Graph Theoretic Algorithms Adaptable to Quantum Computing

by

Siddhartha Srivastava

A dissertation submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy
(Aerospace Engineering)
in the University of Michigan
2021

Siddhartha Srivastava

sidsriva@umich.edu

ORCID iD: 0000-0002-4684-4423

“A good idea has a way of becoming simpler and solving problems other than that for which it was intended”– Robert Tarjan

ACKNOWLEDGEMENTS

Firstly, I would like to express my sincere gratitude towards my advisor, Prof. Veera Sundararaghavan. He introduced me to this exciting area of research. He gave me full liberty to pursue problems that I was interested in and gave me time and guidance whenever I needed them. I'd like to thank my committee members, Professors Anthony Waas, Daniel Inman, Shravan Veerapaneni, and Alex Gorodetsky, for taking the time to review my work. Their suggestions have not only refined this dissertation but have also given me new ideas for my future research in this area.

I have had the opportunity to learn from great teachers at UM. Prof. John Shaw's courses on Mechanics and Stability have considerably streamlined my outlook on mechanics. It has helped me a lot in pursuing research in this area. The courses on Differential Equations and Analysis by Professors Sijue Wu, Zaher Hani, and Anna Gilbert have also influenced a great portion of my research. I am grateful to Professors Jenny Wilson, Alex Wright, and Alexandro Uribe for their Topology and Geometry courses.

A majority of this work had humble beginnings in coffee shop conversations. I am thankful to Aaditya for all the interesting discussions we had and the copious amount of caffeine we consumed with it. I also want to thank Kunal and Rohit for humoring my passion for disproving statements by counterexample. Many of our early ideas fell prey to it, but I hope that the wisdom we gained in the process will find some utility in the future. I had great fun collaborating with Pinar and Reza. I hope we will get to work together in the future as well. I would like to thank all the other MSSL research group members: Adam, Arunabh, Christian, Gurmeet, Iman, Jiyangi, Nick, Shardul, Srihari, and Sriram.

A special thanks to Krystal for her companionship over the past few years. A fun fact - she is the only person who has reviewed all my work. I am thankful to Puneet for being one phone call away whenever I needed help. Over the years, I have been in the company of some wonderful fellow grad students, my flatmates: Daning, Liren, Sicheng, and Zhengnan, and my office-mates: Abhinav, Ana, Brittany, and Logan. I appreciate them for their cheerful presence. I am thankful to the Aerospace graduate coordinators, Denise Phelps and Ruthie Freeman, the hidden figures behind successful Ph.D. defenses. This thesis would not have been possible without all these people who offered their advice and support during my stay in Ann Arbor.

Lastly, I'd like to express my gratitude to my parents and my brother, Siddhant, for their continuous encouragement and unfailing support throughout my years of study, including the process of researching and writing this thesis.

I also acknowledge the Air Force Office of Scientific Research Materials for Extreme Environments Program (Grant No. FA9550-18-1-0091) for financial support and the Universities Space Research Association (USRA) for providing access to the use of the D-Wave quantum computer in the USRA-NASA-Google quantum Artificial Intelligence Laboratory at NASA's Ames Research Center.

TABLE OF CONTENTS

LIST OF FIGURES

Figure

LIST OF TABLES

LIST OF APPENDICES

ABSTRACT

Computational methods are rapidly emerging as an essential tool for understanding and solving complex engineering problems, which complement the traditional tools of experimentation and theory. When considered in a discrete computational setting, many engineering problems can be reduced to a graph coloring problem. Examples range from systems design, airline scheduling, image segmentation to pattern recognition, where energy cost functions with discrete variables are extremized. However, using discrete variables over continuous variables introduces some complications when defining differential quantities, such as gradients and Hessians involved in scientific computations within solid and fluid mechanics. Consequently, graph techniques are under-utilized in this important domain. However, we have recently witnessed great developments in quantum computing where physical devices can solve discrete optimization problems faster than most well-known classical algorithms. This warrants further investigation into the re-formulation of scientific computation problems into graph-theoretic problems, thus enabling rapid engineering simulations in a soon-to-be quantum computing world.

The computational techniques developed in this thesis allow the representation of surface scalars, such as perimeter and area, using discrete variables in a graph. Results from integral geometry, specifically Cauchy-Crofton relations, are used to estimate these scalars via submodular functions. With this framework, several quantities important to engineering applications can be represented in graph-based algorithms. These include the surface energy of cracks for fracture prediction, grain boundary energy to model microstructure evolution, and surface area estimates (of grains and fibers) for generating conformal meshes. Combinatorial optimization problems for these applications are presented first.

The last two chapters describe two new graph coloring algorithms implemented on a physical quantum computing device: the D-wave quantum annealer. The first algorithm describes a functional minimization approach to solve differential equations. The second algorithm describes a realization of the Boltzmann machine learning algorithm on a quantum annealer. The latter allows generative and discriminative learning of data, which has vast applications in many fields. Theoretical aspects and the implementation of these problems are outlined with a focus on engineering applications.

CHAPTER I

Introduction

Many physical systems can be represented using discrete states, to name a few, polarization in magnetic lattices, different phases in a material, grain identifiers in a polycrystalline microstructure, etc. The state of numerical solvers for solving optimal distributions of these discrete variables, termed 'discrete combinatorial optimization', have greatly improved over the past few years. One such problem is on estimation of the ground state of classical Ising energy where optimum values of discrete variables (S_i) on a lattice are described by the minima of the following energy expression:

$$E(\boldsymbol{S}) = \sum_{i \in \text{ Nodes}} H_i S_i + \sum_{(i,j) \in \text{ Neighbors}} J_{ij} S_i S_j, \qquad S_i \in \{+1, -1\} \tag{1.1}$$

where J_{ij} and H_i are coupling weights and biases. The term pairwise energy encapsulates the idea that the system's total energy can be estimated as a summation of localized energies that depend only on the states of pairs of neighbors. Although local in energy description, the solutions are capable of showing long-range orders (correlation in states of objects which are far apart). As an example, see figure 1.1 where a ferromagnet and antiferromagnet is modelled using the Ising energy. It can be observed that changing the parameter, J, from negative to positive (or vice-versa) results in a global change of response of the model. This capacity to represent complicated solutions makes these energy models a lucrative choice to study complex

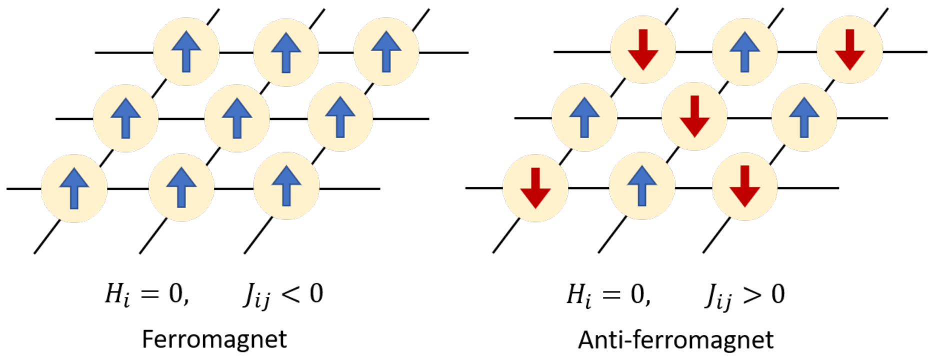


Figure 1.1: Ground states of Ising model on a square lattice. Arrows represent the +1(up) and -1(down) states

physical phenomena.

These models were initially developed to describe interacting spins on a crystalline lattice. Since then, they have become archetypal models in other fields involving operations research, network theory, and phase transition physics. Graner and Glazier described the motion of biological cells [5] using a large-Q Potts model where they represented cells as clusters of points with the same spin. A similar approach was used in [6] to study grain boundary motion in polycrystalline microstructures during thermally induced grain growth and recrystallization process. In such studies, the system's dynamics is represented as a transition probability governed by the model's energy description. These problems can be simulated using Monte Carlo-based simulations. On the other hand, there are problems where the equilibrium solutions are required; for instance, in computer vision, Potts model is often used to describe the cut energy of a segmentation problem (c.f.[7]).

In the completely general case, the problem of finding the ground state of this energy is NP-Hard. This implies no polynomial-time algorithm to solve this problem exists for a classical computer (based on a deterministic Turing machine). This problem is also shown to be exp-APX-complete, which implies no polynomial-time algorithm exists that can approximate a solution bounded by some constant factor of the global minimizer. However, many approximate algorithms are now available to solve certain restricted cases of this problem. Moreover, alternate computing architectures like quantum annealing [8] give access to the general case's ground state in much more reasonable time. This is one of the prime motivations for pursuing solutions of such problems on quantum computers.

Quantum computation also addresses the issue of high power consumption in computational data centers. A highly optimized large-sized data center (containing 100-400 thousand cores) has approximately 20 Watts energy consumption per core. The primary energy budget for a quantum processing unit is in the cooling costs as the quantum chip itself has a negligible energy budget. This contrasts with data centers that use four-fifths of the total energy for computations. Cooling to cryogenic temperatures (in the tens of milliKelvin range) requires approximately 20kW. A thousand-time speedup implies an approximate classical cost of 20kW (i.e., 1000 cores running at 20watts per core). This implies that energy costs are currently equivalent. However, with an increase in the number of qubits, it is expected that quantum computers will have lower power consumption than classical cores since the cooling costs are relatively unaffected by the number of qubits. These advances in computational prowess for solving pairwise energy models are the motivators for identifying, formulating and solving computational mechanics problems in this form. It is of interest to first look at problems that can be encoded as ground states of this model and therefore their physics can be emulated by directly solving the energy form. In contrast, in certain cases, it is also required to identify the model parameters that best represents the physics of the system. These problems are illustrated in Fig 1.2 where the problems of former type are referred to as the 'Forward problem' and the latter as the 'Inverse problem'.

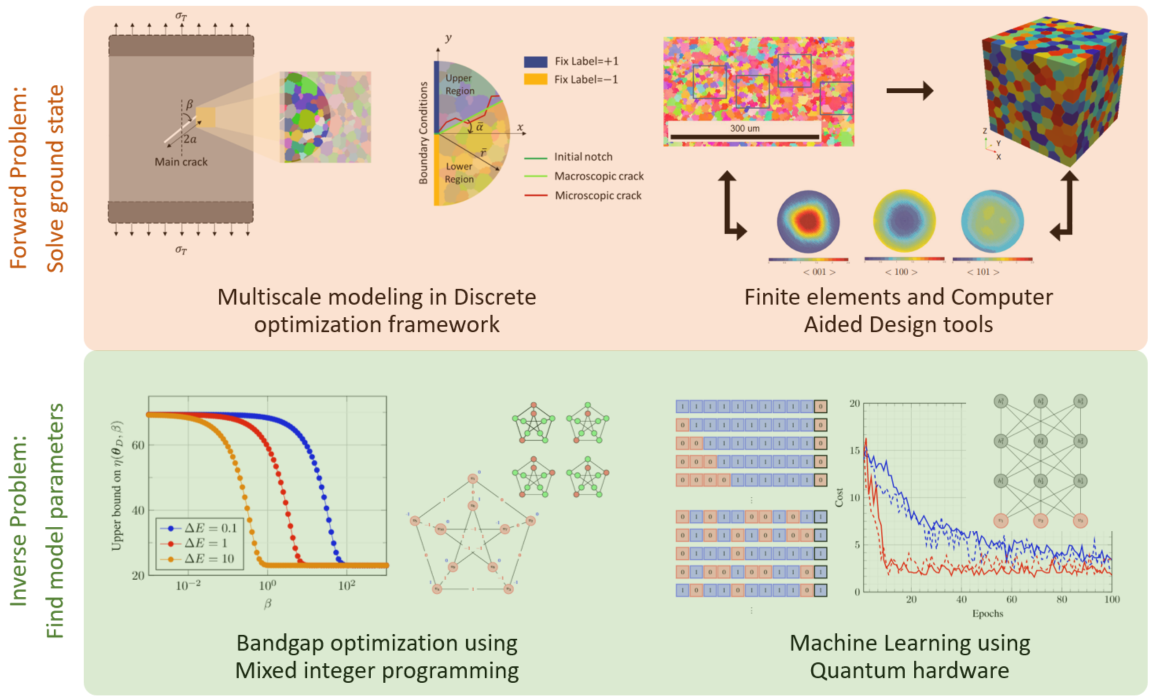


Figure 1.2: Illustration of research goals of this thesis

This chapter introduces the pairwise energy model, a generalization to the Ising model shown above. Related terminology like the ground state, bandgap, and probability of a state is also introduced. Some useful classifications of this energy and the complexity of solving these problems are also discussed. A discussion about these models' training strategies (Inverse problem) is presented in Section 1.3. The concept of Quantum Annealing is introduced in Section 1.4.

1.1 Discrete Pairwise energy models

A discrete pairwise energy model is defined on an undirected simple graph. In lieu of introducing some useful terms, following definition for graph is used:

Definition I.1 (Graph)**.** A graph, $\mathcal{G}$, is a pair of sets $(\mathcal{V}, \mathcal{C})$, where $\mathcal{V}$ is the set of vertices and $\mathcal{C}$ is the set of edges/connections. For each element $e \in \mathcal{C}$ there is a corresponding ordered pair $(x, y); x, y \in \mathcal{V}$ i.e. $\mathcal{C} \subseteq \mathcal{V} \times \mathcal{V}$. A Graph, $\mathcal{G} = (\mathcal{V}, \mathcal{C})$ is undirected if an edge does not have any directionality i.e $(x, y) \equiv (y, x)$. A graph is simple if $(x, x) \notin \mathcal{C}$ for all $x \in \mathcal{V}$.

Also, this work requires the graph to be finite, i.e., the number of vertices is finite. Next, the definition of Pairwise energy is introduced.
Consider a finite undirected simple graph $\mathcal{G}(\mathcal{V}, \mathcal{C})$. The number of vertices are denoted by $N_V = |\mathcal{V}|$ and the number of edges are denoted by $N_C = |\mathcal{C}|$. The indices of connections and vertices are related using the maps, π_1 and π_2 such that for a connection with index, $k \in \{1, .., N_C\}$, the index of the corresponding vertices are $\pi_1(k)$ and $\pi_2(k)$ with $1 \leq \pi_1(k) < \pi_2(k) \leq N_V$. This essentially means $e_k \equiv (v_{\pi_1(k)}, v_{\pi_2(k)})$. Each vertex, $v_i \in \mathcal{V}$ is assigned a state $s_i \in \{l_1, l_2, \ldots, l_{N_L}\}$ for all $i \in 1, \ldots, N_V$. This determines the complete state of the graph as an ordered tuple $\boldsymbol{S} = (s_1, \ldots, s_i, \ldots, s_n) \in \{l_1, \ldots, l_{N_L}\}^{N_V}$. The set of all possible states is referred to as $\mathcal{S} = \{l_1, \ldots, l_{N_L}\}^{N_V}$ with the total number of states denoted by $N_{TS} = |\mathcal{S}| = N_L^{N_V}$.

The energy for a particular state can be evaluated as follows:

$$E(\boldsymbol{S}) = \sum_{i=1}^{N_V} U_i(s_i) + \sum_{k=1}^{N_C} V_k \left(s_{\pi(k,1)}, s_{\pi(k,2)} \right) \tag{1.2}$$

where, $U_i : \mathcal{L} \to \mathbb{R}$ is the energy of labeling i^{th} vertex with some label, and $V_k : \mathcal{L} \times \mathcal{L} \to \mathbb{R}$ is the energy of labeling two connected vertices. The former term U_i is known as the unary cost and the latter term V_k is known as the Pairwise term.
If the pairwise term is discarded, then the minimization of Eq (1.2) can be carried out by independently minimizing the unary cost of each vertex. However, pairwise cost removes this independence and makes the optimization process complicated. Despite the local nature of the pairwise term, long-range effects are introduced in the overall optimization.

Definition I.2 (Potts Model)**.** Following simplified form of energy is referred to as the Potts model in this thesis:

$$E(\boldsymbol{S}) = \sum_{i=1}^{N_V} U_i(s_i) + \sum_{k=1}^{N_C} J_k V\left(s_{\pi(k,1)}, s_{\pi(k,2)}\right) \tag{1.3}$$

The parameters, J_k are real valued constant. It is easy to see that this a special case of Energy 1.2. Usually, Potts model is referred to a specific instance of Eq (1.3) with $V(a,b) = 1$ when $a \neq b$ and $V(a,a) = 0$ and $J_k > 0$, but no such assumption is made here.

Moreover, following symmetry is imposed which signifies that the pairwise only depends on the combination of two labels and not their order:

$$V_k(b,a) = V_k(a,b), \qquad a, b \in \{1, ..., N_L\}$$

Definition I.3 (Ising Model)**.** There are two popular versions of this model. The first one uses labels, $\mathcal{L} = \{0, 1\}$ and the second one uses labels $\mathcal{L} = \{-1, +1\}$. The energy in both cases is evaluated as:

$$E(\boldsymbol{S}) = \sum_{i=1}^{N_V} H_i s_i + \sum_{k=1}^{N_C} J_k s_{\pi(k,1)} s_{\pi(k,2)} \tag{1.4}$$

The parameters, H_i and J_k are referred to as the field and interaction parameter, respectively. This is a special case of the above energies with

$$U_i(l) = l, \quad V(l_a, l_b) = l_a l_b$$

1.1.1 Ground states and band gap

The parameters that define the unary and pairwise cost can be abstracted and denoted as a set, $\boldsymbol{\theta}$. The set of ground states ($\mathcal{S}_G(\boldsymbol{\theta}) \subseteq \mathcal{S}$) is the set of states with minimum energy, $E_0(\boldsymbol{\theta})$), i.e.

$$\mathcal{S}_G(\boldsymbol{\theta}) = \text{argmin}_{\boldsymbol{S}\in\mathcal{S}}\, E(\boldsymbol{S}|\boldsymbol{\theta}), \qquad E_0(\boldsymbol{\theta}) = \min_{\boldsymbol{S}\in\mathcal{S}}\, E(\boldsymbol{S}|\boldsymbol{\theta})$$

In contrast, all the non-minimal states are referred to as exited states. The set of all excited states, denoted by $\mathcal{S}_E(\boldsymbol{\theta})$, can be evaluated as:

$$\mathcal{S}_E(\boldsymbol{\theta}) = \mathcal{S} - \mathcal{S}_G(\boldsymbol{\theta})$$

The cardinalities of the set of ground states ($\mathcal{S}_G$) and excited states ($\mathcal{S}_E$) are denoted by N_{GS} and N_{ES}, respectively. All excited states may or may not have the same energy. However, the minimum excited energy referred to as the 'first excited energy' is used in defining the band gap and is evaluated as:

$$E_1(\boldsymbol{\theta}) = \min_{\boldsymbol{S}\in\mathcal{S}_E(\boldsymbol{\theta})}\, E(\boldsymbol{S}|\boldsymbol{\theta})$$

It should be noted that no assumption is made on the multiplicity of states with energy $E_1(\boldsymbol{\theta})$. The band gap (a positive quantity) defines the energy gap between $\mathcal{S}_G$ and $\mathcal{S}_E$. It is estimated as:

$$\Delta E(\boldsymbol{\theta}) = E_1(\boldsymbol{\theta}) - E_0(\boldsymbol{\theta})$$

1.1.2 Probability distribution

At any given temperature, T, the probability of occurrence of a state, $\boldsymbol{S}$ is described by the Boltzmann distribution as:

$$p(\boldsymbol{S}|\boldsymbol{\theta}, \beta) = \frac{1}{Z} e^{-\beta E(\boldsymbol{S})} \tag{1.5}$$

where $\beta = 1/k_B T$ is the inverse thermodynamic temperature, k_B is the Boltzmann constant and Z denotes the partition function which is estimated as

$$Z = \sum_{\boldsymbol{S} \in \mathcal{S}} e^{-\beta E(\boldsymbol{S})}$$

1.1.3 Classification

The computational complexity of finding ground states of this model, as shown in the next section, depends on (i) Topology of the graph, and (ii) Pairwise cost function. The classification of these Pairwise functions are presented in this section. Most definition are adopted from Bagon's thesis [1]. Without loss of generality, it can be assumed that $\mathcal{L} \subseteq \mathbb{R}$. This provides a natural way of inducing order as well as metric on the set $\mathcal{L}$.

Definition I.4 (Submodular set function)**.** In general, a submodular function is defined as a set function. Consider a set Ω and function $f : \mathcal{P}(\Omega) \to \mathbb{R}$ where $\mathcal{P}(\Omega)$ is the power set of Ω. Then f is submodulur set function iff for all $X, Y \subseteq \Omega$,

$$f(X) + f(Y) \geq f(X \cap Y) + f(X \cup Y)$$

A pedogogical example for a submodular function is the entropy function and a non-submodular function is the cardinality.
In the context of Pairwise energy, this definition can be specialized as follows:

Definition I.5 (Submodular energy)**.** For any choice of labels, $a, b, c, d \in \mathcal{L}$ such that $l_a \leq l_b, l_c \leq l_d$, Eq (1.2) is submodular iff following inequality holds true for all k:

$$V_k(l_a, l_b) + V_k(l_c, l_d) \leq V_k(l_a, l_d) + V_k(l_b, l_c)$$

Example: ℓ_1-distance, estimated as $V_k(l_a, l_b) = |l_a - l_b|$ is submodular.

Equivalent definition: Submodularity can also defined using the following recurssive relation:

$$V_k(l_a, l_b) + V_k(l_{a+1}, l_{b+1}) \leq V_k(l_a, l_{b+1}) + V_k(l_{a+1}, l_b)$$

Remark 1: The second definition highlights the useful information that the matrix $(M)_{ab} \equiv V_k(l_a, l_b)$ represents a *Monge Matrix*. It is also easier to verify for a given pairwise energy

Remark 2: In the binary labeling problem, the submodularity condition becomes:

$$V_k(l_1, l_1) + V_k(l_2, l_2) \leq V_k(l_1, l_2) + V_k(l_2, l_1)$$

In the case of symmetric pairwise energy, this suggests that the cost can be minimized either by assinging labels (l_1, l_1) or (l_2, l_2) but contrasting labels are sub-optimal. This property in fact can be generalized to multilabel case to assert that submodular energy lowers the cost of non-contrasting labeling.

Definition I.6 (Convex energy)**.** Eq (1.2) is convex iff for each k

$$V_k(l_a, l_b) = g_k(l_a - l_b)$$

where $g_k : \mathbb{R} \to \mathbb{R}$ is a real-valued convex function i.e. for any $x, y \in \mathbb{R}$ and $t \in [0, 1]$,

$$g(tx + (1-t)y) \leq tg(x) + (1-t)g(y)$$

Example: $V_k(l_a, l_b) = (l_a - l_b)^2$ is convex. ℓ_1-distance is not convex.

Remark: All convex pairwise energies are submodular [1].

Definition I.7 (Relaxed metric energy)**.** Eq (1.2) is Relaxed metric iff

$$V_k(l_a, l_a) + V_k(l_b, l_c) \leq V_k(l_a, l_b) + V_k(l_a, l_c)$$

Example: ℓ_1-distance and truncated ℓ_1- distance $V_k(l_a, l_b) = \min\{(l_a - l_b), \tau\}$ is relaxed metric. Truncated ℓ_1-distance is not submodular.

Remark: Submodular energies in general need not be metric.

Definition I.8 (Relaxed semi-metric energy)**.** Eq (1.2) is Relaxed semi-metric iff

$$V_k(l_a, l_a) + V_k(l_b, l_c) \leq V_k(l_a, l_b) + V_k(l_a, l_c)$$

Example: $V_k(l_a, l_b) = \min\{(l_a - l_b)^2, \tau\}$ is relaxed semi-metric.

Remark: Every relaxed metric energy is relaxed semi-metric.

The containment of different classes is presented in Fig 1.3. In the next section, classical computation methods for the different classes are discussed.

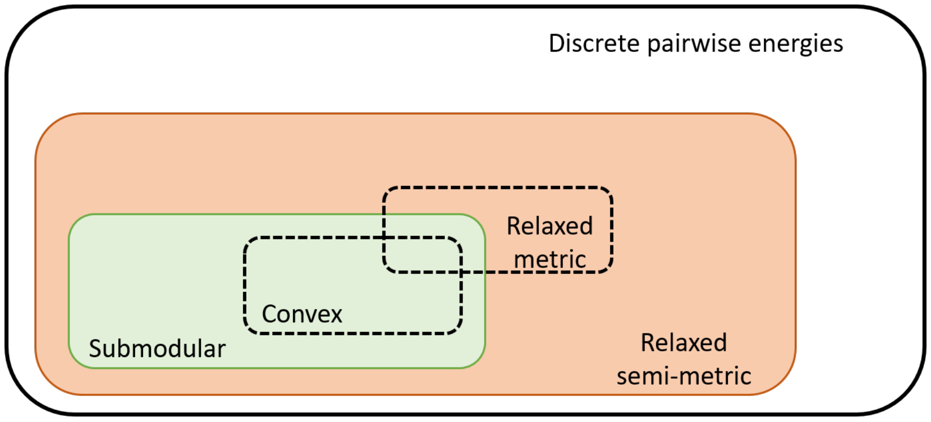


Figure 1.3: Classification of Pairwise discrete energies adopted from [1]. Green indicates the existence of global minimization algorithms. For energies in red there are good approximation algorithms.

1.2 Classical methods and Computational Complexity

It is a known fact that finding ground states of Eq (1.2) is NP-Hard [9, 10, 11]. This statement is proved in [9] by reducing this problem to the maximum independent set problem, which is known to be NP-hard. The authors showed it for the case with binary labels, but the result extends to more than 3 labels as it contains the former case. They also showed that binary labeling could be solved in polynomial time if the energy is submodular. [12] introduced a mapping from multi-label problems with convex energies to binary labeling problems with submodular energies, thereby showing a polynomial-time algorithm for multi-label convex energies. A generalization to this approach is presented in [11] where the reduction is made from submodular multi-label problem to submodular binary problem, thereby proving the polynomial time complexity of the multi-label submodular problem. These special cases can also be defined in terms of Graph connectivity. For instance, Belief propagation can be employed on graphs with a tree structure (no cycles are present) to obtain the global minimum in polynomial time. Another specialization of this problem was given by [10] where it was shown that the Multiway Cut problem on a planar graph could be solved in polynomial time. The definition of the Multiway Cut problem is given below:

Definition I.9 (Multiway K-Cut problem [13])**.** Given an undirected graph, $\mathcal{G} = (\mathcal{V}, \mathcal{C})$, connection weights, $w_i : e_i \rightarrow \mathbb{R}^+ \ \forall e_i \in \mathcal{C}$, and a set of terminals $\mathcal{T} = \{t_1, t_2, ..., t_k\} \subseteq \mathcal{V}$, a multiway cut is a set of edges (connections) $\mathcal{C}'$ that leaves each of the terminals in a separate component. The cost of the multiway cut is defined as the sum of the weights of the edges (connections) in $\mathcal{C}'$. The goal of the multiway cut problem is to minimize this cost such that removing $\mathcal{C}'$ from $\mathcal{C}$ separates all terminals. In other words, no connected component of $\mathcal{G}(V, \mathcal{C} - \mathcal{C}')$ contains two terminals from $\mathcal{T}$.

Remark: This problem can equivalently and rather more intuitively be presented as a labeling problem where each vertex in $\mathcal{V}-\mathcal{T}$ is assigned a label with $N_L = k$. Following form of Potts energy is minimized where δ refers to Kronecker delta function:

$$E(\boldsymbol{S}) = \sum_{i=1}^{N_C} w_i \left(1 - \delta_{s_{\pi(i,1)}, s_{\pi(i,2)}}\right) \tag{1.6}$$

The multiway cut is defined as a set of connections between vertices with different labels.

Although global minima are obtained for these specializations in polynomial time, the runtimes are rarely fast enough to be employed in practical scenarios. This problem has led to extensive research in the area of approximate algorithms for solving these problems. In classical computation, these approximate methods can be classified as local update methods and global update methods. One of the simplest approaches is Iterated Conditional Modes (ICM) [14] which is a greedy algorithm that iterates over each vertex and picks the best label while keeping all the other vertices' labels fixed. This method works for a general pairwise energy model but can get stuck in local minima and is sensitive to the initial condition. A probabilistic version of this method is Simulated Annealing (SA) [15]. In this approach, the fictitious temperature is lowered (to emulate the annealing process), and the states are modified based on the Boltzmann probability distribution (Eq (1.5)). Belief propagation (BP) methods are also local update methods which use a message-passing approach between neighbor nodes to identify the optimal label [16]. When the graph has a tree structure, these messages are passed from the root to the leaves and vice-versa. This forward-backward propagation drives the system to the globally optimal solution even for the general case of energy. For the graph with cycles, a variant of BP called the loopy BP is proposed. However, it does not guarantee convergence to the globally optimal solution. In another variant of BP, called the Tree Re-Weighted (TRW) BP [17], a continuous Linear programming relaxation of the problem was proposed. This assumption is used to establish a message passing scheme tailored for the convex optimization of Linear Programming. TRW-BP has been shown to achieve a global minimum for relaxed metric energies.

In contrast to local methods, large move methods are based on combinatorial optimization techniques. This name comes from the fact that each step of the iteration involves large binary moves. The overall efficiency of these methods are hence, based on the efficiency of these iteration steps. The three popular variants of these methods are α-expansion, $\alpha\beta$-swap [7], and fusion-moves [18]. The α-expansion method is appropriate for relaxed metric energy. In this case, each iteration involves solving a binary submodular problem. Moreover, convergence is guaranteed after finite steps, and the global error (difference in the energy of approximated and global minima) is also bounded. The $\alpha\beta$-swap method is appropriate for relaxed semi-metric energy. In this case, as well, each iteration step involves solving a binary submodular problem. Convergence is guaranteed after finite steps, but there is no non-trivial bound on the global error. Fusion-moves method is based on Quadratic Pseudo-Boolean Optimization (QPBO), which involves adding redundant nodes in the graph to get rid of non-submodular edge weights. This addition of node adds redundancy to the labeling of the initial graph. In this case, the global minimum recovery is no longer guaranteed, but the algorithm may recover partial labeling that is guaranteed to be part of some globally optimal solution. It has been shown by [19] that general energy minimization, even in the 2-label pairwise case, and planar energy minimization with three or more labels are exp-APX-complete, i.e., there exists no polynomial-time method that can approximate a solution that is at worst a constant factor worse than the global minimum (in terms of energy).

1.3 Inverse parameter estimation

Much like development of simulation techniques, it is also of interest to develop training (parameter estimation) technique to deploy these methods for practical engineering problems. Inverse parameter estimation entails identification of parameters (eg. H and J in eq. 1.1) such that desired ground states (S) are obtained. This problem is useful for classification and regression problems where a graph model is to be constructed that reproduces known data. Traditionally, these models are trained by considering them as Markov Random Fields (MRFs) and using gradient-based approaches to maximize the likelihood [20]. However, analytical estimates of the gradients are hard to obtain. Among approximate techniques, Hinton's contrastive divergence method [21] provides an efficient way to approximate the gradients in the parameter optimization problem successively. An excellent review of this subject is presented in [22]. It should however be noted that these likelihood-based technique use the Boltzmann distribution to sample model's states. Consequently, the techniques assume finite temperature for simulation. This restricts estimation of precise parameters for encoding desired ground states in the model. As an example, the negative log-likelihood of a model trained using Likelihood-based technique is presented in Fig 1.4. It can be seen that the minimum is close to the training β (inverse temperature), which was chosen as $\beta = 1$.

In contrast, an approach based on the band gap maximization while ensuring the ground states are data states guarantees that the states' probability distribution gets closer to that of the data set as the temperature is reduced. Moreover, it ensures that the model adequately represents the data set for a broad range of temperatures. However, the downside of this approach is that there is no guarantee of the existence of parameters for every data set. This fact can be easily motivated by noticing that the number of ground states can be more than the number of model parame-

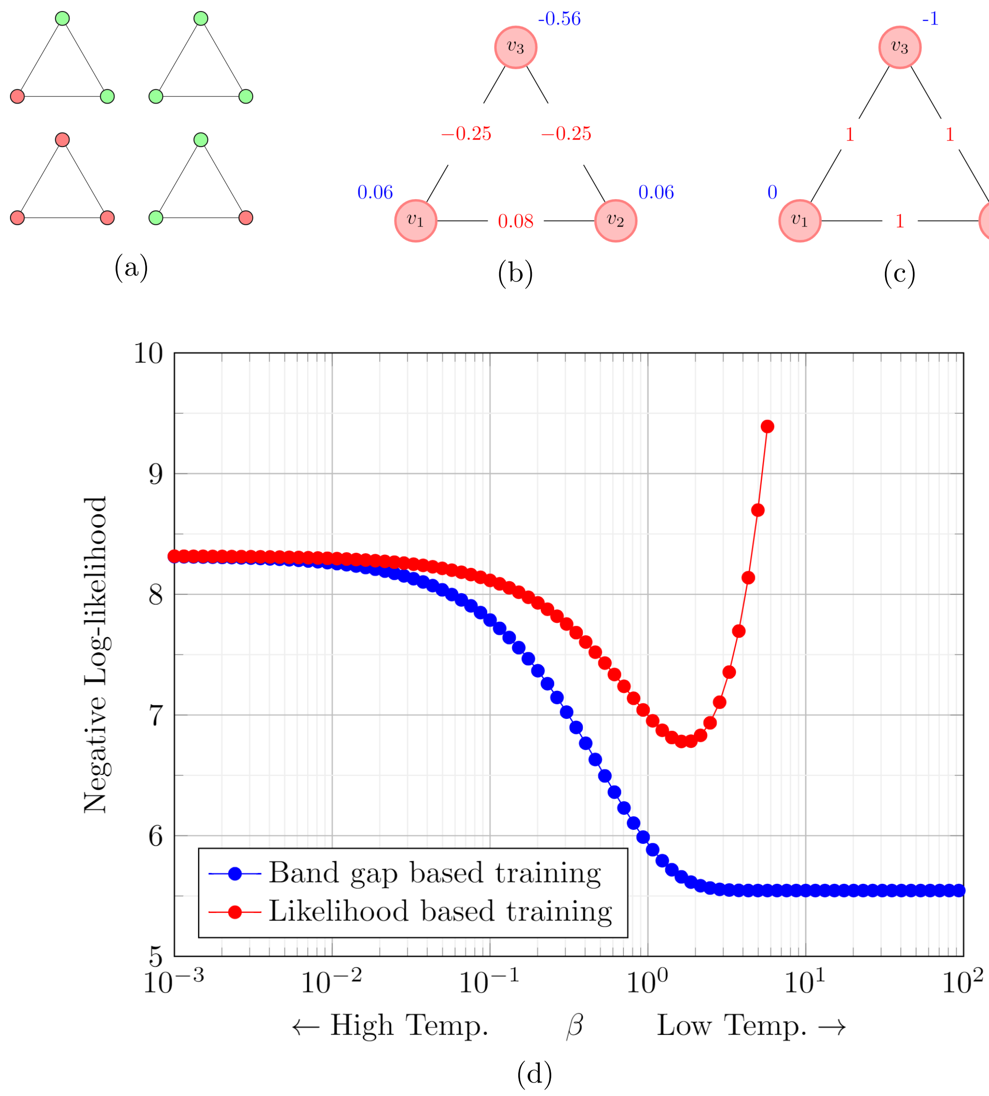


Figure 1.4: The energy of the graph is modeled using Ising model Eq (1.12) with $|H| \leq 1$ and $|J| \leq 1$. (a) Training data set of states with green representing a '+1' state and red representing a '-1' state. (b) Optimized graph using minimization of Negative Log-likelihood at $\beta = 1$ (c) Optimized graph using PEPDAS method (Appendix C). The field terms are mentioned in blue color and interaction terms are mentioned in red color (d) Comparative analysis of likelihoods of models trained using the Likelihood maximization and band gap maximization. A lower value of Negative Log likelihood signifies a better trained model

ters and may result in an over-constrained optimization problem. Such problems do not exist at a non-zero temperature as all the states appear with non-zero probability.

A Mixed Integer Linear Programming (MILP) formulation is presented in Appendix C to estimate a specialized version of Potts model (Eq (1.3)) where $U_i(l) = H_i U(l)$ so the total energy is given as:

$$E(\boldsymbol{S}) = \sum_{i=1}^{N_V} H_i U(s_i) + \sum_{k=1}^{N_C} J_k V\left(s_{\pi(k,1)}, s_{\pi(k,2)}\right) \tag{1.7}$$

The model is optimized for the parameters, $\{H_i\}$ and $\{J_k\}$ while fixing the energy functions U and V. Two variations of the algorithm are presented. The first algorithm assigns a prescribed data set as the model's ground states while maximizing the bandgap. The second algorithm identifies a set of ground states with a prescribed multiplicity while maximizing bandgap. It should be noted that the computational complexity of both the algorithms grows exponentially with the size of the problem. Therefore, these methods are only suited for small graph structures. These problems arise in designing energies of smaller motifs in a lattice structure.

The parameter set is represented as a vector, $\boldsymbol{\theta} = \left[\theta_1, \ldots, \theta_{N_V+N_C}\right]^T$. In this work, it is specialized to following form:

$$\boldsymbol{\theta} = \left[H_1, \ldots, H_{N_V}, J_1, \ldots, J_{N_C}\right]^T$$

This notation allows to describe energy as a matrix-product evaluated as $E(\boldsymbol{S}|\boldsymbol{\theta}) = \boldsymbol{\varepsilon}(\boldsymbol{S})\boldsymbol{\theta}$ where $\boldsymbol{\varepsilon}(\boldsymbol{S})$

$$\boldsymbol{\varepsilon}(\boldsymbol{S}) = \left[U(s_1), \ldots, U(s_{N_V}), V\left(s_{\pi_1(1)}, s_{\pi_2(1)}\right), \ldots, V\left(s_{\pi_1(N_C)}, s_{\pi_2(N_C)}\right)\right]$$

Given a data set, $\mathcal{S}_D \subseteq \mathcal{S}$, the parameters set, $\boldsymbol{\theta}$, is optimized such that the states in $\mathcal{S}_D$ have higher probability of occurrence at a prescribed β value. Mathematically, this procedure entails minimization of negative log-likelihood as defined below:

$$\eta(\boldsymbol{\theta}, \beta) = -\sum_{\boldsymbol{S} \in \mathcal{S}_D} \log p(\boldsymbol{S}|\boldsymbol{\theta}, \beta) \tag{1.8}$$

It can be observed that at high temperatures i.e. $\beta \to 0$, all states occur with equal likelihood and therefore

$$\eta_0 = \lim_{\beta \to 0} (\boldsymbol{\theta}, \beta) = N_{DS} \log(N_{TS})$$

where $N_{DS} = |\mathcal{S}_D|$. On the other hand, at low temperatures i.e. $\beta \to \infty$, only ground states occur with equal probability and occurrence of any other state has probability 0. Consequently, the value of η in this limit is finite only when $\mathcal{S}_{\mathcal{D}} \subseteq \mathcal{S}_G$. It is evaluated as:

$$\eta_\infty(\boldsymbol{\theta}) = \lim_{\beta \to \infty} (\boldsymbol{\theta}, \beta) = N_{DS} \log(N_{GS})$$

It is desirable to estimate parameters such that the ground state replicates the data set, and the bandgap is maximized. The reason will be apparent after the next theorem.

Theorem I.10. *For a given set of parameters, $\boldsymbol{\theta}_D$, such that (i) $\mathcal{S}_G(\boldsymbol{\theta}_D) = \mathcal{S}_D$ (ii) $\Delta E > 0$, following statements hold true:*

(a) $\eta(\boldsymbol{\theta}_D, \beta)$ monotonically decreases with β and the low temperature limit

$$\eta_\infty(\boldsymbol{\theta}_D) = \lim_{\beta \to \infty} \eta(\boldsymbol{\theta}_D, \beta) = N_{GS} \log(N_{GS}) \tag{1.9}$$

(b) $\eta(\boldsymbol{\theta}_D, \beta)$ *is bounded as:*

$$N_{GS}\log(N_{GS}) < \eta(\boldsymbol{\theta}_D, \beta) \leq N_{GS}\log\left(N_{GS} + N_{ES}e^{-\beta\Delta E}\right) \tag{1.10}$$

(c) For any $\epsilon > 0$*, there exists a* β^* *such that for all* $\beta > \beta^*$*,* $\eta(\boldsymbol{\theta}_D, \beta) - \eta_\infty(\boldsymbol{\theta}_D, \beta) < \epsilon$ *where* β^* *is estimated as:*

$$\beta^* = \frac{1}{\Delta E}\left(\log\frac{N_{ES}}{N_{GS}} - \log\left(e^{\epsilon/N_{GS}} - 1\right)\right) \tag{1.11}$$

■

The consequence of this theorem is that it is guaranteed that if the parameters are chosen appropriately, η will approach to its global minimum in the low temperature (high β) limit. Moreover, at a finite β, η is bounded from above by a decreasing function. It can be seen in Fig 1.5(a), that the bound gets tighter for higher values of ΔE. It is also shown that the trained model is efficient in the range of β determined by $[\beta^*, \infty)$. Fig 1.5(a) shows that a higher bandgap allows a broader range of temperatures.

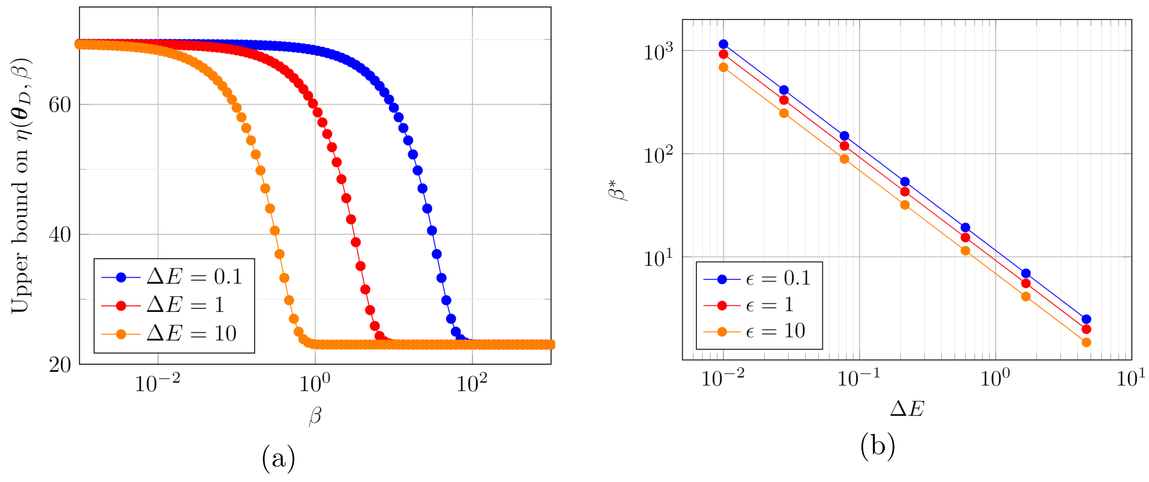


Figure 1.5: An illustration of bounds for a trained Potts model with $N_{GS} = 10$ and $N_V = 10$. (a) The upper bound on η with respect to β for various values of energy gap (b) β^* as a function of band gap for various bounds on η

1.3.1 Examples

Two classical algorithms are analyzed for estimating parameters of Potts model using a Mixed Integer Linear Programming approach (details presented in Appendix C). The functionality of each method is as follows:

1. PEPDAS (Parameter Estimation for Potts model with Data Set): This method estimates the parameters to exactly replicate the ground states as the prescribed data set.

2. PEPGSM (Parameter Estimation for Potts model with Ground State Multiplicity): method estimates the parameters to identify ground states based on their prescribed number.

The parametric estimation of Ising model is presented as an application of the methods. In this model, the states take a binary form i.e. $N_L = 2$. Traditionally the labels are denoted as $\{+1, -1\}$ and the corresponding energy functions are defined as:

$$U(+1) = +1, \quad U(-1) = -1$$
$$V(+1,+1) = V(-1,-1) = 1, \quad V(+1,-1) = -1$$

Therefore, the energy can be effectively written as:

$$E(\boldsymbol{S}) = \sum_{i=1}^{N_V} H_i s_i + \sum_{k=1}^{N_C} J_k s_{\pi(k,1)} s_{\pi(k,2)} \tag{1.12}$$

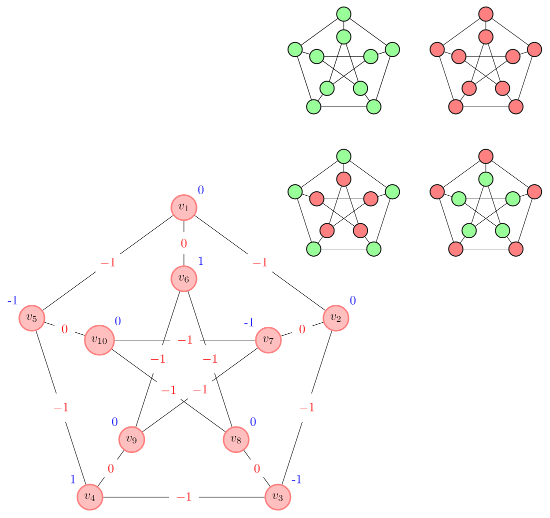
This model is applied on a 10-noded Peterson graph with $|H| \leq 1$ and $|J| \leq 1$. First, the graph is trained by prescribing up to 4 data states using the PEPDAS method. Next, the graph is trained by prescribing the number of states from 1 to 4 using the PEPGSM method. The predicted band gaps are shown in Table 1.1. It can be observed that the PEPGSM method predicts the same bandgap as the PEPDAS method for data sets with a size up to 3. However, for 4 data points, the PEPGSM method can identify ground states that provide higher bandgap. The predicted parameters for a graph with four ground states are shown in Fig 1.6. Likelihood estimates are not well defined in the case of PEPGSM method as it is not trained using the data. However, for comparison, η is estimated using the set of ground states in place of the data set. The results for negative log-likelihood of the PEPDAS predicted model and PEPGSM predicted model are shown in Fig 1.6(c). As expected, PEPGSM predicted model performs better than PEPDAS predicted model in terms of the range of β for which they can be used. The details of the other three models are presented in Fig 1.7 and Fig 1.8.

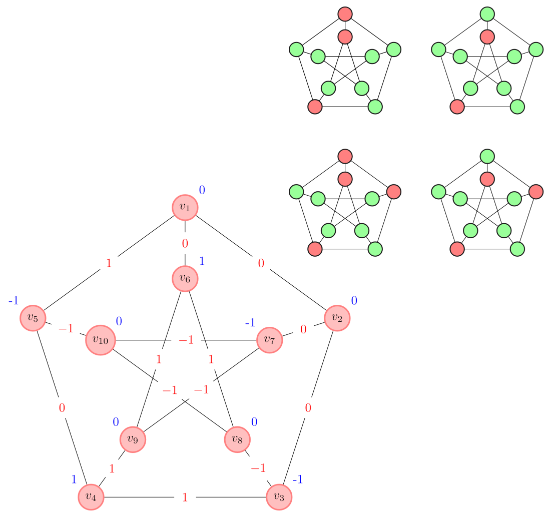
Algorithm	$N_{GS} = 1$	$N_{GS} = 2$	$N_{GS} = 3$	$N_{GS} = 4$
PEPDAS	8.0	6.0	4.0	6.0
PEPGSM	8.0	6.0	4.0	4.0

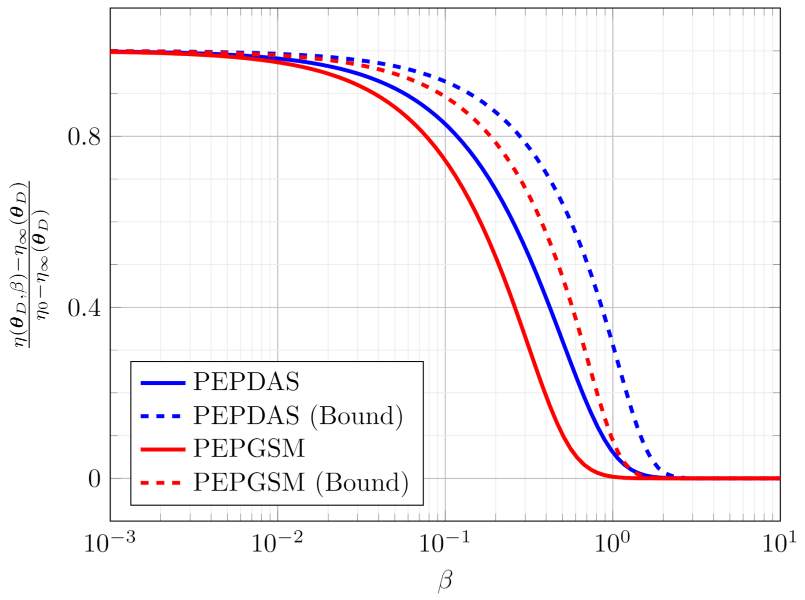
Table 1.1: Predicted maximum band gap for Peterson graph

Both algorithms maximize the band gap between the ground and excited states of the model. However, these two methods do not scale well with the graph size, and their usage should be restricted to small problems. It is shown by example that the predicted η decays and is bounded. Moreover, the PEPGSM method can predict ground states that provide higher bandgap compared to randomly picked ground states. It has been shown in Appendix C.2, that the computational cost of these methods grow exponentially with the size of the graph. This is an expected outcome considering the general difficulty in finding the solution of the labeling problem. Therefore, this proposed ‘classical’ method is only suited to small graph structures. For larger graphs, training is restricted to approximate methods as suggested at the beginning of the section.

However, even in the likelihood-based techniques, sampling independent states is a


(a)


(b)



(c)

Figure 1.6: Optimal Ising parameters of a Peterson graph with 4 ground states found using (a) PEPDAS method, and (b) PEPGSM method. The ground states are presented as the colored graph in the top right corner of each image. A green label denotes the '+1' state, and the red label denotes the '−1' state. (c) The normalized Negative log-likelihood of the optimized graphs

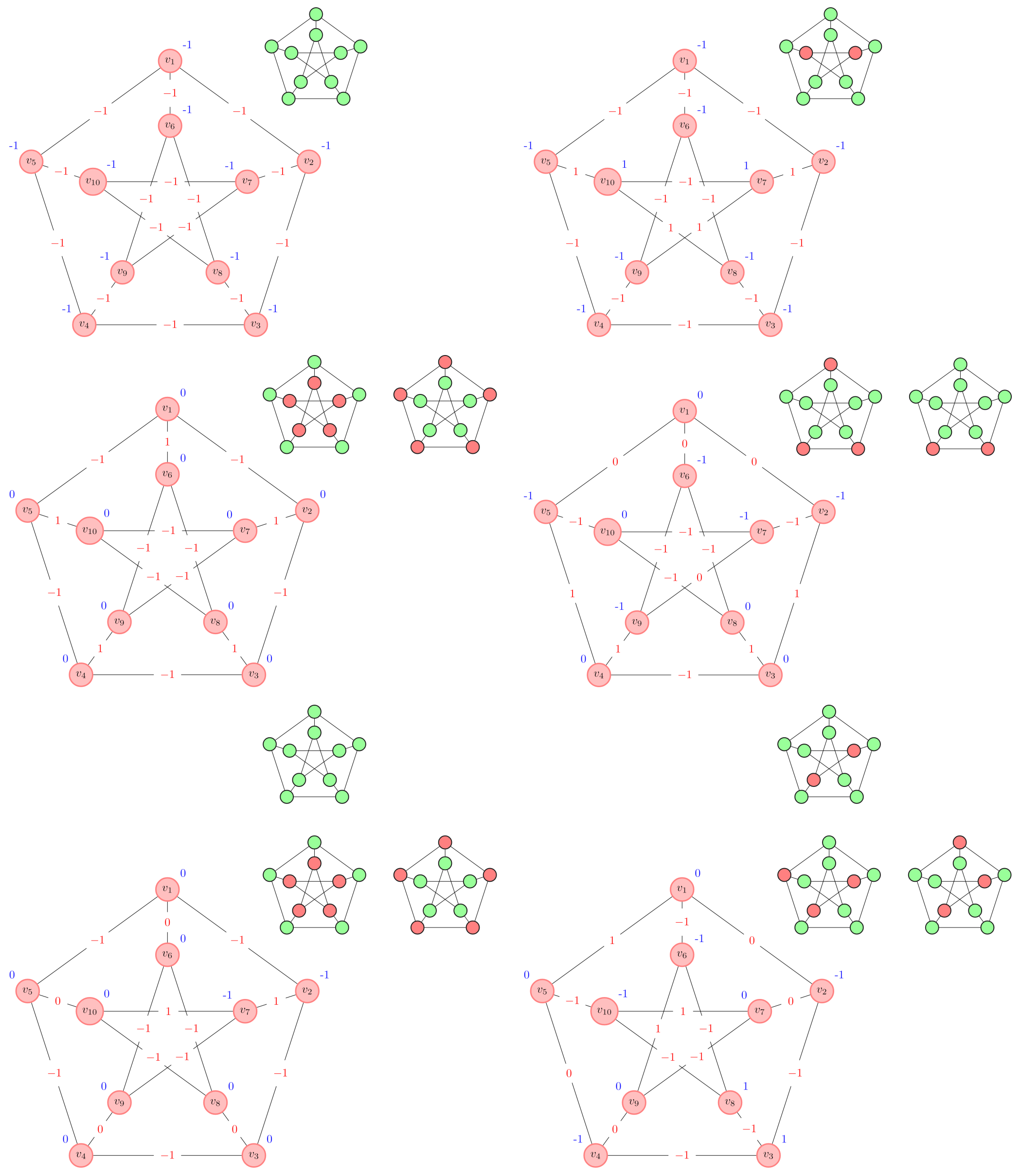


Figure 1.7: The energy of the graph is modeled using Ising model Eq (1.12) with $|H| < 1$ and $|J| < 1$. Optimal Ising parameters of a Peterson graph found using PEPDAS method (left) and PEPGSM method (right). The ground states are presented as the colored graph in the top right corner of each image. A green label denotes the '+1' state and the red label denotes the '−1' state.

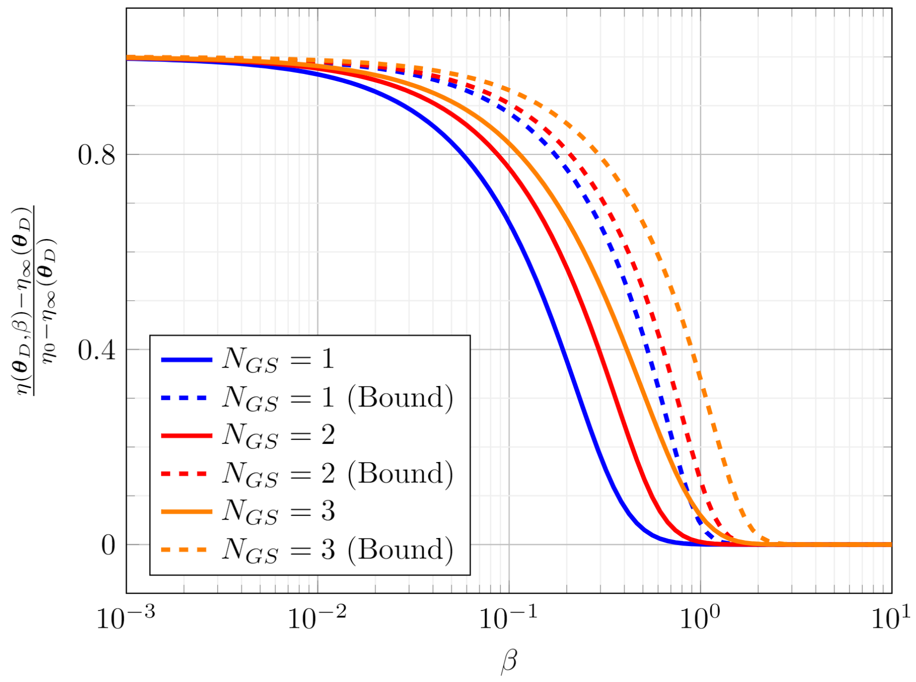


Figure 1.8: Normalized Negative log likelihood and their respective bounds for Peterson graphs trained using PEPDAS method

challenging task. This motivates the use of unorthodox computation architecture where these problems can be solved in a reasonable time. In the next section, the concept of Quantum annealing (QA) is presented, which tackles exactly this problem. QA has made it easier to sample states from the model's probability distribution [23]. This development has significantly eased the accurate approximation of the required gradients; consequently, it has lowered the hurdles to use probabilistic techniques to estimate parameters of Ising model.

1.4 Quantum Annealing

Richard Feynman's statement [24] "with a suitable class of quantum machines you could imitate any quantum system, including the physical world" has driven our vision towards a machine that can solve computational problems inaccessible to classical computers. Early versions of such quantum computers have already appeared. Mirroring gate-based classical computers, gate-based quantum computers with a small number of qubits have been demonstrated and promise an eventual path towards universal quantum computation. However, noise limits the number of gate operations that can be enforced before the quantum states decohere.

At the same time, Adiabatic quantum computation (AQC) is another form of quantum computing based on the evolution of the quantum state ($|\psi(t)\rangle$) is governed by a time-dependent Hamiltonian ($H(t)$) that is governed by the following time dependent Schrodinger's equation:

$$i\frac{d}{dt}|\psi(t)\rangle = H(t)\,|\psi(t)\rangle$$

$H(t)$ interpolates between an initial Hamiltonian (H_0), whose ground state is easy to construct, and a final Hamiltonian (H_f), that describes the proposed objective. As an example, one can consider following interpolation where $\tau = t/t_a$ with t being the physical time and t_a being some time scale.

$$H(\tau) = A(\tau)H_0 + (1 - A(\tau))H_f$$

The function A is monotonic with $A(0) = 1$ and $\lim_{\tau\to\infty} A(\tau) = 0$.the Quantum Adiabatic theorem, in loose terms, states that if the $H(t)$ maintains a positive band gap and the transition from initial to final Hamiltonian is done slowly enough, then the state $|\psi(t)\rangle$ moves towards the ground state of H_f [25]. This theoretical process has been adapted into the Quantum annealing procedure where following Hamiltonians are used for interpolation:

$$H_i = -\sum_{v\in\mathcal{V}} \sigma_i^x, \qquad H_f = \sum_{i=1}^{N_V} h_i\sigma_i^z + \sum_{k=1}^{N_C} J_k\sigma^z_{\pi(k,1)}\sigma^z_{\pi(k,2)}$$

where $\sigma_i^{x,z}$ are Pauli matrices that operate on spin or qubit i, h_i and J_k are tunable parameters. Moreover, finite time for interpolation is considered i.e. $A(0) = 0$ and $A(1) = 1$. In this case, t_a is referred to as the annealing time. The ground states of the Hamiltonian can be directly mapped to the ground states of Ising Energy Eq (1.4) with $\{-1, +1\}$ states. It can be verified that this energy is non-submodular when $J_k > 0$ and hence difficult to approximate on the classical computation. It has been shown by [26], that the state departs from the initial state with a power law asymptotics. It should be noted that there are no safeguards in place to maintain the assumptions of the Quantum Adiabatic theorem. In fact, having short annealing time (in the order of few μs) is in clear violation of the slow transition assumption. However, experimental studies support the efficiency of this procedure even in the adverse theoretical environment.

At present, the two popular QA devices are (i) the "DWave 2000Q" system with approximately 2000 qubits connected in a Chimera topology and (ii) the "DWave Advantage" system with approximately 5000 qubits connected in a Pegasus topology. The sparsity of the physical devices limits the size of the largest graph for different problems simulated on these systems. The largest graph size for popular problems is presented in Table 1.2. These systems operate at extremely low (of the order of few mK) but non-zero temperatures. Consequently, the QA sampled states follow a Boltzmann distribution as described by Eq (1.5). However, instead of being the annealing temperature, the sampling temperature is shown to be instance-dependent [27] i.e., dependent on the graph's topology and energy parameter, etc. As a result, the ground state is usually selected by post-processing the sample data.
The inverse problem of parameter estimation can be interpreted as a problem of training Boltzmann machines. However, the learning process of General Boltzmann machines on classical computers is slow and cumbersome. The data-dependent expectations in maximum likelihood estimation are not easy to compute using classical

	Clique	NAE3SAT ($r = 3$)	NAE3SAT ($r = 2.1$)	3-Regular	3D Lattice w/defects	Native
2000Q	64	90	102	304	512	2030
Advantage	124	242	286	784	2354	5455

Table 1.2: Maximum size of popular graphs that can be embedded into the DWave 2000Q and Advantage system

techniques. Quantum annealers (QA) have provided a promising way forward to tackle this problem of expectation estimation. QA has been recently employed to train BM's with densely connected graph structures. For instance, [28] trained Boltzmann machines to recover missing data in the images of Chemical vapor deposition (CVD) growth for a MoS_2 monolayer. In chapter VI, an efficient algorithm for inverse parameter estimation using a quantum annealer is described.

1.5 Outline of thesis

This thesis aims to formulate mechanics problems using pairwise energy models and solve them (exactly or approximately) in the most efficient way. The computational tractability of pairwise energy problems varies greatly depending on the graph's topology and the nature of the interaction energies. Certain classes of problems are solvable on classical computers using metaheuristic schemes, like GraphCut [7]. One such class is that of the Multiway K-Cut problem (Definition I.9). The interaction energy in this problem for more than 2 labels is of the relaxed-metric type - suitable for approximate methods. In the case of 2 labels, this problem simplifies as a submodular energy form that can be solved exactly. In Chapters 2-4, problems relevant to polycrystalline materials are formulated in this type and approximated using the Graph-cut method.

Chapter 2 develops a procedure to label the elements of a finite element mesh to better represent polycrystalline microstructures and multi-phase materials with smooth boundaries. The formulated problem is a Multiway K-Cut problem on a planar graph.

Chapter 3 introduces a multiscale approach for studying crack propagation in brittle polycrystalline materials. The formulated energy is of binary submodular type on a planar graph. Although this problem can be solved exactly in polynomial time, approximate methods are used for their favorable computational efficiency.

Chapter 4 studies the temporal evolution of microstructure in a thermally simulated environment. The formulated problem is a Multiway K-Cut problem on a non-planar graph.

The next two chapters move away from studying polycrystalline materials and explore this formulation's utility in dealing with general problems like solving differential equations and machine learning. Unlike the previous case, the next problems do not adhere to any of the simpler classes and are tractable only on Noisy Intermediate-Scale Quantum (NISQ)-era hardware, like QA.

Chapter 5 introduces an iterative combinatorial optimization technique for solving differential equations. Finite element approximations are used to construct sparse graph energy models that can be efficiently solved using QA.

Chapter 6 develops a training methodology for General Boltzmann machines - a machine learning architecture. The finite temperature of QA makes it suitable for sampling states from the Boltzmann distribution. This property of QA is utilized to carry out likelihood estimation on general graphs.

Finally, **Chapter 7** concludes the contributions and proposes future directions for this thesis's work.

CHAPTER II

Graph Coloring Method for Conformal Mesh Generation

This chapter provides a novel approach for mesh generation for materials which have distinct spatial components with a smooth boundary between them. Experimental data is used in pixel/voxel format to label elements in a generic finite element mesh of a representative volume element (RVE). The basis of this approach is a novel Potts energy formulation to allow integer optimization on the dual of the Finite Element (FE) mesh. The Potts energy can be decomposed into two terms: the field energy/data cost and the interaction energy/smoothing cost. The field term is used to represent the likelihood of a grain label on an element based on the experimental voxel data. The interaction term encodes a prior on this labeling, in particular, it is used for smoothening the phase boundary. Energy minimization of this system leads to a multi-way cut problem which is solved using Graph-cuts. This methodology allows capturing smooth boundaries in materials with non-equiaxed morphologies. Applications to polycrystalline microstructures and woven composites are presented. The extension to non-equiaxed morphologies is presented using the Riemannian distance measure. This procedure allows re-usability of a FE mesh by adaptively assigning pixel/voxel information to elements while preserving important features like the phase boundary surface length/area.

2.1 Background

Structural metallic materials used in the aerospace industry (eg. Aluminum 2000 and 7000 series, advanced titanium alloys) are composed of aggregates of crystals. With the emergence of the paradigm of integrated computational materials engineering (ICME) [29], multi-scale design/optimization approaches for tailoring engineering properties of materials through controlled microstructure [30, 31] are of great interest. A multiscale approach for the design of turbine blades is presented in Fig 2.1 to illustrate stress variation in the macroscopic and microscopic scale. Such a simulation involves solving for microstructure–dependent properties for macro–scale analysis, which in turn requires meshing at the microstructural level that captures the grain size and shape features [32]. Meshing of 3D microstructures to conserve such grain features is of immense interest as these play an important role in processes such as localization [33] and fracture [34].

Computational tools, like finite element methods, are almost ubiquitously present in the form of both commercial and private software. In the realm of polycrystalline materials, numerical procedures generally require the knowledge of the microstructure which is often provided through experimental methods such as Electron Back Scatter Diffraction (EBSD). EBSD provides the spatial distribution of crystallographic orientations of different grains and can be used to estimate quantities like average misorientation. A complete review of experimental and computational EBSD techniques are provided in [35]. From the computational perspective, these experimental images offer information about the spatial distribution of phases in a pixelated format for 2D or voxelated format for 3D. This data also provides a convenient form of a mesh which consists of uniform quadrilateral/hexahedral elements. An open source package is developed in [36] for generation of voxelated representations of serial sectioned EBSD maps. However, voxelation leads to a stepped, block–like representation of otherwise smooth boundaries. This affects the quality of computational simulation through the introduction of spurious stresses [37]. There are other drawbacks in using structured meshes of quadrilateral/hexahedral elements for 2D/3D elements as well, for example, the boundary length/area does not converge to the correct value with refinement.

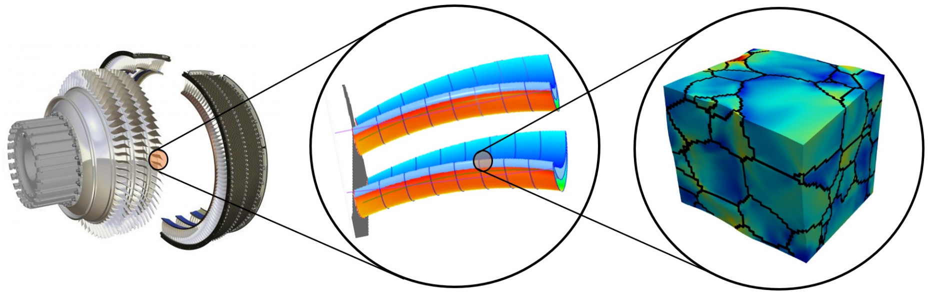


Figure 2.1: Multiscale design of turbine blades for aircraft engines.

Unstructured grids where element surfaces conform with the phase boundaries offer a better approximation of the spatial morphology. An unstructured meshing scheme which is widely employed, generates objects in the lowest dimension and then re-uses these objects as seeds for generating higher order objects. More precisely, nodes (0-dimensional objects) are generated and are connected using lines (1-dimensional objects) where more points are sampled. Faces (2-dimensional objects) are created by joining these lines and are meshed using triangles with edge points as seeds. Then, these faces are connected to create a grain structure (3-dimensional objects) and are meshed with tetrahedrons using face triangles as seeds. This procedure allows for a smoother phase boundary and improves the estimation of quantities like Phase Boundary Energy. This procedure is also implemented for mesh generation in polycrystalline materials in an open source software package developed in [38]. However, this procedure can be computationally expensive as it requires re-meshing for every new experimental image.

A new technique for optimal partitioning of an arbitrary unstructured mesh into different phases is proposed in this chapter. In practice, partitioning the mesh means assigning an index to each element where elements with similar indexes form a single phase. Thus, any experimental image can be superimposed on a generic unstructured FE mesh by effectively partitioning the mesh based on voxelated data. Two conditions are expected to be satisfied. Firstly, the phase index assignment to the finite element mesh should be as close as possible to the experimental voxel data. Secondly, the phase index assignments should be such that the boundaries are as smooth as possible. Seminal contribution in the development of the Graph partitioning algorithm are made in [39], [40] and [41]. Spectral methods (for instance, see [42]) are often used. Recent advances in this field have been reviewed in [43]. Improved results are achieved with global methods, such as multilevel approaches which operate in stages with refinement. METIS and KaHIP are good open source graph partitioners which utilize multi-level methods and are developed in [44] and [45], respectively. Since GP is often studied as a combinatorial optimization problem, the methods introduced in Chapter I are also widely employed.

The overall objective of this chapter is to provide an effective way of labeling the elements of a mesh into the representative phases such that the spatial distribution of the phase is comparable to the experimental data while maintaining the smoothness in the boundaries. This is done by formulating a graph coloring problem on a graph embedding on the dual of an FE mesh. A Potts energy is formulated where the minimization of data term preserves the spatial phase data and the smooth term penalizes the rough behavior of phase boundaries. The resultant energy is solved using Graph-Cut algorithm. The above steps formalize the theory of embedding pixelated/voxelated data into an user-generated unstructured grid. The generalization to non-equiaxed morphologies of component phases is presented using a Riemannian metric. Application of the method to fiber composites is also studied.

2.2 Problem Formulation

The formulation for both 2D and 3D mesh generation is presented in this section.

Definition II.1 (Finite Element (FE) Mesh)**.** Given a closed bounded polyhedral domain $\Omega \in \mathbb{R}^3$, one can associate a finite partition $\mathcal{P}_h$ of Ω in polyhedrons P_i(indexed with i) such that

$$\Omega = \bigcup_{P_i \in \mathcal{P}_h} P_i$$

such that:

1. $int(P_i) \neq \phi \; \forall P_i \in \mathcal{P}_h$

2. $int(P_i) \cap int(P_j) = \phi \; \forall P_i, P_j \in \mathcal{P}_h$ s.t. $P_i \neq P_j$

3. if $F = P_i \cap P_j \neq \phi$ for some $P_i, P_j \in \mathcal{P}_h$ and $P_i \neq P_j$, then F is either a whole face (polygon), a whole edge (line segment) or a node (point) of the polygons P_i and P_j

4. $h = \max\limits_{P_i \in \mathcal{P}} h_i$ where h_i denotes the longest euclidean distance between two points of P_i.

where $int(.)$ denotes the interior. The partition P_h is called the *mesh* of Ω
Remarks: In the case of 2D polygonal domain, $\Omega \in \mathbb{R}^2$, this definition can be extended by considering the partitions of polygons, P_i and modifying condition (3) to consider only edges and corners.

The mesh for multiphase materials is defined by assigning a phase to each element of the partition. This assignment or labeling, as it is referred to in the rest of the chapter, is done using an undirected graph, $\mathcal{G}$. The graph, $\mathcal{G}$, is embedded in the mesh, $\mathcal{P}_h$, by placing a single vertex, v_i, in the interior of each P_i. Without loss of generality, the location of v_i can be determined as the centroid of the polygon P_i. It ensures that $v_i \in int(P_i)$ when P_i is convex (generally true for all elements of FE meshes). A connection is introduced between two vertices v_i and v_j if and only if P_i and P_j share a common face. In case of 2D meshes, this connection is introduced if and only if P_i and P_j share a common edge. Furthermore, each connection is endowed with a weight determined by the area and the unit normal of the shared face. Consequently in 2D meshes, the weight is determined by the length and the unit normal of the shared edge. In particular, if $\overrightarrow{n}$ and Δs are the unit normal and the area (length) of the face (edge), respectively, then weight w_i of the connection e_i is given as $w_i = \gamma(\overrightarrow{n})\Delta s$. The function γ is restricted to be strictly positive and symmetric i.e. $\gamma > 0$ and $\gamma(\overrightarrow{n}) = \gamma(-\overrightarrow{n})$. This construction is illustrated in Fig 2.2 where a connection between 2 representative elements **A** and **B** is introduced with a weight determined by the unit normal $\overrightarrow{n}$ and length of the edge Δs.

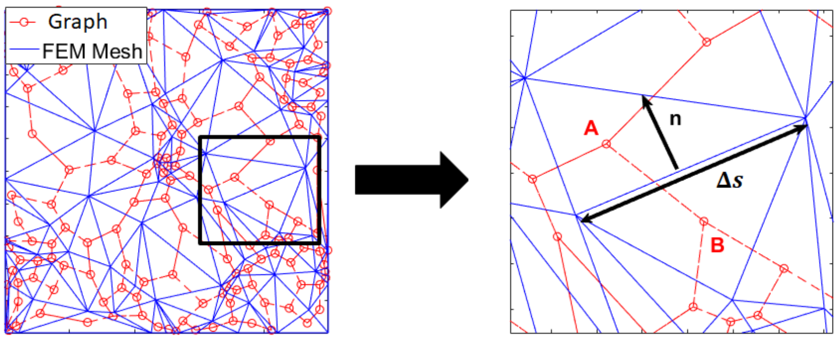


Figure 2.2: Construction of the graph G=(V,E) from a finite element mesh.

In this work, labels ($\mathcal{L}$) are identified as different phases. For a particular choice of labels , and the corresponding multiway cut E' (see Definition I.9), a specific form of Potts energy (see Definition I.2) is chosen:

$$E(\boldsymbol{S}) = \sum_{i=1}^{N_V} U_i(s_i) + \sum_{k=1}^{N_C} w_k \left(1 - \delta_{s_{\pi(k,1)}, s_{\pi(k,2)}}\right) \tag{2.1}$$

where $w_k \in \mathbb{R}^+$ and δ is the Kronecker delta function. The first term in this form is often referred to as the data term while the second term is referred to as the smooth term. The image information is encoded in the data term using a phenomenological form. Let I be a function which evaluates the label (phase information) from the image data (pixel/voxel). For instance, $I(v_i) = r$ means v_i belongs to the pixel region with phase index r. Additionally, let the function $Vol(P_i)$ evaluate the area/volume of the i^{th} element in case of 2D/3D meshes with M representing the mode of the distribution. As an ansatz, U_i assumes the form of Eq (2.2) where $\alpha \in \mathbb{R}^+$ is a control parameter. It is worth observing that the minimization of this term leads to preservation of phase structure of the input data.

$$U_i(l) = \begin{cases} 0 & \text{if} \quad I(v_i) = l \\ \alpha(1 - e^{-\frac{Vol(P_i)}{M}}) & \text{if} \quad I(v_i) \neq l \end{cases} \tag{2.2}$$

The smooth term, as the name suggests, penalizes any roughness of phase boundaries. In this work, it is specialized to the form presented in Eq (2.3) where $\overrightarrow{n_k}$ and Δs_k represent the unit normal and length of the edge shared between the mesh elements $\pi(k,1)$ and $\pi(k,2)$. The action of the second term can be better understood by taking $\gamma(\overrightarrow{n}) = 1$ for all $\overrightarrow{n}$. In this special case, the second term reduces to the total length of the boundary between each component of the k-way partition, i.e. length of the phase boundary. Minimization of this term leads to the minimization of the phase boundary length which renders a smoothening effect. The total energy is minimized using the alpha expansion method discussed in [7] by solving the equivalent Multiway K-Cut problem. The data structure and the pseudocode for implementing this method is presented in the Appendix B.1.

$$w_k(f) = \gamma(\overrightarrow{n})\Delta s_k \tag{2.3}$$

Simultaneous minimization of the two energy terms gives rise to Pareto optimal solutions i.e. solutions where both terms cannot be simultaneously decreased any further. This means there exists a labeling such that any change in it either leads to deviation from the input image or introduces roughness in grain boundary. The solution is selected based on the choice of α in Eq (2.2). This phenomenon is illustrated in the next section by means of an example.

2.3 Results and Discussion

The salient features of the algorithm are illustrated using two examples. In the first example, polycrystalline materials are used with the grains representing the different phases of the material. Next, a case study of woven fibers composite is considered where the fiber and the matrix are identified as the two distinct phases. In polycrystalline materials, the smooth boundary is an artifact of microstructure evolution which is governed by surface energy minimization. According to the theory discussed in the previous section, the naive labeling based on the pixel location results in rough grain boundaries. The energy minimization problem is not required to estimate this labeling. In fact, it can be uniquely determined by the relation, $s_i = I(v_i)$, which was treated as the basis for formulating the data cost. This grain boundary is smoothened by minimization of energy form given in Eq (2.1). This process is illustrated in Fig 2.3 for a microstructure image with 200x200 pixels. A triangular mesh with 30,625 nodes and 60,547 elements is, first, labeled based on pixel positions. The boundaries are then smoothened by solving the Multiway K-Cut problem on the embedded graph containing 90,470 connections with the data parameter chosen as $\alpha = 0.002$ and the metric function chosen as $\gamma(\overrightarrow{n}) = 1$. The same procedure can be carried out for low-resolution images where pixel-based labeling may result in a very rough grain boundary. Grain identification for mesh elements using pixel-based and Multiway K-Cut based is presented for a low-resolution image of a pixel size of 50×50 in Fig2.4 It is observed that the smoothening effect is less prominent (in comparison to the high-resolution image) due to the pixelated nature of the base image.

The dependence of partitioning on the data term is shown in Fig 2.5. The Grain Boundary length ratio (i.e. the non- dimensional length of the grain boundary with respect to the unit cell) is calculated with respect to labelings for different data cost.

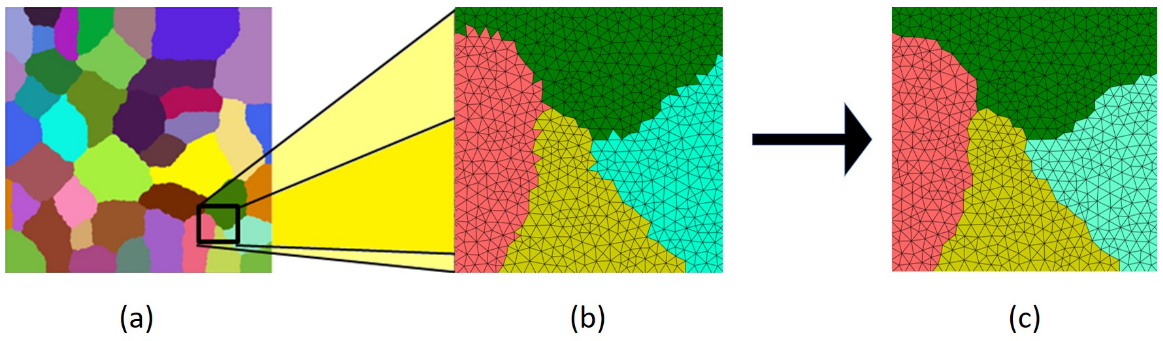


Figure 2.3: An illustration of labeling procedure: (a) Microstructure image (b) Pixel-based labeling (c) Smoothened mesh using Multiway K-Cut based labeling.

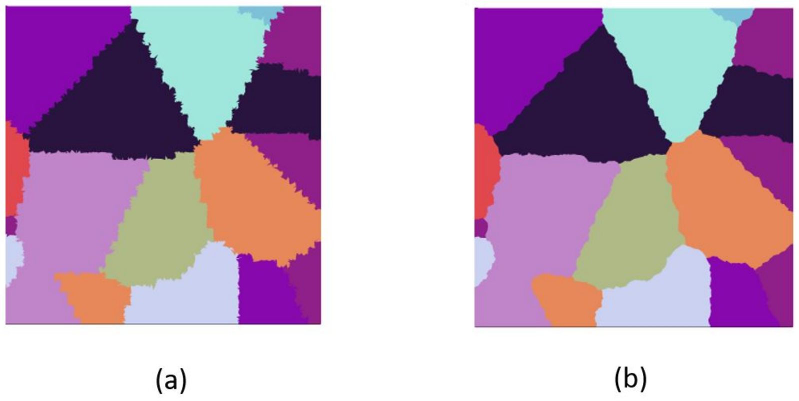


Figure 2.4: Grain identification in a mesh using a 50x50 pixel image: (a) Pixel-based labeling (b) Multiway K-Cut based labeling.

Three regions are observed based on the value of the α defined in Eq (2.2). The first region corresponds to the case when higher weight is given to the minimization of length (low value of α). In this case, the addition of data cost is preferred over the addition of surface energy. On the contrary, labeling is governed by data cost in the third region. Therefore, this region is plagued with rough boundaries and an overall higher estimate of grain boundary length/area. Practical meshes with smooth boundaries and a minimal loss of grain structure are offered in the second region.

A comparison of the performance of this mesh with that of a uniform mesh with pixel-based labeling and an exact mesh (conformal mesh) is presented. A bicrystal with an incident angle of 60° between isotropic components is studied in uniaxial loading. For simplicity, the finite element simulations are presented for linear-elastic

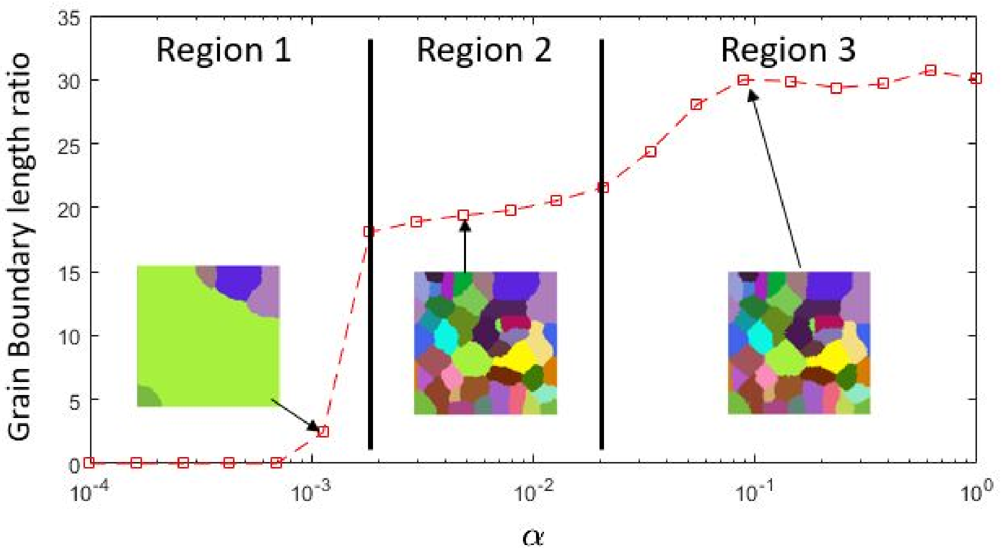


Figure 2.5: Dependence of Graph partitioning on the data term in energy.

deformation assuming plane strain. The distribution of Von-mises stress is presented for the different meshes in Fig 2.6. The grain boundary is exactly mapped by the edges of the conformal mesh. It captures all the localized data as well as the jump in stresses. In comparison, the pixel-based mesh has a diffused solution near the grain boundary and local features like stress concentrations are not well captured. Performance of graph-labeled mesh is very similar to the conformal mesh in terms of capturing discontinuity and localized stresses.

Additive manufacturing techniques, like, selective laser melting (SLM), electron beam melting (EBM) and shaped metal deposition (SMD) often results in microstructure with elongated grains [46, 47, 48]. These structures can be better captured using the characteristic alignment of the grains. Motivated by the elliptical shape of grains, a non-euclidean distance measure can be used to favor or penalize certain directions. A useful class of such distance measures is a Riemannian norm. For this purpose, the function γ can be specialized as Eq (2.4).

$$\gamma(v) = \sqrt{v^T.D.v} \tag{2.4}$$

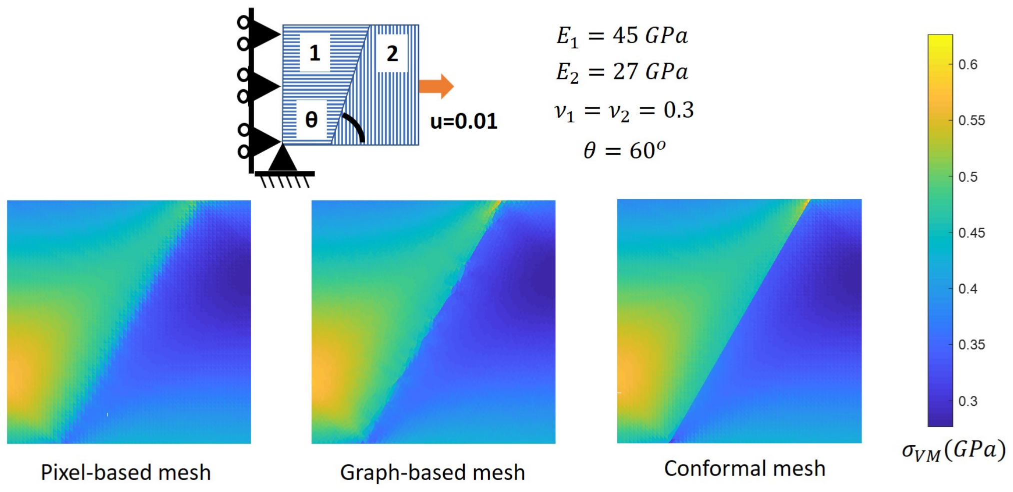


Figure 2.6: Comparative study of Finite element simulation results for a bicrystal in plane strain (linear elastic) using Pixel-based (quad mesh), Graph-based (triangle mesh with smoothening) and Conformal mesh

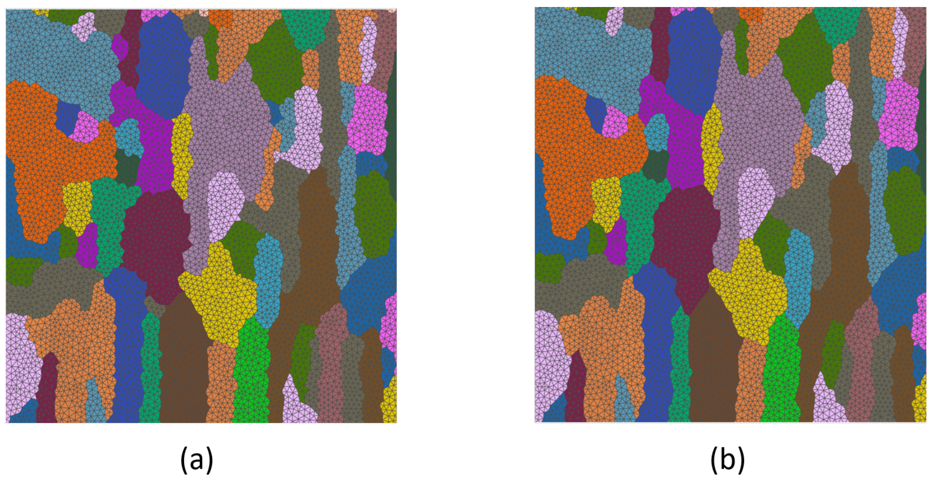


Figure 2.7: Grain identification for elongated polycrystals using (a) Euclidean metric (b) using Riemannian metric

where D is a positive definite matrix. In general, polycrystalline materials with non-equiaxed morphology show smoother grain boundaries in the normal/transverse direction (ND/TD). This directional smoothening is achieved by penalizing the length of a line element in an appropriate direction using the metric presented in Eq (2.4). An example of elongated microstructures is presented in Fig 2.7. It is observed that a metric with $D_{11} = 1$, $D_{22} = 10$ and $D_{12} = D_{21} = 0$, minimizes the kinks in the normal direction. This procedure can be specialized to many other kinds of morphologies

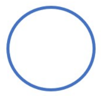
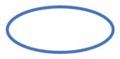
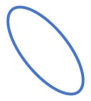
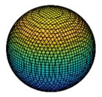
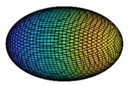
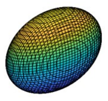
Grain type	D	Iso-surface (Simplified Grain Shape)
Isotropic 2D grain	$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$	
Elongated 2D grain	$\begin{bmatrix} 0.2 & 0 \\ 0 & 1 \end{bmatrix}$	
Oriented 2D grain	$\begin{bmatrix} 0.6 & 0.4 \\ 0.4 & 0.6 \end{bmatrix}$	
Isotropic 3D grain	$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	
Elongated 3D grain	$\begin{bmatrix} 0.3 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$	
Pellet-like 3D grain	$\begin{bmatrix} 0.2 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0.3 \end{bmatrix}$	

Figure 2.8: Illustration of various morphologies based on the metric tensor, **D**

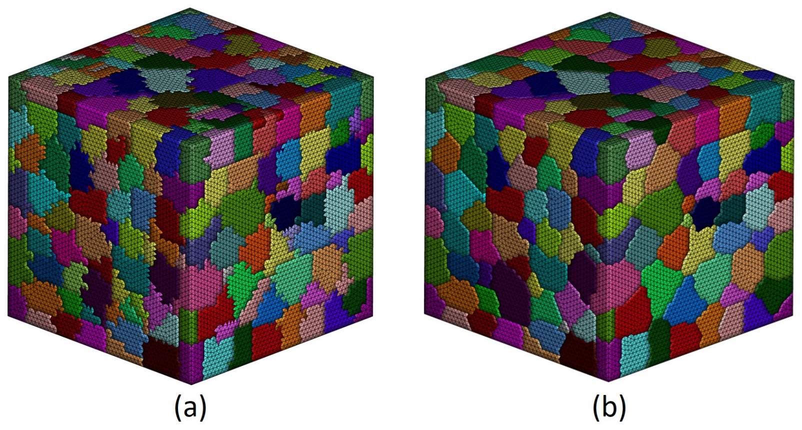


Figure 2.9: Grain identification in 3D meshes using a 64x64x64 voxel base image: (a) Vertex position based labeling (b) Multiway K-Cut based labeling

using the matrix, D. The iso-surfaces for the distance function, $\gamma(\overrightarrow{n})$ for different choices of matrix D are presented as an illustration in Fig 2.8. Based on the theory of Wulff construction, the iso-surface represents the grain shape ([49]).

As suggested in section 2.2, this procedure can be easily extended to 3D meshes. For the purpose of illustration, a $64 \times 64 \times 64$ voxelated image of a microstructure is superimposed on an FE mesh with tetrahedral elements. The sample solution is presented

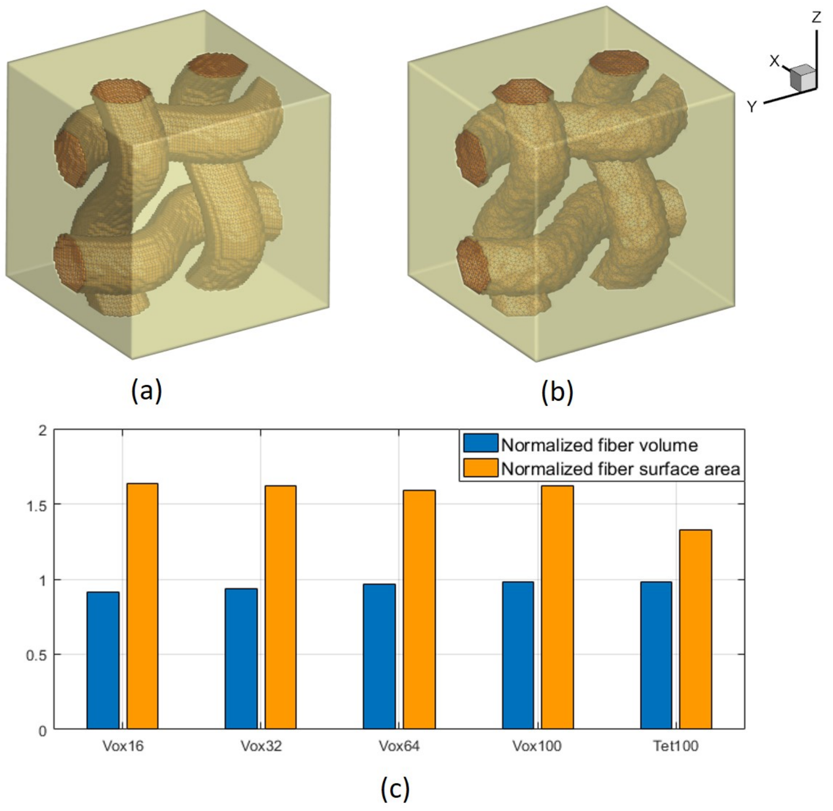


Figure 2.10: Mesh generation for woven fiber composite (a) Voxelated mesh with $\mathbf{10^2} \times \mathbf{10^2} \times \mathbf{10^2}$ elements (b) Tetrahedral mesh with multiway K-Cut based labeling (c) Comparison of fiber volume and surface area of various voxelated meshes and a tetrahedral (smoothened) mesh

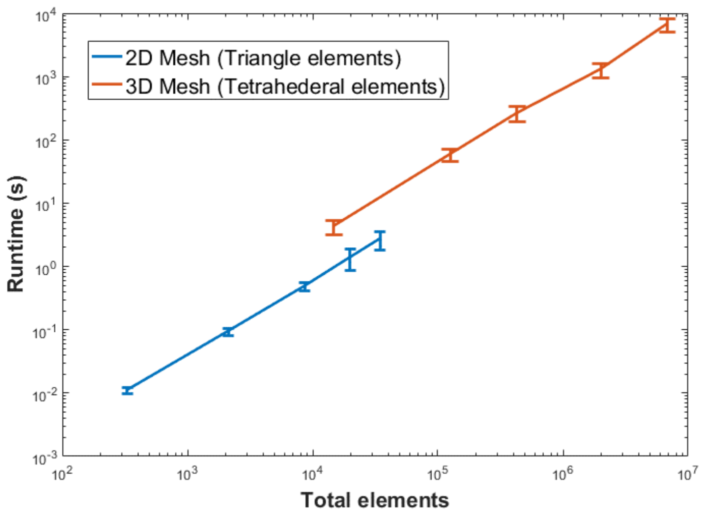


Figure 2.11: Time complexity analysis of the mesh generation procedure.

in Fig 2.9 for a mesh with approximately 3×10^5 elements. This smoothening procedure is also illustrated for a woven fiber composite system. A voxelated mesh and a tetrahedral (smooth) mesh with approximately 10^6 nodes are presented in Fig 2.10. A comparative study of convergence in volume and surface area is also presented in Fig 2.10(c). Different voxelated meshes are used with 'Vox$\langle n \rangle$' representing a mesh with $(n+1)^3$ nodes. The fiber composite with Vox100 mesh is shown in Fig 2.10(a) with the tetrahedral mesh (Tet100) in Fig 2.10(b). The normalization is done with respect to the analytical values of respective quantities. It is observed that the estimate of fiber volume is almost consistent in all the meshes but the estimate of fiber surface area is improved in the tetrahedral mesh.
The time complexity of the algorithm is experimentally studied for 2D and 3D test cases. Ten instances of meshes with a similar number of elements are labeled using graph partitioning. The total average time for graph partitioning using alpha expansion is calculated for the 2D triangle and 3D tetrahedral elements. An almost linear trend is observed in Fig 2.11 (log–log scale is shown but note that the simulation time scales linearly with number of elements). This is faster in comparison to the experimental results presented in [50]. Faster heuristics are seen for the 2D mesh in comparison to 3D for the same number of elements. This is due to the fact that the embedded graph in the 2D case has a planar topology and Graphcut algorithm is known to be more efficient in these cases.

2.4 Conclusion

Based on the ICME paradigm, there has been increasing interest in the use of microstructure modeling tools for optimization of aerospace materials manufacturing processes. Experimental evaluation of the spatial distribution of multiphase materials provides images in a voxelated/pixelated format. However, such a format does not accurately capture features like grain surface area and curvature that are important for modeling fracture. In addition, finite element simulation on voxelated microstructures may lead to the prediction of spurious stresses due to the stepped, block–like representation of otherwise smooth boundaries. In this chapter, a theory is developed for conversion of as–measured voxelated microstructure to an unstructured grid with a smoother representation of boundaries. This problem is solved using Graph Partitioning (GP) theory discussed in Chapter I. This procedure smoothens the grain boundaries using a Potts energy model while optimally preserving the grain label information of the initial experimental data. The trade-off between smoothening and data preservation is controlled using a user–controlled parameter. The performance is tested for both equiaxed and non-equiaxed morphologies using Riemannian metric measures. It is observed that smoother grain boundaries in the normal/transverse direction (ND/TD) can be realized using this approach. The methodology has an efficient run–time and is easily extendable to 3D structures like woven fiber composites and 3D polycrystalline material as demonstrated in this work.

CHAPTER III

Graph Coloring Method for Modeling Microstructural Fracture

A multiscale graph theory-based approach is introduced here to predict the microscale crack path in polycrystalline materials. The crack path is represented as the boundary of the partition of a geometric graph. The partitioning is carried out by optimizing an Ising-type hamiltonian. The hamiltonian parameters are chosen such that each partition cost is the same as the energy of the corresponding crack. The interplay of the loading conditions on the specimen and the microstructure of the material near the crack tip determines the crack growth angle in polycrystalline materials. Two different length scales of macro and micro are incorporated for the crack path by defining the crack total energy as the summation of macroscopic energy release and microscopic surface energy. The former term represents the macroscopically favorable crack growth angle by directing the crack to propagate from the crack tip along the direction of the maximum energy release. The latter term guides the microscopic crack path along the macroscopically preferred direction. At this scale, the crack path naturally accommodates both the intergranular and transgranular fractures. In the case of intergranular fracture, the crack propagates along the grain boundaries, while in the case of transgranular fracture, the crack propagates along the crystallographic cleavage planes. The mixed-mode fracture in a thin foil specimen is studied, and the effect of the dihedral angle of the 2D crack is included in defining the effective surface energy. The model is validated using the analytical results for mixed-mode fracture in an isotropic medium and mode-I fracture in a medium with a preferred crack direction. The proposed method can be used to design materials microstructure with optimal fracture resistance.

3.1 Background

Traditionally, the two main types of fractures considered at the microscale are brittle and ductile. In brittle failure, the metal fractures with little to no plastic deformation, generating surfaces with bright cleaved crystalline facets. The crystallographic planes of the microstructure play an essential role in the mechanism of fracture. For instance, at low temperatures, BCC-Iron (Body centered cubic) cleaves on the {001} family of planes and HCP-Zinc (Hexagonal close packed) cleaves on the basal plane, while FCC (Face-centered cubic) materials are usually immune to such failure. In contrast, fracture in ductile failure is preceded by significant plastic deformation, which redistributes the stress concentration ahead of the crack tip, resulting in different fracture mechanisms. The same kind of material can undergo ductile or brittle depending on the external conditions. One of the earliest pioneering studies of transition from brittle to ductile was carried out by Kelly, Tyson, and Cottrell [51]. They proposed that the ratio of the largest tensile stress and the largest shear stress close to the crack tip governs the crack behavior. A brittle fracture occurs when this ratio exceeds the ratio of ideal cleavage stress and the ideal shear stress.

There are two popular approaches in the literature for the numerical simulation of fracture. The first one is based on cohesive zone models ([52, 53, 54, 55]). These methods have proven to be extremely useful in predicting both the nucleation and crack propagation; however, they are plagued by numerical instabilities. These problems have been mitigated, to a certain degree, using either more stable but computationally extensive load path-following procedures (e.g., Riks method and arc-length method) or by introducing artificial viscosity to the cohesive zone law [56] leading to inaccuracies in load prediction. The second approach is based on Phase-field methods, where cracks are represented using diffuse damage parameters. The evolution of these damage parameters can be modeled using a Ginzburg-Landau type potential ([57, 58]) or using gradient damage theory [59]. There is also the issue of mesh dependence in numerical crack path prediction. It has been shown, for instance, in [60], one of the major reasons for mesh dependence is the stress concentration at the tip. Consequently, mesh-dependent objects like element size near the crack tip can influence the direction of crack growth. These problems are usually addressed using regularization approaches to limit the sensitivity of damage evolution to the stress concentration. Most phase-field studies (for e.g. [59, 61]) use regularization by considering a diffused crack. Moreover, many FEM techniques now allow generating sophisticated crack paths like the node enrichment FEM (e.g. X-FEM)[62] and elemental enrichment FEM (e.g. E-FEM) [63, 64, 65, 66].

One of the major drawbacks of these approaches is that the solution may not lead to a global minimum of energy due to the problem's non-convex nature. This aspect of the global and local solutions can be better understood in the variational model of quasistatic crack evolution proposed by Francfort and Marigo[67]. Consider the following energy of a body with a given crack, Γ, and external load, U:

$$E(\Gamma, U) = E_d(\Gamma, U) + \int_{\Gamma} 2\gamma ds$$

where E_d is the bulk energy of the body, and γ is the surface energy density of the cracked surface. The crack path is estimated using an evolution law where the load is monotonically increased, and the crack progresses in the direction of a maximum decrease in energy. However, this evolution does not need to lead to a global minimizer of energy. In fact, it is mentioned in [67] that these evolution laws are dependent on the load history and have meaningful continuum limits only in certain conditions. At the same time, the reader is cautioned that this not an endorsement of global energy minimization over local growth law in terms of thermodynamic arguments. The global energy minimization is a postulate which seems crucial in understanding the effect of inhomogeneity of surface energy in polycrystalline materials. As an example, an experimental image for crack propagation in WE43 Mg alloy specimen presented in [2] (see Fig 3.1) revealed a crack path which did not propagate completely in the basal direction. Transgranular fracture path in Grain E shows an unusual bending before hitting the grain boundary. A possible explanation is that the intergranular fracture between the grains G and H lowers the overall energy of the crack and therefore influences the crack path in the transgranular region of grain E and F. In comparison, the numerical study of grain boundary effect on the crack path presented in [61] shows this upstream influence at a much smaller length scale. Such results motivate the search for global minimizers of total energy.

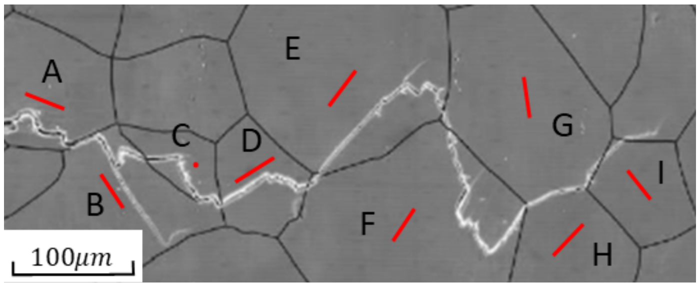


Figure 3.1: Experimental image of a fatigue crack in WE43 Mg alloy (HCP) adapted from [2]. Transgranular fracture followed the basal trace (red line) in grains A-F. Intergranular fracture occurred between grain pairs (A,B), (G,H) and (G,I).

On the other hand, graph-based approaches are posed as combinatorial optimization problems with finite choices where, in some instances, the global minimum can be approximated reasonably quickly. In regards to fracture mechanics, such approaches have been employed in predicting fracture at the microscale. Sundararaghavan and Srivastava [34] developed MicroFract, a graph theory-based software to predict the microstructural crack path in various materials. They validated their model by simulating the intergranular stress corrosion cracking of Inconel. Srinivasan et al. [68] used machine learning along with graph theory to substitute demanding computations with coarse-scale graphs. Hunter et al. [69] also incorporated machine learning along with graph theory and developed reduced-order models, which have sufficient accuracy and low computational cost, to capture brittle fracture. Recent machine learning techniques for fracture prediction have been reviewed in [70]. Needleman and coworkers [71, 72] have developed a model based on discrete unit events along with graph theory to predict intergranular fracture in polycrystalline metals. They investigated the effect of grain size on the propagation of ductile intergranular fracture. There is no graph-theoretic approach that can simulate microscale cracks in polycrystals, including intergranular and transgranular cracks, to the best of authors' knowledge.

This work aims to introduce a general and modular framework for multiscale modeling of a polycrystalline material using a graph-based technique. The macroscopic energy release and the microscopic surface energy, depending on the loading condition and the crystallography of the material, respectively, are encoded into an Ising-type hamiltonian. The non-local component of the hamiltonian (Smoothing term) is obtained as a metric, and therefore, the energy minimization is carried out using the Alpha-expansion method, which guarantees that the min-cut is within a known factor of the global minimum (Kolmogorov and Zabih [9]; Boykov and Kolmogorov [73]). Consequently, the energy of the predicted crack is very close to the globally optimal crack path. The presented scheme can model the crack propagation in mixed-mode-I-mode-II failure in a reasonable time with sufficient accuracy. Furthermore, the algorithm reduces to a Quadratic Unconstrained Binary Optimization (QUBO), which can be solved using quantum annealing systems, like D-wave. A new scheme is developed to incorporate the dihedral angle of cracks in fracture, which considers the effective surface energy of a uniformly oriented out-of-plane crack. The techniques presented here are specialized for brittle fracture in elastic materials. The key idea of this work to represent the fracture energy using the Ising model can be extended to ductile materials, but no such attempt is made in this chapter.

The chapter is organized as follows: Section 3.2 develops the formulation for the total energy of the crack in a multiscale setting and relates it to the Ising hamiltonian of the graph. A procedure to estimate the 2D surface energy of 3D crystals is also presented in this section for studying fracture in thin-film specimens. In section 3.3, the algorithmic details are described, and a discussion on the computational complexity of the method is presented. In section 3.4, the procedure is applied to capture crack paths in two cases : (i) homogeneous media with an anisotropic surface energy and (ii) polycrystalline material. The results for crack growth in homogeneous media are compared against the analytical result. In the case of the polycrystalline material, a qualitative verification of results is presented. The effect of the grain boundary on the crack path is studied, and it is shown that at high loading conditions, the predicted crack path resembles the dynamic fracture path. Section 3.5 provides a summary of the chapter.

3.2 Mathematical Framework

Among the early energy-based methodologies for studying fracture in linear-elastic isotropic materials, the most noted ones are the S-Criterion (Strain Energy Criterion) [74, 75] and the ME-Criterion (Maximum energy release rate criterion) [76]. It was first shown in [77] and later proved in [76] that the ME-criterion is the same as Maximum-Stress Theory [78] i.e., the crack propagates in the direction of maximum circumferential stress. For this reason, the ME-criterion is often favored over the S-Criterion for crack angle prediction in mixed-mode failure. The basic hypothesis for ME-Criterion is that:

1. The crack propagates from the crack tip in a straight line along which the maximum energy is released.

2. The crack propagates when the energy release rate is greater than a critical value.

Consider a small kink from the original main crack tip at an angle, α with the horizontal axis. The application of the ME-Criterion to estimate of the propagation angle can be formalized as:

$$\max_{\alpha} G(\alpha) \geq G_c \tag{3.1}$$

where $G(\alpha)$ denotes the energy release rate for a kink angle, α, and G_c denotes the critical energy release rate. In the case of an ideal brittle fracture, $G_c = 2\gamma$, where γ is the surface energy density of the fractured surface. Now, consider a circular region around the main crack tip determined by the radius, $\overline{r}$, of the branched crack and a microscopic crack path, Γ, parameterized as $\Gamma(s) \equiv (x(s), y(s))$. In this region, the ME-Criterion can be equivalently written in terms of the total energy released (instead of the energy release rate) as follows:

$$\min_{\Gamma} \int_{\Gamma} (2\gamma - G)\, ds \leq 0$$

The above equation was derived by using the fact that G_c, or equivalently, γ is independent of α, which is true for isotropic materials. However, at the microscale, γ depends on the location and the local orientation of the cracked surface. Consequently, the minimizer of this functional is not necessarily a straight line, as postulated in the first hypothesis of the original ME-criterion. Moreover, the second hypothesis is also violated as the equivalent G_c changes with the kink angle. In favor of a multiscale approach, the energy release rate is assumed to be governed by the macroscopic feature of the crack, i.e., kink angle, defined as the average angle of the microscopic crack.

The assumptions considered in the new hypothesis for multiscale crack propagation are stated as:

1. Crack, Γ propagates from the crack tip in a possibly zig-zag direction towards the circular region's circumference so that the path minimizes the total energy of the system, $\varepsilon(\Gamma)$:
$$\varepsilon(\Gamma) = \int\limits_{\Gamma} 2\gamma ds - G(\alpha_\Gamma)$$
2. Crack propagation is initiated when the total energy is negative.

The energy release rate for mixed mode crack in isotropic material is estimated in [76] as follows:

$$G(\alpha) = \frac{\kappa + 1}{8\mu} \cos^2\frac{\alpha}{2}\left(\cos^2\frac{\alpha}{2}K_{II}^2 + \left(\cos\frac{\alpha}{2}K_I - 2\sin\frac{\alpha}{2}K_{II}\right)^2\right) \tag{3.2}$$

where κ is a material constant, which can be obtained as $\kappa = 3 - 4\nu$ for the plane strain condition and $\kappa = (3-\nu)/(1+\nu)$ for the plane stress condition. The variables, μ and ν are the shear modulus and Poisson's ratio for the homogenized material, respectively. Note that the energy release rate for a general anisotropic material ([79]) can be substituted here without influencing the formulation.

3.2.1 Graph representation of Microscopic crack

A random finite element mesh is considered (see Definition II.1) where each element is a polygon represented by P_i with $i \in \{1, ..., n\}$. The goal is to label each element as $+1$ or -1 based on a cost minimization criterion. Therefore, a particular partition of a mesh is given by an array of labels, $L = [l_1, ..., l_n]$ with $l_i \in \{+1, -1\}$. The crack is represented as the element edge with different labels on either side. To define the energy of this partition, an undirected graph (see Definition I.1) is embedded in this mesh. Each element, P_i, is treated as a vertex, v_i, of the graph, $\mathcal{G}$. A connection, e, exists between two vertices if their respective elements share an edge[1]. This cost is determined as the Ising hamiltonian, H_I, on the graph, $\mathcal{G}$, defined as:

$$H_I(L) = \sum_{v_i \in V} g(c_i, l_i)|P_i| + \sum_{(v_i,v_j)\in E} 2\gamma(\overrightarrow{n})\Delta s_{ij}(1 - \delta_{l_i - l_j}) \tag{3.3}$$

where $g(.,+1)$ and $g(.,-1)$ are continuous functions defined over the domain. c_i and $|P_i|$ are the centroid and the area of the element, P_i, respectively. Δs_{ij} and $\overrightarrow{n}$ are the length and a unit normal to the interface between the elements, P_i and P_j, respectively. δ_x is the Kronecker-delta function, i.e. it has value 1 at $x = 0$ and 0 everywhere else. The function γ is strictly positive and represents the surface energy density of the crystal. It can be immediately observed that the second term does not contribute towards the cost if the labels of neighboring pair are the same.

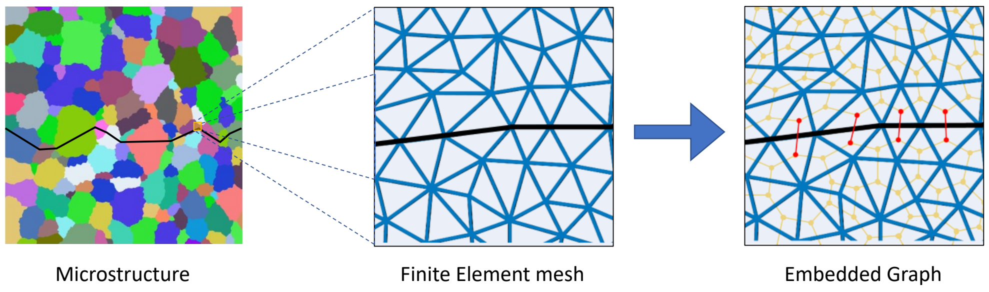


Figure 3.2: A portion of the FE mesh (in blue) with the dual graph, $\mathcal{G}$ (in yellow) embedded in the mesh. Only the connections across the crack (in red) contribute to the cost. The energy of the connection between elements A and B is determined by the length and the normal of the interface edge.

In the next section, the procedure to encode the total energy as an Ising hamiltonian is presented. The first term in Eq (3.3) is used to encode the macroscopic energy release, while the second represents the microscopic surface energy of the crack.

[1]In the subject of Graph theory, this construction is referred to as the weak dual of a primal graph, where the vertices and connections of the primal graph are defined identified as the nodes and the edges of the mesh respectively.

3.2.2 Graph based modeling of Energy release

A semicircular region of radius, $\overline{r}$ can be defined as an area of interest where the macroscopic crack grows from the center to the circumference in a straight line. The net energy release increases linearly with the radius, $\overline{r}$. Morevover, it has functional dependence on the kink angle. To encode this behavior of the energy release, following functional form of g is considered as an ansatz:

$$g(c_i, l_i) = (2-k)\overline{r}^{k-1}\frac{\overline{g}(\theta_i, l_i)}{r_i^k} \tag{3.4}$$

where, θ_i and r_i define the angular and radial coordinates of the point c_i and $k < 2$ for admissibility. Consider a radial crack at an angle α. To understand this choice of function, g, observe that in the continuum limit of the mesh ($\mathcal{P}_h$ with $h \to 0$), the first term of Eq(3.3) can be expanded as (see PropositionA.1):

$$\sum_{v_i \in V} g(c_i, l_i)|P_i| = \overline{r}\int_{\alpha}^{\pi/2} \overline{g}(\theta, +1)d\theta + \overline{r}\int_{-\pi/2}^{\alpha} \overline{g}(\theta, -1)d\theta \tag{3.5}$$

where $\{+1\}$ is the label of upper region and $\{-1\}$ is the label for lower region. Moreover, the above equation represents the negative of the energy release in the continuum limit, i.e.

$$\overline{r}\int_{\alpha}^{\pi/2} \overline{g}(\theta, +1)d\theta + \overline{r}\int_{-\pi/2}^{\alpha} \overline{g}(\theta, -1)d\theta = -\overline{r}G(\alpha) \tag{3.6}$$

Therefore, the initial ansatz for g satisfies the linear dependence on $\overline{r}$ and angular dependence on the crack angle, α. This integral equation form is satisfied by the following form of $g(., l_i)$ (see PropositionA.2):

$$\overline{g}(\alpha, +1) = G'(\alpha)H(G'(\alpha)) + c$$

$$\overline{g}(\alpha, -1) = -G'(\alpha)H(-G'(\alpha)) + c \tag{3.7}$$

where H is the heaviside function and c is given by:

$$c = -\frac{1}{\pi}\left(G\left(-\frac{\pi}{2}\right) + \int_{-\pi/2}^{\pi/2} G'(\theta)H(G'(\theta))d\theta\right)$$

The derivative G' can be estimated from Eq(3.2) as:

$$\begin{aligned} G'(\alpha) = -\frac{\kappa+1}{8\mu}\cos\frac{\alpha}{2}\Bigg(\left(\sin\frac{\alpha}{2} + \sin\frac{3\alpha}{2}\right)\frac{K_I^2}{2} \\ + \left(5\sin\frac{\alpha}{2} - 3\sin\frac{3\alpha}{2}\right)\frac{K_{II}^2}{2} - 2\cos\frac{3\alpha}{2}K_I K_{II}\Bigg) \end{aligned} \tag{3.8}$$

Moreover, a phenomenological choice of $k = 1$ is made in this chapter. To understand the reason behind this, consider a rough crack represented using its angular position (α) as a function of the radial position. The following relation holds true in the continuum limit (see PropositionA.3):

$$\sum_{P_i \in \mathcal{P}_h} g(c_i, l_i)|P_i| = (k-2)\overline{r}\int_0^1 \frac{1}{s^{k-1}}G(\alpha(\overline{r}s))ds \tag{3.9}$$

Therefore, the parameter k can be used to give different weightage to energy release rate at different radial distances. It can be easily observed that for the given choice of k, the excess energy release rate is weighed uniformly.

3.2.3 Modeling surface energy

The surface energy density, γ, in Eq (3.3) varies spatially and is dependent on the angle of the cracked surface. The treatment of edges between elements of the same grain (transgranular) is done differently from the ones with a distinct grain (intergranular). Here, the energies related to both the crack growth pattern of transgranular and intergranular types are elaborated.

Transgranular surface energy: Transgranular cracks usually prefer propagation along crystallographic planes, typically those with low indices. This effect is due to the relatively smaller energy density for these planes. Due to the symmetry of crystals, the surface energy density also obeys these rules, i.e. for the function γ^{tg} to represent the surface energy density of a crystal system, it must satisfy: $\gamma^{tg}(Q \star \overrightarrow{n}) = \gamma^{tg}(\overrightarrow{n})$, where Q is an element of the group, $\mathscr{G}$, consisting of appropriate symmetry operations, and $Q \star \overrightarrow{n}$ denotes the action of Q on $\overrightarrow{n}$. These energies are often available in the literature for specific crystallographic planes. The symmetry conditions can then be encoded using appropriate interpolation. This practice is fairly common in smeared crack approaches; for instance, surface energy density interpolation for crystals with cubic symmetry is introduced in [80].
As an example, a model for transgranular surface energy density of a crystal with cubic symmetry is chosen. In this case, $\mathscr{G}$ is the group generated by the operations:

$$\{R_x(\pi/2), R_y(\pi/2), R_z(\pi/2), -I\}$$

where R_x, R_y and R_z are rotations about the chosen coordinate system and $-I$ is the inversion operator. The surface energy density of the crack for a cubic material is modelled as:

$$\gamma^{tg}(\theta, \phi) = \gamma_0 \left(1 + 3\delta \left(\frac{\cos^2\theta \sin^2 2\phi}{4} + \sin^2\theta\right)\cos^2\theta\right) \tag{3.10}$$

where θ and ϕ are the elevation and the azimuth of the unit normal to the cleavage plane relative to the crystal frame of reference. The $\{001\}$ family of planes has the minimum energy given by the parameter, γ_0, while the $\{011\}$ family has a saddle with energy $\gamma_0(1 + 3\delta/4)$ and the $\{111\}$ family has a maximum with energy $\gamma_0(1 + \delta)$. An illustration of this model is shown in Fig3.3 where the cubic symmetries are evident.

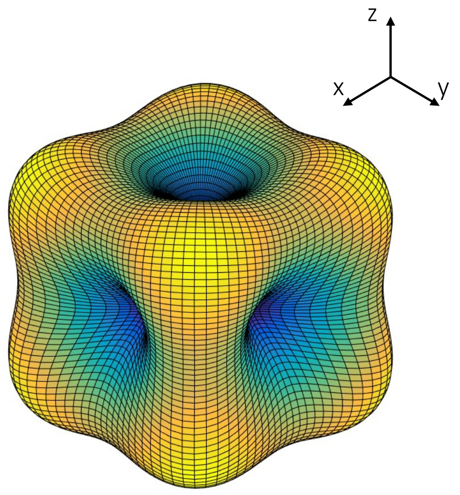


Figure 3.3: An illustration of the 3D surface energy density of a cubic material with $\gamma_0 = 1J/m^2$ and $\delta = 2$ in the direction normal to the cleavage surface.

Intergranular surface energy: An intergranular crack propagates along the grain-boundaries. And the grain-boundary energy, γ_{GB}, determines the surface energy. The effective intergranular surface energy density of the cracked surface, γ^{ig}, is determined as:

$$\gamma^{ig} = \gamma_{\text{surface}} - \frac{1}{2}\gamma_{GB} \tag{3.11}$$

where γ_{surface} is the average energy required to break atomic bonds per unit surface area of individual grain species. As suggested by the above relation, grain boundaries with lower γ_{GB} are more stable as the surface energy is higher along those grain boundaries. These energies are crucial in modeling crack propagation via special grain-boundaries like the coherent twin boundary (CTB), which has the lowest energy.

3.2.3.1 Fracture in thin foils

During crack propagation in these films of crystalline material, unlike isotropic material, the dihedral angle (out of plane angle) cannot be assumed to be zero. Consider a specimen with uniform thickness and a 2D crack that extends in the third dimension with a dihedral angle, ψ_D. Therefore, the surface area generated by a unit 2D crack is

$\sec\psi_D$. The dihedral angle is chosen such that the total surface energy is minimized for all possibles dihedral angles. It is also evident that the dihedral angle depends on the direction of the edge. Without a loss of generality, the microstructure surface is chosen to be on the $x_1 - x_2$ plane. The dihedral angle is defined as the angle between the unit normals, $\overrightarrow{n}_{2D}$ and $\overrightarrow{n}_{3D}$. The general unit normal to the 3D crack can be parameterized with an angle ψ as:

$$\overrightarrow{n}_{3D} = [0, 0, 1]^T \sin\psi + \overrightarrow{n}_{2D} \cos\psi$$

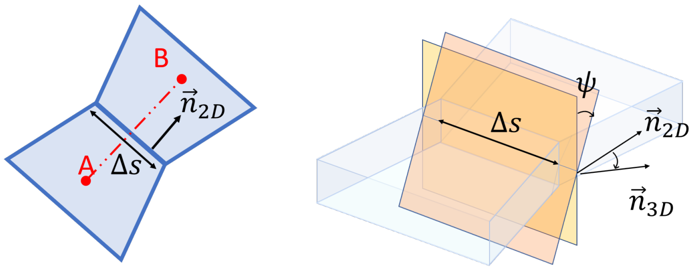


Figure 3.4: An illustration for the dihedral angle of a 2D crack.

And the dihedral angle is estimated as:

$$\psi_D = \text{argmin}_{\psi\in(-90^\circ,90^\circ)}\, \gamma\left(\overrightarrow{n}_{3D}(\psi)\right) \sec\psi \tag{3.12}$$

The effective surface energy for the 2D crack is then calculated as:

$$\widetilde{\gamma} = \gamma(\overrightarrow{n}_{3D}(\psi_D)) \sec\psi_D \tag{3.13}$$

The surface energy density and effective dihedral angle for three different grain orientations are shown in Fig3.5 for a cubic material with the surface energy model defined as Eq(3.10). It can be observed that cracks with lower surface energy density are possible by considering the dihedral angle.

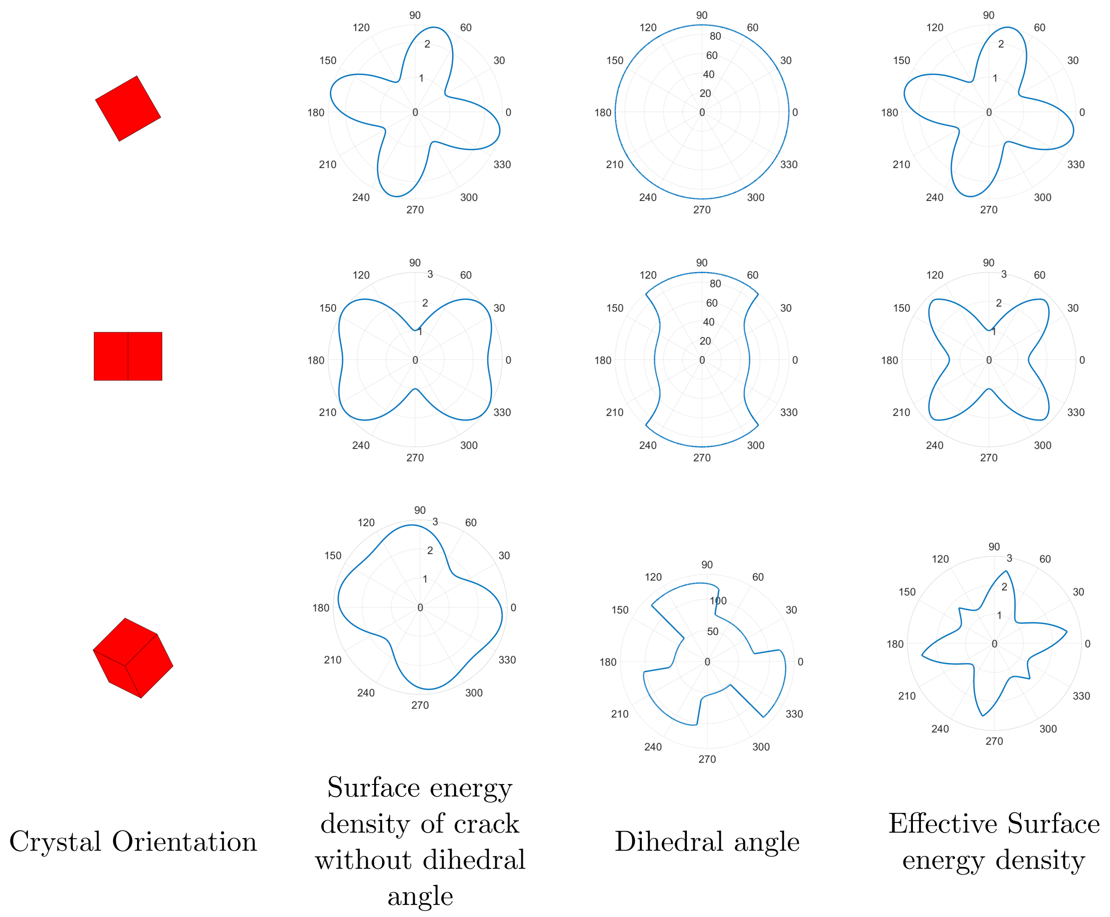


Figure 3.5: 2D surface energy density of a cubic material with $\gamma_0 = 1J/mm^2$ and $\delta = 2$. Each row represents a specific crystal orientation in the specimen frame of reference, and the polar plots are evaluated with respect to cleavage normals. 2D surface energy densities are evaluated for a specimen with unit thickness. The dihedral angle is plotted as $\psi_D + 90°$ for positivity considerations in the polar plot.

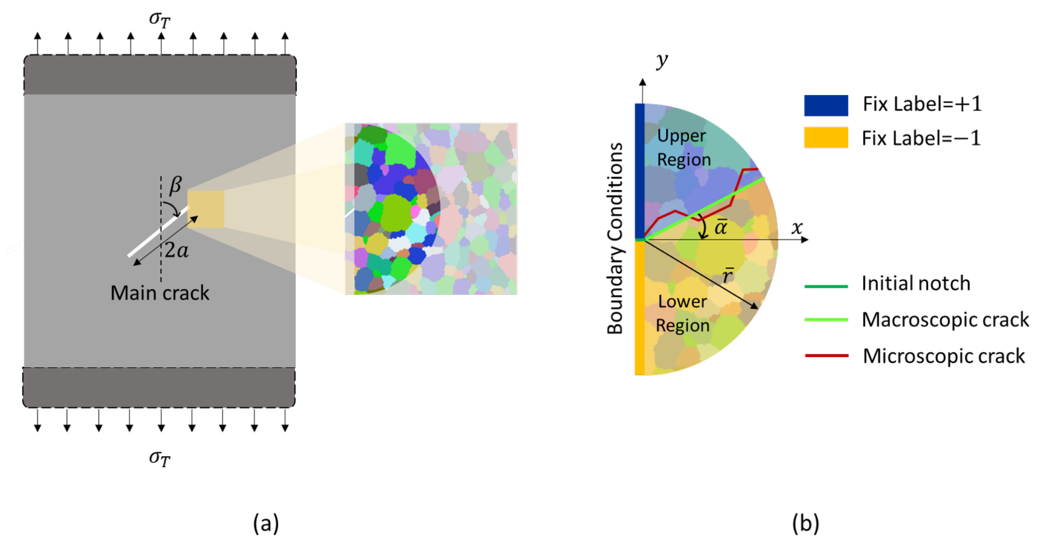


Figure 3.6: (a) A tensile test specimen for mixed mode failure (b) The definition of the macroscopic crack angle from the microscopic crack path.

3.3 Method

3.3.1 Problem Setup

A Tensile test is considered with an inclined main crack to replicate the conditions of Mixed mode failure (shown in Fig 3.6). The main crack has a half-length, a and an inclination angle, β.

The stress intensity factor can be estimated as:

$$\begin{aligned} K_I &= \sigma_T \sqrt{\pi a} \sin^2 \beta \\ K_{II} &= \sigma_T \sqrt{\pi a} \sin \beta \cos \beta \end{aligned} \tag{3.14}$$

where σ_T is the applied tensile stress.

3.3.1.1 Boundary Condition

The minimization of Eq(3.3) leads to two types of solutions: (1) Both labels are present - This means that there is a crack propagating at an effective angle, $\overline{\alpha} \in$

$(-\pi/2, \pi/2)$ (2) Only one label is present. The latter condition should not be confused with a vertical crack because there is no surface energy being generated in this case. Moreover, due to the lack of surface energy, this erroneous solution's total energy can be substantially low in the absence of a strong directional dependence of the macroscopic energy release rate. For this reason, boundary conditions are applied where the labels $+1$ and -1 are imposed on a thin strip on top and bottom of the main crack-tip (as shown in Fig3.6(b)). It is done by artificially increasing the value determined by the term $g(c_i, -1)|P_i|$ in the top strip and $g(c_i, +1)|P_i|$ in the bottom strip.

3.3.1.2 Checking total energy

After enforcing the boundary conditions, the minimization process predicts the crack path as per the Revised ME-Criterion's first hypothesis. However, this procedure does not check whether the total energy is negative; i.e., it violates the second hypothesis. This problem is resolved by checking the total energy after the labeling is done, and the solution is discarded if the total energy is positive.

3.3.2 Computational Procedure

Experimental imaging techniques usually provide a pixelated image with a grain index of each pixel and a table relating grain index to its orientation. This data is superposed on a mesh, which is generated using a Delaunay triangulation of randomly sampled points by determining the grain index of each element using the technique developed in [81]. This method ensures that the grain boundaries in the resulting mesh are smooth. The crack is identified as the boundary between the elements with different labels. This labeling is carried out by minimizing the cost function presented in Eq(3.3) using the Alpha-expansion method. The gco library developed by [7] is used to minimize the cost function. This library requires three costs functions namely, label cost, LC, data cost, DC, and neighbor cost, NC. The label cost is given as:

$$LC(i,j) = 1 - \delta_{(l_i - l_j)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{3.15}$$

The neighbor cost defines the weight of connection between the i^{th} and j^{th} element, which can be defined as follows:

$$NC(i,j) = 2\gamma(\overrightarrow{n})\delta s_{ij} \tag{3.16}$$

The data cost describes the cost of labeling the j^{th} vertex. It is decomposed as, $DC = DC_0 + DC_{BC}$ where DC_0 denotes the data cost for emulating the macroscopic energy release and is given as

$$\begin{aligned} DC_0(1,j) &= g(\theta_j, +1)|P_j|/r_j \\ DC_0(2,j) &= g(\theta_j, -1)|P_j|/r_j \end{aligned} \tag{3.17}$$

The term DC_{BC} denotes the data cost used for imposing boundary condition. Taking k as a large constant, the term DC_{BC} is set as $DC_{BC}(1,j) = k$ when the j^{th} vertex is in the bottom strip and $DC_{BC}(2,j) = k$ when the j^{th} vertex is in the top strip. For all other indices, $DC_{BC} = 0$.

A flowchart for the computational procedure is presented in Fig 3.7. The list of all the variables and pseudo-codes for all the subroutines are provided in Appendix B.2. It can be observed that this procedure allows for a unidirectional workflow with (1) Mesh generation (2) Microstructure specific calculation, i.e., identification of grain ID and setting up Neighbor cost (3) Test-specific calculation, i.e., setting up data cost and estimation of the crack. Independence of Step(1) from Step(2) allows using the same random mesh multiple times by re-estimating the Grain index of the mesh

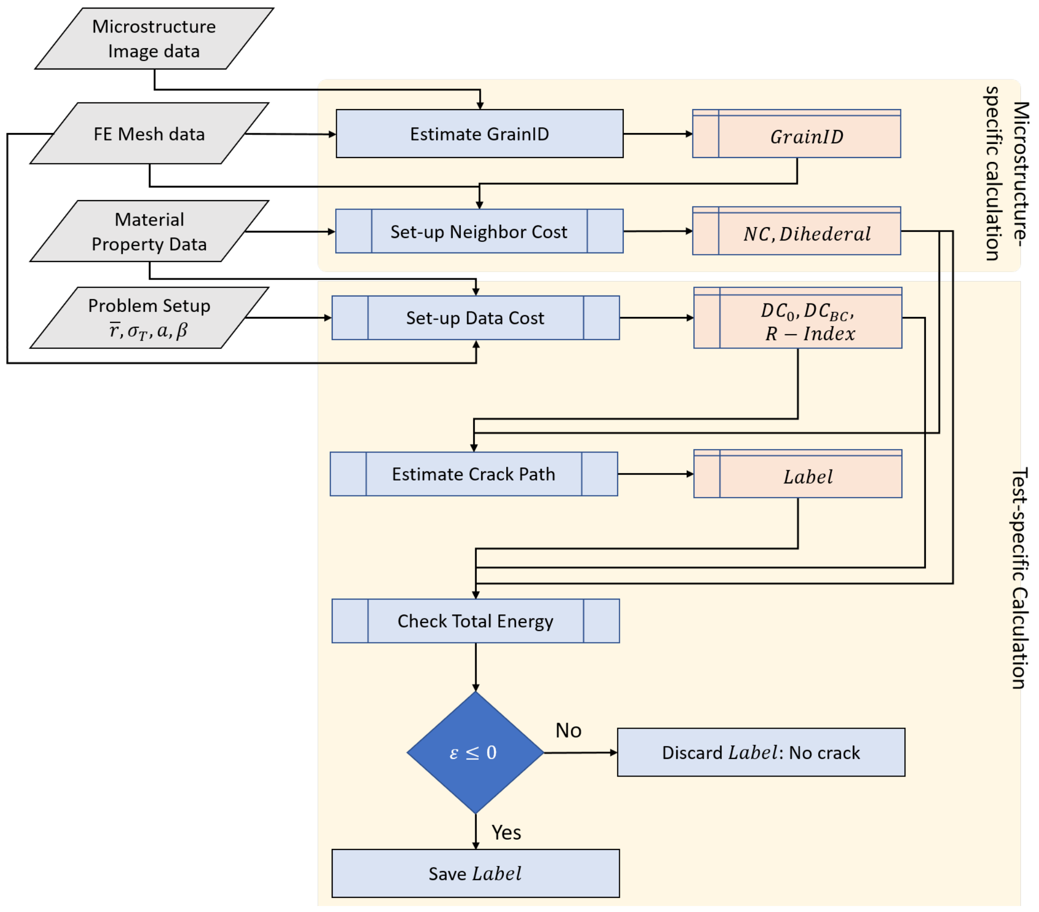


Figure 3.7: Flowchart of the procedure for crack estimation.

for different microstructures. Similarly, the independence of Step(2) from Step(3) relieves the burden of calculation of the Neighbor cost for different loading scenarios on the same microstructure. This effect is achieved by generating the list of indices, *R-Index*, of the elements in the radial region, and using it to extract the Neighbor Cost of the radial region.

3.3.3 Computational complexity

This procedure's total runtime can be estimated as the sum of the individual components of the workflow mentioned above. Before analyzing them separately, it is crucial to understand the resulting graph's topology. The Finite element mesh can also be thought of as a graph with its nodes as the vertices of a graph and the edges as the connections of the graph. Since $\mathcal{G}$ is constructed as a subgraph of a dual of a planar graph (Mesh Graph), it is planar. Based on the construction, the number of vertices, $|V|$ of $\mathcal{G}$ is equal to the number of elements (N) in the mesh. Moreover, the connections, $|E|$ in $\mathcal{G}$ is bounded from above by $3N-6$.

The *Estimate GrainID* subroutine developed in [81] uses a graph with the same topology as used in this work. The main bottleneck in this procedure, as well as the *Estimate Crack Path* procedure, is the GraphCut step. It has been shown in [82], that the runtime complexity for this step is $\mathcal{O}(N \log N)$. The complexity for the *Set-up Neighbor Cost* and *Set-up Data Cost* can be estimated as $\mathcal{O}(|V|)$ and $\mathcal{O}(|E|)$, respectively. Therefore, both the cost setup procedures are $\mathcal{O}(N)$. Finally, the *Check Total Energy* subroutine uses the sparsity of a graph to estimate the cost in $\mathcal{O}(|V|+|E|)$, i.e., it is bounded linearly by N. This analysis shows that the overall runtime complexity of this method is $\mathcal{O}(N \log N)$. The run-time of each process for various mesh-sizes is provided in Fig 3.8. It is observed that the *Set-up Data Cost* takes the most computation time for the given choices of the mesh sizes. However, the computation time for *Estimate Crack Path* shows the fastest growth with mesh size.

3.4 Application to examples

The essential characteristics of the brittle fracture using the proposed formulation are verified in this section. All simulations are conducted for quasistatic crack growth due to the abundance of analytical results in such cases. The quasistatic crack growth condition is replicated by choosing the value of σ_T such that the total energy, ε, is as close to 0 as possible while maintaining non-positivity. The reason for this choice is that, in practice, when the tensile load is increased, the crack propagation initiates as soon as the total energy of the crack is negative. The first set of simulations is conducted on a homogeneous material, and the results are verified against analytical

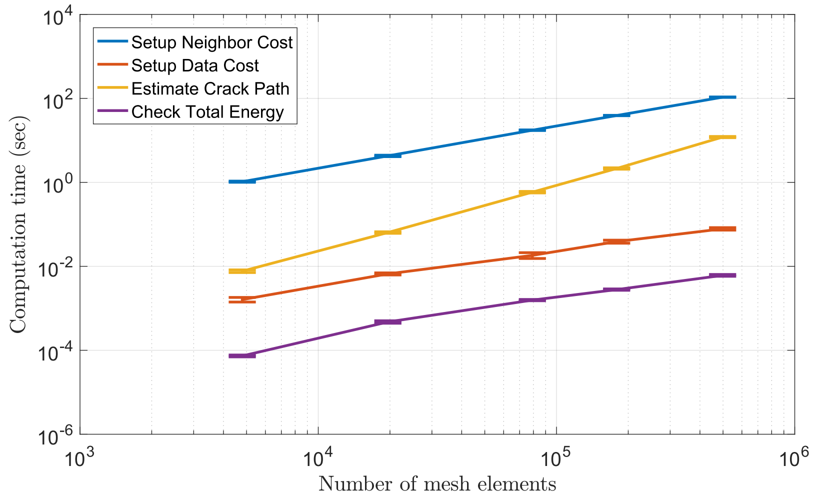


Figure 3.8: Run-time Analysis: Computation time is estimated for each process for different mesh sizes. A sample set of 20 meshes is used for each mesh size. Error bars represent the standard deviation in the computation time for each sample set.

results. In the next set of simulations, multiple grains are considered. In this case, analytical results are rare, and the results are analyzed in a qualitative sense. All numerical studies are done on a square domain, discretized into approximately $180k$ elements. Consequently, the complete graph, $\mathcal{G}$, has around $180k$ nodes and $270k$ connections. When restricted to the semicircular simulation domain, the sub-graph contains around $70k$ nodes and $105k$ connections. A mesh convergence analysis is presented in the case study for isotropic fracture (Section 3.4.1.1) that validates this choice of the mesh size. The radius ($\overline{r}$) of crack–growth region is chosen as half the image's dimension. This hyper–parameter defines the macroscopic length scale of the problem and influences the crack path in the polycrystalline medium. This effect is analyzed in Section 3.4.2.2.

3.4.1 Fracture in a homogeneous medium

Homogeneous materials have uniform properties throughout the domain. Three different cases are considered based on the form of surface energy density. In the first case, an isotropic material is studied. Next, a surface energy density with a single preferential direction is considered. Finally, a surface energy density with multiple preferential directions is studied. For simplicity, all the surface energies are formulated for 2D, so the crack's dihedral angle is uniformly zero in all the cases studied here. It should also be noted that the anisotropy is only induced through the surface energy while the energy release rate is based on isotropic material (as defined in Eq(3.2)).

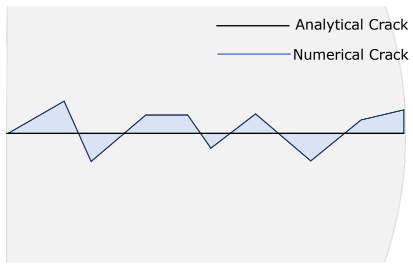
3.4.1.1 Isotropic crack

Firstly, the mesh convergence analysis is conducted using the test case of Mode-I fracture in this medium. The analytical crack grows horizontally while the numerically estimated crack grows in a zig-zag fashion about the horizontal line. The difference between the 'Analytical' and the 'Numerical' crack can be estimated using the Simple matching distance of the corresponding labeling problem. In this case, the distance can be evaluated using the following form:

$$d(C_{\text{Numerical}}, C_{\text{Analytical}}) = 2\frac{\text{Area between}(C_{\text{Numerical}}, C_{\text{Analytical}})}{\pi\bar{r}^2} \tag{3.18}$$

where, $C_{\text{Numerical}}$ and $C_{\text{Analytical}}$ represent the Numerical and the analytical crack path. An illustration of the area along with the estimated between the two cracks is presented in Fig 3.9. It is observed that $d(C_{\text{Numerical}}, C_{\text{Analytical}})$ decreases with an increasing mesh size. The mesh with approximately $180k$ elements is chosen for its reasonable accuracy and computation time.

Next, it is verified that the revised ME-Criterion used in the formulation of the energy minimization problem is, in fact, equivalent to the original ME-criterion for an isotropic material. Erdogan and Sih [78] performed a mixed-mode fracture experiment using plexiglass sheets with oblique cracks under tension. The crack's average



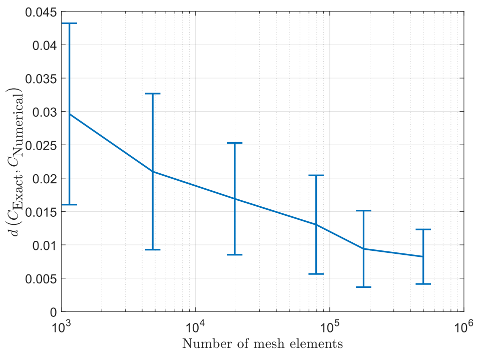


Figure 3.9: Mesh convergence analysis: $d(C_{\text{Numerical}}, C_{\text{Analytical}})$ is estimated for different mesh sizes. A sample set of 20 meshes is used for each mesh size. Error bars represent the standard deviation in $d(C_{\text{Numerical}}, C_{\text{Analytical}})$ for each sample set.

angle in their experiments is presented in Table 3.1. The crack angle, as predicted by the original ME-criterion, is also provided. The numerical simulation using the revised ME-criterion is provided in Fig3.10. The macroscopic crack propagation angle matches reasonably well with the experimental and analytical results.

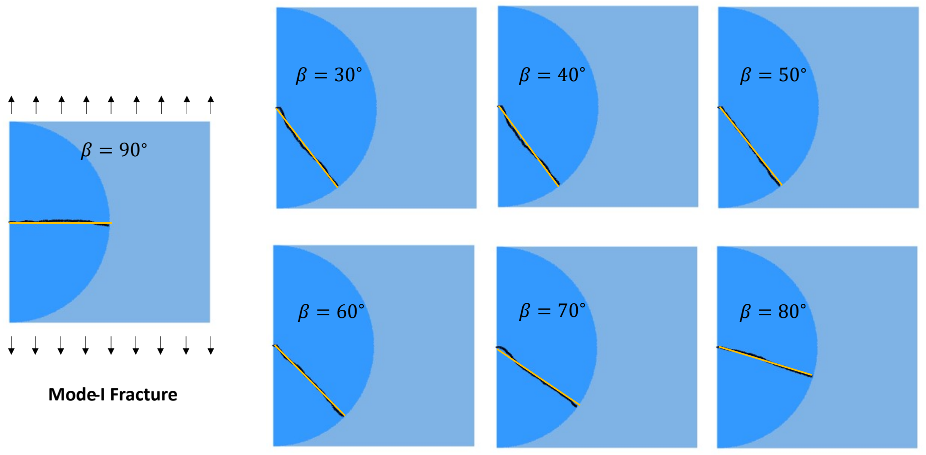


Figure 3.10: The crack path predicted in mixed-mode failure of an isotropic media. The black line represents the numerically simulated crack in the semi-circular region. The overlaid yellow line represents the corresponding macroscopic crack direction.

	$\beta = 30°$	$\beta = 40°$	$\beta = 50°$	$\beta = 60°$	$\beta = 70°$	$\beta = 80°$	$\beta = 90°$
Experiment [78]	$-62.4°$	$-55.6°$	$-51.1°$	$-43.1°$	$-30.7°$	$-17.3°$	-
ME-criterion	$-60.1°$	$-55.5°$	$-50.2°$	$-43.2°$	$-33.2°$	$-19°$	$0°$
Numerical	$-62°$	$-56°$	$-52°$	$-43°$	$-34°$	$-20°$	$0°$

Table 3.1: Crack propagation angle in mixed-mode failure

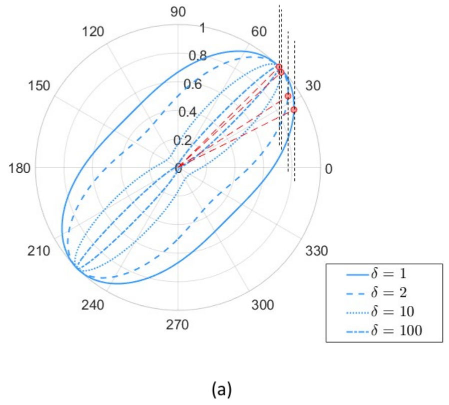
3.4.1.2 Single cleavage plane

As the next example, a material with a preferred cleavage direction for the microscopic crack is studied. The surface energy of the crack is low in the prescribed direction of the cleavage plane. The following 2-parameter model for energy function with $\gamma_0, \delta > 0$ is used:

$$\gamma(\theta) = \gamma_0(1 + \delta \cos^2(\theta - \omega)) \tag{3.19}$$

where θ is the angle of the edge normal and $\omega \in [0, \pi]$ is the preferred direction of the microscopic crack, i.e., the minimum surface energy is generated when the crack propagates along the vector $[\cos\omega, \sin\omega]$. The parameter, γ_0, denotes the preferred cleavage plane's surface energy, and $1 + \delta$ is the ratio of the maximum surface energy to the preferred cleavage plane's surface energy.

A horizontal macroscopic crack is preferred in Mode-I failure. However, the microscopic crack prefers an oblique crack with angle, $\alpha = |\omega - 90°|$. The competition between these two effects results in a crack propagating in the direction with the absolute angle less than the one predicted by the surface energy. Takei et al. [83] introduced a Wulff's diagram based criteria to determine the effective angle of the crack. In the case of a differentiable form of surface energy and Mode-I failure, their method can be described as: the crack angle is determined by the point on the γ^{-1} polar plot (with respect to the crack propagation angle) which is tangentially touched by a vertical line moving continuously from right to left. The polar plot of γ_0/γ with



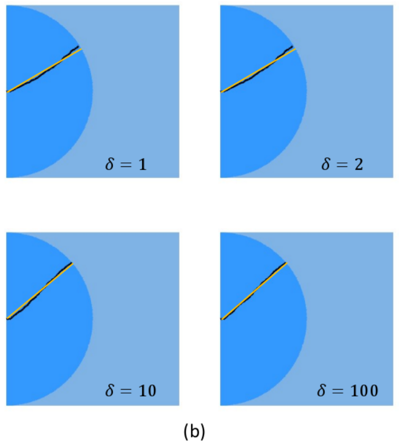


Figure 3.11: The crack path predicted in the mode-I failure of a homogeneous media with a preferred cleavage direction of 45°. (a) Polar plot of the surface energy density for different values of δ. (b) Numerically simulated crack where the black line represents the microscopic crack, and the overlaid yellow line represents the corresponding macroscopic crack direction.

respect to the crack propagation angle, $\theta' = \theta + 90°$, is presented in Fig 3.11(a) for $\delta = 1, 2, 10, 100$. The analytical values of the crack propagation angle for each δ value, as estimated by the geometric method [83], is presented in the Table 3.2. Numerical simulations for crack propagation are conducted for each value of delta, and the results are presented in Fig 3.11(b), with the numerical value of the effective angle provided in table 3.2. The normalized values of σ_T^2 are also provided. The numerically predicted effective angles of propagation match well with the analytically predicted angles. It should be noted that the surface energy in the preferred direction is the same in all cases. However, it can be observed that the critical tensile stress is significantly higher for higher values of δ. There are two reasons behind this discrepancy: (1) The effective angle of the macroscopic crack is at an angle which is different from the preferred cleavage direction, and (2) the effective surface energy is overestimated due to the zig-zag nature of the crack. The former is the desired effect predicted by [83]; however, the latter is due to mesh discretization and is undesirable. To a certain degree, it can be suppressed by using adaptive meshing techniques to iteratively generate a better-suited mesh for each test case based on the predicted crack. However, this step can lead to high computational costs.

δ	1	2	10	100
Analytical [83]	26.4°	35°	42.6°	44.8°
Numerical	28°	31°	40°	42°
$\frac{(\kappa+1)a\pi}{8\mu\gamma_0}\sigma_T^2$	2.7	3.1	5.4	30.8

Table 3.2: Crack propagation angle in mixed-mode failure

3.4.1.3 Bi-directional preference of cleavage plane

Crystalline solids usually have multiple cleavage directions, and their relative strength, along with their relative orientation to the loading direction, determines the crack path. In this section, the effect of crystal orientation on the crack path in a simplified setting is studied. Consider a material with two cleavage planes perpendicular to each other with equal cleavage energy in a mode-I test. The surface energy of the material can be chosen as follows:

$$\gamma(\theta) = \gamma_0 \left(1 + \delta \sin^2\left(2\left(\theta - \omega\right)\right)\right)$$

The physical interpretation of the parameters, γ_0, and δ is the same as in the previous case. The preferred direction of the microscopic crack is along the vector $[\cos\omega, \sin\omega]$ and $[-\sin\omega, \cos\omega]$. The polar plot of γ_0/γ is presented in Fig 3.12(a,b) for two different values of ω. It was shown by [83] that the regions of local minimum in these polar plots have a neighborhood around them in which the crack never propagates. They referred to these regions as the forbidden regions, which they determined to be the regions of negative curvature in the polar plot. The geometric method based on Wulff's diagram is extended to such cases by determining all the points that have vertical tangents in the polar plot and then removing those in the forbidden region.

For instance, in the case of $\omega = 30°$, there are two optimal crack propagation angles, while in the case of $\omega = 10°$ there is only one (as shown in Fig 3.12(a,b)) . The existence of such a region implies that in the presence of multiple optimal crack propagation angles with a forbidden region between them, the local orientation of the crack path rapidly switches between these optimal angles to accommodate the microscopically preferred path. The result is a saw-tooth pattern on the cracked surface. The fractured surfaces for $\omega = 30°$ and $\omega = 10°$ as estimated by the method in this chapter are shown in Fig 3.12(c). The saw-tooth pattern is present in the case of $\omega = 30°$ but absent in $\omega = 10°$ as expected.

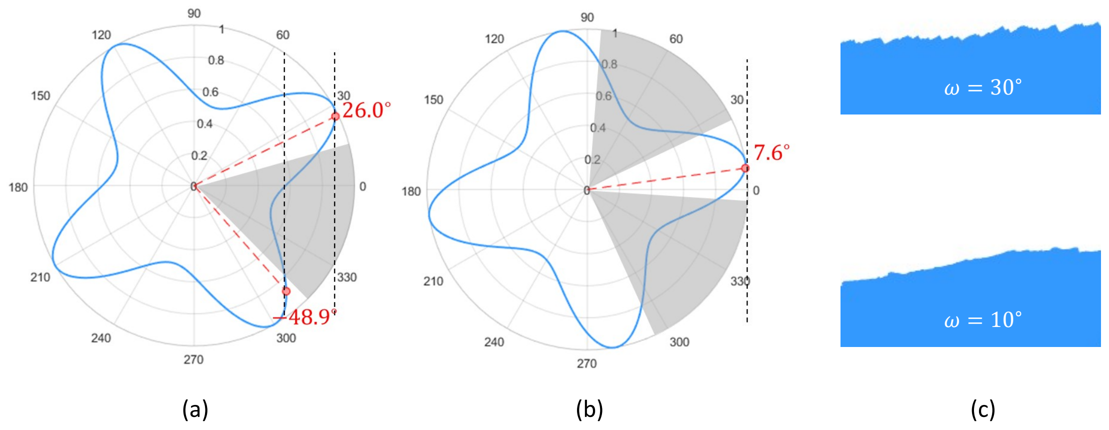


Figure 3.12: Polar plot for γ_0/γ with respect to the crack propagation angle, $\theta' = \theta - 90°$ for a bi-directional surface energy with $\delta = 1$ and the crystal orientation (a) $\omega = 30°$, (b) $\omega = 10°$. The grey area represent the forbidden regions (c) The lower region of the cracked surface for $\omega = 30°$ (top) and $\omega = 10°$ (bottom)

3.4.2 Cracks in Polycrystalline material

The presence of multiple grains has two effects: (1) The spatial variation of the preferred cleavage directions, and (2) intergranular fracture along grain boundaries. These effects are qualitatively analyzed in this section. Next, the effect of having non-zero surface energy is studied.

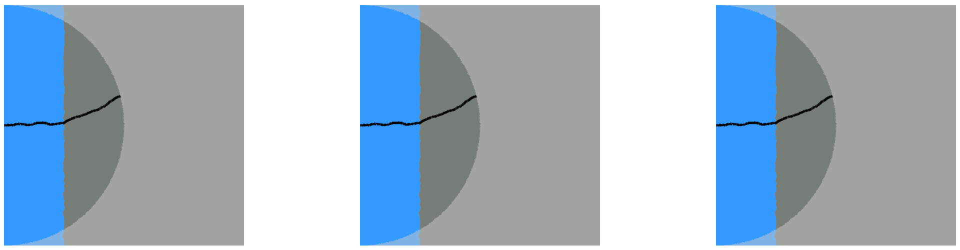
3.4.2.1 Effect of grain boundary

A bi-crystal in the mode-I fracture is considered in this section, with each grain having a single preferred cleavage plane. Both grains have the same parameters for energy described by Eq(3.19) with $\delta = 2$. The bicrystal's geometric construction is such that the grain boundary is a straight line passing through the point $0.25\overline{r}, 0.5\overline{r}$ at an angle, χ, to the horizontal axis. The crystal in the top left corner is labeled Grain 1, and the one in the bottom right corner is labeled as Grain 2. The orientation for Grain 1 is chosen to be $\omega = 0°$, and for Grain 2 is chosen to be $\omega = 45°$. Based on the analysis from section 3.4.1.2, the effective crack path angles in the single crystal of Grain 1 and 2 in mode-I fracture are $\alpha = 0°$ and $\alpha \approx 35°$, respectively. Numerical simulations are carried out by prescribing the surface energy of the grain-boundary, γ^{ig}, in the range: $\{0.5\gamma_0, \gamma_0, 1.5\gamma_0\}$ and the boundary angle, χ, in the range: $\{0°, 30°, 60°, 90°\}$. The results from these numerical simulations are presented in Fig 3.13.
The key observations from the numerical simulations are as follows:

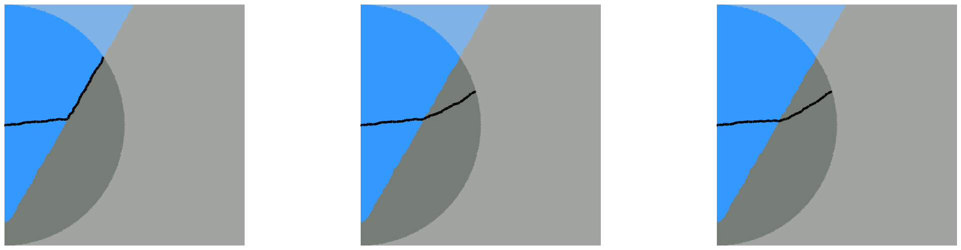
1. As γ^{ig} increases, the transgranular fracture is preferred over the intergranular fracture. This can be observed from Fig 3.13(b) where increasing γ^{ig} from $0.5\gamma_0$ to γ_0 shows the transition from transgranular to intergranular fracture.

2. The influence of the grain-boundary is more prominent when it is aligned closer to the macroscopically preferred crack path. The reason is that the crack path simultaneously minimizes the macroscopic energy release and surface energy. Therefore, the crack switches from transgranular to intergranular when the difference of surface energy compensates for the lower macroscopic energy release.

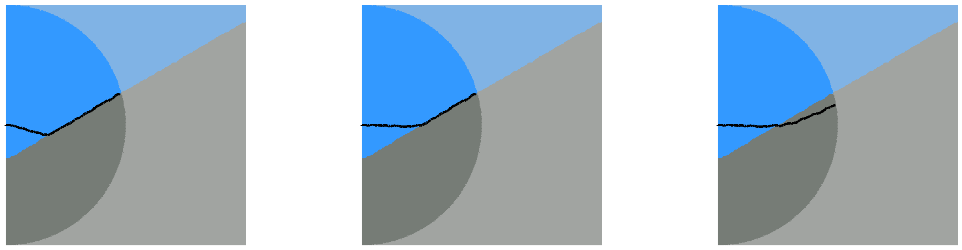
3. The grain-boundary can influence the crack path before it passes through it. This can be observed in Fig 3.13(c) with $\gamma^{ig} = 0.5\gamma_0$. The preferred crack propagation angle in Grain 1 is $0°$. However, the lower value γ^{ig} influences the crack to dip downwards to lower the crack's overall surface energy. This result



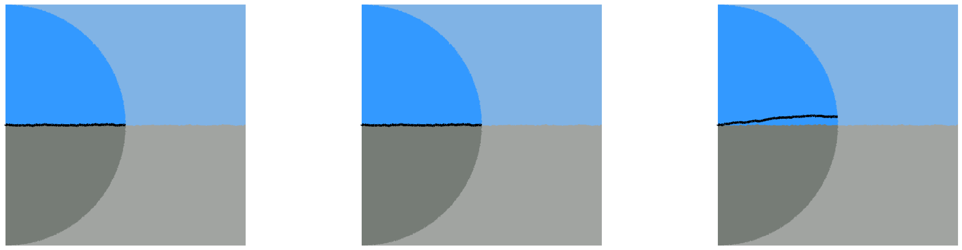
(a) $\chi = 90^\circ$ with $\gamma^{ig} = 0.5\gamma_0$ (left), $\gamma^{ig} = \gamma_0$ (center), $\gamma^{ig} = 1.5\gamma_0$ (right)



(b) $\chi = 60^\circ$ with $\gamma^{ig} = 0.5\gamma_0$ (left), $\gamma^{ig} = \gamma_0$ (center), $\gamma^{ig} = 1.5\gamma_0$ (right)



(c) $\chi = 30^\circ$ with $\gamma^{ig} = 0.5\gamma_0$ (left), $\gamma^{ig} = \gamma_0$ (center), $\gamma^{ig} = 1.5\gamma_0$ (right)



(d) $\chi = 0^\circ$ with $\gamma^{ig} = 0.5\gamma_0$ (left), $\gamma^{ig} = \gamma_0$ (center), $\gamma^{ig} = 1.5\gamma_0$ (right)

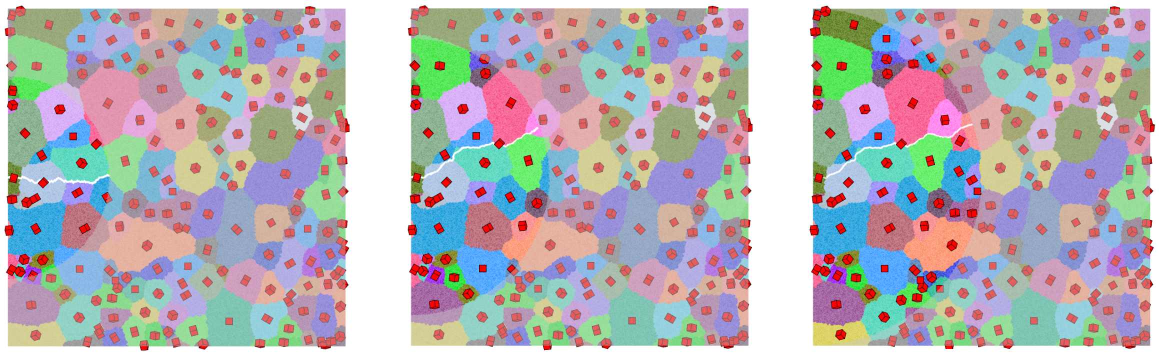
Figure 3.13: Effect of grain boundary on crack propagation

is an artifact of the global energy minimization.

A similar analysis was conducted in [61], where a phase-field model was used to analyze intergranular and transgranular crack propagation in ZrB_2 bicrystal systems. It was observed that the crack path deviates from its initial trajectory before its incidence on the grain boundary. This behavior was more prominent for weaker grain boundaries. These observations are similar to the ones made in this work; however, the length-scale of these deviations is far smaller in the phase-field study.

3.4.2.2 Effect of Macroscopic length scale

The macroscopic length scale is defined based on the choice of the parameter $\overline{r}$. A smaller choice of $\overline{r}$ implies a smaller macroscopic crack. If this parameter is chosen to represent a sub-grain length scale, then the estimated crack is expected to be closer to the corresponding homogeneous medium case. In contrast, if a larger value of $\overline{r}$ is chosen, then the predicted crack path will depend on the microstructure's spatial distribution. A thin foil specimen of polycrystalline material with cubic grains under the Mode-I tensile test is analyzed as the test case. The surface energy is determined by Eq (3.10) with parameters: $\gamma_0 = 1J/mm^2$ and $\delta = 2$. The intergranular surface energy density is taken as $\gamma^{ig} = 1J/mm^2$, and the thickness of the specimen is taken as $1mm$. The crack path is estimated for 3 different values of $\overline{r}$ and the results are presented in Fig3.14. A prominently transgranular fracture in the macroscopically preferred direction is observed for the smallest value of $\overline{r}$. As the value of $\overline{r}$ is increased, the crack path converges towards a prominently intergranular fracture and moves away from the macroscopically preferred direction. It should be pointed out that this convergence is not guaranteed, for instance, in the specimens with large spatial variations in the microstructure's statistical features. For such cases, it is recommended to tune the hyperparameter, $\overline{r}$, using experimental studies.



$\overline{r} = 0.3\times$ Image Size $\qquad \overline{r} = 0.4\times$ Image Size $\qquad \overline{r} = 0.5\times$ Image Size

Figure 3.14: Effect of $\overline{r}$ on crack path

3.4.2.3 Dynamic fracture

In previous examples, the value of σ_T was chosen such that the total energy of the crack was almost zero (i.e., $\varepsilon(\Gamma) \to 0^-$). This was done to emulate quasistatic crack growth conditions. In this section, the effect of the non-zero total energy of the crack is studied.

The test case is a thin foil specimen of cubic material with the same energy parameters as used in Section 3.4.2.2. The crack paths are predicted for three different values of σ_T, including the critical value, representing the quasistatic case. The values of σ_T and the resulting energies are presented in Table 3.3 and the predicted crack paths are shown in Fig 3.15. It should be noted that the crack paths are dependent on the microstructure. For this particular choice of microstructure, it can be observed that: (1) The transgranular crack seems to follow the trace of [100] cleavage plane as expected. (2) In the critical case, an intergranular fracture is dominant, and the crack has a non-zero macroscopic angle. This macroscopic angle of the crack decreases in magnitude with increasing σ_T. It is observed from Table 3.3 that the total energy of the crack decreases with increasing tensile stress on the specimen. However, the total surface energy of the fractured surface increases with tensile stress. This result is expected as the increase in the energy release rate shifts the competition of the microscopic surface energy and macroscopic energy release in favor of the latter energy.

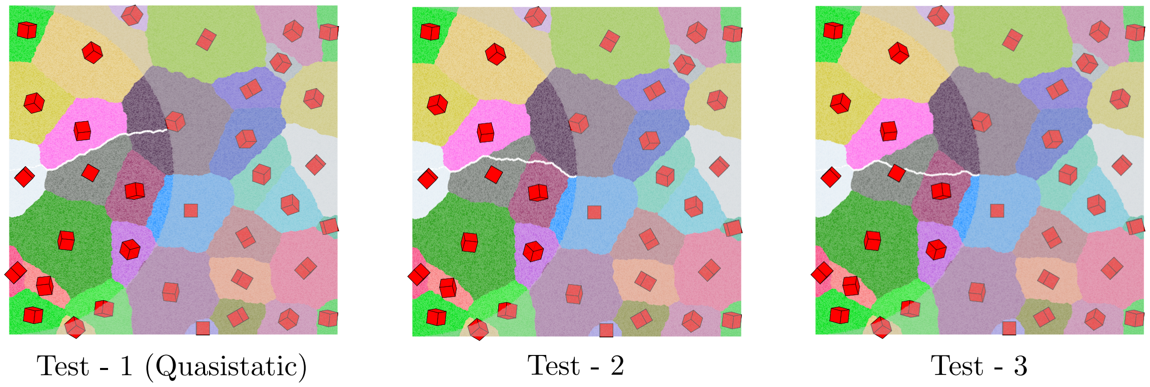


Figure 3.15: Dynamic crack propagation in a polycrystalline material

	Test - 1	Test - 2	Test - 3
$\frac{(\kappa+1)a\pi}{8\mu\gamma_0}\sigma_T^2\ (J/mm)$	2.8	10.0	20.0
Microscopic surface energy (J)	1.36	1.41	1.43
Macroscopic energy release (J)	1.36	4.97	9.97
Total Energy (J)	0.00	-3.56	-8.54

Table 3.3: Crack propagation angle in mixed-mode failure

The second point in the revised ME Criterion hypothesis states that the crack propagation is possible in any loading condition, which satisfies $\varepsilon(\Gamma) < 0$. An elastodynamic extension to Griffith's fracture as presented in [84] showed that an equilibrium crack in-plane strain mode - I failure with finite velocity (v) has an augmented energy release rate $(\widetilde{G})$ given as: $\widetilde{G}(v,\alpha) = A(v)G(\alpha)$. In the limiting case of quasistatic fracture, the solution can be recovered from the finite velocity case as $\lim_{v\to 0^+} A(v) \to 1$. Similar results are available for Mode-II failure as well. This analysis qualitatively agrees with the hypothesis of this work. Essentially, the cracked path in Mode-I dynamic fracture is likely to follow the cracked path as determined by the macroscopic conditions. Moreover, the crack velocity can be estimated by the ratio of macroscopic energy release of the dynamic crack and the respective quasistatic case. Further investigation is needed to verify this claim quantitatively. However, the rarity of experimental and analytical studies for microscopic dynamic growth in a polycrystalline material has prohibited the authors from doing so.

3.5 Conclusion

A graph-based approach is developed to predict the microscopic crack path during brittle fracture in a polycrystalline material. The method is based on an energy minimization principle with the crack's total energy as the cost. This approach naturally allows intergranular and transgranular fractures. The implementation presented in this work specializes in macroscopically isotropic materials; however, it can be easily extended to the anisotropic case by considering the respective energy release rate. One major drawback of this procedure is mesh-dependence as the crack is only allowed to move along the mesh's edges. The present work uses a mesh generated by Delaunay triangulation of randomly sampled points. The final placement of edges is dependent on the sampled random points. However, it is observed from numerical simulations that the crack path from two different mesh has a minimal difference if the mesh is fine enough.

The accuracy of the outlined method is verified against analytical results of quasistatic fracture in homogeneous materials with different types of surface energy forms. In the case of isotropic surface energy, it was observed that the crack path appears to be radial with crack angles similar to those described by the ME-Criterion for mixed-mode failure. In case a single cleavage direction is present, the crack path is again radial and tends to follow the direction prescribed by a Geometric construction method provided by [83]. In the case of multiple cleavage directions, a transition from a smooth radial crack to a saw-tooth pattern is observed, as suggested by [83]. The effect of grain boundary energy is also studied, and the expected transition of intergranular to transgranular fracture is observed when the surface energy of the grain boundary is increased. While studying the effect of the grain boundary's incidence angle, it was observed that the crack path might be affected by the boundary ahead of the intergranular fracture. The reason for this effect is that when the intergranular surface energy is too low, then the global minimization of energy may lead to sub-optimal choice locally in the transgranular region. Meanwhile, the fracture's overall energy is decreased due to the generation of a longer intergranular surface. One contribution of this chapter is in developing this approach for fracture in thin foil specimens by allowing the fractured surface to have a dihedral angle. This approach allows computing the 2D surface energy from the 3D crystal in a natural way. The simulations show that the cracked surface follows the cleavage plane's trace, as seen in many experimental studies. An unintended but favorable consequence of this formulation is that it allows for dynamic fracture as well. Simulations show that cracks with higher total energy tend to have a microscopic path which is macroscopically favorable but microscopically sub-optimal. This effect qualitatively makes sense but requires a rigorous verification with experimental or analytical studies.

The objective of this work is to introduce a graph-based procedure for estimating the microscopic crack path. The modularity of this procedure allows many possible modifications to the formalism of the fracture criterion. For instance, the ME Criterion used in this method is not suited for studying fracture due to compression. A possible remedy to this problem is to use the S-criterion, which can be implemented by replacing the energy release rate (G) with the negative of the strain energy density factor as introduced by [74]. Another possible extension of this method is the crack path prediction in ductile failure. In this case, the plastic strain energy can be included by augmenting the surface energy as $\gamma + \gamma_p$, as defined in [85]. The performance of these extensions will be addressed in subsequent works.

CHAPTER IV

Graph Coloring Method for Modeling Evolving Microstructures

Same materials with different microstructure may show vastly different corrosion resistance, strength, ductility, and toughness. This characterization at the microscale is usually done using the mean grain size and the grain size distribution. Consequently, studying the evolution of this microstructure through different processes is of scientific and engineering importance. Usually, the local energy at the grain boundaries is higher than the grains' corresponding bulk energies. This energy provides a thermodynamic driving force for the motion of the grain boundary. The motion of grain boundaries decreases its surface area (or length in 2D), appearing as growth in a few grains, and decay and consequent annihilation of other grains. As time increases, the total number of grains decreases, and the average grain size increases.

Computational methods are employed to study this problem at multiple length scales. At the atomistic level, methods like Density functional theory (DFT) and Molecular dynamics (MD) allow the characterization of different energies. At the microscale, computational approaches like the Monte Carlo Potts model, Surface Evolver, the front-tracking method, vertex dynamics, phase field methods and cellular automata use these energies to simulate the behavior of multiple grains. At this length scale, a typical engineering problem involves studying the motion of thousands of grains in a representative volume element (RVE).

The normal velocity, v, of a grain boundary Γ between grain i and j, is often modelled as the multiphase flow equation [86]:

$$v = \alpha \left(\gamma_{ij} \kappa + (b_i - b_j) \right) \tag{4.1}$$

where variable γ_{ij} represents the surface energy density of Γ, κ is the curvature of Γ and b_i is the bulk energy density of the grain i. The positive scalar quantity, α, can be regarded as the mobility which relates the driving force to the velocity of grain boundary. At equilibrium (no motion of grains), this results in a supplementary condition on the intersection of three grains, known as the triple point condition:

$$\frac{\sin\theta_1}{f_{23}} = \frac{\sin\theta_2}{f_{31}} = \frac{\sin\theta_3}{f_{12}} \tag{4.2}$$

Osher and Sethian [87] showed that this motion of grains with these velocities can be modelled using an Energy minimization method. Many Level set approaches based on this minimization have been employed since, for instance, [86]. These methods are usually plagued by the problem CFL-dependent stability. In a recent work, Estellers and coworker [88] developed a Level-set based approach that preserves distance functions and removes this CFL-dependence. Energy-based labeling methods have been formalized in [89] in the context of multiphase flow. These methods can be understood in terms of Level-sets which are constant over domains rather than distance functions. These similarities allow borrowing ideas for level-set approaches in developing labeling-based methods.

In this work, first, an energy minimization principle is developed, which emulates the first-order dynamics of multiphase flow. Then, this function is modeled in a pairwise energy framework. A critical aspect of this part is the estimation of the curve's lengths within the pairwise energy framework. This problem is especially challenging for pixel mesh where the technique of adding edges (as developed in Chapter II & III) lead to high metrication error. For this purpose, the methodology developed by [90] is followed. They borrow results from integral geometry to estimate the expected length of curves based on the labels in the pairwise formulation on pixel meshes. This work extends their methodology to Quadtree meshes, allowing simulation of higher resolution images with a low computational cost. This framework is then generalized to incorporate anisotropic grain-growth. The ideas developed here can be directly extended to 3D microstructures, and the relevant changes to the framework are listed. The idea of smoothening developed in Chapter II is extended to voxelated meshes. This method is used to remove noise from 3D experimental images of microstructures.

4.1 Problem formulation

Before formulating the Energy minimizing problem, it is crucial to ensure that the normal velocity defined in Eq (4.7) is enough to describe the grains' motion. The following theorem shows that the tangential velocities can be ignored under certain conditions by reparameterizing the curve. This theorem first appeared in [91] and is adapted in recent texts like [92]. The reader is referred to these works for proof.

Theorem IV.1. *Consider the family of curves $C(p,t)$ that solve the evolution rule, $dC/dt = u\overrightarrow{n_{||}} + v\overrightarrow{n_{\perp}}$ where $\overrightarrow{n_{||}}$ and $\overrightarrow{n_{\perp}}$ denote the tangential and normal components of the curve. The variable p parameterizes the spatial location (eg, arc-length) on the curve and t denotes the time parameter. If v does not depend on the parameterization of the curve (v is thus called an "intrinsic" or "geometric" quantity), the evolution can be converted into the solution of $dC/dt = v\overrightarrow{n_{\perp}}$, by a change of parameterization.*

Remark IV.2. It is clear that the velocity described in Eq (4.7) is in terms of intrinsic parameters, and therefore, Theorem IV.1 applies to this case.

With this property in place, the main idea behind the energy minimization approach is presented next in the form of a theorem (proof in Appendix A). The theorem investigates a single grain's motion, represented by the region, Ω, inside a domain, D. This domain is treated as another grain with some bulk energy.

Theorem IV.3. *Consider, $D \subseteq \mathbb{R}^2$ where D is a fixed region. And a open set $\Omega \subseteq D$ with smooth boundary, $\Gamma = \partial\Omega$. Then for any Ω, with $\overline{\Omega} \neq \phi$ or D, to be the extremizer of functional:*

$$F(\Omega) = b_1 \, \text{Area}(\Omega) + b_0 \, \text{Area}(D - \Omega) + \gamma \, \text{length}(\Gamma)$$

following condition is satisfied on the boundary, Γ, in the interior of domain D:

$$\gamma\kappa + b_1 - b_0 = 0 \tag{4.3}$$

Moreover, in the absence of any fixed/Dirichlet boundary conditions, Γ makes perpendicular intersections with ∂D.

Remark IV.4. It can be verified from the proof that the LHS of the Eq (4.3) is the magnitude of the gradient (upto scaling) of $F(\Omega)$ and the direction of steepest descent is normal to the boundary Γ.

$$\frac{d\Gamma}{dt} \propto (\gamma\kappa + b_1 - b_0) \, \overrightarrow{n_\perp}$$

Remark IV.5. If $\gamma < 0$, then the creation of more Γ will lower the energy. Thus the existence of a sensible solution requires that $\gamma > 0$.

Remark IV.6. A multi-label generalization of this case is presented in [86].

The Global minimization of the functional $F(\Omega)$ results in the following solution:

$$\Omega = \phi \quad \text{if } b_1 > b_0$$

$$\Omega = D \quad \text{if } b_1 < b_0$$

In either case, there is no Γ and the content of the Theorem IV.3 is identically true but moot. The phases should evolve, making small moves in each time step. This intuition conforms with annealing's physical process, where the resulting low-temperature solution is a polycrystalline solid with suboptimal energy (local minimum). These local moves can be modeled by adding a regularization term to the functional F, penalizing large grain boundary motions between time steps. This term is discussed in the next section.

4.1.1 Temporal evolution: First order dynamics

The region Ω, as used in Theorem IV.3 can be equivalently identified using it's boundary, Γ. The first order dynamics refers to the following evolution of the boundary:

$$\frac{d\Gamma}{dt} = v\overrightarrow{n} \tag{4.4}$$

where v denotes the normal velocity and $\overrightarrow{n}$ denotes the unit normal at any point on the curve. This relation is same (upto proportionality) as the motion of boundary along the direction of steepest descent as suggested in Remark IV.4.

The standard L_2 measure for change in boundary is defined as follows where 's' is some arc-length parameter:

$$\left|\left|\frac{d\Gamma}{dt}\right|\right|^2 = \int_\Gamma |v_n|^2 ds \equiv \left\langle \frac{d\Gamma}{dt}, \frac{d\Gamma}{dt} \right\rangle$$

Remark IV.7. The last equivalence shows that the norm is induced by the Euclidean inner product, $\langle ., . \rangle$

This norm is used to approximate a distance function between two contours. Consider a smooth curve, Γ_0, where each point is displaced as $d\Gamma = v_n dt$. The point-wise position of new curve is given as, $\Gamma(s) = \Gamma_0(s) + d\Gamma(s)$ where s is the arc-length parameterization of Γ_0. The distance between Γ_0 and Γ can be estimated using following relation:

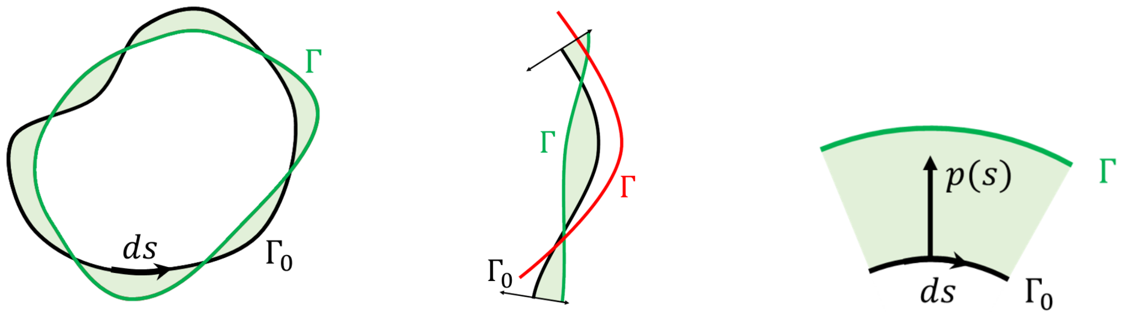
$$\text{dist}(\Gamma, \Gamma_0) \approx ||d\Gamma||^2 = \int_{\Gamma_0} (\Gamma(s) - \Gamma_0(s))^2 ds$$

Remark IV.8. The function $\text{dist}(\Gamma, \Gamma_0)$ is well defined in the case when both represents smooth closed curves. It is easy to see that if Γ_0 is smooth and velocities are smoothly varying, then the closed curve evolves into closed curves (illustrated in Fig 4.1(a)). In case of open curves, this property may not always be true. For instance, in Fig 4.1(b), if each point on Γ_0 (black curve) move in the perpendicular direction to Γ_0, then the curve can evolve into the green curve but not the red curve. Therefore, this function does not tell the distance between the red and the black curve. Moreover, $\text{dist}(.,.)$ is not a metric as this relation is not symmetric i.e. $\text{dist}(\Gamma, \Gamma_0) \neq \text{dist}(\Gamma_0, \Gamma)$. However, it has some important properties like $\text{dist}(\Gamma, \Gamma_0) \geq 0$ and $\text{dist}(\Gamma, \Gamma_0) = 0 \iff \Gamma = \Gamma_0$. It is useful to manipulate $\text{dist}(.,.)$ into the following form:

$$\text{dist}(\Gamma, \Gamma_0) \approx 2 \int_{\Gamma_0} \left(\int_0^{|\Gamma(s) - \Gamma_0(s)|} p dp \right) ds \tag{4.5}$$

where variable p denotes the perpendicular distance from the curve Γ_0. This local coordinate system is illustrated in Fig 4.1(c). Later, it will be made clear that this form is better suited for implementation in the Pairwise formulation.

With these definitions in place, following theorem (c.f. [89], proof in Appendix A) can now be used to devise a functional minimization form for simulating the first order dynamics of the mutiphase flow.



(a) Closed curve (b) Open curve (c) Local coordinates

Figure 4.1: Illustration of $dist(\Gamma, \Gamma_0)$ function where the initial curve Γ_0 evolves to Γ

Theorem IV.9. *Consider the following minimizer*

$$\Gamma_{t+\Delta t} = \arg\min_{\Gamma} \left(\alpha F(\Gamma) + \frac{\mathrm{dist}(\Gamma, \Gamma_t)}{2\Delta t} \right) \tag{4.6}$$

where Γ and Γ_t are closed curves and $F(\Gamma) \equiv F(\Omega)$ with Ω denoting the interior region of the curve. It has following property

$$\Gamma_{t+\Delta t} = \Gamma_t + (v\Delta t)\overrightarrow{n}, \qquad v = \alpha\left(\gamma\kappa + (b_1 - b_0)\right) \tag{4.7}$$

Remark IV.10. It can be easily observed that in the limit of $\Delta t \to 0$, Eq (4.7) represents the first order dynamics, Eq (4.4).

Pairwise formulation of A pairwise formulation of the functional described in Eq (4.6), it is crucial to encode boundary lengths. In the context of random meshes (Chapters II and IV), this was done simply by summing all the individual edges of partitions. Adopting this process as it is can introduce a lot more error in pixelated meshes. It is easy to see that this method estimates the L_1-distance between any two points on the image, which severely overestimates the required L_2-distance. Therefore, a different approach is taken to approximate the length of the curve accurately. This method is developed in the next section.

4.2 Estimation of Length

The length of a curve can be approximated using concepts from Integral geometry, specifically, Cauchy-Crofton relations. A formal discussion of these relations is out of this work's scope. The readers are referred to [93] for a complete and rigorous introduction to this topic. This work will adopt a much more applied outlook.
The main idea that is pursued here is that given a curve, and a set of lines, the total length of the curve is proportional to the number of intersections it makes with the lines (illustrated in Fig 4.2(a)) The following theorem (for proof refer to [94]) establishes this intuition as equality.

Theorem IV.11 (Cauchy-Crofton Formula)**.** *Let C be a regular plane curve with length, ℓ. The measure of the set of straight lines (counted with multiplicities) which meets C is equal to 2ℓ.*

This theorem will be the basis of the method developed later in the section. However, before going into this theorem's applications, it is necessary to define a measure on the set of lines. Consider a straight line L in the plane $\mathbb{R}^2$ determined by its normal parameters ρ and ϕ. In this parameterization, each line is described by a point in $\Xi = \{(\rho, \phi) : \rho \geq 0, \phi \in [0, 2\pi]\}$. This parameterization is illustrated in Fig 4.2. The Lebesque measure for a set of lines $K \subseteq \Xi$ is defined as, $\mu(K) = \int_K d\rho d\phi$. This measure is, in fact, the only measure that is invariant under rigid motions of lines ([94]).
The Cauchy-Crofton formula establishes a connection between Euclidean length $|C|$ of any rectifiable curve $|C|$ in $\mathbb{R}^2$ and a measure of a set of lines intersecting it.

$$2|C|_\epsilon = \int n_c(L) d\mu(L) \tag{4.8}$$

Function $n_c(L)$ specifies a number of times any given line L intersects (see Fig 4.2).

The next step is to adopt this theory for meshed structures. The main ideas of this approach are developed in [90]. The authors developed a pairwise energy formulation that approximates the perimeter of clusters. In terms of the Multiway-Cut energy (see Definition I.9), they prescribe interaction strength w_i such that $|C| = E(\boldsymbol{S})$ (using Eq (1.6)). This procedure's motivation is that it is easy to count intersections, $n_c(L)$, in the Multiway Cut framework. It can be observed that for a given choice of labels, the cardinality of the cut-set, $\mathcal{C}'$, i.e., set of edges with different node labels can be evaluated as:

$$|\mathcal{C}'| = \sum_{i=1}^{N_C} \left(1 - \delta_{s_{\pi(i,1)}, s_{\pi(i,2)}}\right)$$

Therefore the ingenuity of the method proposed in [90] is that they relate the weights, w_i, to μ. This method is described next; however, the final weights suggested here are slightly different from the original work. As will be made clear later in the text, this change is made to formulate parameters with symmetric edge weights and is a consequence of the finite difference scheme chosen here.

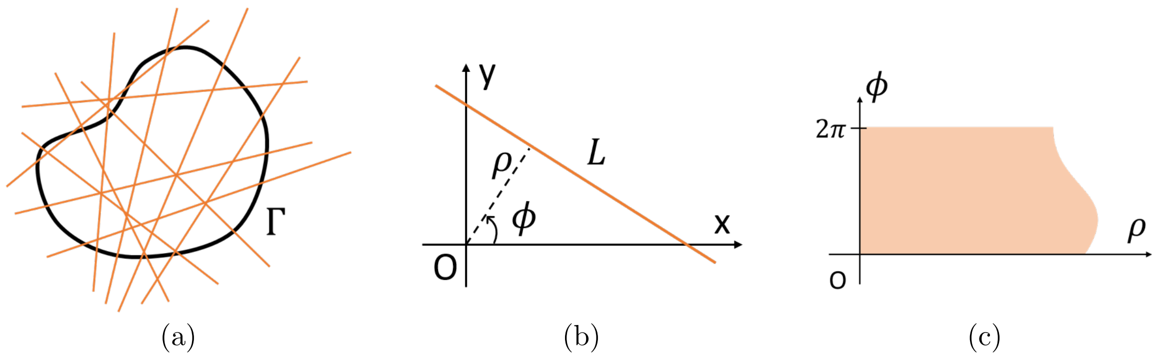


Figure 4.2: (a) Illustration for Cauchy crofton relations: length(Γ) ∝ number of intersections with lines, (b) Parameterization of a line, L, in terms of ρ and ϕ, and (c) Set, Ξ, of all lines in $\mathbb{R}^2$ represented in the (ρ, ϕ)-coordinates

4.2.1 Embedding graphs in Regular 2D grids

Every pairwise formulation starts by defining a Graph, $\mathcal{G}(\mathcal{V},\mathcal{C})$. Each pixel in the 2D regular mesh is treated as a vertex of the graph. Consequently, each vertex of the graph can be indexed as its respective (i, j) position on the grid. The connections between vertices are specified based on the approach described next.

It is desirable to have the following three properties for the connections :

1. Connections of each vertex are locally identical.

2. Connections describe a sparse graph structure.

3. The set of all connections can be used to describe set of all lines.

The first point is addressed by defining connections using a stencil. The idea is that if there is a connection between vertices, $v_{(0,0)}$ and $v_{(a,b)}$, then any other vertex, $v_{(i,j)}$ is connected with $v_{(i+a,j+b)}$. The readers should be aware of the slight sloppiness in this argument as it not necessary for $v_{(i+a,j+b)}$ to exist for all (i, j). In non-periodic images, such connections are simply ignored. While in the case of periodic images, if $v_{(i+a,j+b)}$ goes out of the domain then a connection to the respective periodic node is made. For the sake of discussion, it is assumed that there is a vertex $v_{(i,j)}$ for each ordered pair (i, j) of integers. Under the above stated simplification, a stencil is chosen as a set of vectors, $\mathcal{N}_R = \{\overrightarrow{e}_i \equiv (a_i, b_i) : 1 \leq i \leq k(R)\}$. Here R defines a strategy and $k(R)$ is the number of connections in a given stencil. Consequently, this stencil defines connections between vertex, $v_{(i,j)}$ and following vertices:

$$\{v_{(i+a_\alpha,j+b_\alpha)} | (a_\alpha, b_\alpha) \in \mathcal{N}_R\}$$

The second point in the desirable properties can be addressed by choosing a stencil that contains vectors of small length. This allows connections only between vertices that are relatively closer to each other. It can be achieved by adopting a policy of only allowing stencil's vectors with length smaller than a prescribed value. For notational convenience, from here on, R as in $\mathcal{N}_R$, will denote the maximum length of the vectors contained in $\mathcal{N}_R$ i.e. if $(a, b) \in \mathcal{N}_R \implies a^2 + b^2 \leq R^2$.

Finally, the third requirement is addressed by adopting the following two policies for the stencil: (i) if $\overrightarrow{e}_i, \overrightarrow{e}_j \in \mathcal{N}_R$ then $\overrightarrow{e}_i$ and $\overrightarrow{e}_j$ are not parallel i.e. $a_i b_j \neq a_j b_i$ (ii) if $(a, b) \in \mathcal{N}_R$ then $|a|$ and $|b|$ are relative primes. Therefore, any line, L, passing through origin with a rational slope, p/q where p and q are relative primes, can be written as a union of following connections:

$$\{(v_{(iq,ip)}, v_{(iq+q,ip+p)})|i \in \text{Integers}\}$$

The above mentioned policies avoid overlap of connections in the construction of a line. A consequence of these conditions is that if $\overrightarrow{e}_k \in \mathcal{N}_R$ then $-\overrightarrow{e}_k \notin \mathcal{N}_R$. Without loss of generality, the elements of $\mathcal{N}_R$ are chosen from 1st and 2nd quadrant. Moreover, the set $\mathcal{N}_R$ is ordered based on the angle, φ_k, of each vector, $\overrightarrow{e}_k$ from the horizontal axis. Examples of stencils that follow, these conditions are given below:

$$\begin{aligned}
\mathcal{N}_1 &= \{(0,1),(1,0)\}\\
\mathcal{N}_{\sqrt{2}} &= \{(0,1),(1,1),(1,0),(-1,1)\}\\
\mathcal{N}_{\sqrt{5}} &= \{(0,1),(2,1),(1,1),(1,2),(1,0),(-1,2),(-1,1),(-2,1)\}\\
\mathcal{N}_{\sqrt{10}} &= \{(0,1),(3,1),(2,1),(1,1),(1,2),(1,3),(1,0),(-1,3),(-1,2),(-1,1),\\
&\qquad (-2,1),(-3,1)\}\\
\mathcal{N}_{\sqrt{13}} &= \{(0,1),(3,1),(3,2),(2,1),(1,1),(1,2),(2,3),(1,3),(1,0),(-1,3),(-2,3),\\
&\qquad (-1,2),(-1,1),(-2,1),(-3,2),(-3,1)\}
\end{aligned}$$

It can be noted that a consequence of the conditions imposed on $\mathcal{N}_R$ is that if $R < R'$,

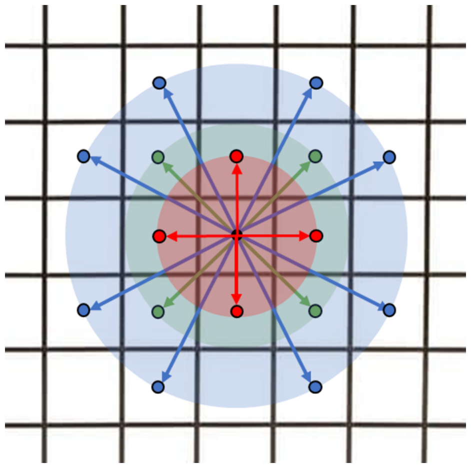


Figure 4.3: Connections in a stencil. The red region denotes the stencil, $\mathcal{N}_1$. The stencil, $\mathcal{N}_{\sqrt{2}}$ is constructed by adding connections in the green region to $\mathcal{N}_1$. The neighborhood, $\mathcal{N}_{\sqrt{5}}$ is constructed by adding the nodes in the blue region to the neighborhood, $\mathcal{N}_{\sqrt{2}}$.

then $\mathcal{N}_R \subseteq \mathcal{N}_{R'}$. The connections in the first three stencils are shown in Fig 4.3.

4.2.2 Numerical approximation to Cauchy-Crofton relation

First, it is essential to establish a relationship between the space of all lines and the set of all edges of the graph. In the previous section, a line, L, passing through the origin and with rational slope was constructed using a set of graph connections. Next, the set of all lines parallel to this line that can be represented on the grid are considered. The next proposition reveals more information about this set of lines.

Proposition IV.12. *Consider the set of all lines with rational slope p/q (p and q are relative prime) that pass through atleast one grid point (i.e a point $(a, b) \in \mathbb{I}^2$). These lines are regularly placed with following distance between two nearest lines:*

$$\Delta\varrho = \frac{1}{\sqrt{p^2 + q^2}} \tag{4.9}$$

Remark IV.13. If the grid spacing is δ instead of the assumed value of 1, then the above relation can be modified as

$$\Delta\varrho = \frac{\delta}{\sqrt{p^2+q^2}}$$

All lines that are representable by the edges of the graph can now be described as a element, $(\varrho, \varphi) \in \Xi_G$ where,

$$\Xi_G = \left\{ \left(\frac{k}{\sqrt{a^2+b^2}}, \varphi \right) \middle| (a,b) \in \mathcal{N}_R, \tan(\phi) = a/b \in [0,\pi), k \in \mathbb{I} \right\} \tag{4.10}$$

Remark IV.14. This representation does not adhere to the one used for describing the set of all lines Ξ in the Cauchy-Crofton relations. The distance parameter was restricted to the non-negative reals in the initial description. However, it is allowed to take negative values in the above notation.

This issue is resolved by adopting a variation of the latter notation, which is better suited for the numerical arguments that are being presented next. More specifically, any line L is now represented as an ordered pair $(\varrho, \varphi) \in \Xi'$ with

$$\Xi' = \{ (\varrho, \varphi) | \, \varrho \in \mathbb{R}, \varphi \in [\omega_0, \pi + \omega_0) \} \tag{4.11}$$

The reason for the arbitrary appearance of the variable, ω_0, will be made clear later on. Nevertheless, this change is summarized as: "The slope, m, of the line is measured as $\arctan m \in [\omega_0$ and $\pi+\omega_0)$. The line's distance from the origin is signed, and positive in the direction along the vector with angle $\pi/2 + \arctan m$ (chosen arbitrarily)."

Remark IV.15. The Cauchy-Crofton relations in this notation can now be reformalized as:

$$|C| = \frac{1}{2}\int n_c d\mu = \frac{1}{2} \int\limits_{\omega_0}^{\pi+\omega_0} \int\limits_{-\infty}^{\infty} n_c(\varrho,\varphi) d\varrho d\varphi \tag{4.12}$$

The numerical estimation of this integral is carried out by discretizing the set, Ξ', using a grid. This is achieved by first discretizing the space Ξ' with horizontal strips with with centers at $0 = \varphi_1 < \varphi_2 < \ldots < \varphi_{\mathcal{N}_R} < \pi$. The φ-axis is divided into regions, $\{[\omega_{i-1}, \omega_i)\}_{i\in\{1,\ldots|\mathcal{N}|_R\}}$ with $\omega_{|\mathcal{N}|_R} = \pi + \omega_0$. The values of ω_i are chosen as the angle bisector of ϕ_i and ϕ_{i+1}. This relation is formalized as follows:

$$\omega_i = \begin{cases} \frac{\varphi_{|\mathcal{N}_R|} - \pi}{2} & \text{if } i = 0 \\ \frac{\varphi_{i+1} + \varphi_i}{2} & \text{if } 1 \leq i \leq |\mathcal{N}_R| - 1 \\ \frac{\varphi_{|\mathcal{N}_R|} + \pi}{2} & \text{if } i = |\mathcal{N}_R| \end{cases} \tag{4.13}$$

The size of each region containing line with angle ϕ_i, denoted as $\Delta\varphi_i$, is estimated as:

$$\Delta\varphi_i = \omega_i - \omega_{i-1} = \begin{cases} \frac{\pi + \varphi_1 + \varphi_2 - \varphi_{|\mathcal{N}_R|}}{2} & \text{if } i = 1 \\ \frac{\varphi_{i+1} - \varphi_{i-1}}{2} & \text{if } 2 \leq i \leq |\mathcal{N}_R| - 1 \\ \frac{\pi - \varphi_{|\mathcal{N}_R|-1}}{2} & \text{if } i = |\mathcal{N}_R| \end{cases} \tag{4.14}$$

Next, the k^{th} strip is further discretized into uniformly spaced elements of length, $\Delta\varrho_k = \delta/||e_k|| \equiv \delta/\sqrt{a_k^2 + b_k^2}$. Thus each element of $\Xi_G \subseteq \Xi'$ is treated as a node in the discretized space of Ξ'. The Cauchy-Crofton relation in this discrete space can be approximated as the following summation:

$$|C| \approx \frac{1}{2} \sum_{k=1}^{|\mathcal{N}_R|} \sum_{j\in\mathbb{I}} n_c(\varrho_j, \varphi_k) \Delta\varrho_k \Delta\varphi_k$$

where $\Delta\varrho_k\Delta\varphi_k$ is the area of the element of the grid corresponding to the lines in the direction, $\overrightarrow{e}_k$. And, $n_c(\varrho_j, \varphi_k)$ is the number of intersection that a curve makes with the line $L(\varrho_j, \varphi_k)$. Since a stencil is used to define these lines and parallel lines are given the same weight, w_k, the dependence on ϱ_j can be dropped. Consequently, the length can be written in terms of $n_c(\varphi_k) = \sum_{j \in \mathbb{I}} n_c(\varrho_j, \varphi_k)$ (i.e. intersections that the curve makes with a family of lines with slope φ_k) as:

$$|C| \approx \sum_{k=1}^{n_G} n_c(\varphi_k) \frac{\delta \Delta \varphi_i}{2||e_k||}$$

In the familiar Multiway cut framework, this is equivalent to choosing weight w_k as:

$$w_k = \frac{\delta \Delta \varphi_i}{2||e_k||} \tag{4.15}$$

The computation of flows with Pixel-meshes can be found in [89]. However, some microstructures have large grains, and the grain boundary motion is often restricted to the peripheral region of these grains. This feature can be utilized to simulate the problem in a smaller graph, where the deep interior of any grain can be treated as a single node that requires higher energy to change labels. For this reason, a Quadtree-based implementation of this theory is developed in the next section.

4.3 Computation with Quadtree structure

Quadtree structures are often used in image processing for reducing the size of an image while at the same time capturing its essential features. They operate on the idea that if many pixels in a neighborhood have similar features, they can be clustered together and saved as a single node. In the context of the microstructure image, only the pixels near the grain boundary have contrasting features. Therefore, the pixels that are further from the grain boundary can be clustered together and treated as a single node in the embedded graph. See Fig 4.4(b) for a comparison of a circular grain represented using a Pixelated and Quadtree mesh with a similar number of nodes. However, the motivation here is not the compression of data. The graph-labeling

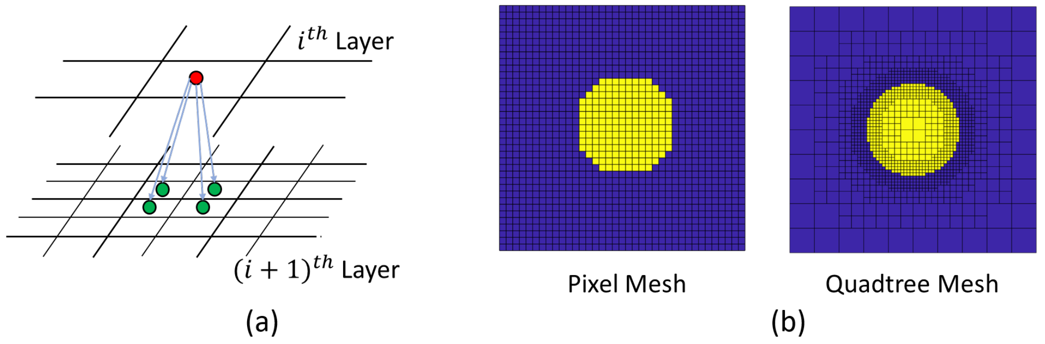


Figure 4.4: (a) Illustration shows the relationship of a pixel in parent layer and its children (b) Comparison of two circular grains represented using Pixel Mesh and Quadtree mesh with similar number of active nodes. The pixelated mesh has a discretization of 38 × 38 while the Quadtree mesh has 4 layers with the initial layer of size 10 × 10

procedure is a time-consuming step, and the capability to embed large problems in smaller graphs gives a tremendous numerical advantage.

The structure of Quadtree can be thought of as a layer of refinement on a pixelated mesh. Starting the coarsest layer on top (of discretization size $N_d \times N_d$ i.e. each direction has N_d pixels), each pixel is split into 4 pixels (2 × 2 grid) as illustrated in Fig 4.4. Thus the second layer has a total of $4N_d^2$ pixels. This procedure of adding children to each pixel is repeated $N_p - 1$ times where N_p is the user-defined number of pixelated layers. The pixels in the last layer, also known as the leaves, have the smallest size. The size of this smallest pixel dictates the resolution of the image. For instance, in Fig 4.4(b), even though Quadtree mesh has fewer nodes than the pixelated mesh of size 38 × 38, it offers the resolution similar to a pixel mesh of size 64 × 64. Next, the minimization problem in these two meshes is formalized in terms of pairwise energy.

4.3.1 Pairwise formulation for pixel mesh

The implementation of multiphase growth on pixelated-mesh can be equivalently posed in term of Eq (1.3) as:

$$\begin{aligned} U(s_i = \ell) &= \delta^2 f_l + \frac{2\delta^2}{\Delta t} \min_{\{j|s_j=\ell\}} d_{ij} \\ V(s_i, s_j) &= \begin{cases} 0 & s_i = s_j \\ 1 & s_i \neq s_j \end{cases} \end{aligned} \tag{4.16}$$

where ℓ represents some index of the grain with f_l denoting its bulk energy density. The second term in U encodes the regularisation term. d_{ij}, refers to the L_2-distance between nodes (pixels) i and j. The minimization step, $\min_{\{j|s_j=\ell\}} d_{ij}$ gives the distance between pixel i and the closest cluster with label ℓ. This minimum approximates the value of p described in Eq (4.5). One can also observe that if the node i is inside a cluster of label ℓ, then the regularization term evaluates to zero, this may seem to conflict with the definition of $dist(.,.)$ function, however, the total regularization evaluated as sum over all labels is still positive so there is no contradiction. The term $V(s_i, s_j)$ is the usual Multiway cut type of energy. The weights J_k in Eq (1.3) are decided using the stencil. It is evident that only the field term, $U(\ell)$, need to be updated in each iteration as the regularization needs to be updated.

4.3.2 Pairwise formulation for Quadtree mesh

The quadtree structure is such that any point in the domain belongs to exactly one pixel. Depending on the layer it belongs to, this pixel may have a different size. Therefore, the pairwise cost parameters described in Eq (4.16) are valid for Quadtree mesh as well with appropriate selection of Pixel size, δ. The main challenge in implementing this procedure in Quadtree mesh is that it does not preserve the uniform structure of pixel mesh, and hence the stencils cannot be employed to write the pairwise energy, i.e., the parameter J_k cannot be determined in the manner it was done in the case of Pixel mesh. However, each layer's structure is uniform, and the interaction weights of pixels within the same layer can be easily computed using the Crofton-relations for pixel mesh using the appropriate pixel size. Therefore, only the connection weights between neighbors belonging to different layers need to be modeled. Choosing a coarser pixel enforces that each of its children's label changes simultaneously to the same value. Thus an intuitive way to model the edge weights between a smaller and larger pixel is by summing all the connection weights between the smaller pixel and the larger pixel's descendants with the same size as the smaller pixel. In a slightly more formal tone, consider the nodes v_i^l and v_j^m where the superscript defines the layer number, and subscripts denote the index of the node. Without loss of generality, assuming $l < m$ i.e. Layer-l is coarser than Layer-m, the weight, w_{ij}, between v_i^l and v_j^m can be estimated as:

$$w_{ij} = \sum_{v_k^m \in \text{Descendants}(v_i^l)} w_{kj}$$

Unlike the usual Quadtree method where a pixel is split when it contains a boundary (Γ), in this case, the pixel is split when one of its neighbors contains a boundary. This neighborhood can be decided by looking at pixels that lie within a ball of a prescribed radius. Choosing a larger ball result in a less efficient quadtree structure. However, choosing a smaller ball can induce more error in length estimation as the neighborhood's resolution defines the discretization for the Crofton relation. The pseudocodes and recommended data structure for implementing this problem are presented in Appendix B.3.

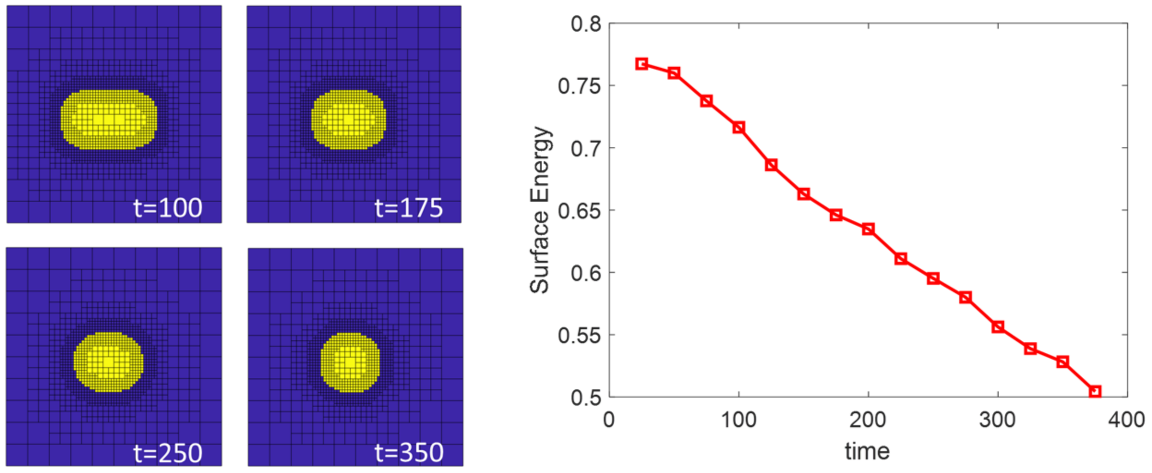


Figure 4.5: Simulation results for a pill shaped grain.

4.4 Simultation Results

As a first test case, simulation of a pill-shaped grain is considered (see Fig 4.5). The problem's setup is similar to the one used in Theorem IV.3 with no bulk energies. It is observed that the grain first reduces in length, and then once it achieves a spherical shape, it starts shrinking. This behavior is expected in this problem. It is also observed that the surface energy (also the total energy) continuously decreases, as is required.

In the next case, a 4-point junction of grains is considered (see Fig 4.6). It is known that these points are not stable, as is evident through the simulation. The initial junction splits into two triple-point junctions. Moreover, the junctions form approximately equal 120° angles between grain boundaries. This observation follows from the triple point junction's analytical solution given in Eq (4.2) with equal surface energies. Moreover, it can also be seen that the grain boundaries make a vertical intercept to the domain. This result is expected for a non-periodic domain, as suggested in Theorem IV.3.

Finally, a simulation of periodic microstructure is considered in Fig 4.7. As expected, the smaller grains are annihilated as time progresses. Meanwhile, larger grains get

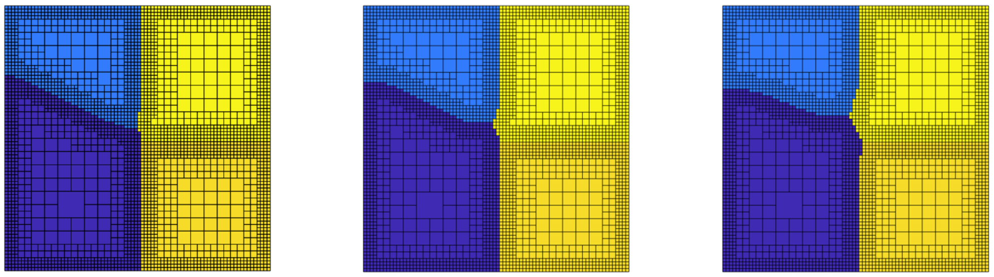


Figure 4.6: A 4-point junction splitting into 2 3-point junctions. All grains have no bulk energy density and surface energy between each pair of grain is also same.

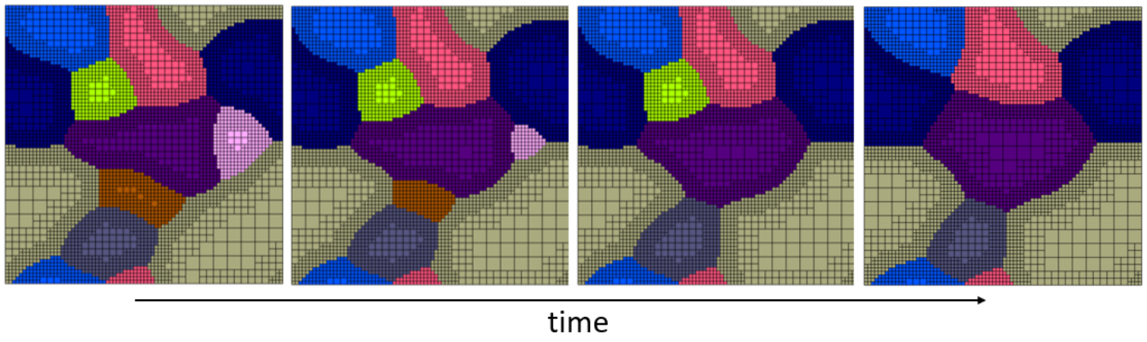


Figure 4.7: simulation of a polycrystalline microstructure

bigger. The condition for perpendicular intercept to the boundary is not valid here because of the periodicity.

4.5 Extensions

The method presented so far only allows growth laws represented as Eq (4.1). However, the framework developed here allows for much more general evolution laws. For instance, spatial variations in the surface energy γ can be directly modeled as suggested in [92, 89]. In this study, the first extension considers the case when surface energies are directionally-dependent or anisotropic. Such surface energies can be computed by manipulating the metric used to estimate the curve's length. Next, the ideas are extended to 3D voxelated structures. In this case, surface area minimization leads to mean curvature flow. This idea is used to develop a physics-inspired method for denoising experimental microstructure images.

4.5.1 Modelling anisotropic surface energies for non-equiaxed microstructure

In this section, the surface energy is treated as a function of the surface normal i.e. $\gamma \equiv \gamma(\overrightarrow{n})$. This better encapsulates the physics of crystal growth where energies are directionally-dependent. The minimization of the total energy (sum of bulk and surface energy) results in a new growth law. This growth law can evaluated using the following theorem (proof in Appendix A):

Theorem IV.16. *In the setup of Theorem IV.3, consider the following functional:*

$$F(\Omega) = f_1 \operatorname{Area}(\Omega) + f_0 \operatorname{Area}(D - \Omega) + \int\limits_{\Gamma} \gamma(\overrightarrow{n}(s))ds$$

where $\gamma(\overrightarrow{n})$ is the surface energy in the direction of unit normal, $\overrightarrow{n}$. The extremizer satisfies the following condition on the boundary, Γ:

$$\nabla . \left(\gamma \overrightarrow{n} + (\nabla \gamma . \overrightarrow{t}) \overrightarrow{t} \right) + f_1 - f_0 = 0 \tag{4.17}$$

where $\overrightarrow{n}$ and $\overrightarrow{t}$ are normal and tangents of Γ. Moreover, in the absence of any fixed/Dirichlet boundary conditions, Γ intersects ∂D at an non-zero angle such that following condition holds true:

$$\gamma \overrightarrow{n} . \overrightarrow{n}_{\partial D} + (\nabla \gamma . \overrightarrow{t}) \overrightarrow{t} . \overrightarrow{n}_{\partial D} = 0$$

Remark IV.17. It is an easy exercise to check that choosing γ as a constant simplifies the expressions to the ones in Theorem IV.3. Moreover, the regularization for temporal evolution is carried out in the same way as before in Theorem IV.9.

The evaluation of surface energy is still carried out using the Crofton-relations. This can be achieved by choosing a metric space, $\mathcal{M}$ such that the following relation holds for all possible boundaries Γ

$$\int\limits_{\Gamma} \gamma(\overrightarrow{n}(s))ds \equiv |\Gamma|_{\mathcal{M}}$$

where $|C|_{\mathcal{M}}$ denotes the length of the boundary in the given metric. [90] considered a special case where this length is measured using a Riemannian metric as follows:

$$|C|_{\mathcal{M}} = \int\limits_{\Gamma} \sqrt{\overrightarrow{t}^{T}.D.\overrightarrow{t}}ds \tag{4.18}$$

where a positive definite matrix D specifies the local Riemannian metric at a given point in the image and $\overrightarrow{t}$ is a unit tangent vector to the contour. In this case, the Crofton formula (Eq (4.8)) is estimated as [90]:

$$2|C|_{\mathcal{M}} = \int \frac{detD}{2\left(u_L^T D u_L\right)^{3/2}} d\mu(L) \tag{4.19}$$

where u_L is the unit vector[1] in the direction of line L. Moreover, the weights of the stencils are estimated as:

$$w_k = \frac{e_k^T e_k \det D}{2(e_k^T D e_k)^{3/2}} \delta\Delta\phi_i \tag{4.20}$$

It should be remarked that arbitrary surface energy functions cannot be modeled in this manner. It was shown by [95], that the most general case that can be considered in this framework is when $|C|_{\mathcal{M}}$ is measured with respect to a subclass of Finsler metric. Interested readers are referred to their work for further discussion on this topic.

[1]The reference to unit tangent is made in the Euclidean metric and not in the Riemannian metric

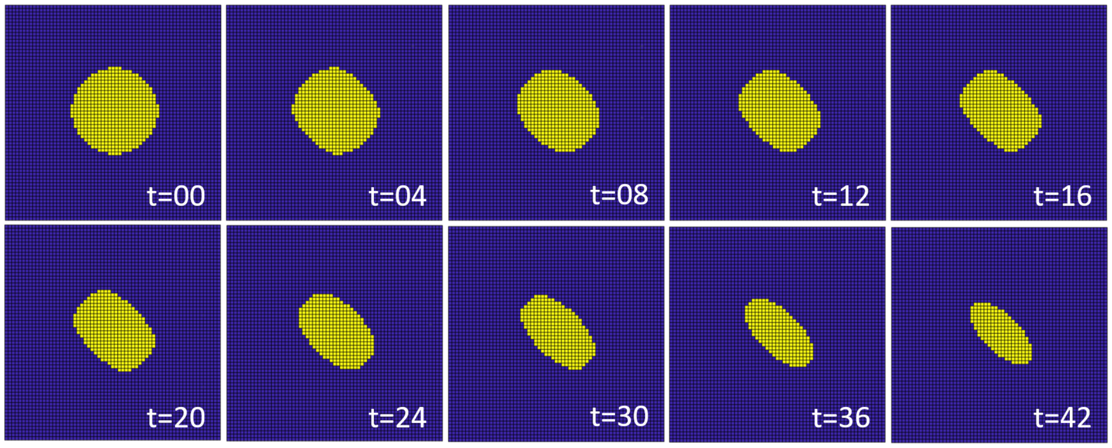


Figure 4.8: Simulation of an initially circular grain with surface energy estimated using Eq (4.18) with $D_{11} = D_{22} = 1, D_{12} = D_{21} = 0.75$.

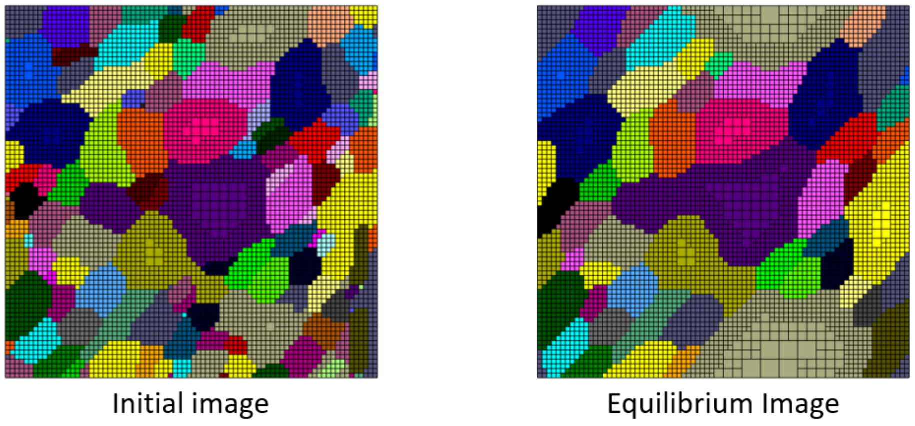


Figure 4.9: Initial and final image of a microstructure estimated using Eq (4.18) with $D_{11} = D_{22} = 2.5, D_{12} = D_{21} = -1.5$.

4.5.2 Modelling 3D grains for denoising experimental Microstructure image

Crofton formula can also be extended to estimate lengths of cures and surface area of hypersurfaces in higher dimensions. These relations can be found in the senior thesis of Sweeney [96] along with the proofs. A particularly useful corollary is for estimating the surface area of surfaces in 3D. In this case, the Crofton is modified as follows:

Theorem IV.18 (3D Crofton)**.** *Let M be a 2D surface with an area, A. The measure of the set of straight lines (counted with multiplicities) which meets M is equal to $\pi^2 A/2$.*

The numerical version of this relation can be formalized by following the same strategy to construct stencils, $\mathcal{N}_R$ in 3D. The weights of the connections are estimated as:

$$w_k = \frac{\delta^2}{\pi||e_k||}\Delta\phi_k$$

Here, the $\Delta\phi_k$ represents the solid angle in contrast to the 2D case. It is evaluated using the strategy proposed in [97, 98]. Each vector $\overrightarrow{e}_k$ is projected onto a unit sphere. The intersecting points are then used as a germ/grain for the Voronoi tessellation of the sphere. The surface area of each spherical domain is then used as the weight for the corresponding direction.

It can be shown that the first variation of the surface area results in Euler-Lagrange equations corresponding $H\overrightarrow{n} = 0$ where H is the mean curvature given in terms of principal curvatures (maximum and minimum of the normal curvature a given point) as: $H = 0.5(\kappa_1 + \kappa_2)$. The proof of this result can be found in [92]. This property is useful in denoising 3D microstructure experimental data. These data are usually provided in voxelated images, with each voxel representing a grain identifier. Taking a similar approach as in Chapter II, the Eq (2.1) is used with the weights described by the Crofton relations based stencil. The results are presented in Fig 4.10, where the procedure shows to smoothen the data by relabeling the noisy voxels. Moreover, using a larger stencil provides better results in terms of the curvature of grains.

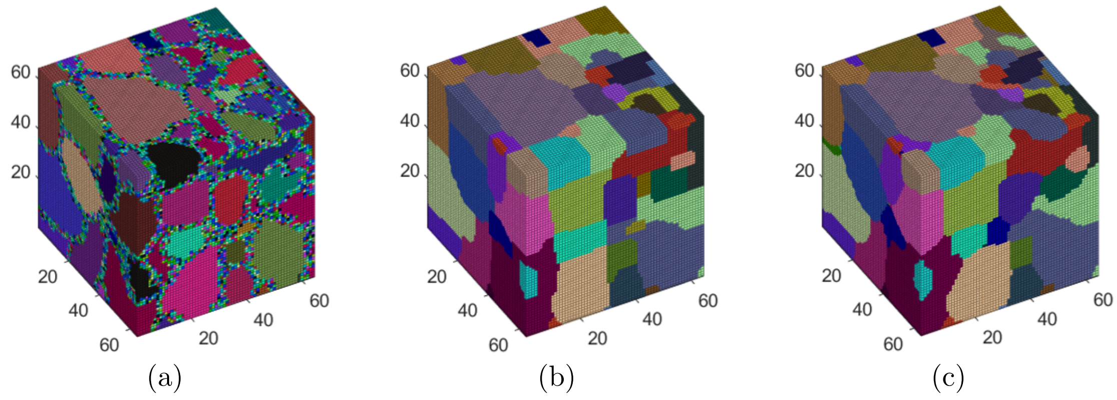


Figure 4.10: Denoising 3D microstructure data as a graph labeling problem. (a) Noisy raw data from experiments (b) Microstructure after smoothening procedure using $\mathcal{N}_1$ (c) Microstructure after smoothening procedure using $\mathcal{N}_{\sqrt{6}}$

4.6 Conclusion

An energy-based model is developed to simulate multiphase flow for studying microstructures evolution. This model is implemented in the pairwise energy form. The interaction energy formalized in this chapter is the same as the one presented in Chapter II. However, the embedded graph is non-planar. Thus this formulation is computationally more intensive than the formulation of the surface on random meshes, but at the same time, metrication errors are severely suppressed. In terms of flexibility to use different surface energy functions, the random mesh has an advantage over the pixelated mesh, limiting the surface energies that can be represented using lengths in a particular class of metric spaces.

A significant contribution of this work is to extend this framework to Quadtree mesh. The evolution of microstructures in both periodic and non-periodic domains are exhibited using examples. It is shown that the methodology captures the necessary physics of microstructure evolution in terms of annihilation and merger of grains.

Moreover, the equilibrated solution matches the analytical conditions at the triple-point junctions. The evolution laws for anisotropic length metrics are identified and are used to simulate non-equiaxed microstructures. The ideas can be extended towards segmentation of 3D microstructures where unlabelled grains in experimental images can be identified by minimizing data-cost and by using a regularization term for the surface energy. This procedure can also be used to remove the noisy voxels of images while preserving the overall data microstructure. Future work may include inclusion of strain energy, in addition to bulk and interfacial energy, in the formulation to model other material science problems such as growth and coarsening of precipitates within a grain [99].

This work has identified a few areas where further research is required for improving microstructure evolution models. In its present state, the surface energies that can be represented in the pairwise formulation are restricted by corresponding length metrics that can be modeled using Crofton relations. This restriction is a significant hurdle in developing a complete simulation suite for microstructure evolution where grain boundary energy is much more complicated. This work has also shown that metrication errors can be suppressed using concepts from integral geometry. Currently, there is no such framework for non-uniform meshes. Introducing such ideas, by possibly including higher-order connectivity in the graphs used in Chapter II and III can remove the restriction of random mesh that was assumed in those cases.

CHAPTER V

Graph Techniques for Solving Differential Equations on a Quantum Annealer

In this work, the use of quantum annealer to solve differential equations is discussed. This is done by recasting a finite element model in the form of an Ising Hamiltonian. The discrete variables involved in the Ising model introduce complications when defining differential quantities, for instance, gradients involved in scientific computations of solid and fluid mechanics. To address this issue, a graph coloring based methodology is proposed which searches iteratively for solutions in a subspace of weak solutions defined over a graph, hereafter called the 'box algorithm.' The box algorithm is demonstrated by solving mechanics problems on the D-Wave quantum computer.

5.1 Introduction

While differential equations are ubiquitous in models of physical phenomena, the use of quantum annealers for scientific computing in solid and fluid mechanics has not yet been explored. Scientific computing mostly involves solving a linear system of equations $Ax = b$ defined on a continuum domain discretized with finite elements. The matrix A, generally being sparse, structured, and positive definite matrix obtained by assembling element-level stiffness matrices. In the past, gate–based quantum computing algorithms have been devised to solve the system of linear equations using QLSA algorithms (HHL algorithm [100]) and its variants [101, 102, 103, 104]. This algorithm, unlike a classical solver, does not give a direct solution x but rather allows sampling from the solution vector. Nevertheless, this has spawned several works in differential equation modeling on quantum computers ([105, 106, 107, 108, 109, 110, 111, 112, 113]). The sampling task by itself requires solving $Ax = b$. In the classical setting, the complexity scales with the size of the problem and goes as $O(Nsk\log(1/\epsilon))$ for conjugate gradient method where N is the number of unknowns, k is the condition number, s is the sparsity of A, and ϵ is the precision of the solution. On the other hand, the QLSA [100] has a favorable running time of $O(\log(N)k^2s^2/\epsilon)$ which scales logarithmically with the size of the problem. Quantum annealers are especially attractive for scientific computing with the ability to scale up the simulations to a more significant number of qubits. However, algorithms for the solution of differential equations have not been devised yet on these systems[114].

The solution to $Ax = b$ can be encoded in an equivalent minimization problem $min\left(\frac{1}{2}x^TAx - x^Tb\right)$ which contains field and interaction terms similar to an Ising model. In a similar way, some differential equations admit to an energy minimization formulation. In this work, the mapping of this energy to an Ising hamiltonian is explored. It is found that the idea of finite elements can be used to develop Ising Hamiltonians with sparse graphs, thus allowing embedding of larger problems on the NISQ-era hardware. An illustration of this procedure is presented in Fig 5.1. A discretized version of the differential equation is solved using energy minimization on a graph.

For the representation of a real-valued function, the qubits must encode a rational number. However, the qubit encoding the Ising lattice point carries two discrete levels (up/down spin) in the ground state. In classical computers, with similar binary (0/1)

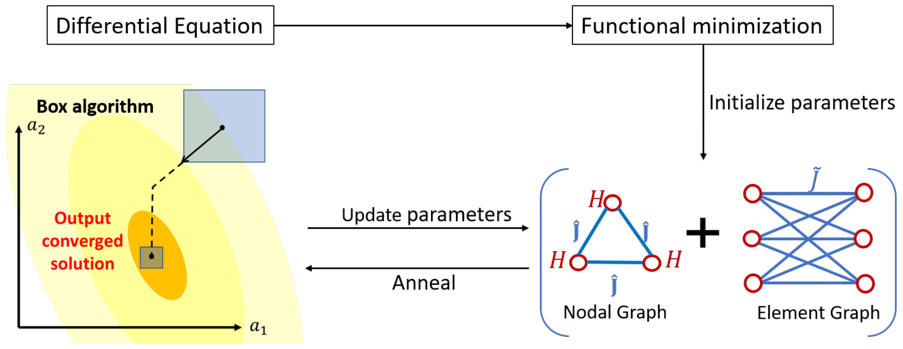


Figure 5.1: Illustration of procedure for solving differential equation

encoding, anywhere from 32 bits (float) to 80 bits (long double) of memory can be used to encode more than 12 million high precision variables in 1 GB memory. In contrast, currently available quantum annealers have a limited number of physical qubits. This restriction makes the representation of solutions of double-precision similar to a classical computer extremely expensive. In ([115, 116]), the problem of minimizing $||Ax-b||$ in the least-squares sense was posed by encoding physical qubits to represent rational numbers using a radix 2 representation. This format requires a significant number of physical qubits and connections to represent positive rational numbers and an additional qubit to represent the sign of the number ([115]). In comparison, the box algorithm searches within a small discrete set of up/down qubit values with each element of the set mapped to a double-precision value, thereby eliminating the need for additional qubits to achieve higher precision.

A self-adjoint form of a second-order differential equation is considered as the model problem. The problem statement and the relevant mathematical details are presented in Section 5.2. The Graph representation of the problem is formulated in Section 5.3. The iterative procedure, referred to as 'Box algorithm,' is presented in Section 5.4. All procedures are accompanied by numerical examples for elucidation. This algorithm is demonstrated by solving a truss mechanics problem on the D-Wave quantum computer in Section 5.5. Finally, this method is generalized to the Beam bending problem. It is shown that non-convexity in energy leads to incorrect solutions.

5.2 Mathematical Preliminaries

A self-adjoint form of a second order differential equation on an interval (x_l, x_r) is defined as,

$$-(p(x)u'(x))' + q(x)u(x) = f(x) \qquad x_l < x < x_r \tag{5.1}$$

Dirichlet boundary conditions are considered at both ends i.e. $u(x_l) = u_l$ and $u(x_r) = u_r$. Well-posedness of this problem requires $p(x) \geq p_{min} > 0$ and $q(x) \geq q_{min} \geq 0$ Furthermore, for convenience, it is assumed that $p, q \in C([x_l, x_r])$ and $f \in L^2([x_l, x_r])$. These conditions are sufficient to show the existence of a unique solution to the weak form ([117]).

5.2.1 Functional minimization

Motivated by the intractability of direct integration of the differential equation (5.1), it is often convenient to employ functional minimization techniques. Calculus of variations can be used to observe that the minimization of the functional (5.2) leads to the strong form described in Eq (5.1).

$$\Pi\left[u\right] = \int\limits_{x_l}^{x_r} \left(\frac{1}{2}pu'^2 + \frac{1}{2}qu^2 - fu\right) dx \tag{5.2}$$

Square integrability of u and its first derivative are required in this definition of $\Pi\left[u\right]$. The implication is that the minimizing solution, u lies in the Sobolov space $H^1([x_l, x_r])$. A discrete problem is obtained by using a finite basis for the solution defined in Eq (5.3), which satisfies the Dirichlet boundary conditions. The admissible choices of $\mathbf{a} = (a_0, a_1, ..., a_N)$ satisfy $u_N(x_d) = u_d$ where x_d is a Dirichlet boundary and u_d is the prescribed value at that point. This approximation reduces the infinite-dimensional functional minimization problem to finite dimensions. The approximated functional Π_N is entirely determined by the representation of u in the finite basis as shown in Eq (5.4). It is worth observing that the choice of $\phi_i(x)$ is such that $\phi_i \in H^1([x_l, x_r])$ i.e. for any $u_N \in V_N = span\{\phi_1, \phi_2, ..., \phi_N\} \subseteq H^1([x_l, x_r])$. Additionally the proper inclusion, $V_i \subseteq V_{i+1}$, guarantees convergence of the solution with increasing N.

$$u_N(x) = \sum_{i=0}^{N} a_i \phi_i(x) \tag{5.3}$$

$$\Pi_N\left[a_0, ..., a_r, .., a_N\right] = \int_{x_l}^{x_r} \frac{p}{2} \left(\sum_{i=0}^{N} a_i \phi_i'\right)^2 + \frac{q}{2} \left(\sum_{i=0}^{N} a_i \phi_i\right)^2 - f \left(\sum_{i=0}^{N} a_i \phi_i\right) dx \tag{5.4}$$

As the solution is completely determined by the variable $\mathbf{a}$, the functional minimization of Eq (5.4) is reformalized as Eq (5.5) where $\mathbf{a}^{b.a.}$ refers to the coefficients of best approximation of solution, u_N, in the subspace V_N

$$\mathbf{a}^{b.a.} = \arg\min_{\mathbf{a}} \Pi_N(\mathbf{a}) \tag{5.5}$$

5.2.2 Finite Element approximation

The finite element basis provides a popular choice of compactly supported shape functions. For the purpose of simplicity, 'tent/hat functions' (defined in Eq (5.6)) are used in this work. The domain is split into N elements with $N + 1$ nodes. The generalization to higher-order families of piecewise-continuous basis functions is immediate but is omitted for brevity.

$$\phi_i(x) = \begin{cases} 0, & x < x_{i-1}, \\ (x - x_{i-1})/(x_i - x_{i-1}), & x_{i-1} \leq x < x_i, \\ 1 - (x - x_i)/(x_{i+1} - x_i), & x_i \leq x < x_{i+1}, \\ 0, & x \geq x_{i+1}\,. \end{cases} \tag{5.6}$$

The usage of a compact basis further reduces the complexity by reducing the integration over the whole domain to a summation of integration over smaller elements. It is shown in section 5.3 that this choice of shape functions leads to a relatively sparse graph. It simplifies the computation by reducing the size of the graph optimization problem. The simplified form of Π specialized for the hat-functions is presented in Eq (5.7).

$$
\begin{aligned}
\Pi_N(\mathbf{a}) = \sum_{i=1}^{N} a_{i-1}^2 \left(\int_{x_{i-1}}^{x_i} \frac{p}{2}\phi_{i-1}'^2 + \frac{q}{2}\phi_{i-1}^2 dx \right) + a_i^2 \left(\int_{x_{i-1}}^{x_i} \frac{p}{2}\phi_i'^2 + \frac{q}{2}\phi_i^2 dx \right) \\
+ a_{i-1}a_i \left(\int_{x_{i-1}}^{x_i} p\phi_{i-1}'\phi_i' + q\phi_{i-1}\phi_i dx \right) - a_{i-1}\left(\int_{x_{i-1}}^{x_i} f\phi_{i-1} dx \right) - a_i \left(\int_{x_{i-1}}^{x_i} f\phi_i dx \right)
\end{aligned}
\tag{5.7}
$$

This form of Π promotes modularity in computation and allows expressing the functional as

$$
\Pi_N = \sum_{i=1}^{N} \mathbf{A_i}.\mathbf{S_i} \tag{5.8}
$$

where vectors $\mathbf{A_i} \equiv \mathbf{A_i}(a_{i-1}, a_i)$ and $\mathbf{S_i} \equiv \mathbf{S_i}(p, q, f)$ are defined for each element in (5.9). The vector $\mathbf{S_i}$ is independent of state $\mathbf{a}$ and is therefore only computed once in the whole procedure.

$$
\begin{aligned}
\mathbf{A_i} &= \begin{bmatrix} a_{i-1}^2 & , & a_i^2 & , & a_{i-1}a_i & , & a_{i-1} & , & a_i \end{bmatrix}^T \\
\mathbf{S_i} &= \Bigg[\int_{x_{i-1}}^{x_i} \frac{p}{2}\phi_{i-1}'^2 + \frac{q}{2}\phi_{i-1}^2 dx \quad , \quad \int_{x_{i-1}}^{x_i} \frac{p}{2}\phi_i'^2 + \frac{q}{2}\phi_i^2 dx \quad , \\
&\quad \int_{x_{i-1}}^{x_i} p\phi_{i-1}'\phi_i' + q\phi_{i-1}\phi_i dx \quad , \quad -\int_{x_{i-1}}^{x_i} f\phi_{i-1} dx \quad , \\
&\quad -\int_{x_{i-1}}^{x_i} f\phi_i dx \Bigg]^T
\end{aligned}
\tag{5.9}
$$

Example

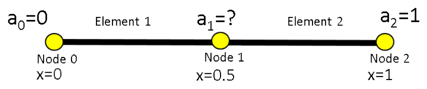
Consider the differential equation with boundary conditions $u(0) = 0$ and $u(1) = 1$.

$$\frac{d^2u}{dx^2} = 0 \qquad 0 < x < 1$$

Functional:

$$\Pi[u] = \frac{1}{2}\int_0^1 u'^2 dx$$

For simplicity, consider a grid with a uniform mesh of 2 elements and 3 nodes:



Using linear interpolants for the elements,

$$u(x) = \begin{cases} a_0(1-2x) + a_1(2x) & 0 < x \leq 0.5 \\ \\ a_1(2-2x) + a_2(2x-1) & 0.5 < x \leq 1 \end{cases}$$

The functional with the FE discretization:

$$\Pi_N(\mathbf{a}) = (a_0 - a_1)^2 + (a_1 - a_2)^2$$

Modular representation of functional ($\Pi_N = \mathbf{A_1}.\mathbf{S_1} + \mathbf{A_2}.\mathbf{S_2}$):

$$\mathbf{A_1} = \begin{bmatrix} a_0^2 & , & a_1^2 & , & a_0a_1 & , & a_0 & , & a_1 \end{bmatrix}^T$$
$$\mathbf{A_2} = \begin{bmatrix} a_1^2 & , & a_2^2 & , & a_1a_2 & , & a_1 & , & a_2 \end{bmatrix}^T$$

$$\mathbf{S_1} = \mathbf{S_2} = \begin{bmatrix} 1 & , & 1 & , & -2 & , & 0 & , & 0 \end{bmatrix}^T$$

5.3 Graph Coloring Problem

Quantum annealing methods are tailored to find the lowest energy states in an Ising system defined in Eq (1.12). The Ising hamiltonian defines a binary graph coloring problem with each vertex of graph or qubit labeled as $+1$ or -1. The value of the qubits determine the free variable, in this case, $\mathbf{a}$. The parameters H_i and J_{ij} are defined such that the Ising hamiltonian, for labeling representing the state, $\mathbf{a}$, corresponds to the functional $\Pi_N(\mathbf{a})$. These problems, namely, the representation of state and estimation of parameters, are addressed in this section.

5.3.1 Representation of State

Representation of a functional in terms of continuous variables is not feasible on quantum architectures. Due to this limitation, the values of each a_i (i^{th} component of $\mathbf{a}$) are chosen from a finite set of values based on the labeling of qubits. The representation presented here permits 3 possible values of a_i at each node. In particular, for each node (indexed 'i'), the state (a_i) is exactly determined by the labeling of qubits q_1^i, q_2^i and q_3^i with the i^{th} node taking values in the set $\{v_{i_1}, v_{i_2}, v_{i_3}\}$. Eq (5.10) defines a mapping $(q_1^i, q_2^i, q_3^i) \rightarrow a_i$ as tabulated in Table 5.1. It is observed that the mapping results in $a_i \in \{v_{i_1}, v_{i_2}, v_{i_3}\}$ when two qubits are labeled -1 and one qubit is labeled $+1$. Next, it is shown that the Ising parameters can be manipulated to make these labelings energetically favourable, thereby eliminating the occurrence of undesirable labels.

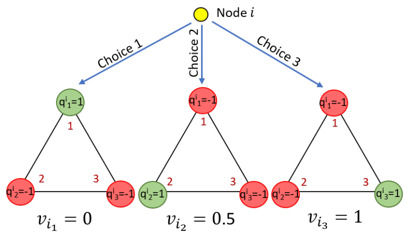
$$a_i = \sum_{j=1}^{3} v_{i_j} \frac{q_j^i + 1}{2} \tag{5.10}$$

(q^i_1, q^i_2, q^i_3)	a_i
$(1, 1, 1)$	$v_{i_1} + v_{i_2} + v_{i_3}$
$(1, 1, -1)$	$v_{i_1} + v_{i_2}$
$(1, -1, 1)$	$v_{i_1} + v_{i_3}$
$(1, -1, -1)$	v_{i_1}
$(-1, 1, 1)$	$v_{i_2} + v_{i_3}$
$(-1, 1, -1)$	v_{i_2}
$(-1, -1, 1)$	v_{i_3}
$(-1, -1, -1)$	0

Table 5.1: The mapping from qubits to state a_i at node

Example (Continued)

In general, the set $\{v_{i_1}, v_{i_2}, v_{i_3}\}$ is different for each node. However, for simplicity, consider the same set of admissible states for all three nodes given by $\{v_{i_1}, v_{i_2}, v_{i_3}\} \equiv \{0, 0.5, 1\}$. Each node is defined by three qubits as follows:



The three qubits, each defining the solution at the first and last nodes, should take up choices 1 and 3, respectively, due to boundary conditions. The choice for the second node is to be solved.

5.3.2 Parameter Estimation

To promote modularity, the graph representation is decomposed into two component subgraphs, namely, nodal graph and element graph. Each node and element of the FE discretization is endowed with a node graph and element graph, respectively. This allows refining the mesh by extending the graph.

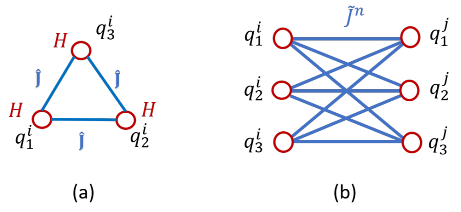


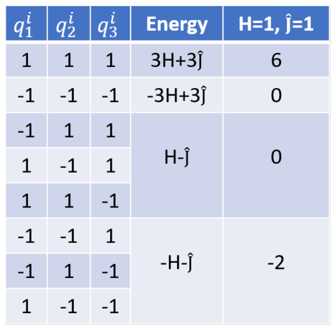
Figure 5.2: Connectivity of (a) nodal graph (b) element graph.

5.3.2.1 Nodal Graph

The nodal graph is given by a fully connected graph with three vertices representing the three qubits of the FE node. The nodal graph ensures that the energy minimizing states of the Ising hamiltonian corresponds to state **a** with favorable choice of $a_i \in \{v_{i_1}, v_{i_2}, v_{i_3}\}$ with equal probability. As mentioned earlier, the set of favorable labeling of qubits at a node is given by $\{Q_1, Q_2, Q_3\} \equiv \{(1,-1,-1),(-1,1,-1),(-1,-1,1)\}$. Since each of the three labelings is equally likely in the absence of any functional minimization, it is expected that the same value of the coupling strength $(\hat{J})$ for each connection and the field strength (H) for each node is used. A choice of $\hat{J}$ and H that fulfill these conditions is presented in Fig 5.3. Here, all the field and interaction terms for the nodal graph are given a value of one. The application of the Dirichlet boundary condition is also made by augmenting the field strength of the nodal graph. For example, by switching the field term H corresponding to the second qubit q_2 of a boundary node 'b' to -1 forces a lower value of the functional for the boundary node state of $(-1,+1,-1)$, which corresponds to the solution v_{b_2}. This allows us to encode the value at the boundary to be v_{b_2}.

5.3.2.2 Element Graph

The element graph is used to make the energy of minimizing states of graph correspond to the value of functional Π_N of the continuous problem. Each element graph encodes the contribution of the respective element to the functional. Since the contribution of each element is dependent on the values at the nodes of the element,



q_1^i	q_2^i	q_3^i	Energy	H=1, Ĵ=1
1	1	1	3H+3Ĵ	6
-1	-1	-1	-3H+3Ĵ	0
-1	1	1	H-Ĵ	0
1	-1	1		
1	1	-1		
-1	-1	1	-H-Ĵ	-2
-1	1	-1		
1	-1	-1		

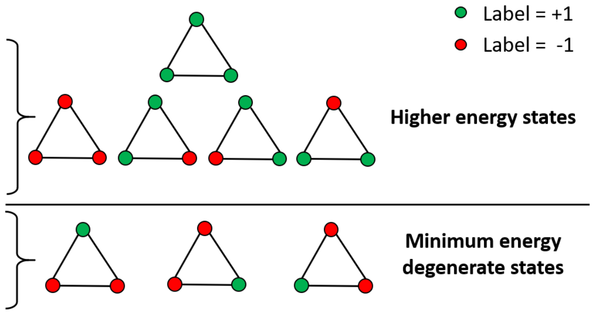


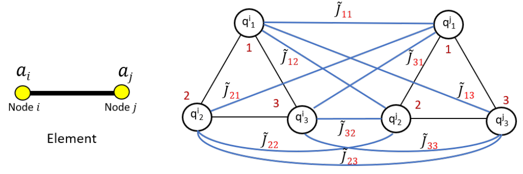
Figure 5.3: Self interaction and site-site interaction parameters for nodal subgraph.

the element graph is constructed by connecting the vertices of neighboring nodes. In particular, the site-site interaction in the n^{th} element graph can be estimated as a matrix, $\widetilde{J}^n$, where $(\widetilde{J}^n)_{kl}$ represents the coupling energy between qubits q_k^i (k^{th} qubit of i^{th} node) and q_l^j (l^{th} qubit of j^{th} node) with i, j being the nodes of n^{th} element. As shown in the previous section, the contribution of the n^{th} element towards the functional, based on the choice of a compact basis function, is evaluated as $\mathbf{A_n}.\mathbf{S_n}$. The elements of the vector, $\mathbf{A_n} \equiv \mathbf{A_n}(a_i, a_j)$ can therefore, take nine (3×3) possible values based on the values of (a_i, a_j). For a particular choice of labeling of qubit the Ising energy of element graph is estimated as $E = \sum_{k=1}^{3} \sum_{l=1}^{3} (\widetilde{J}^n)_{kl} q_k^i q_l^j$. When the labeling is chosen appropriately (each node has two '-1' and one '+1' label), this energy equals to the value of functional for the corresponding state, $\mathbf{a}$, as shown in Eq (5.11). This relation can be used to estimate $\widetilde{J}^n$ by solving a set of nine independent linear equations presented. It is important observation that the independence of these set of equations relies on the fact that for any node, $v_{i_k} \neq v_{i_l}$ for $k \neq l$. Additionally, the energy of the element graph breaks the symmetry between the states that minimize the energy of a nodal graph. However, the values of $\widetilde{J}^n$ should be judiciously scaled (uniformly along elements) such that the energy of unfavorable states remains high.

$$\sum_{k=1}^{3} \sum_{l=1}^{3} (\widetilde{J}^n)_{kl} q_k^i q_l^j = \mathbf{A_n}(a_i, a_j).\mathbf{S_n} \tag{5.11}$$

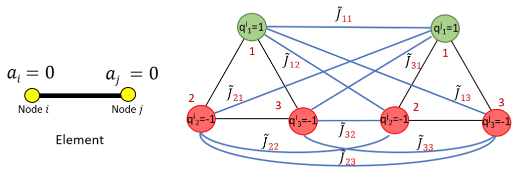
Example (Continued)

A single element with two nodes admits to the following connectivity:



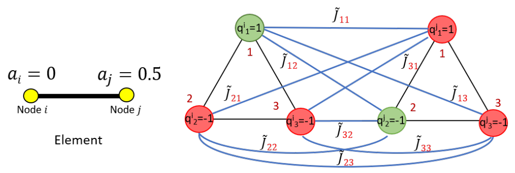
The estimated parameters reflect the contribution of element to the functional for a given choice of labeling:

Sample 1:



In the above Figure, both nodes take up choice 1 ($a_i = a_j = 0$). The interaction energy for qubits: $E = \widetilde{J}_{11} - \widetilde{J}_{12} - \widetilde{J}_{13} - \widetilde{J}_{21} + \widetilde{J}_{22} + \widetilde{J}_{23} - \widetilde{J}_{31} + \widetilde{J}_{32} + \widetilde{J}_{33} = (a_i - a_j)^2 = 0$

Sample 2:



In the above Figure, node i takes up choice 1 ($a_i = 0$), while node j takes up choice 2 ($a_j = 0.5$). The interaction energy for qubits:

$$E = -\widetilde{J}_{11} + \widetilde{J}_{12} + \widetilde{J}_{13} + \widetilde{J}_{21} - \widetilde{J}_{22} - \widetilde{J}_{23} - \widetilde{J}_{31} + \widetilde{J}_{32} + \widetilde{J}_{33} = (a_i - a_j)^2 = 0.25$$

Collectively solving the equation for all 9 such possibilities (as shown in Eq (5.11)).

$$\begin{bmatrix} +1 & -1 & -1 & -1 & +1 & +1 & -1 & +1 & +1 \\ -1 & +1 & +1 & +1 & -1 & -1 & -1 & +1 & +1 \\ -1 & +1 & +1 & -1 & +1 & +1 & +1 & -1 & -1 \\ -1 & +1 & -1 & +1 & -1 & +1 & +1 & -1 & +1 \\ +1 & -1 & +1 & -1 & +1 & -1 & +1 & -1 & +1 \\ +1 & -1 & +1 & +1 & -1 & +1 & -1 & +1 & -1 \\ -1 & -1 & +1 & +1 & +1 & -1 & +1 & +1 & -1 \\ +1 & +1 & -1 & -1 & -1 & +1 & +1 & +1 & -1 \\ +1 & +1 & -1 & +1 & +1 & -1 & -1 & -1 & +1 \end{bmatrix} \begin{bmatrix} \widetilde{J}^n_{11} \\ \widetilde{J}^n_{12} \\ \widetilde{J}^n_{13} \\ \widetilde{J}^n_{21} \\ \widetilde{J}^n_{22} \\ \widetilde{J}^n_{23} \\ \widetilde{J}^n_{31} \\ \widetilde{J}^n_{32} \\ \widetilde{J}^n_{33} \end{bmatrix} = \begin{bmatrix} (v_{i_1} - v_{j_1})^2 \\ (v_{i_2} - v_{j_1})^2 \\ (v_{i_3} - v_{j_1})^2 \\ (v_{i_1} - v_{j_2})^2 \\ (v_{i_2} - v_{j_2})^2 \\ (v_{i_3} - v_{j_2})^2 \\ (v_{i_1} - v_{j_3})^2 \\ (v_{i_2} - v_{j_3})^2 \\ (v_{i_3} - v_{j_3})^2 \end{bmatrix}$$

$$\widetilde{J}^1 = \widetilde{J}^2 = \begin{bmatrix} 0.1250 & 0.3750 & 0.3750 \\ 0.3750 & 0.5000 & 0.3750 \\ 0.3750 & 0.3750 & 0.1250 \end{bmatrix}$$

The above parameters will exactly reproduce the functional in the interaction term. The boundary conditions are enforced, by setting self interaction term for qubits q_1^0, q_3^2 to $H = -1$. This locks the state at the 1st boundary node as $a_0 = v_{0_1} = 0$ and at the 2nd boundary node as $a_2 = v_{2_3} = 1$. Energy minimization of the resulting Ising hamiltonian gives $a_1 = v_{1_2} = 0.5$, which is the exact solution for the discretized problem.

The process of the graphical representation of the discretized functional using the nodal and element graphs is referred to as "Assembly". Each node and element is endowed with a nodal and an element graph, respectively. The effective site-site interaction energy is estimated by summing the nodal coupling strength, $\hat{J}$, over all nodes, and element coupling strength, $\widetilde{J}$, over all elements. Due to the nature of discretization, N element graphs and $N+1$ nodal graphs are required for representing an N-element discretization of the domain. The assembled graph, from here on, is referred to as the logical graph. Connectivity of logical graphs for one-element and four-element discretization is presented in Fig 5.4.

Two fundamental issues in this approach are addressed next using the box algorithm. Firstly, the choices at a node $\{v_{i_1}, v_{i_2}, v_{i_3}\}$ were set in stone during initialization. The box algorithm makes this choice flexible. Secondly, as the number of nodes increase, three choices are insufficient. The number of qubits needed at a node must increase to make more choices available. Box algorithm, however, only requires three qubits per node for any level of discretization.

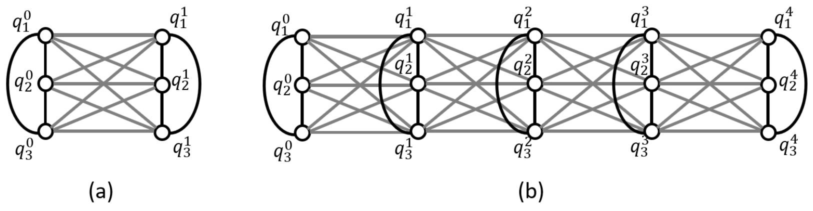


Figure 5.4: Assembled graph for a domain discretized with (a) 1 element (b) 4 elements.

5.4 Box Algorithm

In this section, an iterative procedure is developed to minimize the functional, Π_N, using the graph coloring representation discussed in the previous section. For a particular choice of $\{v_{i_1}, v_{i_2}, v_{i_3}\}$, defined as Eq (5.12), the possible values of the state a_i at the i^{th} node are specialized to the set $\{u_i^c - r, u_i^c, u_i^c + r\}$, i.e.,

$$v_{i_j} = u_i^c + r(j-2) \tag{5.12}$$

The quantities, $\mathbf{u^c} = (u_0^c, u_2^c, ..., u_N^c)$ and r are hereafter referred to as box center and the slack variable, respectively. The intention is to approximate functions using the box center, while the slack variable provides a bound on this approximation. The precise meaning of this bound is presented later in this section. A linear approximation of $f(x) = \sqrt{x}$ using ten nodes is presented in Fig 5.5 for different box centers and the slack variable (which can be interpreted as the box size). The function, $f(x)$, is approximated as $\mathbf{u^c}$ at the nodes with linear interpolation in between the nodes. The blue region describes the possible value of the interpolation if the value at any node is perturbed within the range of $\pm r$. In Fig 5.5(a), an exact approximation of the function at the nodes is presented with a slack variable of 0.2. In (b), the same approximation with a slack variable of 0.02 is presented. The same approximation is given in the two cases, but the bound on nodal values of (b) is tighter than (a). In part (c), the approximation is not exact; however, it lies within the bounds. In part (d), the approximation is neither exact nor within the bound. In the context of the vectorial representation of the coefficients, $\mathbf{a}$, these bounds are represented as $3^N - 1$ points on the surface of a box, defined as, $||\mathbf{a} - \mathbf{u^c}||_\infty = r$. An illustration for the vectorial representation of a two nodes element is presented in Fig 5.6. The solution is sampled from a 3×3 grid in the $a_1 - a_2$ vector space.

In the discrete setting, the solution to the differential equation can be equivalently reduced to minimization of a function of the form: $\mathbf{a^T M a}$ where $\mathbf{M}$ is some positive definite matrix. The vector $\mathbf{a}$ takes value in one of the 3^N possibilities. The minimizer (not necessarily unique) is given by Eq (5.13). The solution, $\mathbf{a}^{min}$, need not coincide with the best approximation solution, $\mathbf{a}^{b.a.}$ of the continuous problem. In the illustration presented in Fig 5.6, the center is depicted as the solution ($\mathbf{a}^{min} = \mathbf{u}^c$), the minimum is then contained within the elliptic region of the contour with $\mathbf{a}^{min}$ on the edge. Geometrically, this gives $||\mathbf{a}^{min} - \mathbf{a}^{b.a}|| \leq d \leq \sqrt{2}r(1 + \lambda_{max}/\lambda_{min})$

where λ_{max} and λ_{min} are the maximum and the minimum non–zero eigenvalues of M, respectively. This suggests that as the box size decreases, the corresponding $\mathbf{u}^c$ approaches to the best approximation solution of u. This argument is extended to n dimensions with the bound given in Eq (5.14).

$$\mathbf{a}^{min} = \underset{a_i \in \{u_i^c - r, u_i^c, u_i^c + r\}}{\arg\min} \quad \mathbf{a}^{\mathbf{T}}\mathbf{M}\mathbf{a} \tag{5.13}$$

$$||\mathbf{a}^{min} - \mathbf{a}^{b.a}|| \leq 2\left(1 + (n-1)\frac{\lambda_{max}}{\lambda_{min}}\right)\frac{r}{\sqrt{n}} \tag{5.14}$$

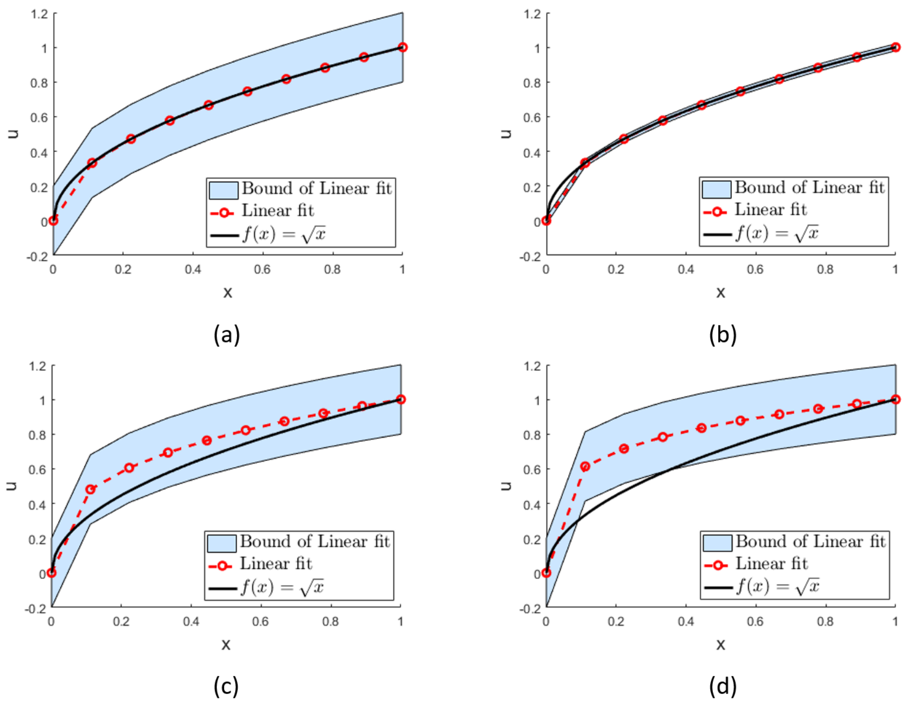


Figure 5.5: Approximation of $\sqrt{x}$ function using boxed domain: (a) Exact fit with a slack of 0.2 (Loose fit) (b) Exact fit with a slack of 0.02 (tight fit) (c) inexact fit but bounded in a box size of 0.2 (d) inexact fit and unbounded by a slack of 0.2.

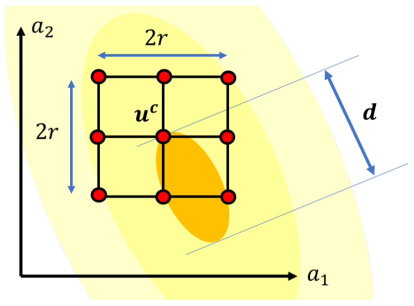


Figure 5.6: An illustration of a two-node approximation in $a_1 - a_2$ vector space with contours plot of the functional, $\Pi_2(a_1, a_2)$ and a representative box with center at $\mathbf{u^c}$ and a box size of r.

5.4.1 Iterative Procedure

In this section, the details of the iterative procedure are presented. It estimates the solution of the discretized problem, $\mathbf{a}^{min}$, and updates the box center and slack variable such that $\mathbf{u}^C$ approaches the solution of the continuous problem (in the sense of best approximation).

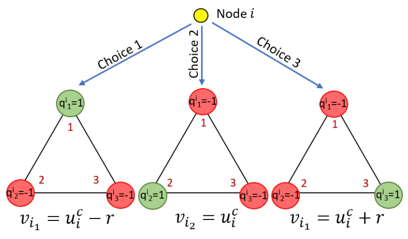
The necessary information required for defining the functional is stored in the vector $\mathbf{S_i}$. It is computed once at the beginning of the procedure as the problem definition stage. The procedure is initiated with a guessed solution of the vector, $\mathbf{a}$, provided as a box centered at $\mathbf{u^c}$. The boundary nodes with Dirichlet boundary conditions are assigned the boundary value as the initial guess. The slack variable is initialized with an arbitrary scalar value. A better initial guess for r is the one which bounds the solution in the box defined by $\mathbf{u^c}$. Such initial guesses require fewer iterations in comparison to arbitrary ones; however, starting with a good guess is not a necessary condition for convergence. The Ising parameters, H, $\hat{J}$ and $\widetilde{J}$ are estimated as discussed in section 5.3.

In this work, D-Wave's 2000Q processor is used. This processor has a Chimera-type structure with 2048 qubits, and 6016 couplers [118]. A direct solution of the optimization problem by re-numeration of qubits is not possible as the assembled graph cannot be found in any subgraph of the physical graph, i.e., the processor. Therefore, it is required that the logical graph is mapped onto the physical graph via the process of embedding. This problem in itself is NP-hard and is not discussed here for brevity. The reader is referred to [119] for a discussion on this topic. The topology of the logical graph remains unchanged over the iterations. The search for the embedding of a map is only conducted once, and in subsequent iterations, the self-interaction and the site-site interaction parameters are updated for the same embedding.

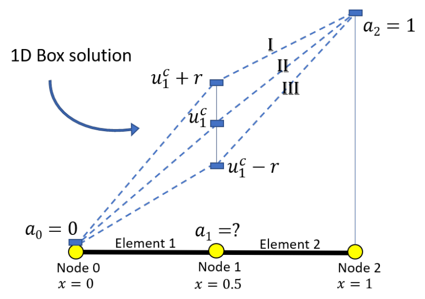
The use of three qubits per node allows the D-wave system to search for a minimum over a space of 3^N solution vectors in a single run. In each iteration, the box center is translated to the energy minimizer, $\mathbf{a}^{min}$. This move is referred to as the translation step. In the case where the minimizing state is found at the center, the box size is reduced, and the search is continued with a smaller bound on the error. This move is referred to as the contraction step. The complete procedure is presented in Algorithm 10.

Example (Continued)

In the box algorithm, the set $\{v_{i_1}, v_{i_2}, v_{i_3}\}$ is constructed using the box center and the slack variable.



With the application of the boundary condition, the favourable labeling of qubits give following three choices in the solution (**I**, **II**, **III**).



The values of u_1^c and r are initialized arbitrarily. One of the solutions among **I**, **II** or **III** is selected by the annealer. If the minimizer is found to be solution **II** then the algorithm proceeds with the contraction step by halving the value of r. If solutions **I** or **III** are selected, then the algorithm proceeds with the translation step by setting the new box center to $u_1^c + r$ or $u_1^c - r$, respectively.

5.5 Results

The deformation of a bar under axial loading is modelled using an equation of form (5.1). In particular, the deflection (u) of a bar is related to the elastic stiffness, (E), cross-sectional area (A), and the applied body force, (f) using Eq (5.15). The functional (Eq 5.2) is referred to as the potential energy of the system. The corresponding discretized form of the potential energy for piece-wise linear E,A and f is calculated using Eq (5.16) where E_i,A_i and f_i represents the elastic stiffness, area and applied body force, respectively, at the center of the i^{th} element.

$$(EAu')' + f = 0 \tag{5.15}$$

$$\Pi_N\left[u\right] = \sum_{i=1}^{N} \frac{N}{2} E_i A_i (a_i - a_{i-1})^2 - \frac{1}{2N} f_i (a_i + a_{i-1}) \tag{5.16}$$

Two test cases are presented in Fig 5.7. In test case (a), a bar with a discontinuity

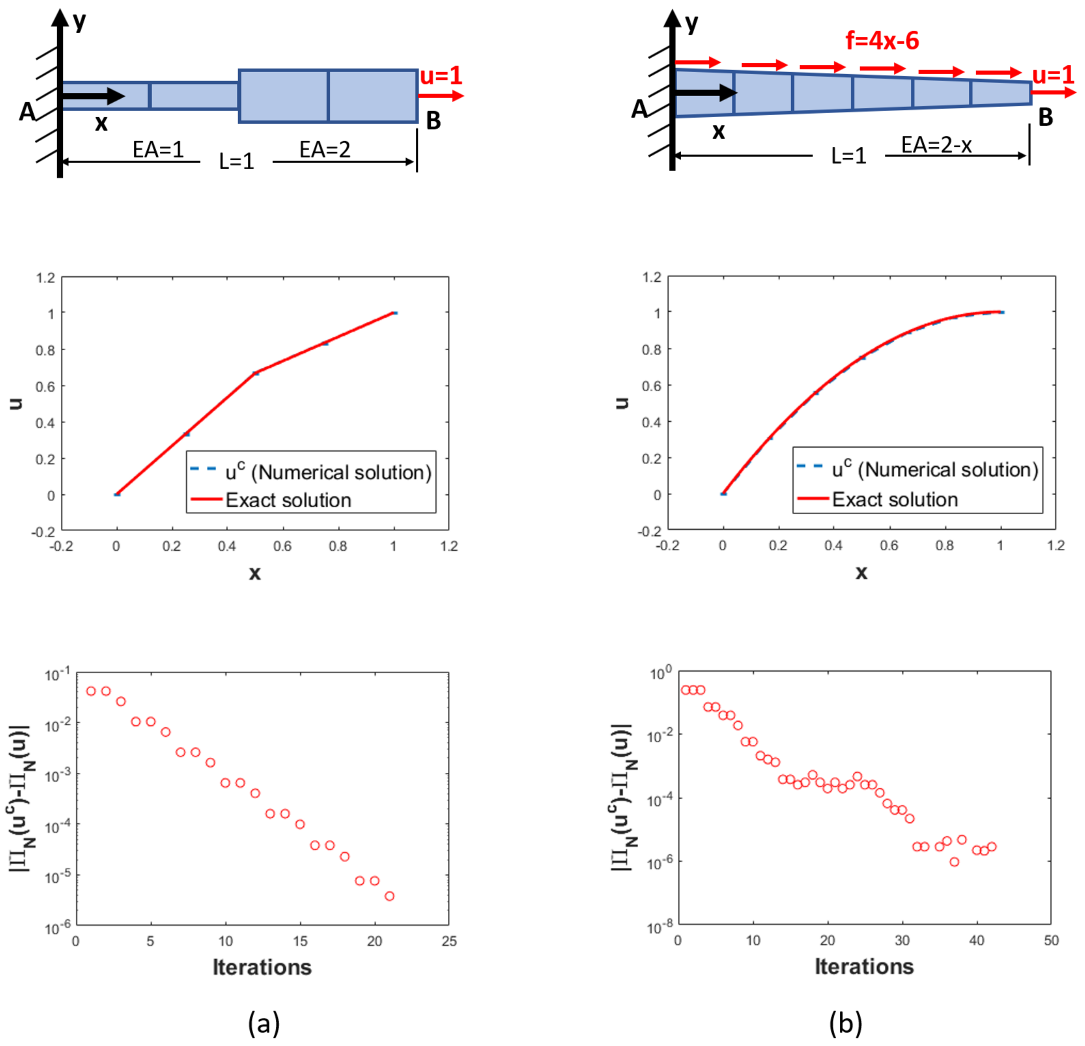


Figure 5.7: Axial deformation of a bar with (a) a discontinuous cross-section with a tip displacement (b) a continuously varying cross section with a body force and a tip displacement.

in EA is simulated. The body force is not applied in this case. A four-element discretization is used. The initial guesses are taken as $\mathbf{u^c} = \{0, 0.25, 0.5, 0.75, 1\}$ and $r = 0.2$. The numerical solution is observed to approach the exact solution in this case. The convergence in the functional is also evident.

In the test case (b), a bar with continuously varying EA is simulated. A linearly varying body-force is supplied. A six-element discretization of the bar is used with $\mathbf{u^c} = \{0, \frac{1}{6}, \frac{2}{6}, \frac{3}{6}, \frac{4}{6}, \frac{5}{6}, 1\}$ and $r = 0.2$. Based on the theory of finite element methods, the exact minimization of the energy in discretized space leads to a stiffer solution in comparison to the exact solution. It is observed that the numerical solution approaches the exact solution at nodes, which is characteristic of finite element methods. Energy is also observed to be converging towards the finite element solution u^{fem} in this case. The mismatch of u within the element is expected to decrease with refinement in discretization.

Some implementation details on the D-wave architecture are relevant here. Although the mapping only requires three qubits per node, the embedding of this graph into the Chimera graph produced an overhead of 9 qubits per node - constant over a range of discretizations. It is understandable since a complete graph of three qubits used to represent a node is not directly represented on the Chimera graph. Another important task in quantum computing is error suppression. Quantum processors, unlike classical computers, do not have parity correction algorithms due to the no-cloning theorem. A compilation of popular methods for quantum error correction is presented in [120]. Energy rescaling is one of the simpler approaches and is employed in this work. Here, in the estimation of $\widetilde{J}^n$, the energy was rescaled to ensure that the energy gap between feasible and unfeasible states is increased while maintaining a similar energy landscape. This step is a heuristic remedy for minimizing noise in quantum computation and has no bearing in the theoretical convergence of the algorithm.

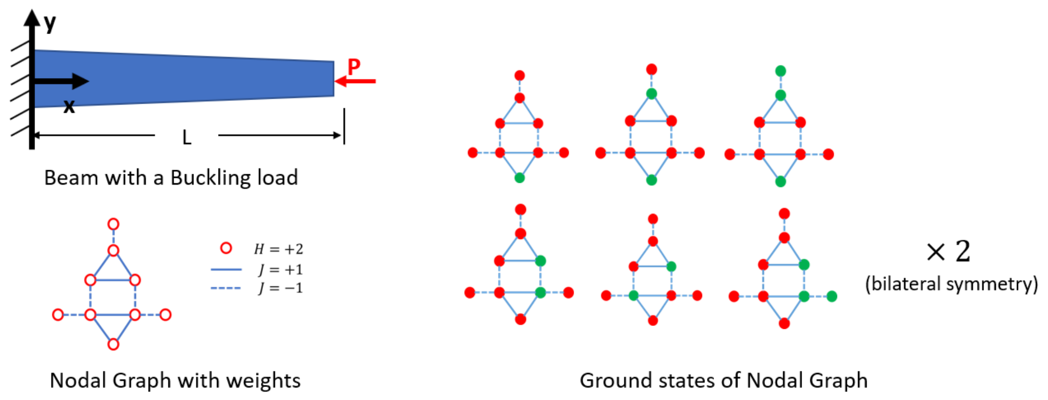


Figure 5.8: Illustration of beam bending problem with buckling load and the nodal graph with respective ground states

5.5.1 Generalization to Beam bending problem

In this section, Box algorithm for Beam bending problem is discussed. Beam bending is modeled as a 4^{th}-order differential equation. A special case is considered with a buckling load and no distributed load (as illustrated in the Fig 5.8). The differential equation can be stated in terms of the vertical displacement, w as:

$$EIw'''' + Pw'' = 0, \qquad 0 < x < L$$

Each boundary ($x = x_b$) has following two boundary conditions:

1. Either the displacement, $w(x_b)$, or the shear force, $V(x_b) = -EIw'''(x_b)$ is prescribed.

2. Either the slope, $w'(x_b)$, or the moment, $M(x_b) = -EIw''(x_b)$ is prescribed.

Considering the case with Dirichlet boundary conditions and/or homogeneous Neumann conditions (the first condition in each of the boundary condition listed above), the energy is estimated as:

$$\Pi[w] = \frac{1}{2}\int_0^L EI(w'')^2 - P(w')^2 dx$$

In this case, the Finite element approximation is required to maintain the continuity of slopes at the nodes. It can be accomplished by considering Hermite cubic polynomials. Thus each node has 2 degrees of freedom, the displacement, w, and the slope, $\theta \equiv w'$. In this case, a Nodal graph with 9 ground states is required, where each of the ground states is mapped to one of the following states:

$$\{w_c^i - r, w_c^i, w_c^i + r\} \times \{\theta_c^i - r, \theta_c^i, \theta_c^i + r\}$$

where the notation of previous sections are adopted for box center and slack variable. The $\{.\} \times \{.\}$ represents cartesian product of sets. A nodal graph with 9 nodes can be employed, as shown in Fig 5.8. The element graph can be formed by connecting all 81 edges between the two adjacent nodal graphs. Moreover, it can be verified that the system of equations formed by relating the 81 possible energies of the element to the 81 interaction parameters is exactly solvable. Therefore, the box algorithm can be applied to the beam problem. In general, these nodal graphs can be generated by using the MILP approach presented in Appendix C. However, the rank of the linear system for the element matrix should be verified. In case there is a rank deficiency, then a different set of ground states need to be chosen.

When P is greater than the critical value, P_{cr} (buckling load for the first mode), then the total energy is non-convex, and this may inhibit the algorithm from minimizing the energy. As an example, consider a two-element discretization of the above problem with pin joints in each end (see Fig 5.9). These boundary conditions can be modeled by considering zero displacement and moments at the ends. In the first buckling mode, there are only two degrees of freedom: (i) the displacement of the middle node and (2) the slope of the end node. It can be seen in Fig 5.9, that if the box center is initially at the origin, then for any slack size, the energy at all the box corners is higher than the origin. Thus the optimization step misses out on the minimizing values (marked as the red region in the figure). This results in an incorrect solution where the beam does not buckle at the critical load. The issue can be alleviated by using different slack variables for each degree of freedom and use of finer increments in the contraction step.

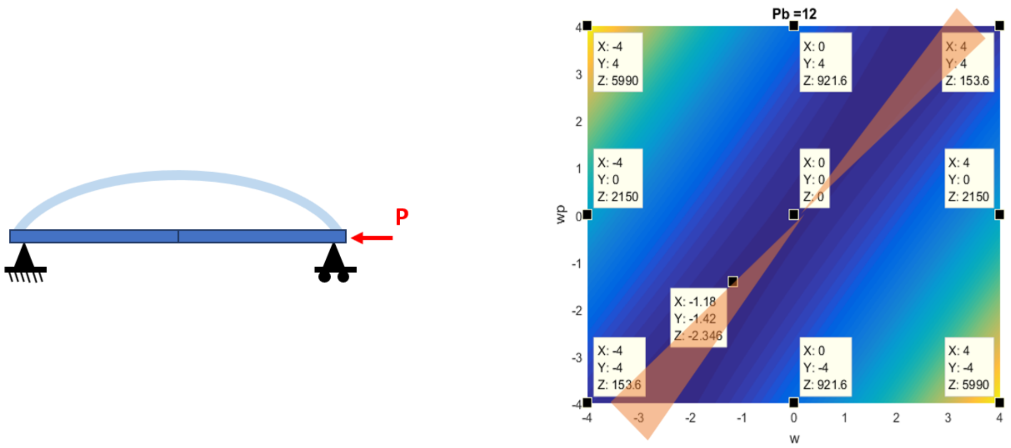


Figure 5.9: The problem of non-convexity in Box algorithm: (left) A two-element discretization of a beam buckling problem, and (right) the energies for displacement functions at $P = P_{cr}$ with x-axis representing the displacement at center of beam and y-axis representing the slope at the corners. The energy decreases along the red region in the direction away from origin.

5.6 Conclusions and Future Work

The use of quantum computing for solving differential equations has, to date, focused on the use of a gate–computing based QLSA algorithm. This algorithm attempts to sample from the solution space of the linear system of equations $Ax = b$. In contrast, the quantum annealer based algorithm described here uses an energy minimization approach. The discretized version of the energy function of the differential equation is mapped to an Ising Hamiltonian. The solution vector, x, is directly obtained as the ground state of the qubits. The algorithm has low memory requirements since the global matrix is not stored, and the local matrices are encoded in the interaction weights of the Ising model. Further, the box algorithm allows mapping of up/down spin states of qubits in the ground state to rational numbers involved in the solution vector.

Since primarily the Ising model is solved, the cost of computation is tied to the performance of the quantum annealer ([121]). Within each iteration, however, Eq (5.11) is solved for each element, leading to at least $O(n)$ operations. It is shown that the box algorithm indeed guarantees convergence to the best approximation of the solution in the discretized space as the box size goes to zero. However, some improvements could be made to reduce the number of minimization runs. The statistics from the sampled data can be used to heuristically develop 'local' potential energy maps that can be used to guide the gradient descent for faster convergence. With future scaling up of quantum annealers up to millions of qubits, it will be possible to solve challenging engineering solid and fluid mechanics problems using this method.

CHAPTER VI

Graph Models for Machine Learning: Boltzmann Machine Implementation on a Quantum Annealer

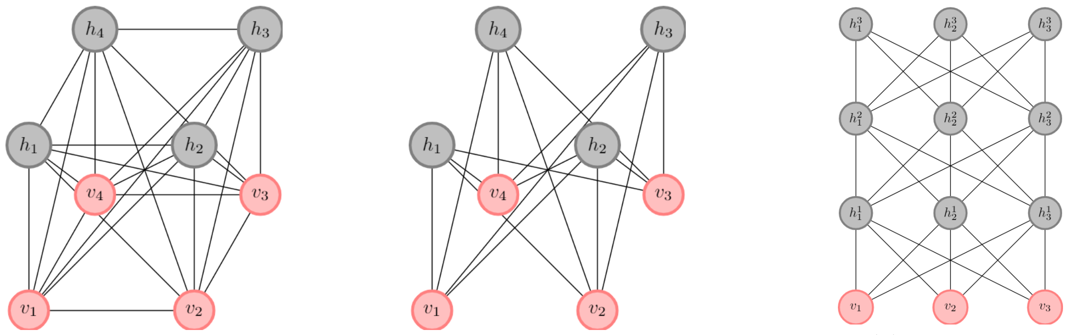
A hybrid quantum-classical method for learning Boltzmann machines (BM) for a generative and discriminative task is presented. Boltzmann machines are undirected graphs with a network of visible and hidden nodes where the former is used as the reading site while the latter is used to manipulate visible states' probability. In Generative BM, the samples of visible data imitate the probability distribution of a given data set. In contrast, the visible sites of discriminative BM are treated as Input/Output (I/O) reading sites where the conditional probability of output state is optimized for a given set of input states. The cost function for learning BM is defined as a weighted sum of Kullback–Leibler (KL) divergence and Negative conditional Log-likelihood (NCLL), adjusted using a hyper-parameter. Here, the KL Divergence is the cost for generative learning, and NCLL is the cost for discriminative learning. A Stochastic Newton-Raphson optimization scheme is presented. The gradients and the Hessians are approximated using direct samples of BM obtained through Quantum annealing (QA). Quantum annealers are hardware representing the physics of the Ising model that operates on low but finite temperature. This temperature affects the probability distribution of the BM; however, its value is unknown apriori. The effect of this unknown temperature on learning is also investigated.

6.1 Background

Boltzmann machine (BM) is an energy-based model defined on an undirected graph and is used for unsupervised learning. The graph vertices are segregated into a set of visible and hidden nodes. The probability of each state is dependent on the total energy of the graph for that state. Moreover, only the state of a visible node is "visible" to the user. Therefore, these visible states' marginalized probabilities are a non-linear function of the energy parameters and can be used to model complicated distributions. These BMs can be trained either using Maximum-likelihood (ML) learning or a Contrastive Divergence (CD) learning techniques. It is well known that ML Learning of Markov random fields (MRF) is a challenging task due to the large state space. Due to this complexity, Markov chain Monte Carlo (MCMC) methods typically take a long time to converge on unbiased estimates. CD learning, on the other hand, provides a computationally inexpensive way of training MRFs. However, it provides biased estimates in general [122].

A subclass of BM called the Restricted Boltzmann Machine (RBM) (see Fig 6.1(b)) was proposed by Hinton (2002) [21] where the hidden and visible nodes had a bipartite structure. This structure allows an independent update of visible states, conditioned on the hidden states' knowledge and vice-versa. This property makes RBM very trained efficiently on a classical computer. Boltzmann machines have received much attention as building blocks of multi-layer learning architectures for speech and image recognition [123, 124]. The idea is that features from one RBM can serve as input to another RBM. By stacking RBMs in this way, one can construct the architecture of a Deep Boltzmann machine (see Fig 6.1(c)). It is known that approximate inference in deep Boltzmann machines can handle uncertainty better and deal with ambiguous data [3].

A comparison between the ML and the CD-based training of RBM is presented in [122]. The authors suggested that an initial CD-based training and a final ML-based



(a) General BM (b) Restricted BM (c) Deep BM

Figure 6.1: *Nomenclature of Boltzmann machines from [3]*

fine-tuning of RBM is the most computationally efficient way of training RBMs with less bias. This bias issue was further studied in [125] where the Persistent Contrastive Divergence (PCD) was developed. In this approach, the Markov Chain is not reset between parameter updates. This step brings the approximate gradient closer to the exact estimate in the limit of a small learning step. This method shows better performance on the testing data than the classical approach; however, it suffers from slow learning rates. A relatively faster approach was provided in [126] where Fast Persistent Contrastive Divergence (FPCD) A tutorial on different training strategies is given in [22].

It is intuitive to see that General BM has more representative power than RBM and its derivatives. However, the efficiency of the above-mentioned training methods is not expected to translate to the general case as the data-dependent expectations are not easy to compute, at least using classical techniques. Quantum annealers (QA) have provided a promising way forward to tackle this problem of expectation estimation. QA are physical devices that operate on Quantum mechanical laws and are designed to minimize the Ising model's energy. As they operate on finite temperatures, the simulations on QA results in sampling from the Boltzmann distribution of the Ising energies. Researchers have recently employed this property of QA to train BMs with a slightly more complicated structure than RBMs. For instance, [28] trained a Limited

Boltzmann machine (LBM) to recover missing data in the images of Chemical vapor deposition (CVD) growth for a MoS_2 monolayer. LBM allows sparse connectivity among the hidden units and, due to this complexity, it is not easy to deal with in a classical setting.

Another direction that researchers have taken is the training of specialized RBMs that can be better represented on the QA architecture, e.g., the chimera RBM which emulates the topology of DWave Quantum annealers [27]. This allows the model expectations to be estimated as a single sampling step instead of the k-step CD method. Meanwhile, the data-dependent expectations are estimated as the 1-step CD method due to the RBM's favorable topology. The result of this progress can be seen in the outburst of new applications of RBM in modern machine learning architectures, for instance, Sampling latent space in Quantum variational autoencoders (QVAE) [127], RBM as an associative memory between generative and the discriminative part of the Associative Adversarial Network Model (AAN) [128, 129] and Hybrid-Autoencoder-RBM approach to learn reduced dimension space [130].

In this work, an ML-based approach is studied for a General BM (see illustration in Fig 6.2). As discussed earlier, the topology of a highly connected graph is not conducive for CD-based approaches. The major hurdle of generating independent samples of BM is circumvented using QA. At present, the two popular QA devices are the "DWave 2000Q" system with approximately 2000 qubits connected in a Chimera topology and the "DWave Advantage" system with approximately 5000 qubits connected in a Pegasus topology. Considering the physical devices' sparsity, the largest complete graph that can be simulated on these systems has size 64 on the 2000Q and 128 on the Advantage system. The past growth in these systems' computational power suggests the prospect of solving a large-scale problem in the near future. Taking the prospect for granted, large and arbitrarily connected BM can benefit from unbiased estimation via QA. The method developed in this work does not use the

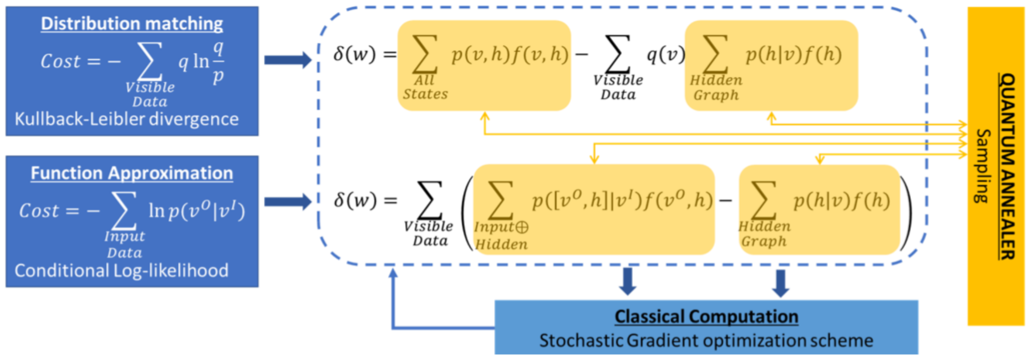

Distribution matching
$Cost = -\sum_{\substack{Visible\\Data}} q \ln \frac{q}{p}$
Kullback-Leibler divergence
Function Approximation
$Cost = -\sum_{\substack{Input\\Data}} \ln p(v^O|v^I)$
Conditional Log-likelihood
$\delta(w) = \sum_{\substack{All\\States}} p(v,h)f(v,h) - \sum_{\substack{Visible\\Data}} q(v) \sum_{\substack{Hidden\\Graph}} p(h|v)f(h)$
$\delta(w) = \sum_{\substack{Visible\\Data}} \left(\sum_{\substack{Input\oplus\\Hidden}} p([v^O,h]|v^I)f(v^O,h) - \sum_{\substack{Hidden\\Graph}} p(h|v)f(h) \right)$
QUANTUM ANNEALER
Sampling
Classical Computation
Stochastic Gradient optimization scheme

Figure 6.2: Illustration of gradient-based for generative and discriminative training of Boltzmann machines

graph's topology and is numerically sub-optimal for the cases when such structures are present (e.g., bipartite graph). For such cases, the readers are encouraged to pursue the literature listed above and the bibliography therein.

This work aims to demonstrate the use of quantum annealers for discriminative and generative tasks involving Boltzmann machines. Generative tasks involve sampling a state from a probability distribution. At the same time, a discriminative BM acts as a classifier for a given dataset. A BM trained for generative and discriminative purposes can be used to sample a labeled dataset from a probability distribution. For example, [131] developed a generative BM for sampling vertical and horizontal stripe patterned images. The second focus of this work to analyze the effect of annealing temperature on training. The probability distribution of the BM is dependent on the temperature, and the sampling temperature in QA is shown to be instance-dependent [27]. The results presented in their study are for an RBM trained for a generative task. Their strategy is extended to general BM for both generative and discriminative tasks.

6.2 Mathematical prerequisites

A Boltzmann Machine is a probabilistic graphical model defined on a complete graph which is partitioned into "visible" nodes taking up values observed during training denoted by the vector, $\boldsymbol{v}$, and "hidden" nodes where values must be inferred taking up values denoted by the vector, $\boldsymbol{h}$. These states collectively define the energy and consequently and the probability of each state. The readers are suggested to review Definitions I.1 and I.3 and Section 1.1.2 for notations. Ising model with the (0/1)-basis is used here. The parameter set, $\boldsymbol{\theta}$ is represented as a vector as in Section 1.3.

6.2.1 Generative Boltzmann Machines

The key idea behind a Boltzmann machine is the segregation of the vertices into visible and hidden states. This allows to write any state $\boldsymbol{S}$ of the graph as the following concatenation:

$$\boldsymbol{S} = [\boldsymbol{v}, \boldsymbol{h}]$$

where $\boldsymbol{v}$ denotes he state of he visible nodes and $\boldsymbol{h}$ denotes the states of the hidden nodes. Only visible states are observed by the user and their probability can be estimated by marginalizing over all hidden states. Therefore, the probability of a particular visible state, $\boldsymbol{v}$, is given as,

$$p(\boldsymbol{v}) = \sum_{\boldsymbol{h}} p(\boldsymbol{v}, \boldsymbol{h}) = \frac{1}{Z} \sum_{\boldsymbol{h}} e^{-\beta E(\boldsymbol{v}, \boldsymbol{h})} \tag{6.1}$$

This marginalization allows the BM to represent complex probability distributions. Consider a data set of N_{DS} visible states, $\mathcal{D} = \{\boldsymbol{v}_1, \cdots, \boldsymbol{v}_{N_{DS}}\}$. Each data state occurs with a probability $q(\boldsymbol{v}_k)$ for all $k \in \{1, ..., N_{DS}\}$, referred to as the true probability of the distribution. The performance of a BM can be judged by comparing the model distribution, $p(\boldsymbol{v})$ with the true distribution. This comparison can be carried out using the Kullback-Leibler divergence $D_{KL}(q||p)$ defined as,

$$D_{KL}(q||p;\boldsymbol{\theta},\beta) = -\sum_{\boldsymbol{v}\in\{\boldsymbol{v}_1,\cdots,\boldsymbol{v}_{N_{DS}}\}} q(\boldsymbol{v}) \ln \frac{p(\boldsymbol{v};\boldsymbol{\theta},\beta)}{q(\boldsymbol{v})}$$

The KL divergence is always non-negative with $D_{KL}(q||p) = 0$ if and only if $q = p$ almost everywhere. For this property, D_{KL} is chosen to be the cost function for training generative BMs.

Remark: In case, there is no meaningful notion of probability distribution of the data states. The true probability distribution can be substituted as $q(\boldsymbol{v}_k) = 1/N_{DS}$ for all $k \in \{1,\cdots,N_{DS}\}$. In this case, the KL Divergence is equal to the Log-likelihood of the data set normalized with the cardinality of the data set, N_{DS}.

6.2.2 Discriminative Boltzmann Machines

It is often desired to generate a labelled data set which entails assigning a classification to each each visible data point. This classifier can be included in our notation by further segregating the visible state into input-output pair. Consequently the state of the BM is represented as:

$$\boldsymbol{S} = [\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h}]$$

where, $\boldsymbol{v}^I$ and $\boldsymbol{v}^O$ denotes the "input" and "output" visible state. The state, $\boldsymbol{v}^O$ is used to encode the classification of state $\boldsymbol{v}^I$. Discriminative BMs, also referred to as conditional BMs in literature, are trained for classification using labelled data set. The cost function in this case is taken as the Negative Conditional Log-likelihood $\mathcal{N}$

defined as,

$$\mathcal{N}(\boldsymbol{\theta}, \beta) = -\sum_{[\boldsymbol{v}^I, \boldsymbol{v}^O] \in \{\boldsymbol{v}^1, \ldots, \boldsymbol{v}^D\}} \ln p(\boldsymbol{v}^O | \boldsymbol{v}^I; \boldsymbol{\theta}, \beta)$$

where, the conditional probability, $p(\boldsymbol{v}^O|\boldsymbol{v}^I)$ is estimated as:

$$p(\boldsymbol{v}^O|\boldsymbol{v}^I) = \frac{p(\boldsymbol{v}^I, \boldsymbol{v}^O)}{\sum_{\widetilde{\boldsymbol{v}}^O} p(\boldsymbol{v}^I, \widetilde{\boldsymbol{v}}^O)} \equiv \frac{\sum_{\boldsymbol{h}} p(\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h})}{\sum_{\widetilde{\boldsymbol{v}}^O, \widetilde{\boldsymbol{h}}} p(\boldsymbol{v}^I, \widetilde{\boldsymbol{v}}^O, \widetilde{h})}$$

6.3 Training Method

For a general purpose training strategy, the cost is set as a weighted average of KL Divergence and Negative conditional log-likelihood as described below

$$\mathrm{C} = \alpha D_{KL} + \frac{1-\alpha}{N_{DS}} \mathcal{N}(\theta), \qquad \alpha \in [0, 1] \tag{6.2}$$

where the $\alpha = 0$ signifies a generative BM while $\alpha = 1$ signifies a discriminative BM. Gradient based techniques are used to carry out the optimization procedure. The gradient is estimated as:

$$\begin{aligned} \frac{1}{\beta}\frac{\partial C}{\partial \theta_i} = -\alpha \mathbb{E}\left(\frac{\partial E}{\partial \theta_j}\right) + \sum_{\boldsymbol{v} \in \{\boldsymbol{v}^1, \ldots, \boldsymbol{v}^D\}} \Bigg(\left(\alpha q(\boldsymbol{v}) + \frac{1-\alpha}{D}\right) \mathbb{E}\left(\frac{\partial E}{\partial \theta_j}\middle| \boldsymbol{v}\right) \\ -\frac{1-\alpha}{D}\mathbb{E}\left(\frac{\partial E}{\partial \theta_j}\middle| \boldsymbol{v}^{\boldsymbol{I}}\right)\Bigg) \end{aligned} \tag{6.3}$$

And, the hessian is estimated as:

$$\begin{aligned} \frac{1}{\beta^2}\frac{\partial^2 C}{\partial \theta_i \partial \theta_j} = \alpha \operatorname{Cov}\left(\frac{\partial E}{\partial \theta_i}, \frac{\partial E}{\partial \theta_j}\right) - \sum_{\boldsymbol{v} \in \{\boldsymbol{v}^1, \ldots, \boldsymbol{v}^D\}} \left(\alpha q(\boldsymbol{v}) + \frac{1-\alpha}{D}\right) \operatorname{Cov}\left(\frac{\partial E}{\partial \theta_i}, \frac{\partial E}{\partial \theta_j}\middle| \boldsymbol{v}\right) \\ + \sum_{\boldsymbol{v} \in \{\boldsymbol{v}^1, \ldots, \boldsymbol{v}^D\}} \frac{1-\alpha}{D} \operatorname{Cov}\left(\frac{\partial E}{\partial \theta_i}, \frac{\partial E}{\partial \theta_j}\middle| \boldsymbol{v}^I\right) \end{aligned} \tag{6.4}$$

The definitions of all statistical quantities are presented in Appendix D.1 and the derivatives of cost functions are estimated in Appendix D.2.

6.3.1 Optimization Scheme

Stochastic gradient and Newton methods have been widely employed in such problems. A comparative performance of many variants of such stochastic methods are studied in [132]. In stochastic optimization methods, it has been shown that both newton methods and gradients-based methods have a local linear convergence rate. However, [133] developed a Newton method that is independent of the condition number in contrast to the gradient-based scheme. These developments motivate the use of Hessian in the optimization process. Such schemes are very useful in problems concerning sampling from sensitive devices like Quantum annealers. Analyzing the different variations of stochastic methods is out of scope of this work. A mini batch momentum-based approach is adopted. This approach can be easily substituted for one of the more sophisticated ones presented in [132, 133]. The following momentum-based update rule is used:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} + \Delta\boldsymbol{\theta}^{(t)}, \qquad \Delta\boldsymbol{\theta}^{(t)} = \eta\boldsymbol{r}^{(t)} - \lambda\boldsymbol{\theta}^{(t)} + \nu\Delta\boldsymbol{\theta}^{(t-1)} \tag{6.5}$$

The parameter, η defines the learning rate of the method and ν defines a momentum rate. A higher momentum rate means the current learning direction has a higher influence for the previous learning rate. In general, the momentum is kept low at the beginning and slowly increased as the learning progresses. This technique is employed to reduce oscillations in the final stages of training. The parameter λ modifies the cost function to minimize the magnitude of the learned parameter. In this work. this parameter is identically set to 0 in all test cases and is mentioned only for completeness. The variable, $\boldsymbol{r}$, denotes the rate of update. In the gradient-based method, it is estimated as:

$$\boldsymbol{r}^{(t)} = -\nabla_{\boldsymbol{\theta}^{(t)}} C$$

In Newton method, rate of update is estimated as:

$$\boldsymbol{r}^{(t)} = -(\nabla^2_{\boldsymbol{\theta}^{(t)}} C)^{-1} \nabla_{\boldsymbol{\theta}^{(t)}} C$$

Remark 1: The Hessian matrix estimated from the sampling process, is usually rank deficient. The main reason is under-sampling. Therefore, inversion of these matrices pose a challenge. In this work, Tikhonov regularization is used where singularity of $\nabla^2_{\boldsymbol{\theta}^{(t)}} C$ is alleviated by adding positive terms to the diagonal as:

$$\nabla^2_{\boldsymbol{\theta}^{(t)}} C \to \nabla^2_{\boldsymbol{\theta}^{(t)}} C + \epsilon^2 \mathbb{I}$$

where $\mathbb{I}$ is the identity matrix. This regularization results in the following useful property for the rate of update:

$$\boldsymbol{r}^{(t)} = \text{argmin}_{\widetilde{\boldsymbol{r}}} \, ||(\nabla^2_{\boldsymbol{\theta}^{(t)}} C)\widetilde{\boldsymbol{r}} + \nabla_{\boldsymbol{\theta}^{(t)}} C||_2 + \epsilon^2 ||\widetilde{\boldsymbol{r}}||_2$$

Remark 2: The above update rule works for unconstrained optimization. A constrained optimization problem can be considered by employing lagrange multipliers. In this study, the constraints are much simpler, $|H_i| < H_0$ and $|J_k| < J_0$. These constraints represent the practical range of parameters for Quantum annealers. These bounds are implemented by using following scaling parameter:

$$\delta = \max \left\{ \frac{\max_{i \in \{1, \ldots N_V\}} |H_i|}{H_0}, \frac{\max_{k \in \{1, \ldots N_C\}} |J_k|}{J_0} \right\}$$

In any optimization step, if $\delta > 1$, then the parameters are scaled as: $\boldsymbol{\theta}^{(t)} \to \boldsymbol{\theta}^{(t)}/\delta$

and the corresponding $\Delta\boldsymbol{\theta}^{(t)}$ is updated. The optimization procedure is presented in Algorithm 12. The procedure uses a subroutine *EstimateDerivatives* (Algorithm 11) to estimate the gradients and hessian. This subroutine is discussed in the next section. For gradient based approaches, the estimation of hessian can be tactically avoided from algorithm 11 by ignoring all covariance terms and the hessian in this case is set to a Identity matrix.

6.3.2 Numerical Estimation of Gradient and Hessian

QA can be treated as a black-box that samples the states for a given set of parameters. DWave Quantum annealer is used in this work that operates based on the $\{+1, -1\}$ states. The energy parameters for the $\{+1, -1\}$ states and the $\{1,0\}$ states can be transformed using the relation presented in Appendix D.3[1]. The user can specify the number of samples. The probability of a particular sample state is then be estimated as the ratio of the number of occurrences of that state to the specified number of samples. It is easily noticeable that the gradients and hessian described in terms of statistics of the energy gradient $\nabla_{\boldsymbol{\theta}}E$. The first term in the gradient and the hessian, requires estimation of $\mathbb{E}(\nabla_{\boldsymbol{\theta}}E)$ and $\mathrm{Cov}(\nabla_{\boldsymbol{\theta}}E)$. In the notation described in Section 1.3, given a sample state $\boldsymbol{S}$, energy gradient is estimated as:

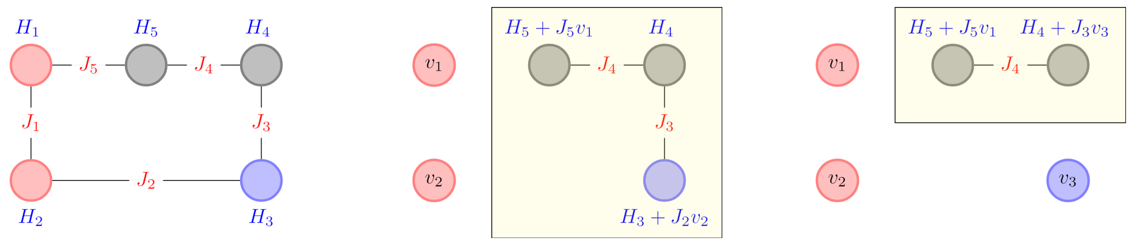
$$\nabla_{\boldsymbol{\theta}}E(\boldsymbol{S}) = \boldsymbol{\varepsilon}(\boldsymbol{S})$$

The latter terms in Eq (6.3) and Eq (6.4) are more complicated to compute as they require summation over each visible data. Two possible strategies can be employed in this case. In the first approach, all sampled states are grouped according to the visible/input states. The conditional probabilities are then estimated by treating each data state's respective groups as the sample set. Since the samples from This

[1]Basis transformation may scale the parameters outside the allowed range of DWave. This problem can be mitigated by choosing appropriate bounds on Field and Interaction parameters in the optimization process

estimation is unbiased as the QA samples are independent. In theory, the accuracy of these conditional probabilities increases with sample size. However, in practice, the number of samples is finite and should be kept to a minimum possible value to reduce computational cost. This is a critical drawback of this approach. For instance, not every data state needs to appear in the samples. In such cases, the KL Divergence cannot be estimated.

The other approach is to run independent simulations for each data state on the subgraph of hidden nodes, $\mathcal{G}_H$, and the subgraph of hidden and output nodes, $\mathcal{G}_{HO}$. The energy parameters of these subgraphs depend on the visible states (for $\mathcal{G}_H$), and input states (for $\mathcal{G}_{HO}$). The field terms are augmented to include the energy for fixing the states of the removed nodes. An illustration of this procedure is presented in Fig 6.3. One can observe that this process leads to a shift in energy states. For instance, same states in Fig 6.3(a)&(b) have an energy difference of $H_1v_1 + H_2v_2 + J_1v_1v_2$. However, the Boltzmann distribution remains unchanged by a uniform shift in energies. The drawback of this method is that this procedure's computational cost grows with the training data size. This growth by itself is not a problem; however, the sampling step is usually the most time-consuming. In general, CD-1 steps are used to determine this term in RBMs quickly. However, this method is not extendable to General BMs. The authors are currently unaware of any scheme that circumvents this computation. This second approach of running independent simulations for each data will be adopted from here onward. The procedure for estimating gradient and hessian from the sampled data is presented in Algorithm 11. It should be noted that the estimation of RHS of Eq (6.3) and Eq (6.4) only yields the direction of gradient and hessian, respectively. The size of the update can be subsumed in the learning rates. However, the value of β influences the probability distribution and, consequently, influences trained parameters' value. This issue is discussed in the next section.



(a) (a) Graph, $\mathcal{G}$ (b) (b) Subgraph, $\mathcal{G}_{HO}$ (c) (c) Subgraph, $\mathcal{G}_H$

Figure 6.3: Illustration for estimating parameters of the subgraph. The Input, Output and Hidden nodes are represented with red, blue and grey colors, respectively. The field parameters and interaction parameters are written in blue and red fonts, respectively. The subgraphs are presented in a yellow box. (a) Cyclic graph, $\mathcal{G}$ with 2 input nodes, 1 output node and 2 hidden nodes (b) Subgraph of output and hidden nodes, $\mathcal{G}_{HO}$: Fixing the visible input $\boldsymbol{v}^I = [v_1, v_2]$ results in an augmented field term on the output and hidden nodes (c) Subgraph of hidden nodes, $\mathcal{G}_H$: Fixing the visible data $\boldsymbol{v} = [v_1, v_2, v_3]$ results in an augmented field term on the hidden nodes

6.3.3 Effect of Annealing Temperature

Experimental evidence has shown that the apparent annealing temperature, i.e., the temperature corresponding to the Boltzmann distribution of samples, is instance-dependent [27]. The corresponding inverse temperature is referred to as β^* in this section. The consequence of this instance-dependence is that the quantum annealing systems cannot be rated for specific temperatures, and β^* has to be estimated from the samples for each instance of the graph. The knowledge of β^* is crucial in developing models capable of being implemented on different computational devices. Even in the same machine, two different embeddings of the same logical graph may lead to different annealing temperatures and consequently show disparities in performance. The key idea behind the estimation of β^* is that the Boltzmann distribution of a state can be equivalently written as follows by taking a log on both sides:

$$\log p(\boldsymbol{S}) = -\beta E(\boldsymbol{S}) - \log Z$$

β^* is estimated as the slope of this line. The exact form is as follows:

$$\beta^* = -\frac{\sum \left(E - \mathbb{E}\left(E\right)\right)\left(\ln p - \mathbb{E}\left(\ln p\right)\right)}{\sum (E - \mathbb{E}(E))^2} \tag{6.6}$$

A similar approach was developed by [27], where they used the statistics of different energy levels:

$$p(E;\beta) = D_g(E)e^{-\beta E}$$

where $D_g(E)$ is the degeneracy of a state. The authors used the log of ratio of probability, $l(\beta) = p(E_1;\beta)/p(E_2;\beta)$ for some fixed energy levels. They manipulated the value of β by rescaling the parameters and were able to estimate the β from the slope of $l(\beta)$ and the scaling factor. Readers should notice that although these two approaches are based on a similar argument, they differ greatly in their application. In the former approach, the intuition is that β represents the slope of the following sampled data, $(E(S), \log p(S))$. Meanwhile, the latter approach uses the data of the probability distribution of energy levels. Both methods have their pros and cons. The second method requires sampling at rescaled parameters; thus, it assumes that β remains invariant with rescaled parameters. This is usually true in the range of parameters specified by the QA manufacturers, but this condition may fail if the required rescaled value does not adhere to these limits. For the first approach, binning has to be done for each unique state that is sampled, and this step is computationally more expensive than binning energy levels, especially in the limit of large graphs. On the favorable side, it requires only one set of samples as a rescaling of parameters is not needed.

The rescaling technique can also be applied to look for model parameters with better performance in training cost. The key idea is that once the training of BM is done at some β^*, it might be possible to vary the β and further reduce the cost. This optimal value of β, (say β^o) can then be used to rescale the parameters as $\boldsymbol{\theta} \to \boldsymbol{\theta}.\beta^o/\beta^*$. The key challenge is that, it is not possible to simulate different β value other than using rescaled parameters. However, the KL Divergence and the Negative Conditional Log-likelihood can be approximated as their respecve Taylor expansions. Moreover, the coefficients can be estimated from the sampled states at β^*. The Taylor expansion till the second term is as follows:

$$\begin{aligned} D_{KL}^{\text{app}}(\beta) &= D_{KL}^* + \left.\frac{\partial D_{KL}}{\partial \beta}\right|_{\beta^*}(\beta-\beta^*) + \frac{1}{2}\left.\frac{\partial^2 D_{KL}}{\partial \beta^2}\right|_{\beta^*}(\beta-\beta^*)^2 + \cdots \\ \mathcal{N}^{\text{app}}(\beta) &= \mathcal{N}^* + \left.\frac{\partial \mathcal{N}}{\partial \beta}\right|_{\beta^*}(\beta-\beta^*) + \frac{1}{2}\left.\frac{\partial^2 \mathcal{N}}{\partial \beta^2}\right|_{\beta^*}(\beta-\beta^*)^2 + \cdots \end{aligned} \tag{6.7}$$

where

$$\begin{aligned} \frac{\partial D_{KL}}{\partial \beta} &= -\mathbb{E}_{\boldsymbol{v},\boldsymbol{h}}(E) + \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v}) \sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h}) p(\boldsymbol{h}|\boldsymbol{v}) \\ \frac{\partial^2 D_{KL}}{\partial \beta^2} &= \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v}) \left(-\operatorname{Var}(E|\boldsymbol{v}) + \operatorname{Var}(E)\right) \\ \frac{\partial \mathcal{N}}{\partial \beta} &= \sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} \mathbb{E}(E|\boldsymbol{v}) - \mathbb{E}(E|\boldsymbol{v}^I) \\ \frac{\partial^2 \mathcal{N}}{\partial \beta^2} &= \sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} -\operatorname{Var}(E|\boldsymbol{v}) + \operatorname{Var}(E|\boldsymbol{v}^I) \end{aligned}$$

The comparison of the approximated and exact behavior of a trained BM (see Section 6.4) is presented in Fig 6.4. The exact behavior is estimated by evaluating the Boltzmann distribution, Eq (1.5). The value of β^o is estimated as the minimizer of the approximated quadratic cost function, $C^{\text{app}} = \alpha D_{KL}^{\text{app}} + (1-\alpha)\mathcal{N}^{\text{app}}/N_{DS}$. Therefore, when approximated cost is convex, the β^o is estimated as:

$$\beta^o = -\left(\frac{\partial^2 C}{\partial \beta^2}\right)^{-1} \frac{\partial C}{\partial \beta}$$

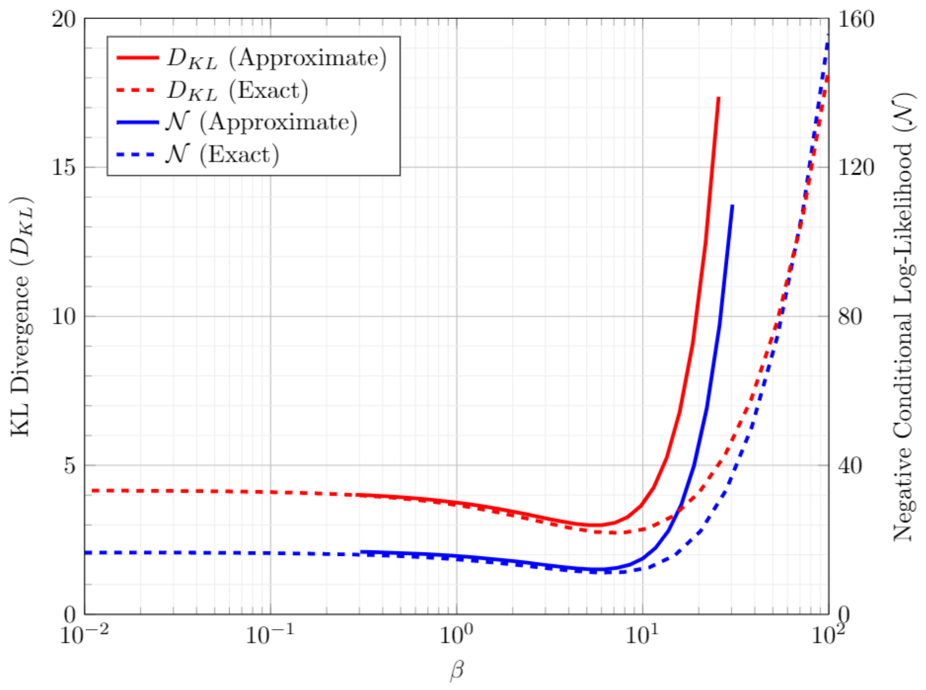


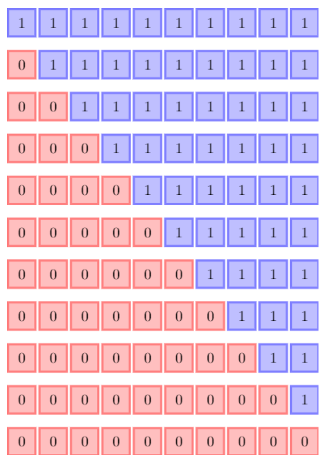
Figure 6.4: Comparison of approximate and exact performance of BM

6.4 Examples

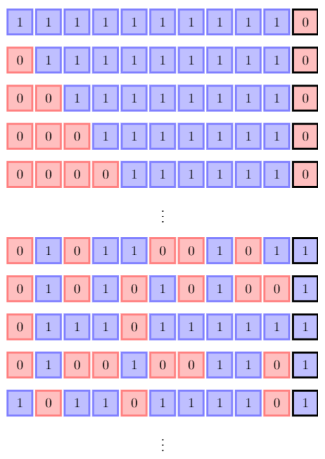
As a toy problem, the data illustrated in Fig 6.5 is used to train the BM models. The first data set (Fig 6.5(a)) is unlabelled and has 11 data points. The second data set (Fig 6.5(b)) has 40 data points with 11 labelled as '0' and 29 labelled as '1'.

In all the cases, the training parameters of Eq (6.5) have a constant value of $\eta = 0.1$, $\nu = 0.7$, and $\lambda = 0$. The Hessian is inverted using Tikhonov regularization with $\epsilon = 10^{-3}$. The run time data (see Fig 6.6) shows that the Newton approach performs better than the gradient-based approach. The fluctuations in the deterministic training case (NumBatch = 1) are due to the DWave sampling step's stochastic nature. It should also be remarked that the parameters were not optimized for individual cases and were arbitrarily picked from a suitable range.

The stochastic training method with 2 batches was employed for the first data set (Generative learning). Two BM's with 3 hidden nodes were considered, first with complete connectivity and the second with Restricted BM architecture. The variation



(a) Data points for generative training of Boltzmann machine



(b) Data points for discriminative training of Boltzmann machine

Figure 6.5: Visible data states, $\mathcal{D}$, for training Boltzmann machines. Each row represents a single data point. (a) Each data point represents the phase of a state at 10 spatial points. The '0' phase is accumulated to the left, and the '1' phase is accumulated to the right with at most 1 boundary. (b) Labeled data set with the data points described in part(a) are labeled as '0', and data points with random spatial distribution labeled as '1'. The label is appended at the end of each data point in a black box.

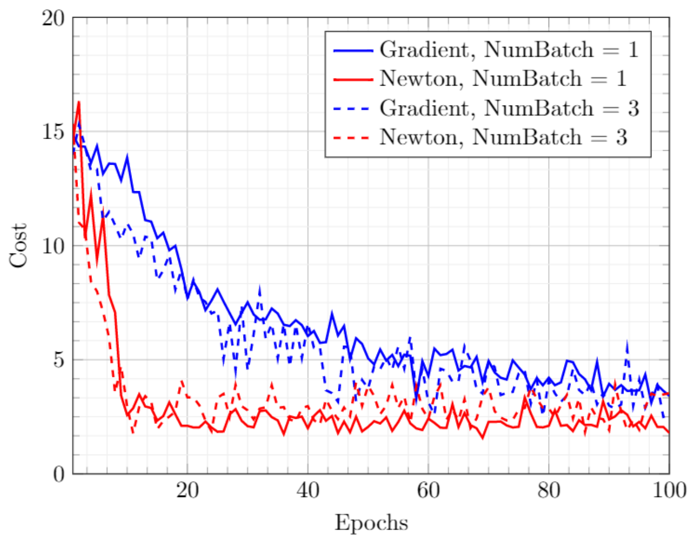


Figure 6.6: Training data for a General BM with 3 Hidden nodes. The cost of training is defined by Eq (6.2) with $\alpha = 0.5$.

of KL Divergence (training Cost) with the annealing temperature is shown in Fig 6.7. It is observed that trained BM of the general type performs better than the Restricted type. This is an intuitive result as RBM is a specialized case of the General BM and has less representation capability in comparison.

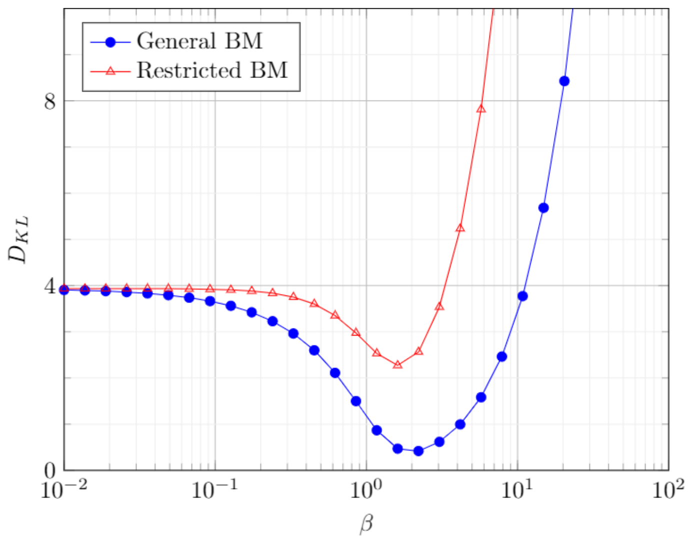


Figure 6.7: Trained Generative BM with 3 hidden nodes

The effect of cost parameter, α was studied for training General BM with the second dataset. The results are presented in Fig 6.7. The training is carried out with 70/30 split into training and testing data. A large reduction in KL Divergence is observed, even for a small value of α. Moreover, there was no substantial change in the conditional likelihood. This result suggests that the performance of Conditional/Discriminative BM can be enhanced by adding a small KL Divergence component to the training cost.

6.5 Conclusion

Quantum annealing has the potential to improve the training of General Boltzmann machines significantly. The stochastic Newton and gradient-based training methods can be employed using direct sampling from quantum states. This procedure can

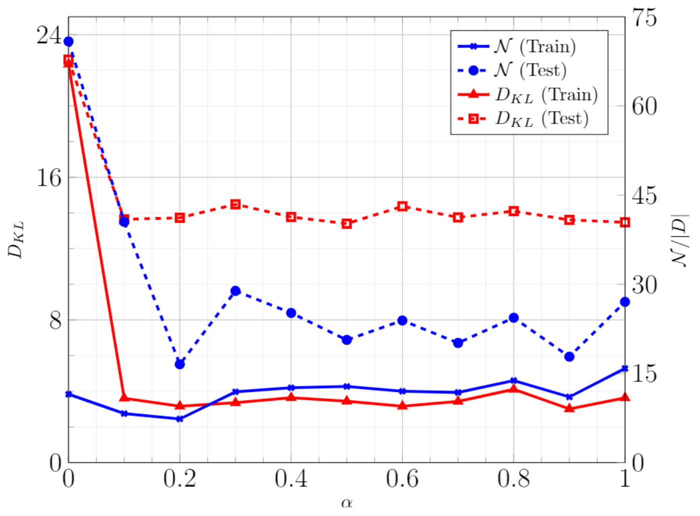


Figure 6.8: Performance of trained BM with 4 hidden nodes with respect to the cost parameter, α.

accelerate the training of a General Boltzmann machine which have higher representation capability. The use of quantum annealers is promising for quantum/classical training since many qubits are available, and the training takes advantage of measurements on the physical realization of the Boltzmann machine [127, 134]. Unlike the other popular methods like the Contrastive Divergence, this method does not utilize the suggested BM's special topology. However, in practice, having a sparse connection is desirable as that allows embedding larger graphs in the DWave architecture. These methods were employed to carry out generative and discriminative training in toy problems. The numerical results suggested that stochastic Newton optimization performs better than gradient-based optimization. It was also observed that adding a small weightage to KL Divergence in discriminative cost greatly improves BM's performance. A major contribution of this work is in developing a procedure to approximate the behavior of BM in slightly perturbed temperature. This procedure is useful in approximating better parameters for BM. This method is also useful for transferring BM from one quantum annealing system to another, which may have different annealing temperatures. It must be remarked that due to the instance-dependence of the QA sampler, this transfer also refers to using different embedding of the same BM on the same QA machine. In the future, this work will be tested for benchmark problems. A rigorous analysis of training parameters is also expected in future publications.

CHAPTER VII

Conclusions and Future Work

7.1 Summary

This thesis's primary goal is to identify, formalize, and solve important mechanics problems in the discrete pairwise energy form. In the most general case, these problems can be solved on the Quantum annealer. Meanwhile, certain special cases can be approximated with great efficiency on classical computers. These special cases can still be efficiently solved using Quantum annealing methods; in fact, as the Quantum architectures advance, it is likely, that using Quantum approaches for large scale problems will have a greater advantage than the classical computation method in terms of computation time. We have pursued both kinds of problems here. In the first part of the thesis, we developed methodologies to capture surface phenomena using this model. These problems come under the category of problems that can be solved classically. Within this part, we made the following contributions:

- We developed a methodology to label the elements of a finite element such that it can replicate experimental images of materials with smooth boundaries like polycrystalline and composite materials [81].
- We took a similar approach as above to model cracked surface in a finite element mesh [135]. In this case, we developed a modular framework for mul-

tiscale modeling of crack paths in polycrystalline materials. The modularity allows Plug'n'Play capability for the testing of different energy-based criteria for fracture. The problem pursued here employed Maximum energy release rate criteria to estimate brittle failure in elastic polycrystalline material with anisotropic surface energies.

- We also developed an approach for studying multiphase flow in uniform meshes. Surface energies drive this phenomenon; however, accurately estimating them in these meshes is challenging. We extended the known methodology for pixel mesh developed by [89] to Quadtree meshes. This modification can significantly improve the computational cost for high fidelity simulation. These methodologies are extended to anisotropic grains and 3D microstructure. Application to denoising microstructure is also discussed.

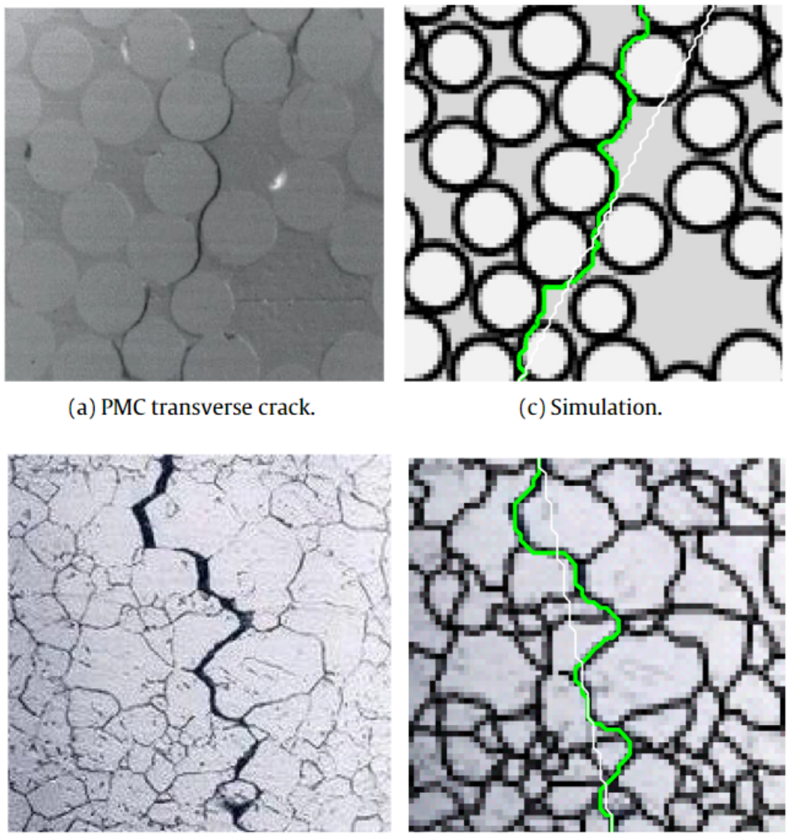
We developed a novel technique to simulate finite-element problems in a pairwise energy framework. The proposed Ising model cannot be solved efficiently on classical computers. Consequently, it is solved using quantum annealers. The algorithms developed in this work classically computes the Ising optimization problem from the finite element formulation with no additional order of complexity. This work's most important contribution is the 'Box representation' of solutions, which allows a function to be represented by a sparsely connected graph. The next step in this research is to utilize this concept to develop low-depth circuits for gate-based computing. This technique can provide great advantages in Black-Box optimization.
In addition to the work listed so far, where the problem is solved by identifying the ground states of the model (forward problem), methodologies are developed to train these models to emulate a prescribed behavior. Firstly, a mixed linear integer program is developed to do band-gap optimization in the Potts model. The methods developed in this case can provide optimal parameters to replicate the prescribed ground states or its cardinality. This method grows exponentially in size with the number of nodes.

Consequently, it can only be used for smaller graphs, for instance, repeating units of periodic lattices or Nodal graph in the box algorithm.

For larger graph sizes, exact band-gap minimization cannot be done with reasonable computing resources. Usually, Maximum likelihood principles are used to pursue such problems. We developed a quantum annealing-based training methodology for Boltzmann machines – a machine learning architecture based on the Ising model. Statistical data from Quantum annealing samples is used to estimate the gradient and hessian for the optimization procedure. The method developed here is independent of the graph's topology, which allows the use of densely connected graphs that have higher representability. A major contribution of this work is the technique for estimating annealing temperature and the local behavior of costs. This procedure is useful in approximating better parameters by temperature rescaling. It is also useful for transferring BM from one quantum annealing system to another, which may have different annealing temperatures.

7.2 Applications of proposed methodologies

Fracture: The approach proposed for fracture has significant advantages over continuum methods with diffuse damage (smeared crack, gradient damage, or phase field methods). The fracture surface geometry is clearly identified (sharp interface), allowing one to calculate the surface energy along the path accurately. Thus, the surface energy of the crystal and grain boundaries modeled using smaller-scale simulations such as molecular dynamics [136] or density functional theory can be directly used in this approach. In a previous work (MicroFract code [34]), we had shown the applicability of the model to composite (matrix–fiber) crack paths (Fig 7.1). Here, matrix–inclusion or matrix–fiber interface properties can come from molecular dynamics [137, 138, 139]. It also showed the applicability of the method to modeling intergranular stress–corrosion cracks, where the effect of alloying or impurities on the grain boundary energy can come from density functional theory [140].


(a) PMC transverse crack.

(c) Simulation.

(b) SCC crack in Inconel.

(d) Simulation.

Figure 7.1: (a,b) Transverse cracks under tension normal to fibers in a polymer matrix composite and intergranular stress corrosion cracking of Inconel (c,d) Graph cuts solution:green line indicates the computed crack trajectory.

Further work on validating microstructurally short crack paths in 2D thin foil specimens against experimental results [141, 142] from Allison group at the University of Michigan is underway and has shown very promising results. Code extension to 3D fracture modeling has been already enabled (Fig 7.2), however formal theoretical models of anisotropic 3D crack tip stress intensity factor need to be developed for the model before comparison against experimental 3D fracture paths obtained from high energy synchrotron measurements (Allison group, [143]). It will be of interest to extend this method for modeling failure, including plasticity. In this case, one could split the surface energy into two terms, cleavage energy based on crack normal as described in this thesis and a plastic work that additionally depends on the shear direction obtainable from crystal plasticity models [144, 145, 146]. Beyond a post-fracture method, it is possible to use the code to identify possible fracture paths during in-situ testing when using plastic strain localization images from methods such as digital image correlation [147, 148] or elastic stress localization images from high resolution electron backscatter diffraction as inputs [149, 150].

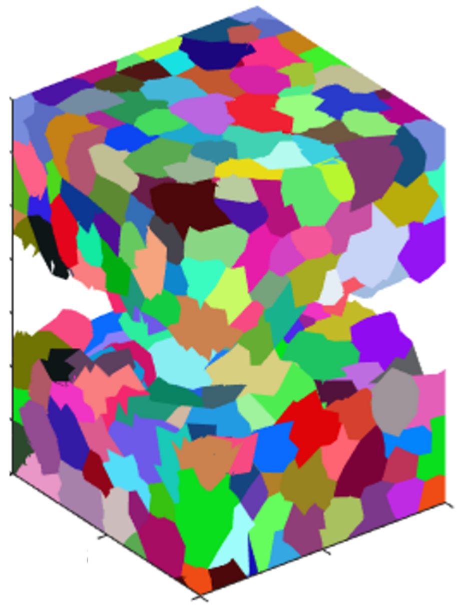


Figure 7.2: 3D fracture modelled using the graph coloring method proposed in this thesis.

Segmentation and denoising polycrystals Recently, new and powerful techniques based on feature–matching algorithms [151] or MRF techniques have been developed for synthetic reconstruction of multiphase alloys microstructures [152, 153, 154, 155, 156]. The techniques work by matching observed features in an example 2D image to a synthesized 3D microstructure. However, these techniques do not enforce any particular physics-based constraints on grain boundary structure. In materials science of GBs, certain facets are preferred, and GB junctions often follow certain geometric (angular) constraints. Graph segmentation methods provide a pathway towards including such physics-based constraints to improve the overall reconstruction. As proposed in the chapter on grain evolution, an energy-based approach can be used as a post-processing step, allowing for minimization of GB energy in segmented MRF images (Fig 7.3). The benefits are three-fold: the grains are segmented from the color space outputs of these models; it gives the users an option to force desired grain shapes (e.g., equiaxed or columnar grains) during segmentation, and finally, the GB angles can be corrected based on global energetics. The Riemannian metric dictates optimal grain shapes. GB energies of all pairs of grains in the microstructure (as a function of crystallography only) can be included in the form of Ising interaction term. As a result, minimization of the cost function can lead to an energy-efficient description of GB interfaces in synthetic microstructure models.

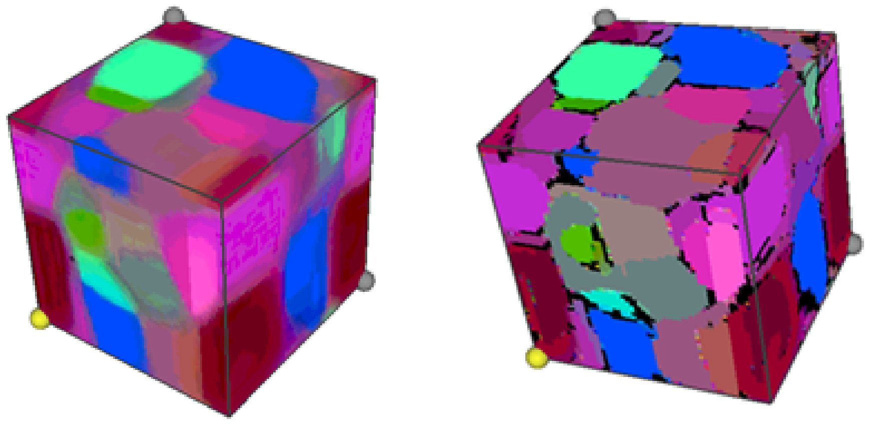


Figure 7.3: (left) Synthetic microstructure from MRF algorithm. (right) Use of graph cut segmentation to identify the grain interfaces (shown in black)

Applications of quantum machine learning methods

Quantum architectures, unlike Classical, promise exponential compression of memory, allowing for great speed-ups. However, classical algorithms cannot be directly transmuted to a quantum one, and an enormous amount of research effort is required in this area. The most notable contribution in the last few decades has been the HHL algorithm (and variants) to solve a sparse linear system of equations. The ripples of this development have given rise to many new advances in the field. However, we are qubit-limited, so in the near-term quantum-classical hybrid algorithms are critical.

These algorithms use classical algorithms on traditional computers for the most part and switch to using quantum computers at bottlenecks where quantum computing can outperform classical computers. In this thesis, we demonstrated the advantage of quantum annealers to speed-up sampling from Boltzmann machines (a classically hard problem)[157]. The classical algorithm takes care of performing tasks such as optimization of quantum hardware parameters.

The Boltzmann data mining algorithm has several applications, but specific to our research group, can be used for exploration of microstructure–process–property relationships in engineering materials [158, 159, 160]. Engineering properties of metallic alloys such as modulus, strength, and thermal conductivity are dependent on the substructure of the material at lengths of a few hundred microns and below. Microstructural features such as the orientation distribution function (ODF) [161, 162] describe this substructure. Microstructural features can be tailored so that desired properties can be achieved through controlled deformation or thermal treatment [163, 164, 165]. Recent work has focused on identifying process parameters, for example, scanning width in additive manufacturing [166] or the initial texture that can be cold rolled to achieve a desired final texture [167], but the problem of sequence identification is largely unexplored. A typical process for manufacturing a component such as a turbine blade can contain as many as twenty distinct processing stages, each taking the microstructure from one state to another, eventually achieving the desired microstructure problem of 'processing path design' aims to discover a sequence of known processes to realize microstructures with optimal properties. This problem is challenging because of the combinatorial complexity of permuting sequence of processing operations and can take advantage of machine learning, but the problem has been sparsely explored [168].

A recent work in our group in reference [4] has led to the development of an open-source database of 346000 microstructural features that result from sequences of up to four stages of tension, compression, and rolling along with different directions in various permutations. The use of General Boltzmann machines is suggested to develop process–microstructure–property relationships from the database. In the limited-qubit scenario, early use-cases of the quantum algorithm can rely on apriori data reduction to reduce the number of input variables (due to the small number of available qubits) [130]. For example, a small set of microstructures features (e.g., latent variables generated by an autoencoder, as shown in Fig 7.4 representing 346000 microstructures in a 2D latent space) can be used to link process and microstructures.

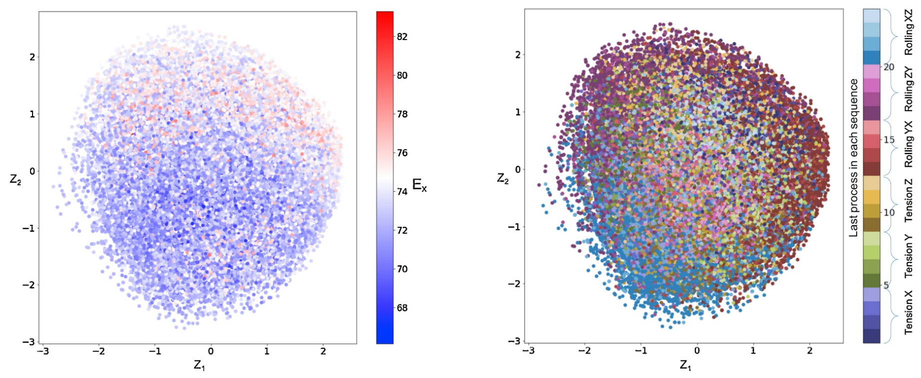


Figure 7.4: The database from reference [4] with 346000 microstructures. A latent space representation is shown. Each point in the latent space representation corresponds to one microstructure from the training set. Figure (a) shows points colored by the elastic modulus, and figure (b) shows points colored by the last process from each sequence. The challenge in this future work is to learn the relationship between the latent variables and the process using General Boltzmann machines.

Inverse modeling of differential equations

Inverse modeling tools will be useful for modeling the uncertainties in the processing path as a function of uncertainties in the desired properties or microstructure, an important problem in materials sciences [169, 170, 171]. Consider a relationship of the form: $L(u, \theta) = 0$, where u represents the desired properties, and θ defines a material processing path. Provided a set of noisy property data, where each data point is a requirement of u at some locations in the specimen, we want to estimate process parameters θ. Such inverse problems based on observed error-prone state variables are ubiquitous in control theory. Markov chain Monte Carlo (MCMC) techniques are typically used to make posterior inference requiring the repeated solution of PDEs with thousands of candidate parameter values. Using a quantum annealer, solutions of PDEs can be represented [172] with parameters θ embedded in the biases and coupling strengths. The optimal parameters can be retrieved by training the Boltzmann machine on the noisy data. Inverse modeling has several other practical applications in non–destructive testing of materials or in manufacturing, e.g., to optimize the flow rate of coolants in machining.

Early machine learning algorithms on quantum computers will take advantage of quantum computers' ability to sample complex probability distributions. This sampling step is essential in generative model development, and classical algorithms will either fail or take very long when the probability distributions to be learned are of a complex multi-modal type. For example, the Boltzmann machine can be trained on a microstructure database's input latent space with outputs encoding the process variables. The trained outputs represent a probability distribution of process sequences and provide a rich set of information for identifying the most economical processing route among those sampled.

APPENDICES

APPENDIX A

Theorems and Propositions

Theorem I.10: For a given set of parameters, $\boldsymbol{\theta}_D$, such that (i) $\mathcal{S}_G(\boldsymbol{\theta}_D) = \mathcal{S}_D$ (ii) $\Delta E > 0$, following statements hold true:

(a) $\eta(\boldsymbol{\theta}_D, \beta)$ monotonically decreases with β and the low temperature limit

$$\eta_\infty(\boldsymbol{\theta}_D) = \lim_{\beta\to\infty} \eta(\boldsymbol{\theta}_D, \beta) = N_{GS} \log(N_{GS}) \tag{A.1}$$

(b) $\eta(\boldsymbol{\theta}_D, \beta)$ is bounded as:

$$N_{GS} \log(N_{GS}) < \eta(\boldsymbol{\theta}_D, \beta) \leq N_{GS} \log\left(N_{GS} + N_{ES} e^{-\beta\Delta E}\right) \tag{A.2}$$

(c) For any $\epsilon > 0$, there exists a β^* such that for all $\beta > \beta^*$, $\eta(\boldsymbol{\theta}_D, \beta) - \eta_\infty(\boldsymbol{\theta}_D, \beta) < \epsilon$ where β^* is estimated as:

$$\beta^* = \frac{1}{\Delta E}\left(\log\frac{N_{ES}}{N_{GS}} - \log\left(e^{\epsilon/N_{GS}} - 1\right)\right) \tag{A.3}$$

Proof. **(a)** Since $\mathcal{S}_G(\boldsymbol{\theta}, \beta) = \mathcal{S}_D$, the Negative Log Likelihood, $\eta(\boldsymbol{\theta}_D, \beta)$, is estimated as:

$$\eta(\boldsymbol{\theta}_D, \beta) = N_{GS}\beta E_0 + N_{GS} \log Z$$

The derivative is estimated as:

$$\frac{d\eta}{d\beta} = N_{GS} \left((E_0 - \mathbb{E}(E) \right) \tag{A.4}$$

where

$$\mathbb{E}(E) = \sum_{\boldsymbol{S} \in \mathcal{S}} E(\boldsymbol{S}) p(\boldsymbol{S}|\boldsymbol{\theta}_D, \beta)$$

Since $\Delta E > 0$, the expected energy is strictly bounded below as $\mathbb{E}(E) > E_0$. Consequently:

$$\frac{d\eta}{d\beta} < 0$$

In the low temperature limit, Eq (1.5) estimates that the probability of all excited states approaches 0 while all ground states are equally likely with probability $(N_{GS})^{-1}$. Therefore, the value of η in this limit is estimated as Eq (A.1).

(b) Let $\boldsymbol{S}_G \in \mathcal{S}_G$ and $P = p(\boldsymbol{S}_G|\boldsymbol{\theta}_D, \beta)$ so that $\eta(\boldsymbol{\theta}_D, \beta) = -N_{GS} \log P$. The probability of occurrence of a ground state is given by $N_{GS}P$ and occurrence of a excited state is given as $(1 - N_{GS}P)$. Moreover, for any finite value of β both of these probabilities are finite. Therefore, the expectation of energy, $\mathbb{E}$, can be bounded as

$$\mathbb{E} = N_{GS}PE_0 + \sum_{\boldsymbol{S} \in \mathcal{S}_E} E(\boldsymbol{S}) p(\boldsymbol{S}|\boldsymbol{\theta}_D, \beta) \leq N_{GS}PE_0 + (1 - N_{GS}P)E_1$$

Substituting in Eq (A.4),

$$\frac{d\eta}{d\beta} = E_0 - \mathbb{E}(E) \leq (N_{GS}P - 1)\, N_{GS}\Delta E$$

Substituting $P = e^{-\eta/N_{GS}}$ gives the following differential inequality

$$\frac{d\eta}{d\beta} \leq \left(N_{GS}e^{-\eta/N_{GS}} - 1\right) N_{GS}\Delta E \tag{A.5}$$

Consider the differential equation for $\beta \in [0, \infty)$,

$$\frac{d\xi}{d\beta} = \left(N_{GS}e^{-\xi/N_{GS}} - 1\right) N_{GS}\Delta E \tag{A.6}$$

with initial condition $\xi(\boldsymbol{\theta}_D, 0) = \eta(\boldsymbol{\theta}_D, 0) = N_{GS}\log N_{TS}$. Noting that $N_{GS}e^{-\xi/N_{GS}} - 1 = N_{GS}P - 1 > 0$, this ODE is integrated to give the following solution:

$$\xi(\boldsymbol{\theta}_D, \beta) = N_{GS}\log\left(N_{GS} + N_{ES}e^{-\beta\Delta E}\right) \tag{A.7}$$

Using Comparison Lemma [173], for all $0 < \beta < \infty$,

$$\eta(\boldsymbol{\theta}_D, \beta) \leq \xi(\boldsymbol{\theta}_D, \beta) \tag{A.8}$$

This proves the upper bound. The lower bound is a direct consequence from monotonicity proved in part 1.

(c) For any $\beta < \infty$

$$\eta(\boldsymbol{\theta}_D, \beta) - N_{GS}\log N_{GS} \leq N_{GS}\log\left(1 + \frac{N_{ES}}{N_{GS}}e^{-\beta\Delta E}\right)$$

For any $\epsilon > 0$, choose a $\beta > \beta^*(\epsilon)$ using Eq (A.3) and observe that,

$$N_{GS} \log \left(1 + \frac{N_{ES}}{N_{GS}} e^{-\beta \Delta E}\right) < \epsilon$$

This proves the third statement. $\square$

Proposition A.1. *Given a straight crack at an angle α, defined as a mesh partition with label $+1$ in the upper sector and -1 in the lower sector, Eq (3.6) holds true in the continuum limit of the mesh when g has the following form where $k < 2$:*

$$g(c_i, l_i) \equiv (2-k)\overline{r}^{k-1} \frac{\overline{g}(\theta_i, l_i)}{r_i^k}$$

Proof. Taking the continuum limit:

$$\begin{aligned}
\lim_{h \to 0} \sum_{P_i \in \mathcal{P}_h} (2-k)\overline{r}^{k-1} \frac{\overline{g}(\theta_i, l_i)}{r_i^k} |P_i| &= \int_A (2-k)\overline{r}^{k-1} \frac{\overline{g}(\theta, l)}{r^k} dA \\
&= \int_{-\pi/2}^{\pi/2} \int_0^{\overline{r}} (2-k)\overline{r}^{k-1} \frac{\overline{g}(\theta, l)}{r^k} r dr d\theta \\
&= \overline{r} \int_{-\pi/2}^{\pi/2} \overline{g}(\theta, l) d\theta
\end{aligned}$$

Choosing the labels as $l = 1$ when $\theta \in [\alpha, \pi/2]$ and $l = -1$ when $\theta \in [-\pi/2, \alpha)$ gets the desired result. $\square$

Proposition A.2. *Eq (3.7) satisfies Eq (3.6)*

Proof. Taking derivative of Eq (3.6) with respect to α on both sides:

$$\overline{g}(\alpha, +1) - \overline{g}(\alpha, -1) = G'(\alpha)$$

Clearly the form provided in Eq (3.7) satisfies the above equation. The variable c can be estimated by substituting this form back in Eq (3.6) and evaluating it for $\alpha = \pi/2$ to get:

$$c = \frac{1}{\pi}\left(-G(\pi/2) + \int_{-\pi/2}^{\pi/2} G'(\theta)H(-G'(\theta))d\theta\right)$$

Similarly, evaluating it for $\alpha = -\pi/2$ gives:

$$c = \frac{1}{\pi}\left(-G(-\pi/2) - \int_{-\pi/2}^{\pi/2} G'(\theta)H(G'(\theta))d\theta\right)$$

Note that both values are equal as:

$$G(\pi/2) - G(-\pi/2) = \int_{-\pi/2}^{\pi/2} G'(\theta)d\theta = \int_{-\pi/2}^{\pi/2} G'(\theta)(H(G'(\theta)) + H(-G'(\theta)))d\theta$$

$$\implies -G(-\pi/2) - \int_{-\pi/2}^{\pi/2} G'(\theta)H(G'(\theta))d\theta = -G(\pi/2) + \int_{-\pi/2}^{\pi/2} G'(\theta)H(-G'(\theta))d\theta$$

□

Proposition A.3. *Given a rough crack represented using its angular position (α) as a function of the radial position, Eq* (3.9) *holds in the continuum limit.*

Proof. As earlier, the crack is defined as a mesh partition with label +1 in the upper sector and −1 in the lower sector. Taking the continuum limit:

$$\begin{aligned}\lim_{h\to 0}\sum_{P_i\in\mathcal{P}_h} g(c_i, l_i)|P_i| &= \int_A (2-k)\overline{r}^{k-1}\frac{\overline{g}(\theta, l)}{r^k}dA \\ &= \int_0^{\overline{r}}\int_{-\pi/2}^{\pi/2} (2-k)\overline{r}^{k-1}\frac{\overline{g}(\theta, l)}{r^k}r d\theta dr\end{aligned}$$

$$= (2-k)\overline{r}^{k-1} \int_0^{\overline{r}} \frac{1}{r^{k-1}} \left(\int_{\alpha(r)}^{\pi/2} \overline{g}(\theta,+1)d\theta + \int_{-\pi/2}^{\alpha(r)} \overline{g}(\theta,-1)d\theta \right) dr$$

Using Eq (3.6),

$$\lim_{h\to 0} \sum_{P_i \in \mathcal{P}_h} g(c_i, l_i)|P_i| = (k-2)\overline{r}^k \int_0^{\overline{r}} \frac{1}{r^{k-1}} G(\alpha(r))dr$$

Normalizing the radial variable from r to $s = r/\overline{r}$,

$$\lim_{h\to 0} \sum_{P_i \in \mathcal{P}_h} g(c_i, l_i)|P_i| = (k-2)\overline{r} \int_0^{1} \frac{1}{s^{k-1}} G(\alpha(\overline{r}s))ds$$

□

Proposition A.4. *For a planar curve defined as a level set function $C = \psi^{-1}(a)$ with $a \in \mathbb{R}$. Following geometric relations are true at any $(x,y) \in C$:*

1. *The normal, $\overrightarrow{n}$:*

$$\overrightarrow{n} = \left.\frac{\nabla\psi}{|\nabla\psi|}\right|_{(x,y)}$$

2. *The curvature, κ,*

$$\kappa = -\nabla . \left.\frac{\nabla\psi}{|\nabla\psi|}\right|_{(x,y)}$$

Proof. Let s denote the arc-length parameterization of the level set. As level set is constant over C, one has, $\frac{\partial^n \psi}{\partial s^n} = 0$ for all n.

1. Taking the case, $n = 1$:

$$\psi_x x_s + \psi_y y_s = 0$$

$$C_s . \nabla \psi = 0$$

By definition of $\partial C / \partial s$ is tangential to the curve. Therefore, $\nabla \psi$ is orthogonal to the curve. The sign is arbitrarily chosen to be positive in the direction of $\nabla \psi$.

2. By definition of curvature of curves, $\kappa \overrightarrow{n} = \frac{\partial^2 C}{\partial s^2}$. The curvature is estimated by considering, $\frac{\partial^2 \psi}{\partial s^2} = 0$.

$$\begin{aligned}
\frac{\partial}{\partial s}\left(\psi_x x_s + \psi_y y_s\right) &= 0 \\
\psi_{xx} x_s^2 + 2\psi_{xy} x_s y_s + \psi_{yy} y_s^2 + \phi_x x_{ss} + \phi_y y_{ss} &= 0 \\
\psi_{xx} x_s^2 + 2\psi_{xy} x_s y_s + \psi_{yy} y_s^2 + \nabla \psi . C_{ss} &= 0 \\
\psi_{xx} x_s^2 + 2\psi_{xy} x_s y_s + \psi_{yy} y_s^2 + \kappa \nabla \psi &= 0
\end{aligned}$$

Using the normals,

$$x_s = \frac{\psi_y}{|\nabla \psi|}, \qquad y_s = -\frac{\psi_x}{|\nabla \psi|}$$

Substituting it back,

$$\begin{aligned}
\frac{\psi_{xx}\psi_y^2 - 2\psi_{xy}\psi_x\psi_y + \psi_{yy}\psi_x^2}{|\nabla \psi|^2} + \kappa \nabla \psi &= 0 \\
\nabla . \frac{\nabla \psi}{|\nabla \psi|} + \kappa &= 0
\end{aligned}$$

□

Theorem IV.3 Consider, $D \subseteq \mathbb{R}^2$ where D is a fixed region. And a open set Ω with smooth boundary, $\Gamma = \partial \Omega$. Then for any Ω, with $\overline{\Omega} \neq \phi$ or D, to be the extremizer of functional:

$$F(\Omega) = f_1 \,\mathrm{Area}(\Omega) + f_0 \,\mathrm{Area}(D - \Omega) + \gamma \,\mathrm{length}(\Gamma)$$

following condition is satisfied on the boundary, Γ, in the interior of domain D:

$$\gamma\kappa + f_1 - f_0 = 0 \tag{A.9}$$

Moreover, in the absence of any fixed/Dirichlet boundary conditions, Γ makes perpendicular intersections with ∂D.

Proof. The level set approach of [86] is adapted here. Consider $\psi(x, y)$, a Lipschitz continuous Level-set function satisfying:

$$\begin{aligned} \psi(x,y) > 0 \quad & \text{for } (x,y) \in \Omega \\ \psi(x,y) = 0 \quad & \text{for } (x,y) \in \partial\Omega \\ \psi(x,y) < 0 \quad & \text{for } (x,y) \in D - \overline{\Omega} \end{aligned} \tag{A.10}$$

The boundary is estimated as the zero level set, $\Gamma = \psi^{-1}(0)$. The normal, $\overrightarrow{n}$ and curvature, κ are estimated in Proposition A.4. The geometric properties of Ω is estimated as follows:

$$\begin{aligned} \mathrm{length}(\partial\Omega) &= \int\int \delta(\psi(x,y))|\nabla\psi(x,y)|dxdy \\ \mathrm{Area}(\partial\Omega) &= \int\int H(\psi(x,y))dxdy \end{aligned}$$

where $H(x)$ is the Heaviside function and $\delta(x)$ is the Dirac delta function. An important property that will be used later in the proof is that the distributional derivative of $H(x)$ is $\delta(x)$. Writing $F(\Omega)$ using the level set:

$$F(\Omega) = \int\int \gamma\delta(\psi(x,y))|\nabla\psi(x,y)| + (f_1 H(\psi) + f_0 H(-\psi))\, dxdy$$

Taking the Gateaux derivative with respect to ψ in the direction of $\widetilde{\psi}$:

$$\left(\frac{\partial F}{\partial \psi}, \widetilde{\psi}\right) = \int\limits_D \gamma \left(\delta'(\psi)|\nabla\psi|\widetilde{\psi} + \frac{\delta(\psi)\nabla\psi.\nabla\widetilde{\psi}}{|\nabla\psi|}\right) + (f_1\delta(\psi) - f_0\delta(-\psi))\,\widetilde{\psi}dxdy$$

Applying Divergence theorem with $\overrightarrow{n}_{\partial D}$ as the normal to ∂D:

$$\begin{aligned} &= \int\limits_D \left(\gamma\left(\delta'(\psi)|\nabla\psi| - \nabla.\frac{\delta(\psi)\nabla\psi}{|\nabla\psi|}\right) + (f_1 - f_0)\,\delta(\psi)\right)\widetilde{\psi}dxdy \\ &\quad + \int\limits_{\partial D} \delta(\psi)\gamma\frac{\nabla\psi}{|\nabla\psi|}.\overrightarrow{n}_{\partial D}ds \\ &= \int\limits_D \delta(\psi)\left(-\gamma\nabla.\frac{\nabla\psi}{|\nabla\psi|} + (f_1 - f_0)\right)\widetilde{\psi}dxdy + \int\limits_{\partial D}\delta(\psi)\gamma\frac{\nabla\psi}{|\nabla\psi|}.\overrightarrow{n}_{\partial D}ds \end{aligned}$$

where the following fact (in the sense of distributions) is used:

$$\nabla\left(\delta(\psi)\nabla\psi/|\nabla\psi|\right) = \delta'(\psi)|\nabla\psi| + \delta(\psi)\nabla.\left(\nabla\psi/|\nabla\psi|\right)$$

Setting the Gateaux derivative to zero gives following results:

$$\begin{aligned} \gamma\kappa + f_1 - f_0 &= 0 \qquad \text{on} \qquad \Gamma \cap \operatorname{Int}(D) \\ \gamma\overrightarrow{n}.\overrightarrow{n}_{\partial D} &= 0 \qquad \text{on} \qquad \Gamma \cap (\partial D)_{\text{Neumann}} \end{aligned}$$

It should be noted that on the boundary, if $\Gamma \cap (\partial D)_{\text{Neumann}}$ are disconnected points then $\overrightarrow{n}$ and $\overrightarrow{n}_{\partial D}$ orthogonal to each other. However, they are not allowed to be parallel to each other. This creates an unimaginable situation where Γ intersects ∂D but does not merge with it. This contradiction arise from the way the surface energy is introduced through levels sets. Notice that there is no surface energy on ∂D when it does not intersect Γ while that is not the case when Ω merges with the boundary. It stands to reason, that the surface energy of Γ should not be included when it it lies on the boundary. There is no attempt here to resolve this problem with complete mathematical rigour. However, following intuitive picture can be used to convince oneself that this contradiction does not interfere with the physical process described here. Consider an open set $D_0 \supset D$ and extend ψ to D_0, now the merger of grain to ∂D can be understood as a portion of Γ being present in $D_0 - D$.

$\square$

Theorem IV.9 Consider the following minimizer

$$\Gamma_{t+\Delta t} = \arg\min_{\Gamma} \left(\alpha F(\Gamma) + \frac{\text{dist}(\Gamma, \Gamma_t)}{2\Delta t} \right)$$

where Γ and Γ_t are closed curves and $F(\Gamma) \equiv F(\Omega)$ with Ω denoting the interior region of the curve. It has following property

$$\Gamma_{t+\Delta t} = \Gamma_t + (v\Delta t)\overrightarrow{n}, \qquad v = \alpha\left(\gamma\kappa + (b_1 - b_0)\right)$$

Proof. The same level set approach as in the proof of Theorem IV.3 is followed here. The level set for Γ_t is denoted as ψ_t and the level set for Γ is denoted as ψ. All the properties mentioned in Eq (A.10) are applied to the respective level sets. In addition, the level set functions ψ and ψ_t are endowed with a special property that they represent the negative signed distance to the curves $\psi^{-1}(0)$ and $\psi_t^{-1}(0)$, respectively. In concise notations: $p_{\Gamma_t}(x, y) = |\psi_t(x, y)|$ and $p_\Gamma(x, y) = |\psi(x, y)|$ where p is the perpendicular distance from respective curves as defined in Eq (4.5). It is clear that this extra condition does not interfere with the properties defined in Eq (A.10). Moreover, the proof for theorem IV.3 remains unaffected as it only depends on the local behavior of cure at the boundary.

The regularizer in the energy can be approximately represented as:

$$\begin{aligned}\frac{\text{dist}(\Gamma,\Gamma_t)}{2\Delta t} &\approx \frac{1}{2\Delta t}\int\limits_{\Gamma_t}\int\limits_{0}^{|\Gamma(s)-\Gamma_t(s)|} p\,dp\,ds \\ &= \int\int |\psi_t|\left(H(\psi)H(-\psi_t)+H(-\psi)H(\psi_t)\right)dxdy\end{aligned}$$

The term $H(\psi)H(-\psi_t)+H(-\psi)H(\psi_t)$ has a value of 1 in the region between the zero level sets of ψ_t and ψ. Noting that ψ_t is constant and taking the Gateaux derivative with respect to ψ in the direction of $\widetilde{\psi}$:

$$\begin{aligned}\left(\frac{\text{dist}(\Gamma,\Gamma_t)}{\Delta t},\widetilde{\psi}\right) &= -\frac{1}{\Delta t}\int\int \delta(\psi)|\psi_t|\left(H(\psi_t)-H(-\psi_t)\right)\widetilde{\psi}dxdy \\ &= -\frac{1}{\Delta t}\int\int \delta(\psi)\psi_t\widetilde{\psi}dxdy\end{aligned}$$

The Gateaux derivative of the functional described in Eq 4.6 is now estimated as:

$$\begin{aligned}\left(\alpha F(\Gamma)+\frac{\text{dist}(\Gamma,\Gamma_t)}{2\Delta t},\widetilde{\psi}\right) &= \int\limits_{D}\delta(\psi)\left(-\gamma\nabla.\frac{\nabla\psi}{|\nabla\psi|}+(f_1-f_0)-\frac{\psi_t}{\Delta t}\right)\widetilde{\psi}dxdy \\ &+ \int\limits_{\partial D}\delta(\psi)\gamma\frac{\nabla\psi}{|\nabla\psi|}.\overrightarrow{n}_{\partial D}ds\end{aligned}$$

Noting that for distance level sets, $\psi_t\delta(\psi)=(\Gamma-\Gamma_t).\overrightarrow{n}$ (in terms of distributions), setting the Gateaux derivative to zero yields following equation in the interior:

$$\alpha(\gamma\kappa+f_1-f_0)-\frac{\psi_t}{\Delta t}=0$$

The above equation can then be rearranged in terms of the extremizer, $\Gamma_{t+\Delta t}\equiv\Gamma$,

$$\Gamma_{t+\Delta t}=\Gamma_t+\alpha(\gamma\kappa+f_1-f_0)\Delta t\overrightarrow{n}$$

□

Theorem IV.16 In the setup of Theorem IV.3, consider the following functional:

$$F(\Omega) = f_1 \operatorname{Area}(\Omega) + f_0 \operatorname{Area}(D - \Omega) + \int\limits_{\Gamma} \gamma(\overrightarrow{n}(s))ds$$

where $\gamma(\overrightarrow{n})$ is the surface energy in the direction of unit normal, $\overrightarrow{n}$. The extremizer satisfies the following condition on the boundary, Γ:

$$\nabla.\left(\gamma\overrightarrow{n} + (\nabla\gamma.\overrightarrow{t})\overrightarrow{t}\right) + f_1 - f_0 = 0 \tag{A.11}$$

where $\overrightarrow{n}$ and $\overrightarrow{t}$ are normal and tangents of Γ. Moreover, in the absence of any fixed/Dirichlet boundary conditions, Γ intersects ∂D at an non-zero angle such that following condition holds true:

$$\gamma\overrightarrow{n}.\overrightarrow{n}_{\partial D} + (\nabla\gamma.\overrightarrow{t})\overrightarrow{t}.\overrightarrow{n}_{\partial D} = 0$$

Proof. For most part, the proof follows from the proof of Theorem IV.3, however, Gateaux derivative of the surface term needs to be calculated:

$$\left(\int\limits_{D} \gamma\left(\frac{\nabla\psi}{|\nabla\psi|}\right)\delta(\psi)|\nabla\psi|, \widetilde{\psi}\right) = \int\limits_{D} \gamma\left(\frac{\nabla\psi}{|\nabla\psi|}\right)\left(\delta'(\psi)|\nabla\psi|\widetilde{\psi} + \frac{\delta(\psi)\nabla\psi.\nabla\widetilde{\psi}}{|\nabla\psi|}\right) + \delta(\psi)\left(\nabla\gamma.\nabla\widetilde{\psi} - \frac{(\nabla\gamma.\nabla\psi)(\nabla\psi.\nabla\widetilde{\psi}))}{|\nabla\psi|^2}\right)dxdy$$

Using Divergence theorem::

$$= \int\limits_{D} \gamma\delta'(\psi)|\nabla\psi|\widetilde{\psi} - \nabla.\left(\gamma\delta(\psi)\frac{\nabla\psi}{|\nabla\psi|} + \delta(\psi)\nabla\gamma - \delta(\psi)\frac{(\nabla\gamma.\nabla\psi)\nabla\psi}{|\nabla\psi|^2}\right)\widetilde{\psi}dxdy + \int\limits_{\partial D} \delta(\psi)\left(\frac{\gamma}{|\nabla\psi|}\nabla\psi + \nabla\gamma - \frac{\nabla\gamma.\nabla\psi}{|\nabla\psi|^2}\nabla\psi\right).\overrightarrow{n}_{\partial D}\widetilde{\psi}ds$$

Simplifying:

$$= \int_D -\delta(\psi)\nabla.\left(\gamma\frac{\nabla\psi}{|\nabla\psi|} + \nabla\gamma - \frac{(\nabla\gamma.\nabla\psi)}{|\nabla\psi|^2}\nabla\psi\right)\widetilde{\psi}dxdy$$
$$+ \int_{\partial D} \delta(\psi)\left(\frac{\gamma}{|\nabla\psi|}\nabla\psi + \nabla\gamma - \frac{\nabla\gamma.\nabla\psi}{|\nabla\psi|^2}\nabla\psi\right).\overrightarrow{n}_{\partial D}\widetilde{\psi}ds$$

Observe that $\nabla\gamma - (\nabla\gamma.\overrightarrow{n})\overrightarrow{n}$ is the component of $\nabla\gamma$ that is tangential to the Γ. The rest follows as the proof of Theorem IV.3.

□

Proposition IV.12: Consider the set of all lines with rational slope p/q (p and q are relative prime) that pass through atleast one grid point (i.e a point $(a,b) \in \mathbb{I}^2$). These lines are regularly placed with following distance between two nearest lines:

$$\Delta\varrho = \frac{1}{\sqrt{p^2+q^2}} \tag{A.12}$$

Proof. It is easy to see that if the line passes through one grid point, then it passes through infinitely many grid points as if $(a,b) \in \mathbb{I}^2$ lie on the line then so does $(a+q, b+p) \in \mathbb{I}^2$. To see these lines are regularly placed, observe that these lines can be constructed by choosing a single line and then constructing the next parallel line as the one passing through a grid point nearest to the initial line. Without loss of generality, consider that the initial line is passing through the origin,

$$L := qy - px = 0$$

The distance of any general point (x_0, y_0) from line L is given as:

$$d_L(x_0,y_0) = \frac{|qy_0 - px_0|}{\sqrt{p^2+q^2}}$$

Clearly, if p, q, x_0, y_0 are integers, then the minimum possible non-zero value of $|qy_0 - px_0| = 1$. In fact, the minimum value is exactly 1. This is true because $qy_0 - px_0 = 1$ is a diaphantine equation and $\gcd(p, q) = 1$. This is a sufficient condition for an integral solution of the above equation to exist. This proves that the minimum non zero distance between a vertex position and the line L is $\frac{1}{\sqrt{p^2+q^2}}$. $\square$

APPENDIX B

Pseudo-Codes

B.1 Conformal mesh generation

The data structure and the pseudo-code for the mesh labeling algorithm are presented here. There are 2 kinds of data that are required in this algorithm: (1) FE Mesh data and (2) Image data. It is recommended to calculate and save all the variables as defined below at the time of mesh generation and be passed on to mesh labeling algorithm. The FE mesh data structure is pre-processed to include the following variables:

- *Dimension*: Value is 2 for 2D and 3 for 3D
- *MeshSize*: The x-dimension of mesh is in the range $[0, \mathit{MeshSize}(1)]$ and the y-dimension is in the range $[0, \mathit{MeshSize}(2)]$. For 3D mesh, the z-direction is in the range $[0, \mathit{MeshSize}(3)]$.
- *Coordinate*: Location of each node of the mesh
- *Nnode*: Number of nodes

- *Connectivity*: Contains tuples with nodes of each element
- *Nelement*: Number of elements
- *Neighbor*: In 3D meshes this data contains each pair of elements which share a face. In case of 2D, it contains each pair of elements which share an edge.
- *Npair*: Number of Neighbors
- *ElementVolume*: Defined only for 3D meshes and contains the volume of each element.
- *Face*: Defined only for 3D meshes and contains tuples with nodes of the shared shared faces (indexed as the *Neighbor* data)
- *FaceArea*: Area of face defined in *Face* data
- *FaceNormal*: Unit Normal of face defined in *Face* data
- *ElementArea*: Defined only for 2D meshes and contains the area of each element.
- *Edge*: Defined only for 2D meshes and contains tuples with nodes of shared edges (indexed as the *Neighbor* data)
- *EdgeLength*: Length of each edge defined in *Edge* data
- *EdgeNormal*: Unit Normal of each edge defined in *Edge* data

The Image data structure includes the following variables:

- *ImageSize*: The tuple (N_x, N_y, N_z) represents the Voxel size of the image in each direction. In case of 2D, this variable contains a pair (N_x, N_y)

- *XGridData*: Contains the x-coordinate value of the Voxelated/Pixelated image with size determined by *ImageSize*.
- *YGridData*: Contains the y-coordinate value of the Voxelated/Pixelated image with size determined by *ImageSize*.
- *ZGridData*: Defined only for 3D image data. Contains the z-coordinate value of the Voxelated/Pixelated image with size determined by *ImageSize*.
- *LabelGridData*: Label values at each grid point.
- *NLabel*: Number of Labels
- *LabelValues* Contains the label values of enumerated from 1 to Nlabels

As discussed in the main text, this method uses an energy minimization approach. Pseudo-code for estimation of Data cost amd Smooth cost is presented in Algorithm 1 and Algorithm 2, respectively. The energy minimization is carried out using the GCO library developed in [7]. The pseudocode for using this library is presented in Algorithm 3

Algorithm 1 Set-up Data cost (2D/3D)

```
1: $M \leftarrow$ Mesh data                                  ▷ Import mesh data
2: $I \leftarrow$ Image data                                ▷ Import Experimental image data
3: $Unary \leftarrow$ zeros($M.Nlabel$,$M.Nelement$)        ▷ Initialize Data cost variable as an matrix of zeros
4: if $M.Dimension == 3$ then                               ▷ 3D Case
5:     $temp \leftarrow$ mode($M.ElementVolume$)            ▷ Evaluate mode of the distribution of element volume
6:     $Factor \leftarrow \alpha(1 - \exp(-M.ElementVolume/temp))$   ▷ Evaluate Eq (2.2)
```

```
7:      for index1 ← 1 to M.Nelement do                                   ▷ Loop over each element
8:          xpos ← mean(M.Coordinate(M.Connectivity(index1,:),1))   ▷ Coordinates of center of the element
9:          ypos ← mean(M.Coordinate(M.Connectivity(index1,:),2))
10:         zpos ← mean(M.Coordinate(M.Connectivity(index1,:),3))
11:         nx = floor(xpos × I.ImageSize(1)/M.MeshSize(1))+1   ▷ Location of center with respect to image grid data
12:         ny = floor(ypos × I.ImageSize(2)/M.MeshSize(2))+1
13:         nz = floor(zpos × I.ImageSize(2)/M.MeshSize(3))+1
14:         for index2 ← 1 to I.Nlabel do                          ▷ Loop over each Label value
15:             if LabelValues(index2) == I.LabelGridData(nx,ny,nz) then
16:                 Unary(index2,index1) ← 0        ▷ No cost if the Label matches the image data
17:             else
18:                 Unary(index2,index1) ← Factor(index1)     ▷ Add cost determined by Eq(2.2)
19:             end if
20:         end for
21:     end for
22: else
23:     temp ← mode(M.ElementArea)          ▷ Evaluate mode of the distribution of element area
24:     Factor ← $\alpha(1 - \exp(-M.ElementArea/temp))$                ▷ Evaluate Eq (2.2)
25:     for index1 ← 1 to M.Nelement do                                   ▷ Loop over each element
26:         xpos ← mean(M.Coordinate(M.Connectivity(index1,:),1))   ▷ Coordinates of center of the element
27:         ypos ← mean(M.Coordinate(M.Connectivity(index1,:),2))
```

```
28:         $nx$ = floor($xpos \times I.ImageSize(1)/M.MeshSize(1)$)+1  ▷ Location of center
    with respect to image grid data
29:         $ny$ = floor($ypos \times I.ImageSize(2)/M.MeshSize(2)$)+1
30:         for $index2 \leftarrow 1$ to $I.Nlabel$ do              ▷ Loop over each Label value
31:             if $LabelValues(index2) == I.LabelGridData(nx,ny,nz)$ then
32:                 $Unary(index2,index1) \leftarrow 0$       ▷ No cost if the Label matches the
    image data
33:             else
34:                 $Unary(index2,index1) \leftarrow Factor(index1)$    ▷ Add cost determined
    by Eq(2.2)
35:             end if
36:         end for
37:     end for
38: end if
39: return $Unary$
```

Algorithm 2 Set-up Smooth cost (2D/3D)

```
 1: $M \leftarrow$ Mesh data                                         ▷ Import mesh data
 2: $I \leftarrow$ Image data                             ▷ Import Experimental image data
 3: $Pairwise \leftarrow$ zeros($Nelement$)        ▷ Initialize Smooth cost variable as a square
    matrix of zeros
 4: if $M.Dimension == 3$ then                                              ▷ 3D Case
 5:     for $index \leftarrow 1$ to $M.Npair$ do
 6:         $paircost \leftarrow g(M.FaceNormal(index)) \times M.FaceArea(index)$      ▷ Evaluate
    Eq((2.3))
 7:         $Pairwise(M.Neighbor(index,1),M.Neighbor(index,2)) \leftarrow paircost$            ▷
    Update $Pairwise$ variable
 8:     end for
```

```
9:  else                                                              ▷ 2D Case
10:     for $index \leftarrow 1$ to $M.Npair$ do
11:         $paircost \leftarrow$ g($M.EdgeNormal(index)$)× $M.EdgeArea(index)$      ▷ Evaluate
        Eq((2.3))
12:         $Pairwise(M.Neighbor(index,1),M.Neighbor(index,2)) \leftarrow paircost$      ▷
        Update $Pairwise$ variable
13:     end for
14: end if
15: return $Pairwise$
```

Algorithm 3 Using GCO Library for estimating the labeling

```
1: Input: $Unary$, $Pairwise$, $LabelValues$, $NLabel$, $Nelement$
2: $h \leftarrow$ GCO_Create($NElement$,$NLabel$)
3: GCO_SetDataCost($h$,$Unary$)
4: $Labelcost \leftarrow$ ones($Nlabel$) - Id($Nlabel$)
5: GCO_SetSmoothCost($h$),$Labelcost$)
6: GCO_SetNeighbors($h$),$Pairwise$)
7: GCO_Expansion(h)
8: $Label \leftarrow LabelValues$(GCO_GetLabeling($h$))
9: return $Label$
10: end
```

B.2 Modeling microstructural fracture

The data structure used in the algorithm is presented. There are two kinds of data that are required in this algorithm: (1) FE Mesh data and (2) Image data. It is recommended to calculate and save all the variables as defined below at the time of mesh generation and be passed on to the mesh labeling algorithm.

FE Mesh Data: The basic structure for any FE Mesh includes folling bare-bone data:

- *Nnode*: Number of nodes of the mesh
- *Coordinate*: Location of each node of the mesh
- *Nelement*: Number of elements
- *Connectivity*: Contains tuples with nodes of each element

It is post-processed to include the following derived quantitites:

- *ElementCenter*: Coordinates of a point in the interior of the element (possibly centroid)
- *Neighbor*: It contains each pair of elements which share an edge.
- *Npair*: Number of Neighbors
- *ElementArea*: Area of each element
- *Edge*: It contains tuples with nodes of shared edges (indexed as the *Neighbor* data)
- *EdgeLength*: Length of each edge defined in *Edge* data
- *EdgeNormal*: Unit Normal of each edge defined in *Edge* data

Finally, the smooth labeling procedure [81] provides the *GrainID* of each element in the mesh, based on the experimental image.

Material Property Data: The Polycrystalline material properties are stored the form of the following variables:

- *GrainOrientation*: Axis-Angle representation for each value of GrainID

- *SurfaceEnergyParameter*: Surface energy parameters for each value of GrainID
- *ShearModulus*: Effective Shear Modulus of the specimen
- *PoissonRatio*: Effective Poisson's Ratio of the specimen
- *IntergranularEnergy*: γ^{ig} for each pair of grains

Next the basic Pseudocode modules are presented as used in 3.3.2:

Algorithm 4 Set-up Neighbor Cost

```
1: $M \leftarrow$ Mesh data                                   ▷ Import mesh data
2: $P \leftarrow$ Material Property Data          ▷ Import Material Property Data
3: $NC \leftarrow$ zeros($Nelement$)   ▷ Initialize Smooth cost variable as a square matrix of
   zeros
4: $dihedral \leftarrow$ zeros($Npair$,1) ▷ Initialize Smooth cost variable as a square matrix of
   zeros
5: for $i \leftarrow 1$ to $M.Npair$ do                          ▷ Loop over all shared edges
6:     $Grain1 \leftarrow M.GrainID(M.Neighbor(i,1)$
7:     $Grain2 \leftarrow M.GrainID(M.Neighbor(i,2)$
8:     if $Grain1 == Grain2$ then                              ▷ Transgranular edge
9:         $dihedral(i) \leftarrow \psi_D$ (evaluated using Eq(3.12))
10:        $SurfaceDensity \leftarrow \widetilde{\gamma}$ (evaluated using Eq(3.13))
11:    else                                                     ▷ Intergranular edge
12:        $SurfaceDensity \leftarrow P.IntergranularEnergy(Grain1, Grain2)$
13:    end if
14:    $NC(M.Neighbor(i,1),M.Neighbor(i,2)) \leftarrow 2SurfaceDensity \times M.EdgeLength(i)$
   ▷ Update Neighbor Cost
15: end for
16: return $NC$, $dihedral$
```

Algorithm 5 Set-up Data Cost

```
1: **Input:** $\overline{r}$, $\sigma_T$, $\beta$, $a$ ▷ Input domain radius, Tensile stress, crack angle and length
2: $M \leftarrow$ Mesh data                                              ▷ Import mesh data
3: $P \leftarrow$ Material Property Data                       ▷ Import Material Property Data
4: $K_I, K_{II} \leftarrow$ Evaluate Eq(3.14) using argument $\beta$
5: $R\text{-}Index \leftarrow$ (distance($M.ElementCenter$,$Cracktip$)¡ $\overline{r}$) ▷ Index of elements within
   domain defined by $\overline{r}$
6: $(r_i, \theta_i)_{i \in R\text{-}Index}$ = cartesian2polar($M.ElementCenter$-$Cracktip$)          ▷ Find Polar
   coordinate of each element
7: **for** i $\in R\text{-}Index$ **do**
8:     $G'(\theta_i) \leftarrow$ Evaluate Eq(3.8) using arguments $K_I, K_{II}$,$P.ShearModulus$ and
   $P.PoissonRatio$.
9:     $g(i, .) \leftarrow$ Evaluate Eq(3.7)
10:    $DC_0(., i) \leftarrow$ Evaluate Eq(3.17) using arguments $g(i, .)$, $r_i$, $M.ElementArea(i)$
11: **end for**
12: $UpperBoundaryIndex \leftarrow$ ($M.ElementCenter \in$ UpperBoundary)
13: $LowerBoundaryIndex \leftarrow$ ($M.ElementCenter \in$ LowerBoundary)
14: $DC_{BC}(2,UpperBoundaryIndex) \leftarrow$ LargeNumber
15: $DC_{BC}(1,LowerBoundaryIndex) \leftarrow$ LargeNumber
16: **return** $DC_0$, $DC_{BC}$,$R\text{-}Index$
```

Algorithm 6 Estimate Crack Path

```
1: **Input:** $DC_0$, $DC_{BC}$, $NC$, $R\text{-}Index$                              ▷ Input variables
2: $h \leftarrow$ GCO_Create($NNodes$ = length(R-Index),NLabel=2)
3: GCO_SetDataCost($h$,$DC_0 + DC_{BC}$)
4: $LC \leftarrow$ ones(2) - Id(2)
5: GCO_SetSmoothCost($h$,$LC$)
6: GCO_SetNeighbors($h$,$NC(R\text{-}Index, R\text{-}Index)$)          ▷ Use Neighbor cost only for
```

```
   vertices in radial domain
7: GCO_Expansion(h)
8: $Label \leftarrow$ GCO_GetLabeling($h$)   ▷ This outputs label in {1,2} format. Change it
   to {+1,-1} format, if desired.
9: return $Label$
```

Algorithm 7 Evaluate total energy

```
1: Input: $DC_0$, $NC$, $Label$, $R\text{-}Index$
2: $EnergyRelease \leftarrow -\left(\sum_j DC(Label, j)\right)$
3: x ← $(Label == 1)$   ▷ TypeCast bool to float type for next calculation
4: y ← $(Label == 2)$   ▷ Or equivalently, $Label == -1$. TypeCast bool to float type
5: $NC_R \leftarrow NC(R\text{-}Index, R\text{-}Index)$   ▷ Use Neighbor cost only for vertices in radial
   domain
6: $SurfaceEnergy \leftarrow \sum_{ij} x_i y_j (NC_R)_{ij}$
7: return $\varepsilon(\Gamma) \leftarrow SurfaceEnergy$ - $EnergyRelease$
```

B.3 Multiphase flow

Only the pseudo-codes for Quadtree method is presented. The pixelated case can be treated as a specific instance of the Quadtree method with only one layer. Computational cost in this procedure can be reduced by pre-processing and saving the complete quadtree structure before solving the problem. This information can be saved in form of following variables:

- *LayerSize*: The size of i^{th} layer from top is given as $4^{i-1} \times N_d^2$.
- N_T: Total number of nodes, estimated as $\sum_{i=1}^{N_p}(LayerSize)$.
- *LayerIndex*: Nodes are enumerated sequentially starting from top layer to bottom layer. The nodes in the i^th layer ranges from $LayerIndex(i) + 1$ to $LayerIndex(i)+LayerSize(i)$.

- *Child*: List of children of each node. Nodes in bottom layer have no child.
- *Parent*: List of parent of each node. Nodes in top layer have no parent.
- *Descendent*: List of all children, children of children and so on for each node.
- *NumberDescendent*: Number of descendant of each node
- *LeafChild*: List of descendants that are in the bottom layer.
- *NumberLeafChild*: Number of descendants of each node.
- $\overline{\mathcal{N}}$: A list of neighbors of each node (within the same layer) inlcuding the node itself. This neighborhood is used to locate regions with similar labels. The set of neighbors for any node can be determined by a user prescribed strategy and it need not be same as the neighbors in the Crofton's relation.
- *Coord*: Coordinates of each node.
- *DistMap*: Square matrix with distance between each pixel.
- *Adjacency*: A block diagonal matrix with each block being the adjacency of the connections within the same layer. These connections are made using the Crofton's stencil, $\mathcal{N}_R$.

The multiphase flow is simulated by successively iterating over each time-step. The pseudocode for each time-step is presented in Algorithm 9. As before, the optimization is carried out using the GCO library developed in [7]. The pseudocode for using this library is presented in Chapter II.
Algorithm uses following important temporary variables. *NodeType*: is an array taking values in $\{-1, 0, 1\}$ for each node, v of the Quadtree. *NodeType*(v)=1 if the descendants of v and its neighbors have the same label but v's parent does not have

this property, *NodeType*(v)=0 if one of its ancestor is of *NodeType*=1, *NodeType*(v)=-1 if one of its descendants is of *NodeType*=1 *NodeLabel*: is an array which assign a label to each node of graph. The label of last layer's nodes are prescribed/estimated by the algorithm. The label of nodes with *NodeType*=1 and 0 is assigned based on its respective *LeafChild*. The label of nodes with *NodeType*=2 is assigned arbitrarily. *ActiveParentNode*: is an array which is initialized as the index of each node. If the *NodeType* of a node is 0, then the ActiveParentNode of that node is set as its ancestor's index which is of *NodeType*=1.

Algorithm 8 EstimateCost

```
1:  Input: $CurrLabel$, $\Delta t$
2:  $NodeType \leftarrow \mathbf{1}_{N_t \times 1}$                    ▷ Initialize variables
3:  $ActiveNodeLabel \leftarrow \mathbf{0}_{N_T \times 1}$
4:  $ActiveParentNode \leftarrow [1, ..., N_T]^T$
5:  for $i = 1 : N_p - 1$ do
6:      for $j = 1$ :$LayerSize(i)$ do
7:          $CurrNode \leftarrow LayerIndex(i) + j$
8:          if $NodeType\ (CurrNode) == 1$ then
9:              if Leaves of $CurrNode$'s neighbors have same label then
10:                 Set $NodeType$ of $CurrNode$'s descendant as 0
11:                 Set $ActiveParentNode$ of $CurrNode$'s descendants as $CurrNode$
12:                 Set $ActiveNodeLabel$ of $CurrNode$ and descendants as the leaf's label
13:             else
14:                 Set $NodeType$ of $CurrNode$ as $-1$
15:             end if
16:         end if
17:     end for
```

```
18: end for
19: $ActiveNodeIndex \leftarrow$ Index of nodes with $NodeType$=1
20: $InactiveChildIndex \leftarrow$ Index of nodes with $NodeType$=0
21: Set $NewIndex$ to map $ActiveNodeIndex$ to $\{1, ..., |ActiveNodeIndex|\}$
22: $Pairwise \leftarrow Adjacency(ActiveNodeIndex, ActiveNodeIndex)$    ▷ Initiate $Pairwise$
    cost
23: for $v \in InactiveChildIndex$ do
24:     $temp \leftarrow Adjacency(ActiveNodeIndex, v)$
25:     $vParentIndex \leftarrow NewIndex(ActiveParentNode(v))$
26:     $Pairwise(:, vParentIndex) += temp$
27:     $Pairwise(vParentIndex, :) += temp^T$
28: end for
29: $Unary \leftarrow \mathbf{0}_{N_L \times N_T}$
30: for $i \in [1, ..., N_L]$ do
31:     $CurrNodes \leftarrow (CurrLabel == l_i)$
32:     $NotCurrNodesIndex \leftarrow$ Index of nodes s.t. with $CurrNodes$==0
33:     if there is atleast one node with label $\ell$ then
34:         for $v \in NotCurrNodesIndex$ do
35:             $Unary(\ell, v) \leftarrow$ Area$(v) \times$ ( $f(\ell)$ + min $DistMap(CurrNodes, v)$)    ▷
    Define $p$ (Eq(4.5)) as the minimum distance to the boundary $\Gamma$
36:         end for
37:     end if
38: end for
39: return $Unary$, $Pairwise$, $ActiveNodeIndex$
```

Algorithm 9 Iterative step for Multiphase flow

```
1: Input: $CurrLabel$, $\Delta t$
2: $Unary$, $Pairwise$, $ActiveNodeIndex \leftarrow$ EstimateCost($CurrLabel$, $\Delta t$)
```

```
3: NewActiveNodeLabel ← minimize using GCO: Unary, Pairwise  ▷ Labeling step
   on Quadtree graph with reduced nodes
4: Estimate NewLabel from NewActiveNodeLabel  ▷ Transpose the Labels of Active
   node to the respective leaves
5: if NewLabel == CurrLabel then
6:     Increase $\Delta t$ and goto step 2
7: end if
8: return NewLabel, $\Delta t$
```

B.4 Solving differential equations on Quantum annealers

Algorithm 10 Box Algorithm

```
1: Problem definition: Calculate $\mathbf{S_i}$
2: Initialize $\{u_i^c\}$, $r$
3: Estimate $H$, $\hat{J}$ and $\widetilde{J}$
4: Find embedding: Logical graph $\xrightarrow{\text{embed}}$ Physical graph
5: while $r > r_{\text{min}}$ do
6:     Update $\widetilde{J}$ for current $(u_i^c, r)$
7:     Anneal for $\{q_j^i\}$
8:     Map to $\mathbf{a}^{min}$, $(\Pi_N)_{min}$
9:     if $(\Pi_N)_{min} < \Pi_N[\mathbf{u^c}]$ then
10:        $u_i^c = \mathbf{a}^{min}$ (Translation step)
11:    else
12:        $r = \frac{r}{2}$ (Contraction step)
13:    end if
14: end while
15: end
```

B.5 Boltzmann Machine

Algorithm 11 *EstimateDerivative*: Estimation of Gradient and Hessian

```
1: $\{\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{DS}\}, \{q(\boldsymbol{v}_1), \ldots, q(\boldsymbol{v}_D)\} \leftarrow$ Batch Data
2: $\{\boldsymbol{S}\} \leftarrow$ Sample state of $\mathcal{G}$
3: Estimate random variable $\nabla_{\boldsymbol{\theta}} E$ from $\{\boldsymbol{S}\}$
4: Estimate $\mathbb{E}(\nabla_{\boldsymbol{\theta}} E)$, $\mathrm{Cov}(\nabla_{\boldsymbol{\theta}} E)$
5: for $i \leftarrow 1$ to $DS$ do
6:     $[\boldsymbol{v}^I, \boldsymbol{v}^O] \leftarrow \boldsymbol{v}_i$
7:     Update Parameters of $\mathcal{G}_{HO}$ and $\mathcal{G}_H$
8:     $\{\boldsymbol{h}\} \leftarrow$ Sample state of $\mathcal{G}_H$
9:     Estimate random variable $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{v})$ from $\{[\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h}]\}$
10:    Estimate $\mathbb{E}(\nabla_{\boldsymbol{\theta}} E|\boldsymbol{v})$, $\mathrm{Cov}(\nabla_{\boldsymbol{\theta}} E|\boldsymbol{v})$
11:    $\{[\widetilde{\boldsymbol{v}}^O, \widetilde{\boldsymbol{h}}]\} \leftarrow$ Sample state of $\mathcal{G}_{HO}$
12:    Estimate random variable $\nabla_{\boldsymbol{\theta}} E(\boldsymbol{v}^I)$ from $\{[\boldsymbol{v}^I, \widetilde{\boldsymbol{v}}^O, \widetilde{\boldsymbol{h}}]\}$
13:    Estimate $\mathbb{E}(\nabla_{\boldsymbol{\theta}} E|\boldsymbol{v}^I)$, $\mathrm{Cov}(\nabla_{\boldsymbol{\theta}} E|\boldsymbol{v}^I)$
14: end for
15: Evaluate Eq(6.3) and Eq(6.4)
16: end
```

Algorithm 12 Optimization Step (per epoch)

```
1: Input: $M$ (Number of Batches), $\boldsymbol{\theta}^{(t)}$ (Current Parameters), $\{\eta, \lambda, \nu\}$ (Learning parameters)
2: Partition $\mathcal{D}$ into $M$ partitions $\mathcal{D}_1, ..., \mathcal{D}_M$
3: For all $\{i \in 1, ..., M\}$, define $q_i : \mathcal{D}_i \to \mathbb{R}$ such that $q_i(\boldsymbol{v}_k) = q(\boldsymbol{v_k}) / \sum_{\boldsymbol{v}_j \in \mathcal{D}_i} q(\boldsymbol{v}_j)$ for all $\boldsymbol{v}_k \in \mathcal{D}_i$
4: $\boldsymbol{\theta}^{(t,0)} \leftarrow \boldsymbol{\theta}^{(t-1)}$
5: $\Delta\boldsymbol{\theta}^{(t,0)} \leftarrow \Delta\boldsymbol{\theta}^{(t-1)}$
```

```
6: for $i \leftarrow 1$ to $M$ do
7:     BatchData $\leftarrow \{\mathcal{D}_i, q_i\}$
8:     $\nabla_{\boldsymbol{\theta}^{(t,i)}} C, \nabla^2_{\boldsymbol{\theta}^{(t,i)}} C \leftarrow$ *EstimateDerivatives*.BatchData        ▷ Algorithm 11
9:     Estimate $\boldsymbol{r}^{(t,i)}$
10:    $\Delta\boldsymbol{\theta}^{(t,i)} \leftarrow \eta\boldsymbol{r}^{(t,i)} - \lambda\boldsymbol{\theta}^{(t,i)} + \nu\Delta\boldsymbol{\theta}^{(t,i-1)}$
11:    $\boldsymbol{\theta}^{(t,i+1)} \leftarrow \boldsymbol{\theta}^{(t,i)} + \Delta\boldsymbol{\theta}^{(t,i)}$
12:    Apply constraints
13: end for
14: $\boldsymbol{\theta}^{(t+1)} \leftarrow \boldsymbol{\theta}^{(t,M+1)}$
15: $\Delta\boldsymbol{\theta}^{(t)} \leftarrow \Delta\boldsymbol{\theta}^{(t,M+1)}$
16: return $\boldsymbol{\theta}^{(t+1)}$, $\Delta\boldsymbol{\theta}^{(t)}$
```

APPENDIX C

Bandgap Optimization in Potts Model

Problem Statement: Given a finite undirected simple graph $\mathcal{G}(\mathcal{V}, \mathcal{C})$, find parameters, $\boldsymbol{\theta}$ that maximizes the band gap in following two situations:

Case 1: $\mathcal{S}_D$ is prescribed and $\mathcal{S}_G(\boldsymbol{\theta}_D) = \mathcal{S}_D$.

Case 2: Ground state multiplicity, N_{GS}, is prescribed.

To make this optimization problem well posed, it is additionally imposed that $H_i^{\min} \leq H_i \leq H_i^{\max}$ and $J_k^{\min} \leq J_k \leq J_k^{\max}$. Moreover, the functions $U(s)$ and $V(s_i, s_j)$ are predetermined and not calibrated in the optimization process.

C.1 Methods

A Mixed Integer Linear Programming (MILP) problem is formalized for parameter estimation of Potts model. A brief overview of the MILP formulation is presented below:

Definition C.1 (Mixed Integer Linear Programming (MILP))**.** An optimization problem is considered to be of MILP type when the objective function is linear in the decision variables and some of the decision variables are integer. A typical setup of

MILP problem is given in Eq(C.1) where $\boldsymbol{x}$ is the decision variable of size N, I is the set of indices of $\boldsymbol{x}$ which are integers and the matrices $\boldsymbol{A_{eq}}$, $\boldsymbol{b_{eq}}$, $\boldsymbol{A}$ and $\boldsymbol{B}$ are used to define linear constraints.

$$\begin{aligned} \text{Optimize:} \quad & \min_{\boldsymbol{x}} \boldsymbol{cx} \\ \text{Inequality constraints:} \quad & \boldsymbol{Ax} \leq \boldsymbol{b} \\ \text{Equality constraints:} \quad & \boldsymbol{A_{eq}}\mathbf{x} = \mathbf{b}_{eq} \\ \text{Bounds:} \quad & \mathbf{lb} \leq \mathbf{x} \leq \mathbf{ub} \\ \text{Integer variables:} \quad & \boldsymbol{x_I} \in \mathbb{Z} \end{aligned} \tag{C.1}$$

The MILP formulation for the two cases is presented next. In both cases, the decision variables include the parameters, $\boldsymbol{\theta}$, and some auxiliary variables. These variables are introduced along with the algorithm description. Moreover, the algorithms do not enforce that $\Delta E > 0$. Therefore, the results are accepted only if this condition is met.

C.1.1 Parameter Estimation for Potts model with Data Set (PEPDAS)

The energies of individual states can be evaluated as a matrix product operation which works well with linear programming framework. However, the calculation of band gap requires calculation of a minimum of energy over $\mathcal{S}_E$. This operation introduces a non-linearity. Thus, following auxiliary variables are introduced to pose this optimization as a linear programming problem:

- E_1 (real valued scalar): It represents the energy of the 1st excited state.
- $\boldsymbol{m} = [m_1, ..., m_{N_{ES}}]$ (binary valued vector of size N_E): It is defined such that it's value is 1 on exactly one index and 0 everywhere else. The index with value 1 must correspond to one of the 1st excited state.

- M (real valued scalar): It represents a large positive number. For computational purposes it can be evaluated as:

$$M = \left(\max_{s} |U(s)|\right) \sum_{i=1}^{N_V} \left(|H_i^{\max}| + |H_i^{\min}|\right) + \left(\max_{s_1,s_2} |V(s_1, s_2)|\right) \sum_{k=1}^{N_C} \left(|J_k^{\max}| + |J_k^{\min}|\right) \tag{C.2}$$

The decision variable in this formulation are given as:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{\theta}, & E_1, & \boldsymbol{m} \end{bmatrix}^T$$

Consider a data set, $\mathcal{S}_D = \{\overline{\boldsymbol{S}}_1, ..., \overline{\boldsymbol{S}}_{N_{DS}}\}$. The optimization cost $(-\Delta E)$ is estimated by substituting the $E(\overline{\boldsymbol{S}}_1)$ as that of ground state and E_1 for the 1st excited state energy. Thus the cost is evaluated as:

$$\text{Cost} = E(\overline{\boldsymbol{S}}_1) - E_1$$

The energy of all data states are explicitly equated as follows:

$$E(\boldsymbol{S}_1) - E(\boldsymbol{S}_i) = 0, \qquad \forall i \in \{2, ..., N_{DS}\}$$

The 1^{st} excited energy, E_1 is estimated by bounding it from above by energies of all the excited states. It is bounded from below by the energy of state corresponding to the index at which $m_i = 1$. The upper bound on E_1 insures that if $m_i = 1$, then $E_1(\boldsymbol{\theta}) = E(\boldsymbol{S}_i)$. These conditions can be imposed using following set of equations and inequality:

$$E(\boldsymbol{S}_i) - E_1 + M m_i \leq M, \qquad \forall i \in \{1, ..., N_{ES}\}$$

$$-E(\boldsymbol{S}_i) + E_1 \leq 0, \qquad \forall i \in \{1, ..., N_{ES}\}$$

$$\sum_{i=1}^{N_{ES}} m_i = 1$$

Most computing software only allows integer valued variables. In such a case, the binary value of variable $\boldsymbol{m}$ can be explicitly enforced by setting following bounds on integer valued $\boldsymbol{m}$:

$$0 \leq m_i \leq 1, \qquad \forall i \in \{1, ..., N_{ES}\}$$

This formulation is presented in Box 1 in the matrix format.

Optimization cost:

$$\boldsymbol{c} = \begin{bmatrix} \boldsymbol{\varepsilon}(\overline{\boldsymbol{S}}_1) & -1 & \mathbf{0}_{1\times N_{ES}} \end{bmatrix}$$

Inequality constraints

$$\boldsymbol{A} = \begin{bmatrix} \vdots & \vdots & \vdots \\ \boldsymbol{\varepsilon}(\boldsymbol{S}_i) & -1 & [0, ..., 0, \underbrace{M}_{i^{th}\text{index}}, 0, ..., 0]_{1\times N_{ES}} \\ -\boldsymbol{\varepsilon}(\boldsymbol{S}_i) & 1 & \mathbf{0}_{1\times N_{ES}} \\ \vdots & \vdots & \vdots \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} \vdots \\ M \\ 0 \\ \vdots \end{bmatrix}$$

Equality constraints:

$$\boldsymbol{A_{eq}} = \begin{bmatrix} \mathbf{0}_{1\times N_V} & 0 & \mathbf{1}_{1\times N_{ES}} \\ \boldsymbol{\varepsilon}(\overline{\boldsymbol{S}}_2) - \boldsymbol{\varepsilon}(\overline{\boldsymbol{S}}_1) & 0 & \mathbf{0}_{1\times N_{ES}} \\ \vdots & \vdots & \vdots \\ \boldsymbol{\varepsilon}(\overline{\boldsymbol{S}}_{N_{DS}}) - \boldsymbol{\varepsilon}(\overline{\boldsymbol{S}}_1) & 0 & \mathbf{0}_{1\times N_{ES}} \end{bmatrix}, \boldsymbol{b_{eq}} = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Bounds:

$$\boldsymbol{lb} = \left[H_1^{\min}, ..., H_{N_V}^{\min}, J_1^{\min}, ..., J_{N_C}^{\min}, -M, \mathbf{0}_{1\times N_{ES}}\right]$$
$$\boldsymbol{ub} = \left[H_1^{\max}, ..., H_{N_V}^{\max}, J_1^{\max}, ..., J_{N_C}^{\max}, M, \mathbf{1}_{1\times N_{ES}}\right]$$

Box C.1: MILP formulation for PEPDAS method

C.1.2 Parameter Estimation for Potts model with Ground State Multiplicity (PEPGSM)

In this formulation only the variable N_{GS} is provided by the user in stead of $\mathcal{S}_{Data}$. This condition adds the complexity of locating the ground states and evaluating the ground state energy, $E_0(\boldsymbol{\theta})$. This problem is resolved by including following auxiliary variables:

- E_0 (real valued scalar): It represents the ground state energy.
- $\boldsymbol{l} = [l_1, ..., l_{N_{TS}}]$ (binary valued vector of size N_{TS}): It is defined such that it's value is 1 on exactly N_{GS} indices and 0 everywhere else. The index has value 1 if and only if it corresponds to the ground state.
- E_1 and M as defined in algorithm 1
- $\boldsymbol{m} = [m_1, ..., m_{N_{TS}}]$ (binary valued vector of size N_{TS}): It is same as algorithm 1, except that the index are now enumerated based on the set $\mathcal{S}$

The decision variable in this formulation are given as:

$$\boldsymbol{x} = \begin{bmatrix} \boldsymbol{\theta} & E_0 & E_1 & \boldsymbol{l} & \boldsymbol{m} \end{bmatrix}$$

The optimization cost is given as:

$$\text{Cost} = E_0 - E_1$$

The estimation of E_0 is done using the same idea of bounding E_0 from above and below. The bound is tight only for indices where $l_i = 1$.

$$-E(\boldsymbol{S}_i) + E_0 \leq 0, \qquad \forall i \in \{1, ..., N_{TS}\}$$
$$E(\boldsymbol{S}_i) - E_0 + Ml_i \leq M, \qquad \forall i \in \{1, ..., N_{TS}\}$$

$$\sum_{i=1}^{N_{TS}} l_i = N_{GS}$$

For the estimation of E_1, the upper bound is lifted on indices corresponding to ground states. This allows to estimate minimum over non-optimal states. Moreover, index of 1^{st} excited state cannot coincide with ground state i.e. $l_i = 1$ and $m_i = 1$ cannot occur simultaneously. These conditions are imposed using following inequalities and equations:

$$-E(\boldsymbol{S}_i) + E_1 - Ml_i \leq 0, \qquad \forall i \in \{1, ..., N_{TS}\}$$
$$E(\boldsymbol{S}_i) - E_1 + Mm_i \leq M, \qquad \forall i \in \{1, ..., N_{TS}\}$$
$$l_i + m_i \leq 1, \qquad \forall i \in \{1, ..., N_{TS}\}$$
$$\sum_{i=1}^{N_{TS}} l_i = 1$$

The condition of binary valued variables is imposed on integer variables as follows:

$$0 \leq l_i, m_i \leq 1, \qquad \forall i \in \{1, ..., N_{ES}\}$$

This formulation is presented in Box 2 in the matrix format.

C.2 Computation size

One of the limiting features of these algorithms is that it grows exponentially with the graph size. An exact number of variables and equations is provided in TableC.1. It should be noted that the number of states, $N_{TS} = N_L^{N_V}$ and is the reason for the large size of the decision variable. The system of equations and inequalities in both algorithms have large sparse blocks which provide some computational easing. It should also be noted that the sparsity of graph, G, does not give considerable

Optimization cost:

$$\boldsymbol{c} = \begin{bmatrix} \mathbf{0}_{1\times(N_V+N_C)} & 1 & -1 & \mathbf{0}_{1\times N_{TS}} & \mathbf{0}_{1\times N_{TS}} \end{bmatrix}$$

Inequality constraints

$$\boldsymbol{A} = \begin{bmatrix} \mathbf{0}_{1\times(N_V+N_C)} & 0 & 0 & \mathbf{1}_{1\times N_{TS}} & \mathbf{1}_{1\times N_{TS}} \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ -\boldsymbol{\varepsilon}(\boldsymbol{S}_i) & 1 & 0 & \mathbf{0}_{1\times N_{TS}} & \mathbf{0}_{1\times N_{TS}} \\ \boldsymbol{\varepsilon}(\boldsymbol{S}_i) & -1 & 0 & [...,0,\underbrace{M}_{i^{th}\text{index}},0,...] & \mathbf{0}_{1\times N_{TS}} \\ -\boldsymbol{\varepsilon}(\boldsymbol{S}_i) & 0 & 1 & [...,0,\underbrace{-M}_{i^{th}\text{index}},0,...] & \mathbf{0}_{1\times N_{TS}} \\ \boldsymbol{\varepsilon}(\boldsymbol{S}_i) & 0 & -1 & \mathbf{0}_{1\times N_{TS}} & [...,0,\underbrace{M}_{i^{th}\text{index}},0,...] \\ \vdots & \vdots & \vdots & \vdots & \vdots \end{bmatrix}, \boldsymbol{b} = \begin{bmatrix} 1 \\ \vdots \\ 0 \\ M \\ 0 \\ M \\ \vdots \end{bmatrix}$$

Equality constraints:

$$\boldsymbol{A_{eq}} = \begin{bmatrix} \mathbf{0}_{1\times(N_V+N_C)} & 0 & 0 & \mathbf{1}_{1\times N_{TS}} & \mathbf{0}_{1\times N_{TS}} \\ \mathbf{0}_{1\times(N_V+N_C)} & 0 & 0 & \mathbf{0}_{1\times N_{TS}} & \mathbf{1}_{1\times N_{TS}} \end{bmatrix}, \boldsymbol{b_{eq}} = \begin{bmatrix} N_{GS} \\ 1 \end{bmatrix}$$

Bounds:

$$\boldsymbol{lb} = \begin{bmatrix} H_1^{\min}, ..., H_{N_V}^{\min}, J_1^{\min}, ..., J_{N_C}^{\min}, -M, -M, \mathbf{0}_{1\times N_{TS}}, \mathbf{0}_{1\times N_{TS}} \end{bmatrix}$$
$$\boldsymbol{ub} = \begin{bmatrix} H_1^{\max}, ..., H_{N_V}^{\max}, J_1^{\max}, ..., J_{N_C}^{\max}, M, M, \mathbf{1}_{1\times N_{TS}}, \mathbf{0}_{1\times N_{TS}} \end{bmatrix}$$

Box C.2: MILP formulation for PEPGSM method advantage in the algorithm as the size of the problem is mainly dictated by the number of labels, N_T, and the number of vertices, N_V.

Quantity	PEPDAS	PEPGSM
Total variables	$N_V + N_C + 1 + N_{ES}$	$N_V + N_C + 2 + 2N_{TS}$
Integer (Binary) variables	N_{ES}	$2N_{TS}$
Inequality conditions	$2N_{ES}$	$4N_{TS} + 1$
Equality conditions	N_{DS}	2

Table C.1: Variable size for Algorithms PEPDAS and PEPGSM

APPENDIX D

Boltzmann Machines

D.1 Definition of statistical quantities

For completeness of notation, the definition of each statistical quantity is provided in context of the random variables. Conditional quantities require following conditional probabilities:

$$
\begin{aligned}
p(\boldsymbol{h}|\boldsymbol{v};\boldsymbol{\theta},\beta) &= \frac{p([\boldsymbol{v},\boldsymbol{h}];\boldsymbol{\theta},\beta)}{\sum_{\widetilde{\boldsymbol{h}}} p([\boldsymbol{v},\widetilde{\boldsymbol{h}}];\boldsymbol{\theta},\beta)} \\
p([\boldsymbol{v}^O,\boldsymbol{h}]|\boldsymbol{v}^I;\boldsymbol{\theta},\beta) &= \frac{p([\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}];\boldsymbol{\theta},\beta)}{\sum_{\widetilde{\boldsymbol{v}}^O,\widetilde{\boldsymbol{h}}} p([\boldsymbol{v}^I,\widetilde{\boldsymbol{v}}^O,\widetilde{\boldsymbol{h}}];\boldsymbol{\theta},\beta)}
\end{aligned}
$$

D.1.1 Expectations

$$
\begin{aligned}
\mathbb{E}(E;\boldsymbol{\theta},\beta) &= \sum_{\boldsymbol{S}} E(\boldsymbol{S};\boldsymbol{\theta})p(\boldsymbol{S};\boldsymbol{\theta},\beta) \\
\mathbb{E}\left(\frac{\partial E}{\partial \theta_i};\boldsymbol{\theta},\beta\right) &= \sum_{\boldsymbol{S}} \frac{\partial E}{\partial \theta_i}(\boldsymbol{S};\boldsymbol{\theta})p(\boldsymbol{S};\boldsymbol{\theta},\beta) \\
\mathbb{E}\left(\left.\frac{\partial E}{\partial \theta_i}\right|\boldsymbol{v};\boldsymbol{\theta},\beta\right) &= \sum_{\boldsymbol{h}} \frac{\partial E}{\partial \theta_i}([\boldsymbol{v},\boldsymbol{h}];\boldsymbol{\theta})p(\boldsymbol{h}|\boldsymbol{v};\boldsymbol{\theta},\beta)
\end{aligned}
$$

$$\mathbb{E}\left(\left.\frac{\partial E}{\partial \theta_i}\right| \boldsymbol{v}^I; \boldsymbol{\theta}, \beta\right) = \sum_{\boldsymbol{v}^O, \boldsymbol{h}} \frac{\partial E}{\partial \theta_i}([\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h}]; \boldsymbol{\theta}) p([\boldsymbol{v}^O, \boldsymbol{h}] | \boldsymbol{v}^I; \boldsymbol{\theta}, \beta)$$

D.1.2 Covariances

The dependence on $\boldsymbol{\theta}$ and β is dropped for notational convenience.

$$\begin{aligned}
\operatorname{Cov}\left(\frac{\partial E}{\partial \theta_i}, \frac{\partial E}{\partial \theta_j}\right) &= \sum_{\boldsymbol{S}} \frac{\partial E}{\partial \theta_i} \frac{\partial E}{\partial \theta_j}(\boldsymbol{S}) p(\boldsymbol{S}) - \mathbb{E}\left(\frac{\partial E}{\partial \theta_i}\right) \mathbb{E}\left(\frac{\partial E}{\partial \theta_j}\right) \\
\operatorname{Cov}\left(\frac{\partial E}{\partial \theta_i}, \left.\frac{\partial E}{\partial \theta_j}\right| \boldsymbol{v}\right) &= \sum_{\boldsymbol{h}} \frac{\partial E}{\partial \theta_i} \frac{\partial E}{\partial \theta_j}([\boldsymbol{v}, \boldsymbol{h}]) p(\boldsymbol{h} | \boldsymbol{v}) - \mathbb{E}\left(\left.\frac{\partial E}{\partial \theta_i}\right| \boldsymbol{v}\right) \mathbb{E}\left(\left.\frac{\partial E}{\partial \theta_j}\right| \boldsymbol{v}\right) \\
\operatorname{Cov}\left(\frac{\partial E}{\partial \theta_i}, \left.\frac{\partial E}{\partial \theta_j}\right| \boldsymbol{v}^I\right) &= \sum_{[\boldsymbol{v}^O, \boldsymbol{h}]} \frac{\partial E}{\partial \theta_i} \frac{\partial E}{\partial \theta_j}([\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h}]) p([\boldsymbol{v}^O, \boldsymbol{h}] | \boldsymbol{v}^I) \\
&\quad - \mathbb{E}\left(\left.\frac{\partial E}{\partial \theta_i}\right| \boldsymbol{v}^I\right) \mathbb{E}\left(\left.\frac{\partial E}{\partial \theta_j}\right| \boldsymbol{v}^I\right)
\end{aligned}$$

D.1.3 Variances

$$\begin{aligned}
\operatorname{Var}(E) &= \sum_{\boldsymbol{S}} E^2(\boldsymbol{S}) p(\boldsymbol{S}) - \mathbb{E}^2(E) \\
\operatorname{Var}(E | \boldsymbol{v}) &= \sum_{\boldsymbol{h}} E^2([\boldsymbol{v}, \boldsymbol{h}]) p(\boldsymbol{h} | \boldsymbol{v}) - \mathbb{E}^2(E | \boldsymbol{v}) \\
\operatorname{Var}(E | \boldsymbol{v}^I) &= \sum_{[\boldsymbol{v}^O, \boldsymbol{h}]} E^2([\boldsymbol{v}^I, \boldsymbol{v}^O, \boldsymbol{h}]) p([\boldsymbol{v}^O, \boldsymbol{h}] | \boldsymbol{v}^I) - \mathbb{E}^2(E | \boldsymbol{v}^I)
\end{aligned}$$

D.2 Estimation of derivatives

D.2.1 Gradient of KL Divergence

The gradient of Log-likelihood for a single data is estimated as:

$$\begin{aligned}
\frac{\partial \ln p(\boldsymbol{v})}{\partial \theta_j} &= \frac{\partial}{\partial \theta_j}\left(\ln \sum_{\boldsymbol{h}} e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}\right) - \frac{\partial}{\partial \theta_j}\left(\ln \sum_{\boldsymbol{v}',\boldsymbol{h}'} e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}\right) \\
&= -\beta \sum_{\boldsymbol{h}} \frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}'} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}')}} \frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial \theta_j} + \beta \sum_{\boldsymbol{v}',\boldsymbol{h}'} \frac{e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}}{\sum_{\boldsymbol{v}'',\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v}'',\boldsymbol{h}'')}} \frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial \theta_j} \\
&= -\beta \sum_{\boldsymbol{h}} \frac{\frac{1}{Z} e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\frac{1}{Z}\sum_{\boldsymbol{h}'} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}')}} \frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial \theta_j} + \beta \sum_{\boldsymbol{v}',\boldsymbol{h}'} \frac{e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}}{Z} \frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial \theta_j} \\
&= -\beta \sum_{\boldsymbol{h}} \frac{p(\boldsymbol{v},\boldsymbol{h})}{p(\boldsymbol{v})} \frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial \theta_j} + \beta \sum_{\boldsymbol{v}',\boldsymbol{h}'} p(\boldsymbol{v}',\boldsymbol{h}') \frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial \theta_j} \\
&= -\beta \sum_{\boldsymbol{h}} p(\boldsymbol{h}|\boldsymbol{v}) \frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial \theta_j} + \beta \sum_{\boldsymbol{v}',\boldsymbol{h}'} p(\boldsymbol{v}',\boldsymbol{h}') \frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial \theta_j} \\
&= \beta\left(\mathbb{E}\left(\frac{\partial E}{\partial \theta_j}\right) - \mathbb{E}\left(\left.\frac{\partial E}{\partial \theta_j}\right| \boldsymbol{v}\right)\right)
\end{aligned} \tag{D.1}$$

The gradient of KL Divergence is estimated as:

$$\begin{aligned}
\frac{\partial D_{KL}(q||p)}{\partial \theta_j} &= -\sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v}) \frac{\partial}{\partial \theta_i}\left(\ln \frac{p(\boldsymbol{v})}{q(\boldsymbol{v})}\right) \\
&= \beta\left(-\mathbb{E}\left(\frac{\partial E}{\partial \theta_j}\right) + \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v}) \mathbb{E}\left(\left.\frac{\partial E}{\partial \theta_j}\right| \boldsymbol{v}\right)\right)
\end{aligned} \tag{D.2}$$

D.2.2 Gradient of Negative Conditional Log-likelihood

$$
\begin{aligned}
\frac{\partial \mathcal{N}}{\partial \theta_j} &= -\sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} \left(\frac{\partial \ln \sum_{\boldsymbol{h''}} e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}')}}{\partial \theta_j} - \frac{\partial \ln \sum_{\boldsymbol{v'^O},\boldsymbol{h}'} e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v'^O},\boldsymbol{h'})}}{\partial \theta_j} \right) \\
&= \beta \sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} \left(\sum_{\boldsymbol{h''}} \frac{Zp_\theta(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h''})}{Zp_\theta(\boldsymbol{v}^I,\boldsymbol{v}^O)} \frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}'')}{\partial \theta_j} \right. \\
&\qquad \left. - \sum_{\boldsymbol{v'^O},\boldsymbol{h}'} \frac{Zp_\theta(\boldsymbol{v}^I,\boldsymbol{v'^O},\boldsymbol{h'})}{Zp_\theta(\boldsymbol{v}^I)} \frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v'^O},\boldsymbol{h'})}{\partial \theta_j} \right) \\
&= \beta \sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} \left(\sum_{\boldsymbol{h}''} p(\boldsymbol{h}''|\boldsymbol{v}^I,\boldsymbol{v}^O) \frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}'')}{\partial \theta_j} \right. \\
&\qquad \left. - \sum_{\boldsymbol{v'^O},\boldsymbol{h}'} p(\boldsymbol{v'^O},\boldsymbol{h}'|\boldsymbol{v}^I) \frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v'^O},\boldsymbol{h'})}{\partial \theta_j} \right) \\
&= \beta \sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} \left(\mathbb{E}\left(\left. \frac{\partial E}{\partial \theta_j} \right| \boldsymbol{v^I},\boldsymbol{v^O} \right) - \mathbb{E}\left(\left. \frac{\partial E}{\partial \theta_j} \right| \boldsymbol{v^I} \right) \right)
\end{aligned}
\tag{D.3}
$$

D.2.3 Hessian of KL Divergence

Hessian of Log-likelihood for a single data is estimated first. It uses the fact that in the case of Ising type energy, $\frac{\partial^2 E}{\partial \theta_i \partial \theta_j} = 0$ for all possible i and j.

$$
\begin{aligned}
\frac{\partial^2 \ln p}{\partial\theta_i\partial\theta_j}(\boldsymbol{v}) =& -\beta\sum_{\boldsymbol{h}}\frac{\partial p(\boldsymbol{h}|\boldsymbol{v})}{\partial\theta_i}\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial\theta_j} + \beta\sum_{\boldsymbol{v}',\boldsymbol{h}'}\frac{\partial p(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_i}\frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_j}\\
=& -\beta\sum_{\boldsymbol{h}}\frac{\partial}{\partial\theta_i}\left(\frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}''}e^{-\beta E(\boldsymbol{v},\boldsymbol{h}'')}}\right)\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial\theta_j}\\
&+\beta\sum_{\boldsymbol{v}',\boldsymbol{h}'}\frac{\partial}{\partial\theta_i}\left(\frac{e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}}{\sum_{\boldsymbol{v}''',\boldsymbol{h}'''}e^{-\beta E(\boldsymbol{v}''',\boldsymbol{h}''')}}\right)\frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_j}\\
=&\beta^2\sum_{\boldsymbol{h}}\left(p(\boldsymbol{h}|\boldsymbol{v})\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial\theta_i} - p(\boldsymbol{h}|\boldsymbol{v})\sum_{\boldsymbol{h}''}p(\boldsymbol{h}''|\boldsymbol{v})\frac{\partial E(\boldsymbol{v},\boldsymbol{h}'')}{\partial\theta_i}\right)\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial\theta_j}\\
&-\beta^2\sum_{\boldsymbol{v}',\boldsymbol{h}'}\left(p(\boldsymbol{v}',\boldsymbol{h}')\frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_i} - p(\boldsymbol{v}',\boldsymbol{h}')\sum_{\boldsymbol{v}''',\boldsymbol{h}'''}p(\boldsymbol{v}''',\boldsymbol{h}''')\frac{\partial E(\boldsymbol{v}''',\boldsymbol{h}''')}{\partial\theta_i}\right)\\
&\frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_j}\\
=&\beta^2\left(\sum_{\boldsymbol{h}}p(\boldsymbol{h}|\boldsymbol{v})\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial\theta_i}\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial\theta_j}\right)\\
&-\beta^2\left(\sum_{\boldsymbol{h}}p(\boldsymbol{h}|\boldsymbol{v})\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial\theta_i}\right)\left(\sum_{\boldsymbol{h}}p(\boldsymbol{h}|\boldsymbol{v})\frac{\partial E(\boldsymbol{v},\boldsymbol{h})}{\partial\theta_j}\right)\\
&-\beta^2\left(\sum_{\boldsymbol{v}',\boldsymbol{h}'}p(\boldsymbol{v}',\boldsymbol{h}')\frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_i}\frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_j}\right)\\
&+\beta^2\left(\sum_{\boldsymbol{v}',\boldsymbol{h}'}p(\boldsymbol{v}',\boldsymbol{h}')\frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_i}\right)\left(\sum_{\boldsymbol{v}',\boldsymbol{h}'}p(\boldsymbol{v}',\boldsymbol{h}')\frac{\partial E(\boldsymbol{v}',\boldsymbol{h}')}{\partial\theta_j}\right)\\
=&\beta^2\left(\mathrm{Cov}\left(\frac{\partial E}{\partial\theta_i},\frac{\partial E}{\partial\theta_j}\middle|\,\boldsymbol{v}\right) - \mathrm{Cov}\left(\frac{\partial E}{\partial\theta_i},\frac{\partial E}{\partial\theta_j}\right)\right)
\end{aligned}
\tag{D.4}
$$

The hessian of KL Divergence is estimated as:

$$
\frac{\partial^2 D_{KL}(q||p)}{\partial\theta_i\partial\theta_j} = \beta^2\left(\mathrm{Cov}\left(\frac{\partial E}{\partial\theta_i},\frac{\partial E}{\partial\theta_j}\right) - \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}}q(\boldsymbol{v})\,\mathrm{Cov}\left(\frac{\partial E}{\partial\theta_i},\frac{\partial E}{\partial\theta_j}\middle|\,\boldsymbol{v}\right)\right)
\tag{D.5}
$$

D.2.4 Hessian of Negative Conditional Log-likelihood for a single data

Hessian for a single data is estimated as: $\boldsymbol{v}\equiv[\boldsymbol{v}^I,\boldsymbol{v}^O]$

$$
\begin{aligned}
\frac{\partial^2 \mathcal{N}}{\partial\theta_i\partial\theta_j} =&\beta\left(\sum_{\boldsymbol{h}''}\frac{\partial p(\boldsymbol{h}''|\boldsymbol{v}^I,\boldsymbol{v}^O)}{\partial\theta_i}\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}'')}{\partial\theta_j}-\sum_{\boldsymbol{v}'^O,\boldsymbol{h}'}\frac{\partial p(\boldsymbol{v}'^O,\boldsymbol{h}'|\boldsymbol{v}^I)}{\partial\theta_i}\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}'^O,\boldsymbol{h}')}{\partial\theta_j}\right)\\
=&\beta\left(\sum_{\boldsymbol{h}''}\frac{\partial}{\partial\theta_i}\left(\frac{e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}'')}}{\sum_{\overline{\boldsymbol{h}}}e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^O,\overline{\boldsymbol{h}})}}\right)\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}'')}{\partial\theta_j}\right.\\
&\left.-\sum_{\boldsymbol{v}'^O,\boldsymbol{h}'}\frac{\partial}{\partial\theta_i}\left(\frac{e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}'^O,\boldsymbol{h}')}}{\sum_{\widetilde{\boldsymbol{v}}^O,\widetilde{\boldsymbol{h}}}e^{-\beta E(\boldsymbol{v}^I,\widetilde{\boldsymbol{v}}^O,\widetilde{\boldsymbol{h}})}}\right)\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}'^O,\boldsymbol{h}')}{\partial\theta_j}\right)\\
=&-\beta^2\left(\left(\sum_{\boldsymbol{h}''}p(\boldsymbol{h}''|\boldsymbol{v}^I,\boldsymbol{v}^O)\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}'')}{\partial\theta_i}\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}'')}{\partial\theta_j}\right)\right.\\
&-\left(\sum_{\overline{\boldsymbol{h}}}p(\overline{\boldsymbol{h}}|\boldsymbol{v}^I,\boldsymbol{v}^O)\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}^O,\overline{\boldsymbol{h}})}{\partial\theta_i}\right)\left(\sum_{\boldsymbol{h}''}p(\boldsymbol{h}''|\boldsymbol{v}^I,\boldsymbol{v}^O)\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}^O,\boldsymbol{h}'')}{\partial\theta_j}\right)\\
&-\left(\sum_{\boldsymbol{v}'^O,\boldsymbol{h}'}p(\boldsymbol{v}'^O,\boldsymbol{h}'|\boldsymbol{v}^I)\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}'^O,\boldsymbol{h}')}{\partial\theta_i}\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}'^O,\boldsymbol{h}')}{\partial\theta_j}\right)\\
&\left.+\left(\sum_{\widetilde{\boldsymbol{v}}^O\widetilde{\boldsymbol{h}}}p(\widetilde{\boldsymbol{v}}^O,\widetilde{\boldsymbol{h}}|\boldsymbol{v}^I)\frac{\partial E(\boldsymbol{v}^I,\widetilde{\boldsymbol{v}}^O,\widetilde{\boldsymbol{h}})}{\partial\theta_i}\right)\left(\sum_{\boldsymbol{v}'^O,\boldsymbol{h}'}p(\boldsymbol{v}'^O,\boldsymbol{h}'|\boldsymbol{v}^I)\frac{\partial E(\boldsymbol{v}^I,\boldsymbol{v}'^O,\boldsymbol{h}')}{\partial\theta_j}\right)\right)\\
=&\beta^2\left(\mathrm{Cov}\left(\frac{\partial E}{\partial\theta_i},\frac{\partial E}{\partial\theta_j}\middle|\boldsymbol{v}^I\right)-\mathrm{Cov}\left(\frac{\partial E}{\partial\theta_i},\frac{\partial E}{\partial\theta_j}\middle|\boldsymbol{v}\right)\right)
\end{aligned}
\tag{D.6}
$$

Now considering the visible data, $[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,...,\boldsymbol{v}^D\}$,

$$
\frac{\partial^2 \mathcal{N}}{\partial\theta_i\partial\theta_j} =\beta^2\sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,...,\boldsymbol{v}^D\}}\left(\mathrm{Cov}\left(\frac{\partial E}{\partial\theta_i},\frac{\partial E}{\partial\theta_j}\middle|\boldsymbol{v}^I\right)-\mathrm{Cov}\left(\frac{\partial E}{\partial\theta_i},\frac{\partial E}{\partial\theta_j}\middle|\boldsymbol{v}^I,\boldsymbol{v}^O\right)\right)
\tag{D.7}
$$

D.2.5 Derivative of KL Divergence w.r.t. Inverse temperature

$$
\begin{aligned}
\frac{dD_{KL}}{d\beta} =& -\sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,...,\boldsymbol{v}^D\}}q(\boldsymbol{v})\frac{d}{d\beta}\ln\frac{p(\boldsymbol{v})}{q(\boldsymbol{v})} = -\sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,...,\boldsymbol{v}^D\}}\frac{q(\boldsymbol{v})}{p(\boldsymbol{v})}\frac{d}{d\beta}\frac{\sum_{\boldsymbol{h}}e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{v}',\boldsymbol{h}'}e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}}\\
=& -\sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,...,\boldsymbol{v}^D\}}\frac{q(\boldsymbol{v})}{p(\boldsymbol{v})}\left(\frac{-\sum_{\boldsymbol{h}}E(\boldsymbol{v},\boldsymbol{h})e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{v}',\boldsymbol{h}'}e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}}\right.
\end{aligned}
$$

$$+\frac{\left(\sum_{\boldsymbol{h}} e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}\right)\left(\sum_{\boldsymbol{v}',\boldsymbol{h}'} E(\boldsymbol{v}',\boldsymbol{h}')e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}\right)}{\left(\sum_{\boldsymbol{v}'',\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v}'',\boldsymbol{h}'')}\right)^2}\Bigg)$$

$$= \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} \frac{q(\boldsymbol{v})}{p(\boldsymbol{v})}\left(\sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h})p(\boldsymbol{v},\boldsymbol{h}) - p(\boldsymbol{v})\sum_{\boldsymbol{v}',\boldsymbol{h}'} E(\boldsymbol{v}',\boldsymbol{h}')p(\boldsymbol{v}',\boldsymbol{h}')\right)$$

$$= \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v})\left(\sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h})p(\boldsymbol{h}|\boldsymbol{v}) - \sum_{\boldsymbol{v}',\boldsymbol{h}'} E(\boldsymbol{v}',\boldsymbol{h}')p(\boldsymbol{v}',\boldsymbol{h}')\right)$$

$$= -\mathbb{E}_{\boldsymbol{v},\boldsymbol{h}}(E) + \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v})\sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h})p(\boldsymbol{h}|\boldsymbol{v})$$

$$\frac{d^2 D_{KL}}{d\beta^2} = \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v})\left(\sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h})\frac{d}{d\beta}p(\boldsymbol{h}|\boldsymbol{v}) - \sum_{\boldsymbol{v}',\boldsymbol{h}'} E(\boldsymbol{v}',\boldsymbol{h}')\frac{d}{d\beta}p(\boldsymbol{v}',\boldsymbol{h}')\right)$$

$$= \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v})\Bigg(\sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h})\,\underbrace{\frac{d}{d\beta}\frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}'')}}}_{\text{Term } I}$$

$$- \sum_{\boldsymbol{v}',\boldsymbol{h}'} E(\boldsymbol{v}',\boldsymbol{h}')\,\underbrace{\frac{d}{d\beta}\frac{e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}}{\sum_{\boldsymbol{v}'',\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v}'',\boldsymbol{h}'')}}}_{\text{Term } II}\Bigg)$$

Term I is evaluated as:

$$\frac{d}{d\beta}\frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}'')}} = -\frac{E(\boldsymbol{v},\boldsymbol{h})e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}'')}} + \frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}\sum_{\boldsymbol{h}'} E(\boldsymbol{v},\boldsymbol{h}')e^{-\beta E(\boldsymbol{v},\boldsymbol{h}')}}{\left(\sum_{\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}'')}\right)^2}$$

$$= -E(\boldsymbol{v},\boldsymbol{h})p(\boldsymbol{h}|\boldsymbol{v}) + p(\boldsymbol{h}|\boldsymbol{v})\sum_{\boldsymbol{h}'} E(\boldsymbol{v},\boldsymbol{h}')p(\boldsymbol{h}'|\boldsymbol{v})$$

Term II is evaluated as:

$$\frac{d}{d\beta}\frac{e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}}{\sum_{\boldsymbol{v}'',\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v}'',\boldsymbol{h}'')}} = -\frac{E(\boldsymbol{v}',\boldsymbol{h}')e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}}{Z} + \frac{e^{-\beta E(\boldsymbol{v}',\boldsymbol{h}')}\sum_{\boldsymbol{v}'',\boldsymbol{h}''} E(\boldsymbol{v}'',\boldsymbol{h}'')e^{-\beta E(\boldsymbol{v}'',\boldsymbol{h}'')}}{Z^2}$$

$$= -E(\boldsymbol{v}',\boldsymbol{h}')p(\boldsymbol{v}',\boldsymbol{h}') + p(\boldsymbol{v}',\boldsymbol{h}')\sum_{\boldsymbol{v}'',\boldsymbol{h}''} E(\boldsymbol{v}'',\boldsymbol{h}'')p(\boldsymbol{v}'',\boldsymbol{h}'')$$

Combining the two terms:

$$\begin{aligned}
\frac{d^2 D_{KL}}{d\beta^2} &= \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v}) \left(\sum_{\boldsymbol{h}} -E^2(\boldsymbol{v},\boldsymbol{h}) p(\boldsymbol{h}|\boldsymbol{v}) + \left(\sum_{\boldsymbol{h}'} E(\boldsymbol{v},\boldsymbol{h}') p(\boldsymbol{h'}|\boldsymbol{v}) \right)^2 \right. \\
&\qquad \left. + \sum_{\boldsymbol{v}',\boldsymbol{h}'} E^2(\boldsymbol{v}',\boldsymbol{h}') p(\boldsymbol{v}',\boldsymbol{h}') - \left(\sum_{\boldsymbol{v}'',\boldsymbol{h}''} E(\boldsymbol{v''},\boldsymbol{h}'') p(\boldsymbol{v}'',\boldsymbol{h}'') \right)^2 \right) \\
&= \sum_{\boldsymbol{v}\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} q(\boldsymbol{v}) \left(-\operatorname{Var}(E|\boldsymbol{v}) + \operatorname{Var}(E) \right)
\end{aligned}$$

D.2.6 Derivative of Negative Like-Likelihood w.r.t. Inverse temperature

The derivative of log-likelihood of conditional probability for a single data, $\boldsymbol{v} \equiv [\boldsymbol{v}^I, \boldsymbol{v}^O]$ is calculated first:

$$\begin{aligned}
\frac{d \ln p(\boldsymbol{v}|\boldsymbol{v}^I)}{d\beta} &= \frac{d}{d\beta}\left(\ln \frac{p(\boldsymbol{v})}{p(\boldsymbol{v}^I)} \right) = \frac{d}{d\beta} \left(\ln \sum_{\boldsymbol{h}} e^{-\beta E(\boldsymbol{v},\boldsymbol{h})} - \ln \sum_{\overline{\boldsymbol{v}}^O,\boldsymbol{h}} e^{-\beta E(\boldsymbol{v}^I,\overline{\boldsymbol{v}}^O,\boldsymbol{h})} \right) \\
&= \sum_{\boldsymbol{h}} -E(\boldsymbol{v},\boldsymbol{h}) \frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}'} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}')}} + \sum_{\boldsymbol{v}^{O'},\boldsymbol{h}'} E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}') \frac{e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')}}{\sum_{\boldsymbol{v}^{O''},\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O''},\boldsymbol{h}'')}} \\
&= \sum_{\boldsymbol{h}} -E(\boldsymbol{v},\boldsymbol{h}) p(\boldsymbol{v},\boldsymbol{h}|\boldsymbol{v}) + \sum_{\boldsymbol{v}^{O'},\boldsymbol{h}'} E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}') p(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}|\boldsymbol{v}^I) \\
&= -\mathbb{E}(E|\boldsymbol{v}) + \mathbb{E}(E|\boldsymbol{v}^I)
\end{aligned}$$

The second derivative is estimated as

$$\begin{aligned}
\frac{d^2 \ln p(\boldsymbol{v}|\boldsymbol{v}^I)}{d\beta^2} &= \sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h}) \underbrace{\frac{d}{d\beta} \frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}'} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}')}}}_{\text{Term } I} \\
&\quad - \sum_{\boldsymbol{v}^{O'},\boldsymbol{h}'} E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}') \underbrace{\frac{d}{d\beta} \frac{e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')}}{\sum_{\boldsymbol{v}^{O''},\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O''},\boldsymbol{h}'')}}}_{\text{Term } II}
\end{aligned}$$

Term I is evaluated as:

$$\begin{aligned}
\frac{d}{d\beta}\frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}'} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}')}} &= -\frac{E(\boldsymbol{v},\boldsymbol{h})e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}}{\sum_{\boldsymbol{h}'} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}')}} + \frac{e^{-\beta E(\boldsymbol{v},\boldsymbol{h})}\sum_{\boldsymbol{h}'} E(\boldsymbol{v},\boldsymbol{h}')e^{-\beta E(\boldsymbol{v},\boldsymbol{h}')}}{\left(\sum_{\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v},\boldsymbol{h}'')}\right)^2} \\
&= -E(\boldsymbol{v},\boldsymbol{h})p(\boldsymbol{h}|\boldsymbol{v}) + p(\boldsymbol{h}|\boldsymbol{v})\sum_{\boldsymbol{h}'} E(\boldsymbol{v},\boldsymbol{h}')p(\boldsymbol{h}'|\boldsymbol{v}) \\
&= -E(\boldsymbol{v},\boldsymbol{h})p(\boldsymbol{h}|\boldsymbol{v}) + p(\boldsymbol{h}|\boldsymbol{v})\mathbb{E}(E|\boldsymbol{v})
\end{aligned}$$

Term II is evaluated as:

$$\begin{aligned}
\frac{d}{d\beta}\frac{e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')}}{\sum_{\boldsymbol{v}^{O''},\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O''},\boldsymbol{h}'')}} &= -E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')\frac{e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')}}{\sum_{\boldsymbol{v}^{O''},\boldsymbol{h}''} e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O''},\boldsymbol{h}'')}} + \\
&\quad \frac{e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')}\sum_{\boldsymbol{v}^{O''},\boldsymbol{h}''} E(\boldsymbol{v}^I,\boldsymbol{v}^{O''},\boldsymbol{h}'')e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O''},\boldsymbol{h}'')}}{\left(\sum_{\boldsymbol{v}^{O'''},\boldsymbol{h}'''} e^{-\beta E(\boldsymbol{v}^I,\boldsymbol{v}^{O'''},\boldsymbol{h}''')}\right)^2} \\
&= -E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')p(\boldsymbol{v}^{O'},\boldsymbol{h}'|\boldsymbol{v}^I) + p(\boldsymbol{v}^{O'},\boldsymbol{h}'|\boldsymbol{v}^I)\mathbb{E}(E|\boldsymbol{v}^I)
\end{aligned}$$

Combining the two terms:

$$\begin{aligned}
\frac{d^2 \ln p(\boldsymbol{v}|\boldsymbol{v}^I)}{d\beta^2} &= \sum_{\boldsymbol{h}} -E^2(\boldsymbol{v},\boldsymbol{h})p(\boldsymbol{h}|\boldsymbol{v}) + \mathbb{E}(E|\boldsymbol{v})\sum_{\boldsymbol{h}} E(\boldsymbol{v},\boldsymbol{h})p(\boldsymbol{h}|\boldsymbol{v}) \\
&\quad + \sum_{\boldsymbol{v}^{O'},\boldsymbol{h}'} E^2(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')p(\boldsymbol{v}^{O'},\boldsymbol{h}'|\boldsymbol{v}^I) \\
&\quad - \mathbb{E}(E|\boldsymbol{v}^I)\sum_{\boldsymbol{v}^{O'},\boldsymbol{h}'} E(\boldsymbol{v}^I,\boldsymbol{v}^{O'},\boldsymbol{h}')p(\boldsymbol{v}^{O'},\boldsymbol{h}'|\boldsymbol{v}^I) \\
&= -\mathbb{E}(E^2|\boldsymbol{v}) + \mathbb{E}^2(E|\boldsymbol{v}) + \mathbb{E}(E^2|\boldsymbol{v}^I) - \mathbb{E}^2(E|\boldsymbol{v}^I) \\
&= -\operatorname{Var}(E|\boldsymbol{v}) + \operatorname{Var}(E|\boldsymbol{v}^I)
\end{aligned}$$

The first derivative of Negative conditional log-likelihood is estimated as:

$$\frac{d\mathcal{N}}{d\beta} = \sum_{[\boldsymbol{v}^I,\boldsymbol{v}^O]\in\{\boldsymbol{v}^1,\ldots,\boldsymbol{v}^D\}} \mathbb{E}(E|\boldsymbol{v}) - \mathbb{E}(E|\boldsymbol{v}^I)$$

The second derivative of Negative conditional log-likelihood is estimated as:

$$\frac{d^2\mathcal{N}}{d\beta^2} = \sum_{[\boldsymbol{v}^I, \boldsymbol{v}^O] \in \{\boldsymbol{v}^1, \ldots, \boldsymbol{v}^D\}} - \operatorname{Var}(E|\boldsymbol{v}) + \operatorname{Var}(E|\boldsymbol{v}^I)$$

D.3 Transformation of Ising basis

It is a common practice to define the Ising states as either $\{0, 1\}$ or $\{-1, +1\}$ states. The latter format is employed on the DWave machine. Here, the details about conversion between these formats are presented. For the purpose of discussion, the $\{0, 1\}$ Ising model is represented with variables, $\{\boldsymbol{S}, \{H_i\}_{i=1}^{N_V}, \{J_i\}_{i=1}^{N_C}\}$ and the $\{-1, 1\}$ Ising with model is represented with overlined variables, $\{\overline{\boldsymbol{S}}, \{\overline{H}_i\}_{i=1}^{N_V}, \{\overline{J}_i\}_{i=1}^{N_C}\}$. The three variables represent the state, field energy and interaction energy respectively. The states of the system can be interchanged using the following equation:

$$\overline{\boldsymbol{S}} = 2\boldsymbol{S} - 1 \tag{D.8}$$

The interaction parameters can be interchanged as:

$$\overline{J}_k = \frac{1}{4} J_k \tag{D.9}$$

And the field parameter as:

$$\overline{H}_i = \frac{1}{2} H_i + \frac{1}{4} \sum_{\pi(k,1)=i \text{ OR } \pi(k,2)=i} J_k \tag{D.10}$$

This transformation shifts the energy of each state with a constant value and hence leaves the Boltzmann probability unchanged as required.

BIBLIOGRAPHY

BIBLIOGRAPHY

[1] Shai Bagon. Discrete energy minimization, beyond submodularity: Applications and approximations. *arXiv preprint arXiv:1210.7362*, 2012.

[2] Jacob Adams. *Investigating microstructural effects on short crack growth and fatigue life behavior of WE43 Magnesium*. PhD thesis, University of Michigan, Ann Arbor, 2018.

[3] Ruslan Salakhutdinov and Geoffrey Hinton. Deep boltzmann machines. In *Artificial intelligence and statistics*, pages 448–455, 2009.

[4] Srihari Sundar and Veera Sundararaghavan. Database development and exploration of process–microstructure relationships using variational autoencoders. *Materials Today Communications*, 25:101201, 2020.

[5] François Graner and James A Glazier. Simulation of biological cell sorting using a two-dimensional extended potts model. *Physical review letters*, 69(13):2013, 1992.

[6] Mark Miodownik. Monte carlo potts model. In Koenraad G.F. Janssens, Dierk Raabe, Ernst Kozeschnik, Mark A. Miodownik, and Britta Nestler, editors, *Computational Materials Engineering*, pages 47 – 108. Academic Press, Burlington, 2007.

[7] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Transactions on pattern analysis and machine intelligence*, 23(11):1222–1239, 2001.

[8] Hayato Ushijima-Mwesigwa, Christian FA Negre, and Susan M Mniszewski. Graph partitioning using quantum annealing on the d-wave system. In *Proceedings of the Second International Workshop on Post Moores Era Supercomputing*, pages 22–29, 2017.

[9] Vladimir Kolmogorov and Ramin Zabin. What energy functions can be minimized via graph cuts? *IEEE transactions on pattern analysis and machine intelligence*, 26(2):147–159, 2004.

[10] Elias Dahlhaus, David S. Johnson, Christos H. Papadimitriou, Paul D. Seymour, and Mihalis Yannakakis. The complexity of multiterminal cuts. *SIAM Journal on Computing*, 23(4):864–894, 1994.

[11] Dmitrij Schlesinger and Boris Flach. *Transforming an arbitrary minsum problem into a binary one.* TU, Fak. Informatik, 2006.

[12] Hiroshi Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE transactions on pattern analysis and machine intelligence*, 25(10):1333–1336, 2003.

[13] Vijay V. Vazirani. *Multiway Cut and k-Cut*, pages 38–46. Springer Berlin Heidelberg, Berlin, Heidelberg, 2003.

[14] Julian Besag. On the statistical analysis of dirty pictures. *Journal of the Royal Statistical Society: Series B (Methodological)*, 48(3):259–279, 1986.

[15] Scott Kirkpatrick, C Daniel Gelatt, and Mario P Vecchi. Optimization by simulated annealing. *science*, 220(4598):671–680, 1983.

[16] Daphne Koller and Nir Friedman. *Probabilistic graphical models: principles and techniques.* MIT press, 2009.

[17] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE transactions on pattern analysis and machine intelligence*, 28(10):1568–1583, 2006.

[18] Victor Lempitsky Carsten Rother Stefan Roth and Andrew Blake. Fusion moves for markov random field optimization. *IEEE Transaction on Pattern Analysis Machine Intelligence*, 32(8):13921405, 2010.

[19] Mengtian Li, Alexander Shekhovtsov, and Daniel Huber. Complexity of discrete energy minimization problems. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 834–852, Cham, 2016. Springer International Publishing.

[20] Xavier Descombes, Robin D Morris, Josiane Zerubia, and Marc Berthod. Estimation of markov random field prior parameters using markov chain monte carlo maximum likelihood. *IEEE Transactions on Image Processing*, 8(7):954–963, 1999.

[21] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14(8):1771–1800, 2002.

[22] Asja Fischer and Christian Igel. An introduction to restricted boltzmann machines. In *Iberoamerican Congress on Pattern Recognition*, pages 14–36. Springer, 2012.

[23] Steven H Adachi and Maxwell P Henderson. Application of quantum annealing to training of deep neural networks. *arXiv preprint arXiv:1510.06356*, 2015.

[24] Richard P Feynman. Simulating physics with computers. *International journal of theoretical physics*, 21(6-7):467–488, 1982.

[25] Edward Farhi, Jeffrey Goldstone, Sam Gutmann, and Michael Sipser. Quantum computation by adiabatic evolution. *arXiv preprint quant-ph/0001106*, 2000.

[26] Satoshi Morita and Hidetoshi Nishimori. Mathematical foundation of quantum annealing. *Journal of Mathematical Physics*, 49(12):125210, 2008.

[27] Marcello Benedetti, John Realpe-Gómez, Rupak Biswas, and Alejandro Perdomo-Ortiz. Estimation of effective temperatures in quantum annealers for sampling applications: A case study with possible applications in deep learning. *Physical Review A*, 94(2):022308, 2016.

[28] Jeremy Liu, Ankith Mohan, Rajiv K Kalia, Aiichiro Nakano, Ken-ichi Nomura, Priya Vashishta, and Ke-Thia Yao. Boltzmann machine modeling of layered mos2 synthesis on a quantum annealer. *Computational Materials Science*, 173:109429, 2020.

[29] Michael Sangid, John F Matlik, Akin Keskin, Ben H Thacker, Barron J Bichon, Dale L Ball, Stephen P Engelstad, Charles Ward, Vasisht Venkatesh, H A Kim, et al. Integrating icme practices into design systems and structural analysis. In *55th AIAA Aerospace Sciences Meeting*, page 0874, 2017.

[30] Veera Sundararaghavan and Nicholas Zabaras. A statistical learning approach for the design of polycrystalline materials. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 1(5):306–321, 2009.

[31] Pinar Acar and Veera Sundararaghavan. Utilization of a linear solver for multiscale design and optimization of microstructures. *AIAA Journal*, pages 1751–1759, 2016.

[32] Shang Sun and Veera Sundararaghavan. A probabilistic crystal plasticity model for modeling grain shape effects based on slip geometry. *Acta Materialia*, 60(13-14):5233–5244, 2012.

[33] S. Sun and V. Sundararaghavan. A peridynamic implementation of crystal plasticity. *International Journal of Solids and Structures*, 51(19):3350 – 3360, 2014.

[34] Veera Sundararaghavan and Siddhartha Srivastava. Microfract: An image based code for microstructural crack path prediction. *SoftwareX*, 6:94–97, 2017.

[35] Adam J Schwartz, Mukul Kumar, Brent L Adams, and David P Field. *Electron backscatter diffraction in materials science.* Springer, 2014.

[36] Michael A Groeber and Michael A Jackson. Dream. 3d: a digital representation environment for the analysis of microstructure in 3d. *Integrating Materials and Manufacturing Innovation*, 3(1):5, 2014.

[37] Guodong Fang, Bassam El Said, Dmitry Ivanov, and Stephen R. Hallett. Smoothing artificial stress concentrations in voxel-based models of textile composites. *Composites Part A: Applied Science and Manufacturing*, 80:270 – 284, 2016.

[38] Romain Quey, PR Dawson, and Fabrice Barbe. Large-scale 3d random polycrystals for the finite element method: Generation, meshing and remeshing. *Computer Methods in Applied Mechanics and Engineering*, 200(17-20):1729–1745, 2011.

[39] Brian W Kernighan and Shen Lin. An efficient heuristic procedure for partitioning graphs. *The Bell system technical journal*, 49(2):291–307, 1970.

[40] Charles M Fiduccia and Robert M Mattheyses. A linear-time heuristic for improving network partitions. In *Papers on Twenty-five years of electronic design automation*, pages 241–247. ACM, 1988.

[41] Peter Sanders and Christian Schulz. High quality graph partitioning. *Graph Partitioning and Graph Clustering*, 588(1), 2012.

[42] Mark EJ Newman. Spectral methods for community detection and graph partitioning. *Physical Review E*, 88(4):042822, 2013.

[43] Aydın Buluç, Henning Meyerhenke, Ilya Safro, Peter Sanders, and Christian Schulz. Recent advances in graph partitioning. In *Algorithm Engineering*, pages 117–158. Springer, 2016.

[44] George Karypis and Vipin Kumar. A fast and high quality multilevel scheme for partitioning irregular graphs. *SIAM Journal on scientific Computing*, 20(1):359–392, 1998.

[45] Peter Sanders and Christian Schulz. Think locally, act globally: Highly balanced graph partitioning. In *International Symposium on Experimental Algorithms*, pages 164–175. Springer, 2013.

[46] Lawrence E Murr, Sara M Gaytan, Diana A Ramirez, Edwin Martinez, Jennifer Hernandez, Krista N Amato, Patrick W Shindo, Francisco R Medina, and Ryan B Wicker. Metal fabrication by additive manufacturing using laser and electron beam melting technologies. *Journal of Materials Science & Technology*, 28(1):1–14, 2012.

[47] Dirk Herzog, Vanessa Seyda, Eric Wycisk, and Claus Emmelmann. Additive manufacturing of metals. *Acta Materialia*, 117:371–392, 2016.

[48] Bernd Baufeld, Omer Van der Biest, and Rosemary Gault. Additive manufacturing of ti–6al–4v components by shaped metal deposition: microstructure and mechanical properties. *Materials & Design*, 31:S106–S111, 2010.

[49] Roland Lvovĭc Dobrushin, Roman Koteckỳ, and Senya Shlosman. *Wulff construction: a global shape from local interaction*, volume 104. American Mathematical Society Providence, 1992.

[50] Frank R. Schmidt, Eno Töppe, and Daniel Cremers. Efficient planar graph cuts with applications in computer vision. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, Miami, Florida, Jun 2009.

[51] A Kelly, WR Tyson, and AH Cottrell. Ductile and brittle crystals. *The Philosophical Magazine: A Journal of Theoretical Experimental and Applied Physics*, 15(135):567–586, 1967.

[52] Michael Ortiz and Anna Pandolfi. Finite-deformation irreversible cohesive elements for three-dimensional crack-propagation analysis. *International journal for numerical methods in engineering*, 44(9):1267–1282, 1999.

[53] Igor Simonovski and Leon Cizelj. Cohesive zone modeling of intergranular cracking in polycrystalline aggregates. *Nuclear Engineering and Design*, 283:139–147, 2015.

[54] M Samimi, JAW Van Dommelen, and MGD Geers. An enriched cohesive zone model for delamination in brittle interfaces. *International journal for numerical methods in engineering*, 80(5):609–630, 2009.

[55] Yan Li, DL McDowell, and Min Zhou. A multiscale framework for predicting fracture toughness of polycrystalline metals. *Materials Performance and Characterization*, 3(3):157–172, 2014.

[56] YF Gao and AF Bower. A simple technique for avoiding convergence problems in finite element simulations of crack nucleation and growth on cohesive interfaces. *Modelling and Simulation in Materials Science and Engineering*, 12(3):453, 2004.

[57] Charlotte Kuhn and Ralf Müller. A continuum phase field model for fracture. *Engineering Fracture Mechanics*, 77(18):3625–3634, 2010.

[58] Vincent Hakim and Alain Karma. Laws of crack motion and phase-field models of fracture. *Journal of the Mechanics and Physics of Solids*, 57(2):342–368, 2009.

[59] Christian Miehe, Fabian Welschinger, and Martina Hofacker. Thermodynamically consistent phase-field models of fracture: Variational principles and multi-field fe implementations. *International Journal for Numerical Methods in Engineering*, 83(10):1273–1311, 2010.

[60] S Murakami and Y Liu. Mesh-dependence in local approach to creep fracture. *International Journal of Damage Mechanics*, 4(3):230–250, 1995.

[61] Arezoo Emdadi and Mohsen Asle Zaeem. Phase-field modeling of crack propagation in polycrystalline materials. *Computational Materials Science*, 186:110057, 2021.

[62] Nicolas Moës, John Dolbow, and Ted Belytschko. A finite element method for crack growth without remeshing. *International journal for numerical methods in engineering*, 46(1):131–150, 1999.

[63] Javier Oliver, Alfredo Edmundo Huespe, and Pablo Javier Sánchez. A comparative study on finite elements for capturing strong discontinuities: E-fem vs x-fem. *Computer methods in applied mechanics and engineering*, 195(37-40):4732–4752, 2006.

[64] Shardul Panwar, Shang Sun, and Veera Sundararaghavan. Modeling fatigue failure using the variational multiscale method. *Engineering Fracture Mechanics*, 162:290–308, 2016.

[65] Shang Sun and Veera Sundararaghavan. Modeling crack propagation in polycrystalline microstructure using variational multiscale method. *Mathematical Problems in Engineering*, 2016(4715696):1 – 14, 2016.

[66] Shang Sun, Ali Ramazani, and Veera Sundararaghavan. A hybrid multi-scale model of crystal plasticity for handling stress concentrations. *Metals*, 7(9):345, 2017.

[67] Gilles A Francfort and J-J Marigo. Revisiting brittle fracture as an energy minimization problem. *Journal of the Mechanics and Physics of Solids*, 46(8):1319–1342, 1998.

[68] Gowri Srinivasan, Jeffrey D Hyman, David A Osthus, Bryan A Moore, Daniel O'Malley, Satish Karra, Esteban Rougier, Aric A Hagberg, Abigail Hunter, and Hari S Viswanathan. Quantifying topological uncertainty in fractured systems using graph theory and machine learning. *Scientific reports*, 8(1):1–11, 2018.

[69] Abigail Hunter, Bryan A Moore, Maruti Mudunuru, Viet Chau, Roselyne Tchoua, Chandramouli Nyshadham, Satish Karra, Daniel O'Malley, Esteban Rougier, Hari Viswanathan, et al. Reduced-order modeling through machine learning and graph-theoretic approaches for brittle fracture applications. *Computational Materials Science*, 157:87–98, 2019.

[70] Frederic E. Bock, Roland C. Aydin, Christian J. Cyron, Norbert Huber, Surya R. Kalidindi, and Benjamin Klusemann. A review of the application of machine learning and data mining approaches in continuum materials mechanics. *Frontiers in Materials*, 6:110, 2019.

[71] Shmuel Osovski, Alan Needleman, and Ankit Srivastava. Intergranular fracture prediction and microstructure design. *International Journal of Fracture*, 216(2):135–148, 2019.

[72] Abhilash Molkeri, Ankit Srivastava, Shmuel Osovski, and Alan Needleman. Influence of Grain Size Distribution on Ductile Intergranular Crack Growth Resistance. *Journal of Applied Mechanics*, 87(3), 11 2019. 031008.

[73] Y. Boykov and V. Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 26(9):1124–1137, 2004.

[74] George C Sih. Strain-energy-density factor applied to mixed mode crack problems. *International Journal of fracture*, 10(3):305–321, 1974.

[75] George C Sih. *Mechanics of fracture initiation and propagation: surface and volume energy density applied as failure criterion*, volume 11. Springer Science & Business Media, 2012.

[76] RJ Nuismer. An energy release rate criterion for mixed mode fracture. *International journal of fracture*, 11(2):245–250, 1975.

[77] Karuppagounder Palaniswamy. *Crack propagation under general in-plane loading*. PhD thesis, California Institute of Technology, 1972.

[78] F. Erdogan and G. C. Sih. On the Crack Extension in Plates Under Plane Loading and Transverse Shear. *Journal of Basic Engineering*, 85(4):519–525, 12 1963.

[79] Abbas Azhdari and Sia Nemat-Nasser. Energy-release rate and crack kinking in anisotropic brittle solids. *Journal of the Mechanics and Physics of Solids*, 44(6):929–951, 1996.

[80] S Teichtmeister, D Kienle, F Aldakheel, and M-A Keip. Phase field modeling of fracture in anisotropic brittle solids. *International Journal of Non-Linear Mechanics*, 97:1–21, 2017.

[81] Siddhartha Srivastava and Veera Sundararaghavan. Graph coloring approach to mesh generation in multiphase media with smooth boundaries. *AIAA Journal*, 58(1):198–205, 2020.

[82] F. R. Schmidt, E. Toppe, and D. Cremers. Efficient planar graph cuts with applications in computer vision. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 351–356, 2009.

[83] Atsushi Takei, Benoît Roman, José Bico, Eugenio Hamm, and Francisco Melo. Forbidden directions for the fracture of thin anisotropic sheets: an analogy with the wulff plot. *Physical review letters*, 110(14):144301, 2013.

[84] L. B. Freund. *Energy concepts in dynamic fracture*, page 221–295. Cambridge Monographs on Mechanics. Cambridge University Press, 1990.

[85] Egon Orowan. Fracture and strength of solids. *Reports on progress in physics*, 12(1):185, 1949.

[86] Hong-Kai Zhao, Tony Chan, Barry Merriman, and Stanley Osher. A variational level set approach to multiphase motion. *Journal of computational physics*, 127(1):179–195, 1996.

[87] Stanley Osher and James A Sethian. Fronts propagating with curvature-dependent speed: algorithms based on hamilton-jacobi formulations. *Journal of computational physics*, 79(1):12–49, 1988.

[88] Virginia Estellers, Dominique Zosso, Rongjie Lai, Stanley Osher, Jean-Philippe Thiran, and Xavier Bresson. Efficient algorithm for level set method preserving distance function. *IEEE Transactions on Image Processing*, 21(12):4722–4734, 2012.

[89] Yuri Boykov, Vladimir Kolmogorov, Daniel Cremers, and Andrew Delong. An integral solution to surface evolution pdes via geo-cuts. In *European Conference on Computer Vision*, pages 409–422. Springer, 2006.

[90] Yuri Boykov and Vladimir Kolmogorov. Computing geodesics and minimal surfaces via graph cuts. In *null*, page 26. IEEE, 2003.

[91] C. L. Epstein and Michael Gage. *The Curve Shortening Flow*, pages 15–59. Springer US, New York, NY, 1987.

[92] Ron Kimmel. *Numerical geometry of images: Theory, algorithms, and applications*. Springer Science & Business Media, 2012.

[93] Luis A. Santaló and Mark Kac. *Crofton's Formulas and the Kinematic Fundamental Formula in Noneuclidean Spaces*, page 316–329. Cambridge Mathematical Library. Cambridge University Press, 2 edition, 2004.

[94] Manfredo P Do Carmo. *Differential geometry of curves and surfaces: revised and updated second edition*. Courier Dover Publications, 2016.

[95] Vladimir Kolmogorov and Yuri Boykov. What metrics can be approximated by geo-cuts, or global optimization of length/area and flux. In *Tenth IEEE International Conference on Computer Vision (ICCV'05) Volume 1*, volume 1, pages 564–571. IEEE, 2005.

[96] Paul Sweeney Jr. Crofton formulæ. 2018. [Unpublished Senior thesis], The University of Notre Dame.

[97] Joachim Ohser and Frank Mücklich. *Statistical analysis of microstructures in materials science*. Wiley, 2000.

[98] Gaëtan Lehmann and David Legland. Efficient n-dimensional surface estimation using crofton formula and run-length encoding. *Efficient N-Dimensional surface estimation using Crofton formula and run-length encoding*, 2012.

[99] Ellen LS Solomon, Anirudh Raju Natarajan, Arunabha Mohan Roy, Veera Sundararaghavan, Anton Van der Ven, and Emmanuelle A Marquis. Stability and strain-driven evolution of β' precipitate in mg-y alloys. *Acta Materialia*, 166:148–157, 2019.

[100] Aram W Harrow, Avinatan Hassidim, and Seth Lloyd. Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502, 2009.

[101] Andris Ambainis. Variable time amplitude amplification and a faster quantum algorithm for solving systems of linear equations. *arXiv preprint arXiv:1010.4458*, 2010.

[102] Andrew M Childs, Robin Kothari, and Rolando D Somma. Quantum linear systems algorithm with exponentially improved dependence on precision. *arXiv preprint arXiv:1511.02306*, 83, 2015.

[103] B David Clader, Bryan C Jacobs, and Chad R Sprouse. Preconditioned quantum linear system algorithm. *Physical review letters*, 110(25):250504, 2013.

[104] Leonard Wossnig, Zhikuan Zhao, and Anupam Prakash. Quantum linear system algorithm for dense matrices. *Physical review letters*, 120(5):050502, 2018.

[105] Dominic W Berry. Quantum algorithms for solving linear differential equations. *arXiv preprint arXiv:1010.2745*, 2010.

[106] Sarah K Leyton and Tobias J Osborne. A quantum algorithm to solve nonlinear differential equations. *arXiv preprint arXiv:0812.4423*, 2008.

[107] Ashley Montanaro and Sam Pallister. Quantum algorithms and the finite element method. *Physical Review A*, 93(3):032324, 2016.

[108] Yudong Cao, Anargyros Papageorgiou, Iasonas Petras, Joseph Traub, and Sabre Kais. Quantum algorithm and circuit design solving the poisson equation. *New Journal of Physics*, 15(1):013021, 2013.

[109] Jian Pan, Yudong Cao, Xiwei Yao, Zhaokai Li, Chenyong Ju, Hongwei Chen, Xinhua Peng, Sabre Kais, and Jiangfeng Du. Experimental realization of quantum algorithm for solving linear systems of equations. *Physical Review A*, 89(2):022313, 2014.

[110] René Steijl and George N Barakos. Parallel evaluation of quantum algorithms for computational fluid dynamics. *Computers & Fluids*, 2018.

[111] Pedro Costa, Stephen Jordan, and Aaron Ostrander. Quantum algorithm for simulating the wave equation. *arXiv preprint arXiv:1711.05394*, 2017.

[112] F Fillion-Gourdeau and E Lorin. Simple digital quantum algorithm for symmetric first order linear hyperbolic systems. *arXiv preprint arXiv:1705.09361*, 2017.

[113] He Sun and Jing Zhang. Solving lyapunov equation by quantum algorithm. *Control Theory and Technology*, 15(4):267–273, 2017.

[114] Stephen Russell Lee, Francis Joseph Alexander, Kipton Marcos Barros, Marcus G. Daniels, James R. Gattiker, Michael Scott Hamada, James Walter Howse, Josip Loncaric, Scott D. Pakin, Rolando Diego Somma, and Louis James Vernon. An early quantum computing proposal. *Los Alamos National Lab Technical Report*, LA-UR-16-22253, 2016.

[115] Ajinkya Borle and Samuel J Lomonaco. Analyzing the quantum annealing approach for solving linear least squares problems. *arXiv preprint arXiv:1809.07649*, 2018.

[116] Daniel O'Malley, Velimir V Vesselinov, Boian S Alexandrov, and Ludmil B Alexandrov. Nonnegative/binary matrix factorization with a d-wave quantum annealer. *PloS one*, 13:e0206653, 2018.

[117] Boris Moiseevich Levitan, Ishkhan Saribekovich Sargsian, and Iščhan S Sargsjan. *Introduction to spectral theory: selfadjoint ordinary differential operators: Selfadjoint Ordinary Differential Operators*, volume 39. American Mathematical Soc., 1975.

[118] "D-Wave Systems Inc.". *Technical description of the d-wave quantum processing unit*. 09-1109a-e, 2018.

[119] Jun Cai, William G Macready, and Aidan Roy. A practical heuristic for finding graph minors. *arXiv preprint arXiv:1406.2741*, 2014.

[120] Simon J Devitt, William J Munro, and Kae Nemoto. Quantum error correction for beginners. *Reports on Progress in Physics*, 76(7):076001, 2013.

[121] Catherine C McGeoch and Cong Wang. Experimental evaluation of an adiabiatic quantum system for combinatorial optimization. In *Proceedings of the ACM International Conference on Computing Frontiers*, page 23. ACM, 2013.

[122] Miguel A Carreira-Perpinan and Geoffrey E Hinton. On contrastive divergence learning. In *Aistats*, volume 10, pages 33–40. Citeseer, 2005.

[123] Navdeep Jaitly and Geoffrey Hinton. Learning a better representation of speech soundwaves using restricted boltzmann machines. In *2011 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pages 5884–5887. IEEE, 2011.

[124] SM Ali Eslami, Nicolas Heess, Christopher KI Williams, and John Winn. The shape boltzmann machine: a strong model of object shape. *International Journal of Computer Vision*, 107(2):155–176, 2014.

[125] Tijmen Tieleman. Training restricted boltzmann machines using approximations to the likelihood gradient. In *Proceedings of the 25th international conference on Machine learning*, pages 1064–1071, 2008.

[126] Tijmen Tieleman and Geoffrey Hinton. Using fast weights to improve persistent contrastive divergence. In *Proceedings of the 26th Annual International Conference on Machine Learning*, pages 1033–1040, 2009.

[127] Amir Khoshaman, Walter Vinci, Brandon Denis, Evgeny Andriyash, Hossein Sadeghi, and Mohammad H Amin. Quantum variational autoencoder. *Quantum Science and Technology*, 4(1):014001, 2018.

[128] Tarik Arici and Asli Celikyilmaz. Associative adversarial networks. *arXiv preprint arXiv:1611.06953*, 2016.

[129] Max Wilson, Thomas Vandal, Tad Hogg, and Eleanor Rieffel. Quantum-assisted associative adversarial network: Applying quantum annealing in deep learning. *arXiv preprint arXiv:1904.10573*, 2019.

[130] Jennifer Sleeman, John Dorband, and Milton Halem. A hybrid quantum enabled rbm advantage: convolutional autoencoders for quantum image compression and generative learning. In *Quantum Information Science, Sensing, and Computation XII*, volume 11391, page 113910B. International Society for Optics and Photonics, 2020.

[131] Vivek Dixit, Raja Selvarajan, Muhammad A Alam, Travis S Humble, and Sabre Kais. Training and classification using a restricted boltzmann machine on the d-wave 2000q. *arXiv preprint arXiv:2005.03247*, 2020.

[132] Nicolas Loizou and Peter Richtárik. Momentum and stochastic momentum for stochastic gradient, newton, proximal point and subspace descent methods. *Computational Optimization and Applications*, pages 1–58, 2020.

[133] Dmitry Kovalev, Konstantin Mishchenko, and Peter Richtárik. Stochastic newton and cubic newton methods with simple local linear-quadratic rates. *arXiv preprint arXiv:1912.01597*, 2019.

[134] Walter Winci, Lorenzo Buffoni, Hossein Sadeghi, Amir Khoshaman, Evgeny Andriyash, and Mohammad H Amin. A path towards quantum advantage in training deep generative models with quantum annealers. *Machine Learning: Science and Technology*, 1(4):045028, 2020.

[135] Siddhartha Srivastava, Veera Sundraraghavan, and Mohammadreza Yaghoobi. A graph-theoretic approach for multiscale modeling and prediction of crack propagation in polycrystalline materials. *Engineering Fracture Mechanics*, page 107406, 2020.

[136] Sangmin Lee and Veera Sundararaghavan. Calibration of nanocrystal grain boundary model based on polycrystal plasticity using molecular dynamics simulations. *International Journal for Multiscale Computational Engineering*, 8(5):509–522, 2010.

[137] Nicholas Fasanella and Veera Sundararaghavan. Atomistic modeling of thermomechanical properties of swnt/epoxy nanocomposites. *Modelling and Simulation in Materials Science and Engineering*, 23(6):065003, 2015.

[138] Nicholas A. Fasanella and Veera Sundararaghavan. Atomistic modeling of thermal conductivity of epoxy nanotube composites. *JOM*, 68(5):1396–1410, May 2016.

[139] A Lakshmanan, S Srivastava, A Ramazani, and V Sundararaghavan. Thermal conductivity of pillared graphene-epoxy nanocomposites using molecular dynamics. *Applied Physics Letters*, 112(15):151902, 2018.

[140] S. Ganesan and V. Sundararaghavan. An atomistically-informed energy based theory of environmentally assisted failure. *Corrosion Reviews*, 33(6):455 – 466, 2015.

[141] Shardul Panwar and Veera Sundararaghavan. Dislocation theory-based cohesive model for microstructurally short fatigue crack growth. *Materials Science and Engineering: A*, 708:395–404, 2017.

[142] S Panwar, JF Adams, JE Allison, JW Jones, and V Sundararaghavan. A grain boundary interaction model for microstructurally short fatigue cracks. *International Journal of Fatigue*, 113:401–406, 2018.

[143] D Greeley, M Yaghoobi, D Pagan, V Sundararaghavan, and J Allison. Using synchrotron radiation to improve understanding of deformation of polycrystalline metals by measuring, modelling and publishing 4d information. *IOP Conference Series: Materials Science and Engineering*, 580(1):012017, 2019.

[144] Veera Sundararaghavan and Nicholas Zabaras. Design of microstructure-sensitive properties in elasto-viscoplastic polycrystals using multi-scale homogenization. *International Journal of Plasticity*, 22(10):1799 – 1824, 2006.

[145] V. Sundararaghavan, A. Kumar, and S. Sun. Crystal plasticity simulations using nearest neighbor orientation correlation function. *Acta Materialia*, 93:12 – 23, 2015.

[146] Mohammadreza Yaghoobi, Sriram Ganesan, Srihari Sundar, Aaditya Lakshmanan, Shiva Rudraraju, John E Allison, and Veera Sundararaghavan. Prisms-plasticity: An open-source crystal plasticity finite element software. *Computational Materials Science*, 169:109078, 2019.

[147] A Githens, S Ganesan, Z Chen, J Allison, V Sundararaghavan, and S Daly. Characterizing microscale deformation mechanisms and macroscopic tensile properties of a high strength magnesium rare-earth alloy: A combined experimental and crystal plasticity approach. *Acta Materialia*, 186:77–94, 2020.

[148] Sriram Ganesan, Mohammadreza Yaghoobi, Alan Githens, Zhe Chen, Samantha Daly, John Allison, and Veera Sundararaghavan. The effects of heat treatment on the response of we43 mg alloy: crystal plasticity finite element simulation and sem-dic experiment. *International Journal of Plasticity*, page 102917, 2020.

[149] Mohsen Taheri Andani, Aaditya Lakshmanan, Mohammadreza Karamooz-Ravari, Veera Sundararaghavan, John Allison, and Amit Misra. A quantitative study of stress fields ahead of a slip band blocked by a grain boundary in unalloyed magnesium. *Scientific reports*, 10(1):1–8, 2020.

[150] Mohsen Taheri Andani, Aaditya Lakshmanan, Veera Sundararaghavan, John Allison, and Amit Misra. Quantitative study of the effect of grain boundary parameters on the slip system level hall-petch slope for basal slip system in mg-4al. *Acta Materialia*, 200:148–161, 2020.

[151] V Sundararaghavan and N Zabaras. Classification and reconstruction of three-dimensional microstructures using support vector machines. *Computational Materials Science*, 32(2):223 – 239, 2005.

[152] Veera Sundararaghavan. Reconstruction of three-dimensional anisotropic microstructures from two-dimensional micrographs imaged on orthogonal planes. *Integrating Materials and Manufacturing Innovation*, 3(1):19, Jun 2014.

[153] A Kumar, L Nguyen, M DeGraef, and V Sundararaghavan. A markov random field approach for microstructure synthesis. *Modelling and Simulation in Materials Science and Engineering*, 24(3):035015, 2016.

[154] Pinar Acar and Veera Sundararaghavan. A markov random field approach for modeling spatio-temporal evolution of microstructures. *Modelling and Simulation in Materials Science and Engineering*, 24(7):075005, 2016.

[155] Iman Javaheri and Veera Sundararaghavan. Polycrystalline microstructure reconstruction using markov random fields and histogram matching. *Computer-Aided Design*, 120:102806, 2020.

[156] Pınar Acar and Veera Sundararaghavan. Do epistemic uncertainties allow for replacing microstructural experiments with reconstruction algorithms? *AIAA Journal*, pages 1–14, 2019.

[157] Siddhartha Srivastava and Veera Sundararaghavan. Machine learning in quantum computers via general boltzmann machines: Generative and discriminative training through annealing. *arXiv:2002.00792*, 2020.

[158] Pinar Acar and Veera Sundararaghavan. Linear solution scheme for microstructure design with process constraints. *AIAA Journal*, 54(12):4022 – 4031, 2016.

[159] Arindam Paul, Pinar Acar, Ruoqian Liu, Wei-Keng Liao, Alok Choudhary, Veera Sundararaghavan, and Ankit Agrawal. Data sampling schemes for microstructure design with vibrational tuning constraints. *AIAA Journal*, 56(3):1239–1250, 2018.

[160] Pinar Acar and Veera Sundararaghavan. Reduced-order modeling approach for materials design with a sequence of processes. *AIAA Journal*, 56(12):5041–5044, 2018.

[161] V Sundararaghavan and N Zabaras. On the synergy between texture classification and deformation process sequence selection for the control of texture-dependent properties. *Acta Materialia*, 53(4):1015 – 1027, 2005.

[162] Ruoqian Liu, Abhishek Kumar, Zhengzhang Chen, Ankit Agrawal, Veera Sundararaghavan, and Alok Choudhary. A predictive machine learning approach for microstructure optimization and materials design. *Scientific reports*, 5(11551):1 – 12, 2015.

[163] Veera Sundararaghavan and Nicholas Zabaras. A multi-length scale sensitivity analysis for the control of texture-dependent properties in deformation processing. *International Journal of Plasticity*, 24(9):1581 – 1605, 2008.

[164] Pınar Acar and Veera Sundararaghavan. Stochastic design optimization of microstructural features using linear programming for robust material design. *AIAA JOURNAL*, 57(1), 2019.

[165] Arindam Paul, Pinar Acar, Wei-keng Liao, Alok Choudhary, Veera Sundararaghavan, and Ankit Agrawal. Microstructure optimization with constrained design objectives using machine learning-based feedback-aware data-generation. *Computational Materials Science*, 160:334–351, 2019.

[166] Evdokia Popova, Theron M Rodgers, Xinyi Gong, Ahmet Cecen, Jonathan D Madison, and Surya R Kalidindi. Process-structure linkages using a data science approach: application to simulated additive manufacturing data. *Integrating materials and manufacturing innovation*, 6(1):54–68, 2017.

[167] Jaimyun Jung, Jae Ik Yoon, Seong-Jun Park, Jun-Yun Kang, Gwang Lyeon Kim, Yi Hwa Song, Sung Taek Park, Kyeong Won Oh, and Hyoung Seop Kim. Modelling feasibility constraints for materials design: Application to inverse crystallographic texture problem. *Computational Materials Science*, 156:361–367, 2019.

[168] Johannes Dornheim, Lukas Morand, Samuel Zeitvogel, Tarek Iraki, Norbert Link, and Dirk Helm. Structure-guided processing path optimization with deep reinforcement learning. *arXiv preprint arXiv:2009.09706*, 2020.

[169] Pinar Acar and Veera Sundararaghavan. Uncertainty quantification of microstructural properties due to variability in measured pole figures. *Acta Materialia*, 124:100 – 108, 2017.

[170] Pinar Acar and Veera Sundararaghavan. Uncertainty quantification of microstructural properties due to experimental variations. *AIAA Journal*, 55(8):2824 – 2832, 2017.

[171] Pinar Acar, Siddhartha Srivastava, and Veera Sundararaghavan. Stochastic design optimization of microstructures with utilization of a linear solver. *AIAA Journal*, 55(9):3161–3168, 2017.

[172] Siddhartha Srivastava and Veera Sundararaghavan. Box algorithm for the solution of differential equations on a quantum annealer. *Physical Review A*, 99(5):052355, 2019.

[173] Hassan K. Khalil. Fundamental properties. In *Nonlinear systems*, pages 87 – 110. Prentice Hall, 2002.