The theory and practice of benchmarking quantum annealers

Joshua Job

October 18, 2018

Abstract

I present an overview of my work on benchmarking quantum annealing devices, both the experimental work, and my work on understanding the fundamental principles and guidelines required. Here, I provide an overview of much of the work done in the field, define various forms of quantum speedup one may search for and the appropriate statistical methods for benchmarking noisy quantum systems with systematic errors dependent on many free parameters. I then apply much of this thinking in several test cases, including random Ising problems, Ising problems with planted solutions (which address a major challenge of benchmarking — namely the difficulty of success verification for arbitrary large problems), and the training of a binary classifier in the context of a high energy physics seeking to distinguish Higgs boson decays from background processes. I then summarize the lessons learned, and present them to the community in the hopes they can improve the quality and speed of future work in this area.

Acknowledgements

I would like to thank my advisor Professor Daniel Lidar for his support and mentorship throughout my PhD, as this thesis would not have been possible without him and his dedication to accuracy and tracking down all possible leads. I also want to thank all my collaborators, as they were critical to the completion of our research projects together. I want to thank all the teachers I've had throughout the years, who helped get me to my PhD.

I also want to thank my whole family, particularly my parents who indulged my (rather intense) interests in science from a very young age. Without your support of my interests, goals, and development, I wouldn't be where or who I am.

To my friends, who I will not name for fear of offending anyone, thank you for serving as walls to bounce ideas off of, and for many a relevant and utterly irrelevant discussion.

Finally, I'd like to thank all the brilliant physicists and scientists about whom and about whose work I read about in books and all the authors of science fiction that collectively inspired my imagination as a child, and who continue to do so today.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

How do we benchmark the performance of quantum annealers and similar systems? How do we analyze our state of knowledge of that performance, if it is subject to a variety of nuisance parameters? How should we even define quantum speedup? In this dissertation, I will present work that I in collaboration with colleagues have done to answer these questions and apply those answers to a few test cases.

1.1 Introduction

As we look forward to the rapid development of new quantum computing devices with hundreds or thousands of qubits, particularly commercial devices and non-gate-based devices such as quantum annealers, we are faced with a challenge. How does one ensure such devices really do what they claim, and aren't effectively classical? How does one evaluate the performance of such a device, what methods should one use to estimate performance on a given metric, and what metrics should one use? How do we do maintenance on the quantum state and ensure we can prevent or correct breakdowns and errors? These questions have to be settled before we can decide where to take our device on a test drive, and what problems we should use our quantum computing devices to try to solve.

At this time, these new devices and plans for quantum annealing devices and various other quantum computing platforms are no longer the first of their kind. Several generations of programmable quantum annealers from D-Wave Systems have been made available to a small community of researchers, which has worked hard to answer the aforementioned questions. This community began largely groping in the dark, and has over the last five years answered many of the most basic questions, developing techniques to validate quantum annealers, methods to benchmark and estimate performance, and developing methods to suppress errors given the constraints of existing quantum annealers.

I have been fortunate to be members of the aforementioned community,

which has given us an opportunity to work with the first several generations of quantum annealers, starting from the first commercially available such device, the 128-qubit D-Wave One "Rainier" processor, through two more generations of 512 and 1152 qubits, to the current 2048-qubit D-Wave 2000Q processor.[1] I have given significant thought to many of the aforementioned questions – What is quantum speedup? How would we recognize it if it was there? Most importantly, how can we think rigorously about the state of our knowledge about the performance of a quantum annealer, particualrly when it is noisy or subject to a large space of nuisance parameters?

The discussion will draw mainly from the research I and colleagues have done on quantum annealers, and I apologize in advance to the many others who have contributed to this enterprise for not doing their work justice. I expect that some of the lessons learned will inform studies of future classes of quantum computing devices with many qubits.

My presentation of other's work aims to remain at a fairly high level, without giving a detailed technical account, for which I refer the reader to the original literature cited.

In this chapter 1, I will review some of the theory and literature of quantum annealing, describe some of the solvers that are often used to solve the Ising-type problems that existing annealers are meant to solve, and review some of the work on characterizing these systems. This chapter was heavily based on sections of my publications, particularly Ref [1].

In chapter two 2, I will present work on defining and detecting quantum speedup on a quantum annealer, and the initial application of that work toward benchmarking the D-Wave Two annealer using random Ising problems. This chapter was originally published as Ref [2].

In chapter three 3, I will look in detail at the problem of estimating the performance of an algorithm (in particular a noisy quantum annealer). In particular, I'll discuss why one needs to think carefully about one's state of knowledge about the performance of an algorithm and how that differs from certain standard statistical techniques. That discussion also addresses some of the concerns with efficiency of benchmarking difficult problem classes, and proposes a mechanism to potentially significiantly reduce the computational effort of that task.

In chapter four 4, we'll again return to the application of some of the insights from chapters two and three, and give a partial solution to yet another problem in benchmarking – how to compare algorithms on problems so difficult you cannot solve them via bruteforce. This is done through the introduction of planted solutions in Ising problems. We'll also address a theorem concerning

[1] A brief history: the Rainier processor (108 operational qubits) was the first to be installed at the USC-Lockheed Martin Quantum Computing Center at the USC Information Sciences Institute in 2011. Upgrades to the "Vesuvius" (504 operational qubits) and "Washington" (1098 operational qubits) processors followed in 2013 and 2016, respectively. Google installed "Vesuvius" (509 operational qubits) and "Washington" (1097 operational qubits) processors in the same years at NASA Ames. Los Alamos National Lab installed a "Washington" processor (1095 operational qubits) in 2016. The 2000Q processor is now deploye at NASA Ames.

the need to find an optimal annealing time for generic thermal or quantum annealing-type algorithms. This chapter was originally published as Ref [3].

In chapter five 5, we apply all these insights again to a very new context — the use of quantum annealers in a machine learning context, namely binary classification. There we develop a model we call "quantum annealing for machine learning" or QAML and apply it to the problem of identifying Higgs boson decays in a sea of background events. This provides some additional motivation for considering how to benchmark algorithms where the algorithm in question is embedded into a complex analysis pipeline. This chapter was originally published as Ref [4].

Finally, in chapter six 6, we review all we've learned about the task of benchmarking and summarize a set of principles to guide future efforts. This chapter is a partly based on Ref [1].

1.2 The Ising Hamiltonian and quantum annealing

Consider the problem of finding the ground state of an Ising spin glass model described by a "problem Hamiltonian"

$$H_{\text{Ising}} = -\sum_{i\in\mathcal{V}} h_i\sigma_i^z - \sum_{(i,j)\in\mathcal{E}} J_{ij}\sigma_i^z\sigma_j^z \ , \tag{1.1}$$

with N binary variables $\sigma_i^z = \pm 1$. The local fields $\{h_i\}$ and couplings $\{J_{ij}\}$ are fixed and define a problem instance of the Ising model. The spins occupy the vertices $\mathcal{V}$ of a graph $G = \{\mathcal{V}, \mathcal{E}\}$ with edge set $\mathcal{E}$. For most of the work here, the graph is some variant of the Chimera graph, though chapter 5 includes work on fully connected Ising models. For an extremely broad ensemble of graphs, the Ising problem is NP-hard [5].

1.2.1 Quantum annealers

Quantum annealers are envisioned as a way of solving (finding the ground state) of an Ising problem. To perform quantum annealing one maps the Ising variables σ_i^z to Pauli z-matrices and adds a transverse magnetic field in the x-direction to induce quantum fluctuations, thus obtaining the time-dependent quantum Hamiltonian

$$H(t) = -A(t)\sum_i \sigma_i^x + B(t)H_{\text{Ising}} \ , \quad t \in [0, t_a] \ . \tag{1.2}$$

The annealing schedule starts at time $t = 0$ with just the transverse field term (i.e., $B(0) = 0$) and $A(0) \gg k_B T$, where T is the temperature, which is kept constant. The system is then in a simple quantum state with (to an excellent approximation) all spins aligned in the x direction, corresponding to a uniform superposition over all 2^N computational basis states (products of

eigenstates of the σ_i^z). During the annealing process the problem Hamiltonian magnitude $B(t)$ is increased and the transverse field $A(t)$ is decreased, ending with $A(t_a) = 0$, and couplings much larger than the temperature: $B(t_a)\max(\max_{ij}|J_{ij}|, \max_i|h_i|) \gg k_B T$. At this point the system will again be trapped in a local minimum, and by repeating the process one may hope to find the global minimum. Quantum annealing can be viewed as a finite-temperature variant of the adiabatic quantum algorithm [6], typically thought of as being restricted to classical final Hamiltonians, as above.

The D-Wave devices [7, 8, 9, 10] are designed to be physical realizations of quantum annealing using superconducting flux qubits and programmable fields $\{h_i\}$ and couplings $\{J_{ij}\}$, and have connectivity matching the Chimera graph.

The Chimera graph of a DW2X used in tests in chapter 5 is shown in Figure 1.1, with it's annealing schedule in 1.2. The ideal Chimera graph is defined as follows. Each unit cell is a balanced $K_{4,4}$ bipartite graph. In the ideal Chimera graph the degree of each vertex is 6. The left-hand side of each unit cell has connections to corresponding qubits in unit cells above and below, while the right-hand side has connections to corresponding qubits in cells to the left and right.A $N = 8L^2$-vertex Chimera graph comprises an $L \times L$ grid of $K_{4,4}$ unit cells, and the (so-called TRIAD) construction of Ref. [11] can be used to embed the complete $4L + 1$-vertex graph K_{4L+1}. The treewidth of such a graph is $4L+1 \sim \mathcal{O}(\sqrt{N})$ [11]. Dynamic programming can always find the true ground state of the corresponding Ising model in a time that is exponential in the treewidth, i.e., that scales as $\exp\left(a\sqrt{N}\right)$.

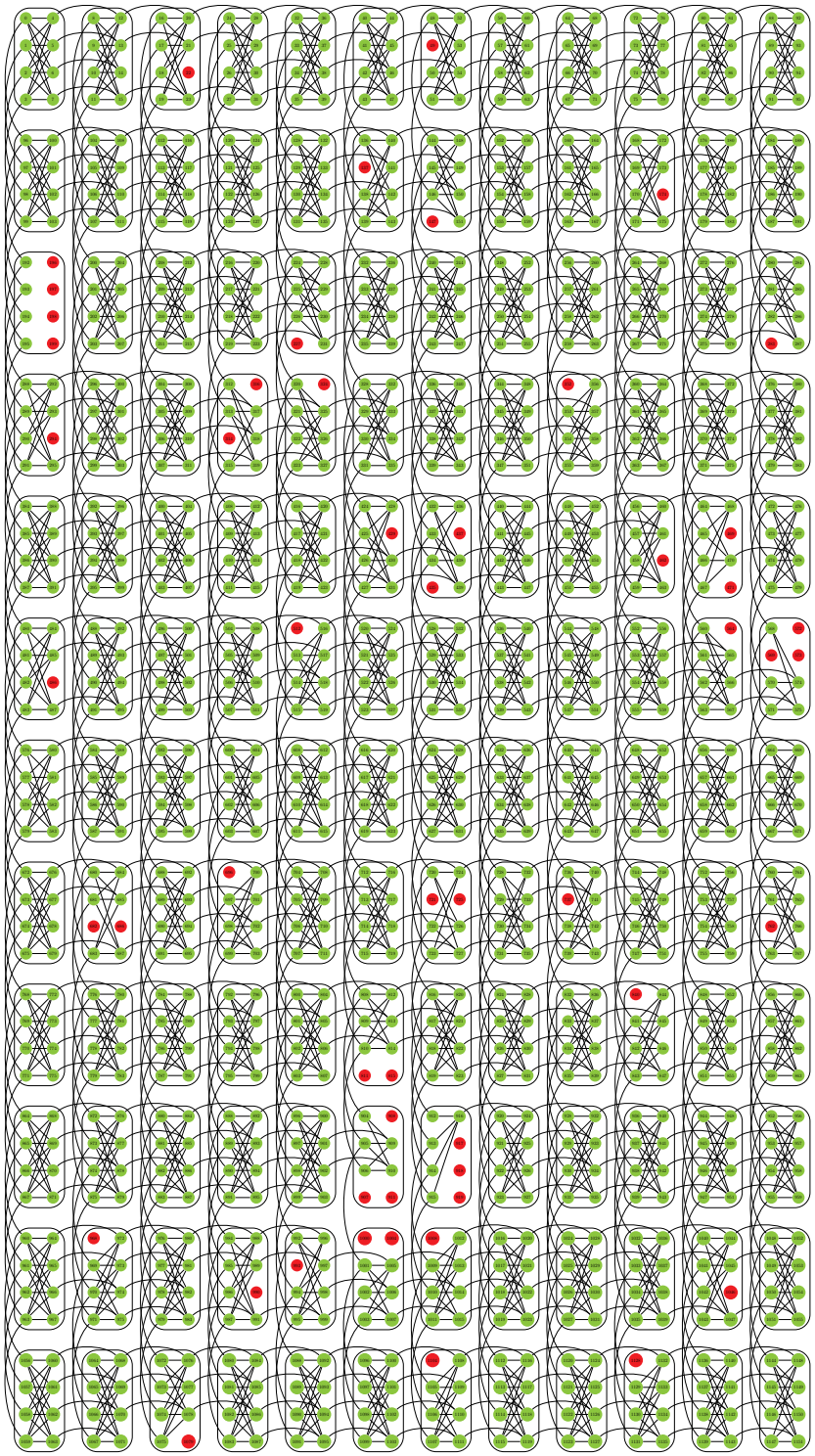


Figure 1.1: An 1152 qubit Chimera graph describing the D-Wave Two X processor at the University of Southern California's Information Sciences Institute. Inactive qubits are marked in red, active qubits (1098) are marked in green. Black lines denote active couplings (where J_{ij} is programmable to be in the range $[-1, 1]$) between qubits.

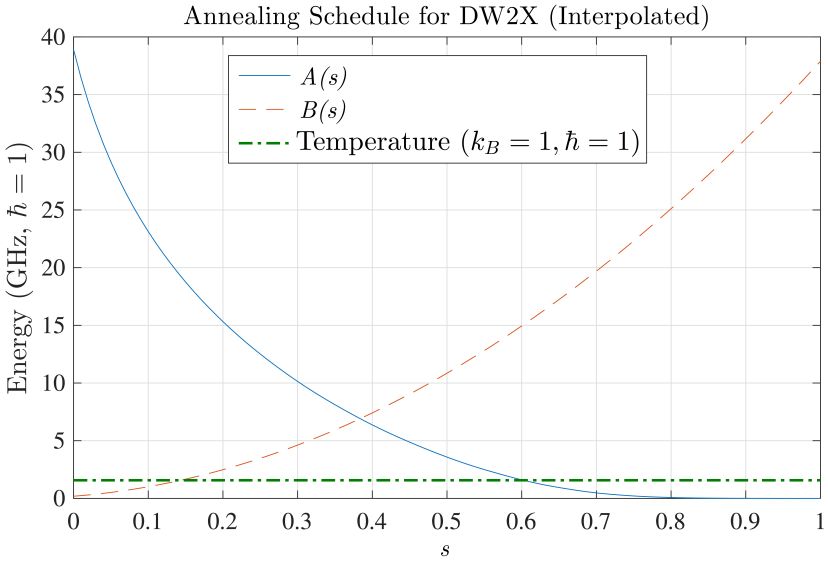


Figure 1.2: Annealing schedules for the D-Wave Two X processor described in Fig. 1.1.

1.3 Classical solvers

The choice of classical solvers against which to compare the quantum device involves a few considerations. It is important to perform an apples-to-apples comparison, in that if the device is probabilistic, it would be misleading to measure its performance against a deterministic algorithm [12, 13, 14]. For a quantum annealer, a performance comparison to known heuristic algorithms for sampling low-energy states from Ising models is natural, such as SA [15, 16], parallel tempering (PT) [17, 18, 19], and the Hamze-Freitas-Selby (HFS) algorithm (which searches all states on nodes that make up induces trees or small-treewidth subgraphs of the Ising model's connectivity graph) [20, 21]. One might also compare to approximations of QA itself, in particular simulated quantum annealing (SQA) [22, 23, 24], or the SVMC algorithm [25]. All of these can be said to be "solvers" for the Ising problem on QA. But, to determine if the quantum device is truly useful in practice, it must also be compared to the *best* algorithm for solving the *original* (typically non-Ising problem) task. For example, when solving the graph isomorphism problem [26], job-shop scheduling [27], operational planning [28], or portfolio optimization [29], the original problem must first be mapped into an Ising problem [30] and then embedded using the existing hardware connectivity graph [31, 11, 32]; the performance of the quantum device must be compared to the best algorithm for solving the original problem, and the mapping plus embedding steps can severely reduce performance. Note also that determining what the truly optimal classical algorithm is can be a daunting, or even impossible, challenge. In many cases one settles for an educated guess: the standard and/or currently best known algorithm(s). Finally, it is important to remember that any tests run on a quantum device that does not enjoy a fault tolerance guarantee cannot be reliably extrapolated to arbitrarily large sizes. I.e., in the absence of such a guarantee, a finite-size device provides evidence of what can be expected at larger sizes only provided that quantities such as the device temperature, coupling to the environment, and calibration and accuracy errors, can be appropriately scaled down. With this in mind, let us turn to a discussion of much of the benchmarking work done so far and some of the considerations that go into using large, noisy quantum devices.

1.3.1 A brief description of some of the most relevant algorithms for solving Ising-type problems

HFS algorithm

The HFS algorithm is due to Hamze & Freitas [20] and Selby [21]. This is a tree-based optimization algorithm, which exploits the sparsity and local connections of the Chimera (or other) graph to construct very wide induced trees and repeatedly optimizes over such trees until no more improvement is likely. It may also be used to sample from the marginal Gibbs state of each tree.

Let's briefly discuss the tree construction of the HFS algorithm. It considers each part of the bipartite unit cell as a single 2^4-dimensional vertex instead of

four distinct 2-dimensional vertices, resulting in a graph as depicted in Fig. 1.3. The tree represented by the dark vertices covers $\approx 78\%$ of the graph, and such trees will cover 75% of the graph in the limit of infinitely large Chimera graphs. Finding the minimum energy configuration of such a tree conditioned on the state of the rest of the graph can be done in $\mathcal{O}(N)$ time where N is the number of vertices. Since the tree encodes so much of the graph, optimizing over such trees can quickly find a low-lying energy state. Since each sample takes a variable number of trees, we generally estimate time to solution as as the average number of trees multiplied by the number of operations per tree, with a time constant of $0.5\mu s$ per operation (derived from experiment).

More generally, one can also implement HFS in a graph-independent way, but building trees dynamically, however no studies in this work used this method.

Simulated Annealing

The simulated annealing algorithm [15, 16] typically uses a single spin-flip Metropolis update method. In a single sweep, each spin is updated once according to the Metropolis rule: the spin is flipped, the change in energy ΔE is calculated. If the energy is lowered, the flip is accepted, and if not, it is accepted with a probability given by the Metropolis probability:

$$P_{\text{Met}} = \min\left(1, \exp(-\beta \Delta E)\right) \tag{1.3}$$

A linear annealing schedule in β is typically used; Starting at $\beta_i = 0.01$, we increment β in steps of $\delta\beta = (\beta_f - \beta_i)/(S-1)$ where S is the number of sweeps, up to β_f.

Parallel Tempering

Parallel tempering (PT) [17, 18, 19] is very similar to SA, except that one encodes many copies of the system at different temperatures, performing Monte Carlo update sweeps on each copy and then performing a swap between neighboring systems in temperature space with another Metropolis update. Choosing the ensemble of temperatures is relatively difficult, but there are heuristics which typically yield good results and are efficient [19]. Advanced versions with clever update mechanisms specifically tailored for Ising models, such as the isoenergetic cluster move discussed in Ref [33], have proven to be exceptionally powerful. However, parallel tempering will receive no further attention here, as it was not used in any of the studies I've published.

SSSV/SVMC

The SSSV (Shin, Smith, Smolin & Vazirani) or SVMC (spin-vector Monte Carlo) model was first proposed [25] as a classical model that reproduced the success probabilities of the DW1 device studied in Ref. [13], although there is growing evidence that this model fails to capture the behavior of the device for specific instances [34, 35, 36]. The model can be understood as describing coherent

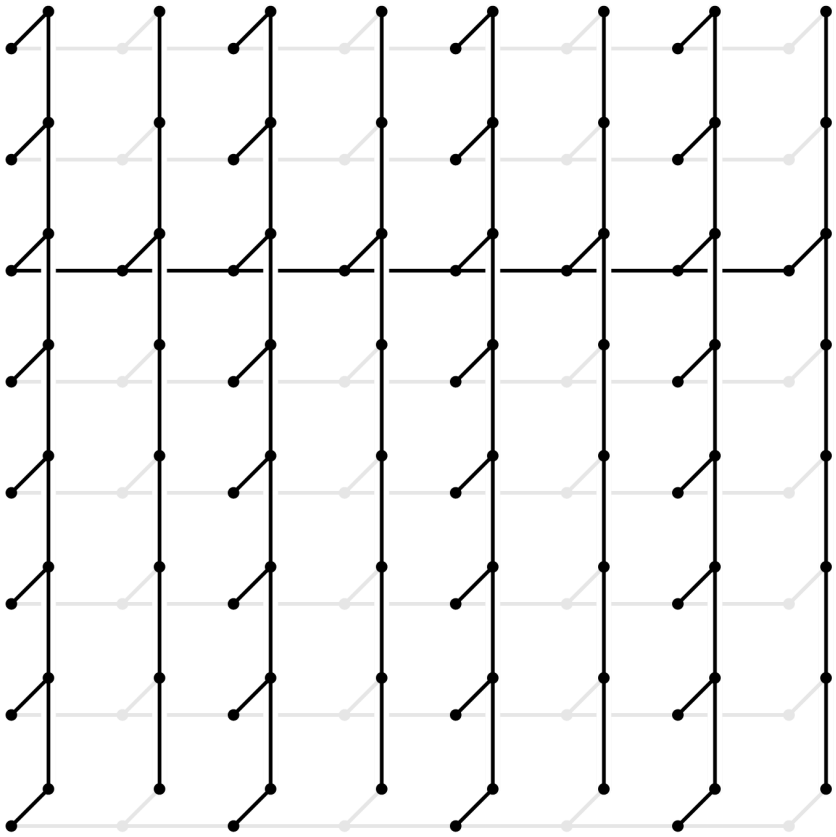


Figure 1.3: An example of the view of the Chimera graph in the HFS algorithm. Each vertex is one half of a unit cell, and is 2^4-dimensional. The algorithm repeatedly finds the minimum energy configuration of the graph over trees (like the one highlighted) given the state of the remaining vertices (in gray).

single qubits interacting incoherently by replacing qubits by O(2) rotors; the Hamiltonian can be generated by replacing $\sigma_i^x \mapsto \sin\theta_i$ and $\sigma_i^z \mapsto \cos\theta_i$. The system is then "evolved" by Monte Carlo updates on the angles $\theta_i \in [0, \pi]$. Although SSSV is not designed to be a fast solver, we studied it in chapter 4 as a potential classical limit of the DW2 and checked what the scaling of such a classical limit would be.

Simulated Quantum Annealing/Path Integral Monte Carlo

SQA/PIMC [22, 23, 24] is an annealing algorithm based on discrete-time path-integral quantum Monte Carlo simulations of the transverse field Ising model but using Monte Carlo dynamics instead of the open system evolution of a quantum system. This amounts to sampling the world line configurations of the quantum Hamiltonian (1.2) while slowly changing the couplings. SQA has been shown to be consistent with the input/output behavior of the DW1 for random instances [13], and we accordingly used a discrete-time quantum annealing algorithm. Generally 64 Trotter slices are used and an inverse temperature of $\beta = 10$ (in dimensionless units, such that $\max(|J_{ij}|) = 1$) with linearly decreasing and increasing $A(t)$ and $B(t)$, respectively are used. Cluster updates are performed only along the imaginary time direction. A single sweep amounts to the following: for each space-like slice, a random spin along the imaginary time direction is picked. The neighbors of this spin are added to the cluster (assuming they are parallel) according to the Wolff algorithm [37] with probability $1 - e^{-2J_\perp}$, where $J_\perp = -0.5 \ln[\tanh A(t)]$ is the spin-spin coupling along the imaginary time direction. When the cluster construction terminates, the cluster is flipped according to the Metropolis probability using the change in energy along the space-like direction associated with flipping the cluster. Therefore a single sweep involves a single cluster update for each space-like slice.

1.4 A quick review of quantum validation testing

Perhaps the first question one might ask when offered a quantum computational device is whether or not it is, in fact, quantum. In the case of quantum computational devices based on the circuit model and/or gates for quantum computing, the task of validation can be reduced to a Clauser-Horne-Shimony-Holt (CHSH) test between two parts of the device that are treated as black boxes [38]. Alternatively, one may opt for quantum process tomography [39, 40] or quantum gate set tomography [41, 42], wherein one applies many small computations and measures the results, verifying that they match the predictions of quantum theory. These predictions are available because the quantum computations in question typically involve few qubits and are thus readily implementable [43, 44].

However, for other quantum computing paradigms, such as quantum annealing (QA) [45, 46] and the broader field inspired by adiabatic quantum computing (AQC) [47, 48, 49, 50], quantum tomography is not currently available for

validation. This is for a variety of reasons. The key difference is that gate-based computations are modular: they can be broken into discrete time-local and space-local operations, operating effectively on only one or two qubits at a time, with the others left essentially unaffected, so the only requirement to validate even a long chain of computations is to validate those one- and two-qubit operations on individual qubits and pairs of qubits. For AQC-like platforms, the quantum computation is composed of a continuously time-varying Hamiltonian with many computational operators acting on the system at the same time. They are non-modular in the sense that they cannot be easily broken down into discrete chunks which can be validated separately. Future versions of such platforms may be more flexible and allow for approaches such as quantum tomography, but will still be unable to validate arbitrarily large computations due to the aforementioned nonmodularity of the computation. Meanwhile, partial alternatives such as tunneling spectroscopy have already been explored [51]. Of course, in the absence of error correction and fault tolerance neither the gate model nor AQC are guaranteed successful validation.

Nevertheless, certain lessons can be ported over to non-gate-based approaches. One should, as in the circuit model, focus on small problems, with a small number of qubits, and one may hope that by studying such problems applied to many such overlapping sets that one can at least partially validate the operation of the device. From here, two paths for validation become available, depending on whether one can "open the black box" and perform measurements during the anneal or use measurements beyond what may be considered "native" to the device, or whether one is only able to use the device's output at the end of complete runs for testing.

1.4.1 Types of validation: proof of quantumness, quantum supremacy, speedup-inferred quantumness, and classical model rejection

In validating quantum annealers, one seeks to create an assignment to the h's and J's such that one can take some measurements which will conclusively demonstrate, for instance, quantum entanglement, in what might be called an experimental "*proof of quantumness*".

A somewhat weaker and indirect type of validation is provided by "*quantum supremacy*" experiments [52], since they have the potential for complexity theoretic guarantees.[2] More specifically, quantum supremacy is a scenario where (part of) the polynomial hierarchy of complexity theory collapses if the quantum result could be replicated classically without slowdown [54, 55, 56, 57, 58, 59]. While weaker than a direct proof of quantumness, a demonstration of quantum supremacy would be considered strong evidence for quantum computational power of a device, which may be considered inherently more interesting than a

[2]The term "supremacy" has generated considerable controversy [53]. While we would prefer the adoption of an alternative such as "hegemony" or "supremeness", we recognize that "supremacy" is likely here to stay due to its current widespread usage.

direct demonstration of, e.g., entanglement.

"*Speedup-inferred quantumness*" is a related type of indirect validation based on a demonstration of quantum speedup [2]2 over the best classical solvers known for a task, which is often considered the holy grail of quantum information processing. Unlike quantum supremacy tests, speedup-inferred quantumness tests do not have complexity theoretic guarantees (an example in the circuit model would be Shor's algorithm [60]). It appears that an unqualified quantum speedup would necessarily have to invoke quantum properties, and this might happen even if these properties remain poorly understood or characterized. Thus a certificate of quantumness might be assigned even in the absence of a direct demonstration of quantum properties such as entanglement. It should be recognized that this carries a certain element of risk. For example, suppose a new *classical* optimization is discovered that outperforms all other classical and quantum optimization algorithms known to date (this is in fact what happened recently in a tug-of-war between quantum and classical optimization for the Max E3LIN2 problem [61]). This algorithm could be deceptively marketed as a quantum algorithm providing speedup-inferred quantumness by a shrewd company claiming to own quantum computers, that provides black-box access only to run the new optimization algorithm. Thus any claim of speedup-inferred quantumness should always be treated with a healthy degree of skepticism as related to its quantum underpinnings, until actual evidence of quantum effects driving the algorithm is presented.

If none of prior three types (proof of quantumness, quantum supremacy, speedup-inferred quantumness) of validation are attainable, one may alternatively seek to show that on sufficiently small scale problems the results are only readily reproducible using a truly quantum model of the device, and cannot be replicated qualitatively using any existing classical model, in what might be called "*classical model rejection*". This type of validation experiment does not provide a certificate of quantumness, since one can always invent a new and better classical model. Instead, one can only hope to exclude all "physically reasonable" classical models for the device. Moreover, classical model rejection can only be performed as long as it is feasible to carry out quantum model simulations, which limits system sizes to about 20 qubits for master equation type models, using the quantum trajectories method [62]. Extrapolations to larger sizes are, as always, risky in the absence of fault tolerance guarantees.

One caveat regarding "proof of quantumness" experiments is noteworthy. While demonstrations of entanglement can be considered "proof of quantumness", they often require additional physical resources and measurement possibilities beyond those that may natively be embedded in a (commercial) quantum computational device or that are strictly required to implement the core algorithm, and thus may be impossible on certain platforms. Additionally, in practice, certain assumptions may be made in a "proof of quantumness" experiment which, when relaxed, render it effectively a "classical model rejection" experiment; we shall shortly see an example of this with the D-Wave quantum annealers.

1.4.2 Experimental implementations of quantum validation tests

The primary "proof of quantumness" experiment for quantum annealers was performed in Ref. [63], using an entanglement test on the D-Wave Two (DW2) generation of processors. Briefly, the work used quantum tunneling spectroscopy [51] to estimate the populations of the first and second excited states of a combined probe-system Hamiltonian. They also measured the energy spectrum and found it to be consistent with the Hamiltonian the device was designed to implement, which provided a justification for the assumption that the measured populations were those of the energy eigenstates of the Hamiltonian. This allowed for a reconstruction of the density matrix under the assumption that it is diagonal in the energy eigenbasis, enabling a computation of the negativity [64] for all possible bipartitions of the system, the geometric mean of which was taken as a measure of the entanglement of the system. As it was found to be nonzero, the system is entangled. Further, by exploiting the theory of entanglement witnesses [65], Ref. [63] was able to show that even if the diagonality assumption is relaxed, the entanglement remains. This was used to conclude that the DW2 system tested displays entanglement at least on the scale of a single 8-qubit unit cell.

It was noted in Ref. [66] that these tests depended on the assumption that the device was well-described by Eq. (1.1) for an appropriate (programmed) choice of local fields and couplers for which the ground state is entangled, and that this assumption is not directly demonstrable by the experiments in [63]. Without that assumption, one must revert to a "classical model rejection" experiment in which one compares results of direct quantum simulations of the device and available classical alternatives to demonstrate that only the quantum model is consistent with the experimental observations. Ref. [66] provides a detailed description of the experiments, but for our purposes the key takeaway is that only the quantum adiabatic master equation [67] can reproduce the output distribution from experiments, validating the approach in Ref. [63].

Another branch of validation experiments of the classical model rejection type are the so-called "quantum signature" Hamiltonians and the consistency tests derived therefrom, introduced in Ref. [68], critiqued in Ref. [69], and further explored in Refs. [70, 34, 71]. Unlike the aforementioned entanglement tests, these experiments do not require access to the system during the annealing process, and are appropriate for cases in which the quantum device is a "black box" in which one can only control the inputs and measure the outputs. An example of a quantum signature Hamiltonian is shown in Fig. 1.4. These Hamiltonians take the form of a ring of tightly bound qubits each connected to a single outer qubit. The resulting Hamiltonian has the property that there is a large ($2^{N/2}$-dimensional) degenerate subspace of ground state configurations corresponding to arbitrary assignments to the outer qubits where all qubits in the inner ring are in the state 0 (forming a "cluster" connected by single spin flips applied to the outer ring). There is one additional ground state corresponding to flipping all the inner qubits to the state 1, dubbed the "isolated" state,

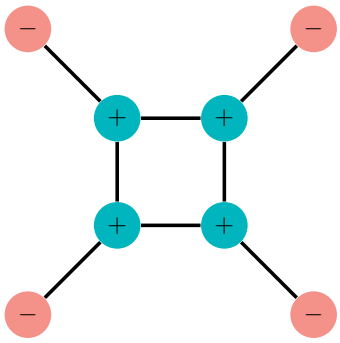


Figure 1.4: The eight-spin Ising quantum signature Hamiltonian introduced in Ref. [68]. The inner "core" spins (green circles) have local fields $h_i = +1$ while the outer spins (red circles) have $h_i = -1$. All couplings are ferromagnetic: $J_{ij} = 1$ (black lines).

since for a signature Hamiltonian with $2N$ qubits it is at least N spin flips away from all other ground states. Thermal algorithms such as classical simulated annealing (SA) [15] will be weighted toward the isolated state, such that it will have the highest probability of occurrence of any ground state configuration, whereas an adiabatic quantum evolution will find the isolated state to be suppressed relative to the cluster states. Extensive simulations and experiments on a 108 qubit D-Wave One (DW1) "Rainier" processor matched qualitatively with the adiabatic master equation across all the parameters and statistics of the output distributions tested, though noise due to cross-talk made it very difficult to find quantitative agreement; at the same time all existing classical models failed to qualitatively match the DW1 in at least one of the tests [34].

A different approach to classical model rejection was taken in Ref. [13], which used random $J_{ij} = \pm 1$ instances of the Ising Hamiltonian in Eq. (1.1) to test the hypothesis that quantum annealing correlates well with two classical models: SA and classical spin dynamics [69] (also known as the Landau-Lifshitz-Gilbert model). The hypothesis was tested using the same DW1 processor. This work showed that these two classical models failed to correlate with the results for the distribution of ground state probabilities generated by the DW1 device, while the DW1 correlated very well with simulated quantum annealing (SQA), implemented using quantum Monte Carlo [22]. This was taken as evidence for quantum annealing on the scale of more than 100 qubits, thus generalizing the conclusion of the earlier result [68] based on the 8-qubit "gadget" shown in Fig. 1.4. Shortly thereafter a new semiclassical Spin-Vector Monte Carlo (SVMC) model was introduced, also known as SSSV, the author initials of Ref. [25].[3] In this model spins are treated as $O(2)$ rotors (effectively as single qubits), evolved according to the annealing schedule given in Eq. (1.2), with

[3]Both the spin-dynamics and SVMC models can be derived in a strong coupling limit from the anisotropic Langevin equation, starting from Keldysh field theory [72].

Monte Carlo angle updates. The SVMC model correlated well with both the DW1 and SQA data, suggesting that although the DW1 device's performance is consistent with quantum annealing, it operated in a temperature regime where, for most random Ising spin glass instances, a quantum annealer may have an effective semi-classical description. This conclusion was challenged in Ref. [35], which considered the excited state distribution rather than just the ground state distribution over random $J_{ij} = \pm 1$ Ising instances, as well as the ground state degeneracy. This work presented evidence that for these new measures neither SQA nor SVMC, which are classically efficient algorithms, correlated well with the DW1 experiments. The close correlation SQA and the SVMC model was explained by showing that the SVMC model represents a semiclassical limit of the spin-coherent states path integral, which forms the foundation for the derivation of the SQA algorithm.

The intense debate that arose around the original classical model rejection tests presented in Refs. [68, 13], in particular the critique presented in Refs. [69, 25], illustrates the risks associated with such tests—risks that materialize whenever a sufficiently clever new classical model is found that agrees with (some of) the data—as well as the fruitfulness of the classical model rejection approach, which can lead to a healthy updating and sharpening of models and assumptions.

Black box classical model rejection tests such as the quantum signature Hamiltonians provide the basis for the testing of new putative quantum devices for which available controls and potential measurements are limited, and ultimately even the best experiments that seek to prove entanglement will depend on a series of such experiments to demonstrate that only quantum models can reproduce the experimental data from the device. Quantum supremacy tests are a type of limiting case of this, in which one can prove that should any classical device be able to produce a particular output distribution in polynomial time then the computational complexity hierarchy will at least partially collapse. Since this is not expected to occur, building a device which can produce said distribution efficiently will then immediately rule out all classical models for the device [54].

Another kind of black box classical model rejection test is based on the phenomenon of quantum tunneling, whereby a quantum state has sizable probability on either side of an energy barrier which the system could not move through classically, or at least will only be able to do so with reasonable probability at high temperature. The first quantum annealing experiments involving tunable tunneling were carried out using the disordered ferromagnet $LiHo_xY_{1-x}F_4$ in a transverse magnetic field [73, 74], and served as inspiration for the design of programmable superconducting flux-qubit based quantum annealers. These experiments indicated that quantum annealing hastens convergence to the optimum state via tunneling, compared to simple thermal hopping models. The first programmable quantum annealer experiment was reported in Ref. [75], in which it was demonstrated that an 8-qubit quantum annealing device was able to reproduce the domain wall tunneling predictions of quantum theory for a chain of superconducting flux qubits by modifying the time during the annealing process

at which a local field is abruptly applied to the qubits. This contradicted the temperature dependence predictions of a classical thermal hopping model, thus serving as a classical model rejection experiment.

More recently, Ref. [36] reported on a specially designed tunneling probe Hamiltonian for quantum annealing, illustrated in 1.5. The probe uses two unit cells of the D-Wave Chimera graph, binding each one together tightly so they each act like a single effective spin, or cluster. Opposite magnetic fields are applied to each unit cell, one weak and one one strong, so that the spins in the "strong" cluster align before the spins in the "weak" cluster. Initially, there is only a single minimum. A second minimum develops over the course of the anneal, and eventually becomes the global minimum of the final Ising Hamiltonian. The only way to reach the global minimum is to overcome an energy barrier whose strength increases as the anneal progresses, a classic example of tunneling. Using the non-interacting blip approximation (NIBA) it was shown in Ref. [36] that the system effectively acts like a two-level system even in the open-system setting with a strongly coupled bath. NIBA-based predictions without free parameters for tests at different values of h_L and different temperatures demonstrated very good agreement with experiments involving a DW2 device, and were not reproducible using classical models for the device such as SVMC [25]. A variant of this experiment was reported on in Ref. [76], which introduced a new class of problem instances which couples the weak-strong clusters of the tunneling probe as sub-blocks of the Hamiltonian. This work can be interpreted as an attempt to go from classical model rejection to speedup-inferred quantumness, as it claimed a large tunneling-induced constant-factor speedup over classical simulated annealing and simulated quantum annealing for a DW2X device. However, this claim was critiqued in Ref. [77] on the basis of a comparison to classical algorithms with better performance. Moreover, as we discuss below, speedup-inferred quantumness requires a demonstration of an optimal annealing time ([2] and 2), which was absent in the results reported in Ref. [76].

Validating non-gate based quantum devices will continue to be a challenge as new such systems come online, but applying combinations of the techniques discussed above, from the construction of quantum signature Hamiltonians and tunneling probes to (in)direct proofs of entanglement via entanglement witnesses and direct computation of entanglement, should allow one to boost confidence that the system obeys the predictions of quantum theory *over small scales*. The challenge remains to extend these techniques so that they are able to demonstrate conclusively that a device with hundreds or thousands of qubits displays coherence and long-range entanglement. Due to decoherence this presents a challenge for gate-based quantum devices as well even at a smaller scale [78, 79], and speedup-inferred quantumness tests may prove to be simpler to execute than direct quantumness tests even in the gate-model setting.

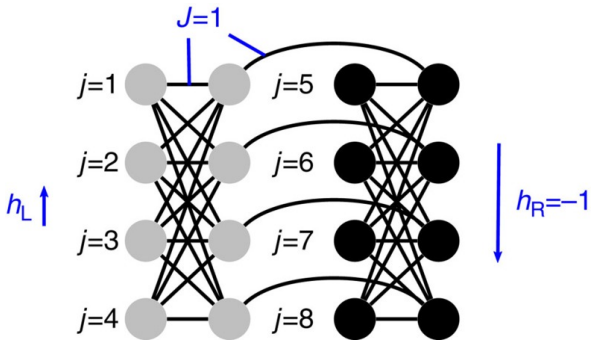


Figure 1.5: The 16-spin Ising Hamiltonian composed of two $K_{4,4}$ unit cells introduced in Ref. [36]. All couplings are set to $J = 1$, all qubits in the left unit cell have a local field $0 < h_L < 0.5$ applied to them while all spins in the left unit cell have $h_R = 1$ applied to them. Two local minima form, one with the cells internally aligned but in opposite states from each other (a local minimum) and the other with all states aligned with h_R (the global minimum). By tightly binding each unit cell, they effectively act like single large spins.

1.5 Benchmarking

Assume we have at our disposal a device verified to be quantum, at least provisionally on the small scales covered by classical model rejection, and we would like to compare its performance to competing classical solvers. This is the task we refer to here as benchmarking, which belongs more generally to the field of experimental algorithmics [80]. Specifically, consider the problem of estimating the value of some function of merit (or "reward") R from the output of a given solver (e.g., our quantum device or some classical algorithm) for a given problem family $\mathcal{P} = \{P\}$. Each problem instance P is parametrized by some parameters θ. In the case of quantum annealers, particularly studies of the D-Wave devices thus far, the goal has generally been to find the ground state of Ising Hamiltonians as defined in Eq. (1.1). In that context, typically the reward is taken to be the negative of the time to solution (TTS), defined as $\mathrm{TTS} = t_f \log(1 - p_d)/\log(1 - p)$ for a probability p of finding the ground state at least once with desired probability p_d (typically 0.99), and annealing time t_f.[4] In the language above, $R = -\mathrm{TTS}$ (one would like to minimize the TTS), and the problem is parametrized by $\theta = \{h_i, J_{ij}\}$. Many similar metrics have been proposed, such as time-to-epsilon and time-to-target [81], which amount to mild generalizations of TTS. A more elaborate notion of cost, based on optimal stopping theory, has also been considered and shown to recover the previous metrics as special cases [82]. We shall return to this below.

[4] The probability of not finding the ground state even once after k independent runs of duration t_f each is $(1-p)^k$, so the probability of finding it at least once is $1-(1-p)^k$, which we set equal to p_d. Solving for k and substituting into $\mathrm{TTS} = t_f k$ gives the TTS formula. See, e.g., Ref. [2] for a more detailed derivation.

1.5.1 Prior work on benchmarking quantum annealers

Our discussion here will focus exclusively on work that advanced the state of the art in benchmarking quantum annealers, and also primarily on those studies taken prior to or in the early phases of my PhD.

The first comprehensive study benchmarking QA devices was Ref. [13], using a 108 qubit DW1 processor. This article introduced many of the concepts used in later studies in the field, including the above definition of the TTS. It focused on the performance on the set of random Ising problems with binary ± 1 local fields and couplings, and introduced the use of SA and SQA as important comparison algorithms. It also noted the importance of comparing against parallelized versions of classical algorithms, as quantum annealers such as the D-Wave device consume linearly more computational hardware with increasing problem size, and in many cases SA and SQA can be effectively parallelized in much the same way.

Another significant contribution of Ref. [13] was the use of "gauge averaging" in benchmarking, a technique that was introduced in Ref. [68] (where it was called "spin inversions") and which has become so universal that it is now included natively in the D-Wave API for their processors, and which points toward a more general consideration for noisy quantum devices in the absence of quantum error correction. The need for gauge averaging arises from the observation that in QA, one may have per-qubit or per-edge random and systematic biases from stray fields or interactions. In such cases, performance may be dramatically impacted by the choice of mapping from a logical Hamiltonian as defined in Eq. (1.1) to a physically implemented computation. In essence, a gauge transformation corresponds to swapping which physical spin state corresponds to a computational 0 or 1. In an ideal annealer, this transformation commutes with (i.e., is a symmetry of) the total Hamiltonian and so has no dynamical effect. However in the presence of noise, this symmetry is broken and the choice of gauge does make a difference, and indeed was found to have a significant effect on the performance of the DW1 quantum annealer, to such a degree that the device did not even correlate with itself if one compares one gauge to another, or even one gauge with itself when run later (most likely the result of slow drift $1/f$ noise resulting in the effect that each time the annealer is programmed, a small random error term is added to the Hamiltonian). However, when results for the same Hamiltonian were averaged across many gauges, the DW1 processor correlated quite well with itself [13]. Since then, applying many gauge transformations to the same Hamiltonian and averaging the results has become a standard practice in the QA community, and the idea behind it has been steadily generalized since then to include sampling over every known potentially broken symmetry of the Hamiltonian.

For example, if one is solving a fully connected Ising problem, the Hamiltonian has a permutation symmetry. Since every logical spin has an interaction with every other, one can relabel which spin is which without changing anything about the logical problem. However, when one goes to implement such a problem on an actual quantum annealer with limited connectivity, such as the DW2,

one has to perform a minor embedding in which each logical spin is mapped to a chain of spins on the physical device [31, 11]. Those physical spins may have local field biases which vary from chain to chain, and thus the distribution over logical states will depend, in part, on the assignment of the logical spin variables to the physical chains, as shown in Ref. [83]. This work was the first case study of both minor embedding of fully connected problems as well as permutation embeddings for such problems, and demonstrated the importance of optimizing the strength of the coupling in minor embedding applications, a topic which is discussed in more detail in Section **??**.

Finally, Ref. [14] demonstrated evidence for the easy-hard-easy phase transition for Max 2-SAT problems (wherein one wishes to find the maximal number of simultaneously satisfiable two-variable Boolean clauses over a set of variables from some ensemble of clauses) near a clause density of one, on the 108-qubit DW1 processor. It performed a rudimentary benchmarking comparison between the DW1 and an exact Max 2-SAT solver (akmaxsat) (see also Ref. [12]), and noted that there was no correlation between the two solvers over randomly selected instances of Max 2-SAT. This work also introduced the important idea of bootstrapping into the QA community, variants of which (such as the Bayesian bootstrap [84]) formed the backbone of error analyses for later studies, as a non-parametric method for approximating the distribution over the problem space and over the aforementioned broken computational symmetries, and will be discussed in much more detail in the next chapter, 3.

Chapter 2

Defining and detecting quantum speedup

The development of small-scale quantum devices raises the question of how to fairly assess and detect quantum speedup. Here I show how to define and measure quantum speedup, and how to avoid pitfalls that might mask or fake such a speedup. I illustrate my discussion with data from tests run on a D-Wave Two device with up to 503 qubits. This study was done as a collaboration with my co-authors on the original paper, [2]. Using random spin glass instances as a benchmark, we found no evidence of quantum speedup when the entire data set is considered, and obtain inconclusive results when comparing subsets of instances on an instance-by-instance basis. Our results do not rule out the possibility of speedup for other classes of problems and illustrate the subtle nature of the quantum speedup question.

Denoting the time used by a specific classical device or algorithm to solve a problem of size N by $C(N)$ and the time used on the quantum device by $Q(N)$, we define quantum speedup as the asymptotic behavior of the ratio

$$S(N) = \frac{C(N)}{Q(N)} \tag{2.1}$$

for $N \to \infty$. Subtleties appear in the choice of classical algorithms, in defining $C(N)$ and $Q(N)$ if the runtime depends not just on the size N of a problem but also on the specific problem instance, and in extrapolating to the asymptotic limit.

Depending on our knowledge of classical algorithms for a given problem we may consider five different types of quantum speedup. The optimal scenario is one of a "provable quantum speedup," where there exists a proof that no classical algorithm can outperform a given quantum algorithm. The best known example is Grover's search algorithm [85], which, in the query complexity setting, exhibits a provable quadratic speedup over the best possible classical algorithm [86]. A "strong quantum speedup" was defined in [87] by using the performance of the

best classical algorithm for $C(N)$, whether such an algorithm is known or not. Unfortunately, the performance of the best classical algorithm is unknown for many interesting problems. In the case of factoring, for example, a proof of a classical super-polynomial lower-bound is not known. A less ambitious goal is therefore desirable, and thus one usually defines "quantum speedup" (without additional adjectives) by comparing to the best available classical algorithm instead of the best possible classical algorithm.

A weaker scenario is one where a quantum algorithm is designed to make use of quantum effects, but it is not known whether these quantum effects provide an advantage over classical algorithms or where a device is a putative or candidate quantum information processor. To capture this scenario, which is of central interest to us in this work, we define "limited quantum speedup" as a speedup obtained when comparing specifically with classical algorithms that 'correspond' to the quantum algorithm in the sense that they implement the same algorithmic approach, but on classical hardware. A natural example is quantum annealing [45, 73] or adiabatic quantum optimization [6] implemented on a candidate physical quantum information processor *vs* corresponding classical algorithms such as simulated annealing (SA) [88] (which performs annealing on a classical Monte Carlo simulation of the Ising spin glass and makes no use of quantum effects) or simulated quantum annealing (SQA) [89, 90] (a classical algorithm mimicking quantum annealing in a path-integral quantum Monte Carlo simulation). In this comparison a limited quantum speedup would be a demonstration that quantum effects improve the annealing algorithm.

The standard notion of quantum speedup depends on there being a consensus about the "best available" algorithm, and this consensus may be time- and community-dependent. For example, it may be the case, though it seems unlikely, that a classified polynomial-time factoring algorithm is available to parts of the intelligence community. In the absence of a consensus about what is the best classical algorithm, we define "potential (quantum) speedup" as a speedup compared to a specific classical algorithm or a set of classical algorithms. An example is the simulation of the time evolution of a quantum system, where the propagation of the wave function on a quantum computer would be exponentially faster than a direct integration of Schrödinger's equation on a classical computer. A potential quantum speedup can of course be trivially attained by deliberately choosing a poor classical algorithm, so that here too one must make a genuine attempt to compare against the best classical algorithms known, and any potential quantum speedup might be short-lived if a better classical algorithm is found.

Concerning the limited quantum speedup concept which is of central interest to us in the study focused on in this chapter, we only compare quantum annealing to classical simulated annealing and simulated quantum annealing. Another example of a limited quantum speedup would be Shor's factoring algorithm running on a fully coherent quantum computer *vs* a classical computer where the period finding using a quantum circuit has been replaced by a classical period finding algorithm.

2.1 Classical and quantum annealing of a spin glass

To illustrate the subtleties in detecting quantum speedup, even after a classical reference algorithm is chosen, we will compare the performance of an experimental 503-qubit D-Wave Two (DW2) device to classical algorithms and analyze the evidence for quantum speedup on the benchmark problem of random spin glass instances.

We will consider the distributions of the time to solution over many random spin glass problems with integer weighth and zero local fields on the 'Chimera graph' realized by the DW2 device. This problem is NP-hard[5], as stated in chapter 1 and all known classical algorithms scale super-polynomially not only for the hardest but also for typical instances. While quantum mechanics is not expected to reduce the super-polynomial scaling to polynomial, a quantum algorithm might still scale better with problem size N than any classical algorithm.

The approach adopted here, of seeking evidence of a (limited) quantum speedup, directly addresses the crucial question of whether large-scale quantum effects create a potential for the devices to outperform classical algorithms. To test this possibility we compare the performance of a DW2 device to two 'corresponding' classical algorithms: SA and SQA.

2.2 Considerations when computing quantum speedup

Since quantum speedup concerns the asymptotic scaling of $S(N)$ let's consider the subtleties of estimating it from small problem sizes N, and inefficiencies at small problem sizes that can fake or mask a speedup. In the context of annealing, the optimal choice of the annealing time t_a turns out to be crucial for estimating asymptotic scaling. To illustrate this we first consider the time to solution using SA and SQA run at different fixed annealing times t_a, independent of the problem size N. Figure 2.1A shows the scaling of the median total annealing time (over 1000 different random instances on the D-Wave Chimera graph – see section 2.5) for SQA to find a solution at least once with probability $p = 0.99$. Corresponding times for SA are shown in figure S10. We observe that at constant t_a, as long as t_a is long enough to find the ground state almost every time, the scaling of the total effort is at first relatively flat. The total effort then rises more rapidly, once one reaches problem sizes for which the chosen annealing time is too short, and the success probabilities are thus low, requiring many repetitions. Extrapolations to $N \to \infty$ need to consider the lower envelope of all curves, which corresponds to choosing an optimal annealing time $t_a^{\mathrm{opt}}(N)$ for each N.

Figure 2.1B demonstrates that when using fixed annealing times no conclusion can be drawn from annealing (simulated or in a device) about the asymptotic scaling . The initial slow increase at constant t_a is misleading and instead the optimal annealing time t_a^{opt} needs to be used for each problem size N. To illustrate this we show in Figure 2.1B the real "speedup" ratio of the scaling of

SA and SQA (actually a slowdown), and a fake speedup due to a constant and excessively long annealing time t_a for SQA. Since SA outperforms SQA on our benchmark set, it is our algorithm of choice in the comparisons with the DW2 reported below.

2.2.1 Resource usage and speedup from parallelism

A related issue is the scaling of hardware resources (computational gates and memory) with problem size, which must be identical for the devices we compare. A device whose hardware resources scale as N can almost always achieve an intrinsic parallel speedup compared to a fixed size device. Such is the case for the DW2, which uses N (out of 512) qubits and $\mathcal{O}(N)$ couplers and classical logical control gates to solve a spin glass instance with N spin variables in time $T_{\mathrm{DW}}(N)$. Considering quantum speedup for $N \to \infty$ we need to compare a (hypothetical) larger DW2 device with the number of qubits and couplers growing as $\mathcal{O}(N)$ to a (hypothetical) classical device with $\mathcal{O}(N)$ gates or processing units. Since SA (and SQA) are perfectly parallelizable for the bipartite Chimera graphs realized by the DW2, we can relate the scaling of the time $T_{\mathrm{C}}(N)$ on such a device to the time $T_{\mathrm{SA}}(N)$ for SA on a fixed size classical CPU by $T_C(N) \sim T_{\mathrm{SA}}/N$ and obtain

$$S(N) = \frac{T_{\mathrm{C}}(N)}{T_{\mathrm{DW}}(N)} \sim \frac{T_{\mathrm{SA}}(N)}{T_{\mathrm{DW}}(N)} \frac{1}{N}. \tag{2.2}$$

2.3 Performance of D-Wave Two versus SA and SQA

We finally address the question of how to measure time when the time to solution depends on the specific problem instance. When a device is used as a tool for solving computational problems, the question of interest is to determine which device is better for almost all possible problem instances. If instead the focus is on the underlying physics of a device then it might suffice to find a subclass of instances where a speedup is exhibited. These two questions lead to different quantities of interest.

2.3.1 Performance as an optimizer: comparing the scaling of hard problem instances

To illustrate these considerations we now turn to our results for this benchmarking study. Figure 2.2 shows the scaling of the time to find the ground state for various quantiles, from the easiest instances (1%) to the hardest (99%), comparing the DW2 and SA. We chose the values of the couplings J_{ij} from $2r$ discrete values $\{n/r\}$, with $n \in \pm\{1, \ldots, r-1, r\}$, and call r the "range". Since we do not *a priori* know the hardness of a given problem instance we have to assume the worst case and perform a sufficient number of repetitions R to be able to

solve even the hardest problem instances. Hence the scaling for the highest quantiles is the most informative. Here, we consider only the pure annealing times, ignoring setup and readout times that scale subdominantly (see 2.5 for wall-clock results).

We observe for both the DW2 and SA, for sufficiently large N, that the total time to solution for each quantile q scales as $\exp\left(c_q\sqrt{N}\right)$ (with $c_q > 0$ a constant), as reported previously for SA and SQA [13]. The origin of the $\sqrt{N}$ exponent is well understood for exact solvers as reflecting the treewidth of the Chimera graph [91, 11]. While the SA code was run at an optimized annealing time for each problem size N, the DW2 has a minimal annealing time of $t_a = 20\mu s$, which is longer than the optimal time for all problem sizes 2.5. Therefore the observed slope of the DW2 data can only be taken as a lower bound for the asymptotic scaling. With this in mind, we observe similar scaling for SA and the DW2 for $N \gtrsim 200$.

How can we probe for a speedup in light of this similar scaling? With algorithms such as SA or quantum annealing, where the time to solution depends on the problem instance, it is impractical to experimentally find the hardest problem instance. If instead we target a fraction of $q\%$ of the instances then we should consider the qth quantile in the scaling plots shown in Figure 2.2. The appropriate speedup quantity is then the ratio of these quantiles ("RofQ"). Denoting a quantile q of a random variable X by $[X]_q$ we define this as

$$S_q^{\mathrm{RofQ}}(N) = \frac{[T_{\mathrm{SA}}(N)]_q}{[T_{\mathrm{DW}}(N)]_q}\frac{1}{N}, \tag{2.3}$$

Plotting this quantity for the DW2 *vs* SA in Figure 2.3 (A and B) we find no evidence for a limited quantum speedup in the interesting regime of large N and large q (almost all instances). That is, while for all quantiles and for both ranges the initial slope is positive, when N and q become large enough we observe a turnaround and eventually a negative slope. While we observe a positive slope for quantiles smaller than the median, this is of limited interest since we have not been able to identify a priori which instances will be easy. Taking into account that due to the fixed suboptimal annealing times the speedup defined in Eq. (2.3) is an upper bound, we conclude that there is no evidence of a speedup over SA for this particular benchmark.

2.3.2 Instance-by-instance comparison

$S_q^{\mathrm{RofQ}}(N)$ measures the speedup while comparing different sets of instances for DW and SA, each determined by the respective quantile. Now we consider instead whether there is a speedup for a (potentially small) subset of the same problem instances. To this end we study the scaling of the ratios of the time to solution for individual instances, and display in Figure 2.3 (C and D) the scaling of various quantiles of the ratio ("QofR")

$$S_q^{\mathrm{QofR}}(N) = \left[\frac{T_{\mathrm{SA}}(N)}{T_{\mathrm{DW}}(N)}\right]_q \frac{1}{N}. \tag{2.4}$$

For $r = 7$ all the quantiles bend down for sufficiently large N, so that there is no evidence of a limited quantum speedup. There does seem to be an indication of such a speedup compared to SA in the low quantiles for $r = 1$, i.e., for those instances whose speedup ratio was high. However, the instances contributing here are not run at the optimal annealing time, and more work is needed to establish that the potential $r = 1$ speedup result persists for those instances for which one can be sure that the annealing time is optimal.

Next we consider the distribution of solution times at a fixed problem size. This does not address the speedup question since no scaling can be extracted, but illuminates instead the question of correlation between the performance of the DW2 and SA. To this end we perform individual comparisons for each instance and show in Figure ??A-B the time to solution for the same instances for the DW2 and SA. We observe a wide scatter (in agreement with the DW1 results of Ref. [13]) and find that while the DW2 is sometimes up to $10\times$ faster in pure annealing time, there are many cases where it is $\geq 100\times$ slower.

2.4 Discussion

It is not yet known whether a quantum annealer or even a perfectly coherent adiabatic quantum optimizer can exhibit (limited) quantum speedup at all, although there are promising indications from theory [92], simulation [90], and experiments on spin glass materials [73]. Experimental tests are thus important. There are several candidate explanations for the absence of a clear quantum speedup in the tests discussed in this chapter. Perhaps quantum annealing simply does not provide any advantages over simulated (quantum) annealing or other classical algorithms for the problem class we have studied [93]; or, perhaps, the noisy implementation in the DW2 cannot realize quantum speedup and is thus not better than classical devices. Alternatively, a speedup might be masked by calibration errors, improvements might arise from error correction [94], or other problem classes might exhibit a speedup. Future studies probed these alternatives and continue to aim to determine whether one can find a class of problem instances for which an unambiguous speedup over classical hardware can be observed.

While we used specific processors and algorithms for illustration, the considerations about a reliable determination of quantum speedup presented here are general. For any speedup analysis, using the same scaling of hardware resources for both quantum and classical devices is required to disentangle parallel and quantum speedup. And, for any quantum algorithm where the runtime must be determined experimentally, a careful extrapolation to large problem sizes is important to avoid mistaking inefficiencies at small problem sizes for signs of quantum speedup.

In section 2.5 I provide additional plots and methods information which are not be necessary to understand the main thrust of this chapter, namely the introduction of an ensemble of definitions for speedup and the initial attempts at benchmarking the performance of a quantum annealer, and the admonition to

account for hardware scaling when attempting to compute TTS. I will, however, draw the reader's attention to 2.5.3 for some additional discussion of gauge averaging, particularly as was performed in this study, and 2.5.8 for the introduction of the idea of classes of problems with tunable hardness, which will come into play in chapter 4.

Speaking of gauge averaging, the next chapter will give serious thought on how to rigorously perform gauge averaging and its consequences for the benchmarking task, as well as a more efficient method of estimating TTS across a widely varying ensemble of instances than the brute force method used in the study in this chapter. We'll also see the case for taking seriously what your statistical methods are actually telling you, and thinking carefully about what question you're really trying to answer with benchmarking.

Acknowledgments

This chapter was originally published as [2]. The following are the acknowledgements for that paper. We thank N. Allen, M. Amin, E. Farhi, M. Mohseni, H. Neven, and C. McGeoch for useful discussions and comments. We are grateful to I. Zintchenko for providing the `an_ss_ge_nf_bp` simulated annealing code before publication of the code with Ref. [95]. This project was supported by the Swiss National Science Foundation through the National Competence Center in Research NCCR QSIT, the ARO MURI Grant No. W911NF-11-1-0268, ARO grant number W911NF-12-1-0523, the Lockheed Martin Corporation and Microsoft Research. Simulations were performed on clusters at Microsoft Research and ETH Zurich an on supercomputers of the Swiss Center for Scientific Computing CSCS. We acknowledge hospitality of the Aspen Center for Physics, supported by NSF grant PHY-1066293.

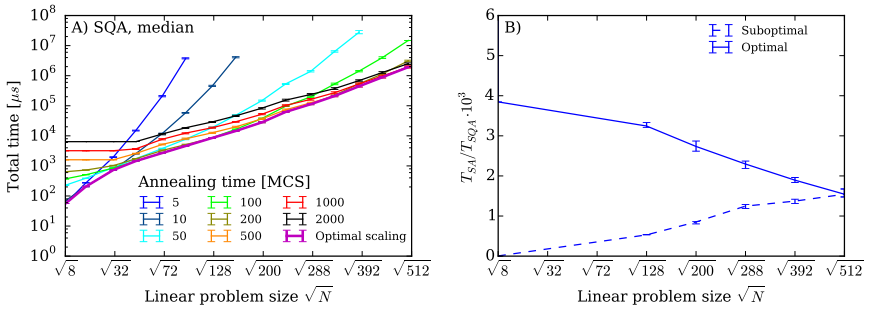


Figure 2.1: **Pitfalls when detecting speedup.** A) The typical (median) time to find a ground state at least once with 99% probability for spin glasses with ± 1 couplings using SQA at constant annealing time. The lower envelope of the curves at constant t_a corresponds to the total effort at an optimal size-dependent annealing time $t_a^{\mathrm{opt}}(N)$ and can be used to infer the asymptotic scaling. The initial, relatively flat slope at fixed N is due to suboptimal performance at small problem sizes N, and should therefore not be interpreted as speedup. Annealing times are given in units of Monte Carlo steps (MCS), corresponding to one update per spin. B) The speedup of SQA over SA for two cases. If SQA is run suboptimally at small sizes by choosing a fixed large annealing time $t_a = 10000$ MCS (dashed line) a speedup is feigned. This is due to suboptimal performance on small sizes and not indicative of the real asymptotic behavior when both codes are run optimally (solid line).

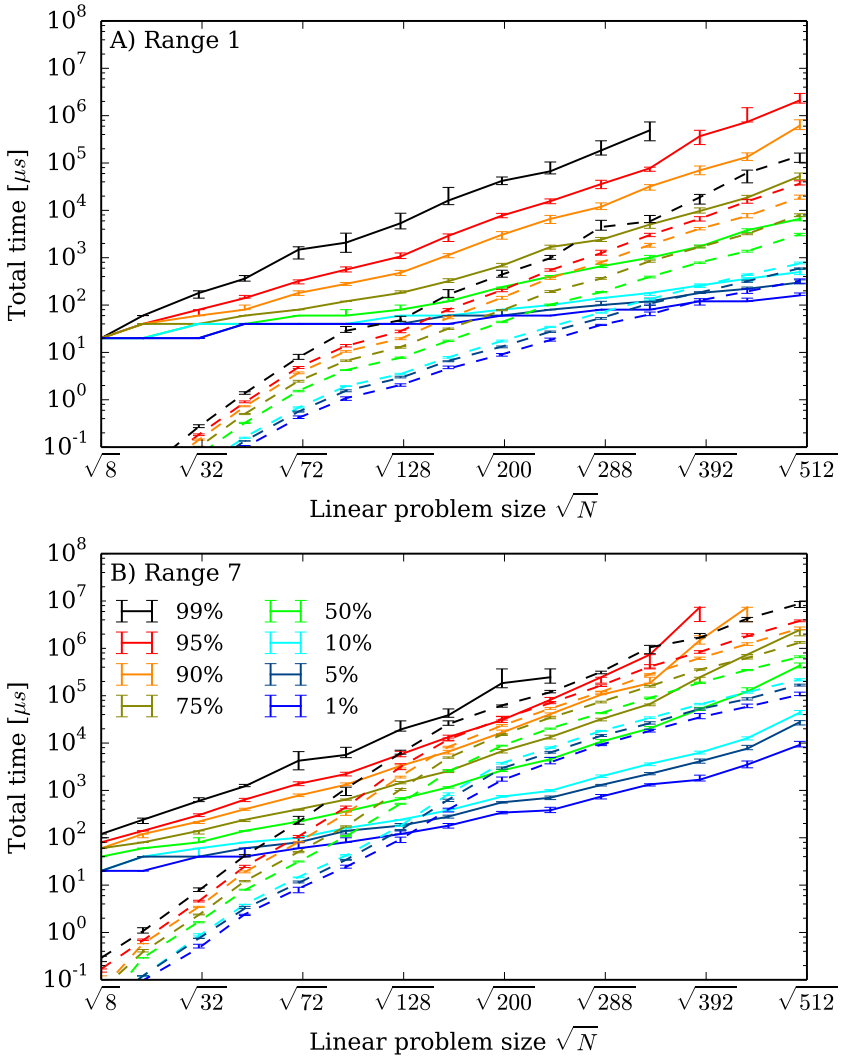


Figure 2.2: **Scaling of time to solution for the ranges $r = 1$ (panel A) and $r = 7$ (panel B).** Shown is the scaling of the pure annealing time to find the ground state at least once with a probability $p = 0.99$ for various quantiles of hardness, for simulated annealing (SA, dashed) and the DW2 (solid). The solid lines terminate for the highest quantiles because the DW2 did not solve the hardest instances for large problem sizes within the maximum number of repetitions (at least 32000) of the annealing we performed.

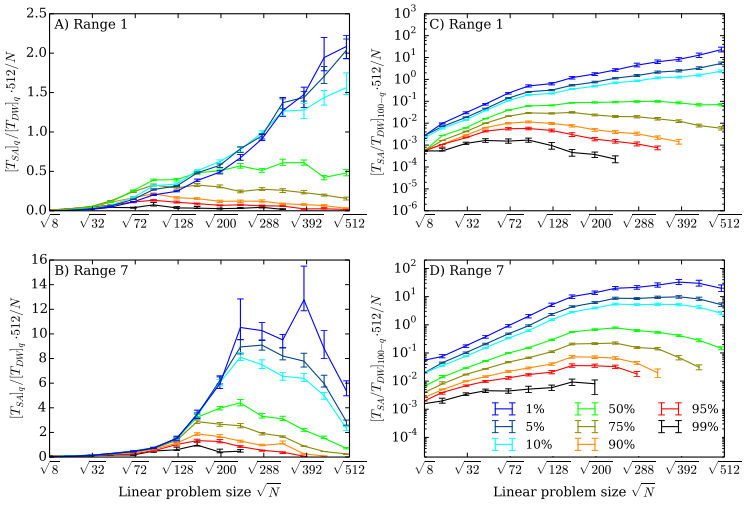


Figure 2.3: **Speedup of the DW2 compared to SA.** A) and B) for the ratio of the quantiles (RofQ), C) and D) for the quantiles of the ratio (QofR). For a random variable X with distribution $p(x)$ and values in $x \in [0, \infty)$ we define, as usual, the qth quantile as $\int_0^{x_q} p(x)dx = q/100$, which we solve for x_q and plot as a function of $\sqrt{N}$. In the QofR case we use x_{100-q} so that high quantiles still correspond to instances that are hard for the DW2. We terminate the curves when the DW2 does not find the ground state for large N at high percentiles. In these plots we multiplied Eqs. (2.3) and (2.4) by 512 so that the speedup value at $N = 512$ directly compares one DW2 processor against one classical CPU. An overall positive slope suggests a possible limited quantum speedup, subject to the caveats discussed in the text. A negative slope indicates that SA outperforms the DW2.

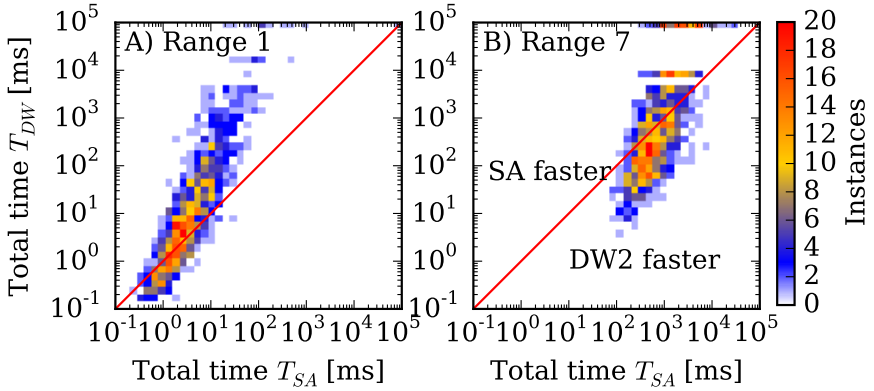


Figure 2.4: **Instance-by-instance comparison of annealing times.** Shown is a scatter plot of the pure annealing time for the DW2 compared to SA using an average over 16 gauges (see 2.5) on the DW2 for A) $r = 1$ and B) $r = 7$. The color scale indicates the number of instances in each square. Instances below the diagonal red line are faster on the DW2, those above are faster using SA. Instances for which the DW2 did not find the solution with 10000 repetitions per gauge are shown at the top of the frame (no such instances were found for SA).

2.5 Supplementary Information

2.5.1 Annealing methods

Quantum annealing

The annealing schedules used in our work are shown in Figure 2.5.

Simulated annealing (SA)

During the annealing schedule we linearly increase the inverse temperature over time from an initial value of $\beta = 0.1$ to a final value of $\beta = 3r$.

For the case of ± 1 couplings ($r = 1$), and for $r = 3$ we use a highly optimized multispin-coded algorithm based on Refs. [96, 97]. This algorithm performs updates on 64 copies in parallel, updating all at once. For the $r = 7$ simulations we use a code optimized for bipartite lattices. Implementations of the simulated annealing codes are available in Ref. [95]. We used the code `an_ms_r1_nf` for $r = 1$, the code `an_ms_r3_nf` for $r = 3$ and the code `an_ss_ge_nf_bp` for $r = 7$.

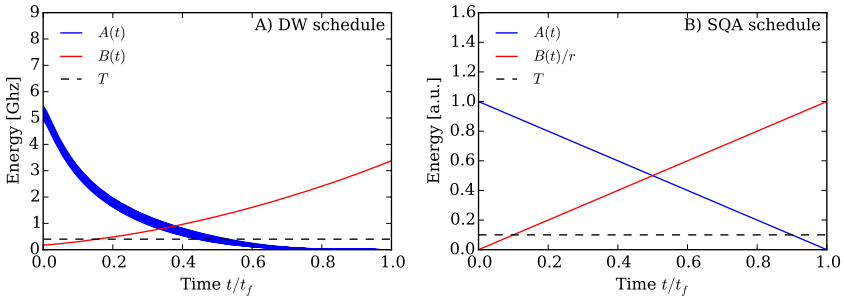


Figure 2.5: **Annealing schedules.** A) The amplitudes of the transverse fields $A_i(t)$ [decreasing, blue] and the longitudinal couplings $B(t)$ (increasing, red) as a function of time. The device temperature of $T = 18$mK is indicated by the black horizontal dashed line. B) The linear annealing schedule used in simulated quantum annealing.

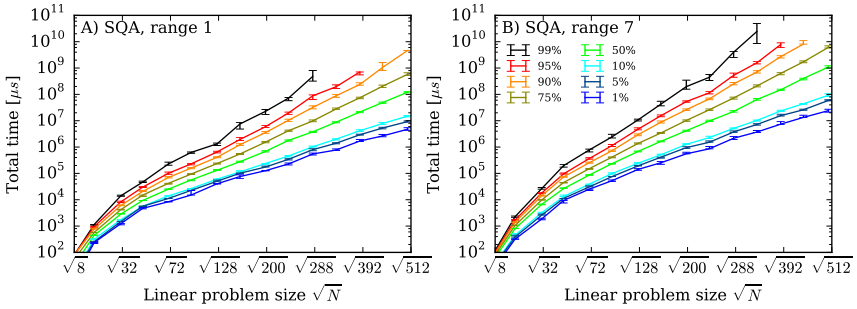


Figure 2.6: **Scaling of time-to-solution for SQA.** Shown is the time-to-solution A) for range $r = 1$ and B) for range $r = 7$.

Simulated quantum annealing (SQA)

The algorithm we used here is similar to that of Ref. [89], but uses cluster updates along the imaginary time direction, typically with 64 time slices. Our annealing schedule is linear, as shown in Figure 2.5B): the Ising couplings are ramped up linearly while the transverse field is ramped down linearly over time. Our SQA results for ranges 1 and 7, complementing Figure 2 in the main text, are shown in Figure 2.6.

All three annealing methods mentioned above are heuristic. They are not guaranteed to find the global optimum in a single annealing run, but only find it with a certain instance-dependent success probability $s \leq 1$. We determine the true ground state energy using an exact belief propagation algorithm [98].

$N \leq 238$	$N = 284, 332$	$N = 385, 439$	$N = 503$
1000	2000	5000	10000

Table 2.1: **Repetitions of annealing runs used on the DW2**. This table summarizes the total number of repetitions used to estimate the success probabilities on the DW2 for various system sizes.

We then perform at least 1000 repetitions of the annealing for each instance, count how often the ground state has been found by comparing to the exact result, and use this to estimate the success probability s for each problem instance. See Table 2.1 for the exact number of repetitions performed on the DW2 device.

2.5.2 The D-Wave Two Vesuvius device

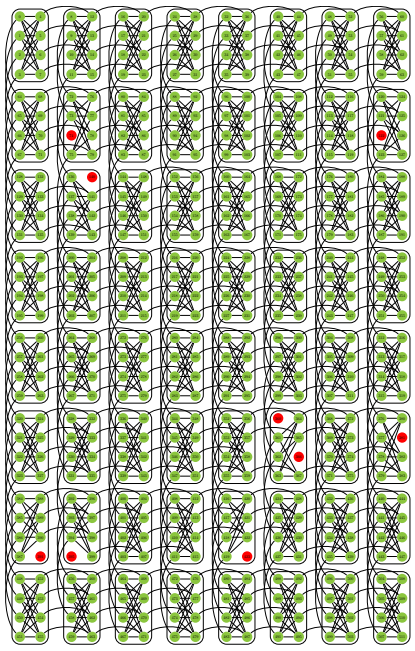


Figure 2.7: **Qubits and couplers in the D-Wave Two device.** The DW2 "Vesuvius" chip consists of an 8×8 two-dimensional square lattice of eight-qubit unit cells, with open boundary conditions. The qubits are each denoted by circles, connected by programmable inductive couplers as shown by the lines between the qubits. Of the 512 qubits of the device located at the University of Southern California used in this work, the 503 qubits marked in green and the couplers connecting them are functional.

The annealing schedules $A_i(t)$ and $B(t)$ used in the device are shown in

figure 2.5A).

Our problem class

The family of problem instances we use for our benchmarking tests employ couplings J_{ij} on all edges of $N = 8LL'$-vertex subgraphs of the Chimera graph of the DW2, comprising $L \times L'$ unit cells, with $L, L' \in \{1, \ldots, 8\}$. We set the fields $h_i = 0$ since nonzero values of the fields h_i destroy the spin glass phase that exists at zero field, thus making the instances easier [99]. We choose the values of the couplings J_{ij} from $2r$ discrete values $\{n/r\}$, with $n \in \{-r, -r-1, \ldots, -1, 1, \ldots, r-1, r\}$, and call r the "range". Thus when the range $r = 1$ we only pick values $J_{ij} = \pm 1$. This choice is the least susceptible to calibration errors of the device, but the large degeneracy of the ground states in these cases makes finding a ground state somewhat easier. At the opposite end we consider $r = 7$, which is the upper limit given the four bits of accuracy of the couplings in the DW2. These problem instances are harder since there are fewer degenerate minima, but they also suffer more from calibration errors in the device. We also used an annealing time of 20μs.

2.5.3 Gauge averaging

Calibration inaccuracies cause the couplings J_{ij} and h_i that are realized in the DW2 to be slightly different from the intended and programmed values ($\sim 5\%$ variation). These calibration errors can sometimes lead to the ground states of the model realized in the device being different from the perfect model. To overcome these problems it is advantageous to perform annealing on the device with multiple encodings of a problem instance into the couplers of the device [13]. To realize these different encodings we use a gauge freedom in realizing the Ising spin glass: for each qubit we can freely define which of the two qubits states corresponds to $\sigma_i = +1$ and $\sigma_i = -1$. More formally this corresponds to a gauge transformation that changes spins $\sigma_i^z \to a_i \sigma_i^z$, with $a_i = \pm 1$ and the couplings as $J_{ij} \to a_i a_j J_{ij}$ and $h_i \to a_i h_i$. The simulations are invariant under such a gauge transformation, but (due to calibration errors which break the gauge symmetry) the results returned by the DW2 are not.

If the success probability of one annealing run is denoted by s, then the probability of failing to find the ground state after R independent repetitions (annealing runs) each having success probability s is $(1-s)^R$, and the total success probability of finding the ground state at least once in R repetitions is

$$P = 1 - (1-s)^R. \tag{2.5}$$

Thus the number of repetitions needed to find the ground state at least once with probability P is found by isolating R in Eq. (2.5).

Following [13], after splitting these repetitions into R/G repetitions for each of G gauge choices with success probabilities s_g, the total success probability

becomes

$$P^{(G)} = 1 - \prod_{g=1}^{G} (1 - s_g)^{R/G}. \tag{2.6}$$

If we use the geometric mean of the failure probabilities of the individual gauges to define

$$\overline{s} = 1 - \prod_{g=1}^{G} (1 - s_g)^{1/G}, \tag{2.7}$$

then Eq. (2.6) can be written in the same form as Eq. (2.5):

$$P^{(G)} = 1 - (1 - \overline{s})^{R}. \tag{2.8}$$

We thus use the geometric mean $\overline{s}$ in our scaling analysis.

2.5.4 Annealing and wall-clock times

A complementary distinction is that between wall-clock time, denoting the full time-to-solution, and the pure annealing time. Wall-clock time is the total time to find a solution and is the relevant quantity when one is interested in the performance of a device for applications and has been used in Ref. [12]. It includes the setup, cooling, annealing and readout times on the DW2, and the setup, annealing and measurement time for the classical annealing codes. The pure annealing time for R repetitions is straightforwardly defined as

$$t_{\text{anneal}} = Rt_a, \tag{2.9}$$

where t_a the time used for a single annealing run. It is the relevant quantity when one is interested in the intrinsic physics of the annealing processes and in scaling to larger problem sizes on future devices.

In order to measure wallclock times on the DW2 we have performed tests with varying numbers of repetitions R and performed a linear regression analysis to fit the total wall clock time for each problem size to the form $t_p(N) + Rt_r(N)$, where $t_p(N)$ is the total preprocessing time and $t_r(N)$ is the total run time per repetition for an N-spin problem. The values of t_p and t_r are summarized in Table 2.2. With these numbers we obtain the total wall clock time for R annealing runs split over G gauges (with R/G annealing runs each) as

$$t_{\text{total}}(N) = Gt_p(N) + Rt_r(N). \tag{2.10}$$

To calculate pure annealing times for the simulated annealer we determine the total effort in units of Monte Carlo updates (attempted spin flips), and then convert to time by dividing by the number of updates that the codes can perform per second [95]. Our classical reference CPU is an 8-core Intel Xeon E5-2670 CPU, which was introduced around the same time as the DW2.

To obtain wall-clock times we measure the actual time needed to perform a simulation on the same Intel Xeon E5-2670 CPU. Since the multi-spin codes

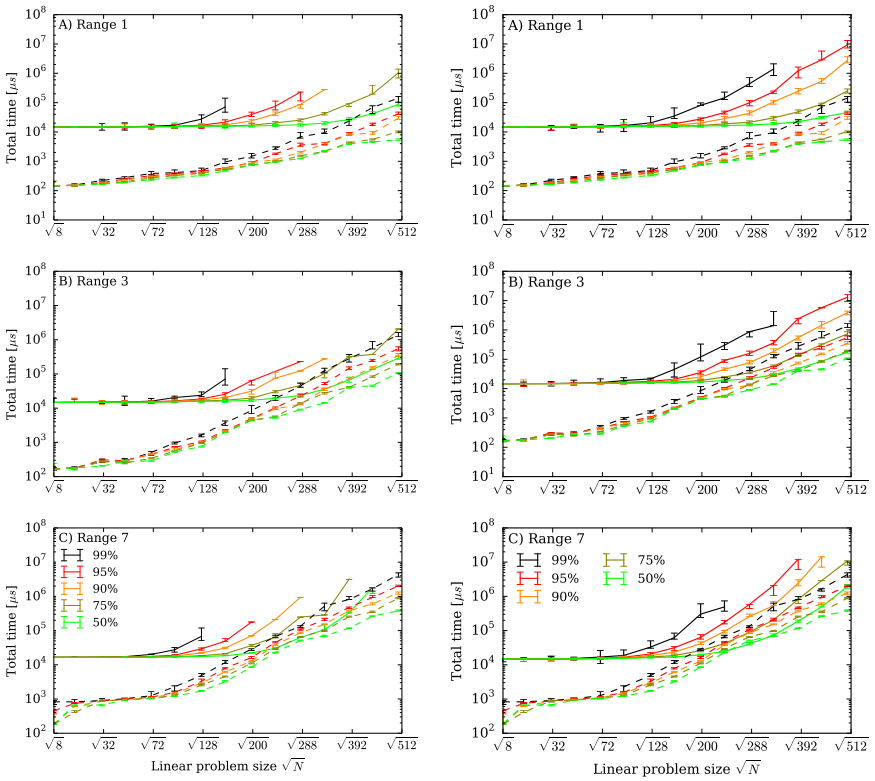


Figure 2.8: **Comparing wall-clock times.** A comparison of the wall-clock time to find the solution with probability $p = 0.99$ for SA running on a single CPU (dashed lines) compared to the DW2 [solid lines] using a single gauge choice in the left column and 16 gauges in the right column. A) for range $r = 1$, B) for range $r = 3$, C) for range $r = 7$. Shown are curves from the median (50th quantile) to the 99th quantile. The large constant programming overhead of the DW2 masks the exponential increase of time-to-solution that is obvious in the plots of pure annealing time.

N	t_p [ms]	t_r [μs]
8	14.7 ± 0.3	51.0 ± 0.2
16	14.8 ± 0.3	53.0 ± 0.2
31	14.8 ± 0.3	57.9 ± 0.2
47	14.9 ± 0.4	60.6 ± 0.2
70	15.0 ± 0.4	64.5 ± 0.2
94	15.2 ± 0.3	68.3 ± 0.2
126	15.6 ± 0.2	73.1 ± 0.2
158	15.5 ± 0.2	78.0 ± 0.2
198	15.5 ± 0.2	80.8 ± 0.2
238	15.7 ± 0.2	83.5 ± 0.2
284	15.8 ± 0.2	83.6 ± 0.1
332	16.0 ± 0.3	87.1 ± 0.2
385	16.6 ± 1.0	87.1 ± 0.6
439	16.6 ± 0.1	90.4 ± 0.1
503	16.6 ± 0.2	90.5 ± 0.1

Table 2.2: **Wallclock times on the DW2**. Listed are measured programming times t_p and annealing plus readout times t_r (for a pure annealing time of 20μs) on the DW2 for various problem sizes.

perform at 64 repetitions in parallel, we always make at least 1024 repetitions when running 16 threads on 8 cores. This causes the initially flatter scaling in wall-clock times as compared to pure annealing times (Figure 2.8). The measured initialization time includes all preparations needed for the algorithm to run, and the spin flip rate was computed for the 99% quantile for 503 qubits. For smaller system sizes or lower quantiles, the spin flip rate is lower since the problems are not hard enough to benefit from parallelization over several cores.

While not as interesting from a complexity theory point of view, it is instructive to also compare wall-clock times for the above benchmarks, as we do in Figure 2.8. We observe that the DW2 performs similarly to SA run on a single classical CPU, for sufficiently large problem sizes and at high range values. Note that the large constant programming overhead of the DW2 masks the exponential increase of time-to-solution that is obvious in the plots of pure annealing time.

Considering the wall-clock times, the advantage of the DW2 seen in Figure 4 A-B (in the main text) for some instances tends to disappear, since it is penalized by the need for programming the device with multiple different gauge choices. Figure 2.9A-F shows that for one gauge choice there are some instances, for $r = 7$, where the DW2 is faster, but many instances where it never finds a solution. Using 16 gauges the DW2 finds the solution in most cases, but is always slower than the classical annealer on a classical CPU for $r = 1$, as can be seen in Figure 2.9A and D. For $r = 7$ the DW2 is sometimes faster than a single classical CPU. Overall, the performance of the DW2 is better for $r = 7$

than for $r = 1$, and comparable to SA only when just the pure annealing time is considered. The difference compared to the results of Ref. [12] is due to the use of optimized classical codes using a full CPU in our comparison, as opposed to the use of generic optimization codes using only a single CPU core in Ref. [12].

2.5.5 Optimal annealing times

As discussed in the main text we need to determine the optimal annealing time $t_a^{\rm opt}$ for every problem size N in order to make meaningful extrapolations of the time to find a solution. To determine $t_a^{\rm opt}$ we perform annealing runs at different annealing times t_a, determine the success probabilities $s(t_a)$ of 1000 instances, and from them the required number of repetitions $R(t_a)$ to find the ground state with a probability of 99%. That is, we solve Eq. (2.8) for R while setting $P^{(G)} = 0.99$:

$$R(t_a) = \left\lceil \frac{\log[1 - P^{(G)}]}{\log[1 - \bar{s}(t_a)]} \right\rceil \ . \tag{2.11}$$

The total effort $R(t_a)t_a$ diverges for $t_a \to 0$ and $t_a \to \infty$ and has a minimum at an optimal annealing time $t_a^{\rm opt}$. The reason is that for short t_a the success probability $\bar{s}(t_a)$ goes to zero, which leads to a diverging total effort, while for large t_a the time also grows since one always needs to perform at least one annealing run and the total effort is thus bounded from below by t_a.

In Figure 2.10 (left) we plot various quantiles of the total effort $R(t_a)t_a$ for the simulated annealer as a function of t_a to determine the optimal annealing time $t_a^{\rm opt}$. For the DW2 we find, as shown in Figure 2.10 (right) that the minimal annealing time of $20\mu s$ is always longer than the optimal time and we thus always use the device in a suboptimal mode. As a consequence the scaling of time-to-solution is underestimated, as explained in detail in the main text.

While one can in principle look for an optimal annealing time for each individual problem instance, our approach is to instead determine an averaged optimal annealing time $t_a^{\rm opt}(N)$ for each problem size N by annealing many instances at various annealing times t_a, and then use these for all future problems of that size.

2.5.6 Resource usage and speedup from parallelism

In the main text we saw that in order to avoid mistaking a parallel speedup for a quantum speedup we need to scale hardware resources (computational gates and memory) in the same way for the devices we compare, and employ these resources optimally. These considerations are not universal but need to be carefully applied for each comparison of a quantum algorithm and device to a classical one. Here we provide several independent derivations that lead us to the same conclusion.

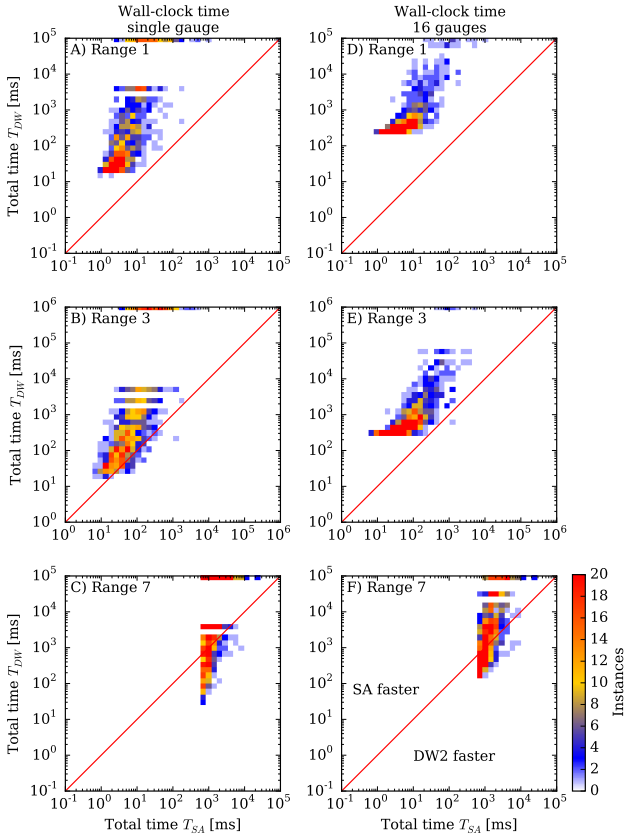


Figure 2.9: **Instance-by-instance comparison.** Shown is a scatter plot of the total time for the DW2 device (DW) compared to a simulated classical annealer (SA) A and D) for $r = 1$, B and E) for $r = 3$, and C and F) for $r = 7$. A, B and C) wall-clock time using a single gauge on the DW2, and, D, E and F) wall-clock time using 16 gauges on the DW2. The color scale indicates the number of instances in each square. Instances below the diagonal red line are faster on the DW2, those above are faster classically. Instances for which the DW2 device did not find the solution are shown at the top. SA found a solution for every instance of this benchmark.

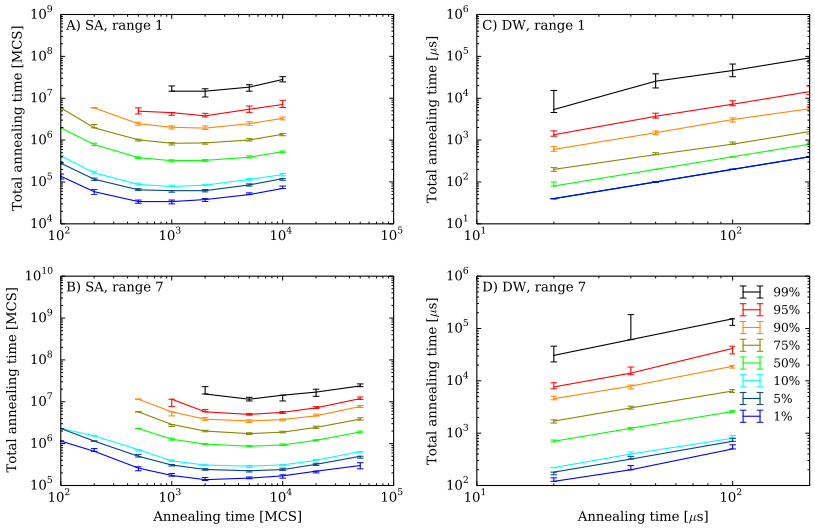


Figure 2.10: **Optimal annealing times for the simulated annealer and for the D-Wave device.** Shown is the total effort $R(t_a)t_a$ as a function of annealing time t_a for various quantiles of problems with $r = 1$ and $r = 7$. A) and B) SA, where the minimum of the total effort determines the optimal annealing time t_a^{opt}. C) and D) DW2, where we find a monotonically increasing total effort, meaning that the optimal time t_a^{opt} is always shorter than the minimal annealing time of 20μs.

Recall that in the main text we argued that the *quantum* part of speedup is estimated by comparing the times required by two devices with the same hardware scaling, giving

$$S(N) = \frac{T_{\mathrm{C}}(N)}{T_{\mathrm{DW}}(N)} \propto \frac{T_{\mathrm{SA}}(N)}{T_{\mathrm{DW}}(N)} \frac{1}{N} \ , \tag{2.12}$$

which is Eq. (3) of the main text. The factor $1/N$ in the speedup calculation discounts for the intrinsic *parallel speedup* of the analog device whose hardware resources scale as N, compared to a fixed size classical device used for the timings.

Derivation assuming fixed computational resources

In the consideration of how to disentangle parallel and quantum speedup it may seem more natural to assume fixed computational resources of a given device. We will show that this leads to the same scaling as Eq. (2.12).

We might be tempted to define the speedup in this case as $S(N) = \frac{T_{\mathrm{SA}}(N)}{T_{\mathrm{DW}}(N)}$. However, in this manner only a fraction $N/512$ of the qubits are used while the classical code uses the available CPU fully, independently of problem size. This

suboptimal use of the DW2 may again be incorrectly interpreted as speedup. The same issue would appear when comparing a classical analog annealer against a classical simulated annealer.

As in the discussion of optimal annealing times above, we need to ensure an optimal implementation to correctly assess speedup. For the DW2 (or a similarly constructed classical analog annealer) this means that one should always attempt to make use of the entire device: we should perform as many annealing runs in parallel as possible. Let us denote the machine size by M (e.g., $M = 512$ in the DW2 case). With this we define a new, optimized, annealing time

$$T_{\mathrm{DW}}^{\mathrm{opt}}(N) = T_{\mathrm{DW}}(N)\frac{1}{\lfloor M/N \rfloor}, \tag{2.13}$$

and the speedup in our case is then

$$S(N) = \frac{T_{\mathrm{SA}}(N)}{T_{\mathrm{DW}}^{\mathrm{opt}}(N)} = \frac{T_{\mathrm{SA}}(N)}{T_{\mathrm{DW}}(N)}\left\lfloor \frac{M}{N} \right\rfloor. \tag{2.14}$$

Omitting the floor function ($\lfloor\ \rfloor$), which only gives subdominant corrections in the limit $M \to \infty$ we recover Eq. (2.12).

Derivation from the scaling of the annealing time.

The conclusion that the speedup function includes a factor proportional to $1/N$ is validated from yet another perspective, that focuses on the annealing time. Instead of embedding $C \equiv \lfloor M/N \rfloor$ different instances in parallel, we can embed C replicas of a given instance. Each replica r (where $r \in \{1, \ldots, C\}$) results in a guess $E_{r,i}$ of the ground state energy for the ith run, and we can take $E_i = \min_r E_{r,i}$ as the proposed solution for that run. If the replicas are independent and each has equal probability s of finding the ground state, then using C replicas the probability that at least one will find the ground state is $s' = 1 - (1-s)^C$, which is also the probability that E_i is the ground state energy for the ith run. Repeating the argument leading to Eq. (2.11), the number of repetitions required to find the ground state at least once with probability p is then:

$$R' = \left\lceil \frac{\log(1-p)}{\log(1-s')} \right\rceil = \left\lceil \frac{\log(1-p)}{C\log(1-s)} \right\rceil, \tag{2.15}$$

while $R = \left\lceil \frac{\log(1-p)}{\log(1-s)} \right\rceil$. It is easy to show that $R' - 1 \leq R/C \leq R'$. Focusing on the pure annealing time we have $T_{\mathrm{DW}}^{\mathrm{opt}}(N) = t_a R'$ and $T_{\mathrm{DW}}(N) = t_a R$, which yields Eq. (2.13) in the limit of large R.

2.5.7 Arguments for and against a speedup on the DW2

Let us consider in more detail the speedup results discussed in the main text and above. We have argued that the apparent limited quantum speedup seen in the

$r = 1$ results of Figure 3 in the main text must be treated with care due to the suboptimal annealing time. It might then be tempting to argue that, strictly speaking, the comparison with suboptimal-time instances cannot be used for claiming a slowdown either, i.e., that we simply cannot infer how the DW2 will behave for optimal-time instances by basing the analysis on suboptimal times only.

However, let us make the assumption that, along with the total time, the optimal annealing time $t_a^{\rm opt}(N)$ also grows with problem size N. This assumption is supported by the SA and SQA data shown in Figure 1 in the main text and Figure 2.14, and is plausible as long as the growing annealing time does not become counterproductive due to coupling to the thermal bath [100]. By definition, $T_{\rm DW}(N, t_a^{\rm opt}(N)) \leq T_{\rm DW}(N, t_a)$, where we have added the explicit dependence on the annealing time, and t_a is a fixed annealing time. Thus

$$\begin{aligned} S(N) &= \frac{T_{\rm C}(N)}{T_{\rm DW}(N, t_a)} \frac{1}{N} \\ &\leq \frac{T_{\rm C}(N)}{T_{\rm DW}(N, t_a^{\rm opt}(N))} \frac{1}{N} = S^{\rm opt}(N). \end{aligned} \tag{2.16}$$

Under our assumption, $t_a^{\rm opt}(N) < t_a$ for small N, but for sufficiently large N the optimal annealing time grows so that $t_a^{\rm opt}(N) \geq t_a$. Thus there must be a problem size N^* at which $t_a^{\rm opt}(N^*) = t_a$, and hence at this special problem size we also have $S(N^*) = S^{\rm opt}(N^*)$. However, the minimal annealing time of $20\mu s$ is longer than the optimal time for all problem sizes (see Figure 2.10), i.e., $N^* > 503$ in our case. Therefore, if $S(N)$ is a decreasing function of N for sufficiently large N, as we indeed observe in all our RofQ results (recall Figure 3A and B in the main text), then since $S^{\rm opt}(N) \geq S(N)$ and $S(N^*) = S^{\rm opt}(N^*)$, it follows that $S^{\rm opt}(N)$ too must be a decreasing function for a range of N values, at least until N^*. This shows that the slowdown conclusion holds also for the case of optimal annealing times.

For the instance-by-instance comparison (QofR), no such conclusion can be drawn for the subset of instances (at $r = 1$) corresponding to the high quantiles where $S_q^{\rm QofR}(N)$ is an increasing function of N. This limited quantum speedup may or may not persist for larger problem sizes or if optimal annealing times are used.

2.5.8 Additional Discussion

In this work we have discussed challenges in properly defining and assessing quantum speedup, and used comparisons between a DW2 and simulated classical and quantum annealing to illustrate these challenges. *Strong* or *provable* quantum speedup, implying speedup of a quantum algorithm or device over *any* classical algorithm, is an elusive goal in most cases and one thus usually defines *quantum speedup* as a speedup compared to the best available classical algorithm. We have introduced the notion of *limited quantum speedup*, referring

to a more restricted comparison to 'corresponding' classical algorithms solving the same task, such as a quantum annealer compared to a classical annealing algorithm.

Quantum speedup is most easily defined and detected in the case of an exponential speedup, where the details of the quantum or classical hardware do not matter since they only contribute subdominant polynomial factors. In the case of an unknown or a polynomial quantum speedup one must be careful to fairly compare the classical and quantum devices, and, in particular, to scale hardware resources in the same manner. Otherwise *parallel speedup* might be mistaken for (or hide) quantum speedup.

An experimental determination of quantum speedup suffers from the problem that all measurements are limited to finite problem sizes N, while we are most interested in the asymptotic behavior for large N. To arrive at a reliable extrapolation it is advantageous to focus the scaling analysis on the part of the execution time that becomes dominant for large problem sizes N, which in our example is the pure annealing time, and not the total wall-clock time. For each problem size we furthermore need to ensure that neither the quantum device nor the classical algorithm are run suboptimally, since this might hide or fake quantum speedup.

If the time to solution depends not only on the problem size N but also on the specific problem instance, then one needs to carefully choose the relevant quantity to benchmark. We argued that in order to judge the performance over many possible inputs of a randomized benchmark test, one needs to study the high quantiles, and define speedup by considering the ratio of the quantiles of time to solution. If, on the other hand, one is interested in finding out whether there is a speedup for some subset of problem instances, then one can instead perform an instance-by-instance comparison by focusing on the quantiles of the ratio of time to solution.

We chose to focus here on the benchmark problem of random zero-field Ising problems parametrized by the range of couplings. We did not find evidence of limited quantum speedup for the DW2 relative to simulated annealing in our particular benchmark set when we considered the ratio of quantiles of time to solution, which is the relevant quantity for the performance of a device as an optimizer. We note that random spin glass problems, while an interesting and important physics problem, may not be the most relevant benchmark for practical applications, for which other benchmarks may have to be studied.

When we focus on subsets of problem instances in an instance-by-instance comparison, we observe a possibility for a limited quantum speedup for a fraction of the instances. (The fact that some problems are solved faster on classical hardware and some on the DW2 raises the possibility of a hybrid approach that benefits by solving a problem instance using both, and then selecting the best solution found.) However, since the DW2 runs at a suboptimal annealing time for most of the corresponding problem instances, the observed speedup may be an artifact of attempting to solve the smaller problem sizes using an excessively long annealing time. This difficulty can only be overcome by fixing the issue of suboptimal annealing times, e.g., by finding problem classes for which the

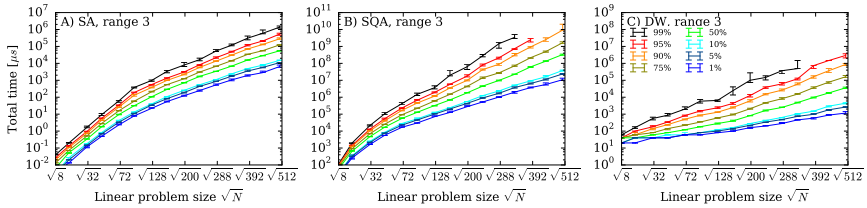
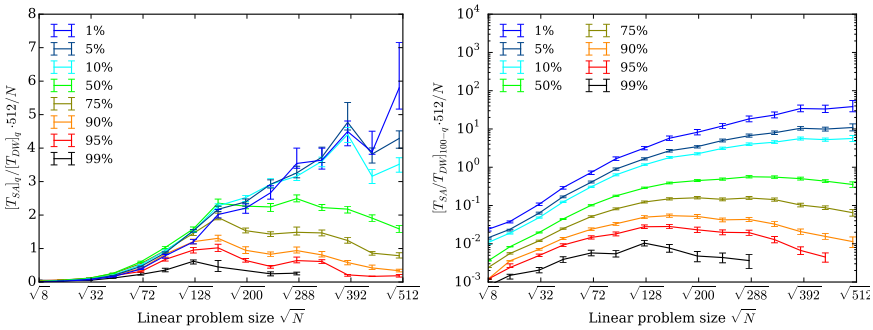


Figure 2.11: **Scaling of time-to-solution for $r = 3$.** Shown is the scaling for the time to find the ground state with a probability of 99% for various quantiles of hardness for A) the simulated annealer, B) the simulated quantum annealer, and C) the DW2 device.



(a) (b)

Figure 2.12: **Speedup for the DW2 device compared to SA for instances with $r = 3$.** 16 gauges were used. Left: the ratio of quantiles (RofQ). Right: quantiles of the ratio (QofR). The results are intermediate between the $r = 1$ and $r = 7$ results as discussed in the text.

annealing time is demonstrably already optimal. Alternatively, other problem classes might exhibit a speedup. For example, it is possible that a randomized benchmark with a tunable "hardness knob", such as the clause density in the MAX-2SAT problem [14], will allow for a more fine-tuned exploration of the performance potential of a quantum annealer than a purely randomized benchmark such as we have used here. It was also proposed that embedding 3D spin-glass problems into the Chimera graph, with a finite critical temperature, might yield problem classes that are better suited as benchmarks for quantum annealing [93].

2.5.9 Additional scaling data: range 3

Scaling plots for range 3 instances, requiring 3 bits of precision in the couplings, are shown in Figure 2.11, complementing Figure 2 in the main text. Figure 2.12

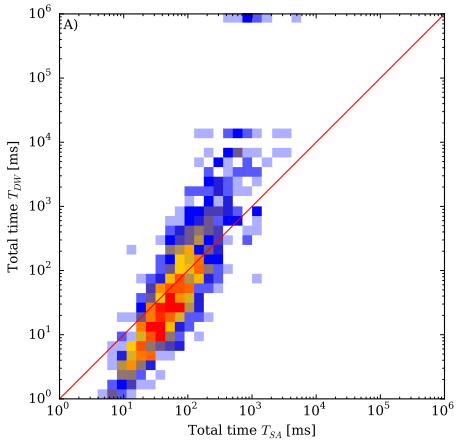


Figure 2.13: **Instance-by-instance comparison for $r = 3$.** Comparison between time-to-solution for SA and DW2 using pure annealing times and using 16 gauges.

(left) displays the ratio of quantiles and, like Figure 3 in the main text, does not exhibit a limited quantum speedup. Figure 2.12 (right) displays the results for the quantiles of ratio of time-to-solution. The results are intermediate between those seen in Figure 3 in the main text for $r = 1, 7$, namely, while for $r = 1$ there appears to be a limited quantum speedup (relative to SA) for the higher quantiles, this speedup disappears for $r = 7$; for $r = 3$ we observe a flattening of the speedup curves starting at the 10th percentile, while the higher percentiles bend down. The same suboptimality remarks discussed in this context in the main text apply. We have obtained similar results (not shown) for ranges $r = 2, 4, 5, 6$ and for instances including random longitudinal fields.

Figure 2.13 shows the instance-by-instance comparisons for the pure annealing time. The conclusions are very similar to those obtained based on Figure 4 in the main text using the $r = 1, 7$ data.

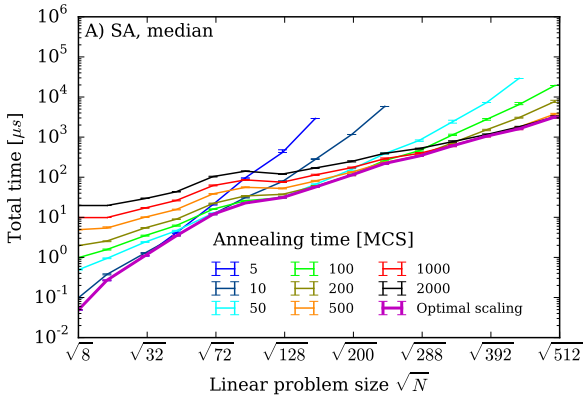


Figure 2.14: **Optimal annealing times for SA.** Shown is the time-to-solution for various quantiles as a function of annealing time (Monte Carlo steps) for SA range $r = 1$. This supplements Figure 1 in the main text.

Chapter 3

Benchmarking or: How I learned to stop worrying and love Bayesian nonparametrics and optional stopping

Benchmarking is a difficult task, as partly discusse at the end of chapter 1, and if one does it naively one can easily fool oneself into believing false things with high confidence. In the hopes of aligning data analysis and data collection procedures across our community, this chapter presents a basic overview of the effects of gauges, the modeling of benchmarking time to solution (TTS) for algorithms for Ising-type problems, Bayesian nonparametric modeling, and applies the insights learned to the benchmarking problem. The final procedure for data analysis is quite similar to previous methods, while the data collection procedure is quite different.

In general, the conclusion for quantum annealers is that one should collect ≈100 anneals for each programming cycles/gauge, unless there is a specific reason to change it, and continue running until the coefficient of variation of one's posterior distribution for probability of success or TTS falls below a threshold of 0.2 (or 0.1). Estimates of the posterior should be performed by drawing samples from the distribution defined by first sampling a value from each gauge's posterior distribution for p_s, Beta$(x, 100 - x)$ for x successes in that gauge, and then performing a weighted average with weights drawn from the Dir(1,1,1,...,1) distribution (a Bayesian bootstrap).

The development of the D-Wave line of quantum annealers has sparked a flurry of activity in the optimization and quantum computation community, and spurred a small cottage industry around benchmarking their performance. This

chapter is not concerned with the results of any such analysis, but with the method of analysis itself. The goal is to lay out guidelines which will hopefully aid all future researchers in this field benchmark future generations of D-Wave devices, as well as competing optimization algorithms, in a consistent and standard way.

This chapter is organized into three parts. The first addresses the problem of data analysis alone, introducing the benchmarking problem formally again and discussing the ideal solution as well as realistic solutions forced upon us by practical necessities. It includes a brief introduction to the field of Bayesian nonparametrics and its bread and butter, the Dirichlet distribution and the associated Dirichlet process, and grounds the analysis procedure in this field, while exploiting a technique which is simpler computationally called the Bayesian bootstrap. It concludes with a comparison between the Bayesian and classical bootstraps, arguing that the classical bootstrap is conceptually flawed and computes a distribution we, as scientists, don't care about (the sampling distribution of a statistic) while the Bayesian bootstrap gives us the distribution we typically think the classical bootstrap gives us — namely a posterior distribution on the parameter or function of interest.

The second part addresses the question of data collection procedures, and outlines an approach based on optional stopping, where one uses the data itself to determine when one has enough data, on the fly, instead of specifying sample sizes in advance. While this has the downside of resulting in an estimator with a small bias, this bias is controllable (it decreases linearly with the number of gauges sampled), small (a few percent at most), and acts effectively as an overall constant factor on the TTS rather than as an element with scaling. It thus has no bearing on scaling results, which involve fitting to an exponential. Moreover, the resulting posterior credible intervals have very good frequentist coverage properties. The chief benefit is a potentially enormous time savings when one seeks to learn the TTS for most instances in a collection.

The third section presents the conclusions of a simulation study using various artificial distributions meant to highlight various possible cases and to validate the collection procedure and recommendations given in the rest of the paper.

A few notes on style before we begin: First, this paper is written in a tone more similar to a presentation than a traditional paper. Second, I use "I" throughout to refer to the author, and "we" to denote "the reader and I". It is also rather opinionated. As this chapter has not appeared in any form in a publication, and as it was wholly original and written solely by the author of this dissertation, it will likely be the most idiosyncratic chapter.

3.1 Please lie down on the couch: Data analysis

Suppose that you have a problem, H whose properties depend on some set of parameters which you control, call them λ, as well as a set of nuisance parameters θ you do not control and/or do not care about. A solver takes in some problem $H(\lambda, \theta)$ and outputs a trial solution $\vec{s}$ under some set of solver param-

eters χ. We have some function Φ which maps a solution $\vec{s}$ into a binary output $s = 0, 1$ for "failure" and "success" respectively. Note that this language may be misleading — any binary classifier on the samples $\vec{s}$ qualifies. Applying the map to all solutions sampled from the solver, and we can represent a solver as a simple generator of a sequence of 0s and 1s.

The problem of interest is to learn the probability under some defined distribution over the parameters θ that a given solution will be classified as a "success" or 1. In other words, we seek to estimate $p_s(\lambda, \chi) = \langle Pr(s = 1|H(\lambda, \theta), \chi)\rangle_\theta$, and thus also the time to solution (related to p_s through a nonlinear transformation $log(0.01)/log(1 - p_s)$, as defined in previous chapters).

Here, λ could be the list of all couplers specifying the Hamiltonian, but it could also include meta-parameters such as the range of random Ising problems, the loop density in the frustrated loop planted solutions, chain strength in an embedded problem, etc. Any property of an instance which, contextually, we wish to account for and track explicitly. The θ constitute variables to be averaged over, like the physical couplers of the programmed Hamiltonian (which are inaccessible and taken to be stochastic).

Certain properties may lie on either side of this boundary depending on context. If one is studying the correlation between particular embeddings of full connected problems into Chimera, then embeddings would be in λ. If one instead simply views different embeddings as different but effectively equal representations of the same underlying problem, then the embedding would be in θ. Variables like annealing time, number of sweeps, initial and final temperature, number of replicas in parallel tempering, number of Trotter slices in PIMC, etc. constitute χ.

The task of benchmarking perfomance on a particular instance is, to repeat in this language, to estimate $p_s(\lambda, \chi)$, or, for fixed χ, simply $p_s(\lambda)$.

3.1.1 Paradiso: the ideal solution

If one resamples θ for each trial according to its intrinsic distribution, then the expected probability of success at any trial is exactly p_s (this is, indeed, what it means to "marginalize" over a variable, i.e. to average over it). By resampling θ, we effectively render the success or failure of each new solution a simple Bernoulli trial. We then seek to estimate the probability of success of this Bernoulli trial. To do so, the default procedure is to choose a conjugate prior to the Bernoulli distribution, namely a Beta distribution, and update via Bayes rule on the evidence from repeated trials. Conjugate priors have the helpful property that the posterior distribution will be of the same family as the prior after updating on new evidence [101]. For the Beta distribution with a prior $Beta(c_1, c_2)$ and a sequence of N trials with x successes, the posterior is simply $Beta(x + c_1, N - x + c_2)$. c_1 can thus be considered the prior's number of pseudosuccesses and c_2 the prior's number of pseudofailures.

This can be immediately seen, as the probability density p of $Beta(\alpha, \beta)$ is $p \propto p^{\alpha-1}(1-p)^{\beta-1}$, and the likelihood of observing x successes from a Binomial random variable in N trials is simply $\propto p^x(1 - p)^{N-x}$.

To review for those unfamiliar with Bayesian terminology, a **prior** is one's initial information or state of belief about a value, and a **posterior** is one's information after adjusting one's views according to the rules of probability after acquiring new information.

The choice of prior has negligible impact to the posterior distribution when $c_1 \ll x$ and $c_2 \ll N - x$. Generally, $c_1 = c_2 = c$, and there are three common choices each with arguments in its favor – $c = 0$, $\frac{1}{2}$, or 1. $c = 1$ is the uniform distribution on $[0, 1]$. $c = \frac{1}{2}$ is the arcsine distribution, and is the Jeffrey's prior for the Bernoulli/Binomial distribution, namely the prior which is the square root of the determinant of the Fisher information matrix (and is thus invariant under reparameterization of the distribution). Finally, $c = 0$ is the Haldane prior, which is a half-delta function at 0 and another at 1. The Haldane distribution is the only one for whom the mean of the posterior is equal to x/N, in general it is equal to $\frac{x+c_1}{N+c_1+c_2}$ for a $Beta(c_1, c_2)$ prior. The standard deviation is $\left(\frac{(x+c_1)(N-x+c_2)}{(N+c_1+c_2)^2(N+c_1+c_2)}\right)^{\frac{1}{2}}$. The coefficient of variation, $\sigma/\mu = \sqrt{\frac{N-x+c_2}{(x+c_1)(N+c_1+c_2)}}$. If c_1 and c_2 are small relative to $N - x$ and N, this is approximately $\frac{1}{x} - \frac{1}{N}$, and for large N is just $\frac{1}{x}$.

We immediately see an important lesson: the coefficient of variation of p_s is controlled by the number of successes that we see. Thus, if we wish to estimate to within a fixed variation, we require the number of successes to be approximately constant.

3.1.2 The realistic solution

While a method of operation like the above is available to us for some algorithms (those on classical hardware for instance), to resample θ at each trial on D-Wave hardware is untenable due to the long programming time. If run in such a mode, the D-Wave chip will be annealing only $\approx 1\%$ of the time. In a given time T, one can perform $G = T/(\tau_{prog} + \tau_{anneal}R)$ gauges where $\tau_{prog} \approx 15ms$ is the programming time per gauge, and $\tau_{anneal} \approx 150ms$ is the total time per anneal (including readout). For $R = 1$ you can perform approximately 66 gauges per second with 66 samples. For $R = 100$ you can perform approximately 33 gauges per second with 3300 samples, and for $R = 200$ you can perform approximately 22 gauges with 4400 samples. One sees there is a shift, from a moderate decrease in the number of gauges causing enormous increases in the number of samples (and thus better probability estimates) to moderate tradeoffs in the number of gauges corresponding to moderate increases in the number of samples. For $R > 1000$ this tradeoff is quite catastrophically large, moving from $R = 1000$ to $R = 50000$ increases the number of samples per second by less than 10% while decreasing the number of gauges per unit time by roughly a factor of 50. Since we have no interest in the individual probability per gauge except what it tells us about the mean over θ, we should expect that restricting ourselves to $R \approx 100$ will yield the most efficient inferences. Later simulation studies will show that quite generally $R = 100$, i.e. that R such that the quantum annealer spends approximately half of it's time programming and half annealing, is indeed the

optimal number of runs per gauge for our probability estimates to converge with the minimum wall-clock time, i.e. the minimum effort. Thus, out of practical necessity we must perform multiple trials from each value of θ and then take an average to estimate p_s.

For each set of trials (equivalent to a programming cycle on the D-Wave processor; I will call them "progcycles" or "gauges" interchangeably) we can use a Beta distribution to encode our knowledge about the probability of success for that progcycle, and then compute our belief about the average value over all progcycles. Now, the choice of prior is no longer practically irrelevant (unlike the ideal case). Since we know we cannot run an unlimited number of anneals with a single progcycle we will inevitably reach a point where the overwhelming majority of our progcycles have zero successes. If our prior on $p_{success}$ for a gauge has any bias in the average value, then we'll have a bias of the same magnitude in our estimate for the average, p_s. Thus, for very difficult problems, with low success probabilities, our prior will swamp our data and we will learn nothing no matter how many anneals we perform. Thus, we are forced to choose the Haldane prior, $Beta(0, 0)$, for the prior on the probability of success of each gauge, as it is the only Beta distribution prior whose mean is equal to the empirical success rate and thus is not biased by prior information.

3.1.3 I was blind but now I see: Bayesian nonparametrics

We can construct a Bayesian model for our problem: $Y_i \sim Bin(R, p_i)$ are the number of successes for each progcycle i given some probability of success $p_i \sim F$ for some unknown distribution F. To complete a full Bayesian model specification, we must specify a prior on F. This may at first seem quite difficult, even impossible, as F can in principle be any probability distribution defined on the interval $[0, 1]$. Moreover, if the prior isn't conjugate it may be infeasible to compute posterior densities. There is, however, a density which is conjugate to iid sampling, and it is called the Dirichlet process.

The Dirichlet Distribution

Before discussing the Dirichlet process (DP) in more detail, allow me first to quickly introduce a related distribution: the Dirichlet distribution. Denoted by $\text{Dir}(\vec{\alpha})$ for an arbitrary vector of positive real parameters $\vec{\alpha}$, the Dirichlet distribution is conjugate to the categorical and multinomial distributions.

Suppose you wish to estimate the probability of rolling each side of a six-sided die. Just as for the Beta distribution there are a set of common choices of prior Dirichlet distribution priors. The one whose mean corresponds to the maximum likelihood estimate (MLE) is the one with all 0s in $\vec{\alpha}$ [84]. The posterior with this prior, assuming one saw x_i instances of the number i after a total of $\sum_i x_i = N$ rolls of the die will be $\text{Dir}(x_1, x_2, x_3, x_4, x_5, x_6)$. The marginal density for $p(i|\vec{x})$ =Beta$(x_i, N - x_i)$, peaked at the empirical probability of success. The resulting posterior density will be a smooth version of the posterior

derived from the classical bootstrap (which is discussed later, but is simply resampling values from your empirical distribution [102]).

An important property of the Dirichlet distribution is that one can repartition the space of outcomes and the result will still be Dirichlet. For instance, the distribution of "(1 OR 2), 3, 4, 5, 6" would be given by $\mathrm{Dir}(x_1+x_2, x_3, x_4, x_5, x_6)$ (while the corresponding conditional distribution just on 1 and 2 would be $\mathrm{Dir}(x_1, x_2)$). This ability to regroup elements in the space of outcomes at will and easily compute marginal distributions is one of the reasons why the Dirichlet distribution and it's extension, the Dirichlet process, are so useful in Bayesian statistics.

The Dirichlet process was originally proposed in [103]. Since, there have been a number of characterizations of it, but for our purposes here we need only deal with the first. A Dirichlet process $\mathrm{DP}(\alpha F_0)$ is a probability distribution on the support of the "base distribution" F_0 that is almost surely discrete. It has the property that for any partition of the support of F_0, call it $(X_1, X_2, ..., X_N)$, one finds that realizations F of $DP(\alpha F_0)$ obey the following property:

$$(F(X_1), F(X_2), ..., F(X_N)) \sim Dir(\alpha F_0(X_1), ..., \alpha F_0(X_N))$$

, where $A(X)$ denotes the total support of whatever distribution A on the interval X. This rule applies for every partition $\{X\}$ simultaneously. While at first it is not obvious such a process is possible, it is and it is actually fairly trivial to sample realizations to arbitrary accuracy from a Dirichlet Process.

The Dirichlet process is conjugate to iid sampling. If one samples a set of N values from F $\{\theta_i\}_{i=1}^N$, one's posterior distribution for F is $\mathrm{DP}(\alpha * F_0 + \sum_i \delta_{\theta_i})$, where δ_θ denotes an atom with weight 1 on the point θ. Thanks to the above partitioning property, one can rewrite this (at least to arbitrary accuracy) as

$$\mathrm{DP}\big(\alpha * F_0 + \sum_i \delta_{\theta_i}\big) = p * \mathrm{DP}(\alpha F_0) + (1-p)\mathrm{DP}\big(\sum_i \delta_{\theta_i}\big)$$

$$p \sim \mathrm{Beta}(\alpha, N)$$

by separating the points $\{\theta_i\}$ and arbitrarily small regions around each point into one partition and the rest of the interval (which is effectively the entire interval, minus N holes) into another. The distribution $\mathrm{DP}(\sum_i \delta_{\theta_i})$ is quite interesting, in that it effectively only has support on a finite set of points. Indeed, it is identical to a $\mathrm{Dir}(\vec{1}^N)$ (N 1's) distribution on the collection of points θ_i. We will return to this distribution shortly. (Note, to my knowledge this is actually an original representation of the posterior of a DP.)

What role do α and F_0 have? α acts as a strength parameter, essentially representing how confident we are that the true distribution is in the "vicinity" of F_0. If α is very large, it will take quite a lot of data for the data to overwhelm the prior (as seen from the weights between the components of the posterior, namely $Beta(\alpha, N)$). If α is very small, the prior has negligible influence on the posterior estimates even at small sizes. In the limit $\alpha \to 0$, the distribution becomes purely the above $\mathrm{Dir}(\vec{1}^N)$ on the points $\{\theta_i\}$. This distribution has been

given another name, the Bayesian bootstrap, introduced in Ref [84], and will henceforth be denoted as $BB(\vec{\theta})$ The Bayesian bootstrap has no free parameters, and is wholly determined by the data vector $\vec{\theta}$. One can see that the Bayesian bootstrap also dominates the Dirichlet Process in the limit of $N \to \infty$.

With all these preliminaries in mind, we can complete our Bayesian model specification as

$$\begin{aligned} Y_i &\sim \text{Bin}(R, p_i) \\ p_i &\sim F \\ F &\sim \text{DP}(\alpha F_0) \end{aligned}$$

for some values of α and F_0 denoting our prior beliefs (which may, in principle, vary from person to person). This problem was actually investigated in [104], where they computed an approximation to the true posterior distribution. [105] demonstrated an importance sampling technique to attempt to approximate the posterior for conjugate priors F_0. In the last two decades, additional MCMC techniques have been developed for inference in the general case for Dirichlet processes [106].

These have several disadvantages, however. First and foremost, these MCMC approaches are generally quite slow to converge, and indeed can take minutes, hours, and even days when one has tens of thousands of data points (as would be the case for benchmarking fairly hard problems, with success probabilities in the 10^{-5} range). This is simply impracticle for our datasets. A number of approximation techniques have been proposed, such as [107].

Approximations don't overcome the other challenge, however, which is eliciting sensible prior values for α and F_0. Often researchers place hyperpriors on these, and learn their values, too, from the data, but this only makes the convergence problem of the MCMC algorithms worse. Moreover, our community is not composed solely of Bayesians, and soliciting good priors is a notoriously difficult problem in any case. Rather than attempt to do so, it is my suggestion that we instead adopt the Bayesian bootstrap directly as our inference mechanism.

3.1.4 Why the Bayesian bootstrap?

This is justified in several ways. First and foremost, the Bayesian bootstrap is the only way the data enters into the posterior of the Dirichlet process for a given sample from F $\vec{\theta}$. Since our prior will obviously be weak (as we don't have any good idea of what F should look like *a priori*), and we will want to take a large amount of data in any case, the posterior will likely be very near the Bayesian bootstrap anyway. $\alpha \to 0$ serves as a fairly natural noninformative prior for Dirichlet processes, acting in a manner similar to the Haldane and $\text{Dir}(\vec{0})$ priors for Binomial and Multinomial distributions.

Finally, this prior is extremely tractable and flexible. Since we know that each element of $\vec{\theta}$ must be distinct (as they are simply the true probabilities of success for each gauge we ran and the underlying distribution is almost surely

continuous and nonatomic), we can treat each θ_i independently, deriving a posterior distribution for each and then performing a Monte Carlo average. The posterior for θ_i is $\text{Beta}(x_i, R - x_i)$ given a Haldane prior, which also corresponds to taking the likelihood function $L(x_i|\theta_i)$ and interpreting it as a distribution on θ_i. We would then place a $\text{Dir}(\vec{1})$ distribution on the weight of each sample in our posterior estimation. That is:

$$\begin{aligned}
\theta_i &\sim \text{Beta}(x_i, R_i) \\
w_i &\sim \text{Dir}(1, 1, 1, ..., 1) \\
p_s &= \sum_i w_i \theta_i
\end{aligned}$$

MC simulations of this type can be performed in seconds even with tens of thousands of data points, making them suitable for use in online learning of our parameters and their credible intervals. Indeed, due to the decomposition property of the Dirichlet distribution, one can even update an existing ensemble of points sampled from one's posterior to update to a new sample from one's posterior on the acquisition of new information, leading to effective sample times of well under a second for thousands of realizations.

Let us imagine that for some problem, $x = R$ or $x = 0$ for all of N gauges run. If $x = R$, one's posterior for θ_i is a delta function at 1, while for $x = 0$ it is a delta at 0. As a result, one can regroup elements in the Dirichlet distribution into "deltas at 0" and "deltas at 1", yielding $\text{Dir}(N - X, X)$, where X being the number of gauges with all 1s. The sampled values from p_s will then be $(1 - p)$ for $p \sim \text{Beta}(N - X, X)$, i.e. $\text{Beta}(X, N - X)$. Notice that if one sets $R = 1$, one is actually performing the ideal experiment and the analysis procedure yields a posterior that is exactly the posterior for the ideal case (with a Haldane prior). This is an extremely attractive feature, and another source of justification for using the Bayesian bootstrap.

How does the Bayesian bootstrap compare to the classical bootstrap?

3.1.5 To Bayes, or not to Bayes, that is the question: Bayesian v. classical bootstrap

While the Bayesian bootstrap has a natural connection to Bayesian non-parametrics, it isn't obvious what the payoff is over the classical bootstrap unless one is a committed Bayesian and frequentism makes one queasy. Why bother when we already have the classical bootstrap if they're so similar? To explain the differences, let's review the classical bootstrap and compare the assumptions each bootstrap makes.

In a classical bootstrap [102], one makes the following assumptions:

1. All values that may be sampled from the distribution have been. (One has seen every type in the population.)

2. All frequencies of the types in the population are exactly those in the empirical sample.

One then estimates the distribution for a parameter of interest under these assumptions. One does so by creating new pseudodata vectors by sampling with replacement N values from the original vector (of length N).

A key point is that the classical bootstrap mimics the *sampling* distribution of a parameter, not the distribution of the parameter in the population. In the case of benchmarking this distinction proves fatal for the classical bootstrap. Take, for instance, the case where one has 10 gauges, 9 of whom have 0 successes and 1 of whom has all successes, after R trials in each gauge. We know the Bayesian bootstrap will yield $\text{Beta}(1, 9)$ as discussed above. For the classical bootstrap, we would be sampling from $\frac{1}{n}\text{Bin}(10, 0.1)$. A plot of this is provided in 3.1.

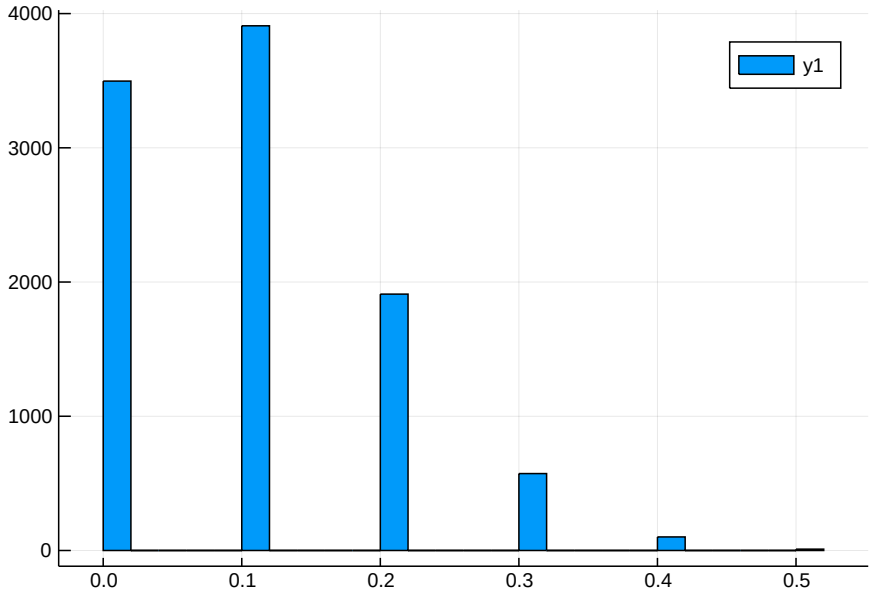


Figure 3.1: Results from 10000 frequentist bootstrap samples from a sample of 10 gauges, nine with zero successes and one with all successes. If one tried to interpret this as a distribution of one's uncertatiny about the mean success probability, averaged over all gauges, then one has significant weight on the mean success rate is 0, even though one actually observed successes, a clear contradiction.

As you can see, you have non-negligible support on $p_s = 0$. But of course, this isn't p_s, the gauge-marginalized mean success probability, this is our bootstrapped estimate of the sampling distribution for our statistic. The practitioner of the classical bootstrap is left with few options when attempting to move to

sampling from the distribution for time to solution, which goes like $1/p_s$ and thus has weight on TTS$= \infty$. Any time there is a gauge with all zeros, the classical bootstrap will place support on infinite time to solution, whether we happen to have sampled those elements from the distribution or not. Thus, while we may be able to "get away with it" in the sense that using the classical bootstrap won't always fail in practice due to the presence of a single 0, the fact remains that we know analytically the true mean of the bootstrap estimate for TTS is ∞.

This gets to a deeper issue: if the classical bootstrap is supposed to somehow represent my knowledge or information about p_s, then it clearly fails. We have seen successes, yet the frequentist bootstrap places nonzero weight on the belief that the average probability of success is 0, and thus observing a success is impossible. But I've seen one. More than one in fact! It strikes one as incredibly odd that one would use a procedure which seems to predict that there is a chance one couldn't possibly have observed a value one did, in fact, observe.

Fundamentally, we aren't interested in the sampling distribution of our parameter of interest; we're interested in learning about the parameter itself, i.e. a posterior distribution, which is quite different conceptually and, as one will see, in practice. Often, statisticians can get away with ignoring the difference, but one does so at one's own peril (such as by attempting to produce a scaling plot of TTS and having an error bar that goes to ∞).

One may still counter "So long as the weight on 0 is small enough, it won't matter for practical purposes." To couch the differences in a more pragmatic context, consider also the case where we have 100 gauges (normally considered plenty for problems with appreciable probability of success in our field, say $p_s > 0.01$), each with 2000 anneals, and a distribution of probability of success chosen from a mixture of Normal$(0.05, 0.003)$ with weight 0.95 and Normal$(0.4, 0.01)$ with weight 0.05, and observe what the classical bootstrap has to say about p_s in this case as well 3.2

As you can see, the sampling distribution has a number of peaks which one would not expect from a posterior distribution for a mean. These arise due to the discrete nature of the classical bootstrap — in each sample, the bootstrap has sampled a certain discrete number of the large-mean distributions. The variation in the number of samples from Normal$(0.4, 0.01)$ overwhelms the variation over the other samples, effectively creating a Binomial distribution on the number of said gauges with Gaussian noise from the other gauges centered on each discrete value.

While this example can be "fixed" by simply collecting more data (in this example, 200 gauges would be approximately sufficient), the fundamental problem will remain — frequentist statistics estimate uncertainty due to sampling, not our uncertainty about parameters themselves.

Rubin [84] proposed the Bayesian bootstrap. It can be understood (entirely separate from the above derivation from the Dirichlet process) as based on two assumptions:

1. All values that may be sampled from the distribution have been. (One

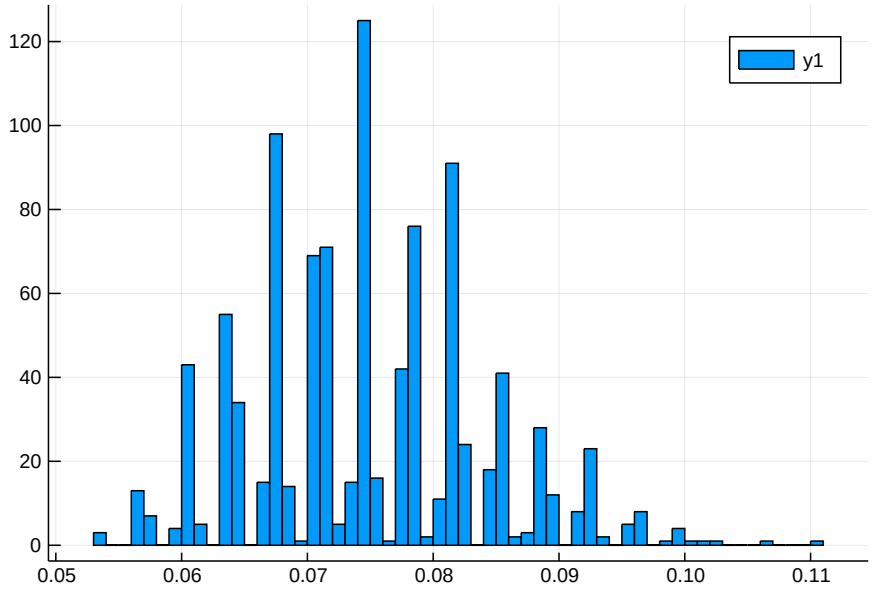


Figure 3.2: Results from 1000 frequentist bootstrap samples from a set of 100 gauges whose success probability was sampled from a mixture distribution of Normal$(0.05, 0.003)$ with weight 0.95 and Normal$(0.4, 0.01)$ with weight 0.05. The multiple peaks arise from the discrete binomial distribution over samples from the high probability component.

has seen every type in the population.)

2. The true population distribution is unknown. We merely have sampled G times and seen all G types (as per 1).

To encode (2) mathematically, we recall that the conjugate prior to the categorical and multinomial distributions is the Dirichlet distribution (whose equivalent of the Haldane prior is Dirichlet$(0, 0, 0, 0, 0, ...)$). Thus, if we take the standard prior for the G-dimensional Dirichlet distribution and update based on having seen each type once, we get the Dir$(1, 1, 1, 1, ..., 1)$ distribution (for convenience, we will refer to this distribution as Dir$(\vec{1})$ where the dimension is implicit), aka the Bayesian bootstrap as defined above from DPs. Rather than sampling a vector of finite length at each sample as in the classical bootstrap, one samples from the Dirichlet distribution to find the weights for each element in the population to get a trial population on which to compute whatever statistic one desires.

The major practical benefit of the Bayesian bootstrap is that it yields continuous results for the mean for any continuous probability distribution, and has zero weight on all probability densities which have zero weight on an observation — that is, if you observed a success, the Bayesian bootstrap will never sample

$p_s = 0$. It also has a more reasonable assumption than the classical bootstrap — rather than asserting certainty about the distribution in the population, it assumes that we are ignorant of this distribution and uses only the knowledge we have. Finally, rather than representing some sampling distribution for a statistic, we are computing a genuine posterior distribution for our information about the statistic.

Let's examine the cases where the classical bootstrap broke down again, to see the advantages of the Bayesian bootstrap. In the case of 10 gauges with 9 having no successes and one having only successes, as discussed above, the posterior will be Beta(1,9). Beta(1,9) has no support on 0, i.e. there is no weight on $TTS = \infty$. The Bayesian bootstrap never claims it may be impossible to observe observed values.

In the second case, we have the result in figure 3.3.

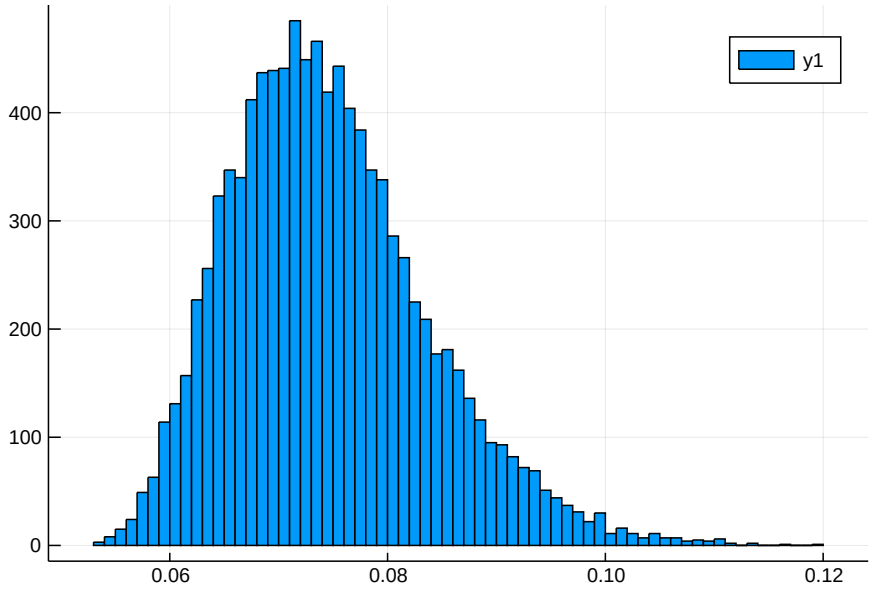


Figure 3.3: Results from 10000 Bayesian bootstrap samples from a set of 100 gauges whose success probability was sampled from a mixture distribution of Normal$(0.05, 0.003)$ with weight 0.95 and Normal$(0.4, 0.01)$ with weight 0.05. It is now a smooth density, unlike in the frequentist case.

As one can see, this is smooth and continuous, due to the continuous nature of the Bayesian bootstrap, which is also a far more reasonable posterior distribution for a mean.

The Bayesian bootstrap can be nested, just like the classical bootstrap. For instance, one may desire to compute the posterior for the median over a class of instances, such as range-1 random Ising problems for C_8 chimera graphs. To do so, for each instance sample a value from the Bayesian bootstrap distribution

for p_s. Then, compute the median over a population with those values of p_s with population weights sampled from the $\text{Dir}(\vec{1})$ (this may be very different than the median over the vector of sampled p_s alone) to extract a single sample of the median, and repeat many times to construct a Monte Carlo estimate of the posterior for the median. In general posteriors estimated in this way will be significantly smoother than estimates generated from the classical bootstrap, though may still be multimodal by the essentially discrete nature of order statistics on finite populations.

So long as the practitioner doesn't want weight on the problem being impossible after having solved it or expects posterior estimates of means to be smooth and unimodal (i.e. everyone) then the practitioner should use the Bayesian bootstrap rather than the classical bootstrap.

Now that we know how to estimate p_s, it is time to draw this paper to a close. Given an arbitrary distribution we have a technique to estimate p_s and the benchmarking problem, at least as stated above, is solved. But a way to analyze data you already have isn't really a full solution to the real benchmarking problem. We know what to do if someone hands us a collection of data. But our goal is to perform an investigation ourselves. How should one go about collecting the data in the first place?

3.2 Goldilocks and the three samples sizes: why there is no universal "right" sample size

Traditionally in science, particularly in frequentist circles, scientists are admonished to design an experimental protocol in advance, including selecting sample sizes. This is due to concerns over the possibility of induced bias in estimates under conditions of "optional stopping", i.e. determining sample sizes by peeking at the data as you go along. Indeed, the debate about optional stopping has raged for over fifty years.

Why are sample sizes usually determined in advance? A simple stopping rule can point out how optional stopping could lead to incorrect inferences: continue sampling until such time as you can reject your null hypothesis with a p-value of 0.05. Obviously, this rule will cause you to always reject the null hypothesis, even if it's true, as for any distribution there is a remote possibility of sampling a chain which would satisfy it. This proposal was given by Armitage at the Savage Forum in 1959 [108], and has been the topic of quite a bit of debate since.

Nevertheless, despite the potential dangers of optional stopping, it has one dramatic advantage for the cause of benchmarking: it allows us to run relatively few samples for easy problems, and only run many samples for problems which are very hard. This could, in principle, enormously reduce the time to gather sufficient statistics.

To gauge how significant the time savings may be, recall that we expect the number of observed successes to control the accuracy of our inference, namely

we suspect that number of gauges $G \propto 1/p_s$. We can examine the distribution over p_s for problems of interest and it's scaling as a function of system size to approximately determine the time savings. One can see in Figure 3.4, which shows approximately how many gauges would be necessary to complete gathering all data for various sizes of range-1 Ising problems tested in chapter 2, assuming we wish to solve at least 95% of them (which seems a reasonable number). As one can see, optional stopping would reduce the time to gather all data by roughly an order of magnitude for the largest system size. These benefits should be expected to compound as we go to larger systems, with a broader distribution for p_s across instances. Given that problems are also going to get exponentially harder, requiring longer data collection times, this could serve as an enormous practical reduction in difficulty.

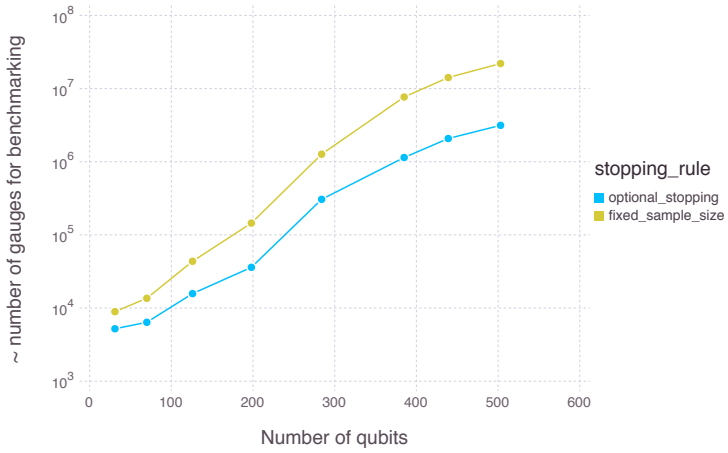


Figure 3.4: A comparison of the approximate number of gauges required to gather sufficient data to know TTS for the easiest 95% of instances for various sizes of range-1 Ising problems. Optional stopping consumes more than an order of magnitude less time for the largest size, and we should expect this advantage to continue to grow as we go to larger systems with broader distributions for p_s.

Given that we desire to use some optimal stopping rule, how might we go about selecting one?

First, it seems natural to perform inference in TTS space, as that is the space that (a) correlates with how long it takes to gather the data and (b) is the space in which costs are directly proportional, rather than connected by a nonlinear function as in p_s space. Thus, we convert our posterior densities on p_s to TTS before applying whatever stopping rule we wish. Since we want to have a large amount of information about the value of TTS, we will place a threshold on the posterior coefficient of variation (equal to σ/μ for standard deviation σ

and mean μ), such that we will stop gathering data once our estimated posterior $\sigma/\mu < c$ for small c.

This stopping rule corresponds directly to placing a threshold on the number of successes in the ideal $R = 1$ case. Such a stopping rule was analyzed in [109] and found to be virtually unbiased (with a bias upper bounded by $-1/N$ where N is the number of successes; this corresponds to a threshold $c = 1/\sqrt{N}$ in the ideal case) with guarantees on the confidence level (essentially, resulting from c). While I was unable to make significant progress in analyzing the non-ideal $R > 1$ case analytically, simulation studies detailed below will demonstrate that this proposed stopping rule is also virtually unbiased and has extremely good frequentist coverage properties (indeed, the credible intervals are often overly conservative, which could be a good or a bad thing depending on your point of view — it's either somewhat wasteful or good protection against being overconfident).

Before we see those results, what properties do we expect of this stopping rule? By the law of large numbers we expect the posterior standard deviation to decrease like $\approx 1/\sqrt{G}$, and thus for the number of gauges to be roughly inversely proportional to the square of the threshold. Thus, the bias term decreases quadratically, with inverse c^2, as do the number of gauges. $G \propto \frac{1}{\mu}$, as would be expected. In general, we expect G to also grow as the inverse square of the coefficient of variation of the underlying distribution.

3.3 The proof is in the pudding: a simulation study

To study the entire above procedure's performance, we perform a simulation study. The data collection algorithm is as follows:

1. Sample a random p from the test distribution
2. Sample a number of successes from Bin(R,p)
3. Every hundred gauges (or roughly every 3 seconds of wall-time-equivalent on D-Wave) we perform a brief Bayesian bootstrap to check if $\sigma/\mu < c$, if so exit and report the results of the trial. Else, repeat.

We test 12 distributions, across various orders of magnitude of the probability of success. I tested each at several different values of R = 50, 100, 150, and 200, and thresholds c = 0.05, 0.1, and 0.2. The distributions are:

1. Beta(1,99)
2. Beta(1,999)
3. Beta(1,9999)
4. $\delta(0.01)$

5. $\delta(0.001)$
6. $\delta(0.0001)$
7. Uniform(0,1)
8. Beta(1/2,100)
9. MixtureModel([Beta$(10, 1000)$, Beta$(1, 1)$, Beta$(1/2, 100)$], $[0.2, 0.01, 0.79]$)
10. MixtureModel([Beta$(33, 7800)$, Beta$(12, 2^14)$], $[0.3, 0.7]$)
11. MixtureModel([Beta$(5, 10^4)$, Beta$(2, 2)$], $[0.998, 0.002]$)
12. MixtureModel([Beta$(3, 1000)$, $\delta(0.9)$], $[0.95, 0.05]$)

The first 6 are relatively simple, fairly peaked, across several orders of magnitude. The Uniform distribution is also quite simple. Beta(1/2,100) is unbounded near 0 with an average near 1/200. The mixture models are written as a list of probabiliy densities and their associated weights in the model. The first mixture model is multimodal with a rare uniform mixture, that I would anticipate will yield poor performance. The final 3 are what I call "Nixon" distributions, in that they lie about their true nature, potentially tricking an experimenter into believing the average of the distribution is quite different than it's true value.

First, let us address wallclock time. How long will it take to gather sufficient data, for each of the threshold and number of runs? In figure 3.5, we can see that there is an essentially linear relationship between the inverse mean for various thresholds. In figure 3.6, we see that the number of runs makes relatively little difference in the limited range I ran in these simulations. Nevertheless, both in these and in additional distributions, not shown in these figures, it seems that approximately 100 anneals per programming cycle was either optimal or near-optimal for the vast majority of distributions tested. For a threshold of 0.2, $R = 100$ is optimal across essentially the entire range of distributions tested. Additional checks were made at much larger (and more traditional) values for R, such as 1000 and 10000, however, those were universally dramatically worse. Indeed, I was unable to build a distribution for p_s such that even $R = 1000$ could be remotely competitive to $R = 100$ in wallclock time.

This finding is fairly reasonable, and indeed was what I expected. For $R = 1$, the chip would be programming 99% of the time, yielding just 66 samples per second with 66 gauges. At $R = 100$, the chip is programming approximately half the time, yielding approximately 3300 samples per second with 33 gauges. A mere factor of 2 in sampling over gauges yields a fifty fold increase in the number of samples per second. Beyond $R = 100$, one achieves only moderate increases in the number of samples per second in exchange for a reduced number of gauges. By the time one moves much beyond $R = 200$, purchasing additional samples comes at the cost of a great number of gauges. $R = 50000$ yields only 10% more samples per second than $R = 1000$ at a fifty fold decrease in the

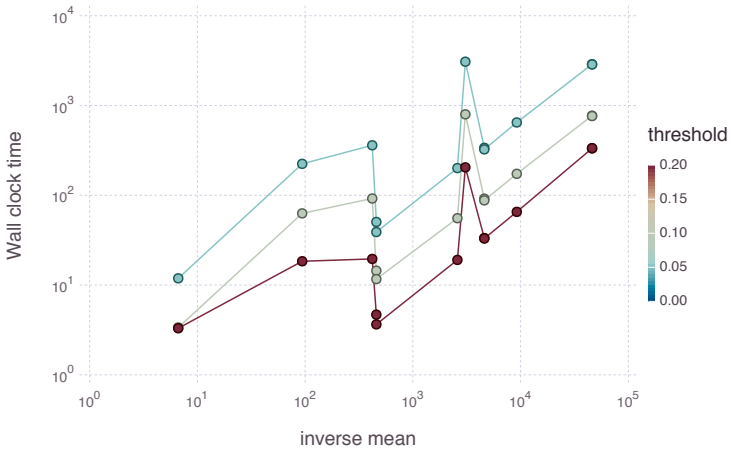


Figure 3.5: Wall clock time, estimated based on 15 ms programming time and 180 ms per anneal, vs inverse mean for the various distributions at R=100. The vertical variation along a single mean line are caused by variations in the variation among instances at a particular mean. It is an approximately linear relationship, meaning it will take linearly more time to gather sufficient statistics as the probability decreases.

number of gauges. Notice, also, that at $R = 100$, one is only sacrificing half of the maximum number of anneals possible, in exchange for 500 times as many gauges per second.

We know optional stopping may lead to biased estimates, but how biased, on average? Figure 3.7 shows histograms for different values of R at different thresholds. Clearly, 0.2 is prone to error, even quite large errors, across all values of the number of runs. However, since the time to gather data increases by a factor of four between $c = 0.2$ and $c = 0.1$, and again between $c = 0.1$ and $c = 0.05$, it may be in our interests to accept being more easily misled on a particular instance in exchange for running four times as many experiments. That is a question that must be decided on careful deliberations and may vary from experiment to experiment (or change in the middle). It should also be noted that these biases do not really exhibit any scaling and thus should not pose any issue for asymptotic scaling or speedup analyses.

Finally, let us review the credible intervals of our estimates. A credible interval is very similar to a confidence interval, however it is the product of selecting a region of high posterior density rather than the output of some frequentist procedure. For ease of computation, we will here use equal-tailed credible intervals, rather than the strict highest-posterior density interval, as the true HPD interval is not invariant, i.e. our credible intervals for time to solution and p_s would not be related by the TTS equation. Moreover, HPD intervals

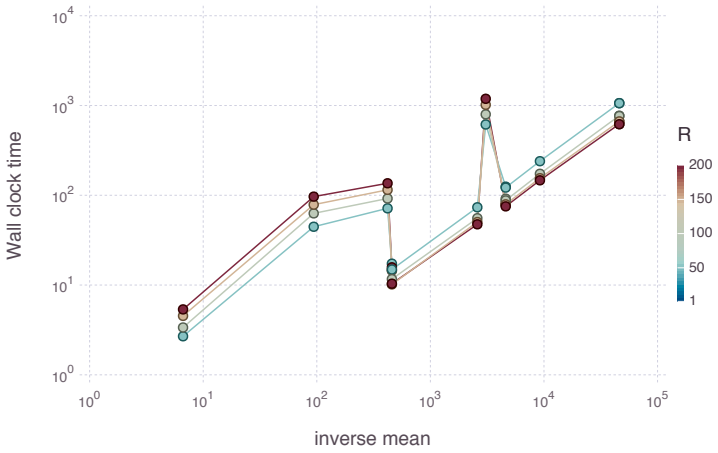


Figure 3.6: Wall clock time, estimated based on 15 ms programming time and 180 ms per anneal, vs inverse mean for the various distributions for a threshold of 0.1. The vertical variation along a single mean line are caused by variations in the variation among instances at a particular mean. We notice that R=100 is optimal or near optimal across the distributions.

are relatively difficult to compute, while equal-tailed intervals are rather trivially estimated using order statistics (the α% equal-tailed confidence interval is simply the middle α% of the posterior distribution).

Figures 3.8, 3.9, and 3.10 display how many, out of 100, simulated collections of data yielded the mean inside of the 67, 95, and 99 percent confidence intervals derived via the Bayesian bootstrap. As you can see, essentially across all thresholds, the performance of the estimators is quite good from a frequentist perspective, and indeed they are often overly conservative. It should also be noted that for 100% of the simulations for all tested distributions, the true mean was within one standard deviation of the estimated mean. Thus, we can be virtually certain that the true mean will be within the range $1 \pm c$ of our estimated mean for threshold c.

Given all of the above results and arguments, it is my opinion that we should adopt $R = 100$ as a convention, and a threshold on σ/μ of 0.2 or, if it is desired to be more conservative, 0.1. Obviously, some upper time limit will have to be imposed as well, but this is essentially a question of convenience. Checks of the exit condition can be made after every gauge if one uses the efficient update scheme mentioned in the the first section.

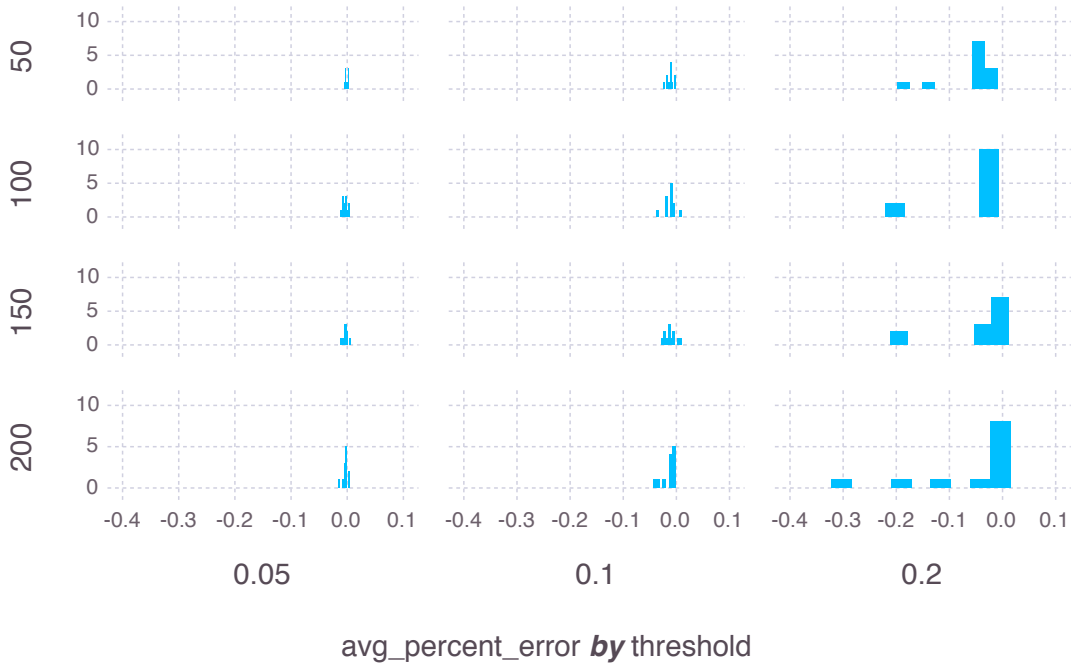


Figure 3.7: Histogram of the average relative bias of the optional stopping estimator for 100 simulations across all the listed distributions for different thresholds and numbers of runs. We see that a threshold of 0.2 can lead one fairly far astray, while the number of runs per gauge doesn't make a very significant difference. However, given that the wallclock time is inversely proportional to the square of the threshold, it may be deemed worth the trade off to accept a slightly greater degree of bias in exchange for significant reduction in cost.

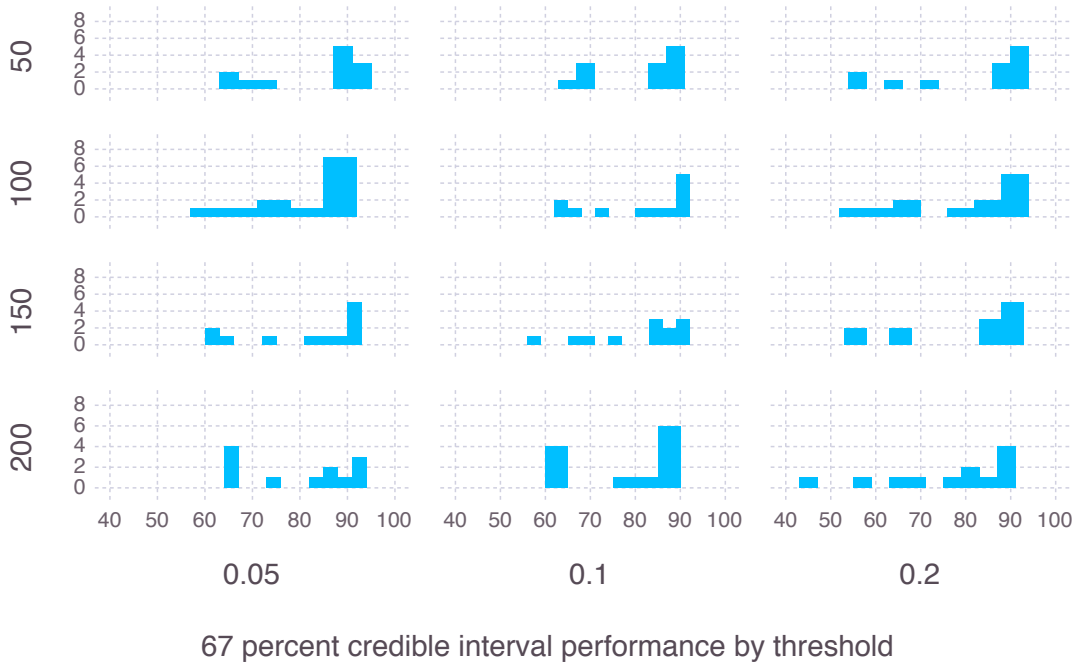


Figure 3.8: Histogram of the performance of the 67 percent credible interval out of 100 simulations for each distribution, across thresholds and runs. The x-axis in each subplot denotes the number of 100 simulations for which the mean was inside the interval. The histogram is out of the 12 distributions presented in this paper.

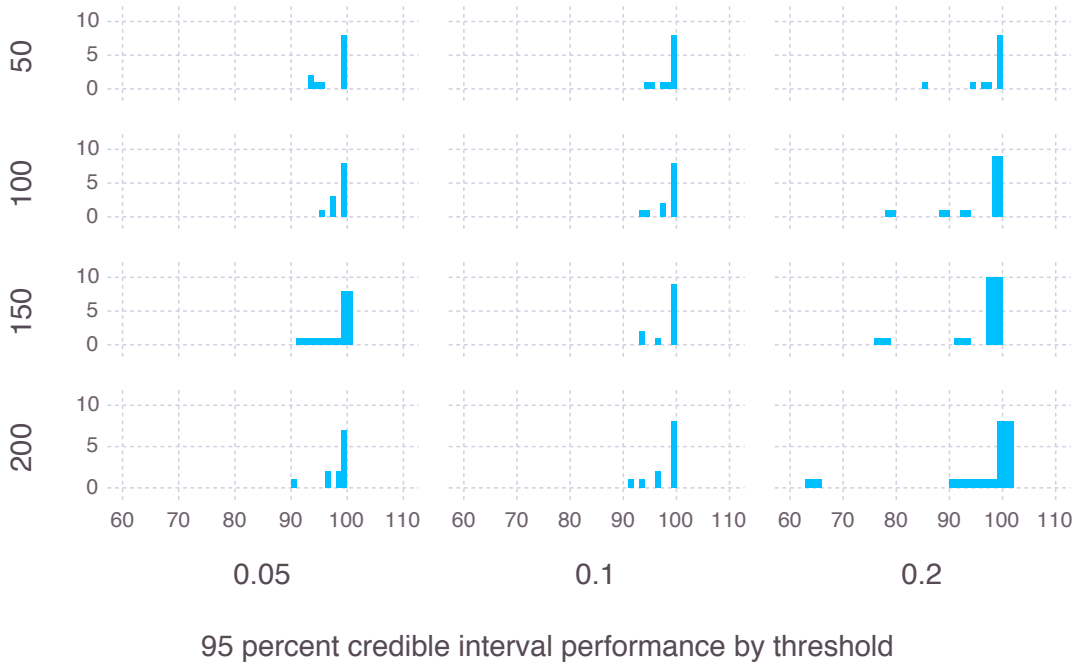


Figure 3.9: Histogram of the performance of the 95 percent credible interval out of 100 simulations for each distribution, across thresholds and runs. The x-axis in each subplot denotes the number of 100 simulations for which the mean was inside the interval. The histogram is out of the 12 distributions presented in this paper.

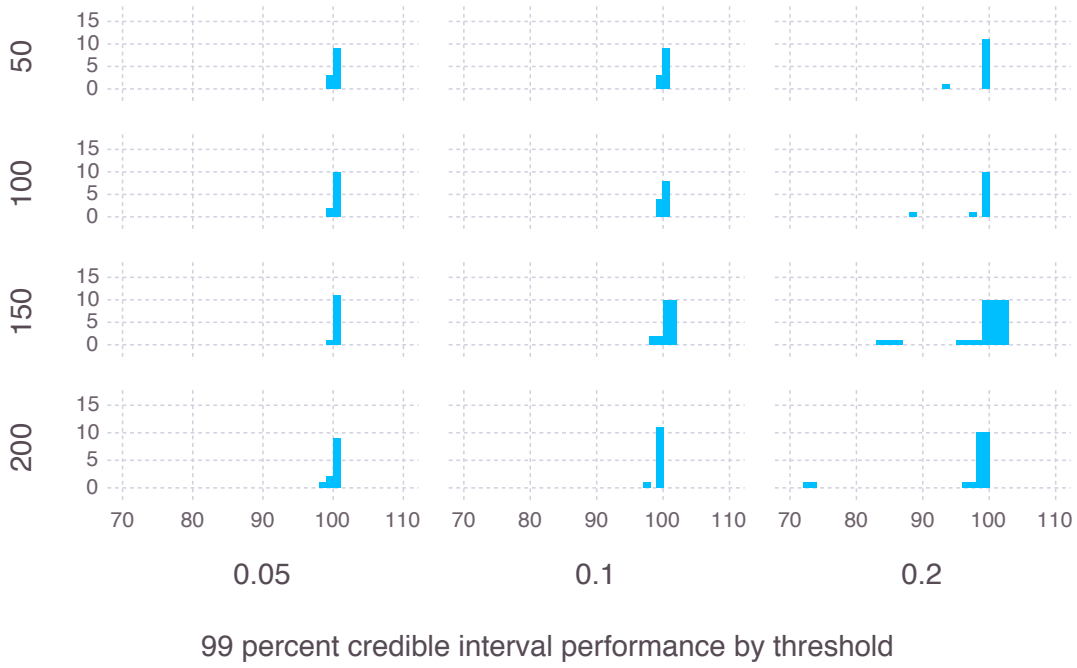


Figure 3.10: Histogram of the performance of the 99 percent credible interval out of 100 simulations for each distribution, across thresholds and runs. The x-axis in each subplot denotes the number of 100 simulations for which the mean was inside the interval. The histogram is out of the 12 distributions presented in this paper.

3.4 Conclusion

To review, we have shown that the Bayesian bootstrap is significantly better, both theoretically and practically, than the classical bootstrap. Theoretically, as it never claims it may be impossible to see things one has seen and thus never puts weight on TTS= ∞ unless you have seen no successes, and is the natural outcome of a Bayesian nonparametric analysis in the face of very weak prior information. Practically, as it yields more reasonable posteriors for the mean (unimodal, first and foremost). We have also seen how optional stopping, while being slightly biased, can save an order of magnitude or more in time, a savings that can only reasonably be expected to grow as we seek to benchmark larger and larger annealers, and still yields high quality credible intervals in a variety of circumstances. By taking seriously the question of how to represent our state of knowledge at each stage of a benchmarking experiment, we can ensure that we arrive at sensible answers at every moment in time and save significant amounts of time and effort when studying problems of widely varying hardness.

Generically, then, to effectively keep track of one's knowledge, the community should adopt a Bayesian bootstrap over gauges and a cutoff on the coefficient of variation of their posterior density for the time to solution.

Moreover, much of the reasoning above also applies in many other potential contexts that our community may encounter as we move forward. Similar procedures may be used when trying to estimate performance of quantum annealers for machine learning tasks, such as QAML (see chapter 5). Moreover, experiments on new circuit-based devices may also benefit from more rigorous analyses of the state of knowledge. Very generally, the most important task for the researcher isn't running the experiment, but accurately representing the knowledge gained from the experiment, and for all the reasons listed above, a Bayesian bootstrap procedure with a variational cutoff is likely to be both the simplest, most honest, and most efficient mechanism for completing that task in a wide variety of circumstances.

In the following two chapters, we'll see some partial applications of the techniques discussed here — partial because in the first case, chapter [**?**], I was still developing these ideas, and in the second, chapter [**?**], the parameters of interest were not time to solution or probabiltiy of success, but classifier accuracy. However, our data analysis in that case was informed by the basic principles outlined here — namely, taking seriously our state of knowledge and the properties of our parameter of interest, and sampling from meaningful posterior densities in order to produce accurate results.

Chapter 4

Probing for quantum speedup in spin glass problems with planted solutions

4.1 Introduction

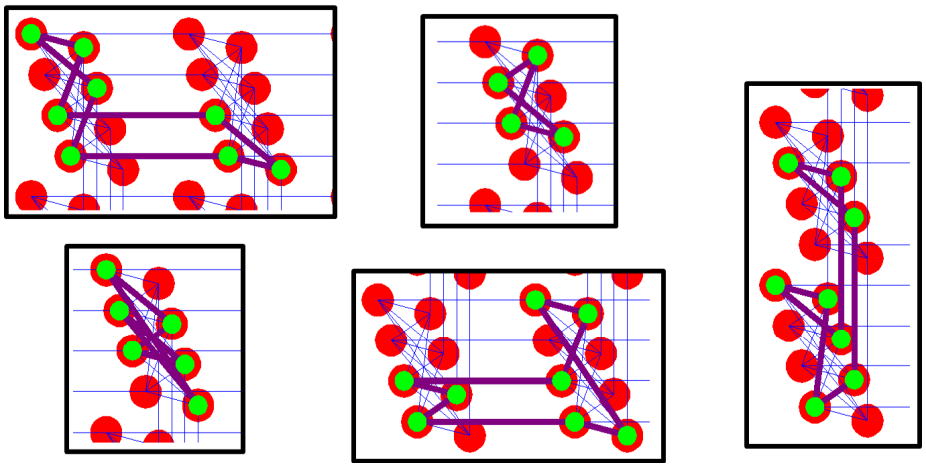
In this chapter, we go beyond earlier studies of random Ising problems, and introduce a method to construct a set of frustrated Ising-model optimization problems with tunable hardness. We study the performance of a D-Wave Two device (DW2) with up to 503 qubits on these problems and compare it to a suite of classical algorithms, including a highly optimized algorithm designed to compete directly with the DW2 (namely, HFS). The problems are generated around predetermined ground-state configurations, called planted solutions, which makes them particularly suitable for benchmarking purposes. The problem set exhibits properties familiar from constraint satisfaction (SAT) problems, such as a peak in the typical hardness of the problems, determined by a tunable clause density parameter. We bound the hardness regime where the DW2 device either does not or might exhibit a quantum speedup for this problem set. While we do not find evidence for a speedup for the hardest and most frustrated problems in our problem set, we cannot rule out that a speedup might exist for some of the easier, less frustrated problems. Our empirical findings pertain to the specific D-Wave processor and problem set we studied and leave open the possibility that future processors might exhibit a quantum speedup on the same problem set. This chapter was originally published as Ref [3], with minor modifications here.

Here we report on experimental results that probe the possibility that a putative quantum annealer may be capable of speeding up the solution of certain

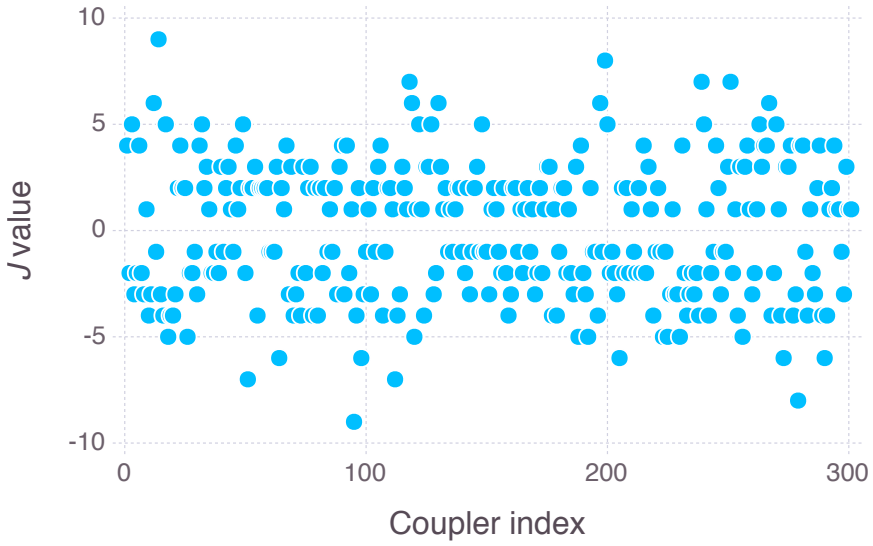
carefully designed optimization problems. We refer to this either as a *limited* or as a *potential* quantum speedup, since we study the possibility of an advantage relative to a portfolio of classical algorithms that either "correspond" to the quantum annealer (in the sense that they implement a similar algorithmic approach running on classical hardware), or implement a specific, specialized classical algorithm, in accordance with Ref [2] and chapter 2. In addition, for technical reasons detailed below we must operate the putative quantum annealer in a suboptimal regime. We achieve this by designing Ising model problems that exhibit frustration, a well-known feature of classically-hard optimization problems [110, 111].

The putative quantum annealer used in this study is the D-Wave Two (DW2) device [112], the same as in chapter 2.

A demonstration of speedup on a meaningful problem has so far been an elusive goal, possibly because the random Ising problems chosen in previous benchmarking tests [13, 2], up to the date of the study discussed here, were too easy to solve for the classical algorithms against which the D-Wave devices were compared; namely such problems exhibit a spin-glass phase only at zero temperature [93]. The Sherrington-Kirkpatrick model with random ± 1 couplings, exhibiting a positive spin-glass critical temperature, was tested on a DW2 device, but the problem sizes considered were too small (due to the need to embed a complete graph into the Chimera graph) to test for a speedup [83]. The approach we outline next allows us to directly probe for a quantum speedup using frustrated Ising spin glass problems with a tunable degree of hardness, though we do not know whether these problems exhibit a positive-temperature spin-glass phase.



(a)



(b)

Figure 4.1: **Examples of randomly generated loops and couplings on the DW2 Chimera graph.** Top: qubits and couplings participating in the loops are highlighted in green and purple, respectively. Only even-length loops are embeddable on the Chimera graph. Bottom: distribution of J values for a sample problem instance with $N = 126$ spins and edges, and 101 loops. It is virtually impossible to recover the loop-Hamiltonians H_j from a given H_{Ising}. The couplings are all eventually rescaled to lie in $[-1, 1]$. We always set the local fields h_i to zero as non-zero fields tended to make the problems easier.

4.2 Frustrated Ising problems with planted solutions

In this section we introduce a method for generating families of benchmark problems that have a certain degree of "tunable hardness", achieved by adjusting the amount of frustration, a well-known concept from spin-glass theory [113]. In frustrated optimization problems no configuration of the variables simultaneously minimizes all terms in the cost function, often causing classical algorithms to get stuck in local minima, since the global minimum of the problem satisfies only a fraction of the Ising couplings and/or local fields. We construct our problems around "planted solutions" – an idea borrowed from constraint satisfaction (SAT) problems [114, 115]. The planted solution represents a ground state configuration of Eq. (1.1) that minimizes the energy and is known in advance. This knowledge circumvents the need to verify the ground state energy using exact (provable) solvers, which rapidly become too expensive computationally as the number of variables grows, and which were employed in earlier benchmarking studies ([13, 2],2).

To create Ising-type optimization problems with planted solutions, we make use of frustrated "local Ising Hamiltonians" H_j, i.e., Ising problem instances defined on sub-graphs of the Chimera graph in lieu of the clauses appearing in the SAT formulas. The total Hamiltonian for each problem is then of the form $H_{\text{Ising}} = \sum_{j=1}^{M} H_j$, where the sum is over the (possibly partially overlapping) local Hamiltonians. Similarly to SAT problems, the size of these local Hamiltonians, or clauses, does not scale with the size of the problem. Moreover, to ensure that the planted solution is a ground state of the total Hamiltonian, we construct the clauses so that each is minimized by that portion of the planted solution that has support over the corresponding subgraph.[1] The planted solution is therefore determined prior to constructing the local Hamiltonians, by assigning random values to the bits on the nodes of the graph. The above process generates a Hamiltonian with the property that the planted solution is a simultaneous ground state of all the frustrated local Hamiltonians.[2]

The various clauses H_j can be generated in many different ways. This freedom allows for the generation of many different types of instances, and here we present one method. An N-qubit M-clause instance is generated as follows.

1. A random configuration of N bits corresponding to the participating spins of the Chimera graph is generated. This configuration constitutes the planted solution of the instance.

2. M random loops are constructed along the edges of the Chimera graph.

[1] To see that the planted solution minimizes the total Hamiltonian, assume that a configuration x_* is a minimum of $f_j(x)$ for all j, where $\{f_j(x)\}$ is a set of arbitrary real-valued functions. Then, by definition, for each j, $f_j(x) \geq f_j(x_*)$ for all possible configurations x. Let us now define $f(x) \equiv \sum_j f_j(x)$. It follows then that $f(x) \geq \sum_j f_j(x_*)$. Since also $f(x_*) = \sum_j f_j(x_*)$, x_* is a minimizing configuration of $f(x)$.

[2] Somewhat confusingly from our perspective of utilizing frustration, such Hamiltonians are sometimes called "frustration-free" [116].

The loops are constructed by placing random walkers on random nodes of the Chimera graph, where each edge is determined at random from the set of all edges connected to the last edge. The random walk is terminated once the random walker crosses its path, i.e., when a node that has already been visited is encountered again. If this node is not the origin of the loop the "tail" of the path is discarded. Examples of such loops are given in Fig. 4.1. We distinguish between "short loops" of length $\ell = 4, 6$, and "long loops" of length $\ell \geq 8$, as these give rise to peaks in hardness at different loop densities. Here we focus on long loops; results for short loops, which tend to generate significantly harder problem instances, will be presented elsewhere.

3. On each loop, a clause H_j is defined by assigning $J_{ij} = \pm 1$ couplings to the edges of the loop in such a way that the planted solution minimizes H_j. As a first step, the J_{ij}'s are set to the ferromagnetic $J_{ij} = -s_i s_j$, where the s_i are the planted solution values. One of the couplings in the loop is then chosen at random and its sign is flipped. This ensures that no spin configuration can satisfy every edge in that loop, and the planted solution remains a global minimum of the loop, but is now a frustrated ground state.[3]

4. The total Hamiltonian is then formed by adding up the M-loop clauses H_j. Note that loops can partially overlap, thereby also potentially "canceling out" each other's frustration, a useful feature that will give rise to an easy-hard-easy pattern we discuss below. Since the planted solution is a ground state of each of the H_j's, it is also a ground state of the total Hamiltonian H_{Ising}.

Ising-type optimization problems with planted solutions, such as those we have generated, have several attractive properties that we utilize later on: i) Having a ground-state configuration allows us to readily precompute a measure of frustration, e.g., the fraction of frustrated couplings of the planted solution. We shall show that this type of measure correlates well with the hardness of the problem, as defined in terms of the success probability of finding the ground state or the scaling of the time-to-solution. ii) By changing the number of clauses M we can create different classes of problems, each with a "clause density" $\alpha = M/N$, analogous to problem generation in SAT.[4] The clause density can be used to tune through a SAT-type phase transition [117], i.e., it may be used to control the hardness of the generated problems. Here too, we shall see that the clause density plays an important role in setting the hardness of the

[3] To see that there can be no spin configuration with an energy lower than that of the planted solution, consider a given loop and a given spin in that loop; note that every spin participates in two couplings. Either both couplings are satisfied after the sign flip, or one is satisfied and the other is not. Correspondingly, flipping that spin will thus either raise the energy or leave it unchanged. This is true for all spins in the loop.

[4] Note that for small values of M, the number of spins actually participating in an instance will be smaller than N, the number of spins on the graph from which the clauses are chosen.

problems. Note that when the energy is unchanged under a spin-flip the solution is degenerate, so that our planted solution need not be unique.

4.3 Algorithms and scaling

A judicious choice of classical algorithms is required to ensure that our test of a limited or potential quantum speedup is meaningful, as discussed in previous chapters. We considered (i) simulated annealing (SA), (ii) simulated quantum annealing (SQA), (iii) the Shin-Smolin-Smith-Vazirani (SSSV) thermal rotor model (also called spin-vector Monte Carlo or SVMC), and (iv) the Hamze-Freitas-Selby (HFS) algorithm [20, 21], an algorithm that is fine-tuned for the Chimera graph and appears to be the most competitive at this time. Of these, the HFS algorithm is the only one that is designed to exploit the scaling of the treewidth of the Chimera graph (see 4.6) , which renders it particularly efficient in a comparison against the D-Wave devices [118]. These are the same algorithms given an overview back in the first chapter, 1.

The D-Wave devices and all the algorithms we considered are probabilistic, and return the correct ground state with some probability of success. We thus perform many runs of the same duration τ for a given problem instance, and estimate the success probability empirically as the number of successes divided by the number of runs. This is repeated for many instances at a given clause density α and number of variables N, and generates a distribution of success probabilities. Let $p(\lambda)$ denote the success probability for a given set of parameters $\lambda = \{N, \alpha, q\}$, where q denotes the qth percentile of this distribution; e.g., half the instances for given N and α have a higher empirical success probability than the median $p(N, \alpha, 0.5)$. The *number of runs* required to find the ground state at least once with a desired probability p_d, once again, is [13, 2][5]

$$r(\lambda) = \frac{\log(1 - p_d)}{\log(1 - p(\lambda))} \ , \tag{4.1}$$

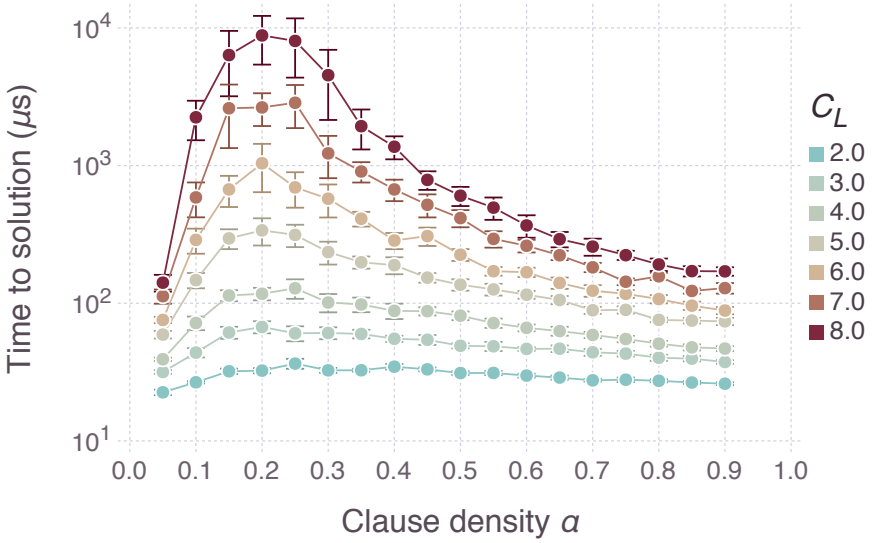
and henceforth we set $p_d = 0.99$ as in previous chapters. Correspondingly, the *time-to-solution* is $\mathrm{TTS}(\lambda) = r(\lambda)\tau$, where for the D-Wave device τ signifies the annealing time t_a (at least 20μs for the DW2), while for SA, SQA, and SSSV τ is the number of Monte Carlo sweeps s (a complete update of all spins) multiplied by the time per sweep τ_X for algorithm X, again as in chapter 2.[6] For SA, we further distinguish between using it as a *solver* (SAS) or as an *annealer* (SAA): in SAS mode we keep track of the energies found along the annealing schedule (which we take to be linear in the inverse temperature β) and take the lowest, while in SAA mode we always use the final energy found. Thus SAA can never be better than SAS, but is a more faithful model of an analog annealing device. A similar distinction can be made for SQA (i.e., SQAA and SQAS), but we primarily consider the annealer version since it too more closely mimics

[5] We prefer to define r in this manner, rather than rounding it as in [13, 2], as this simplifies the extraction of scaling coefficients.

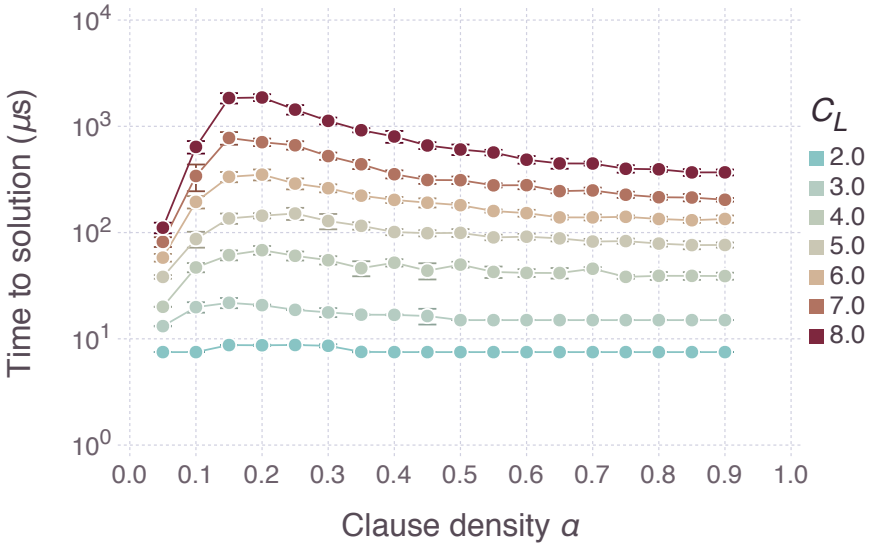
[6] In our simulations $\tau_{\mathrm{SA}} = 3.54\mu$s, $\tau_{\mathrm{SQA}} = 9.92\mu$s, and $\tau_{\mathrm{SSSV}} = 10.34\mu$s.

the operation of DW2 (note that SAA and SQAA were also the modes used in Refs. [13, 2]; SQAS results are shown in Section 4.7). For the HFS algorithm, TTS(λ) is calculated directly from the distribution of runtimes obtained from 10^5 identical, independent executions of the algorithm. Further timing details are given in 4.6.

We next briefly discuss how to properly address resource scaling, as a refresher from chapter 2. The D-Wave devices use N qubits and $O(N)$ couplers to compute the solution of the input problem. Thus, it uses resources which scale linearly in the size of the problem, and we should give our classical solvers the same opportunity. Namely, we should allow for the use of linearly more cores and memory for our classical algorithms as we scale the problem size. For annealers such as SA, SQA, and SSSV, this is trivial as we can exploit the bipartite nature of the Chimera graph to perform spin updates in parallel so that each sweep takes constant time and a linear number of cores and memory. The HFS algorithm is not perfectly parallelizable but as I explain in more detail for interested readers in 4.6, we take this into account as well. Finally, note that dynamic programming can always find the true ground state in a time that is exponential in the Chimera graph treewidth, i.e., that scales as $\exp\left(c\sqrt{N}\right)$ [11]. The natural size scale in our study is the square Chimera subgraph C_L comprising L^2 unit cells of 8 qubits each, i.e., $N = 8L^2$. Therefore, henceforth we replace N by L so that $\lambda = \{L, \alpha, q\}$. This is now standard practice for benchmarking studies of Chimera-structured annealers, but was not as of the time of the study in chapter 2.



(a)



(b)

Figure 4.2: **Time to solution as a function of clause density.** Shown is $\mathrm{TTS}(L, \alpha, 0.5)$ (log scale) for (a) DW2 and (b) HFS, as a function of the clause density. The different colors represent the different Chimera subgraph sizes, which continue to $L = 12$ in the HFS case. In both cases there is a clear peak. From the HFS results we can identify the peak position as being at $\alpha = 0.17 \pm 0.01$, which is consistent with the peak position in the DW2 results. Error bars represent 2σ confidence intervals.

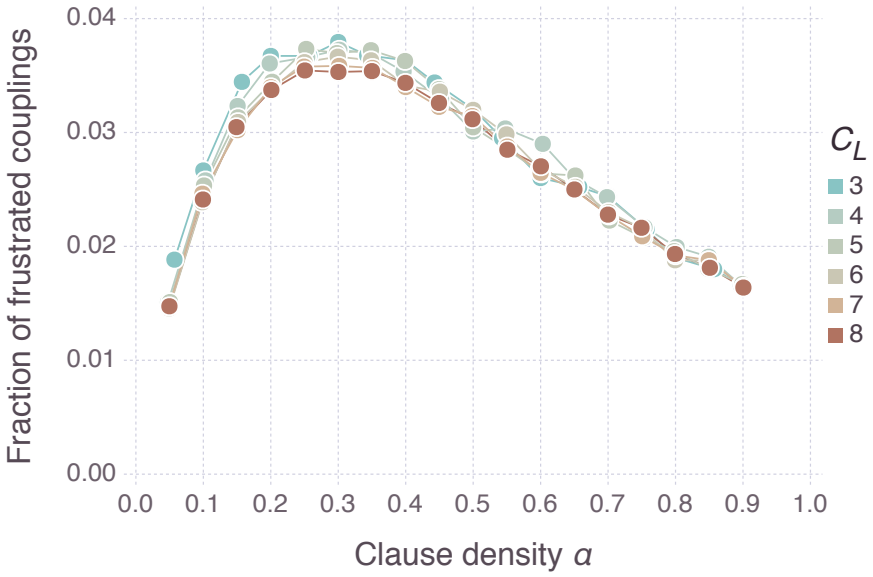


Figure 4.3: **Frustration fraction.** Shown is the fraction of frustrated couplings (the number of frustrated couplings divided by the total number of couplings, where a frustrated coupling is defined with respect to the planted solution) as a function of clause density for different Chimera subgraphs C_L, in the case of loops of length ≥ 8, averaged over the 100 instances for each given α and N. There is a broad peak at $\alpha \approx 0.25$. This is the clause density at which there is the largest fraction of frustrated couplings, and is near where we expect the hardest instances to occur, in good agreement with Fig. 4.2.

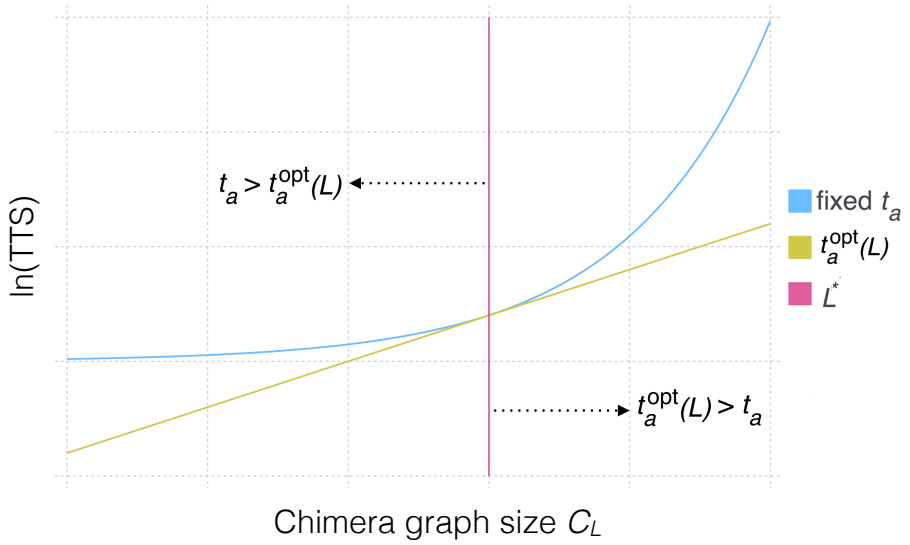


Figure 4.4: **Sketch of the relation between the** log(**TTS**) **curves for optimal and suboptimal annealing times.** The annealing time needs to be optimized for each problem size. Blue represents the TTS with a size-independent annealing time t_a. Red represents the optimal TTS corresponding to having an optimal size-dependent annealing time $t_a^{\text{opt}}(L)$, i.e. the lower envelope of the full series of fixed annealing time TTS curves. This curve need not be linear as depicted, though we expect it to be linear for NP-hard problems. The blue line upper bounds the red line since by definition $\text{TTS}_{\text{DW2}}(\lambda, t_a^{\text{opt}}(L)) \leq \text{TTS}_{\text{DW2}}(\lambda, t_a)$. The vertical dotted line represents the problem size L^* at which $t_a = t_a^{\text{opt}}(L^*)$. To the left of this line $t_a > t_a^{\text{opt}}$ and the slope of the fixed-t_a TTS curve lower-bounds the slope of the optimal TTS curve, since for very small problem sizes a large t_a results in insensitivity to problem size, and the success probability is essentially constant. The opposite happens to the right of this line, where $t_a < t_a^{\text{opt}}$, and where the success probability rapidly drops with L at fixed t_a.

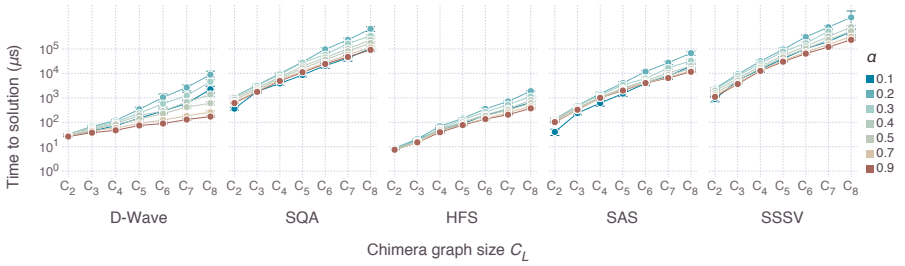


Figure 4.5: **Median time-to-solution over instances.** Plotted is $\mathrm{TTS}(L, \alpha, 0.5)$ (log scale) as a function of the Chimera subgraph size C_L, for a range of clause densities and for all solvers we tested. Note that only the scaling matters and not the actual TTS, since it is determined by constant factors that vary from processor to processor, compiler options, etc. All algorithms' timing reflects the result after accounting for parallelism, as described in 4.6. Error bars represent 2σ confidence intervals. The DW2 annealing time is $t_a = 20\mu$s.

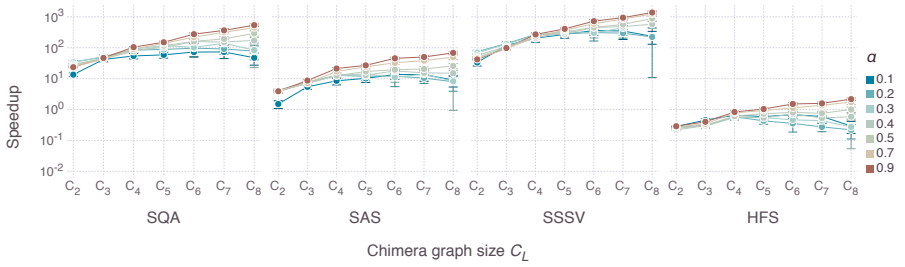


Figure 4.6: **Speedup ratio.** Plotted is the median speedup ratio $S_X(L, \alpha, 0.5)$ (log scale) as defined in Eq. (4.2) for all algorithms tested. A negative slope indicates a definite slowdown for the DW2. A positive slope indicates the possibility of an advantage for the DW2 over the corresponding classical algorithm. This is observed for $\alpha > 0.4$ in the comparison to SAS, SQA, SSSV, and HFS (see Fig. 4.10 for a more detailed analysis). Error bars represent 2σ confidence intervals.

4.4 Probing for a quantum speedup

We now come to our main goal for this study, which is to probe for a limited or potential quantum speedup on our frustrated Ising problem set. We reserve the term "limited speedup" for our comparisons with SA, SQA, and SSSV, while the term "potential speedup" refers to the comparison with the HFS algorithm, which unlike the other three algorithms, does not implement a similar algorithmic approach to a quantum annealer.

4.4.1 Dependence on clause density

We first analyze the effect of the clause density. This is shown in Fig. 4.2, where we plot $\mathrm{TTS}(L, \alpha, 0.5)$ for the DW2 and the HFS algorithm.[7] We note that the worst-case TTS of $\sim$10ms is smaller than that observed for random Ising problems in Ref. [2] [$\sim$100ms for range 7, C_8, and $q = 0.5$ (median)]. However, as we shall demonstrate, the classical algorithms against which the DW2 was benchmarked in Ref. [2] (SA and SQA) scale significantly less favorably in the present case, i.e., whereas in Ref. [2] no possibility of a limited speedup against SA was observed for the median, here we will find that such a possibility remains. In this sense, the problem instances considered here are relatively harder for the classical solvers than those of Ref. [2].

For our choice of random loop characteristics, the time-to-solution peaks at a clause density $\alpha \approx 0.17$, reflecting the hardness of the problems in that regime. To correlate the hardness of the instances with their degree of frustration, we plot the frustration fraction, defined as the ratio of the number of unsatisfied edges with respect to the planted solution to the total number of edges on the graph, as a function of clause density. The frustration fraction curve, shown in Fig. 4.3, has a peak at $\alpha \approx 0.25$, confirming that frustration and hardness are indeed correlated.[8] The hardness peak is reminiscent of the analogous situation in SAT, where the clause density can be tuned through a phase transition between satisfiable and unsatisfiable phases [117]. The peak we observe may be interpreted as a finite-size precursor of a phase transition. This interpretation is corroborated below by time-to-solution results of all other tested algorithms which will also find problems near the critical point the hardest. Indeed, all the algorithms we studied exhibit qualitatively similar behavior to that seen in Fig. 4.2 (see Fig. 4.16 in Section 4.7), with an easy-hard-easy pattern separated at $\alpha \approx 0.2$. This is in agreement with previous studies, e.g., for MAX 2-SAT problems on the DW1 [14], and for k-SAT with $k > 2$, where a similar pattern is found for backtracking solvers [119]. It is important to note that we do not claim that this easy-hard-easy transition coincides with a spin-glass phase transition; we have not studied which phases actually appear in our problem set as

[7] In this study we focus mostly on the median since with 100 instances per setting, higher quantiles tend be noise-dominated.

[8] Note that in our definition of frustration fraction, frustration is measured with respect to all edges of the graph from which clauses are chosen, similarly to the way clause density is defined in SAT problems.

we tune the clause density.

A qualitative explanation for the easy-hard-easy pattern is that when the number of loops (and hence α) is small they do not overlap and thus each loop becomes an easy optimization problem. In the opposite limit many loops pass through each edge, thus tending to reduce frustration, since each loop contributes either a "frustrated" edge with small probability $1/\ell$ (where ℓ is the loop length) or an "unfrustrated" edge with probability $1-1/\ell$. The hard problems thus lie in between these two limits, where a constant fraction (bounded away from 0 and 1) of loops overlap.

4.4.2 General considerations concerning scaling and speedup

Of central interest is the question of whether there is any scaling advantage (with L) in using a quantum device to solve our problems. Therefore, we define a quantum speedup ratio for the DW2 relative to a given algorithm X as in chapter 2.

$$S_X(\lambda, t_a) = \frac{\mathrm{TTS}_X(\lambda)}{\mathrm{TTS}_{\mathrm{DW2}}(\lambda, t_a)} \ . \tag{4.2}$$

Since this quantity is specific to the DW2 we refer to it simply as the empirical "DW2 speedup ratio" from now on, though of course it generalizes straightforwardly for any other putative quantum annealer or processor against which algorithm X is compared.

We must be careful in using $S_X(\lambda, t_a)$ in assessing a speedup, since the annealing time must be optimized for each problem size L in order to avoid the pitfall of a fake speedup [2]. Let us denote the (unknown) optimal annealing time by $t_a^{\mathrm{opt}}(L)$. By definition, $\mathrm{TTS}_{\mathrm{DW2}}(\lambda, t_a^{\mathrm{opt}}(L)) \leq \mathrm{TTS}_{\mathrm{DW2}}(\lambda, t_a)$, where t_a is a fixed annealing time. We now define the *optimized speedup ratio* as

$$S_X^{\mathrm{opt}}(\lambda, t_a^{\mathrm{opt}}(L)) = \frac{\mathrm{TTS}_X(\lambda)}{\mathrm{TTS}_{\mathrm{DW2}}(\lambda, t_a^{\mathrm{opt}}(L))} \tag{4.3}$$

and clearly $S_X(\lambda, t_a) \leq S_X^{\mathrm{opt}}(\lambda, t_a^{\mathrm{opt}}(L))$, i.e., the speedup ratios computed using Eq. (4.2) are lower bounds on the optimized speedup. However, what matters for a speedup is the *scaling* of the speedup ratio with problem size. Thus, we are interested not in the numerical value of the speedup ratio but rather in the slope dS/dL (recognizing that this is a formal derivative since L is a discrete variable). A positive slope would indicate a DW2 speedup, while a negative speedup slope would indicate a slowdown.

We thus define the *DW2 speedup regime* as the set of problem sizes $\mathcal{L}^+$ where $\frac{d}{dL} S_X(\lambda, t_a^{\mathrm{opt}}(L)) > 0$ for all $L \in \mathcal{L}^+$. Likewise, the *DW2 slowdown regime* is the set of problem sizes $\mathcal{L}^-$ where $\frac{d}{dL} S_X(\lambda, t_a^{\mathrm{opt}}(L)) < 0$ for all $L \in \mathcal{L}^-$.

From a computational complexity perspective one is ultimately interested in the asymptotic performance, i.e., the regime where L becomes arbitrarily large. In this sense a *true* speedup would correspond to the observation that $\mathcal{L}^+ = [L^+_{\mathrm{min}}, L^+_{\mathrm{max}}]$, with L^+_{min} a positive constant and $L^+_{\mathrm{max}} \to \infty$. Of course, such a definition is meaningless for a physical device such as the DW2, for which

$L^+_{\max}$ is necessarily finite. Thus the best we can hope for is an observation that $\mathcal{L}^+$ is as large as is consistent with the device itself, which in our case would imply that $\mathcal{L}^+ = [1, 8]$. However, as we shall argue, we can in fact only rule out a speedup, while we are unable to confirm one. I.e., we are able to identify $\mathcal{L}^- = [L^-_{\min}, L^-_{\max}]$, but not $\mathcal{L}^+$.

The culprit, as in earlier benchmarking work [2] and as we establish below, is the fact that the DW2 minimum annealing time of $t_a = 20\mu$s is too long (see also Section 4.7). This means that the smaller the problem size the longer it takes to solve the corresponding instances compared to the case with an optimized annealing time, and hence the observed slope of the DW2 speedup ratio should be interpreted as a *lower bound* for the optimal scaling. This is illustrated in Fig. 4.4. Without the ability to identify $t_a^{\rm opt}(L)$ we do not know of a way to infer, or even estimate $\mathcal{L}^+$. However, as we now demonstrate, under a certain reasonable assumption we can still *bound* $\mathcal{L}^-$.

The assumption is that if $t_a > t_a^{\rm opt}$ then $\frac{d}{dL}\mathrm{TTS}_{\rm DW2}(\lambda, t_a) \leq \frac{d}{dL}\mathrm{TTS}_{\rm DW2}(\lambda, t_a^{\rm opt}(L))$ for all $L < L^*$, the problem size for which $t_a = t_a^{\rm opt}(L^*)$. This assumption is essentially a statement that the TTS is monotonic in L [9], as illustrated in Fig. 4.4. Next we consider the (formal) derivatives of Eqs. (4.2) and (4.3):

$$\frac{\frac{d}{dL}S_X(\lambda, t_a)}{S_X(\lambda, t_a)} = \frac{\partial_L \mathrm{TTS_X}(\lambda)}{\mathrm{TTS_X}(\lambda)} - \frac{\partial_L \mathrm{TTS}_{\rm DW2}(\lambda, t_a)}{\mathrm{TTS}_{\rm DW2}(\lambda, t_a)} \tag{4.4a}$$

$$\frac{\frac{d}{dL}S_X(\lambda, t_a^{\rm opt}(L))}{S_X(\lambda, t_a^{\rm opt}(L))} = \frac{\partial_L \mathrm{TTS_X}(\lambda)}{\mathrm{TTS_X}(\lambda)} - \frac{\frac{d}{dL}\mathrm{TTS}_{\rm DW2}(\lambda, t_a^{\rm opt}(L))}{\mathrm{TTS}_{\rm DW2}(\lambda, t_a^{\rm opt}(L))} \,. \tag{4.4b}$$

Collecting these results we have

$$\frac{S_X(\lambda, t_a)}{S_X(\lambda, t_a^{\rm opt})}\frac{d}{dL}S_X(\lambda, t_a^{\rm opt}) < \frac{d}{dL}S_X(\lambda, t_a) \text{ if } t_a > t_a^{\rm opt}(L).$$

Therefore, if we find that $\frac{d}{dL}S_X(\lambda, t_a) < 0$ in the suboptimal regime where $t_a > t_a^{\rm opt}(L)$, then it follows that $\frac{d}{dL}S_X(\lambda, t_a^{\rm opt}(L)) < 0$. In other words, *a DW2 speedup is ruled out if we observe a slowdown using a suboptimal annealing time.*

4.4.3 Scaling and speedup ratio results

In Fig. 4.5 we show the scaling of the median time-to-solution for all algorithms studied, for a representative set of clause densities. All curves appear to match the general dynamic programming scaling for $L \gtrsim 4$, i.e., $\mathrm{TTS}(\lambda) \sim \exp[b(\alpha)L]$, but the scaling coefficient $b(\alpha)$ clearly varies from solver to solver. This scaling is similar to that observed in previous benchmarking studies of random Ising instances [13, 2].

[9] Individual instances may not satisfy this assumption [120, 121], but we are not aware of any cases where averaging over an ensemble of instances violates this assumption.

In Fig. 4.6 we show the median scaling of S_X for the same set of clause densities as shown in Fig. 4.5. We observe that in all cases there is a strong dependence on the clause density α, with a negative slope of the DW2 speedup ratio for the lower clause densities, corresponding to the harder, more frustrated problems. In this regime the DW2 exhibits a scaling that is worse than the classical algorithms and by Eq. (4.5) there is no speedup. The possibility of a DW2 speedup remains open for the higher clause densities, where a positive slope is observed, i.e., the DW2 appears to find the easier, less frustrated problems easier than the classical solvers. This apparent advantage is most pronounced for $\alpha \geq 0.4$, where we observe the possibility of a potential speedup even against the highly fine-tuned HFS algorithm (this is seen more clearly in Fig. 4.10). Moreover, the DW2 speedup ratio against HFS improves slightly at the higher percentiles (see Fig. 4.20 in Section 4.7), which is encouraging from the perspective of a potential quantum speedup.

4.4.4 Scaling coefficient results

To test the dependence on t_a, we repeated our DW2 experiments for $t_a \in [20, 40]\mu$s, in intervals of 2μs. Figure 4.7a is a success probability correlation plot between $t_a = 20\mu$s and $t_a = 40\mu$s, at $\alpha = 0.35$. The correlation appears strong, suggesting that the device might already have approached the asymptotic regime where increasing t_a does not modify the success probabilities. To check this more carefully let us first define the normalized Euclidean distance between two length-M vectors of probabilities $\vec{p}_1$ and $\vec{p}_2$ as

$$D(\vec{p}_1, \vec{p}_2) := \frac{1}{M} \|\vec{p}_1 - \vec{p}_2\| \tag{4.5}$$

(where $\|\vec{p}\| = \sqrt{\vec{p} \cdot \vec{p}}$); clearly, $0 \leq D(\vec{p}_1, \vec{p}_2) \leq 1$. The result for the DW2 data with $\vec{p}_1$ and $\vec{p}_2$ being the ordered sets of success probabilities for all instances with given α at $t_a = 20\mu$s and $t_a = 40\mu$s respectively, is shown as the blue circles in Fig. 4.8. The small distance for all α suggests that for $t_a \geq 20\mu$s the distribution of ground state probabilities has indeed nearly reached its asymptotic value.

This result means that the number of runs $r(\lambda)$ [Eq. (4.1)] does not depend strongly on t_a either. To demonstrate this we first fit the number of runs to

$$r(L, \alpha, 0.5) = e^{a(\alpha)+b(\alpha)L}, \tag{4.6}$$

in accordance with the abovementioned expectation of the scaling with the treewidth, and find a good fit across the entire range of clause densities (see Fig. 4.25a in Section 4.7). The corresponding scaling coefficient $b(\alpha)$ is plotted in Fig. 4.9 for all annealing times (the constant $a(\alpha)$ is shown in Fig. 4.25b in Section 4.7 and is well-behaved); the data collapses nicely, showing that the scaling coefficient $b(\alpha)$ has already nearly reached its asymptotic value. Also plotted in Fig. 4.9 is the scaling coefficient $b(\alpha)$ for HFS and for SAS with an optimized numbers of sweeps at each α and L, as extracted from the data shown in Fig. 4.5.

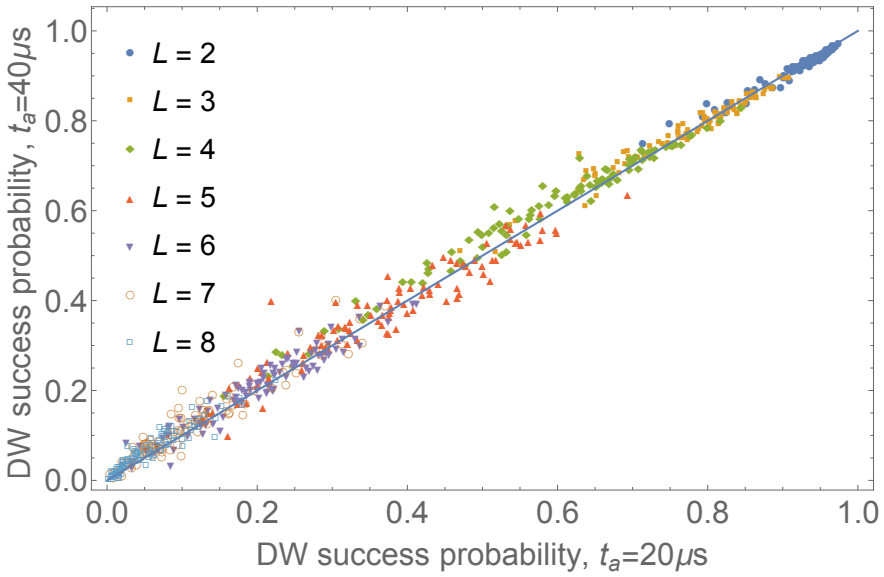
By Eq. (4.5), *where* $b_{\mathrm{DW2}}(\alpha) \geq b_{\mathrm{X}}(\alpha)$ *there is no DW2 speedup* ($\mathcal{L}^- = [4, 8]$), *whereas where* $b_{\mathrm{DW2}}(\alpha) < b_{\mathrm{X}}(\alpha)$, *a DW2 speedup over algorithm X is still possible.*

We thus plot the difference in the scaling coefficients in Fig. 4.10. Figures 4.9 and 4.10 do allow for the possibility of a speedup against both HFS and SAS, at sufficiently high clause densities. However, we stress once more that the smaller DW2 scaling coefficient may be a consequence of $t_a = 20\mu$s being excessively long, and that we cannot rule out that the observed regime of a possible speedup would have disappeared had we been able to optimize t_a by exploring annealing times shorter than 20μs. Note that an optimization of the SAS annealing schedule might further improve its scaling, but the same cannot be said of the HFS algorithm, and it seems unlikely that it could be outperformed even by the fully optimized version of SAS.

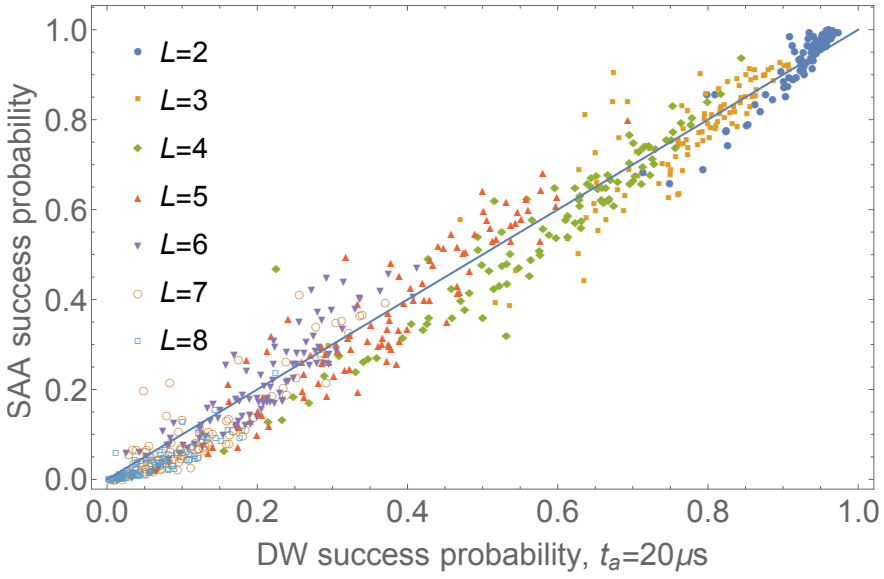
4.4.5 DW2 vs SAA

Earlier work has ruled out SAA as a model of the D-Wave devices for random Ising model problems [13], as well as for certain Hamiltonians with suppressed and enhanced ground states for which quantum annealing and SAA give opposite predictions [68, 34]. The observation made above, that the DW2 success probabilities have nearly reached their asymptotic values, suggests that for the problems studied here the DW2 device may perhaps be described as a thermal annealer. While we cannot conclude on the basis of ground state probabilities alone that the DW2 state distribution has reached the Gibbs state, we can compare the ground state distribution to that of SAA in the regime of an asymptotic number of sweeps, and attempt to identify an effective final temperature for the classical annealer that allows it to closely describe the DW2 distribution. We empirically determine $\beta_f = 5$ to be the final temperature for our SAA simulations that gives the closest match, and $S = 50{,}000$ (corresponding to 150ms, much larger than the DW2's $t_a = 20\mu$s) to be large enough for the SAA distribution to have become stationary (see Fig. 4.26a in Section 4.7 for results with additional sweep numbers confirming this). The result for the Euclidean distance measure is shown in Fig. 4.8 for the two extremal annealing times, and the distance is indeed small. To more closely assess the quality of the correlation we select $\alpha = 0.35$, the value that minimizes the Euclidean distance as seen in Fig. 4.8, and present the correlation plot in Fig. 4.7b. Considering the results for each size L separately, it is apparent that the correlation is good but also systematically skewed, i.e., for each problem size the data points approximately lie on a line that is not the diagonal line. Additional correlation plots are shown in Fig. 4.22 of Section 4.7.

As a final comparison we also extract the scaling coefficients $b(\alpha)$ and compare DW2 to SAA in Fig. 4.9. It can be seen that in terms of the scaling coefficients the DW2 and SAA results are essentially statistically indistinguishable. However, we stress that since the SAA number of sweeps is not optimized, one should refrain from concluding that the equal scaling observed for the DW2 and SAA rules out a DW2 speedup.



(a)



(b)

Figure 4.7: **Success probability correlations.** The results for all instances at $\alpha = 0.35$ are shown. Each datapoint is the success probability for the same instance, (a) for DW2 at $t_a = 20\mu$s and $t_a = 40\mu$s, (b) for DW2 at $t_a = 20\mu$s and SAA at 50000 sweeps and $\beta_f = 5$ [in dimensionless units, such that $\max(J_{ij}) = 1$]. Perfect correlation means that all data points would fall on the diagonal, and a strong correlation is observed in both cases. The data is colored by the problem size L and shows a clear progression from high success probabilities at small L to large success probabilities at small L. Qualitatively similar results are seen for $t_a = 20\mu$s vs $t_a = 40\mu$s at all α values, and for DW2 vs SAA at intermediate α values (see Figs. 4.21 and 4.22 in Section 4.7).

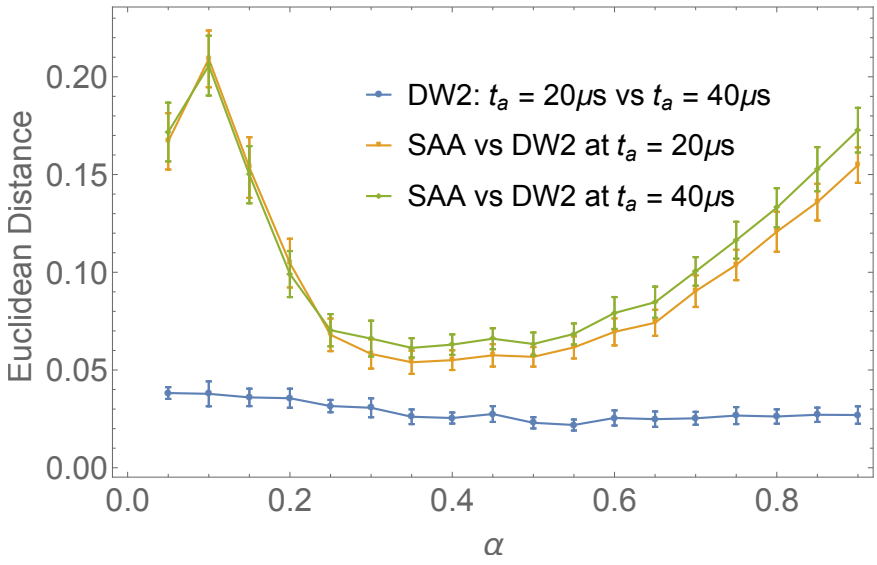


Figure 4.8: **Correlation between the DW2 data for two different annealing times and SAA.** Plotted is the normalized Euclidean distance $D(\vec{p}_1, \vec{p}_2)$ for $\vec{p}_1$ and $\vec{p}_2$ being, respectively, the ordered success probability for DW2 at $t_a = 20\mu$s and $t_a = 40\mu$s (blue circles), DW2 at $t_a = 20\mu$s and SAA (yellow squares), DW2 at $t_a = 40\mu$s and SAA (green diamonds). For comparison, the Euclidean distance between two random vectors with elements $\in [0, 1]$ is ~ 0.4. SAA data is for 50,000 sweeps and $\beta_f = 5$. The correlation with SAA degrades slightly for $t_a = 40\mu$s. Error bars represent 2σ confidence intervals and were computed using bootstrapping (see Appendix 4.6.2 for details). In each comparison, to construct $\vec{p}_1$ and $\vec{p}_2$ we fixed α and used half the instances (for bootstrapping purposes) for $L \in [2, 8]$.

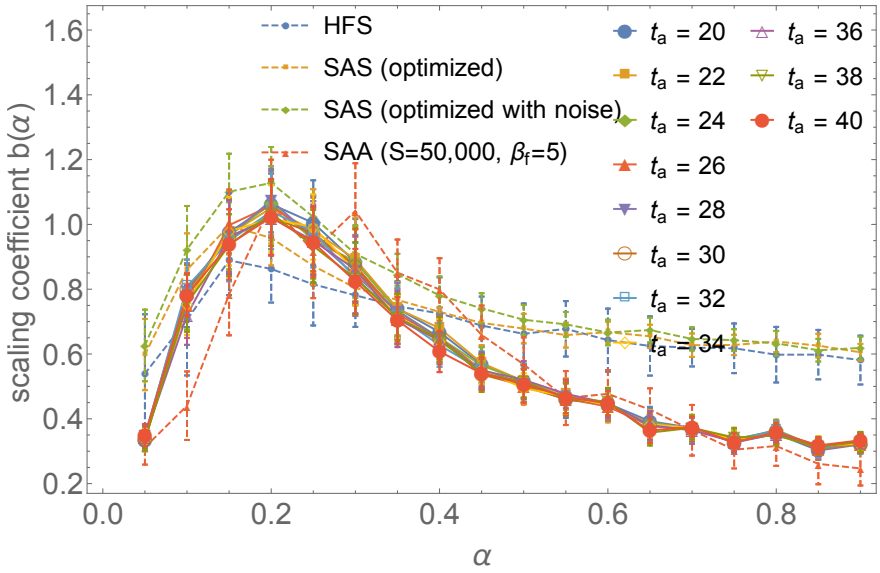


Figure 4.9: **Scaling coefficients of the number of runs.** Plotted here is $b(\alpha)$ [Eq. (4.6)] for the DW2 at all annealing times (overlapping solid lines and large symbols), for the HFS algorithm, for SAS with an optimal number of sweeps, for SAS with noise and an optimal number of sweeps, and for SAA with a large enough number of sweeps that the asymptotic distribution has been reached at $\beta_f = 5$. The scaling coefficients of HFS and of optimized SAS each set an upper bound for a DW2 speedup against that particular algorithm. In terms of the scaling coefficient the DW2 result is statistically indistinguishable (except at $\alpha = 0.1$) from SAA run at $S = 50{,}000$ and $\beta_f = 5$. The coefficients shown here are extracted from fits with $L \geq 4$ (see Fig. 4.25a in Section 4.7). Error bars represent 2σ confidence intervals.

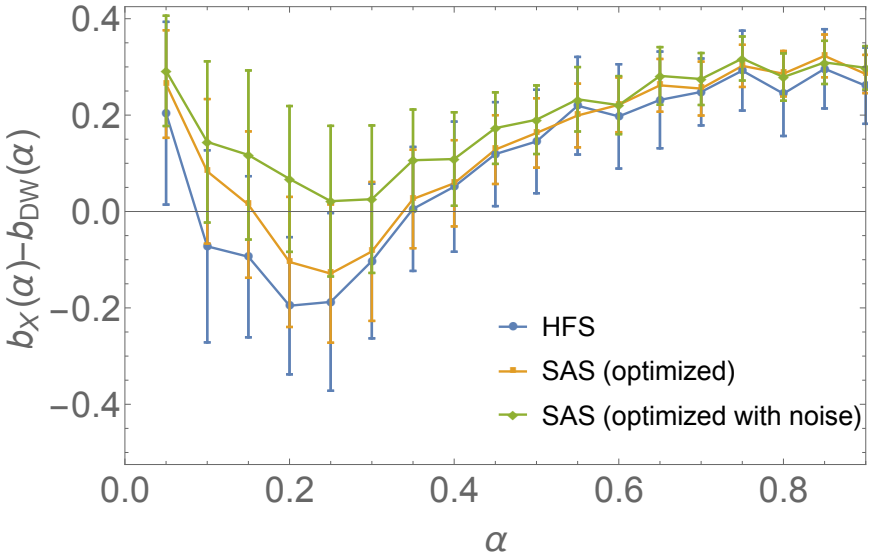


Figure 4.10: **Difference between the scaling coefficients.** Plotted here is the difference between the scaling coefficients in Fig 4.9, $b_{\mathrm{X}}(\alpha) - b_{\mathrm{DW2}}(\alpha)$, where X denotes the HFS algorithm, SAS with an optimal number of sweeps, SAS with noise and an optimal number of sweeps, or SAA with $S = 50,000$ and $\beta_f = 5$. When the difference is non-positive there can be no speedup since optimizing t_a can only increase $b_{\mathrm{DW2}}(\alpha)$; conversely, when the difference is positive a speedup is still possible, i.e., not accounting for the error bars, for $\alpha \leq 0.05$ and $\alpha \geq 0.4$ for HFS, and for $\alpha \leq 0.15$ and $\alpha \geq 0.35$ for SAS without noise. These ranges shrink if the error bars are accounted for, but notably, for most α values SAS with noise does not disallow a limited speedup, suggesting that control noise may play an important factor in masking a DW2 speedup. Error bars represent 2σ confidence intervals.

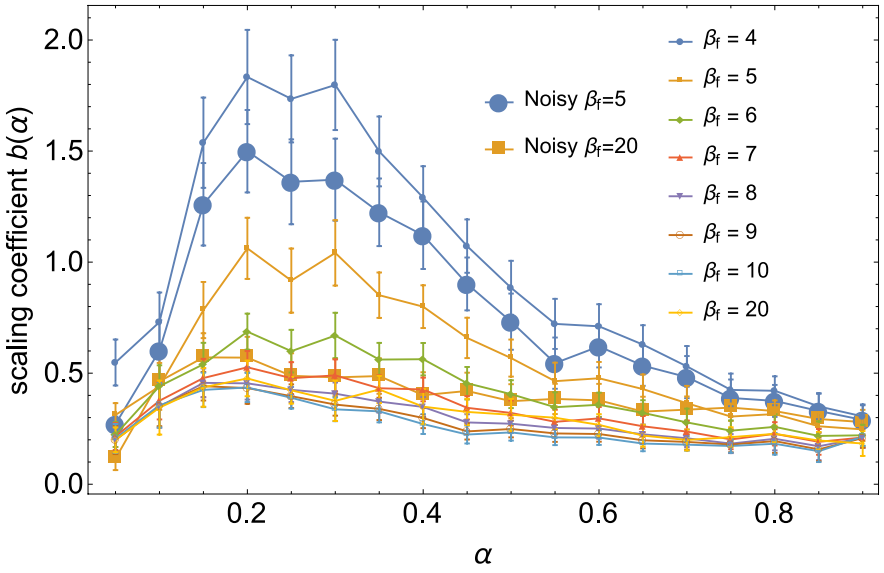


Figure 4.11: **Scaling of SAA at different final temperatures.** SAA is run at $S = 50{,}000$ and various final inverse temperatures. The peak at $\alpha \sim 0.2$ remains a robust feature. The data marked "Noisy" is with 5% random noise added to the couplings J_{ij}.

4.5 Discussion

In this study we proposed and implemented a method for generating problems with a range of hardness, tuned by the clause density, mimicking the phase structure observed for SAT problems. By comparing the DW2 device to a number of classical algorithms we delineated where there is no DW2 speedup and where it might still be possible, for this problem set. No advantage is observed for the low clause densities corresponding to the hardest optimization problems, but a speedup remains possible for the higher clause densities. In this sense these results are more encouraging than for the random Ising problems studied in Ref. [2], where only the lowest percentiles of the success probability distribution allowed for the possibility of a DW2 speedup. In our case there is in fact a slight improvement for the higher percentiles (see Fig. 4.20 in Section 4.7). In the same sense our findings are also more encouraging than recent theoretical results showing that quantum annealing does not provide a speedup relative to simulated annealing on 2SAT problems with a unique ground state and a highly degenerate first excited state [122].

The close match between the DW2 and SAA scaling coefficients seen in Fig. 4.9 suggests that SAA in the regime of an asymptotic number of sweeps can serve as a model for the expected performance of the D-Wave device as its temperature is lowered. Thus we plot in Fig. 4.11 the scaling coefficient for SAA at various final inverse temperatures, at a fixed (and still asymptotic) value of $S = 50{,}000$. Performance improves steadily as β_f increases, suggesting that SAA does not become trapped at 50,000 sweeps for the largest problem sizes we have studied (this may also indicate that even at the hardest clause densities these problems do not exhibit a positive-temperature spin-glass phase). We may infer that a similar behavior can be expected of the D-Wave device if its temperature were lowered.

An additional interesting feature of the fact that the asymptotic DW2 ground state probability observed here is in good agreement with that of a thermal annealer, is that it gives the ground state with a similar probability as expected from a Gibbs distribution. This result is consistent with the weak-coupling limit that underlies the derivation of the adiabatic Markovian master equation [123], i.e., it is consistent with the notion that the system-bath coupling is weak and *decoherence occurs in the energy eigenbasis* [124]. This rules out the possibility that decoherence occurs in the computational basis, as this would have led to a singular-coupling limit master equation with a ground state probability drawn not from the Gibbs distribution but rather from a uniform distribution, if the single-qubit decoherence time is much shorter than the total annealing time [123]. This is important, as the weak-coupling limit is compatible with decoherence between eigenstates with different energies while maintaining ground state coherence, a necessary condition for a speedup via quantum annealing. In contrast, in the singular-coupling limit no quantum effects survive and no quantum speedup of any sort is possible.

We note that an important disadvantage the DW2 device has over all the classical algorithms we have compared it with, is control errors in the program-

ming of the local fields and couplings [125, 126, 127, 128]. As shown in Section 4.7 (Fig. 4.27) for those interested, many of the rescaled couplings J_{ij} used in our instances are below the single-coupler control noise specification, meaning that with some probability the DW2 is giving the right solution to the wrong problem. We are unable to directly measure the effect of such errors on the DW2 device, but their effect is demonstrated in Figs. 4.10 and 4.11, where both SAS and SAA with $\beta_f = 5, 20$, respectively, are seen to have substantially increased scaling coefficients after the addition of noise that is comparable to the control noise in the DW2 device. In fact, the DW2 scaling coefficient is smaller than the scaling coefficient of optimized SAS with noise over almost the entire range of α, suggesting that a reduction in such errors will extend the upper bound for a DW2 speedup against SAS over a wider range of clause densities. The effect of such control errors can be mitigated by improved engineering, but also emphasizes the need for the implementation of error correction on putative quantum annealing devices. The beneficial effect of such error correction has already been demonstrated experimentally [94, 129] and theoretically [130], albeit at the cost of a reduction in the effective number of qubits and reduced problem sizes.

To summarize, we believe that at least three major improvements will be needed before it becomes possible to demonstrate a conclusive (limited or potential) quantum speedup using putative quantum annealing devices: (1) harder optimization problems must be designed that will allow the annealing time to be optimized, (2) decoherence and control noise must be further reduced, and (3) error correction techniques must be incorporated. Another outstanding challenge is to theoretically design optimization problems that can be unequivocally shown to benefit from quantum annealing dynamics.

Finally, the methods introduced here for creating frustrated problems with tunable hardness should serve as a general tool for the creation of suitable benchmarks for quantum annealers. Our study directly illustrates the important role that frustration plays in the optimization of spin-glass problems for classical algorithms as well as for putative quantum optimizers. It is plausible that different, perhaps more finely-tuned choices of clauses to create novel types of benchmarks, may be used to establish a clearer separation between the performance of quantum and classical devices. These may eventually lead to the demonstration of the coveted experimental annealing-based quantum speedup.

Note added. Work on problem instances similar to the ones we studied here, but with a range of couplings above the single-coupler control noise specification, appeared shortly after our preprint [131]. The DW2 scaling results were much improved, supporting our conclusion that such errors have a strong detrimental effect on the performance of the DW2.

Finally, I'll point out a few places where the study in this chapter diverges from the discussion of chapter 3. In this study, I had not yet come up with or hit upon the notion of optional stopping, and thus it was not used. Secondly, while I did use the Beta distribution for estimation of the probability of success, as detailed below for enterprising readers. Also, as we were not interested in the TTS of individual instances, essentially at all, we neglected to gauge average.

In retrospect, I would not do this today, or if I did I would have taken a more rigorous approach to estimating my posterior density for the various percentiles of problem hardness (here we used the classical bootstrap over the problem ensembles). In essence, this study represents a kind of halfway house between the period from chapter 2, where we were groping in the dark after a fashion doing blind classical bootstraps of every single parameter, and the more rigorous approach discussioned in chapter 3. The analysis in the next chapter is more rigorous and inspired by the analysis in the previous chapter, but has a number of different properties due to the different parameter of interest (not TTS, but classifier accuracy).

Acknowledgements for this chapter

Part of the computing resources were provided by the USC Center for High Performance Computing and Communications. This research used resources of the Oak Ridge Leadership Computing Facility at the Oak Ridge National Laboratory, which is supported by the Office of Science of the U.S. Department of Energy under Contract No. DE-AC05-00OR22725. I.H. and D.A.L. acknowledge support under ARO grant number W911NF-12-1-0523. The work of J.J., T.A. and D.A.L. was supported under ARO MURI Grant No. W911NF-11-1-0268, and the Lockheed Martin Corporation. M.T. and T.F.R. acknowledges support by the Swiss National Science Foundation through the National Competence Center in Research NCCR QSIT, and by the European Research Council through ERC Advanced Grant SIMCOFE. We thank Mohammad Amin, Andrew King, Catherine McGeoch, and Alex Selby and for comments and discussions. I.H. would like to thank Gene Wagenbreth for assistance with the implementation of solution-enumeration algorithms. J.J. would like to thank the developers of the Julia programming language [132], which was used extensively for data gathering and analysis.

4.6 Methods

In this section I describe various technical details regarding our methods of data collection and analysis.

4.6.1 Experimental details

For each clause density α and problem size N (or C_L) we generated 100 instances, for a total of 12,600 instances. We performed approximately $990000\mu s/t_a$ annealing runs (experiments) for each problem instance and for each annealing time $t_a \in [20, 40]\mu$s, in steps of 2μs, for a total of more than 10^{10} experiments. No gauge averaging [13] was performed because we are not concerned with the timing data for a single instance but a collection of instances, and the variation over instances is larger than the variation over gauges.

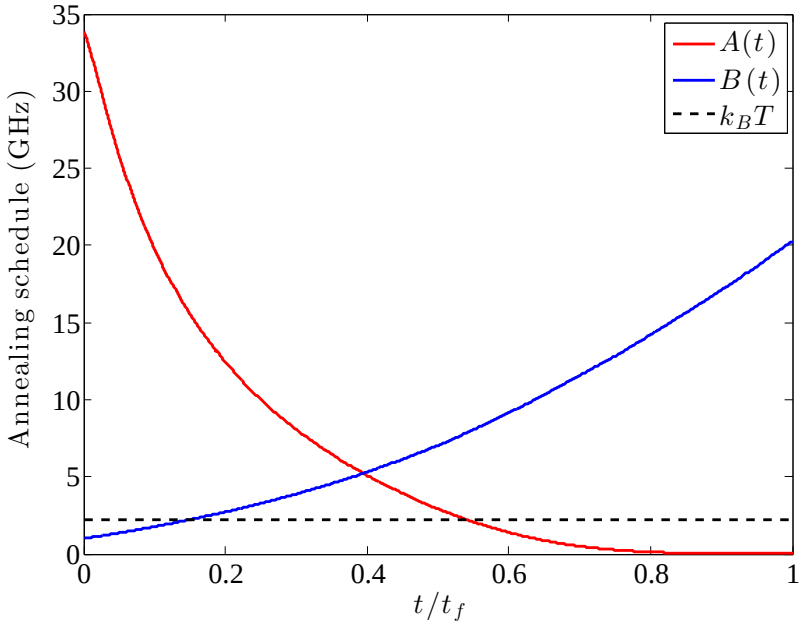


Figure 4.12: **Annealing schedule of the DW2.** The annealing curves $A(t)$ and $B(t)$ are calculated using rf-SQUID models with independently calibrated qubit parameters. Units of $\hbar = 1$. The operating temperature of 17mK is also shown.

Solver details

HFS runs in a serial manner on a single core of a CPU. Since we allow the DW2 to use $\mathcal{O}(N)$ resources, we should also parallelize the HFS algorithm. In principle, HFS proceeds in a series of steps — at each step, one shears off each of the leaves of the tree (i.e., the outermost vertices) and then proceeds to the next step until the tree has collapsed to a point. Each of the leaves can be eliminated separately, however each elimination along a particular branch is dependent on the previous eliminations. As a result, we must perform between $L + 2$ and $\frac{3}{2}L + 2$ (average $\frac{5}{4}L + 2$) irreducibly serial operations when reducing the trees on an $L \times L$ unit cell Chimera graph (depending on which of the trees we use and the specifics for how we do the reduction). Since we deal only with square graphs and are concerned with asymptotic scaling, we ignore the constant steps and use $\frac{5}{4}L$ basic operations per tree (see Sec. 4.6.2 for additional details).

HFS and SA are run 10^4 times each to estimate probability of success, while SQA is run 1000 times, and DW2 is run with 1000 samples each. We used the DW2 annealing schedule in Fig. 4.12, and the temperature was kept constant at 10.56mK when performing simulations with SVMC/SQA

4.6.2 Error estimation

As discussed above, we performed primarily classical bootstraps for error estimates, though our probabilities of success were sampled from the correct posterior distribution for each instance.

Annealers

If we have M instances in our set of interest, then we resample with replacement a "new" set of instances $\{\mathcal{I}_i\}_j$, also of length M, from our set $\mathcal{I}$. For each instance $\mathcal{I}_{i,j}$ from this new set we sample a value for its probability from its corresponding distribution $\beta_{\mathcal{I}_{i,j}}$ to get a set of probabilities $\{p_{i,j}\}$. We then calculate whatever function $F_j = f(\{p_{i,j}\})$ we wish on these probabilities, e.g., the median over the set of instances. We repeat this process a large number of times (in our case, 1000), to have many values of our function $\{F_j\}$. We then take the mean and standard deviation over the set $\{F_j\}$ to get a value $\bar{F}$ and a standard deviation σ_F, which form our value and error bar for that size and clause density. This is essentially a classicla bootstrap over instances, but with a proper Bayesian account of the probability of success for each instance (as no guage averaging was performed, we will here neglect gauges as a nuisance parameter).

When we take the ratio of two algorithms A and B, for each pair of corresponding data points for the two algorithms (i.e., each size and clause density) we characterize each point with a normal distributions $\mathcal{N}(\mu_A, \sigma_A)$ and $\mathcal{N}(\mu_B, \sigma_B)$. We then resample from each distribution a large number of times (1000) to get two sets of values for the function we wish to plot F: $\{F_A\}$ and $\{F_B\}$. We take the ratio of the corresponding elements in the two sets $S_i = F_{A,i}/F_{B,i}$ and take the mean and standard deviation of the set $\{S_i\}$ to get our data point and error bar for the ratio.

HFS algorithm

For the HFS algorithm, time to solution for each problem is computed as the mean number of trees per sample multiplied by $0.625\mu s \times L$ for a C_L problem. The number 0.625 is chosen to approximate the times on a standard laptop, and the scaling with L is due to the fact that, in a parallel setting, the number of steps to reduce a tree is linear in L (exactly, it is $\frac{5}{4}L+2$, but the 2 serves only to mask asymptotic scaling and is thus not included in our TTS or speedup plots or the scaling analysis).

Euclidean distance

In order to generate the error bars in Fig. 4.8, instead of calculating the Euclidean distance over the total number of instances at a given α (700 total), we calculated the Euclidean distance over half the number of instances (350 total). We were then able to perform 100 bootstraps over the instances, i.e., we picked

350 instances at random for each Euclidean distance calculation. For each bootstrap, we calculated the Euclidean distance, and the data points in Fig. 4.8 are the mean of the Euclidean distances over the bootstraps, while the error bars are twice the standard deviation of the Euclidean distances.

Difference in slope

In order to generate the error bars in Fig. 4.10, we used the data points and the error bars in Fig. 4.9 as the mean and twice the standard deviation of a Gaussian distribution. We then took 1000 samples from this distribution and calculated their differences. The means of the differences are the data points in Fig. 4.10, and the error bars are twice the standard deviation of the differences.

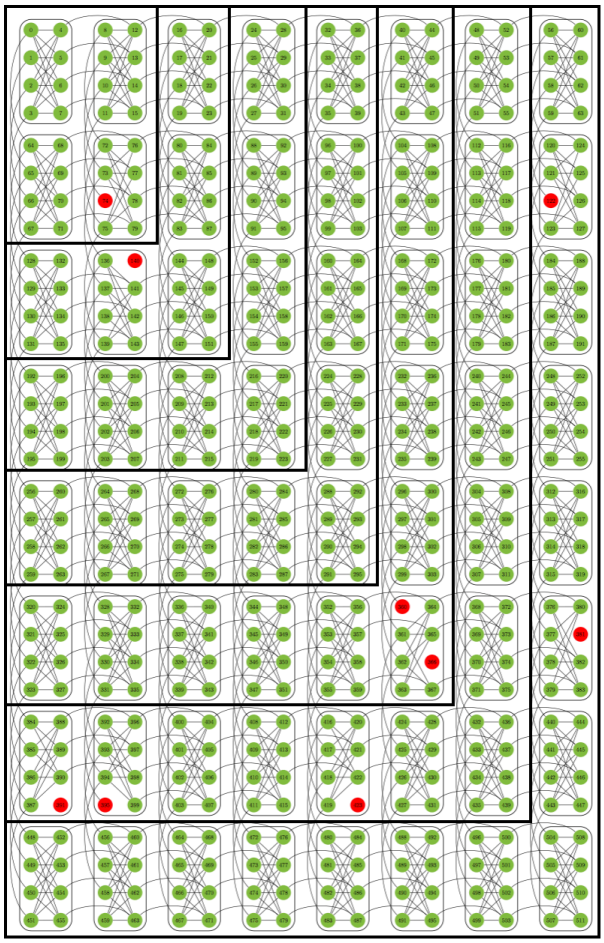


Figure 4.13: **The DW2 Chimera graph.** The qubits or spin variables occupy the vertices (circles) and the couplings J_{ij} are along the edges. Of the 512 qubits, 503 were operative in our experiments (green circles) and 9 were not (red circles). We utilized subgraphs comprising $L \times L$ unit cells, denoted C_L, indicated by the solid black lines. There were $31, 70, 126, 198, 284, 385, 503$ qubits in our $C_2, \ldots, C_8$ graphs, respectively.

4.7 Additional results

In this section we collect additional results in support of the main text of the chapter, for those interested.

4.7.1 Degeneracy-hardness correlation

It is known that a non-degenerate ground state along with an exponentially (in problem size) degenerate first excited state leads to very hard SAT-type optimization problems [133]. Here we focus on the ground state degeneracy and ask whether it is correlated with hardness. We show the ground state degeneracy in Fig. 4.14. It decays rapidly as α grows, and for sufficiently large α (that depends on the problem size) the ground state found is unique (up to the global $\mathbf{Z}_2$ symmetry). This suggests that degeneracy is not necessarily correlated with hardness, since in the main text we found that hardness peaks at $\alpha \sim 0.25$. To this test directly we restrict ourselves to $L = 8$ and $\alpha = 0.4$. We then bin the 100 instances at this α according to their degeneracy and study their median TTS using the HFS algorithm. We show the results in Fig. 4.15, where we see no correlation between degeneracy and hardness for a fixed α. We find a similar result when we use the success probability of the DW2 as the metric for hardness.

4.7.2 Additional easy-hard-easy transition plots

The universal nature of the scaling behavior can be seen in Fig. 4.16, complementing Fig. 4.2 with results for the 25th and 75th percentiles respectively. The peak in the TTS near $\alpha = 0.2$ is a feature shared by all the solvers we considered.

4.7.3 Optimality plots

The absence of an optimal DW2 annealing time was discussed in detail in the main text, along with the optimality of the number of sweeps of the classical algorithms. Figure 4.17 illustrates this: a clear lower envelope is formed by the different curves plotted for SQA, SA, and SSSV, from which the optimal number of sweeps at each size can be easily extracted. Unfortunately no such envelope is seen for the DW2 results [Fig. 4.17a], leading to the conclusion that $t_a = 20\mu$s is suboptimal.

A complementary perspective is given by Fig. 4.18, where we plot the TTS as a function of the number of sweeps, for a fixed problem size $L = 8$. It can be seen that the classical algorithms all display a minimum for each clause density. The DW2 curves all slope upward, suggesting that the minimum lies to the left, i.e., is attained at $t_a < 20\mu$s. We note that in an attempt to extract an optimal annealing time we tried to fit the DW2 curves to various functions inspired by the shape of the classical curves, but this proved unsuccessful since the DW2 curves essentially differ merely by a factor of t_a, as discussed in the main text.

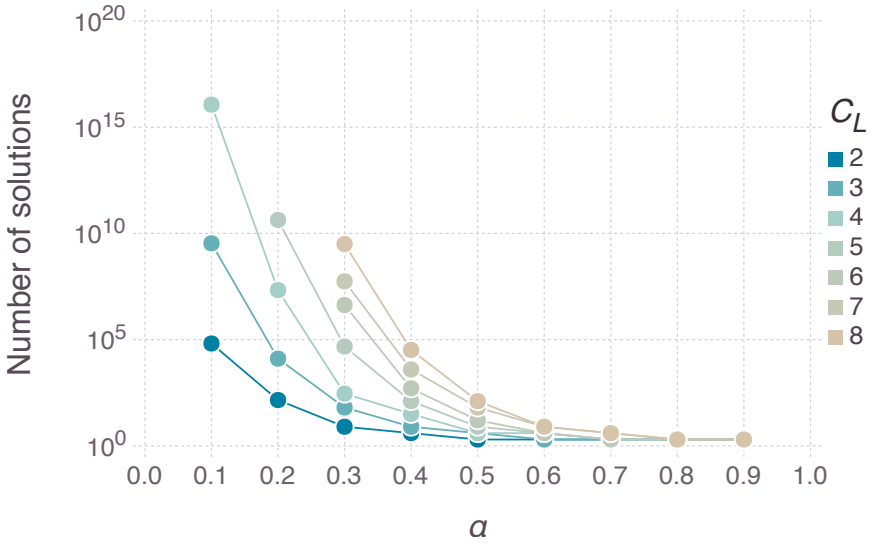


Figure 4.14: **Ground state degeneracy**. Number of unique solutions as a function of clause density for different Chimera subgraph sizes C_L. As the clause density is increased, the number of unique solutions found decreases to one (up to the global bit flip symmetry). Shown is the median degeneracy, i.e., we sort the degeneracies of the 100 instances for each value of L and α, and find the median. Our procedure counts the degenerate solutions and stops when it reaches 10^5 solutions. If the median has 10^5 solutions then we assume that not all solutions were found and hence the degeneracy for that value of L and α is not plotted. These are solutions on the used qubits (e.g., there are many instances for each α at $L = 8$ that use < 503 qubits); to account for the n_{uq} unused qubits we multiply the degeneracy by $2^{n_{uq}}$.

We can take this a step further and use these optimal number of sweeps results to demonstrate that the problems we are considering here really are hard. To this end we plot in Fig. 4.19 the optimal number of sweeps as a function of problems size for SAA. We observe that certainly for the smaller clause densities the optimal number of sweeps s_{opt} appears to scale exponentially in L, which indicates that the TTS (which is proportional to s_{opt}) will also grow exponentially in L.

4.7.4 Additional speedup ratio plots

To test whether our speedup ratio results depend strongly on the percentile of the success probability distribution, we recreate Fig. 4.6 for the 25th and

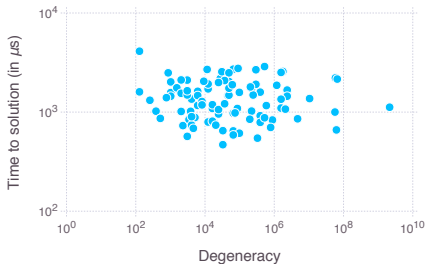


Figure 4.15: Scatter plot of the TTS for the HFS algorithm and the degeneracy at $L = 8$ and $\alpha = 0.4$ (100 instances total). Even though there is a wide range of degeneracy over several orders of magnitude, we do not observe any trend in the TTS. The degeneracy accounts for the fact that some qubits are not coupled into the problem (e.g., if n qubits are not specified for that particular problem, then the degeneracy is 2^n times the directly counted degeneracy). The Pearson correlation coefficient is -0.046.

75th percentiles in Fig. 4.20. The results are qualitatively similar, with a small improvement in the speedup ratio relative to HFS at the higher percentile.

4.7.5 Additional correlation plots

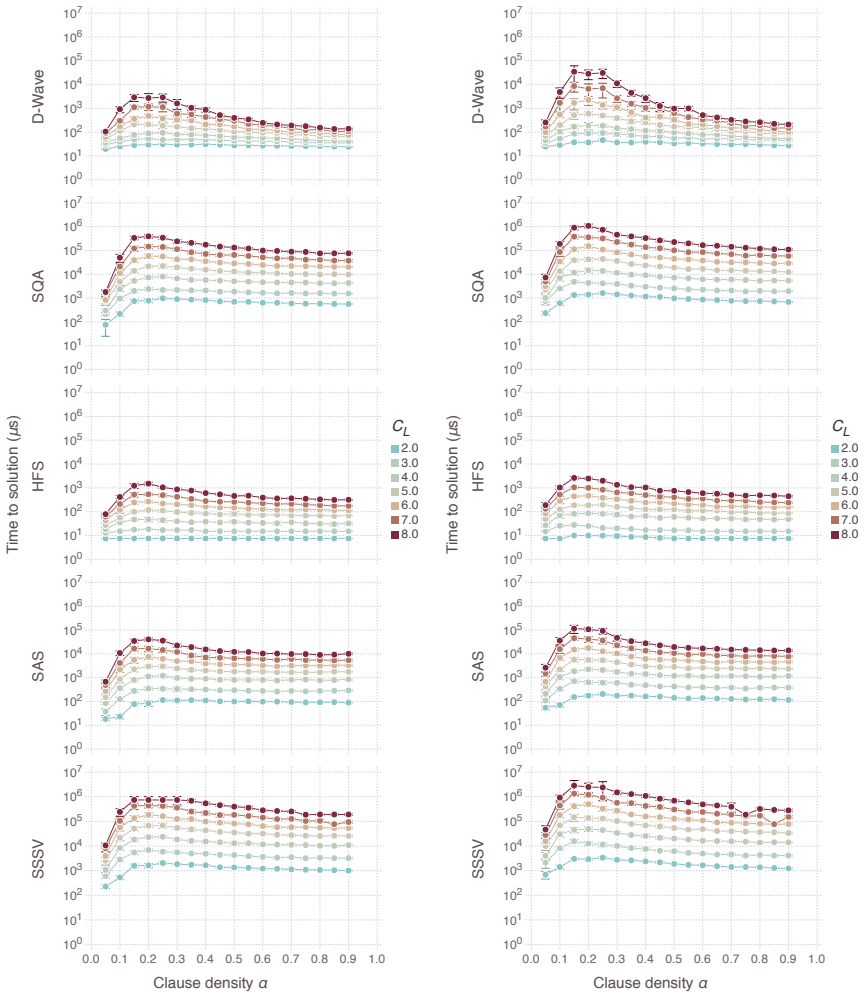
To complement Fig. 4.7, we provide correlation plots for both the DW2 against itself at $t_a = 20\mu$s and $t_a = 40\mu$s (Fig. 4.21), the DW2 vs SAA (Fig. 4.22), the DW2 vs SQAA (Fig. 4.23), and SAA vs SQAA (Fig. 4.24. The DW2 against itself displays an excellent correlation at all clause densities, while the DW2 vs SAA and DW2 vs SQAA continues to be skewed at low and high clause densities. Recall that Fig. reffig:Euclid-dist provides an objective Euclidean distance measure that is computed using all problem sizes and depends only on the clause density.

4.7.6 Additional scaling analysis plots

We provide a few additional plots in support of the scaling analysis presented in the main text.

Figure 4.25a shows the number of runs at different problem sizes and clause densities, and the corresponding least-squares fits. It can be seen that the straight lines fits are quite good. The slopes seen in this figure are the $b(\alpha)$ values for $t_a = 20\mu$s plotted in Fig. 4.9; the intercepts are plotted in Fig. 4.25b for all annealing times, and collapse nicely, just like the $b(\alpha)$.

Finally, Fig. 4.26a is a check of the convergence of SAA to its asymptotic scaling coefficient as the number of sweeps is increased from $5,000$ to 50,000. Convergence is apparent within the 2σ error bars.



(a) (b)

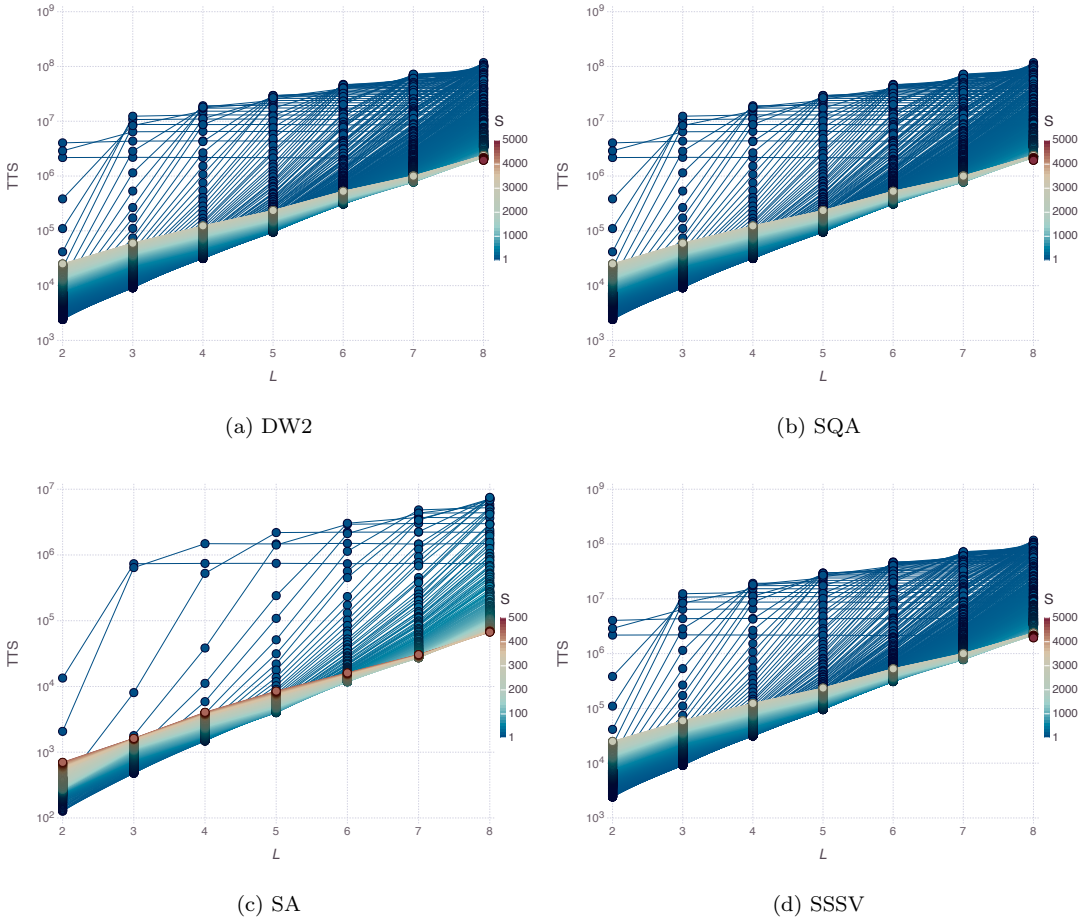
Figure 4.16: Comparison of the 25th (left) and 75th (right) percentiles of the TTS (log scale) for all algorithms as a function of clause density α. The different colors represent the different Chimera sizes tested. All solvers show a peak at the same density value of $\alpha \approx 0.2$.

4.7.7 SQAS vs SAS

In the main text we only considered SQA as an annealer since that is a more faithful representation of the DW2. Here we present a comparison of SQA as a solver (SQAS), where we keep track of the lowest energy found during the entire anneal, with SAS. We present the scaling coefficient $b(\alpha)$ from Eq. (4.6) of these two solvers in Fig. 4.26b. SAS has a smaller scaling coefficient than SQAS for the large α values, but at small α values we cannot make a conclusive determination because of the substantial overlap of the error bars. We note that Ref. [23] reported that discrete-time SQA (the version used here) can exhibit a scaling advantage over SA but that this advantage vanishes in the continuous-time limit. We have not explored this possibility here.

4.7.8 Scale factor histograms

We analyze the effect of increasing clause density and problem size on the required precision of the couplings. Fig. 4.27 shows a trend of scaling factor increasing as α increases for fixed L, and as L increases for fixed α. Increased scaling factor has the effect of relatively amplifying control error and thermal effects in DW2, and can therefore contribute to a decline in performance for the larger problems studied. However, recall that the region where a speedup is possible according to our results is in fact that of high clause densities. Thus, whatever the effect of precision errors is, it does not appear to heavily impact the DW2's performance in the context of our problems. The same is true for SAS, since as can be seen in Fig. 4.10, the scaling coefficient is unaffected by the addition of noise when $\alpha \gtrsim 0.6$.



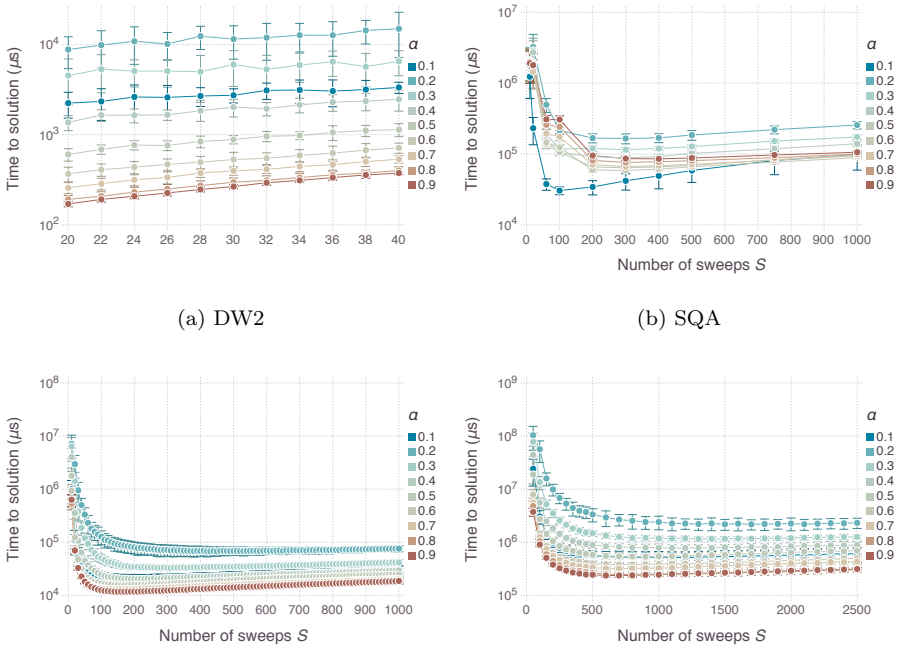
(a) DW2

(b) SQA

(c) SA

(d) SSSV

Figure 4.17: **Suboptimal annealing time and optimal sweeps for $\alpha = 0.2$.** Plotted is the TTS (log scale) as a function of size L for (a) the DW2, with all available annealing times, (b) SQA, (c) SA, and (d) SSSV, for many different sweep numbers. The lower envelope gives the scaling curves shown in Fig. 4.5 for $\alpha = 0.2$. The TTS curves flatten at high L for the following reason: each classical annealer was run N_X times ($N_{\mathrm{SA}} = N_{\mathrm{SSSV}} = 10^4$, $N_{\mathrm{SQA}} = 10^3$), and our β distribution is $\beta(0.5, N_X + 0.5)$ for 0 successes, which has an average value of $\sim 1/(2N_X)$. This reflects the (Bayesian) information acquired after N_X runs with 0 successes (one would not expect the probability to be 0). The flattening has no impact on the scale of the optimal number of sweeps.



(a) DW2

(b) SQA

(c) SA

(d) SSSV

Figure 4.18: TTS (log scale) for $L = 8$ as a function of number of sweeps for DW2, SQA, SA, and SSSV used to identify the optimal number of sweeps.

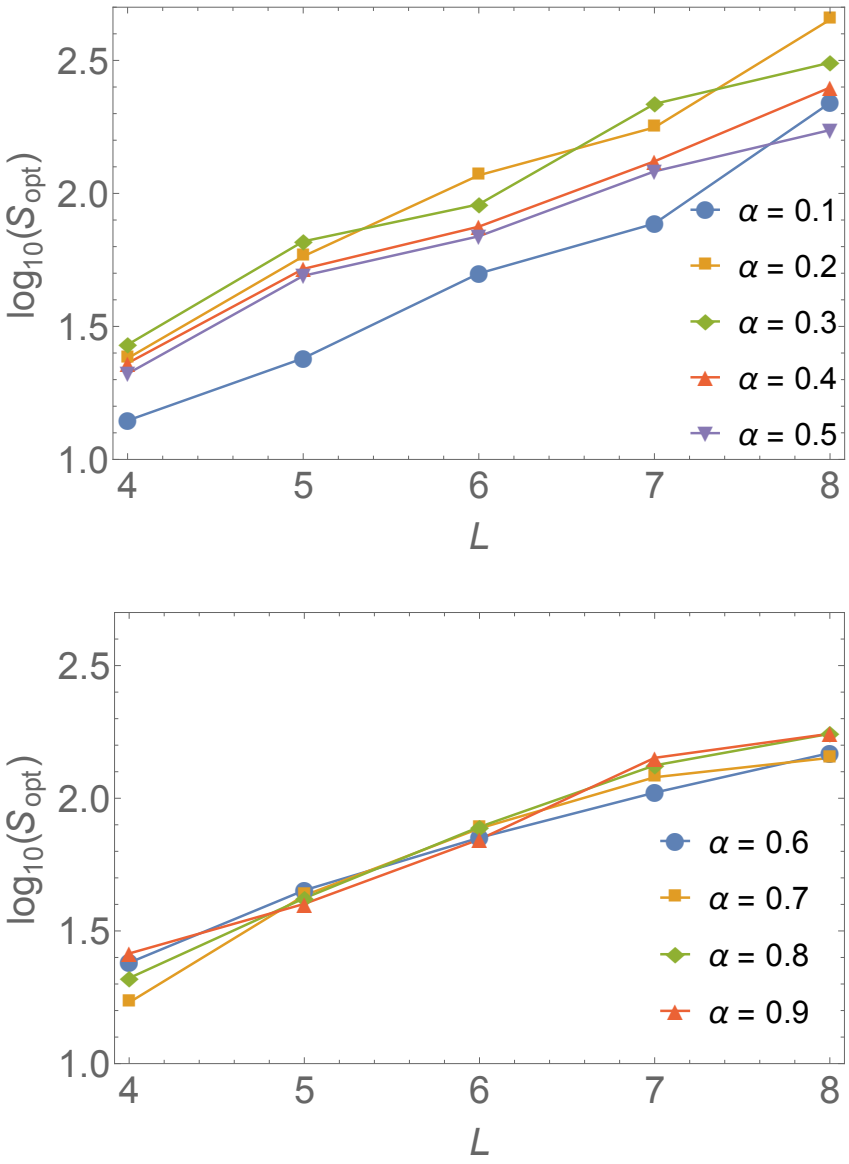
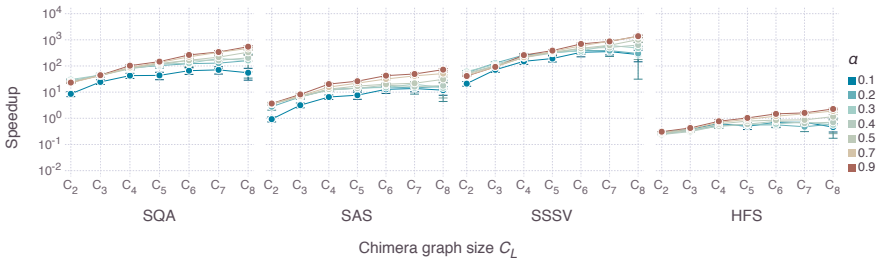
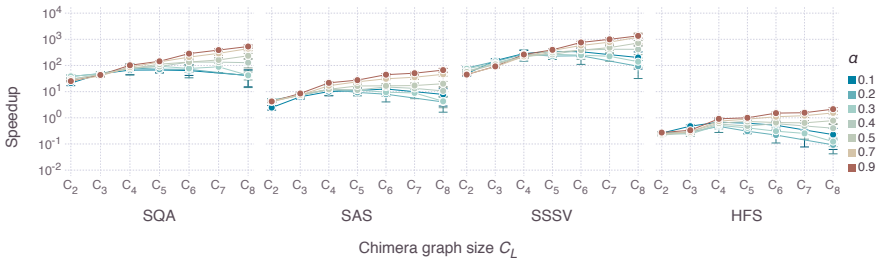


Figure 4.19: **Scaling of the SAA optimal number of sweeps.** The optimal number of sweeps is extracted for each L from Fig. 4.18. The scaling is roughly exponential for the smaller α values, and appears to be close to exponential for the larger α values. Lines are guides to the eye.



(a) 25th percentile



(b) 75th percentile

Figure 4.20: The speedup ratios (log scale) for the 25th and 75th percentile of time-to-solution as a function of system size for various α. The different colors denote a representative sample of clause densities.

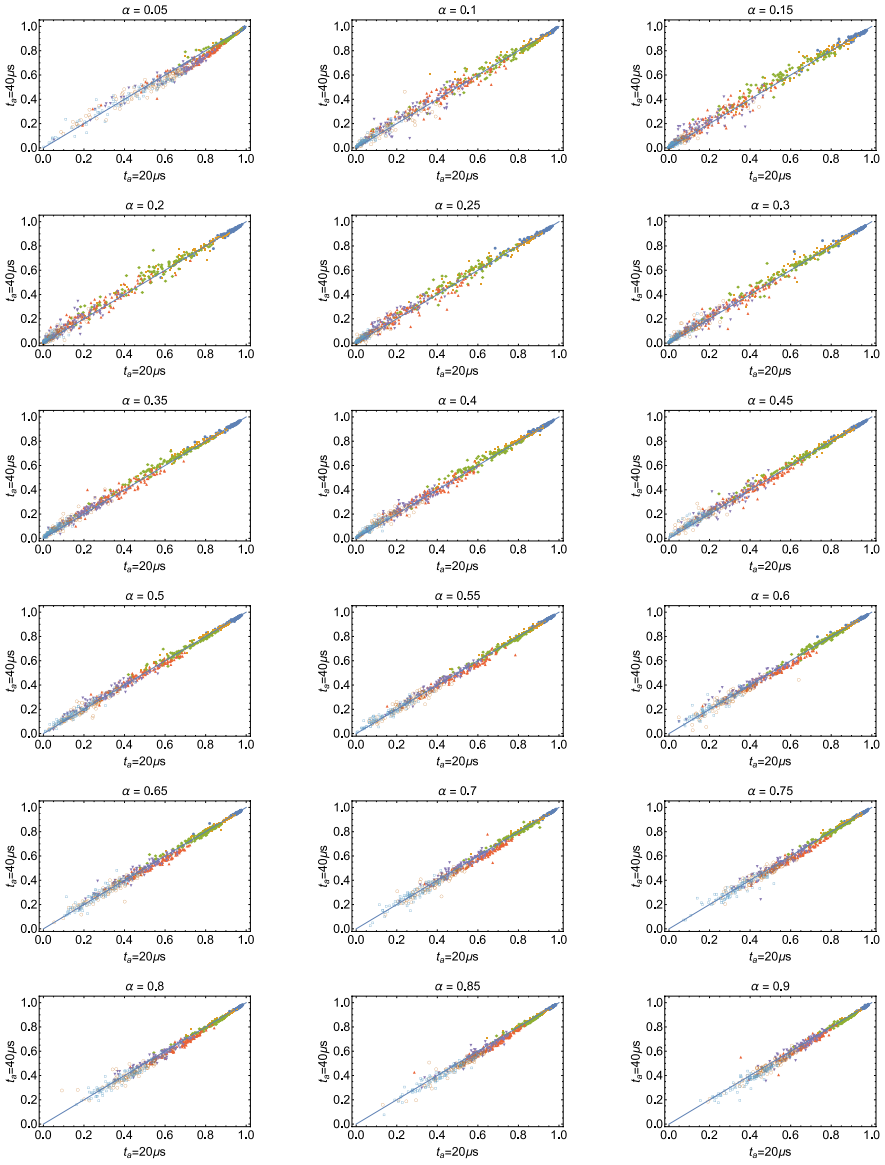


Figure 4.21: **Success probability correlations.** The results for all instances at all α values are shown for the DW2 data at $t_a = 20\mu$s and $t_a = 40\mu$s. This complements Fig. 4.7; the color scheme corresponds to different sizes L as in that figure.

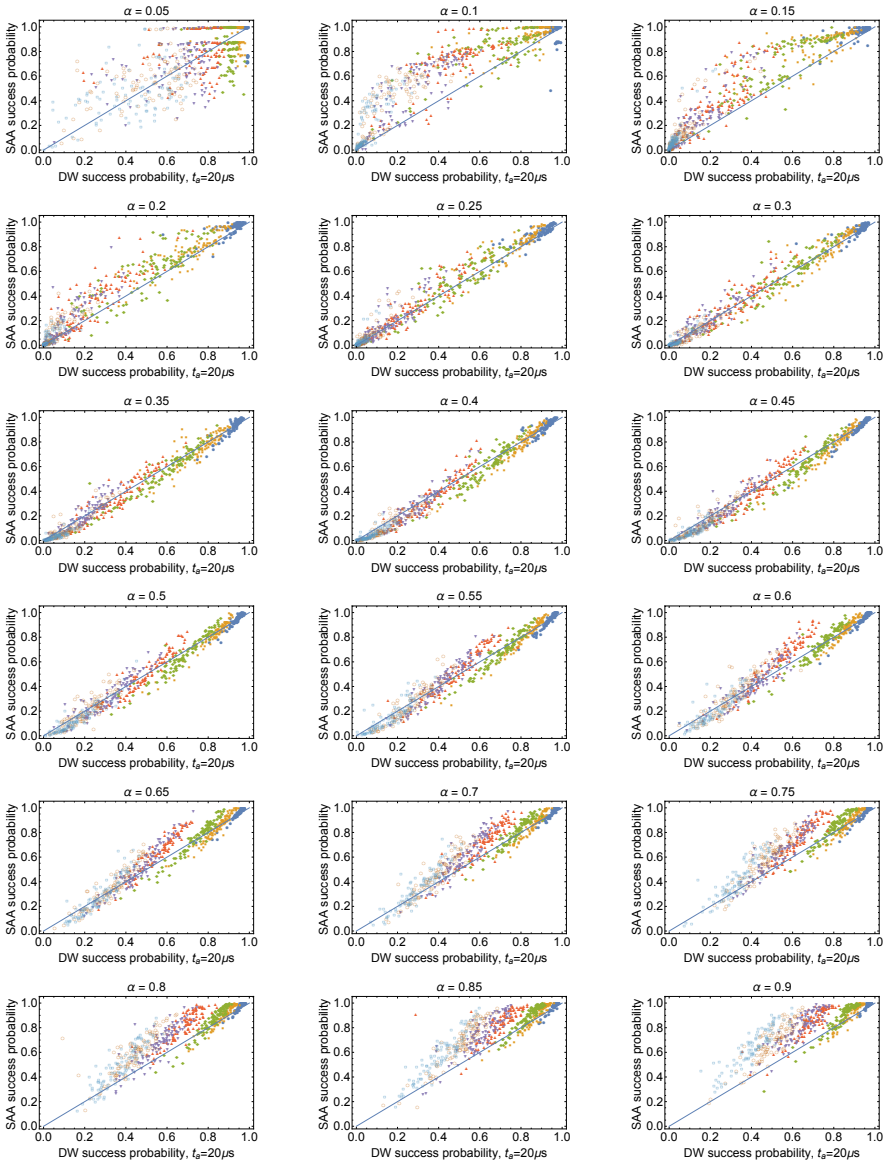


Figure 4.22: **Success probability correlations.** The results for all instances at all α values are shown for the DW2 data at $t_a = 20\mu$s and SAA with $S =$ 50,000 and $\beta_f = 5$. This complements Fig. 4.7; the color scheme corresponds to different sizes L as in that figure. The correlation gradually improves from poor for the lowest clause densities to strong at $\alpha = 0.35$, then deteriorates again.

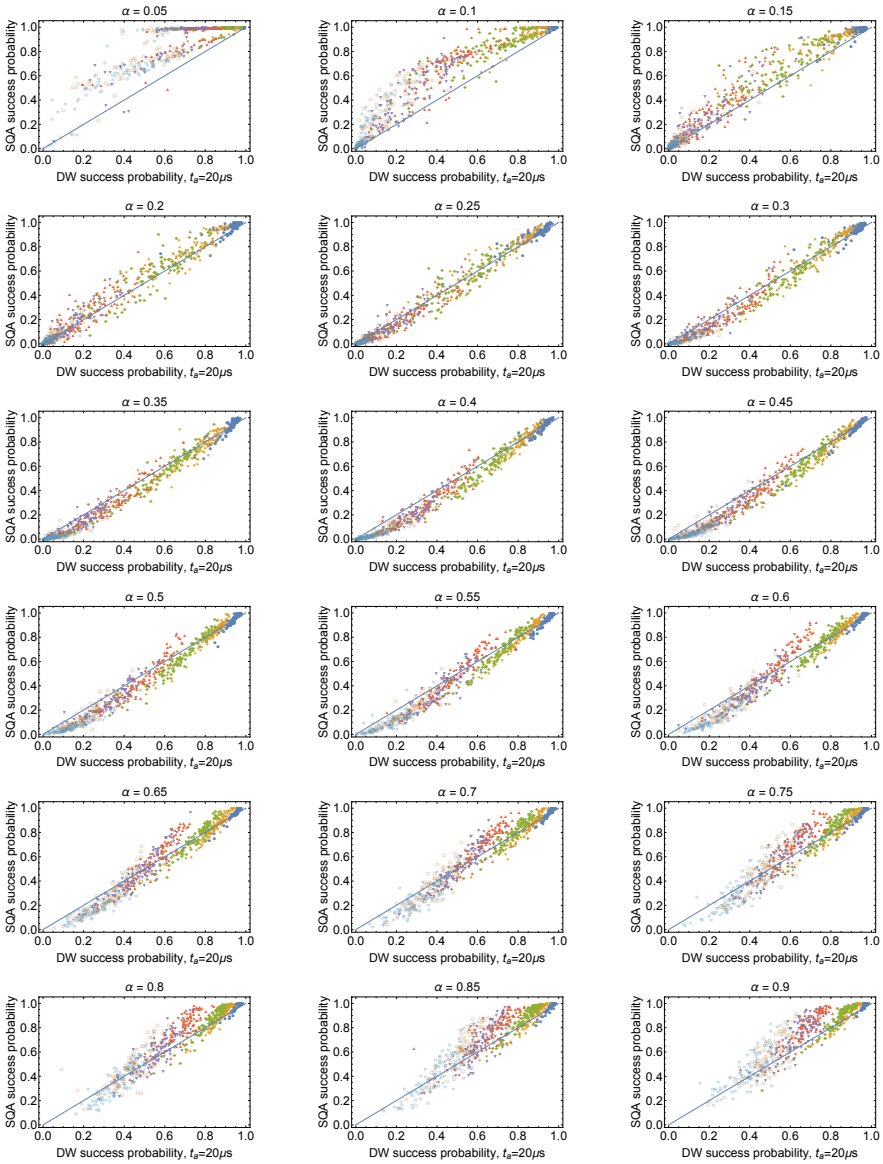


Figure 4.23: **Success probability correlations.** The results for all instances at all α values are shown for the DW2 data at $t_a = 20\mu$s and SQAA with $S = 10{,}000$ and $\beta = 5$. This complements Fig. 4.7; the color scheme corresponds to different sizes L as in that figure. The correlation gradually improves from poor for the lowest clause densities to strong at $\alpha = 0.35$, then deteriorates again.

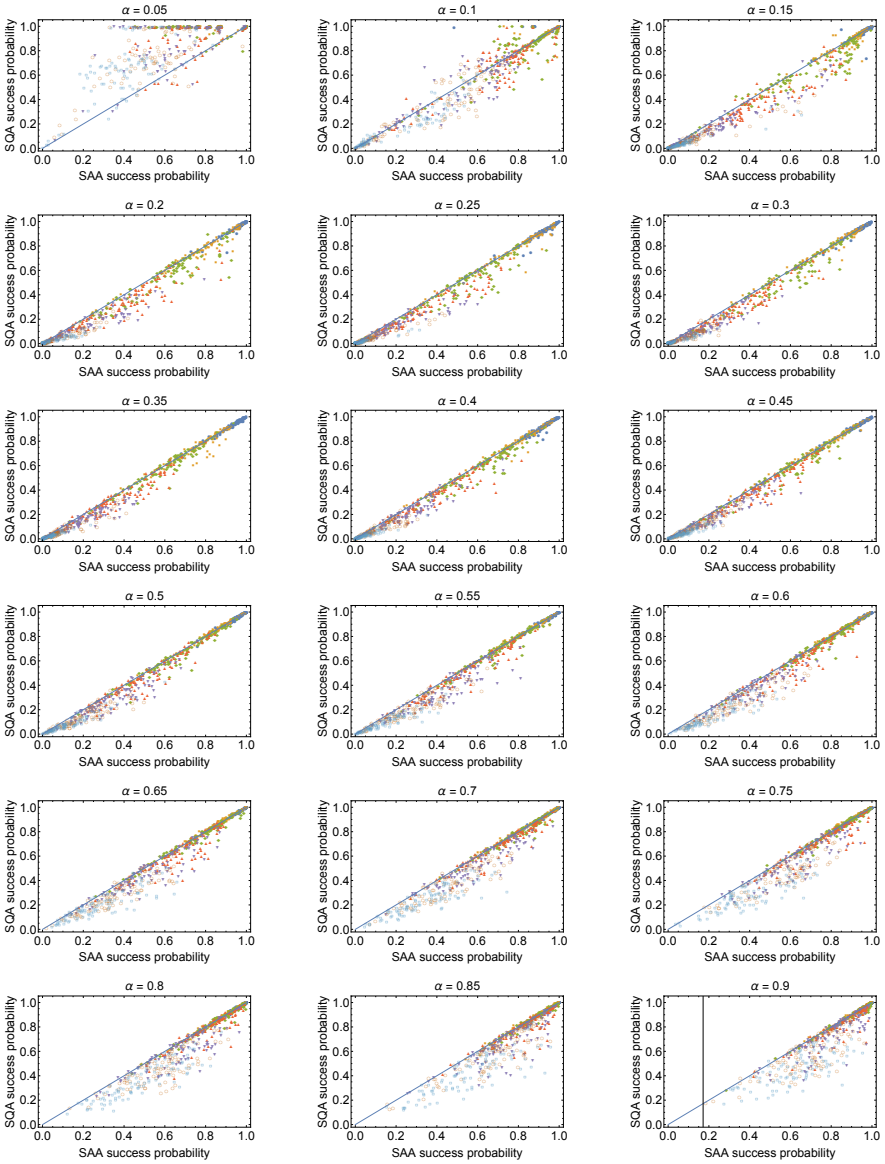
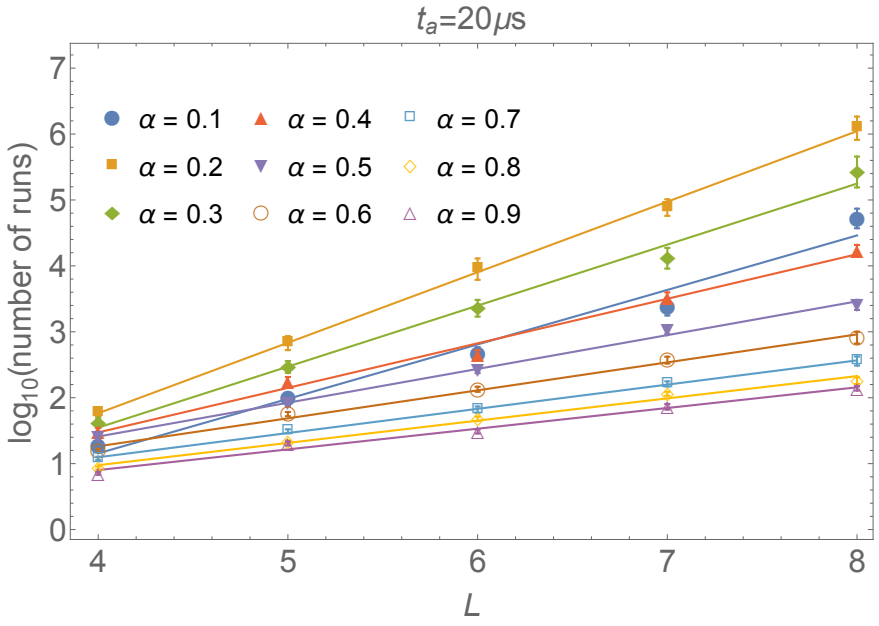
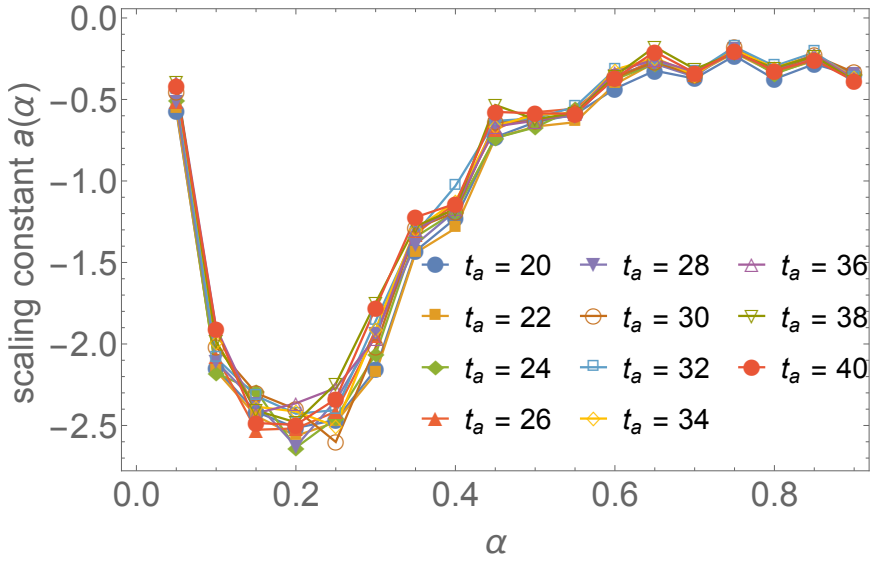


Figure 4.24: **Success probability correlations.** The results for all instances at all α values are shown for the SQAA data at $S = 10{,}000$ and $\beta = 5$ and SAA with $S = 50{,}000$ and $\beta_f = 5$. This complements Fig. 4.7; the color scheme corresponds to different sizes L as in that figure. The correlation gradually improves from poor for the lowest clause densities to strong at $\alpha = 0.35$, then deteriorates again. Note that we did not attempt to optimize the correlations between the two methods.

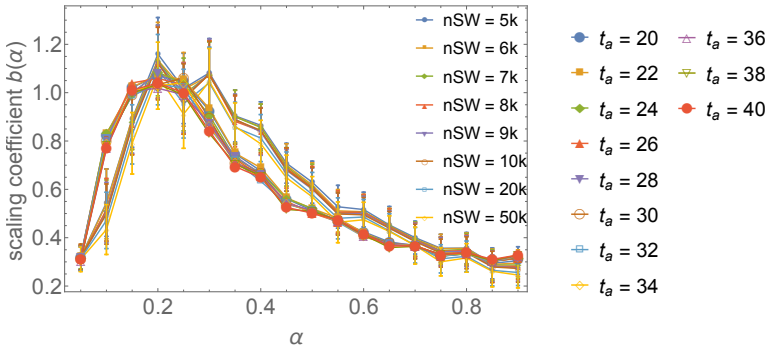


(a)

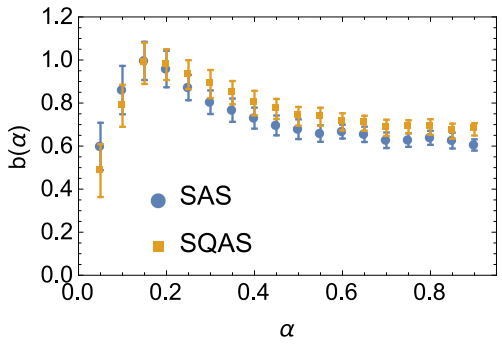


(b)

Figure 4.25: **Exponential fits to the DW2 number of runs.** In accordance with Eq. (4.6), the least-squares linear fits to $\log[r(L, \alpha, 0.5)]$ are plotted in (a) for $t_a = 20\mu$s. To reduce finite-size scaling effects we exclude $L = 2, 3$ and perform the fit for $L \geq 4$. The intercept of the linear fits is the the DW2 scaling constant $a(\alpha)$ from Eq. (4.6), and is shown in (b). Error bars represent 2σ confidence intervals.



(a)



(b)

Figure 4.26: (a) The DW2 scaling coefficients $b(\alpha)$ from Eq. (4.6) for SAA at various sweep numbers and $\beta_f = 5$, along with the DW2 scaling coefficients for all t_as we tried. (b) The SQAS and SAS scaling coefficient $b(\alpha)$ at the optimal number of sweeps for each. SAS had a final temperature of $\beta_f = 5$ and SQAS was operated at $\beta = 5$.

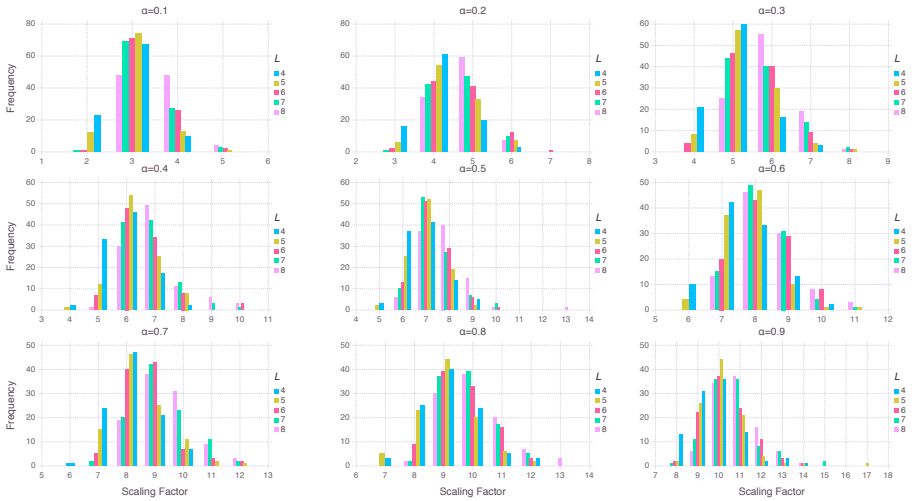


Figure 4.27: Histograms of the scale factor of the instances as a function of system size for various values of α. All problems passed to DW2 must have all coupler in the range $[-1, 1]$, so all couplers in a problem are rescaled down by a factor equal to the maximum absolute value of the couplers in the problem (and hence this quantity is called the scale factor). Since internal control error (ICE) is largely instance-independent, the larger the scaling factor of an instance, the worse the relative impact of ICE will be. We see a drift to larger scaling factors for with increasing size L and increasing clause density α. Larger values of α obviously will have larger scale factors as there are on average more loops per qubit (and thus, per edge) and thus a larger maximum potential coupler strength. Since the edges included in a loop are generated randomly, the more edges available at a fixed clause density, the more opportunities for a single edge to be included in many loops by chance, resulting the average scale factor to drift upward as function of problem size.

Chapter 5

Solving a Higgs optimization problem with quantum annealing for machine learning

5.1 Introduction

The discovery of the Higgs boson decays in a background of Standard Model processes was assisted by machine learning methods [134, 135]. The classifiers used to separate signal from background are trained using highly unerring but not completely perfect simulations of the physical processes involved, often resulting in label noise and systematic errors. In this chapter we investigate the application of quantum[45, 46, 136, 137] and classical annealing[15, 19] in solving a Higgs signal versus background machine learning optimization problem. We bag a set of weak classifiers built based on the kinematic observables of the Higgs decay photons into a strong classifier that is highly resilient against overtraining and errors in the Monte Carlo simulation correlations of the physics observables. We show that the resulting quantum and classical annealing classifier systems perform comparably to current state of the art machine learning methods used in particle physics[138, 139] and are simple functions of directly interpretable experimental parameters with a clear physical meaning. The annealer-trained classifiers exploit the excited states in the vicinity of the ground state and demonstrate some advantage for small training sizes. This technique may find application in other areas of experimental particle physics given the algorithm's relative simplicity and robustness to error. This chapter was originally published as Ref [4], with minor updates and modifications here.

The Higgs boson discovery at the Large Hadron Collider (LHC) [134, 135] marks the beginning of a new era in Particle Physics. Experimental particle

physicists at the LHC are measuring the new boson's properties [140, 141], searching for heavier Higgs bosons[142] and trying to understand if the Higgs boson interacts with dark matter[143]. Cosmologists are trying to understand the symmetry-breaking Higgs phase transition that took place early in the history of the universe and whether that event explains the excess of matter versus antimatter[144]. Curiously the measured value of the Higgs boson mass [142] tells us that the symmetry-breaking quantum vacuum is metastable[145] unless new physics intervenes. The profound implications of the Higgs boson discovery will keep motivating physics research for the years to come.

One of the key requirements to precisely measure the properties of the Higgs boson is selecting large, high-purity samples containing the production and decay of a Higgs particle. Applying machine learning techniques [146] is both a potentially powerful approach and offers challenges. The challenge is greater when an investigation requires faithful simulation not only of the physics observables themselves, but also of their correlations in the data. In the measurement of the properties of the Higgs boson [140], disagreements between simulations and observations result in label noise and systematic uncertainties in the efficiency of the classifiers that adversely impact the classification performance and translate into uncertainties on the measured properties of the discovered particle.

To address these challenges in the Higgs signal versus background optimization problem we study a binary classifier trained with classical simulated annealing (SA) [15, 19] and quantum annealing (QA) [45, 46, 136, 137, 34]. To implement QA we use a programmable quantum annealer (DW) built by D-Wave Systems, Inc. and housed at the University of Southern California's Information Sciences Institute, comprising 1098 superconducting flux qubits (note: this is the first study in this dissertation that uses the DW2X model, previous chapters used the DW2). The optimization problem is mapped to one of finding the ground state of a corresponding Ising spin model. We exploit the excited states in the vicinity of the ground state in the training method to improve the accuracy of the classifiers beyond the baseline ground state finding model. We refer to this approach as quantum annealing for machine learning (QAML).

5.2 Results

In what follows we discuss our results. The classifier accuracy is our criterion for comparison between various classifier construction methods. A classifier that is slow to train may prove practically more useful than a slightly less performant one that is faster to train.

Again, I wish to highlight that unlike in previous chapters, our goal is to estimate the performance of a classifier, not estimate time to solution. As I'll discuss below, this modifies many of our procedures, but the essential ideas remain.

We model the Higgs diphoton decay channel $H \to \gamma\gamma$. See Fig. 5.1 for the Feynman diagrams of the Higgs production and decay processes. We can

represent this system via the momentum of the Higgs particle (H), the momenta of the two photons (γ), the angle θ with the beam axis, and the azimuthal angle ϕ.

More specifically, we select eight of the kinematical variables describing the generated events as the variables for our classifier. The first five are related to the highest (p_T^1) and second highest (p_T^2) transverse momentum (momentum perpendicular to the axis defined by the colliding protons) of the photon pair: $p_T^1/m_{\gamma\gamma}$, $p_T^2/m_{\gamma\gamma}$, $(p_T^1 \pm p_T^2)/m_{\gamma\gamma}$, $p_T^{\gamma\gamma}/m_{\gamma\gamma}$, where $m_{\gamma\gamma}$ is the invariant mass of the diphoton pair and $p_T^{\gamma\gamma}$ is the transverse momentum of the diphoton system. The last three are $\Delta\eta$ of the two photons – the separation in the pseudorapidity $\eta = -\log(\tan(\theta/2))$ (η is a pseudo-invariant proxy to θ commonly used in high energy physics), $\Delta R = \sqrt{\Delta\eta^2 + \Delta\phi^2}$ – the sum in quadrature of the separation in η and ϕ of the two photons, and $|\eta^{\gamma\gamma}|$ – the η value of the diphoton system. Figure 5.2 shows the distribution of these variables for the signal and background datasets. The differences between these distributions are exploited by the classifier to distinguish the signal from the background. In addition to these eight variables, we incorporate the various products [using certain rules explained in the Section 5.7] between said variables, for a total of 36, given in Table 5.2.

We construct weak classifiers from our distributions of kinematical variables as shown in Fig. 5.2, and as described in Methods. We build the corresponding Ising problem as follows[137]. Let $\mathcal{T} = \{\vec{x_\tau}, y_\tau\}$ denote a given set of training events labeled by the index τ, where $\vec{x_\tau}$ is a vector collecting the values of each of the variables we use, and $y_\tau = \pm 1$ is a binary label for whether $\vec{x_\tau}$ is signal ($+1$) or background (-1). If $c_i(\vec{x_\tau}) = \pm 1/N$ denotes the value of weak classifier i on the event, where N is the number of weak classifiers, equal to the number of spins or qubits, then with $C_{ij} = \sum_\tau c_i(\vec{x_\tau})c_j(\vec{x_\tau})$ and $C_i = \sum_\tau c_i(\vec{x_\tau})y_\tau$ and a penalty $\lambda > 0$ to prevent overtraining, the Ising Hamiltonian is $H = \sum_{i,j} J_{ij}s_is_j + \sum_i h_is_i$, where $s_i = \pm 1$ is the i^{th} Ising spin variable, $J_{ij} = \frac{1}{4}C_{ij}$ is the coupling between spins i and j, and $h_i = \lambda - C_i + \frac{1}{2}\sum_{j=1} C_{ij}$ is the local field on spin i. The problem QA or SA attempt to solve is to minimize H and return the minimizing, ground state spin configuration $\{s_i^g\}_i$. The strong classifier is then constructed as $R(\vec{x}) = \sum_i s_i^g c_i(\vec{x}) \in [-1, 1]$ for each new event $\vec{x}$ that we wish to classify [137]. We introduce an additional layer in our study by also constructing strong classifiers from excited state spin configurations.

As benchmarks for traditional machine learning methods we train a deep neural net (DNN) using Keras[138] with the Theano backend,[147] and an ensemble of boosted decision trees using XGBoost[139], using optimized choices for training hyperparameters (details of which can be found for those interested in the Section 5.7).

5.2.1 Variable inclusion

: We compare the ground state configurations for $\lambda \in \{0.01, 0.05, 0.1, 0.2, 0.4, 0.8\}$; a larger λ implies an increasing penalty against including additional variables,

and thus we expect the variables included at $\lambda = 0.8$ to be determining the performance of the classifiers. Table 5.3 presents the relative strength of the variables in determining the classifier performance by showing how often variables are included in the ground state configuration of the full 36 variable problem derived from 20 different training sets with 20k training events each, as a function of the penalty term λ. We find that two of the original kinematical variables, p_T^1 and $\eta_{\gamma\gamma}$, are never included. The number of classifiers included in the ground state of all 20 training samples' corresponding Hamiltonian is 16 out of 36 for $\lambda \leq 0.05$ and the following three for $\lambda = 0.8$: i) $p_T^2/m_{\gamma\gamma}$, ii) $(\Delta R\, p_T^{\gamma\gamma})^{-1}$, and iii) $p_T^2/p_T^{\gamma\gamma}$. Those three dominate the network performance and they would have been difficult to guess *a priori* in their composite form. The physical reason for why these variables are important for the classifier can be gleaned from considering the kinematics of the system. The key difference between an event with a Higgs decaying to two photons and another process that produces two photons in its final state is the production of the heavy particle in the event. A heavy particle will require considerably more energy to boost perpendicular to the beamline and hence we would expect real Higgs events to have a characteristically lower $p_T^{\gamma\gamma}$ than background events. Since the system with the Higgs has less transverse boost, we would expect the two photons to have similar p_T spectra, and hence the momentum of the sub-leading photon will typically be higher than in those events without the heavy process. The p_T of the first photon is largely determined by the overall energy available in the collision, which is also set by $m_{\gamma\gamma}$, hence $p_T^1/m_{\gamma\gamma}$ is largely stochastic and provides little discrimination.

5.2.2 Classifier performance

: We estimate the receiver operating characteristic (ROC) curves on the training set and construct a final output classifier so that for a given signal efficiency (true positive rate) ϵ_S one uses the strong classifier sampled from the annealer which has the maximum background rejection (true negative rate) r_B. We construct such compound classifiers for both SA and DW using excited states within a fraction f of the ground state energy E^g, i.e., all those $\{s_i\}$ returned so that $H(\{s_i\}) < (1-f)E^g$ (note that $E^g < 0$). SA is used as a natural comparison to DW on these fully connected problems.

In our experiments DW struggles to find the true minimum of the objective function. This is likely a consequence of the fact that the current generation of the DW2X quantum annealers suffers from non-negligible noise on the programmed Hamiltonian.[1] Therefore we study and interrogate current-generation quantum annealers and interpret their performance as a lower bound for the performance of future systems with lower noise and denser hardware graphs.

In Fig. 5.3, we plot the ROC curves for each algorithm for $f = 0.05$ at training sizes 100 and 20,000. We observe a clear separation between the annealing-

[1] The problem of noise is compounded by the relatively sparse graph that requires a chain of qubits to embed the fully-connected logical Hamiltonian. In our case 12 qubits are ferromagnetically coupled to act as a single logical qubit

based classifiers and the XGB and DNN classifiers, with the advantage for the annealers appearing at the small training size, but disappearing at the large training size. In Fig. 5.4, we plot the area under the ROC curve (AUROC) for various training sizes and $f = 0.05$ (the largest value we used) for each algorithm. An ideal classifier would have an AUROC of 1. We find that DW and SA have comparable performance, implying high robustness to approximate solutions of the training problem. This feature appears to generalize across QAML domain-applications (in review, by Li, Felice, Rohs, and D.L. (LFRL)). Here the asymptotic performance of the QAML model is achieved with just 1000 training events, and thereafter the algorithm does not benefit from additional data. This is not true of DNN or XGB. A notable finding of this work is that QAML has an advantage over both DNN and XGB when training sizes are small. This is shown in Fig. 5.5 in terms of the integral of the true negative differences over signal efficiency for various ROCs. In the same regime of small training sizes, DW develops a small advantage over SA as the fraction of excited states f used increases, saturating at $f = 0.05$. The uncertainties are too large to draw definitive conclusions in this regard. In the regime of large training sizes, SA has a small advantage over DW, to a significance of approximately 2σ.

5.3 A note on data collction and error analysis, ie benchmarking, on this problem

It seems appropriate to pause and give a discussion here in the main text of this chapter on data collection and error analysis, in light of the discussion in chapter 3.

For both SA and DW, to evalute performance we constructed a histogram of the unique solutions returned from the algorithm, and exclude those states with rates of occurrence low enough that one cannot be certain of their inclusion in further runs. This is done by excluding any solution occurring fewer than three times, as they have a greater than 5% chance of exclusion in subsequent batches of 10^4 solutions. In this way, we are giving a robust lower bound on the ensemble classifier's performance. In essence, this represents a compromise between the very high computational costs of doing the full data analysis detailed in Chapter 3 and the desire to get as accurate an estimate of errors as possible. By excluding elements that have significant weight to not be resampled in a standard bootstrap procedure, we provided a conservative estimate of performance at much lower computational cost. Since the improvements gained by including additional states were relatively small, this is an acceptable compromise. The use of a classical bootstrap was due to the enormous number of samples and our very conservative cutoff. A similar procedure could have also been done with the Bayesian bootstrap, but in this case the classical bootstrap was simpler to code and the differences would be negligible.

5.3.1 Receiver operator characteristic curves and their uncertainty analysis

Any classifier may be characterized by two numbers: the true positive and true negative rates, in our case corresponding to the fraction of events successfully classified as signal or background, respectively. Since our classifiers all return floating point values in $[-1, 1]$, to construct a binary classifier we introduce a cut in this range, above and below which we classify as signal and background, respectively. Since this cut is a free parameter, we vary it across the entire range and plot the resulting parametric curve of signal acceptance (true positive, ϵ_S) and background rejection (true negative, r_B), producing a receiver operating characteristic (ROC) curve [148].

More explicitly, consider a labeled set of validation events, $\mathcal{V} = \{\vec{x_v}, y_v\}$, with $y_v = 0$ or 1 if $\vec{x_v}$ is background or signal, respectively, and a strong classifier $R_{\vec{w}}(\vec{x})$. The latter is constructed from a given set of weak classifiers and a vector of weights $\vec{w}$ previously obtained from training over a training set $\mathcal{T}$. The strong classifier outputs a real number $R_{\vec{w}}(\vec{x_v}) = \sum_i w_i c_i(\vec{x_v})$. To complete the classifier, one introduces a cut, O_c, such that we classify event $\vec{x_v}$ as signal if $R_{\vec{w}}(\vec{x_v}) > O_c$ and background if $R_{\vec{w}}(\vec{x_v}) < O_c$. If we evaluate the strong classifier on each of the events in our validation set $\mathcal{V}$, we obtain a binary vector of classifications, $\vec{C} = \{C_v\}$, with entries 0 denoting classification as background and 1 denoting classification as signal. By comparing C_v to y_v for all v we can then evaluate the fraction of the events which are correctly classified as background, called the true negative rate or "background rejection" r_B (equal to the number of times $C_v = y_v = 0$, divided by the total number of actual background events), and the fraction of events correctly classified as signal or "signal efficiency" ϵ_S (equal to the number of times $C_v = y_v = 1$, divided by the total number of actual signal events). For a given strong classifier, these values will be a function of the cutoff O_c. Plotting $r_B(O_c)$ against $\epsilon_S(O_c)$ yields a parametric curve, dubbed the "receiver operator characteristic" (ROC) curve, as shown in Fig. 5.6. Note that the cutoffs are trivial to adjust while all the computational effort goes into forming the networks, so one can vary O_c essentially for free to tune the performance of the network to suit one's purposes.

In other words, for a given strong classifier, i.e., solution/state, we can evaluate its output as a floating point number on each of the values in our data set, and for any value of a cut on $[-1, 1]$ this results in a single classification of the test data, $\vec{C}$. One can then evaluate the true positive and true negative rates by computing $\vec{C_v} \cdot \vec{y_k}$ where $k \in [S, B]$ (signal, background), $y_k^i = 1$ if datum i is in ensemble k and is 0 otherwise.

When we take $f > 0$ and accept excited states with energy $E < (1 - f)E_{GS}$ as "successes", we have a set of networks (labeled by f) for each training set. We simply take the supremum over the f-labeled set of values of r_B at each value of ϵ_S, to form the ROC curve for the classifier formed by pasting together different classifiers over various ranges of ϵ_S.

To estimate the error due to limited test sample statistics, we reweight each element of the test set with weights $\vec{w}$ drawn from a Poisson distribution with

mean 1, effectively computing $\sum_i w_i p_i y_k^i$. The weights on the elements of the test set are determined for all elements at once and we evaluate all strong classifiers using the same weights. As we are talking about events that occur at random intervals with some fixed rate, the Poisson distribution is the appropriate correlate in a Bayesian bootstrap-like scheme for this problem. For a single weight vector, we evaluate many values of the cut, and use linear interpolation to evaluate it in steps of 0.01 in the region $[0, 1]$. This gives us the true negative rate as a function of the true positive rate for a single weighting corresponding to a single estimated ROC curve. When constructing a composite classifier from multiple states, we are identifying regions of signal efficiency in which one should use one of the states rather than the others, namely, we take the maximum background rejection rate over the states for each value of signal efficiency.

Repeating for many reweightings we get many ROC curves all of which are consistent with our data, and thus the standard deviation across weights on a single training set at each value of signal efficiency serves as an estimate of the statistical uncertainty in our ROC curves.

To estimate variation due to the choice of the training set towards reproductions of the procedure and results, for a given training size we generate multiple disjoint training sets and use the standard deviation in mean performance across training sets as our estimate of the error on the model resulting from the particular choice of training set. When we compute the difference between two ROCs or AUROCs, we hold the training set and weight vector fixed, take the difference, and then perform statistics over the weights and training sets in the same manner as above. Errors in the AUROC were estimated similarly, taking the AUROC for each Poisson weight vector and training set (fold) instead of $r_B(\epsilon_S)$. This is the procedure leading to Fig. 5.3 in the main text. An example of the ROC curves is given in Fig. 5.6. At the scale of that plot, it is virtually impossible to tell the detailed differences between SA and DW or the various values of f, so we use plots of differences of AUCs to extract more detailed information about the ROC curves. This leads to Fig. **??**. Additional difference plots are given in Sec. 5.7.7 below.

As one can see, this error analysis procedure diverges in the specifics from that of chapter 3, but is not fundamentally different in spirit. One is still taking into account what information one has about the variables of interest in light of the available data, but in this case we primarily use a Poisson reweighting scheme rather than a Dirichlet distribution as we are simulating a posterior for a physical process where each element occurs with Poisson weights.

5.4 Discussion

We proceed to discuss and summarize our results from this study. With this study we explore QAML, a simple method inspired by the prospect of using quantum annealing as an optimization technique for constructing classifiers, and apply the technique to the detection of Higgs decays. The training data is

represented in a compact representation of $\mathcal{O}(N^2)$ couplers and local biases in the Hamiltonian for N weak classifiers. The resulting strong classifiers perform comparably to the state-of-the-art standard methods used in high energy physics today, and have an advantage when training sizes are small. The role of QA is that of a subroutine for sampling the Ising problem that may in the future have advantages over classical samplers, either when used directly or as a method of seeding classical solvers with-high quality initial states.

From the perspective of benchmarking, its role as a subroutine necessitates comparing QA to SA in the same computing pipeline in order to guarantee whatever results we have are not a result of merely our transformation to an Ising-type problem. For our case, SA solved the problems quite efficiently, so we did not need to extend our analysis to PIMC/SQA, HFS, or PT. Meanwhile, we still had to compare our QA/SA pipeline (namely QAML) to state of the art methods in the field, namely neural networks and boosted decision trees. Had we not done so, it would have been in principe possible to fool ourselves about the performance of DW or SA. This is a really general principle that can be applied to any benchmarking task: to demonstrate a real speedup of any kind, you have to replace the algorithms for which you're using quantum resources both by swapping the quantum resources directly with classical ones and by comparing entirely different, best-in-class algorithms for the problem class at hand.

QAML is resistant to overfitting, as the method involves an explicit linearization of correlations. It is also less sensitive to errors in the Monte Carlo correlation estimates than DNNs or binary decision trees due to the truncation of the tails of the distributions (see the Section 5.7). A useful aspect of the model is that it is directly interpretable, as each weak classifier directly corresponds to some physically relevant variable or product or ratio of variables, and the strong classifier is a simple linear combination thereof. This is in contrast to the usual creation of black-box machine learning discriminants, such as when using DNNs and XGBoost, where techniques for interpretability are still an active area of research [149].

This example of using quantum annealing to optimize classifiers in a physics problem was the first to-date, to the best of the authors' knowledge, and it opens further opportunities for research. It has since been followed up with a similar study in biology [150]. Quantum machine learning has taken great theoretical strides in the past years [151, 152, 153, 154, 155, 156, 157] and we demonstrate that we can already apply elements of this technique to current and next-generation quantum annealing architectures. The near future will see applications of these techniques to more complex problems in particle physics and other domain sciences. For example, this work has motivated similar studies in computational biology (in review, LFRL).We envision that QAML could be used further in the context of data certification in high energy physics, where training with small dataset size could be particularly useful. One can leverage the robustness of QAML to both significantly reduce the level of human intervention and increase the accuracy in quickly assessing and certifying the collision data for physics analysis. Being impervious to overtraining, QAML is

a good candidate for boosting[139] and we foresee studies to evaluate this using future versions of the D-Wave quantum annealers or other architectures, or even classical optimization techniques such as SA. Multi-stage classifiers—where the most influential variables are found via training and pulled out to form a smarter classifier—could be used to mitigate the influence of hardware noise in the quantum annealer. With more available qubits, or more efficient architectures, integer weights could be used in place of the binary weights through a straightforward extension of the encoding scheme used in this work.

In the next and final, mercifully brief chapter, I will summarize some of the principles of benchmarking we've learned/encountered along the way of this journey, in the hopes of providing a quicker referencefor the community.

5.5 Acknowledgements for this chapter and author contributions statement

This project is supported in part by the United States Department of Energy, Office of High Energy Physics Research Technology Computational HEP and Fermi Research Alliance, LLC under Contract No. DE-AC02-07CH11359. The project is also supported in part under ARO grant number W911NF-12-1-0523 and NSF grant number INSPIRE-1551064. The work is supported in part by the AT&T Foundry Innovation Centers through INQNET, a program for accelerating quantum technologies. We wish to thank the Advanced Scientific Computing Research program of the DOE for the opportunity to first present and discuss this work at the ASCR workshop on Quantum Computing for Science (2015). We acknowledge the funding agencies and all the scientists and staff at CERN and internationally whose hard work resulted in the momentous H(125) discovery in 2012.

A.M. conducted the first mapping of the problem on the D-Wave software architecture, data analysis and machine learning methodology, and along with M.S. provided the core knowledge of the domain science (Higgs physics). J.J conducted QA and SA research and data analysis as well as the machine learning work and along with D.L. provided the core of QA/SA knowledge. J-R.V. conducted QA research, data analysis, machine learning methods and error analysis work with J.J. for the submitted version of the work. D.L. and M.S. provided the overall oversight and scrutiny of the research work, data analysis and results. D.L. conceived the quantum machine learning methodology and M.S. conceived the domain application. We consider the two graduate students (A.M. and J.J.) as operationally the first authors of this work. All authors contributed to writing and reviewing the original manuscript as published in Nature, [4].

The authors declare no competing financial interests.

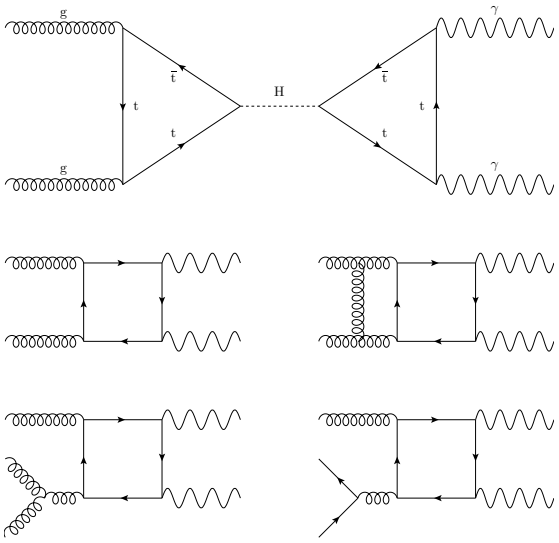


Figure 5.1: **Representative Feynman diagrams** contributing to the simulated distributions for Higgs signal and Standard Model background. The signal is a Higgs production through gluon-fusion which decays into two photons (top). Representative Leading Order and Next-to-Leading Order background processes are Standard Model two photon production processes (bottom).

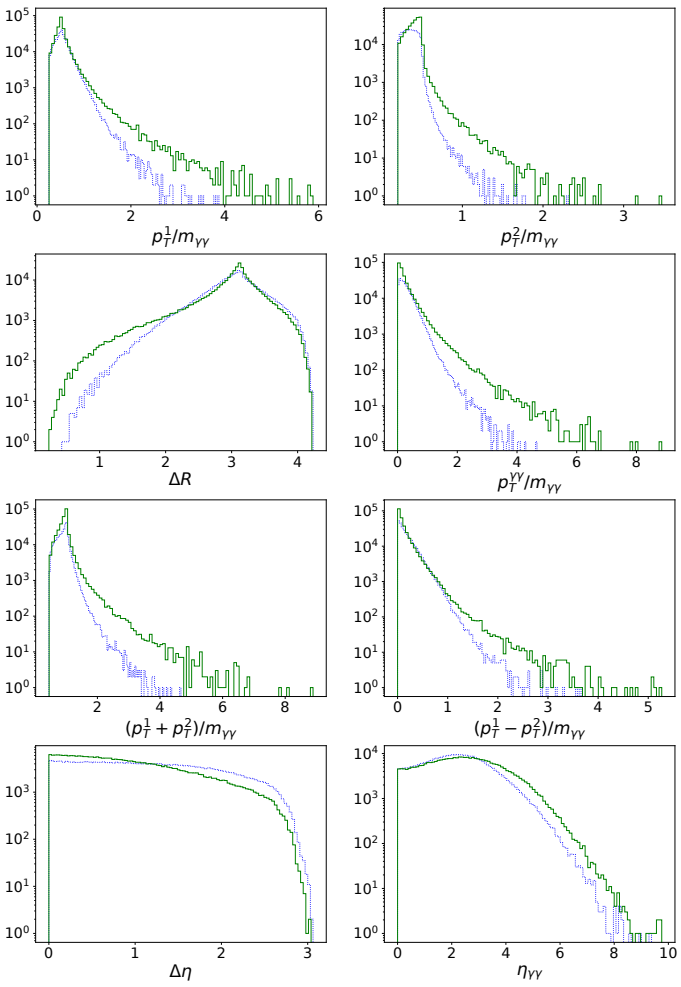


Figure 5.2: **Distributions of the eight kinematical variables** we used to construct weak classifiers. The signal distribution is in green and solid, background in blue and dotted. The vertical axis is the raw count of the number of events. The total number of events simulated in each case is 307732.

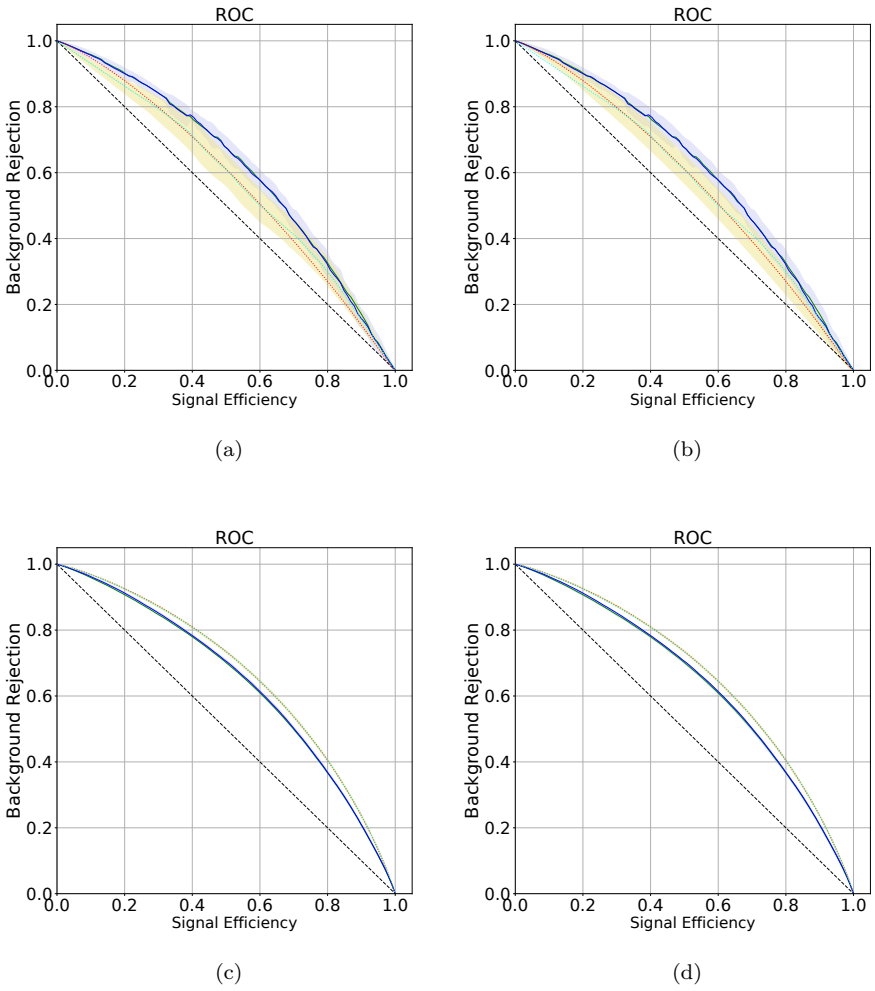


Figure 5.3: **ROC curves** for the annealer-trained networks (DW and SA) at $f = 0.05$, DNN, and XGB. Error bars are defined by the variation over the training sets and statistical error. Both panels show all four ROC curves, with solid lines for DW (green) and SA (blue), and dotted lines for DNN (red) and XGB (cyan). Panels (a) and (c) [(b) and (d)] includes 1σ error bars only for DW and DNN [SA and XGB], in light blue and pale yellow, respectively. Results shown are for the 36 variable networks at $\lambda = 0.05$ trained on 100 events for panels (a) and (b), and on $20,000$ events for panels (c) and (d). For 100 events the annealer trained networks have a larger area under the ROC curve, as shown directly in Fig. 5.4. The situation is reversed for $20,000$ training events, where the error bars are too small to be visible. The much smaller error bars are due to the increased number of events.

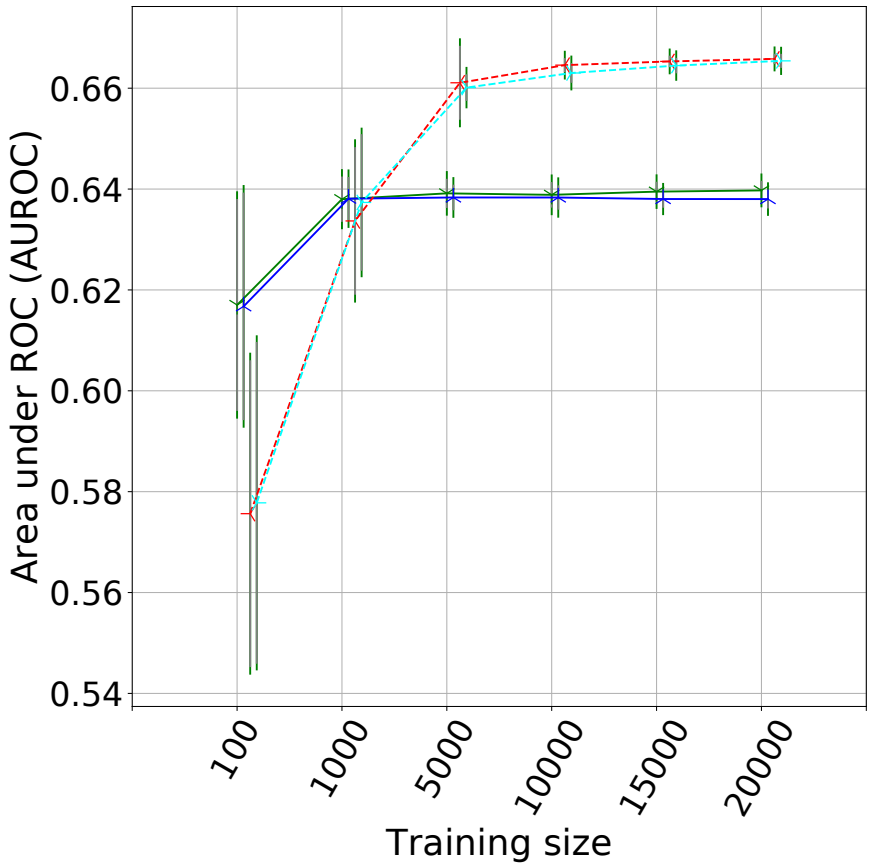
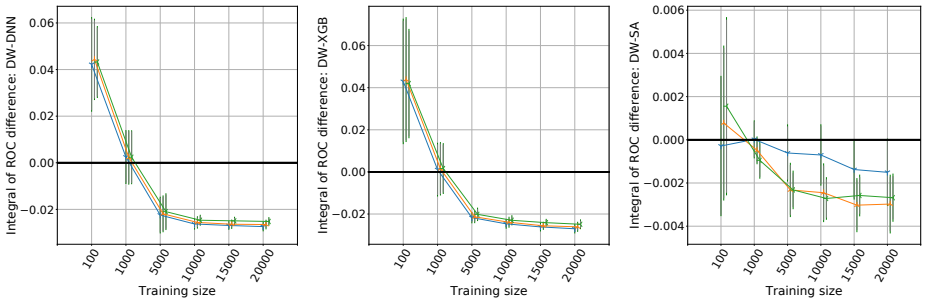


Figure 5.4: **Area under the ROC curves (AUROCs)** for the annealer-trained networks (DW (green) and SA (blue), solid lines) at $f = 0.05$, and the conventional approaches (DNN (red) and XGB (cyan), dotted lines). The vertical lines denote 1σ error bars, defined by the variation over the training sets (grey) plus statistical error (green); see the SI (Sec. 6) for details of the uncertainty analysis. While DNN and XGB have an advantage at large training sizes, we find that the annealer-trained networks perform better for small training sizes. Results shown are for the 36 variable networks at $\lambda = 0.05$. The overall QAML performance and its features, including the advantage at small training sizes and saturation at ≈ 0.64, are stable across a range of values for λ. An extended version of this plot with various values of λ is available in the SI in Fig. 2.

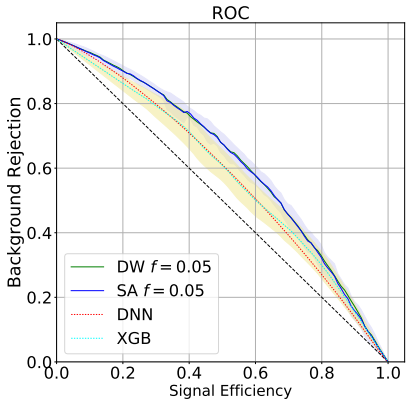


(a) (b) (c)

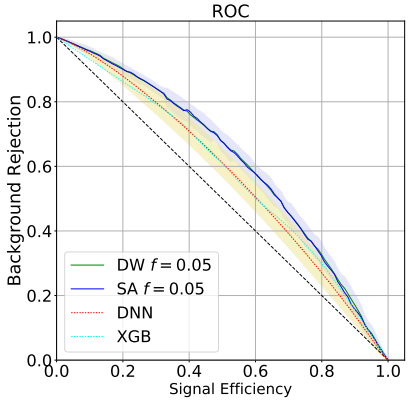
Figure 5.5: **Difference between AUROCs** of (a) DW vs DNN, (b) DW vs XGBoost, and (c) DW vs SA, as a function of training size and fraction f above the minimum energy returned (the same values of f are used for DW and SA in (c)). Formally, we plot $\int_0^1 [r_B^{(\mathrm{DW})}(\epsilon_S) - r_B^{(i)}(\epsilon_S)] d\epsilon_S$, where $i \in \{\mathrm{DNN}, \mathrm{XGBoost}, \mathrm{SA}\}$. The vertical lines denote 1σ error bars. The large error bars are due to noise on the programmed Hamiltonian.

variable	description
$p_T^1/m_{\gamma\gamma}$	transverse momentum of the highest p_T photon divided by the invariant mass of the diphoton pai
$p_T^2/m_{\gamma\gamma}$	transverse momentum of the second-highest p_T photon divided by the invariant mass of the dipho
$(p_T^1 + p_T^2)/m_{\gamma\gamma}$	sum of the transverse momentum of the two photons divided by their invariant mass
$(p_T^1 - p_T^2)/m_{\gamma\gamma}$	difference of the transverse momentum of the two photons divided by their invariant mass
$p_T^{\gamma\gamma}/m_{\gamma\gamma}$	transverse momentum of the diphoton system divided by its invariant mass
$\Delta\eta$	difference in $\eta = -\log\tan\left(\frac{\theta}{2}\right)$, where θ is the angle with the beam axis
ΔR	sum in quadrature of the separation of and ϕ, the azimuthal angle of the two photons ($\sqrt{\Delta\eta^2 + \Delta$
$\lvert\eta^{\gamma\gamma}\rvert$	the η value of the diphoton system

Table 5.1: **The kinematical variables used to construct weak classifiers.**



(a)



(b)

Figure 5.6: The ROC curves for the annealer-trained networks (DW and SA) at $f = 0.05$, DNN, and XGB. Error bars are defined by the variation over the training sets and statistical error. Both panels show all four ROC curves. Panel (a) [(b)] includes 1σ error bars only for DW and DNN [SA and XGB], in light blue and pale yellow, respectively. Results shown are for the 36 variable networks at $\lambda = 0.05$ trained on 100 events. The annealer trained networks have a larger area under the ROC curve

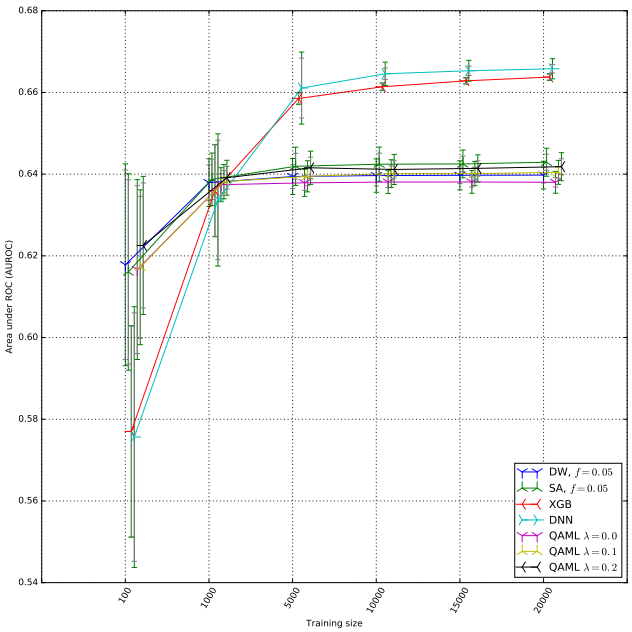


Figure 5.7: A reproduction of Fig. 5.4 from the main text, now including the optimal strong classifier found by SA at $f = 0$ for various values of the regularization parameter $\lambda = 0., 0.1, 0.2$. We find that this parameter has negligible impact on the shape of the AUROC curve, and that performance for SA always saturates at ≈ 0.64, with an advantage for QAML (DW) and SA over XGB and DNNs for small training sizes.

Table 5.2: **Map from number to variable/weak-classifier name**

1	2	3	4	5	6	7	8	9
p_T^1	p_T^2	ΔR	$p_T^{\gamma\gamma}$	$p_T^1 + p_T^2$	$p_T^1 - p_T^2$	$\Delta\eta$	$\eta_{\gamma\gamma}$	$(p_T^1 + p_T^2)\eta_{\gamma\gamma}$
10	11	12	13	14	15	16	17	18
$\frac{p_T^2}{p_T^1 - p_T^2}$	$\frac{p_T^2}{\Delta\eta}$	$p_T^2\eta_{\gamma\gamma}$	$\frac{1}{\Delta R p_T^{\gamma\gamma}}$	$\frac{p_T^1+p_T^2}{\Delta R}$	$\frac{1}{\Delta R(p_T^1-p_T^2)}$	$\frac{1}{\Delta R\Delta\eta}$	$\frac{\eta_{\gamma\gamma}}{\Delta R}$	$\frac{1}{(p_T^1-p_T^2)\Delta\eta}$
19	20	21	22	23	24	25	26	27
$p_T^1 p_T^2$	$\frac{p_T^1}{\Delta R}$	$\frac{p_T^1}{p_T^{\gamma\gamma}}$	$p_T^1(p_T^1 + p_T^2)$	$\frac{p_T^1}{p_T^1-p_T^2}$	$\frac{p_T^1}{\Delta\eta}$	$\frac{p_T^1}{\eta_{\gamma\gamma}}$	$\frac{p_T^2}{\Delta R}$	$\frac{\eta_{\gamma\gamma}}{p_T^1-p_T^2}$
28	29	30	31	32	33	34	35	36
$\frac{p_T^2}{p_T^{\gamma\gamma}}$	$p_T^2(p_T^1 + p_T^2)$	$\frac{p_T^1+p_T^2}{p_T^{\gamma\gamma}}$	$\frac{\eta_{\gamma\gamma}}{p_T^{\gamma\gamma}}$	$\frac{1}{p_T^{\gamma\gamma}\Delta\eta}$	$\frac{1}{p_T^{\gamma\gamma}(p_T^1-p_T^2)}$	$\frac{p_T^1+p_T^2}{p_T^1-p_T^2}$	$\frac{p_T^1+p_T^2}{\Delta\eta}$	$\frac{\eta_{\gamma\gamma}}{\Delta\eta}$

Table 5.3: **Variable inclusion in the ground states of the Ising problem instances**. The variables listed are those from which we selected the various variables included in our tests with varying problem size. We list how many out of 20 training sets had the given variable turned on in the ground state configuration. Three of the 36 variables were included for all values of the penalty term λ and for all of the training sets [p_T^2, $(\Delta R p_T^{\gamma\gamma})^{-1}$, and $\frac{p_T^2}{p_T^{\gamma\gamma}}$], the variables $p_T^2/(p_T^1 - p_T^2)$ and $(p_T^1 + p_T^2)/\Delta\eta$ were present in almost all, while seven were never included, among which the original kinematical variables p_T^1 and $\eta_{\gamma\gamma}$. All momenta $(p_T^1, p_T^2, p_T^{\gamma\gamma})$ are given in units of $m_{\gamma\gamma}$. Variables are given in Table 5.2.

λ	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
0	0	20	20	20	19	20	20	0	5	20	20	20	20	19	20	17	20	20
0.01	0	20	20	20	19	20	20	0	4	20	20	20	20	19	20	17	20	20
0.02	0	20	20	20	19	20	20	0	4	20	20	20	20	19	20	16	20	20
0.05	0	20	20	20	19	20	20	0	1	20	20	20	20	19	20	10	20	17
0.1	0	20	20	20	19	20	20	0	0	20	20	20	20	19	20	6	14	2
0.2	0	20	20	20	19	20	20	0	0	20	14	20	20	12	20	4	1	0
0.4	0	20	0	2	19	20	20	0	0	20	17	20	20	0	20	1	0	0
0.8	0	20	0	0	0	0	9	0	0	18	0	0	20	0	2	0	0	0

λ	19	20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36
0	20	0	0	19	0	20	0	3	0	20	19	7	0	15	0	19	20	20
0.01	20	0	0	19	0	20	0	2	0	20	19	6	0	15	0	19	20	20
0.02	20	0	0	19	0	20	0	1	0	20	19	4	0	15	0	19	20	20
0.05	20	0	0	19	0	20	0	0	0	20	16	1	0	11	0	19	20	20
0.1	20	0	0	1	0	20	0	0	0	20	1	0	0	5	0	16	20	20
0.2	18	0	0	0	0	20	0	0	0	20	0	0	0	0	0	0	20	20
0.4	0	0	0	0	0	7	0	0	0	20	0	0	0	0	0	0	20	3
0.8	0	0	0	0	0	0	0	0	0	20	0	0	0	0	0	0	19	0

The following sections are available for those who want much more detailed information on our problem sets and parameters for our algorithms, more detailed information on ROC curves, and a large number of plots for those interested which are not necessary to understand the thrust of this study.

5.6 Methods

5.6.1 Problem Construction

We simulate $3x10^5$ 125 GeV mass Higgs particle decays produced by gluon fusion at $\sqrt{\hat{s}} = 8$ TeV using PYTHIA 6.4 [158] and $3x10^5$ SM-process background events using SHERPA [159] after restricting to those processes with realistic detector acceptance and trigger requirements that lie directly under the Higgs

peak (to ensure the classifier cannot select on mass information): $|\eta| < 2.5$ with one photon having $p_T > 32\,\text{GeV}$ and the other having $p_T > 25\,\text{GeV}$, with total di-photon invariant mass $122.5\,\text{GeV} < m_{\gamma\gamma} < 127.5\,\text{GeV}$. The dominant Feynman diagrams are given in Figure 5.1. The resulting distributions for eight kinematical variables in this problem are given in Fig. 5.2. The complete procedure of the weak classifier construction is given in the SI.

There are 666 floating point parameters in our Ising Hamiltonian on 36 variables. XGBoost with a maximum depth of 10 has up to 1024 decisions (each a free variable/parameter) in each tree. Our DNN has 2000 local biases and approximately 500000 weights on/between the two 1000 node hidden layers.

5.6.2 Data collection and analysis

For SA, all the Ising Hamiltonians are run 10^4 times at various numbers of sweeps in the range 1000 at initial inverse temperature $\beta = 0.1$ to final inverse temperature $\beta = 5$, with a linear β schedule. Ground state energies are estimated using SA at 10^4 sweeps with a linear schedule from $\beta = 0.1$ to $\beta = 10$. For DW we first create for each instance a heuristic embedding using the D-Wave API. DW is run with 50 gauges [34] at the minimum possible annealing time of $5\mu s$ for 200 samples per gauge at a chain strength of 6. This is in line with my proposal from chapter 3, though here we used a fixed number of gauges because the data analysis component was far too heavy (at least at the time of running) to derive a posterior density efficiently on the fly.

5.6.3 Weak classifier construction

We define $S(v)$ as the distribution for the signal of variable v, and similarly $B(v)$ is the distribution of the background for variable v. For a given value of the variable v, compute the 70^{th} percentile of $S(v)$, call it v_{cut} and find the percentile corresponding to that same value in $B(v)$. If the percentile in $B(v_{\text{cut}}) < 70$ then we center v_{cut} ($v' = v - v_{\text{cut}}$) and reflect across the vertical axis ($v'' = -v'$) so that $S(v'') > B(v'')$ for $v'' > 0$, and thus the region $v'' > 0$ is predominantly signal and $v'' < 0$ is predominantly background. If the percentile in $B(v_{\text{cut}}) > 70$ then we compute the 30^{th} percentile of $S(v)$ (a new v_{cut}) and if $B(v_{\text{cut}}) > 30$ then we center at the new v_{cut} and do not reflect across the vertical axis, as it already satisfies the requirement that $S(v'') > B(v'')$ for $v'' > 0$. If neither of these conditions is satisfied, we reject the variable as unsuitable for weak classifier construction. Now determine the 10^{th} percentile for $S(v)$ and the 90^{th} for $B(v)$ and call them v_{+1} and v_{-1} respectively. The weak classifier is

as follows:

$$c(v) = \begin{cases} +1 & \text{if } v_{+1} < v''(v) \\ \dfrac{v''(v)}{v_{+1}} & \text{if } 0 < v''(v) < v_{+1} \\ \dfrac{v''(v)}{|v_{-1}|} & \text{if } v_{-1} < v''(v) < 0 \\ -1 & \text{if } v''(v) < v_{-1} \end{cases} \tag{5.1}$$

By construction, $c(v)$ has all of the properties we seek in a weak classifier. Now, since this procedure removes information about the tails of the distributions and does not take into account correlations between our kinematical variables, we add products and ratios of the kinematical variables to our description. If we had to flip the distribution for variable i define $g_i = \frac{1}{v_i}$, otherwise $g_i = v_i$. We then add all functions of the form $p(g_i, g_j) = g_i * g_j$, and perform the weak classifier construction on these combinations.

5.6.4 Mapping weak classifier selection to the Ising problem

In this section we closely follow Ref. [137], with slight changes of notation. Let V be the event space, consisting of vectors $\{\vec{x}\}$ that are either signal or background. We define a weak classifier $c_i(\vec{x}) : V \mapsto \mathbb{R}$, $i = 1, \dots N$, as classifying event $\vec{x}$ as signal (background) if $c_i(\vec{x}) > 0$ ($c_i(\vec{x}) < 0$). We normalize each weak classifier so that $|c_i| \leq 1/N$. We introduce a binary weights vector $\vec{w} \in \{0, 1\}^N$ and construct a strong classifier $R_{\vec{w}}(\vec{x}) = \sum_i w_i c_i(\vec{x}) \in [-\|\vec{w}\|/N, \|\vec{w}\|/N]$. The event $\vec{x}$ is correspondingly classified as signal (background) if $R_{\vec{w}}(\vec{x}) > 0$ ($R_{\vec{w}}(\vec{x}) < 0$). The weights $\vec{w}$ are to be determined; they are the target of the solution of the Ising problem.

Let $\mathcal{T} = \{\vec{x_\tau}, y_\tau\}$ denote a given set of training events, where $\vec{x_\tau}$ is an event vector collecting the values of each of the variables we use, and $y_\tau = \pm 1$ is a binary label for whether $\vec{x_\tau}$ is signal (+1) or background (−1). Let $Q_{\vec{w}}(\vec{x}) = \text{sign}[R_{\vec{w}}(\vec{x})]$, so that $Q_{\vec{w}}(\vec{x}) = +1$ (−1) denotes signal (background) event classification. Thus $y_\tau Q_{\vec{w}}(\vec{x}_\tau) = +1$ if $\vec{x}_\tau$ is correctly classified as signal or background (y_τ and $Q_{\vec{w}}(\vec{x}_\tau)$ agree), and $y_\tau Q_{\vec{w}}(\vec{x}_\tau) = -1$ if $\vec{x}_\tau$ is incorrectly classified (y_τ and $Q_{\vec{w}}(\vec{x}_\tau)$ disagree). The cost function $L(\vec{w}) = \sum_\tau [1 - y_\tau Q_{\vec{w}}(\vec{x}_\tau)]/2$ thus counts the number of incorrectly classified training events, and minimizing it over all possible weight vectors returns the optimal set of weights, and hence the optimal strong classifier given the training set $\mathcal{T}$. To avoid overtraining and economize on the number of weak classifiers used, we can introduce a penalty term proportional to the number of weights, i.e., $\lambda\|\vec{w}\|$, where $\lambda > 0$ is the penalty strength. Thus the optimal set of weights for given λ is

$$\vec{w}_{\text{opt}} = \text{argmin}_{\{\vec{w}\}} \left[L(\vec{w}) + \lambda \|\vec{w}\| \right]. \tag{5.2}$$

This optimization problem cannot be directly mapped onto a quantum annealer, due to the appearance of the sign function. Instead we next introduce

a relaxation to a quadratic form that is implementable on the current generation of D-Wave devices. Namely, using the training set we form the vector of strong classifier results $\vec{R}_{\vec{w}} = \{R_{\vec{w}}(\vec{x_\tau})\}_{\tau=1}^{|\mathcal{T}|}$, the Euclidean distance measure $\delta(\vec{w}) = \|\vec{y} - \vec{R}_{\vec{w}}\|^2$ between the strong classifier and the set of training labels, and replace Eq. (5.2) by

$$\vec{w}_{\min} = \text{argmin}_{\{\vec{w}\}}\delta(\vec{w}). \tag{5.3}$$

Finding $\vec{w}_{\text{opt}}$ in this way is equivalent to solving a quadratic unconstrained binary optimization (QUBO) problem:

$$\vec{w}_{\min} = \text{argmin}_{\{\vec{w}\}}\left[\sum_\tau R_{\vec{w}}(\vec{x_\tau})^2 - 2y_\tau R_{\vec{w}}(\vec{x_\tau}) + y_\tau^2\right] \tag{5.4}$$

$$= \text{argmin}_{\{\vec{w}\}}\left[\sum_\tau \Big(\sum_{i,j} w_i w_j c_i(\vec{x_\tau})c_j(\vec{x_\tau}) - 2y_\tau \sum_i w_i c_i(\vec{x_\tau})\Big) + |\mathcal{T}|\right] \tag{5.5}$$

Regrouping the terms in the sum and dropping the constant we find:

$$\vec{w}_{\min} = \text{argmin}_{\{\vec{w}\}}\left[\sum_{i,j} w_i w_j \left(\sum_\tau c_i(\vec{x_\tau})c_j(\vec{x_\tau})\right) - 2\sum_i w_i \left(\sum_\tau c_i(\vec{x_\tau})y_\tau\right)\right] \tag{5.6}$$

$$= \text{argmin}_{\{\vec{w}\}}\left[\sum_{i,j} C_{ij} w_i w_j - 2\sum_i C_i w_i\right] \tag{5.7}$$

where $C_{ij} = \sum_\tau c_i(\vec{x_\tau})c_j(\vec{x_\tau}) = C_{ji}$ and $C_i = \sum_\tau c_i(\vec{x_\tau})y_\tau$.

This has a tendency to overtrain. The reason is that $|R_{\vec{w}}(\vec{x_\tau})| \le \|\vec{w}\|/N$, so that $|y_\tau - R_{\vec{w}}(\vec{x_\tau})|^2 \ge (1 - \|\vec{w}\|/N)^2$, and hence $\delta(\vec{w}) = \sum_\tau |y_\tau - R_{\vec{w}}(\vec{x}_\tau)|^2 \ge |\mathcal{T}|(1 - \|\vec{w}\|/N)^2$. To minimize $\delta(\vec{w})$ the solution will be biased toward making $\|\vec{w}\|$ as large as possible, i.e., to include as many weak classifiers as possible. To counteract this overtraining tendency we add a penalty term that makes the distance larger in proportion to $\|\vec{w}\|$, i.e., $\lambda\|\vec{w}\|$ with $\lambda > 0$, just as in Eq. (5.2). Thus we replace Eq. (5.7) by

$$\vec{w}_{\min} = \text{argmin}_{\{\vec{w}\}}\left[\sum_{i,j} C_{ij} w_i w_j + \sum_i (\lambda - 2C_i)w_i\right], \tag{5.8}$$

The last step is to convert this QUBO into an Ising problem by changing the binary w_i into spin variables $s_i = \pm 1$, i.e., $w_i = (s_i + 1)/2$, resulting in:

$$\vec{s}_{\min} = \text{argmin}_{\{\vec{s}\}}\left[\frac{1}{4}\sum_{i,j} C_{ij} s_i s_j + \frac{1}{2}\sum_{i,j} C_{ij} s_i + \frac{1}{2}\sum_i (\lambda - 2C_i)s_i\right], \tag{5.9}$$

where we use the symmetry of C_{ij} to write the middle term in the second line, and we drop the constant terms $\frac{1}{4}\sum_{i,j} C_{ij}$ and $\frac{1}{2}\sum_i(\lambda - 2C_i)$. We now define the couplings $J_{ij} = \frac{1}{4}C_{ij}$ and the local fields $h_i = \frac{1}{2}\left(\lambda - 2C_i + \sum_j C_{ij}\right)$. The optimization problem is then equivalent to finding the ground state $\vec{s}_{\min} = \text{argmin}_{\{\vec{s}\}} H$ of the Ising Hamiltonian

$$H_{\text{Ising}} = \sum_{i<j}^{N} J_{ij} s_i s_j + \sum_{i=1}^{N} h_i s_i. \tag{5.10}$$

In the main text and hereafter, when we refer to λ it is measured in units of $\max_i(C_i)$ (e.g., $\lambda = 0.05$ is shorthand for $\lambda = 0.05\max_i(C_i)$).

5.6.5 Instances and variable inclusion

We use 8 kinematical variables, listed in Table 5.1. They involve functions of the individual and diphoton mass, as well as the angles of the photons and the diphoton system. Taking the products between them, we get 36 products, with all of them passing the weak classifier construction procedure for the vast majority of the training sets. These 36 weak classifiers (or a subset thereof) are the set from which we built our strong classifiers. For each size of training set in $[100, 1000, 5000, 10000, 15000, 20000]$, we generated 20 training sets and generated the corresponding Ising problem for $\lambda = 0.05$. In order to compare the performance of SA and QA, we estimate ground state solution of these Ising problems by running SA for a large number of sweeps (10^4) with a low final temperature (0.1 in normalized energy units).

Data availability: The data that support the findings of this study are available from the corresponding author upon reasonable request. Source data for all figures are provided with the paper.

5.7 Additional background and pedagogy

5.7.1 DNN and XGB optimization procedure

We benchmark the performance of QAML against DNN and XGB.

We train a DNN using Keras [138] with the Theano backend,[147] a standard tool in deep learning and increasingly popular in high energy physics. Our network has two fully connected hidden layers with 1000 nodes each. The model is optimized using the Adam algorithm [160] with a learning rate of 0.001 and a mini-batch size of 10. We find that network performance is not affected by small changes in the number of nodes or the initial guesses for the weights. The model hyperparameters, regularization terms, and optimization parameters for our deep neural net are selected using the *Spearmint* Bayesian optimization software [161, 162]. Early stopping is used (with patience parameter 10) to avoid overtraining and have sufficient generalization.

We also train an ensemble of boosted decision trees using XGB [139] with a maximum depth of 10, a learning rate of 0.3, and L2-regularization parameter $\lambda = 2000$.

To train and optimize XGB, we use 100 rounds of training and start with the default choices for the various parameters. We evaluate values of the learning rate

$$\eta \in \{0.001, 0.002, 0.003, 0.005, 0.008, 0.01, 0.02, 0.03, 0.05, 0.08, 0.1, 0.2, 0.3, 0.5, 0.8\}$$

at tree depths of 5, 8, 10, 12, 15, and 20. Some of these parameters give small improvements in AUC over the defaults at value of the L2-regularization parameter $\lambda = 1$. Far larger improvements are found when λ is increased. Hence we hold the other parameters fixed and evaluate

$$\lambda \in \{5, 10, 20, 50, 100, 200, 500, 1000, 1500, 1800, 2000, 2200, 2500\}$$

, finding the approximate optimum AUC on the test set at 2000. Testing again, the tree depth and η are found to have minimal effect on the AUC (significantly smaller than the error), and $\eta = 0.3$ and tree depth 10 are chosen as the approximate optimum.

We note that the DNN and XGB settings are selected so as to prevent overtraining.

5.7.2 Robustness of QAML to MCMC mismodelling

Two essential steps are involved in the construction of the weak classifiers in our approach. First, we remove information about the tails of the distributions of each variable and use the corresponding truncated single-variable distributions to construct weak classifiers. Second, since the single-variable classifiers do not include any correlations between variables, we include additional weak classifiers built from the products/ratios of the variables, where after taking the products/ratios we again apply the same truncation and remove tails. That is, our weak classifiers account only for one and two-point correlations and ignore all higher order correlations in the kinematic variable distribution. The particular truncation choice to define the weak classifiers as a piecewise linear function defined only by a central percentile (30th or 70th, chosen during construction) and two percentiles in the tails (10th and 90th) means that the MC simulations only have to approximately estimate those four percentiles of the marginals and the correlations between the variables. Any MC simulation which is unable to approximate the 10th, 30th, 70th, and 90th percentiles of the marginal distribution for each dimension of the dataset and the products between them would surely not be considered acceptably similar to the target distribution for use in HEP data analyses, as it is effectively guaranteed to be wrong in the higher order correlations and thus in its approximation of the true distribution. Meanwhile, typical machine learning approaches for this problem use arbitrary relationships across the entire training dataset, including the tails and high-order correlations, and so are likely to be more sensitive to any mismodelling.

5.7.3 Quantum annealing and D-Wave

Current and near-generation quantum annealers are naturally run in a batch mode in which one draws many samples from a single Hamiltonian. Repeated draws for QA are fast. The DW averages approximately 5000 samples per second under optimal conditions. We take advantage of this by keeping all the trial strong classifiers returned and not restricting to the one with minimum energy.[2]The DW has 1098 superconducting Josephson junction flux qubits arranged into a grid, with couplers between the qubits in the form shown in Fig. 5.8, known as the Chimera graph. The annealing schedule used in the DW processor is given in Fig. 5.9. The Chimera graph is not fully connected, a recognized limitation as the Ising Hamiltonian (5.10) is fully connected, in general. To address this, we perform a *minor embedding operation* [31, 11]. Minor embedding is the process whereby we map a single logical qubit in H_{Ising} into a physical ferromagnetic ($J_{ij} = -1$) chain of qubits on DW. For each instance we use a heuristic embedding found via the D-Wave API, that is as regular and space-efficient as possible for our problem sizes.

Given a minor embedding map of logical qubits into a chain of physical qubits, we divide the local fields h_i equally among all the qubits making up the chain for logical qubit i, and divide J_{ij} equally among all the physical couplings between the chains making up logical qubits i and j. After this procedure, there remains a final degree of freedom: the chain strength J_F. If the strength of the couplers in the ferromagnetic chains making up logical qubits is defined to be 1, then the maximum magnitude of any other coupler is $\max\left(\max_i(\{|h_i|\}), \max_{i,j}(\{|J_{ij}|\})\right) = \frac{1}{J_F}$. There is an optimal value of J_F, generally. This is due to a competition between the chain needing to behave as a single large qubit and the problem Hamiltonian needing to drive the dynamics[83]. If J_F is very large, the chains will "freeze out" long before the logical problem, i.e., the chains will be far stronger than the problem early on, and the transverse field terms will be unable to induce the large, multi-qubit flipping events necessary to explore the logical problem space. Similarly, if J_F is very weak, the chains will be broken (i.e., develop a kink or domain wall) by tension induced by the problem, or by thermal excitations, and so the system will generally not find very good solutions. Ideally, one wants the chains and the logical problem to freeze at the same time, so that at the critical moment in the evolution both constraints act simultaneously to determine the dynamics. For the results shown here, we used $J_F = 6$ with an annealing time $t_a = 5\mu$s. To deal with broken chains, we use majority vote on the chain with a coin-toss tie-breaker for even length chains. Detailed analysis of the performance of this strategy in the context of error correction can be found in the literature [163, 164].

Figure 5.10 shows the average minimum energy returned by the DW, rescaled by the training size (to remove a linear scaling), as a function of the chain strength and training size. We see that the smallest training size ($N = 100$)

[2]The energy is effectively a function of error on the training set of the weak classifiers, hence is distinct from the measures used to directly judge classifier performance, such as the area under the ROC curve.

has a smaller average minimum energy than the rest of the training sizes, and that there is only a very slight downward tendency as the chain strength J_F increases for the larger training sizes.

Figure 5.11 plots the fractional deviation of the minimum energy returned by the DW relative to the true ground state energy, averaged over the training sets. While the DW's minimum energy returned approaches the true ground state, it seems to converge to $\approx 5\%$ (i.e. $f \approx 0.05$) above the ground state as we increase the chain strength, for all training sizes ≥ 1000. In this case, we were not able to find the optimal chain strength in a reasonable range of chain strengths, and instead simply took the best we found, $J_F = 6$. As discussed in Sec. 5.7.5, the DW processor suffers from noise sources on the couplers and thermal fluctuations, and it seems that this poses significant challenges for the performance of the quantum annealer. It is possible that even larger chain strengths may resolve the issue, but given the convergence visible in Fig. 5.11, it seems likely that $J_F = 6$ is already near the optimum.

5.7.4 Simulated annealing

Our simulations used $\beta_{\text{init}} = 0.1$ and $\beta_{\text{final}} = 5$, and used a linear annealing schedule (i.e., if we perform S sweeps, we increase β after each sweep by $\frac{\beta_{\text{final}} - \beta_{\text{init}}}{S}$). These parameters have generally performed well in other studies.[3] All SA data in the main text and presented here is at 1000 sweeps, however we also tested SA at 100 sweeps, and found a negligible difference in overall performance, as seen in Fig. 5.12, where the integrated difference of the ROC curves is found to be statistically indistinguishable from 0.

5.7.5 Effect of noise on processor

Internal control error (ICE) on the current generation of D-Wave processors is effectively modeled as a Gaussian centered on the problem-specified value of each coupler and local field, with standard deviation 0.025, i.e., a coupler J_{ij} is realized as a value drawn from the distribution $N(J_{ij}, 0.025)$ when one programs a Hamiltonian. Figure 5.13 contains a histogram of the ideal values of the embedded couplers corresponding to connections between logical qubits across all 20 problem instances of 36 variables at 20000 training events. One can see that the ideal distribution has some structure, with two peaks. However, if one resamples values from the Gaussian distribution induced by ICE, one finds that many of the features are washed out completely. This suggests that the explanation for the flattening out of the performance of QA as a function of training size (recall Fig. **??** in the main text) is due to this noise issue. Thus we investigate this next.

Figures 5.14-5.16 tell the story of the scaling of the couplers with training size. Figure 5.14 shows linear scaling of the maximum Hamiltonian coefficient with training size. We observe wider variation at the smallest training sizes, but overall the precision scales linearly with training size. This is confirmed in Figure 5.15, which shows the maximum coefficient normalized by the training size.

Since this value is constant for sufficiently large training sizes, the maximum value scales linearly with size. At first glance, this indeed suggests an explanation for why the performance of QA using the DW levels off as a function of training size: the coupling values pass 20 (half the scale of the errors which is $\approx 1/0.025 = 40$). However, absolute numbers are not necessarily informative, and Fig. 5.16 dispels this explanation.

Figure 5.16 shows the ratio of the median coefficient to the maximum coefficient, thereby showing the scale of *typical* Hamiltonian coefficients on the DW prior to rescaling for chain strength (which for the most of the data here would reduce the magnitude by a further factor of 6).

Since all the different types of coefficient ratios are constant with system size, we have effectively no scaling with training size of the precision of the couplers. This means that the scaling of precision with training size cannot explain the saturation of performance with increasing training size.

However, the magnitudes here are quite small, and so once one accounts for rescaling the energies, typical couplers are expected to be subject to a significant amount of noise, even causing them to change sign. This effect likely explains, at least in part, the difficulties the DW has in finding the true ground state, as discussed above and seen in Fig. 5.11, where even at the largest chain strength we still find that the DW's typical minimum energy is $\sim 5\%$ above the ground state energy.

5.7.6 Sensitivity to variation of the parameters of weak classifier construction

When constructing the weak classifiers, we choose to define v_{cut} as the 70th percentile of the signal distribution. This choice is arbitrary. To test the effect of this value on the classifier performance we use identical training sets and values of both 60% and 80% and compare them to our primary estimate of 70%. The results for both the minimum energy returned ($f = 0$) and $f = 0.05$ for each are shown in Fig. 5.17.

Note that every training set has the same ground state configuration at 70% and 80%. The ROCs and AUROC are then invariant across a wide range of v_{cut} values.

Figure 5.7 reproduces Fig. 5.4 from the main text, but also shows the AUROC for SA's optimal classifier (by energy) for various values of the regularization parameter λ. We find no significant variation, with the major features of SA being stable, namely the advantage at small training size and the saturation at around an AUROC of ≈ 0.64.

5.7.7 Difference between ROC curves plots

We show differences between ROC curves for various algorithms in Figs. 5.18-5.23. These form the basis for Fig. 5.4 in the main text, which gives the integral of the difference over signal efficiency. Figures 5.18 and 5.19 show the difference in background rejection $r_B^{\text{DW}} - r_B^{\text{SA}}$ as a function of the signal efficiency

for $f = 0$ and $f = 0.05$, respectively. For $f = 0$ DW and SA are indistinguishable to within experimental error. For $f = 0.05$ SA slightly outperforms DW in the range of low signal efficiencies for training sizes ≥ 5000. The primary conclusion to draw from these plots is that SA differs from DW by roughly one standard deviation or less across the whole range, even though DW for training sizes larger than 100 struggles to find states within less than 5% of the ground state energy. This suggests a robustness of QAML, which (if it generalizes to other problems) significantly improves the potential to exploit physical quantum annealers to solve machine learning problems and achieve close-to-optimal classifier performance, even in the presence of significant processor noise.

Figures 5.20 and 5.21 show the ROC difference between DW and DNN and DW and XGB at $f = 0$, respectively. The two cases have broadly similar shapes. One clearly sees that QAML on DW outperforms DNN and XGB at the smallest training size in a statistically significant manner, but that the trend reverses for sizes ≥ 5000. Note, that at the scale of these diagrams, the gap between $f = 0$ and $f = 0.05$ is negligible.

Figures 5.22 (SA) and 5.23 (DW) show the difference between $f = 0$ and $f = 0.05$. SA and DW exhibit broadly similar behavior, with an improvement with excited states of $\approx 0.4\%$ in background rejection for SA and of $\approx 0.2\%$ for DW. The improvement increases with training size and is slightly larger for SA than DW (though this difference is likely simply noise, as it is less than half the standard deviation of each distribution). It should be noted that since QAML's comparative advantage against other techniques appears to be in the realm of small training sizes. However, this is the same range where including excited states has no benefit.

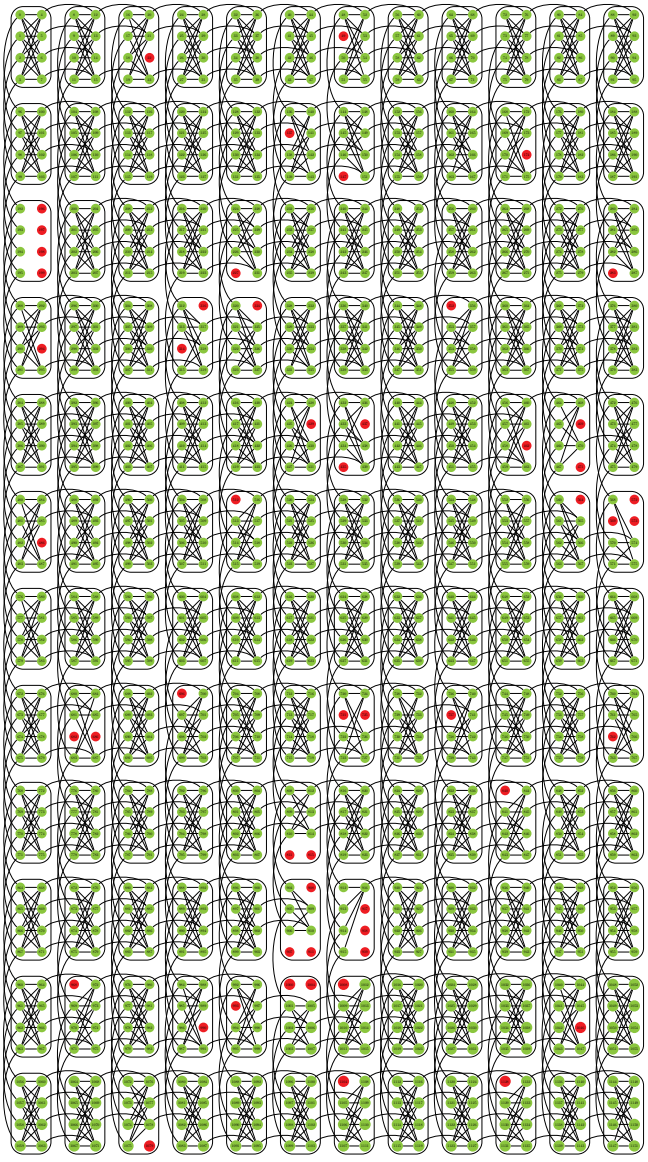


Figure 5.8: An 1152 qubit Chimera graph, partitioned into a 12×12 array of 8-qubit unit cells, each unit cell being a $K_{4,4}$ bipartite graph. Inactive qubits are marked in red, active qubits in green. There are a total of 1098 active qubits in the DW processor used in our experiments. Black lines denote active couplers.

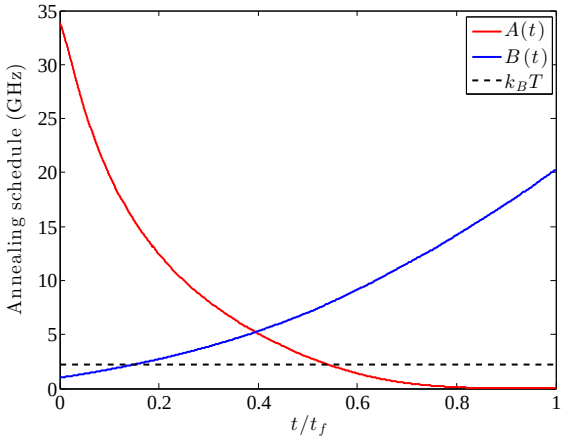


Figure 5.9: Annealing schedule used in our experiments.

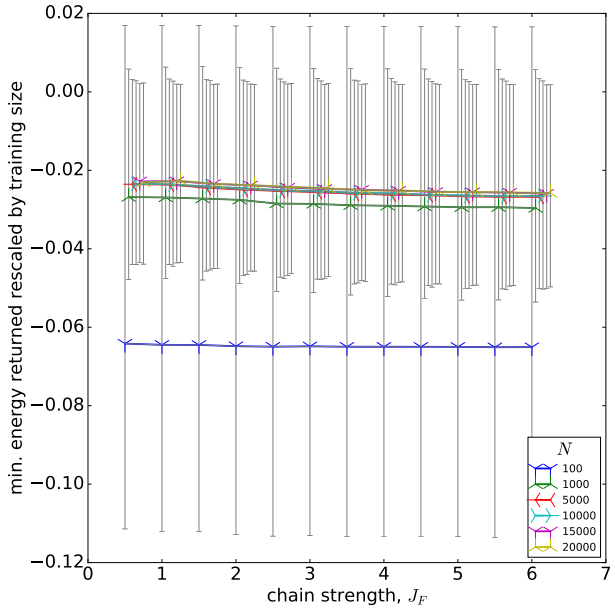


Figure 5.10: A plot of the minimum energy returned by the DW as a function of chain strength, rescaled by the number of training samples. I.e., for training size N, we plot E_m/N for minimum return energy E_m, where N is given in the legend.

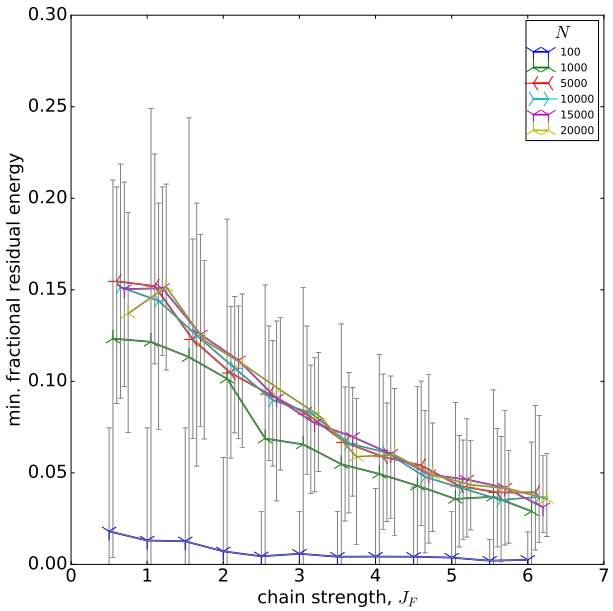


Figure 5.11: Plot of $(E_m - E_0)/E_0$ for minimum energy returned E_m and true ground state energy E_0, i.e., the minimum fractional reserve energy, averaged over the training sets, for each size and chain strength.

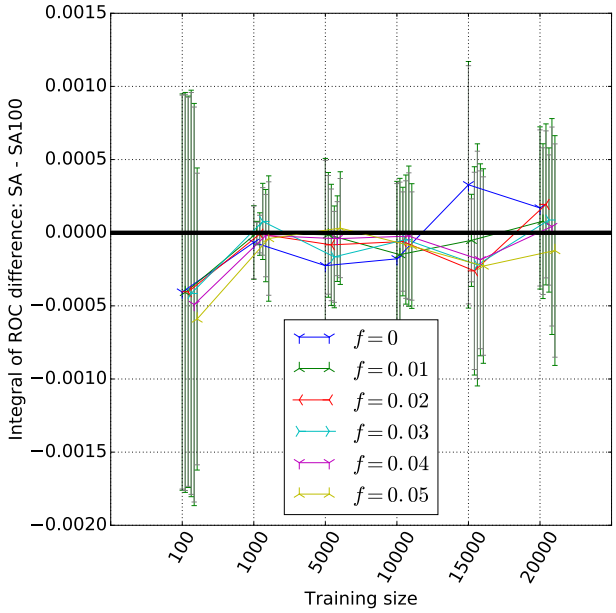
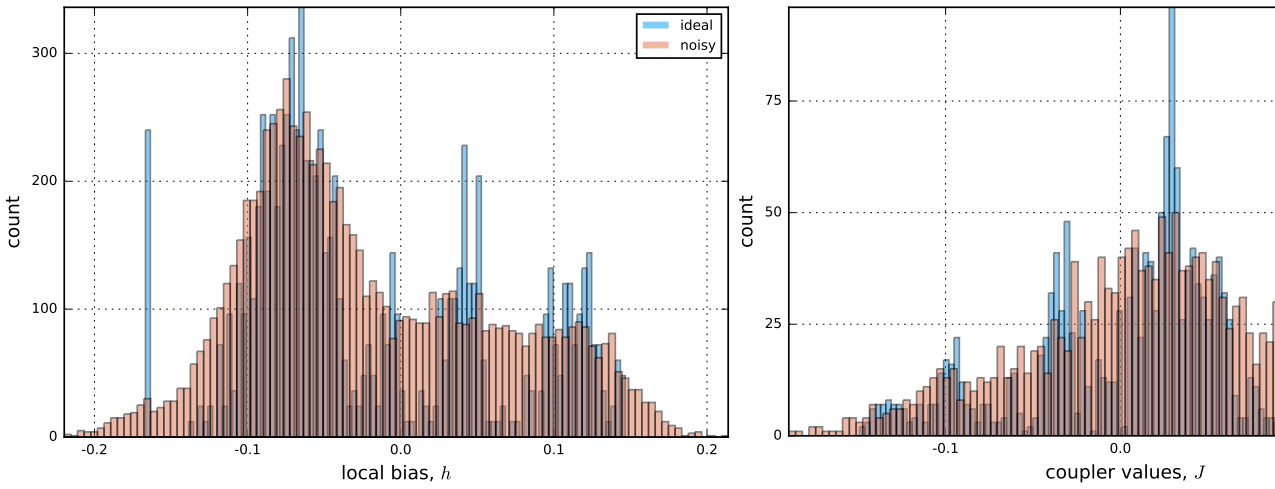


Figure 5.12: The integral of the difference of the ROC curves, i.e., the area between the ROC curves, for SA and SA100 for various thresholds of the energy and training size. SA at 100 and 1000 sweeps are effectively identical by this benchmark.



(a) (b)

Figure 5.13: Histograms for the true (peaked) distribution of local biases and couplers, and the same distribution subject to point-wise Gaussian noise with zero mean and standard deviation 0.025, which is approximately the magnitude of errors on the DW couplers.

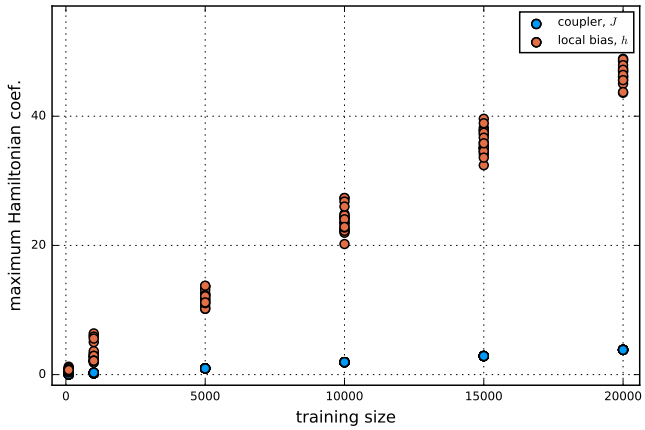


Figure 5.14: The maximum local bias and coupler term in the Hamiltonian across training sizes and training sets.

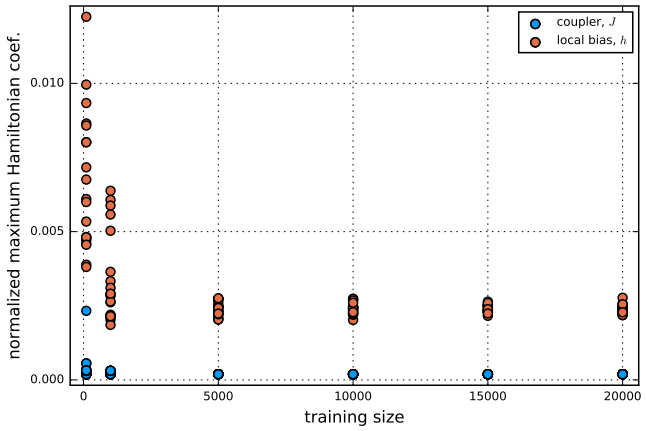


Figure 5.15: The maximum local bias and coupler term in the Hamiltonian across training sizes and training sets, normalized by the number of events in the training set. This makes it clear that the scaling of the Hamiltonian coefficients is linear in the training size, for training sizes ≥ 5000.

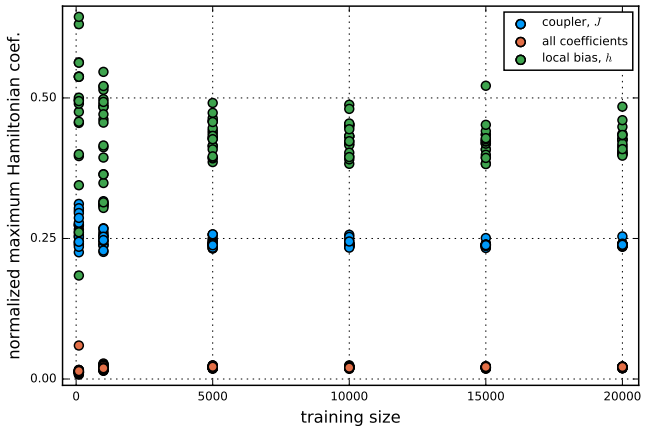


Figure 5.16: The ratio of the median coefficient by the maximum coefficient for the non-zero local biases, couplers, and both taken together.

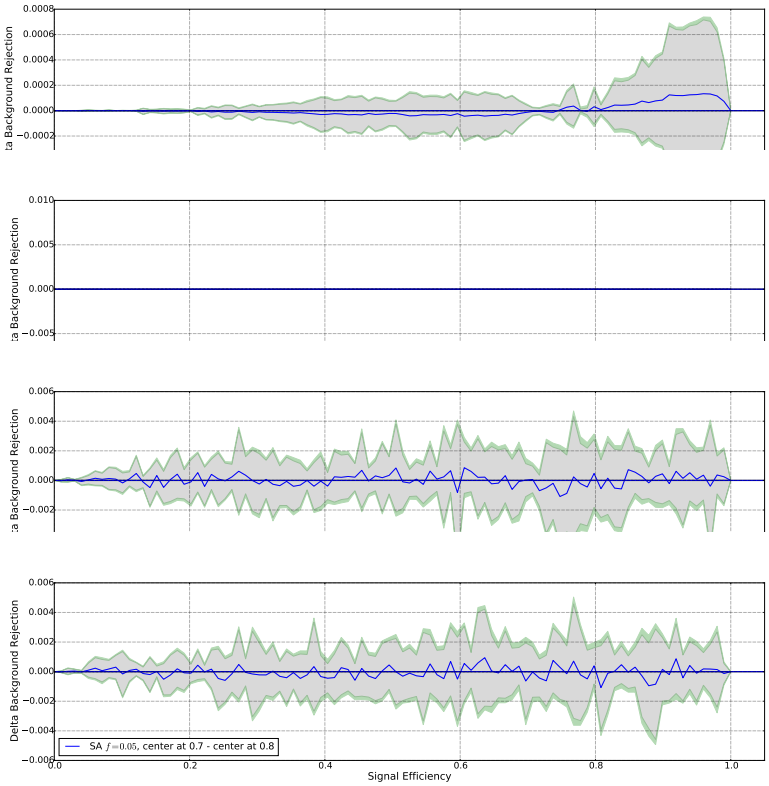


Figure 5.17: Difference between the ROC curve for SA at v_{cut} at the xth percentile during weak classifier construction and the curve using the yth percentile during the same for the ground state configuration. (a) $x = 70$, $y = 60$, $f = 0$. (b) $x = 70$, $y = 80$, $f = 0$. (c) $x = 70$, $y = 60$, $f = 0.05$. (d) $x = 70$, $y = 80$, $f = 0.05$.

(d)

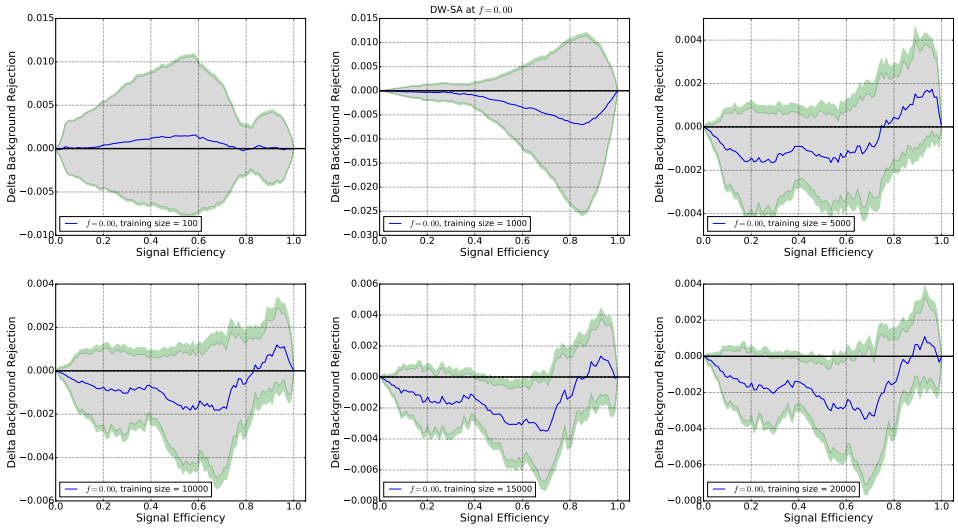


Figure 5.18: Difference between the ROC curves for SA and DW using the minimum energy returned.

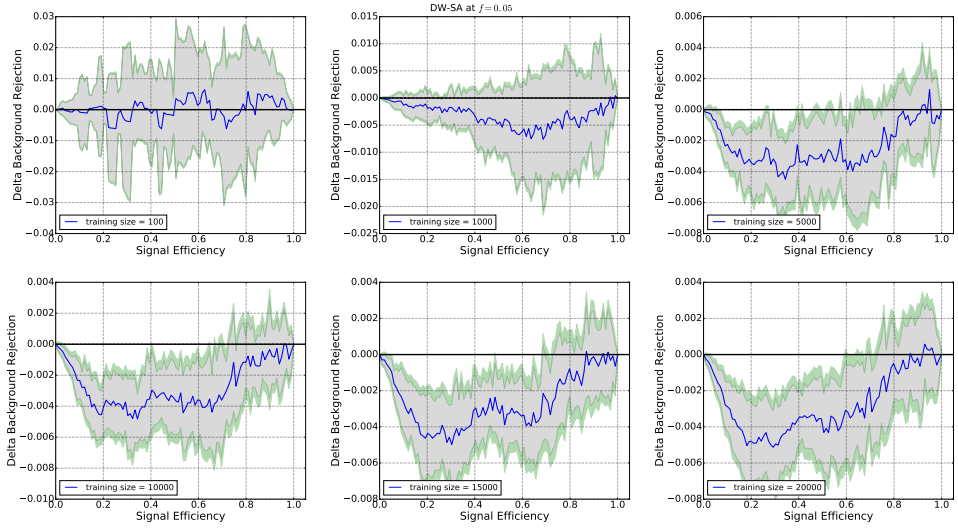


Figure 5.19: Difference between the ROC curves for SA and DW using all states within 5% of the minimum return energy.

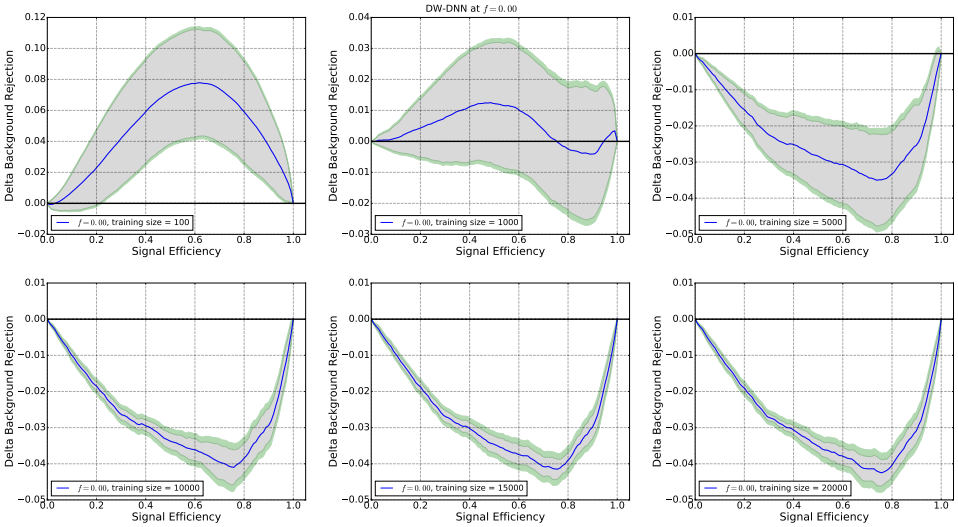


Figure 5.20: Difference between the ROC curves for DW and DNN using the minimum energy configuration from DW.

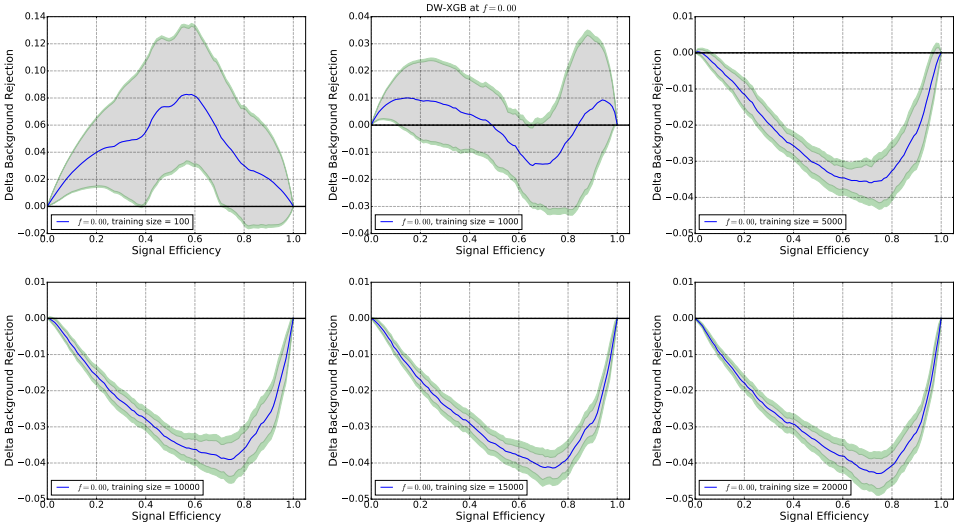


Figure 5.21: Difference between the ROC curves for DW and XGB using the minimum energy configuration from DW.

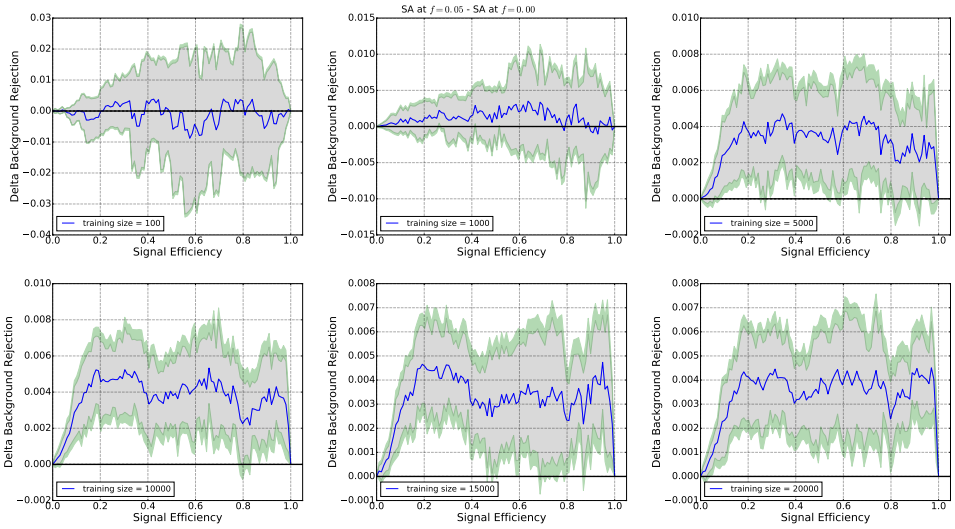


Figure 5.22: Difference between the ROC curves between the true ground state configuration and the $f = 0.05$ composite classifier from SA.

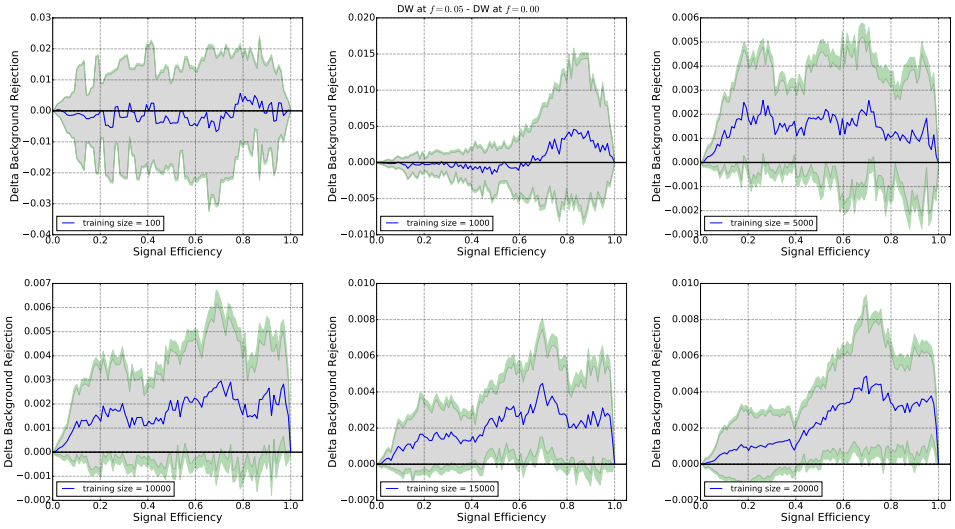


Figure 5.23: Difference between the ROC curves between the minimum energy state returned by DW and the $f = 0.05$ composite classifier from DW.

Chapter 6

Conclusions: Principles of benchmarking

Rather than simply reiterate each chapter that has come before, in this concluding chapter I will instead try to simply state a set of basic principles and lessons that have been learned by the community and that we have encountered in the previous chapters that cover benchmarking quantum annealers in application settings. Before we do that, however, I wish to introduce one last, short, review of the progress that has been made in the field of benchmarking quantum annealers, by those other than myself.

6.1 Recent progress

An interesting critique of the scaling results presented in chapter 2 was made in Ref. [93], which argued that random Ising instances restricted to the Chimera graph are "too easy", essentially since their phase space exhibits only a zero-temperature transition. This would imply that classical thermal algorithms such as SA see a simple energy landscape with a single global minimum throughout the entire anneal (except perhaps at the very end as the simulation temperature is lowered to near zero), instead of the usual glassy landscape with many local traps associated with hard optimization problems. This work highlighted the importance of a careful design of benchmark problems, to ensure that classical solvers would not find them trivial. Of course, it should be stressed that quantum speedup is always relative, and it can be observed even when efficient (polynomial-time) classical algorithms exist, as in, e.g., the solution of linear systems of equations [165]. In light of this one may interpret the message of Ref. [93] to mean that a quantum speedup might not be *detectable* over a finite range of problem sizes if the problem is classically easy, since the difference between the quantum and classical scaling is too subtle to be statistically significant.

Before we turn to a discussion of the evidence for a limited quantum speedup,

we first briefly discuss alternatives to the TTS as a performance measure. One such alternative is the time-to-target (TTT), i.e., the total time required by a solver to reach the target energy at least once with a desired probability, assuming each run takes a fixed time [81]. This reduces to the TTS if the target is the ground state. A unified approach that includes a variety of other measures was presented in Ref. [82], drawing upon optimal stopping theory, specifically the so-called "house-selling" problem [166]. Within this framework one answers the question of how long, *given a particular cost for each sample drawn from a solver*, one should sample in order to maximize one's reward, analogously to the decision problem about when to sell one's house given that bids accrue over time but that waiting longer carries a higher monetary cost. This allows the TTS and TTT, among other measures, to be shown to be specific choices of the cost and reward functions. The optimal stopping framework also paves the way for a more detailed comparison between quantum and classical approaches and the tradeoffs of each, as by altering the cost per sample one can see the impact of the distribution over states (rather than just the ground state) for the various solvers. Optimal stopping is appropriate for applications where finding the minimum energy is not strictly the most important consideration for the application, such as many machine learning contexts and even various business-origin optimization problems. In those cases, there is a tradeoff between the cost to perform the computation and the benefit from receiving a result. Tests were performed demonstrating the optimal stopping approach with a DW2X device (with 1098 qubits) on frustrated loop problems much like those in Ref. [3], demonstrating identical scaling (modulo concerns about the lack of an optimal annealing time) to the HFS algorithm at multiple values of the cost to draw a sample, an improvement over the DW2. However, these results could still not qualify as a limited quantum speedup due to the problem of suboptimal annealing times.

This problem was finally overcome in Ref. [167], which for the first time demonstrates an optimal annealing time, and can thus make positive claims about limited quantum speedup. Previous studies could not find an optimal annealing time since a class of problem instances had not been identified for which the shortest available annealing time ($20\mu s$ in the DW2, $5\mu s$ in all other D-Wave devices) was sufficiently short to observe on optimum given the largest problem size that could be tested. Using the D-Wave 2000Q (DW2KQ) device (with 2027 qubits) Ref. [167] demonstrated a simple one-unit cell gadget Hamiltonian which, when added randomly to a constant fraction of the unit cells on top of similar frustrated loop problems as in Ref. [168], resulted in the observation of an optimal annealing time for frustrated loops defined on the hardware graph (also when using the DW2X device), as well as for frustrated loops defined on the logical graph of unit cells (each unit cell then being bound together tightly as a pseudo-spin in the physical problem, modulo the gadget Hamiltonian). For the latter, logical-planted instances, the DW2KQ exhibited a statistically significant scaling advantage over both single-spin-flip SA and SVMC. These results amount to the first observation of a limited quantum speedup, since the existence of an optimal annealing time was certified. However, this did not amount to an

unqualified quantum speedup since the DW2KQ's scaling was worse than the HFS algorithm, unit-cell cluster-flip SA, and SQA, which was found to have the best scaling. Nevertheless, this result paves the way towards future demonstrations of problems with optimal annealing times and hence certifiable scaling, a necessary requirement for any type of scaling speedup claim. However, even this may not be sufficient since other quantities remain that must eventually be optimized, such as the annealing schedule, which is known to play a crucial role in provable quantum speedups (specifically the Grover search problem [169, 170]), and can conversely be used to potentially overturn (limited) quantum speedup claims.

This provides a reasonable update of the status of benchmarking these systems, and is included here for completeness.

6.2 Guidelines for benchmarking quantum annealing and related noisy quantum computational devices

Now, finally, let us turn to a statement of principles, of sorts, for benchmarking quantum annealers.

Resource use comparisons

It is vitally important to carefully account for resource use, lest one be led astray with a fake speedup. As was discussed in the chapters 2 and 4, classical resources should, in general, scale at least linearly with the system size of one's quantum computer. This is especially true for annealing type devices, as we have seen that annealing type algorithms are often their biggest competitors and those algorithms are typically very parallelizable. While this can be expected to ward off only linear or near-linear polynomial advantages for annealers, as was seen in chapter 4, this can be decisive, particularly in the case of non-asymptotic problem sizes.

Parameter optimization

It is also necessary to optimize the parameters of all solvers as best one can to be able to make any potential claims about speedup or advantage of one over another. In particular, quantum annealing requires a demonstration of an optimal annealing time for a fixed schedule before any definitive conclusion can be drawn about a quantum speedup. We saw this in chapter 2 empirically, and there is further proof (with a proof) of this in chapter 4. More generally, optimizing all known free parameters is almost certainly necessary to demonstrate a quantum speedup which will hold up to scrutiny. Ultimately, if there are free parameters of your various algorithms that have not been optimized, one is not able to make any hard claims about performance. Simulated annealing with an arbitrarily

chosen number of sweeps is obviously going to fail, in general. Parallel tempering with a poorly chosen temperature spacing will also fail. In classification contexts, using suboptimal hyperparemeters of the learning algorithm will yield poor results.

While it is true that for many algorithms one is almost always simply unable to properly optimize all the free parameters due to their enormous number (parallel tempering is an example), one should still choose best-in-class and/or standard methods for selecting said parameters. If a practioner is unfamiliar with such methods, as was the case for my work on the Higgs problem in chapter 5 for XGBoost and DNNs, one should both familiarize oneself as well as consult subject matter/application area experts (as was done in that case). In general, if one leaves free parameters unoptimized or only partially optimized, one can only make lower-bound claims on whatever performance metrics are of interest. This leads to...

Distinguish between types of quantum speedup and take care in algorithm choice

One must distinguish between different types of quantum speedup. Comparisons between a quantum computational device and a single other solver are inherently limited to a demonstration of a "potential quantum speedup". In general, studies of this kind (comparing against only single solvers) have little value to the broader community, as it is simply far too easy to select a "broken" algorithm that is clearly suboptimal for the task at hand. An example is using single-spin flip SA in problems with clusters, as was done in [76], though there have been any number of similar such experiments. The only exception to this is if one is comparing against an algorithm already considered to be best-in-class (for Chimera-structured problems, that would likely be the HFS algorithm, or perhaps PT with isoenergetic cluster moves), and even then one should still, for the community's sake, study other potentially competing algorithms.

To go further, one must be sure to compare performance against a suite of algorithms, in particular those that mimic the device to some degree (such as SA or SQA). A speedup against such solvers would be considered a "limited quantum speedup". (Note: This was also done in [76], as they did compare against PIMC, and did not find any evidence of speedup.) PIMC or some other variant of quantum Monte Carlo is vital in cases of potential quantum speedup, as it has been found to often correlate well with quantum annealers (see references [13, 2] for instance). Finally, if there is a consensus about the solvers that are the best at the original algorithmic task, then a speedup against such solvers would be considered an unqualified "quantum speedup". This would be a game-changing result, but as yet has never been discovered in this space.

Analyze full pipelines

As was done in chapter 5, one should compare one's quantum algorithm not only with classical alternatives to one's calls to the quantum device itself, but

also to the best available classical methods of solving the problem. In that chapter, had we merely introduced QAML and compared DW to SA as a way of solving/sampling from the corresponding Ising Hamiltonian, the study would be been entirely pointless — we would have had no way of knowing if our performance was good or bad, as we would not have had anything to compare QAML, itself, to. Similarly, had we merely compared raw QAML with DW as a solver with DNNs and XGBoost, we would have been unable to make any statements about whether it was the quantum annealer, or merely the QAML algorithm, that produced our performance results. This is a general problem – merely mapping a problem into an Ising model and solving it is insufficient, unless one both tests replacements for the quantum device and algorithms for the original problem class (whether the class be solving Ising models [2], solving SAT problems[14], graph coloring[171], job shop scheduling[171], etc.). In general, this is another case where consultation between our community and subject matter/application area experts is key for rigorous and useful benchmarking studies.

Gauge-averaging

Users of such quantum computational devices should perform something akin to gauge averaging in order to effectively estimate performance, by averaging over many different mappings from the logical problem to physical states, at least so long as the devices are not fully error corrected. Nuisance parameters, local biases, Hamiltonian dependent interactions, etc. are all expected to continue to be an issue long into the future, in the absence of error correction, and as long as that is true, the use of gauges (ie sampling over efficiently applicable maps from the logical problem to physical states) is key. Given that there is typically no good distribution for problem hardness as a function of this ensemble of mappings, nonparameteric techniques are appropriate, as was described in detail in 3. This leads to...

Take your state of knowledge seriously

As was discussed in 3, and again to a degree in 5, practioners should think carefully about what they want to learn from their experiment (time to solution, probability of success, order parameters of TTS over an instance set, classifier performance, etc.) and how the results of the experiment effect their knowledge. As stated above, for things like estimating TTS, the focus of chapter 3, a theoretically well-founded and simple nonparametric approach is the Bayesian bootstrap over gauges, which may be readily extended to estimates of order parameters of TTS over instances. In other cases, something like the reweighting of our very large event test set with Poisson weights is more appropriate as it simulates the physical process which generates the statistical error. Moreover, as stated in 3, by taking one's state of knowledge seriously, one can then readily employ optional stopping with a terminating condition on one's posterior density coefficient of variation to, at times, save enormous amounts of resources

(as demonstrated in the aforementioned chapters simulation studies, an order of magnitude or more). I'll also refer the reader to the work by Vinci & Lidar [82] on optimal stopping for cases where one can define meaningful cost and reward functions (typically, I expect this will be limited to real-world business use cases).

Choice of benchmark problem and the meaning of "success"

Finally, choice of benchmark problem is key, and should be made with an eye toward the day when classical machines are vastly outpaced by quantum devices. For example, the transition from random Ising problems to frustrated loop/planted solutions problems seen between studies presented in chapters 2 and 4 was forced by the need to have reasonable benchmarks for devices so large that classical systems cannot solve them in a human lifetime. If no analogue to planted solutions is feasible in the area of interest, the previous sections' exhortations about testing a wide array of solvers, optimizing parameters, and consulting with application area experts become all the more important, as one will be forced to resort to the use of "the best solution found" as one's definition of a "success", and one doesn't want to have missed an obvious or readily available algorithm that could find better solutions than any solvers one tests.

If the reader adheres to these principles and applies them well in their use cases, and these become standard/common knowledge within the community, I believe it will significantly improve the quality, realiability, and usefulness of future benchmarking studies and, in doing so, accelerate progress in the field.

Bibliography

[1] J. Job, D. Lidar, *Quantum Science and Technology* **3**, 030501 (2018).

[2] T. F. Rønnow, *et al.*, *Science* **345**, 420 (2014).

[3] I. Hen, *et al.*, *Phys. Rev. A* **92**, 042325 (2015).

[4] A. Mott, J. Job, J.-R. Vlimant, D. Lidar, M. Spiropulu, *Nature* **550**, 375 (2017).

[5] F. Barahona, *J. Phys. A: Math. Gen* **15**, 3241 (1982).

[6] E. Farhi, *et al.*, *Science* **292**, 472 (2001).

[7] R. Harris, *et al.*, *Phys. Rev. B* **82**, 024511 (2010).

[8] M. W. Johnson, *et al.*, *Superconductor Science and Technology* **23**, 065004 (2010).

[9] A. J. Berkley, *et al.*, *Superconductor Science and Technology* **23**, 105014 (2010).

[10] M. W. Johnson, *et al.*, *Nature* **473**, 194 (2011).

[11] V. Choi, *Quant. Inf. Proc.* **10**, 343 (2011).

[12] C. C. McGeoch, C. Wang, *Proceedings of the 2013 ACM Conference on Computing Frontiers* (2013).

[13] S. Boixo, *et al.*, *Nat. Phys.* **10**, 218 (2014).

[14] S. Santra, G. Quiroz, G. V. Steeg, D. A. Lidar, *New J. of Phys.* **16**, 045006 (2014).

[15] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, *Science* **220**, 671 (1983).

[16] S. V. Isakov, I. N. Zintchenko, T. F. Rønnow, M. Troyer, *Computer Physics Communications* **192**, 265 (2015).

[17] C. J. Geyer, *Computing Science and Statistics Proceedings of the 23rd Symposium on the Interface*, E. M. Keramidas, ed. (American Statistical Association, New York, 1991), p. 156.

[18] D. J. Earl, M. W. Deem, *Physical Chemistry Chemical Physics* **7**, 3910 (2005).

[19] H. G. Katzgraber, S. Trebst, D. A. Huse, M. Troyer, *J. Stat. Mech.* **2006**, P03018 (2006).

[20] F. Hamze, N. de Freitas, *UAI*, D. M. Chickering, J. Y. Halpern, eds. (AUAI Press, Arlington, Virginia, 2004), pp. 243–250.

[21] A. Selby, *arXiv:1409.3934* (2014).

[22] R. Martoňák, G. E. Santoro, E. Tosatti, *Phys. Rev. B* **66**, 094203 (2002).

[23] B. Heim, T. F. Rønnow, S. V. Isakov, M. Troyer, *Science* **348**, 215 (2015).

[24] E. Crosson, A. W. Harrow, *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)* pp. 714–723 (2016).

[25] S. W. Shin, G. Smith, J. A. Smolin, U. Vazirani, *arXiv:1401.7087* (2014).

[26] K. M. Zick, O. Shehab, M. French, *arXiv:1503.06453* (2015).

[27] D. Venturelli, D. J. J. Marchand, G. Rojo, *arXiv:1506.08479* (2015).

[28] E. G. Rieffel, *et al.*, *Quantum Information Processing* **14**, 1 (2015).

[29] G. Rosenberg, *et al.*, *arXiv:1508.06182* (2015).

[30] A. Lucas, *Front. Phys.* **2**, 5 (2014).

[31] V. Choi, *Quant. Inf. Proc.* **7**, 193 (2008).

[32] C. Klymko, B. D. Sullivan, T. S. Humble, *Quant. Inf. Proc.* **13**, 709 (2014).

[33] Z. Zhu, A. J. Ochoa, H. G. Katzgraber, *Phys. Rev. Lett.* **115**, 077201 (2015).

[34] T. Albash, W. Vinci, A. Mishra, P. A. Warburton, D. A. Lidar, *Phys. Rev. A* **91**, 042314 (2015).

[35] T. Albash, T. F. Rønnow, M. Troyer, D. A. Lidar, *Eur. Phys. J. Spec. Top.* **224**, 111 (2015).

[36] S. Boixo, *et al.*, *Nat Commun* **7** (2016).

[37] U. Wolff, *Phys. Rev. Lett.* **62**, 361 (1989).

[38] B. W. Reichardt, F. Unger, U. Vazirani, *Nature* **496**, 456 (2013).

[39] I. L. Chuang, M. A. Nielsen, *Journal of Modern Optics* **44**, 2455 (1997).

[40] M. Mohseni, A. T. Rezakhani, D. A. Lidar, *Phys. Rev. A* **77**, 032322 (2008).

[41] R. Blume-Kohout, *et al.*, *arXiv:1310.4492* (2013).

[42] D. Greenbaum, *arXiv:1509.02921* (2015).

[43] A. M. Childs, I. L. Chuang, D. W. Leung, *Physical Review A* **64**, 012314 (2001).

[44] R. Blume-Kohout, *et al.*, *Nature Communications* **8**, 14485 (2017).

[45] T. Kadowaki, H. Nishimori, *Phys. Rev. E* **58**, 5355 (1998).

[46] A. Das, B. K. Chakrabarti, *Rev. Mod. Phys.* **80**, 1061 (2008).

[47] E. Farhi, *et al.*, *Science* **292**, 472 (2001).

[48] W. M. Kaminsky, S. Lloyd, *Quantum Computing and Quantum Bits in Mesoscopic Systems*, A. Leggett, B. Ruggiero, P. Silvestrini, eds. (Kluwer Academic/Plenum Publ., 2004).

[49] W. M. Kaminsky, S. Lloyd, T. P. Orlando, *arXiv:quant-ph/0403090* (2004).

[50] T. Albash, D. A. Lidar, *arXiv:1611.04471* (2016).

[51] A. J. Berkley, *et al.*, *Phys. Rev. B* **87**, 020502 (2013).

[52] J. Preskill, *arXiv:1203.5813* (2012).

[53] S. Flammia, The Quantum Pontiff blog: "Quantum Advantage" (2017).

[54] S. Aaronson, L. Chen, *arXiv:1612.05903* (2016).

[55] M. J. Bremner, A. Montanaro, D. J. Shepherd, *Physical Review Letters* **117**, 080501 (2016).

[56] E. Farhi, A. W. Harrow, *arXiv:1602.07674* (2016).

[57] X. Gao, S.-T. Wang, L. M. Duan, *Physical Review Letters* **118**, 040502 (2017).

[58] S. Boixo, *et al.*, *arXiv:1608.00263* (2016).

[59] B. Fefferman, M. Foss-Feig, A. V. Gorshkov, *arXiv:1701.03167* (2017).

[60] P. W. Shor, *SIAM J. Comput.* **26**, 1484 (1997).

[61] E. Farhi, J. Goldstone, S. Gutmann, *arXiv:1412.6062* (2014).

[62] K.-W. Yip, T. Albash, D. Lidar, Quantum trajectories for time-dependent adiabatic master equations, in preparation (2017).

[63] T. Lanting, *et al.*, *Phys. Rev. X* **4**, 021041 (2014).

[64] G. Vidal, R. F. Werner, *Phys. Rev. A* **65**, 032314 (2002).

[65] F. M. Spedalieri, *Phys. Rev. A* **86**, 062311 (2012).

[66] T. Albash, I. Hen, F. M. Spedalieri, D. A. Lidar, *Physical Review A* **92**, 062328 (2015).

[67] T. Albash, S. Boixo, D. A. Lidar, P. Zanardi, *New Journal of Physics* **14**, 123016 (2012).

[68] S. Boixo, T. Albash, F. M. Spedalieri, N. Chancellor, D. A. Lidar, *Nat. Commun.* **4**, 2067 (2013).

[69] J. A. Smolin, G. Smith, *Frontiers in Physics* **2**, 52 (2014).

[70] L. Wang, *et al.*, *arXiv:1305.5837* (2013).

[71] S. W. Shin, G. Smith, J. A. Smolin, U. Vazirani, *arXiv:1404.6499* (2014).

[72] P. J. D. Crowley, A. G. Green, *Physical Review A* **94**, 062106 (2016).

[73] J. Brooke, D. Bitko, T. F., Rosenbaum, G. Aeppli, *Science* **284**, 779 (1999).

[74] J. Brooke, T. F. Rosenbaum, G. Aeppli, *Nature* **413**, 610 (2001).

[75] M. W. Johnson, *et al.*, *Nature* **473**, 194 (2011).

[76] V. S. Denchev, *et al.*, *Phys. Rev. X* **6**, 031015 (2016).

[77] S. Mandrà, Z. Zhu, W. Wang, A. Perdomo-Ortiz, H. G. Katzgraber, *Physical Review A* **94**, 022337 (2016).

[78] T. Monz, *et al.*, *Phys. Rev. Lett.* **106**, 130506 (2011).

[79] J. G. Bohnet, *et al.*, *Science* **352**, 1297 (2016).

[80] Catherine C. McGeoch, *A Guide to Experimental Algorithmics* (Cambridge University Press, Cambride, UK, 2012).

[81] J. King, S. Yarkoni, M. M. Nevisi, J. P. Hilton, C. C. McGeoch, *arXiv:1508.05087* (2015).

[82] W. Vinci, D. A. Lidar, *Physical Review Applied* **6**, 054016 (2016).

[83] D. Venturelli, *et al.*, *Phys. Rev. X* **5**, 031040 (2015).

[84] D. B. Rubin, *et al.*, *The annals of statistics* **9**, 130 (1981).

[85] L. K. Grover, *Phys. Rev. Lett.* **79**, 325 (1997).

[86] C. Bennett, E. Bernstein, G. Brassard, U. Vazirani, *SIAM Journal on Computing* **26**, 1510 (1997).

[87] A. Papageorgiou, J. F. Traub, *Phys. Rev. A* **88**, 022316 (2013).

[88] S. Kirkpatrick, C. D. Gelatt, M. P. Vecchi, *Science* **220**, 671 (1983).

[89] R. Martoňák, G. E. Santoro, E. Tosatti, *Phys. Rev. B* **66**, 094203 (2002).

[90] G. E. Santoro, R. Martoňák, E. Tosatti, R. Car, *Science* **295**, 2427 (2002).

[91] T. F. Rønnow, *et al.*, Supplementary material for "Defining and detecting quantum speedup".

[92] R. D. Somma, D. Nagaj, M. Kieferová, *Phys. Rev. Lett.* **109**, 050501 (2012).

[93] H. G. Katzgraber, F. Hamze, R. S. Andrist, *Phys. Rev. X* **4**, 021008 (2014).

[94] K. L. Pudenz, T. Albash, D. A. Lidar, *Nat. Commun.* **5**, 3243 (2014).

[95] S. Isakov, I. Zintchenko, T. Rønnow, M. Troyer, *arXiv:1401.1084* (2014).

[96] G. Bhanot, D. Duke, R. Salvador, *J. Stat. Phys.* **44**, 985 (1986).

[97] H.-O. Heuer, *Comput. Phys. Commun.* **59**, 387 (1990).

[98] R. Dechter, *Artificial Intelligence* **113**, 41 (1999).

[99] A. P. Young, H. G. Katzgraber, *Phys. Rev. Lett.* **93**, 207203 (2004).

[100] M. S. Sarandy, D. A. Lidar, *Phys. Rev. Lett.* **95**, 250503 (2005).

[101] E. T. Jaynes, *Probability theory: The logic of science* (Cambridge university press, 2003).

[102] B. Efron, *Breakthroughs in statistics* (Springer, 1992), pp. 569–593.

[103] T. S. Ferguson, *The annals of statistics* pp. 209–230 (1973).

[104] D. A. Berry, R. Christensen, *The Annals of Statistics* pp. 558–568 (1979).

[105] L. Kuo, *SIAM Journal on Scientific and Statistical Computing* **7**, 60 (1986).

[106] A. E. Gelfand, A. Kottas, *Journal of Computational and Graphical Statistics* **11**, 289 (2002).

[107] L. Wang, D. B. Dunson, *Journal of Computational and Graphical Statistics* **20**, 196 (2011).

[108] D. G. Mayo, M. Kruse, *Foundations of bayesianism* (Springer, 2001), pp. 381–403.

[109] L. Mendo, J. M. Hernando, *Communications, IEEE Transactions on* **54**, 231 (2006).

[110] K. Binder, A. P. Young, *Reviews of Modern Physics* **58**, 801 (1986).

[111] H. Nishimori, *Statistical Physics of Spin Glasses and Information Processing: An Introduction* (Oxford University Press, Oxford, UK, 2001).

[112] P. I. Bunyk, *et al.*, *IEEE Transactions on Applied Superconductivity* **24**, 1 (Aug. 2014).

[113] M. Mezard, G. Parisi and M.A. Virasoro, *Spin Glass Theory and Beyond*, World Scientific Lecture Notes in Physics (World Scientific, Singapore, 1987).

[114] W. Barthel, *et al.*, *Phys. Rev. Lett.* **88**, 188701 (2002).

[115] F. Krzakala, L. Zdeborová, *Phys. Rev. Lett.* **102**, 238701 (2009).

[116] S. Bravyi, B. Terhal, *SIAM Journal on Computing* **39**, 1462 (2009).

[117] B. Bollobás, C. Borgs, J. T. Chayes, J. H. Kim, D. B. Wilson, *Random Struct. Algorithms* **18**, 201 (2001).

[118] A. Selby, D-wave: comment on comparison with classical computers, http://tinyurl.com/dwave-vs-classical (2013).

[119] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, L. Troyansky, *Nature* **400**, 133 (1999).

[120] M. Amin, *arXiv:1503.04216* (2015).

[121] T. Albash, D. A. Lidar, *Phys. Rev. A* **91**, 062320 (2015).

[122] T. Neuhaus, *arXiv:1412.5460* (2014).

[123] T. Albash, S. Boixo, D. A. Lidar, P. Zanardi, *New J. of Phys.* **14**, 123016 (2012).

[124] A. M. Childs, E. Farhi, J. Preskill, *Phys. Rev. A* **65**, 012322 (2001).

[125] A. D. King, C. C. McGeoch, *arXiv:1410.2628* (2014).

[126] A. Perdomo-Ortiz, J. Fluegemann, R. Biswas, V. N. Smelyanskiy, *arXiv:1503.01083* (2015).

[127] A. Perdomo-Ortiz, B. O'Gorman, J. Fluegemann, R. Biswas, V. N. Smelyanskiy, *arXiv:1503.05679* (2015).

[128] V. Martin-Mayor, I. Hen, *Scientific Reports* **5**, 15324 EP (2015).

[129] K. L. Pudenz, T. Albash, D. A. Lidar, *Phys. Rev. A* **91**, 042302 (2015).

[130] K. C. Young, R. Blume-Kohout, D. A. Lidar, *Phys. Rev. A* **88**, 062314 (2013).

[131] A. D. King, T. Lanting, R. Harris, *arXiv:1502.02098* (2015).

[132] J. Bezanson, A. Edelman, S. Karpinski, V. B. Shah, *arXiv:1411.1607* (2014).

[133] T. Neuhaus, *arXiv:1412.5361* (2014).

[134] S. Chatrchyan, *et al.*, *Phys. Lett.* **B716**, 30 (2012).

[135] G. Aad, *et al.*, *Phys. Lett.* **B716**, 1 (2012).

[136] H. Neven, V. S. Denchev, G. Rose, W. G. Macready, *arXiv:0811.0416* (2008).

[137] K. L. Pudenz, D. A. Lidar, *Quantum Information Processing* **12**, 2027 (2013).

[138] F. Chollet, Keras, https://github.com/fchollet/keras (2015).

[139] T. Chen, C. Guestrin, *arXiv:1603.02754* (2016).

[140] V. Khachatryan *et al.* (CMS collaboration), *Eur. Phys. J.* **C74**, 3076 (2014).

[141] G. Aad *et al.* (ATLAS collaboration), *Phys. Rev.* **D90**, 112015 (2014).

[142] C. Patrignani, *et al.*, *Chin. Phys.* **C40**, 100001 (2016).

[143] C. Englert, T. Plehn, D. Zerwas, P. M. Zerwas, *Phys. Lett.* **B703**, 298 (2011).

[144] D. E. Morrissey, M. J. Ramsey-Musolf, *New J. Phys.* **14**, 125003 (2012).

[145] D. Buttazzo, *et al.*, *JHEP* **12**, 089 (2013).

[146] C. Adam-Bourdarios, *et al.*, *Journal of Physics: Conference Series* **664**, 072015 (2015).

[147] R. Al-Rfou, *et al.*, *arXiv e-prints* **abs/1605.02688** (2016).

[148] J. A. Hanley, B. J. McNeil, *Radiology* **143**, 29 (1982).

[149] J. Chen, *et al.*, *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)* pp. 2429–2433 (2016).

[150] R. Y. Li, R. Di Felice, R. Rohs, D. A. Lidar, *NPJ quantum information* **4**, 14 (2018).

[151] S. Lloyd, M. Mohseni, P. Rebentrost, *Nat. Phys.* **10**, 631 (2014).

[152] N. Wiebe, A. Kapoor, K. Svore, *Quantum Information & Computation* **15**, 0318 (2015).

[153] G. D. Paparo, V. Dunjko, A. Makmal, M. A. Martin-Delgado, H. J. Briegel, *Phys. Rev. X* **4**, 031002 (2014).

[154] P. Rebentrost, M. Mohseni, S. Lloyd, *Phys. Rev. Lett.* **113**, 130503 (2014).

[155] M. Schuld, I. Sinayskiy, F. Petruccione, *Contemporary Physics* **56**, 172 (2015).

[156] I. Cong, L. Duan, *arXiv:1510.00113* (2015).

[157] J. Biamonte, *et al.*, *arXiv:1611.09347* (2016).

[158] T. Sjöstrand, S. Mrenna, P. Skands, *Journal of High Energy Physics* **2006**, 026 (2006).

[159] T. Gleisberg, *et al.*, *Journal of High Energy Physics* **2009**, 007 (2009).

[160] D. P. Kingma, J. Ba, *CoRR* **abs/1412.6980** (2014).

[161] J. Snoek, H. Larochelle, R. P. Adams, *ArXiv e-prints* (2012).

[162] J. Snoek, Spearmint, https://github.com/HIPS/Spearmint (2012).

[163] W. Vinci, T. Albash, G. Paz-Silva, I. Hen, D. A. Lidar, *Phys. Rev. A* **92**, 042310 (2015).

[164] A. Mishra, T. Albash, D. A. Lidar, *Quant. Inf. Proc.* **15**, 609 (2015).

[165] A. W. Harrow, A. Hassidim, S. Lloyd, *Phys. Rev. Lett.* **103**, 150502 (2009).

[166] T. S. Ferguson, Optimal Stopping and Applications (2008).

[167] T. Albash, D. A. Lidar, *arXiv:1705.07452* (2017).

[168] J. King, *et al.*, *arXiv:1701.04579* (2017).

[169] J. Roland, N. J. Cerf, *Phys. Rev. A* **65**, 042308 (2002).

[170] A. T. Rezakhani, A. K. Pimachev, D. A. Lidar, *Phys. Rev. A* **82**, 052305 (2010).

[171] A. Perdomo-Ortiz, *et al.*, *arXiv preprint arXiv:1708.09780* (2017).