Rodríguez-Laguna, J. y Santalla, S. (2018). Building an adiabatic quantum computer simulation in the classroom. *American Journal of Physics*, 86(5), pp. 360-367.

DOI: https://doi.org/10.1119/1.5021360

Building an adiabatic quantum computer simulation in the classroom

Javier Rodríguez-Laguna
Dto. Física Fundamental, Universidad Nacional de Educación a Distancia (UNED), Madrid 28040, Spain

Silvia N. Santalla
Dto. Física & GISC, Universidad Carlos III de Madrid, Madrid 28911, Spain

(Received 18 August 2017; accepted 2 January 2018)

We present a didactic introduction to adiabatic quantum computation (AQC) via the explicit construction of a classical simulator of quantum computers. This constitutes a suitable route to introduce several important concepts for advanced undergraduates in physics: quantum many-body systems, quantum phase transitions, disordered systems, spin-glasses, and computational complexity theory.
https://doi.org/10.1119/1.5021360

I. INTRODUCTION

Interest in quantum computation[1,2] is increasing lately, since it might be the next quantum technology to take off.[3] Exciting new spaces for exploration have appeared, such as the IBM Quantum Experience,[4,5] where a few-qubit computers can be programmed remotely. Being a crossroad for physicists and computer scientists, it presents a steep learning curve for newcomers. Among the different flavors, we have chosen adiabatic quantum computation (AQC)[6–8] and developed a gentle introduction which can be delivered in two sessions, a theoretical and a practical one. The aim is to enable the readers to write their own code to simulate a quantum computer. The proposed simulation is very light computationally, and can run even on a small laptop.[9]

The reasons to include quantum computation in the undergraduate syllabus for physicists are manifold. Beyond the promise to develop true *quantum supremacy*, i.e., computational power beyond classical capabilities, it constitutes a suitable path to introduce several important areas in the classroom: quantum many-body physics, quantum phase transitions, the physics of disordered systems and spin glasses, combinatorial optimization, and computational complexity theory. Moreover, the fundamental interest is increasing following the recent proposals to explain gravity and the structure of space-time itself in terms of quantum computation.[10,11]

There are several proposals in the literature to introduce quantum computation in the classroom, such as those of Scarani,[12] as early as 1998, or the more recent and comprehensive proposal of Candela.[13] They focus on *quantum logic gates* and their chosen applications are usually in cryptography, such as the work of Gerjuoy.[14] We present a novel proposal, based on the alternative route of AQC. Both approaches are known to be equally powerful in principle,[15] but AQC presents some interesting pedagogical advantages, such as the facility with which new problems can be approached. Moreover, there is an ongoing effort in the scientific and technical communities to implement the AQC in practical devices,[16] including the D-wave machine[17] (whose performance is still subject to discussion[18]).

The term *adiabatic* in AQC may present some difficulties for the students, since in this context it is totally unrelated to the usual meaning in thermodynamics. In quantum mechanics, a process is adiabatic when the external parameters evolve so slowly that the system is always able to adapt itself perfectly and smoothly, so non-equilibrium effects are absent. Therefore, it is closer to the concept of *quasistatic process*.

This article is organized as follows. In Sec. II, we introduce the type of problems that we intend to solve. Section III discusses the general idea of analog computers, while the philosophy of adiabatic quantum computation is discussed in Sec. IV in intuitive terms. In Sec. V, we descend to technicalities and describe the AQC process by giving precise instructions to build a classical simulator. The last section discusses the roadmap that opens up before the students after this experience, along with the expected results and limitations of the AQC approach.

II. WHAT TO SOLVE

When computer scientists talk about *problems* they have a rather precise notion in mind. Let us take a look at several examples before providing a general definition:[19]

- The traveling salesperson problem (TSP). You are given a list of cities and the distances between them, and you have to provide the shortest route to visit them all.
- The knapsack problem. You are given the weights and values of a set of objects and you have to provide the most valuable subset of them to take with you, given a certain bound on the total weight.
- Sorting. Given N numbers, return them in non-ascending order.
- The integer factorization problem. You are given a big number M, and you have to provide two integer factors, p and q, such that $M = pq$.
- The satisfactibility problem (SAT). You are given a boolean expression of many variables $x_i \in \{0, 1\}$, for example, $P(x_1, x_2, x_3, x_4) = x_1 \vee \overline{x_2} \wedge (x_3 \vee \overline{x_4})$.[20] Then, you are asked whether there is a valuation of those variables which will make the complete expression true. For example, in that case, making all $x_i = 1$ is a valid solution.

Notice that all these problems have a similar structure: you are given certain input data (the initial numbers, the list of the cities, the list of objects or the integer to factorize) and you are asked to provide a response. The first two problems are written as *optimization* problems, in which a certain *target function* should be minimized (or maximized). The sorting problem can be restated as an optimization problem: we can design a *penalty function* to be minimized, by counting the misplaced consecutive numbers. The factorization problem can also be expressed in that way: find p and q such that

$E = (M - pq)^2$ becomes minimal—and zero if possible. SAT can also be regarded as an optimization problem in which the evaluation of the boolean formula should be maximized.

Thus, all those problems are *combinatorial optimization* problems.[21,22] This means that, in order to solve them by brute force, a finite number of possibilities must be checked. But this number of possibilities grows very fast with the number of variables or the size of the input. Typically, they grow exponentially.

Not all optimization problems are equally hard. Let us focus on the minimal possible time required to solve them as a function of the input size. If a problem can be solved in polynomial time,[23] then it is said to belong to class **P**. If a candidate solution can be evaluated in polynomial time, then it is said to belong to class **NP**. Of course, every problem in **P** is also in **NP**, but it is not known whether every **NP** problem is in **P** or not.[19,24,25] In fact, the **P**= ?**NP** question is one of the most relevant open problems in mathematics.[26] Most practicioners believe that **P**≠ **NP**, i.e.: an easy-to-check problem does not necessarily have an easy solution. But this conjecture appears to be amazingly difficult to prove.[25] See, for example, Ref. 27 for a recent attempt at a proof of their inequivalence.[28]

All the problems in the list above are **NP**: given a candidate solution, it can always be checked in polynomial time. But only one of them is known to be in **P**: the sorting problem, because it can always be solved in time $O(N\log(N))$,[29] which is less than $O(N^2)$. For the factorization problem we do not know whether it is in **P** or not. The other three belong to a special subset: they are **NP**-complete. This means that they belong to a special group with this property: if a polynomial algorithm to solve one of them is ever found, then we will have a polynomial algorithm to solve *all* **NP** problems. The generality and power of this result is simply amazing, and is known as Cook's theorem.[19,24,25]

How can the solution to an **NP**-complete problem be useful to solve all **NP** problems? By using a strategy called *reduction*. An instance of the sorting problem, for example, can be converted into an instance of SAT in polynomial time.[19] Thus, the strategy to solve any **NP**-problem would be: (i) translate your instance to an instance of SAT, (ii) solve that instance of SAT, and (iii) translate back your solution. It is very relevant for physicists to know which combinatorial optimization problems are **NP**-complete for many reasons, and one of the most important is to avoid losing valuable time with a naive attempt to solve them in polynomial time.

A. The spin-glass problem

Let us introduce the example problem that we will use in most of the text, which was born in the physics of magnetism: the *spin-glass* problem.[30,31] Consider a set of N *spins* which can take values ↑ or ↓ (numerically valued +1 or –1), see Fig. 1. They are connected by wires, which can be ferromagnetic (F) or antiferromagnetic (AF), depending on whether they encourage spins to be parallel or antiparallel. The wires need not be equally strong, let J_{ij} be the coupling strength between spins s_i and s_j, positive meaning AF. Moreover, an external magnetic field h_i can be applied on each spin which will give a different energy to the ↑ and ↓ directions. Then, we can write the target function, or energy, as

$$E(s_1, \ldots, s_N) = \sum_{i,j} J_{ij} s_i s_j + \sum_i h_i s_i. \tag{1}$$

Let $h_i = 0$ for now. If the wiring graph is 1-dimensional (without any loops) the problem is trivial: give any value to the first spin, and then set all in order changing value only if the coupling is AF. When the graph presents *loops* this solution can run into contradictions, due to the possibility of *frustration*: maybe not all couplings can be satisfied simultaneously. Choosing which ones to leave out is a very delicate problem. If the graph is 2-dimensional (it can be drawn on a plane without any links crossing), then there is a very clever polynomial algorithm to solve it.[32] If the graph is 3D, the problem becomes **NP**-complete.

The spin-glass problem has a physical origin, but it can also be considered in a more abstract way, e.g., to study restoration of images which have undergone some degradation[30] or to select an optimal subset among a set of incompatible goals.[31]

III. ANALOG COMPUTERS

Nature optimizes. Soap films minimize area, which means that they minimize the energy.[33] In order to *predict* which shape will a soap film acquire, when constrained between certain wires and walls, we find the surface of minimal area with those constraints. Or, if we are interested in knowing which kind of lattice will be formed by a certain type of molecules, then a first approximation proceeds as follows. First, we determine the interaction energy between pairs of atoms, as a function of their relative position, $V(\vec{r}_2 - \vec{r}_1)$. Now, we minimize the following energy function:

$$E(\vec{r}_1, \ldots, \vec{r}_n) = \sum_{i<j} V(\vec{r}_i - \vec{r}_j). \tag{2}$$

Can we use the fact that Nature tends to minimize the energy to solve computational problems? Of course. Those constitute *analog computers*.[34] As a simple example, let us design an analog computer to sort numbers. We get N raw spaghetti strands and cut them so their lengths correspond to the numbers to sort. Then, we pick up the bunch, hold them vertically on a table and allow gravity to act. Mere inspection will allow us to select the spaghetti in descending length order. Another example is constituted by the *Steiner tree* problem: given a set of cities, find the minimal set of railroads connecting all of them, with extra junctions if required.[35] A clever analog computer to solve this problem is just to fix some pegs on a board, representing the positions

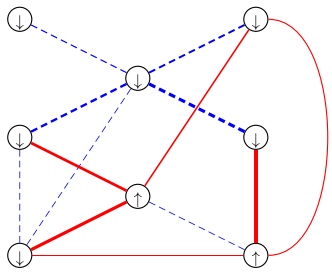

Fig. 1. Example of a spin glass, nodes represent spins. Dashed links are ferromagnetic (parallel) interactions, while continuous are antiferromagnetic (antiparallel). The width of the link represents the intensity of the interaction, J_{ij}.

of the cities, and inserting the board into soapy water. If the board is lifted slowly, a soap film will form, marking exactly the desired positions of the railroad lines.[33,36]

But one should be careful, because Nature can make mistakes. Typical crystals are *not* perfect. They have defects: atoms which are misplaced, vacancies, etc. So, if someone proposes a computational method based on the fact that Nature minimizes the energy, do not forget to ask *how* this minimization proceeds to avoid misunderstandings. In mathematical terms, we should be aware that optimization problems are prone to presenting *local minima*, i.e., configurations for which all small displacements will increase the target function, but not a big one. In physics, we usually called such a configuration a *metastable* state. Interesting optimization problems typically present a huge number of those metastable states, constituting what is typically termed a *complex landscape*. Greedy algorithms—which start with a certain seed configuration and perform only small displacements which reduce the target function—will typically end up in such a metastable state. The actual *global minimum* may never be found that way. This is indeed what happens to real life spin glasses.[21,22]

How to avoid those mistakes? In order to make perfect crystals, metallurgists typically have recourse to *annealing*, i.e., the metal is melted and allowed to reduce its temperature very slowly. When the temperature becomes low enough, the system is expected to be in its global energy minimum with very high probability. Yet, mistakes are bound to take place if the temperature is reduced too fast. Why does this procedure work? Thermal fluctuations allow the system to explore a huge number of configurations. Sometimes, a metastable state is found, but if temperature is still large enough, fluctuations will allow it to escape. The escape probability is always related to the energy barrier separating the metastable state from the deeper energy minima.

So, annealing can be considered an *analog computation*. There is even a numerical technique based on it, called *simulated annealing*.[37] Unfortunately, annealing also fails sometimes. In fact, there is a series of *No Free Lunch* theorems[38] that state that every optimization method has an Achilles heel. Annealing works nicely when the metastable states are separated by low energy barriers. Some target functions have barriers that are tall but very thin.[8] Others will have different peculiarities. The best strategy is to have a huge toolbox of methods, to know your problem, and to choose the one that suits the best.

Is there any alternative analog computation suitable for problems with tall and thin barriers? Yes, there is one. Instead of escaping metastable states through thermal fluctuations, we may try to escape them through *quantum tunneling*, since the probability of such an event is known to decay with the product of the height and width of the energy barriers.[8] Adiabatic quantum computation is the way in which this idea is put into practice.

IV. ANALOG QUANTUM COMPUTERS

This section describes analog quantum computation in a qualitative way. The quantitative description and the instructions to build the simulator are given in Sec. V.

Quantum computation[1,2] was born out of a difficulty. Quantum mechanics takes place in a much larger space than classical mechanics. Consider a system of N classical spins, which can point either $\uparrow$ or $\downarrow$ along the Z-axis. A complete description of that system will be given by N values in the $\{\uparrow,\downarrow\}$ set, that, is, N bits. A quantum description of the same system, assuming a pure state, implies providing the probability amplitude for each of the 2^N possible classical configurations: a vector in a 2^N-dimensional complex Hilbert space. Thus, we recognize an exponential explosion of the dimensionality of the problem: *N qubits* are exponentially more than N bits.

But maybe all that complexity can be harnessed in our favor. Feynman famously proposed in 1982 (Ref. 39) to engineer quantum systems so that their evolution leads to states which solve difficult problems. The core idea is to make each of the 2^N dimensions of the Hilbert space represent a candidate solution of a combinatorial optimization problem. According to the standard ontological interpretation of quantum mechanics, the 2^N probability amplitudes are not mere statistical descriptors: they *do really exist*. Thus, we will try to make them compete and read the solution afterwards. Maybe, if we are lucky, they will solve **NP**-complete problems in a polynomial time.[40]

Among the different theoretical flavors of quantum computation,[1,2] we will focus on *adiabatic quantum computation* (AQC), which is also known as *quantum annealing*.[6–8] In an AQC we engineer a quantum system so that its energy is related to the target function that we want to minimize. Thus, its ground state (GS) will provide the solution to our problem. If we start out at high temperature and cool the system down, then we are simply performing an annealing analog computation. Alternatively, we can operate always at extremely low temperature—so that quantum effects are always important—but add an extra element to the system which forces strong quantum fluctuations. This extra element is then slowly reduced and, when it vanishes, the GS will give us the solution to our problem. In other terms: we make the system Hamiltonian evolve from a certain starting Hamiltonian, H_0—whose GS presents strong quantum fluctuations— to our target Hamiltonian, H_1, whose GS is the solution to our problem.

For example, consider the spin glass problem with the wirings and local magnetic fields set as in Fig. 1 and Eq. (1) constituting H_1. Its GS corresponds to the desired solution to our problem, with all spins pointing either $\uparrow$ or $\downarrow$ along the Z-axis. The extra element in our case will be a strong transverse magnetic field pointing along the X axis. When it is included, we have H_0. Let us recall from basic quantum mechanics that a spin-1/2 pointing in the X-direction can be regarded as pointing simultaneously in the $\uparrow$ and $\downarrow$ directions along the Z-axis. Thus, our quantum state is totally uncertain about the Z-component of each spin. When the external transverse field is slowly reduced, spins will gradually choose between $\uparrow$ and $\downarrow$. Some of them will do it soon, because they are not subject to contradictory constraints. Some others will start fluctuating together, binding to make alliances: "if you stay $\uparrow$, then I will stay $\uparrow$ too" (or $\downarrow$, of course). These alliances, or clusters, can involve nearby or distant spins, and will slowly spread throughout the system and merge together until finally a global solution is singled out. This alliance system, in which spins fluctuate together, is intimately related to *entanglement*.

Entanglement[1,41] is one of the most relevant features of quantum mechanics, and one of the cornerstones of modern theoretical physics. Let us consider a pure state describing a physical system which is split into two parts, A and B. An observer having access only to part A (or B) will be unable to

describe it using a pure state. Typically, she will have to describe it as a mixed state, i.e., a set of several pure states with different probabilities. This is the signature of entanglement: information is lost when only part of the system is considered. If the observer makes an experiment to find out which of the pure states is the correct one to describe part A, the pure state needed to describe part B will be immediately known. Thus, correlations between A and B go far beyond classical statistical correlations. They are not a mere expression of our ignorance. According to our ontological interpretation, they do exist.

Quantum states with low entanglement can be efficiently simulated in a classical computer.[42–44] Thus, an AQC which attains authentic quantum supremacy *must* present high entanglement. Nonetheless, the production of highly entangled states does not present a difficulty for AQC. The true difficulty of AQC stems from the complex energy landscape: to ensure that we have indeed reached the true GS and not a long-lived excited state, which is the quantum analog of a metastable state. Luckily, we count on the *adiabatic theorem*:[45] if the evolution between H_0 and H_1 is performed slowly enough and the energy gap between the ground state and the first excited state never closes, the GS of H_0 will evolve into the GS of H_1. The problem, of course, stems from the gap condition. Experience shows that all known AQC algorithms present low gap at a certain moment of the procedure. Low gap does not spoil adiabaticity, but forces us to proceed slowly. The precise speed at which the algorithm can traverse the low gap stage is given by the Landau-Zener formula,[8,46] which will be discussed in Sec. V. So, the main question is: how does the gap scale with the problem size? The answer to this question will provide the theoretical lower bound to the time needed for the computation.

Thus, two elements are relevant to analyse an AQC proposal: the minimum gap and the maximal entanglement. They usually occur at the same time during the computation, constituting a *quantum phase transition* (QPT).[47] This QPT is, somehow, the barrier that Nature has established between the statement of and the solution to the problem. Successful design of AQC algorithms therefore involves a deep knowledge of QPT in disordered systems. In Sec. V, we will discuss how to build a numerical simulation of any AQC procedure and characterize the ensuing QPT.

V. SIMULATING AN ADIABATIC QUANTUM COMPUTATION

Let us consider the spin-glass problem with a certain set of couplings J_{ij} and external fields, h_i. Each such choice will be called a *realization*. The solution to our problem will be given by the GS of the target Hamiltonian, of Ising type

$$H_1 = \sum_{ij} J_{ij} S_i^z S_j^z + \sum_i h_i S_i^z. \tag{3}$$

The starting Hamiltonian, on the other hand, will be simply

$$H_0 = -\sum_i S_i^x, \tag{4}$$

where we should recall that the action of the local operators S^z and S^x on the states $|\uparrow\rangle$ and $|\downarrow\rangle$ is

$$\begin{aligned} S^z|\downarrow\rangle &= -|\downarrow\rangle, \quad S^z|\uparrow\rangle = |\uparrow\rangle, \\ S^x|\downarrow\rangle &= |\uparrow\rangle, \quad S^x|\uparrow\rangle = |\downarrow\rangle. \end{aligned} \tag{5}$$

Please note that factors $\hbar/2$ are omitted for simplicity.

We can imagine a *dial* parametrized by a certain $\lambda \in [0,1]$ which allows us to interpolate smoothly between these two Hamiltonians[7,44]

$$\begin{aligned} H(\lambda) &= \lambda H_1 + (1-\lambda) H_0 \\ &= \lambda \left(\sum_{ij} J_{ij} S_i^z S_j^z + \sum_i h_i S_i^z \right) - (1-\lambda) \sum_i S_i^x. \end{aligned} \tag{6}$$

This Hamiltonian is known as the *Ising model in a transverse field* (ITF).[8,43,44,48] The AQC prescription is to start with $\lambda = 0$ and obtain the GS, then increase λ adiabatically until $\lambda = 1$ and read the solution. The rest of the section will discuss how to simulate and characterize the quantum state during the procedure.

A. Writing Hamiltonians

Every observable on a quantum system described by a finite dimensional Hilbert space is represented by a Hermitian matrix. Our first step towards building the classical simulator of an AQC consists of writing explicitly the matrices representing the starting and target Hamiltonians, H_0 and H_1. For didactical reasons, we favor a configurational approach, which starts by considering the 2^N basis states, labeled by the spin values at each site, e.g., for the $N=3$ case: $|\downarrow\downarrow\downarrow\rangle$, $|\downarrow\downarrow\uparrow\rangle$, $|\downarrow\uparrow\downarrow\rangle$, $|\downarrow\uparrow\uparrow\rangle \ldots$ $|\uparrow\uparrow\uparrow\rangle$.

Hamiltonian H_1 is the easiest to write, because S_i^z is diagonal in this basis. Let us consider an example system, as in Fig. 2. We have three spins, linked with anti-ferromagnetic (positive) couplings, $J_{12} = +1$, $J_{13} = J_{23} = +2$. Moreover, we only have a local field at spin 1: $h_1 = +1$ all the others are zero. Then, the Hamiltonian action on the basis states is simple

$$\begin{aligned} H_1|\downarrow\downarrow\downarrow\rangle &= (+1+2+2-1)|\downarrow\downarrow\downarrow\rangle = +4|\downarrow\downarrow\downarrow\rangle, \\ H_1|\downarrow\downarrow\uparrow\rangle &= (+1-2-2-1)|\downarrow\downarrow\uparrow\rangle = -4|\downarrow\downarrow\uparrow\rangle, \\ H_1|\downarrow\uparrow\downarrow\rangle &= (-1+2-2-1)|\downarrow\uparrow\downarrow\rangle = -2|\downarrow\uparrow\downarrow\rangle, \\ H_1|\downarrow\uparrow\uparrow\rangle &= (-1-2+2-1)|\downarrow\uparrow\uparrow\rangle = -2|\downarrow\uparrow\uparrow\rangle, \\ H_1|\uparrow\downarrow\downarrow\rangle &= (-1-2+2+1)|\uparrow\downarrow\downarrow\rangle = +0|\uparrow\downarrow\downarrow\rangle, \\ H_1|\uparrow\downarrow\uparrow\rangle &= (-1+2-2+1)|\uparrow\downarrow\uparrow\rangle = +0|\uparrow\downarrow\uparrow\rangle, \\ H_1|\uparrow\uparrow\downarrow\rangle &= (+1-2-2+1)|\uparrow\uparrow\downarrow\rangle = -2|\uparrow\uparrow\downarrow\rangle, \\ H_1|\uparrow\uparrow\uparrow\rangle &= (+1+2+2+1)|\uparrow\uparrow\uparrow\rangle = +6|\uparrow\uparrow\uparrow\rangle. \end{aligned} \tag{7}$$

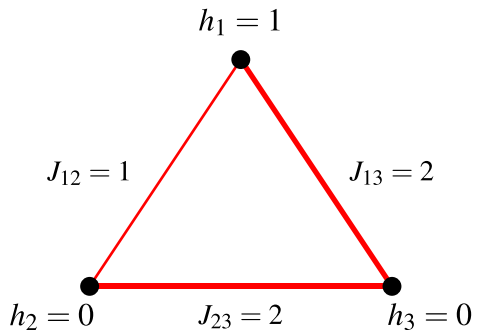

Fig. 2. Example of spin glass structure with $N=3$.

So the full matrix is

$$H_1 = \begin{pmatrix} +4 & & & & & & & \\ & -4 & & & & & & \\ & & -2 & & & & & \\ & & & -2 & & & & \\ & & & & 0 & & & \\ & & & & & 0 & & \\ & & & & & & -2 & \\ & & & & & & & +6 \end{pmatrix}, \quad (8)$$

and all the non-diagonal elements are zero.

The GS, of course, is the state $|\downarrow\downarrow\uparrow\rangle$, with energy –4. Of course, if we can write down H_1 in full, this means that we already know the solution to our problem. But that is not our aim. We want to know how a quantum computer will perform when trying to find it.

Let us write down the matrix representation for the starting Hamiltonian H_0, Eq. (4). We know in advance that in the GS all the spins are pointing in the X-direction. Thus, the state (for N spins) is

$$|\Psi(0)\rangle = \left(\frac{1}{\sqrt{2}}\right)^N (|\downarrow\rangle + |\uparrow\rangle) \otimes \cdots \otimes (|\downarrow\rangle + |\uparrow\rangle). \quad (9)$$

If the tensor products are expanded we notice that $|\Psi(0)\rangle$ contains all configurations with the same weight. It is the *most democratic state*. The physical meaning is as follows: when seen from the Z-axis, a spin-1/2 pointing in the X direction seems to be randomly fluctuating between $\uparrow$ and $\downarrow$. All spins fluctuate independently, therefore all configurations are equally likely. Let us write down the full matrix for the starting Hamiltonian when $N=3$, $H_0 = -(S_1^x + S_2^x + S_3^x)$,

$$\begin{aligned}
(S_1^x + S_2^x + S_3^x)|\downarrow\downarrow\downarrow\rangle &= |\uparrow\downarrow\downarrow\rangle + |\downarrow\uparrow\downarrow\rangle + |\downarrow\downarrow\uparrow\rangle \\
(S_1^x + S_2^x + S_3^x)|\downarrow\downarrow\uparrow\rangle &= |\uparrow\downarrow\uparrow\rangle + |\downarrow\uparrow\uparrow\rangle + |\downarrow\downarrow\downarrow\rangle \\
(S_1^x + S_2^x + S_3^x)|\downarrow\uparrow\downarrow\rangle &= |\uparrow\uparrow\downarrow\rangle + |\downarrow\downarrow\downarrow\rangle + |\downarrow\uparrow\uparrow\rangle \\
(S_1^x + S_2^x + S_3^x)|\downarrow\uparrow\uparrow\rangle &= |\uparrow\uparrow\uparrow\rangle + |\downarrow\downarrow\uparrow\rangle + |\downarrow\uparrow\downarrow\rangle \\
(S_1^x + S_2^x + S_3^x)|\uparrow\downarrow\downarrow\rangle &= |\downarrow\downarrow\downarrow\rangle + |\uparrow\uparrow\downarrow\rangle + |\uparrow\downarrow\uparrow\rangle \\
(S_1^x + S_2^x + S_3^x)|\uparrow\downarrow\uparrow\rangle &= |\downarrow\downarrow\uparrow\rangle + |\uparrow\uparrow\uparrow\rangle + |\uparrow\downarrow\downarrow\rangle \\
(S_1^x + S_2^x + S_3^x)|\uparrow\uparrow\downarrow\rangle &= |\downarrow\uparrow\downarrow\rangle + |\uparrow\downarrow\downarrow\rangle + |\uparrow\uparrow\uparrow\rangle \\
(S_1^x + S_2^x + S_3^x)|\uparrow\uparrow\uparrow\rangle &= |\downarrow\uparrow\uparrow\rangle + |\uparrow\downarrow\uparrow\rangle + |\uparrow\uparrow\downarrow\rangle.
\end{aligned} \quad (10)$$

Or, in matrix form

$$H_0 = -\begin{pmatrix} & 1 & 1 & & 1 & & & \\ 1 & & & 1 & & 1 & & \\ 1 & & & 1 & & & 1 & \\ & 1 & 1 & & & & 1 & 1 \\ 1 & & & & & 1 & & \\ & 1 & & & 1 & & & 1 \\ & & 1 & & 1 & & & 1 \\ & & & 1 & & 1 & 1 & \end{pmatrix}. \quad (11)$$

The first exercise is, therefore, to code a procedure to write down the explicit matrices for H_0 and H_1. Then, we can use standard diagonalization procedures to obtain the ground state of $H(\lambda)$ for all values of the adiabatic parameter λ. Notice that adapting our simulator to other combinatorial optimization problems is straightforward as long as the configurational space is also $\{\downarrow,\uparrow\}^N$, because it only implies changing H_1.

B. The energy gap

The first important magnitude to note is the energy *gap*, the energy difference between the GS and the first excited state of $H(\lambda)$ as a function of λ,

$$\Delta E(\lambda) = E_1(\lambda) - E_0(\lambda). \quad (12)$$

In Fig. 3, we show the gap as a function of the adiabatic parameter λ for 4 different realizations of the spin-glass with $N=8$ qubits. (N.B. This is different from the example in Fig. 2, for which $N=3$).[49] Notice that, in most cases, the gap presents a minimum at a finite value of λ.

Why is this magnitude relevant? Our attempt to drag the GS of $H(\lambda)$ adiabatically must be successful if the conditions of the adiabatic theorem are met: the energy gap must be strictly non-zero, and the evolution must be infinitely slow. What happens if we try to make the full evolution in a finite time? Then we have a certain probability of losing adiabaticity and jumping to an excited state. Once adiabaticity is lost, we have no certainty that the true GS will be obtained at the end of the procedure, as we may get stuck on any low energy excited state. Notice that one of the curves of Fig. 3 shows a gap very close to zero at $\lambda \approx 0.8$, pointing to a very large AQC time in the *worst case scenario*.

In order to ensure adiabaticity, we must make sure that the velocity at which the Hamiltonians change is low enough. The Landau-Zener formula tells us that, under general conditions, the speed should be proportional to the gap squared[8,46]

$$v(\lambda) \propto \Delta E(\lambda)^2. \quad (13)$$

The proportionality constant will depend on our tolerance to making a mistake and other considerations which are not relevant here. If we were always able to adapt the AQC velocity to the Landau-Zener bound, we obtain an expression for the total time needed by the AQC

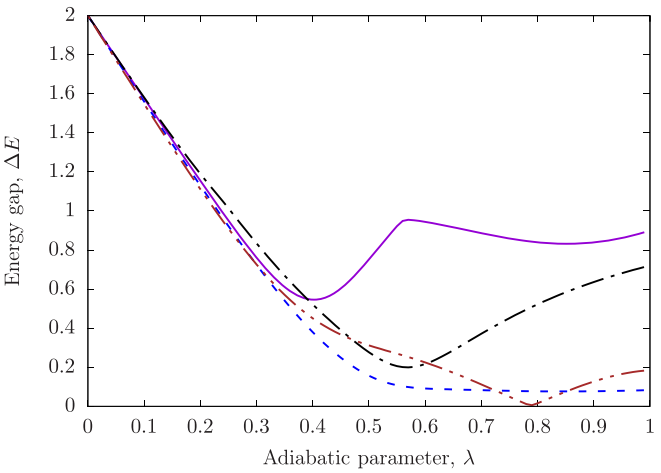

Fig. 3. Energy gap along an AQC, Eq. (12), as a function of the adiabatic parameter λ, for 4 realizations of the spin-glass problem with $N=8$ qubits. Notice that for one of the realizations the gap approaches zero near $\lambda \approx 0.8$.

$$T_{AQC} \sim \int_0^1 \frac{d\lambda}{v(\lambda)}. \tag{14}$$

In order to evaluate the proposed AQC procedure, one should compare how this optimal value for T_{AQC} grows with the input size of our problem, for different types of problems. If it grows polynomially, then our problem belongs to the quantum equivalent of class **P**, which is called **BQP** (bound-error quantum polynomial).[1]

C. Analysing the GS: Magnetization and entanglement

Let us now focus on $|\Psi_0(\lambda)\rangle$, the actual GS as a function of the adiabatic parameter. As a many-qubit state, it is a vector on a 2^N-dimensional complex Hilbert space, which is a difficult object to visualize. We make various attempts in Fig. 4 for a concrete realization of the spin-glass with $N=8$ (i.e., fixed values of J_{ij} and h_i). The first panel of the figure shows the evolution of the expected magnetization of the i-th spin, $\langle S_i^z\rangle$ along the AQC. Notice that some components *make up their mind* rather soon, because their couplings are easy to satisfy. The second panel shows the squared modulus of all the components of the GS wavefunction as a function of λ for the same system, in logarithmic scale. We can see

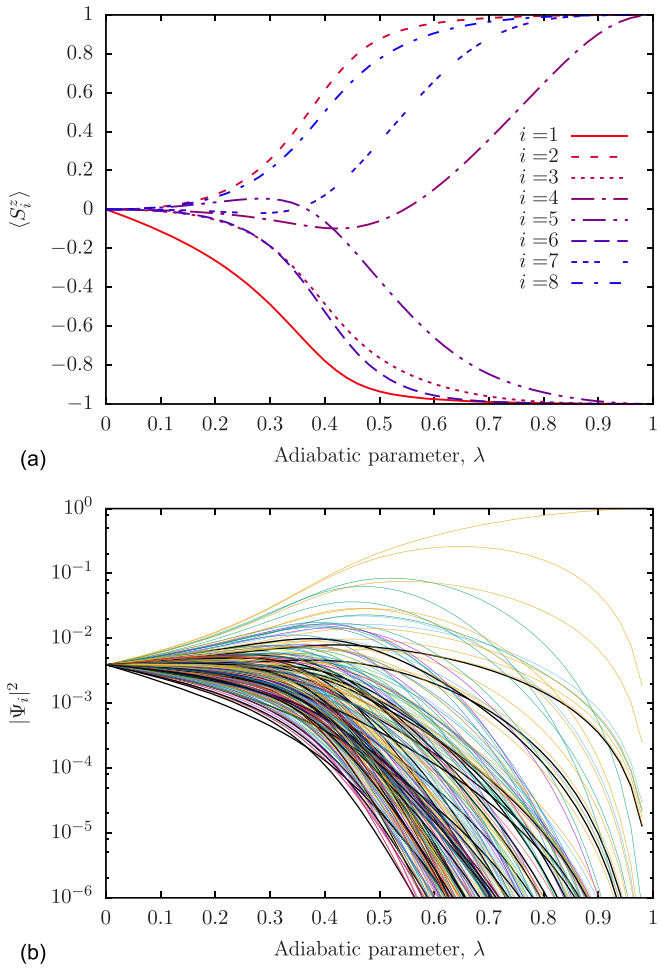

Fig. 4. First panel: expected magnetization $\langle S_i^z\rangle$ for all spins for a realization of the proposed AQC with $N=8$, as a function of the adiabatic parameter λ. Second panel: squared wavefunction amplitudes $|\Psi_i|^2$ for the 2^N configurations for the same realization.

that, for $\lambda=0$, they all take the same value, $1/2^N$, but for $\lambda=1$ only one of them is 1, and all the others have vanished.

Let us now consider the evolution of entanglement of the GS along the adiabatic route. The most usual measurement of entanglement is the *entanglement entropy*.[1,41] Consider any (normalized) pure state $|\Psi\rangle$ and let us divide the system into two parts in any reasonable way, e.g., the left and right halves of a qubit string. Call the parts A and B. We can always write it as

$$|\Psi\rangle = \sum_{i,j} C_{ij}|i\rangle_A \otimes |j\rangle_B, \tag{15}$$

where $|i\rangle_A$ and $|j\rangle_B$ constitute orthonormal basis for the Hilbert spaces of A and B, respectively, and the C_{ij} are just a (rectangular) matrix re-writing of the wavefunction amplitudes. We can always perform a singular value decomposition (SVD)[51] of the matrix C_{ij} and write the pure state as

$$|\Psi\rangle = \sum_{k=1}^{m} \sigma_k |\psi_k\rangle_A \otimes |\psi_k\rangle_B, \tag{16}$$

where $|\psi_k\rangle_A$ are orthogonal states of part A (same for B) and σ_k are the singular values, also called Schmidt coefficients, which convey the weight of each pair of states. Their squares can be considered as a probability distribution, and they add up to one. The value m is called the Schmidt number, and it constitutes also a measurement of entanglement: if $m=1$, the state is not entangled. The von Neumann entanglement entropy can be defined as

$$S = -\sum_k \sigma_k^2 \log(\sigma_k^2), \tag{17}$$

i.e., the Shannon entropy of the probabilities for each pair of states. In order to obtain the σ_k numerically, we perform the following procedure:[52]

- Write the wavefunction components as $C_{i,j}$ where i is an index running through all configurations of part A and j is the same for B.
- Perform a singular value decomposition on that (in general rectangular) matrix. The singular values are real and positive, they constitute the σ_k.

Notice that in our presentation we did not need to make use of the reduced density matrix.[1,41,45]

Figure 5 shows the entanglement entropy along with the gap for an AQC performed on a realization of the spin-glass using $N=8$ spins. The partition chosen for the entropy calculation is simply between the first 4 spins and the rest. We can see that the maximum of the entropy approximately coincides with the minimum of the gap. This near coincidence hints at the existence of a quantum phase transition (QPT) at a finite value of λ, in the thermodynamical limit. That QPT would constitute the most *delicate* moment of the computation and, as we previously discussed, the complexity of the state would be maximal.

VI. CONCLUSIONS

Quantum computation is an exciting area of research which attracts both physics and computer science students. Writing the code for a quantum computer simulation can

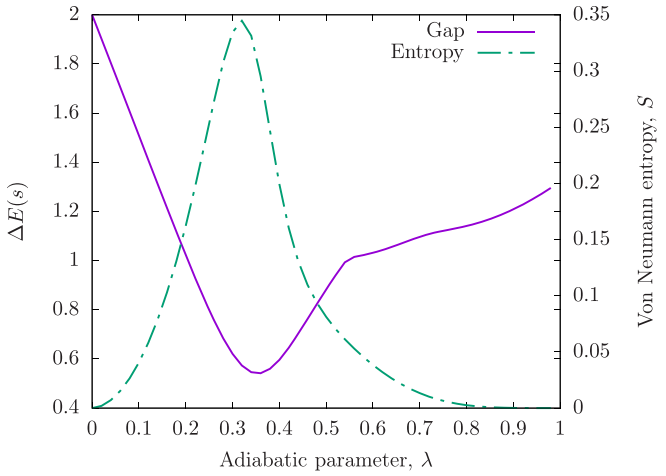

Fig. 5. Entropy and gap for a single AQC of $N=8$ spins. The entropy is computed for the partition between the first 4 spins and the rest. Notice that the minimum of the gap nearly coincides with the maximum of the entropy, showing the approximate location of a (finite-size version of a) quantum phase transition.

constitute a challenging task which can serve as a connecting thread in order to present a lot of interesting material, ranging from computational complexity, combinatorial optimization techniques or annealing to quantum many-body systems, disordered and glassy systems, quantum phase transitions or entanglement in an elementary yet compelling way.

Among the many flavors of quantum computation, we have focused on adiabatic quantum computing for several reasons. The first is our own expertise in the area. The second is the fact that AQC is *universal*,[15] in the sense that any quantum computation that can be performed in polynomial time can be simulated by another AQC in polynomial time. The third reason is the wealth of topics in computation theory and physics that it can introduce. Practical implementations of AQC constitute a very active topic of research,[16] including the D-wave machine[17] which has given rise to a strong controversy in the field.[18] Let us not forget that there are some negative forecasts about the capacity of AQC to solve **NP**-complete problems in polynomial time (and, due to its universality, extendable to all quantum computation), related to the physics of disordered systems and their tendency to present exponentially vanishing gaps.[53] Still, Feynman's claim[39] is still valid: predicting the evolution of a quantum system is much harder than that of a classical one, and we should be able to use that difficulty in our favor.

This presentation was given for two consecutive years at the school of computer science at Universidad Complutense de Madrid, where some testing code was made available for the students during the practical sessions. Please refer to the appendix for instructions about how to use this testing code.

ACKNOWLEDGMENTS

The authors want to acknowledge the faculty and students of the Facultad de Informática of UCM (Madrid) for their kind invitation to deliver this crash course, particularly to I. Rodríguez-Laguna and N. Martí. The authors would also like to thank G. Sierra for very useful comments on the manuscript. This work was funded by the Spanish government through Grant Nos. FIS2015-69167-C2-1-P and FIS2015-66020-C2-1-P.

APPENDIX A: PRACTICAL SESSIONS

The theoretical discussion should be complemented with practical sessions in which the students will write their own code to simulate a quantum computer on their (classical) computers. The instructor should choose a computer language and platform which is known to the students. In our case, we have worked on C++ on GNU-Linux systems.

The main advantage of our pedagogical AQC approach is that the students can choose any combinatorial optimization problem on their own, with only one condition: that it can be formulated as an energy minimization problem on a spin system. For example, the factorization problem can be easily mapped in that way by allowing the spins $\uparrow$ and $\downarrow$ to encode the bits of the possible factors. The teacher should encourage the students to approach different types of problems (sorting, traveling salesperson....) and express them in the spin-glass language suitable for the AQC simulation.

The structure of a simple simulation of an AQC is always the following:

- Establish your target function, $T(s_1 \cdots s_N)$, where each s_i is a always a bit. Typical examples are the spin-glass problem, integer factorization or the knapsack problem. Other problems, such as the traveling salesperson problem, pose further difficulties. Fix a (small) number of bits, e.g., $N=8$ for fast results on a laptop.
- Write down your target Hamiltonian, H_1, which has as diagonal entries the evaluations of the target function for all 2^N configurations, from $T(0,\ldots,0)$ to $T(1,\ldots,1)$.
- Write down a suitable initial Hamiltonian, usually H_0.[54]
- Compute, for all $\lambda \in [0,1]$, $H(\lambda) = \lambda H_1 + (1-\lambda)H_0$.
- Find the ground state and the first excited state of $H(\lambda)$, compute the energy gap.
- Find the bound for the speed at which you can do AQC for that value of λ, given by the energy gap squared.
- Integrate, and find the total time required for the AQC.

Thus, the students can find empirically what the expected time is for an AQC to solve a certain problem, always using a specific adiabatic route. The students may wonder how difficult it would be to build the target Hamiltonian in the laboratory. Many types of few-qubit interactions can be easily built, but the question is extremely relevant in practice. Nonetheless, our concern is more fundamental: *even if* the computer is built, will it be able to find the solution to our problem in a short time? The problem is so open that even undergraduate students under good supervision can find new results.

Let us provide an interesting different example: how to build an AQC to factorize numbers. Let N be our target number. If $N=pq$, with $p \le q$ integers, then $p \le N^{1/2}$. For example, if $N=33$, then any factor should be smaller than $33^{1/2} < 6$. Thus we only need to try values of $p<6$, which can be represented in 3 bits. For each p, our target function to minimize is $T(p) = (N \bmod p)$. We set the $T(0)$ and $T(1)$ values to any arbitrary large values, for example N itself. In bit language, we set

$$\begin{aligned} T(000) &= T(0) = 33, \quad T(001) = T(1) = 33, \\ T(010) &= T(2) = 1, \quad T(011) = T(3) = 0 \\ T(100) &= T(4) = 1, \quad T(101) = T(5) = 3, \\ T(110) &= T(6) = 3, \quad T(111) = T(7) = 5. \end{aligned} \tag{A1}$$

And the rest of the procedure is exactly the same.

APPENDIX B: TECHNICALITIES

We have prepared the code in a Github repository http://github.com/jvrlag/qtoys

The code provides all the data shown in this article. Moreover, it also returns a graphical representation of the full AQC process taking advantage of a visualization technique recently devised by us, known as *Qubism*.[50]

Our code is written in C++, prepared to run on a Linux or Mac system, but it is not hard to adapt to other systems. In order to be able to compile and run easily, you need the following libraries:

- BLAS (basic linear algebra system) and LAPACK (linear algebra package), standard algorithms for matrix manipulation, including diagonalization.
- Xlib, the basic graphics libraries of the X11 system.
- Imlib2, a general purpose image manipulation library.

[1]M. A. Nielsen and I. L. Chuang, *Quantum Computation and Quantum Information* (Cambridge U.P., Cambridge, 2001).
[2]N. D. Mermin, *Quantum Computer Science: An Introduction* (Cambridge U.P., Cambridge, 2007).
[3]D. Castelvecchi, "Quantum computers ready to leap out of the lab in 2017," Nature **541**, 9–10 (2017).
[4]IBM Quantum Experience, <https://www.research.ibm.com/ibm-q/>
[5]M. Hebenstreit, D. Alsina, J. I. Latorre, and B. Kraus, "Compressed quantum computation using a remote five-qubit quantum computer," Phys. Rev. A **95**, 052339 (2017).
[6]E. Farhi, J. Goldstone, S. Gutmann, J. Lapan, A. Lundgren, and D. Preda, "A quantum adiabatic evolution algorithm applied to random instances of an NP-complete problem," Science **292**, 472–475 (2001).
[7]A. Das and B. K. Charkrabarti, "Quantum annealing and analog quantum computation," Rev. Mod. Phys. **80**, 1061–1081 (2008).
[8]G. E. Santoro, R. Martoňák, E. Tosatti, and R. Car, "Theory of quantum annealing of an ising spin glass," Science **295**, 2427–2430 (2002).
[9]The sample code provided to our students, in C++, is freely available online at <http://github.com/jvrlag/qtoys>. The reader is encouraged to download, understand and modify it, following the instructions given in the appendix.
[10]L. Susskind, "Computational complexity and black hole horizons," Fortsch. Phys. **64**, 24–43 (2016).
[11]L. Susskind, "Dear qubitzers, GR=QM," e-print arXiv:1708.03040.
[12]V. Scarani, "Quantum computing," Am. J. Phys. **66**, 956–960 (1998).
[13]D. Candela, "Undergraduate computational physics projects on quantum computing," Am. J. Phys. **83**, 688–702 (2015).
[14]E. Gerjuoy, "Shor's factoring algorithm and modern cryptography. An illustration of the capabilities inherent in quantum computers," Am. J. Phys. **73**, 521–540 (2005).
[15]D. Aharonov, W. van Dam, J. Keme, Z. Landau, S. Lloyd, and O. Regev, "Adiabatic quantum computation is equivalent to standard quantum computation," in *Proceedings of the 45th Annual IEEE Symposium on Foundations of Computer Science* (2004), pp. 42–51.
[16]R. Barends *et al*., "Digitized adiabatic quantum computing with a superconducting circuit," Nature **534**, 222–226 (2016).
[17]S. Boixo *et al*., "Evidence for quantum annealing with more than one hundred qubits," Nat. Phys. **10**, 218–224 (2014).
[18]S. W. Shin, G. Smith, J. A. Smolin, and U. Vazirani, "How 'quantum' is the D-wave machine?," e-print arXiv:1401.7087.
[19]C. Papadimitriou, *Computational Complexity* (Addison Wesley, Reading, MA, 1994).
[20]Where $\vee$ is logical *or*, $\wedge$ is logical *and* and an overline is logical negation.
[21]T. Weise, *Global Optimization Algorithms—Theory and Application* (2009), <www.it-weise.de/projects/book.pdf>.
[22]A. Schrijver, "A course on combinatorial optimization," <homepages.cwi.nl/~lex/files/dict.pdf> (2013).
[23]Time scaling as a power-law of the input size.
[24]S. Mertens, "Computational complexity for physicists," Comput. Sci. Eng. **4**, 31–47 (2002).
[25]S. Aaronson, "**P**=? **NP**," (2017), <http://www.scottaaronson.com/papers/pnp.pdf>
[26]A cautionary remark: rigorously, classes **P** and **NP** are not defined for optimization problems, but for *decision problems*, which are solved by stating Yes or No. For example: is 29 a prime number? But optimization problems can be mapped to decision problems, so we will be a bit loose in the terminology.
[27]N. Blum, "A solution to the P vs NP problem," e-print arXiv:1708.03486.
[28]Of course, there are also problems which are not **NP**, e.g., how to win a chess game, but we will not discuss them in this text.
[29]D. E. Knuth, *The Art of Computer Programming. Volume 3: Sorting and Searching* (Addison-Wesley, Reading, MA, 1998).
[30]H. Nishimori, *Statistical Physics of Spin Glasses and Information Processing: An Introduction* (Clarendon Press, Oxford, 2001).
[31]M. Mezard, G. Parisi, and M. Virasoro, *Spin Glass Theory and Beyond* (World Scientific, Singapore, 1998).
[32]F. Barahona, "On the computational complexity of Ising spin glass models," J. Phys. A: Math. Gen. **15**, 3241–3253 (1982).
[33]C. Isenberg, "Problem solving with soap films, part I," Phys. Educ. **10**, 452–456 (1975).
[34]A. K. Dewdney, *The Armchair Universe*: *An Exploration of Computer Worlds* (W.H. Freeman & Co., New York, 1987).
[35]M. R. Garey, R. L. Graham, and D. S. Johnson, "The rectilinear steiner tree problem is *NP*-complete," SIAM J. Appl. Math. **32**, 826–834 (1977).
[36]D. R. Lovett and J. Tilley, "Illustrating phase transitions with soap films," Am. J. Phys. **59**, 415–421 (1991).
[37]S. Kirkpatrick, C. D. Gelatt, Jr., and M. P. Vecchi, "Optimization by simulated annealing," Science **220**, 671–680 (1983).
[38]D. H. Wolpert and W. G. Macready, "No free lunch theorems for optimization," IEEE Trans. Evol. Comp. **1**, 67–82 (1997).
[39]R. P. Feynman, "Simulating physics with computers," Int. J. Theor. Phys. **21**, 467–488 (1982).
[40]We would now like to dispel a usual misunderstanding. When quantum computation is developed your computers will still be mostly classical. Usual tasks will be undertaken by a classical CPU. But they will at some moments connect to a "quantum co-processor" devoted to solving certain special problems.
[41]L. Susskind and A. Friedman, *Quantum Mechanics: The Theoretical Minimum* (Basic Books, New York, 2014).
[42]G. Vidal, "Efficient simulation of slightly entangled quantum computations," Phys. Rev. Lett. **91**, 147902 (2003).
[43]J. Rodriguez-Laguna, "Quantum wavefunction annealing of spin glasses on ladders," J. Stat. Mech. **2007**, P05008 (2007).
[44]J. Rodriguez-Laguna and S. N. Santalla, "Physical consequences of P≠NP and the density matrix renormalization group annealing conjecture," J. Stat. Mech. **2014**, P07006 (2014).
[45]L. D. Landau and E. M. Lifshitz, *Quantum Mechanics* (Pergamon Press, Oxford, 1958).
[46]C. Wittig, "The Landau-Zener formula," J. Phys. Chem. B **109**, 8428–8430 (2005).
[47]S. Sachdev, *Quantum Phase Transitions* (Cambridge U.P., Cambridge, 1999).
[48]S. Suzuki, J. Inoue, and B. K. Chakrabarti, *Quantum Ising Phases and Transitions in Transverse Ising Models* (Springer, Berlin, 2013).
[49]Choosing both the J_{ij} and the h_i as random uniform deviates in $[-1,1]$.
[50]J. Rodriguez-Laguna, P. Migdał, M. Ibáñez-Berganza, M. Lewenstein, and G. Sierra, "Qubism: self-similar visualization of many-body wavefunctions," New J. Phys. **14**, 053028 (2012).
[51]W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C* (Cambridge U.P., Cambridge, 1988).
[52]Notice also that we did not provide a prescription for the division of the system into parts *A* and B. Different partitions will give rise to different entanglement entropies. Relevant choices usually imply a minimal *boundary* between the parts.
[53]B. Altshuler, H. Krovi, and J. Roland, "Anderson localization makes adiabatic quantum optimization fail," Proc. Natl. Acad. Sci. U.S.A. **107**, 12446–12450 (2010).
[54]The main requirement is that the commutator $[H_x,H_I]\neq 0$, which is immediately fulfilled for any diagonal HI.