eq (A.19) is a way to gain a better estimate of \( f_{v^*} \) given a less accurate estimate, \( f_w \). We can use this progressive update to estimate \( f_{v^*}(x) \) in an iterative approach.

### A.1 Iterative Method

In the previous section we saw to accurately estimating the desired parameter \( v^* \), we had to sample the distribution \( f_{v^*} \), which is unavailable. One way out of this problem is to iteratively approximate \( v^* \) by \( \hat{v}_i \) starting from \( v_0 = u \), and in each iteration use the previous estimation \( \hat{v}_{i-1} \) in Eq (A.19).
Furthermore to prevent high variance in the initial estimations we start from a relatively larger value of $\tau^t$ and decrease $\tau^t$ in each time-step such that $\tau^{t+1} < \tau^t$. In each iteration $\tau^t$ should be chosen such that when $X_i \sim f_{v^t-1}(x)$, then $I_{\{L(X_i) < \tau^t\}}$ is not rare. The common method is to set it to the maximum of top $\rho$ percentile of the samples $- \mathbf{X}_{\text{elite}} = \{X_i \; ; \; L(X_i) < \tau_t\}$ is the set of top samples, called the elite samples. Given $\tau_t$, in each iteration we update our estimate of $v^*$ using

$$\hat{v}^t = \arg \max_v \left\{ \frac{1}{N} \sum_{i=1}^{N} I_{\{L(X_i) < \tau^t\}} W_{u,\hat{v}^t-1}(X_i) \ln f_v(X_i) \right\} \quad (A.20)$$

$$W_{u,v}(x) = \frac{f_u(x)}{f_v(x)} \; , \; X_i \sim f_{\hat{v}^t-1}(x) \quad (A.21)$$

Furthermore to avoid fast convergence to a bad local minima we could use a smoothing parameter, $\zeta$ by which we update distribution in each time-step. In the original algorithm of CE for combinatorial and continuous optimization, the Likelihood Ratio, $W(x) \equiv 1$ (see [41] for more discussion), which gives Eq (1.5) in Section 1.2. Algorithm (4) summarizes general steps of CE method.

**Algorithm 4** OCE optimization method

Initialize $t \leftarrow 0$  
\[ \hat{v}^t \leftarrow u \]  
repeat
  Generate samples: $\{X_1, ..., X_N\} \sim f_{\hat{v}^t}(x)$
  Calculate the error $L(X_j)$ for each sample
  Calculate $\tau^t$ as the level-set separating top $\rho$ percentile of samples.
  Calculate $\hat{v}_{t+1}$ using Eq (A.20) with $W(X_i) = 1$
  \[ \hat{v}_{t+1} \leftarrow \eta \hat{v}_{t+1} + (1 - \eta) \hat{v} \]
  \[ t \leftarrow t + 1 \]
until convergence  
\[ \hat{x}^* \leftarrow \text{mode}(f_{\hat{v}^t}(X)) \]
return \[ \hat{x}^* \]
Appendix B

A More Technical Background on NMR Spectroscopy

In this appendix we provide some technical details to better explain the concept behind NMR spectroscopy. To acquire the NMR signal we need to expose the nuclei of interest to some magnetic pulse, which has a specific strength and frequency. The spectrometers are named based on the frequency at which the proton resonates (e.g., 500 MHz or 600 MHz). Once exposed to the magnetic field during specific pulse width, the nuclei of interest will resonate differently depending on the molecular structure of the corresponding compound. Then, during the acquisition time, the resonance of each resonating nucleus is recorded as a Free Induction Decay or FID. This is basically the raw NMR spectra. This spectrum has the form of an exponentially decaying complex valued sinusoid (see Figure B.1a) resonating with specific frequency and decay constant for each harmonic component (corresponding to one peak in translated signal):

\[ S(t) = \nu e^{i(2\pi f + \phi) - t\psi} \]  

(B.1)

where:

1. \( \nu \) is the amplitude and depends on sampling time and the number of nuclei of that kind (e.g., \( ^1\text{H} \) or \( ^{13}\text{C} \)) in the same molecule.
2. \( \psi \) is the decay constant which depends on sampling time and properties of specific nucleus.
3. \( \alpha \) is the frequency, which is equal to the difference between the frequency of the NMR spectrometer and the resonance frequency of a specific nucleus (e.g., \( ^1\text{H} \) in some specific compound)
4. \( \phi \) is the phase and is based on the spectrometer settings.

The signal is usually transferred to Frequency domain by Fourier Transform. After the Fourier Transform we ideally acquire:

\[ S(y) = \int_{-\infty}^{+\infty} S(t) e^{-i2\pi yt} dt \]  

(B.2)
However the acquisition time from $-\infty$ to $+\infty$ is not possible. Considering the acquisition time-
frame of $t_1$ close to zero (using the fact that the value of the signal is zero for $t < 0$) and $t_2$:

$$S(y) = \int_{t_1}^{t_2} S(t)e^{-i2\pi y t} dt = \frac{e^{i\phi} \left(e^{2i\pi t_1(-y+\alpha)-t_1\psi} - e^{2i\pi t_2(-y+\alpha)-t_2\psi}\right) \nu}{2\pi i(y-\alpha) + \psi}$$  \hspace{1cm} (B.3)

Figure B.1 shows the effect of different parameters on the FID and Fourier Transfer of the
signal. Notice that in both domain the spectra has both imaginary and real components.

![Figure B.1](image)

Figure B.1: Effect of different parameters in the time and frequency domain representation of a
harmonic component (See Eq (B.3) and Eq (B.1)).

The preprocessing steps involve phase correction, for which the effect of non-zero $\phi$ and non-
ideal $t_1$ and $t_2$ is fixed. Setting $\phi = 0$, $t_1 = 0$ and $t_2 = +\infty$ in Eq (B.3), we get:

$$S(y) = \frac{\nu}{2\pi i(y-\alpha) + \psi} = \frac{\nu\psi}{4\pi^2(y-\alpha)^2 + \nu^2} + i\frac{\nu^2(2\pi(y-\alpha))}{\psi^2(1 + (2\pi(y-\alpha))^2)}$$  \hspace{1cm} (B.4)

Since knowing either the real or imaginary part is sufficient to decide the other one, for technical
reasons (e.g., longer tail of imaginary part and therefore higher interference of different peaks) only
the real part is used in analysis. By a change of variable, setting $w = \frac{\nu}{\psi}$ and $a = \frac{\nu\psi}{\psi}$ we have:

$$S(y) = \frac{aw^2}{w^2 + 4(y-\alpha)^2}$$  \hspace{1cm} (B.5)
which is the notation that we have been using for a Lorentzian peak. This substitution produces
more meaningful variables; here \( w \) is basically the width of Lorentzian at its half height, and \( a \) is the
maximum value of the peak [16].

Once the signal is brought to frequency domain (by Fast Fourier Transform), we are dealing with
a set of points (e.g., 65k for 500MHz or 132k for a 800MHz spectrum) in a frequency range known
as the *Spectral Width* or Sweep Width. This quantity may be measured in *Hertz* (e.g., 5k or 6k width)
or a unit that is independent of the operating frequency of the magnet, which is known as *ppm*. A
unit of ppm is representing the frequency difference, with \(^1\)H nuclei of an specific compound called
the Chemical Shift Indicator (e.g., DSS), divided by the operating frequency of the magnet. In all of
our experiments, we have used ppm as the unit for the horizontal axis of spectra.

Another important preprocessing step, once we have the signal, involves removing the effect of
water in the spectra. Since water is present in all our samples in very high magnitude, it affects a
wide range of frequencies in the spectra. In all our experiments with real spectra we dealt with the
water effect by simply neglecting the water area (4.2 - 6 ppm) within which the water effect is more
significant.
Appendix C

Details of CEED for NMR

In this section we explain some details of the way CEED and other methods have been applied to analyze NMR spectra. This includes some details on the algorithm and parameters in preprocessing, defining distributions and the effect of various loss measures.

C.1 Preprocessing

In addition to phase correction and dealing with water suppression (see Appendix B), there are other preprocessing steps to bound concentration and shift variables and also to normalize the spectra and prepare the library for a given spectra. It involves the following sequence of steps:

1. Normalize the library and spectra. Normalization is with respect to a Chemical Shift Indicator\(^1\), a specific chemical (DSS\(^2\) in our samples) with a known concentration (e.g., 0.5 mM) that is separately added to every sample. The width (\(w_k\) in Eq (3.1)) for different peaks are calculated as a ratio to DSS width, and therefore the library needs to be updated after we fit DSS and calculate its width. Spectra should also be normalized so that DSS has height of one. This is because the amplitude of each peak of each metabolite in the library (\(a_k\) value in Eq (3.1)) is given for an specific concentration of that metabolite (\(\beta_{\text{lib},i}\)), assuming that DSS of a given concentration (\(\beta_{\text{DSS},i}\)) has the height of one. Therefore given the concentration of DSS (e.g., 0.5 mM) in the current sample (\(\beta_{\text{DSS,spectra}}\)), the ratio of the area of each cluster to its library version (which we call \(\beta_{\text{raw},i}\)), will specify its real concentration:

\[
\beta_i = \frac{\beta_{\text{lib},i}}{\beta_{\text{DSS,spectra}}} \beta_{\text{DSS},i}
\]

(C.1)

2. Calculate the bounds for the shift variables (\(\alpha_j\)). In practice, bounds for the cluster centers depends on the so called local pH. Since the pH effect is a quantified factor, we could rely on a bounded deviation from the global pH, and use the shift bounds corresponding to such

\(^1\)The horizontal axis of spectra (in ppm unit) represents the difference of frequency of resonance with Chemical Shift Indicator in any point, divided by operating frequency of the magnet. Therefore DSS is always located in the origin of the spectra.

\(^2\)4,4-dimethyl-4-silapentane-1-sulfonic acid
change of pH in our analysis. Cluster centers are quantified to be *logistic* functions of the pH (see Figure C.1):

\[
\alpha_j(pH) = \sum_i c_i e^{d_i x_j + r_i pH} \sum_i c_i' e^{d_i' x_j + r_i' pH}
\]

when \(\{c_i, d_i, r_i, c_i', d_i', r_i'\}\) are in the library for each cluster, \(j\).

Given the measured pH of a sample (e.g., \(pH = 7.00\)), we calculate the bounds for the cluster centers as follows.

\[
\alpha_{j,\text{max}} = \max \left\{ \max_{pH-\epsilon \leq x \leq pH+\epsilon} \{\alpha_j(x)\}, \alpha_j(pH) + \xi \right\}
\]

where \(\epsilon\) is the maximum pH error (e.g., 0.5) and \(\xi\) is the minimum shift (e.g., 0.007 ppm). Calculation of \(\alpha_{j,\text{min}}\) is similar.

3. Calculate a maximum area of effect (\(\hat{Y}_j\)), for each cluster \((j)\), assuming a maximum concentration (e.g., \(\hat{\beta}_{\text{max}} = 50\) mM). That is for each cluster \(j\), let:

\[
S_j(y, \alpha) = \sum_{k \in \Upsilon(j)} \frac{a_k w_k}{w_k + 4(\alpha + d_k - y)^2}
\]

be the cluster’s signal at specific center. Then set its area of effect as all the regions of the spectra for which the cluster (assuming the maximum concentration) has a value larger than a threshold:

\[
\hat{Y}_j = \left\{ y \in \mathcal{Y} \mid \exists \alpha \text{ s.t. } \alpha_{j,\text{min}} \leq \alpha \leq \alpha_{j,\text{max}} \wedge \hat{\beta}_{\text{max}} S_j(y, \alpha) \geq \iota \right\}
\]

when \(\iota\) is representing the threshold which we choose to be the same as the noise level (e.g., 0.002).
Figure C.2: The boarder of effect ($Y_j$ in blue) and shift bounds (in red) for different clusters. Clusters of the same metabolite appear in the same row.

4. Calculate the upper-bound for concentration by taking the minimum of the maximum concentration possible for corresponding clusters. The maximum concentration for each cluster is calculated by convolving the cluster signal in its area of effect. Set the upper-bound for $\beta_i$ as:

$$\beta_{i,\text{max}} = \min_{j \in \Gamma(i)} \left\{ \max_{\alpha_{\text{min}} \leq \alpha \leq \alpha_{\text{max}}} \left\{ \min_{y \in Y_j} \left\{ \frac{\tilde{S}(y) + \iota}{S_j(y, \alpha)} \right\} \right\} \right\}$$  \hspace{1cm} (C.6)

$\beta_{i,\text{min}}$ is always equal to zero.

5. Calculate a new area of effect, by substituting $\hat{\beta}_{\text{max}}$ in Eq (C.5) with $\beta_{i,\text{max}}$ for $j \in \Gamma(i)$. Figure C.2 shows the boarder of effect, $Y_j$ (in blue), and shift bounds, $[\alpha_{j,\text{min}}, \alpha_{j,\text{max}}]$ (in red) for all the clusters of a library. Clusters of the same metabolites are in the same row.

After these steps, we have a normalized signal, a matching library, a set of bounds for all variables and boarders of effect for all clusters.
C.2 Defining Distributions

To use CEED, we need to define a distribution over each concentration and shift value. For cluster centers ($j \in \Gamma(i)$), we define $v_j = [\mu_j, \sigma^2_j]$:

\[
\mu_j = \frac{\alpha_{j,\min} + \alpha_{j,\max}}{2} \quad \text{(C.7)}
\]

\[
\sigma_j = v(\alpha_{j,\max} - \mu_j) \quad \text{(C.8)}
\]

Here $v$ (e.g., 1.5) decides the initial coverage of the distribution. For $\beta_i$ the process is similar:

\[
\mu_i = \frac{\beta_{i,\max}}{2} \quad \text{(C.9)}
\]

\[
\sigma_i = v(\beta_{i,\max} - \mu_i) \quad \text{(C.10)}
\]

When sampling however we neglect the out-of-bound samples, as if already deciding that they will not be among the elite samples (see [23] for similar approach in OCE).

C.3 The Loss Measure

When the optimization is to estimate the model parameters given the real world data, different loss measures bear implicit assumptions about the noise or the shortcomings of the model. For example when we use $L_2(S, \tilde{S}) = \sqrt{\sum_{y \in Y} (\tilde{S}(y) - S(y))^2}$ to minimize the distance of a linear model ($S(y)$) with noisy data, we are assuming an loss of the prediction with a Gaussian distribution around the prediction of the model.

In the case of our problem, it is not clear what loss measure may represent the incompleteness of our model (e.g., missing metabolites, water effect and baseline error). However we tried several measures that are more robust to outlier noise; both $L_1(S - \tilde{S}) = \sum_{y \in Y} |(\tilde{S}(y) - S(y))|$ and $L_{\log}(S - \tilde{S}) = \sum_{y \in Y} \log(1 + |\tilde{S}(y) - S(y)|)$.

The reason behind their robustness is that, of fits of the same $L_2$ loss, they favor fits with more values close to zero. That is, they tend to either completely fit a point or count it as an outlier (more than $L_2$ norm).

We also used the sparsity penalty $L_{\text{sparse}}(j) = \sum_{l \in M(j)} |\beta_{\Lambda(l)}|$ for the metabolites in each area. Although our problem is in general non-linear, if we fix then cluster centers, we are matching a (sparse) linear model. To apply a sparsity penalty the cluster areas also need to be normalized.\(^3\) In practice we did not see any improvement when using sparsity penalty.

It was also important to make sure that the reconstruction always has a lower y-intercept than the given spectra. For this purpose we used an extra penalty term $L_1^+(S - \tilde{S}) = \sum_{y \in Y} [S(y) - \tilde{S}(y)]_+$ for L1 loss , $L_2^+(S - \tilde{S}) = \sqrt{\sum_{y \in Y} [S(y) - \tilde{S}(y)]_+^2}$ for L2 loss and $L_{\log}^+(S - \tilde{S}) = \sum_{y \in Y} \log(1 + [S(y) - \tilde{S}(y)]_+)$ for log error.

\(^3\)Since for each value of the shift, the area of a cluster that is participating in a sub-problem could change, the normalization of each cluster area (length of the basis vector) should be in a sample-by-sample basis.
We ran the experiment of the 4th scenario in Section 3.2.1 for various loss measures on a set of 10 simulated spectra. Table C.1 reports their performance in terms of different loss measures on the recovered concentration values. All the results are for 5 iterations of the algorithm, with $\rho = 0.05$ and 500,000 (i.e., approximately $3 \times 10^{10}$ Lorentzian evaluations) samples per iteration.

Here $\lambda = 1$ and $\gamma = 0.01$ is the best of $\{0.1, 0.001, 0.0001\}$. The average $L_2$ norm of the correct concentration is 19.1.

As we see, unfortunately using different loss measures and penalties terms does not result in a considerable improvement. In the experiments of Section 3.2 we have used the $L_2$ loss with overfitting penalty of one ($\lambda = 1$).

<table>
<thead>
<tr>
<th>Loss Measure</th>
<th>avg. $\kappa_1$ (Eq. 3.3)</th>
<th>avg. $\kappa_2$ (Eq. 3.4)</th>
<th>avg. $L_2(\beta - \tilde{\beta})$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$L_1(S - \tilde{S})$</td>
<td>0.21</td>
<td>0.21</td>
<td>5.40</td>
</tr>
<tr>
<td>$L_{log}(S - \tilde{S})$</td>
<td>0.22</td>
<td>0.21</td>
<td>5.36</td>
</tr>
<tr>
<td>$L_{log}(S - \tilde{S}) + L_{log}^+(S - \tilde{S})$</td>
<td>0.21</td>
<td>0.21</td>
<td>5.36</td>
</tr>
<tr>
<td>$L_2(S - \tilde{S}) + \gamma \sum_{1 \leq j \leq p} L_{sparse}(j)$</td>
<td>0.22</td>
<td>0.21</td>
<td>5.42</td>
</tr>
<tr>
<td>$L_2(S - \tilde{S}) + \lambda L_2^+(S - \tilde{S})$</td>
<td>0.21</td>
<td>0.22</td>
<td>5.45</td>
</tr>
<tr>
<td>$L_2(S - \tilde{S})$</td>
<td>0.22</td>
<td>0.22</td>
<td>5.52</td>
</tr>
</tbody>
</table>

Table C.1: Comparison of performance when using different loss measures.
Appendix D

Quasi-Convexity of $L_2$ Loss

In this appendix we show that $L_2$ loss function for analysis of NMR spectra in the neighborhood of local minima is not convex, but quasi-convex. This fact implies that application of Newton’s method as a fast convex-optimization technique is not feasible and therefore multi-start techniques of global optimization would be slow.

Given a spectrum one can numerically calculate the Hessian matrix for the loss function of Eq (3.2). Although such a calculation is expensive, the overall advantage of using Newton’s method would have been considerable if the Hessian matrix was positive semi-definite. However this is usually not the case.

Consider the simplified case of a single peak at the origin that we would like to fit, with the other peak of the same width (see Figure D.1a). The sum of squared error as a function of peak center and concentration is:

$$Err(\beta, \alpha) = \int_{-\infty}^{+\infty} \left( -\frac{a\beta}{w^2 + 4(\alpha - x)^2} + \frac{a}{w^2 + 4x^2} \right)^2 dx |_{a=1, w=1}$$  \hspace{1cm} (D.1)

Figure D.1b shows this loss function which is linear in the concentration for $\beta \equiv 1$ over different values of $\alpha$. The second derivative of this loss function with respect to the peak center is:

$$\frac{\partial Err(\beta, \alpha)^2}{\partial^2 \alpha} = \frac{a\pi w^3 (-3\alpha^2 + w^2)}{(\alpha^2 + w^2)^3}$$  \hspace{1cm} (D.2)

which is positive only when $-\frac{a}{\sqrt{3}} \leq \alpha \leq \frac{a}{\sqrt{3}}$. Figure D.1c shows the second derivative for different values of $\alpha$.

For a 500MHz urine library of 90 metabolites, 505 clusters and 4097 peaks, the average ratio of $\frac{\alpha}{w}$ is 3.3714, which means for a random cluster center (in the shift bound), almost 83% of the time the loss with respect to the shift of a single peak has a negative second derivative. This provides a good chance of the same behavior from the Hessian matrix that has, as each of its elements, the summation of such values, for peaks in the same clusters. Empirical results confirms this speculation. As an example Figure D shows the $L_2$ loss as a function of cluster center for the final stage of CEED.
Figure D.1: (a) shows a curve fitting problem for a single Lorentzian peak. (b) show the loss function of Eq (D.1) for $\beta = 1$. (c) is the second derivative of loss of Eq (D.1) with respect to $\alpha$. 
Figure D.2: The $L_2$ loss as a function of cluster center for 287 clusters after convergence of CEED.

when applied to analysis of urine spectra. This figure confirms Quasi-convexity of the loss in the neighborhood of local minima with respect to most of variables.