Essay for Quantum Many-Body Physics (QST)

Simulation of Dynamical Quantum Phase Transitions using Cirq

Date: January 13, 2022

Submitted by:

Covone, Sebastiano Davide

03755593

Contents

List of Figures

1 Introduction

This essay focuses on the topic of Dynamical Quantum Phase Transitions (DQPTs). Our goal was to reproduce the results obtained in [[1]] using the library *Cirq*, developed by Google. The results were obtained in two different ways: first, by numerically simulating the wave function time evolution and then by simulating a real experiment, that can in principle be run on state of the art quantum hardware.
The structure of the essay is the following one:

- In 2 we briefly introduce the fundamentals ideas about DQPTs, focusing on the concepts of *Loschmidt rate* in DQPTs. The main references for this section were [2] and [3].
- In 3 we discuss the most important part of the written code and the results obtained by simulating the dynamics of the wavefunction of the considered system of a chains of spin with long ranged interactions. Lastly, in 4 we discuss the results obtained by simulating a "real" experiment.

2 Dynamical Quantum Phase Transitions

Let's consider a closed quantum system Let $H(\lambda)$ and Hamiltonian operator depending on a tunable parameter λ. If we prepare our quantum system in the ground state $|\psi_0\rangle$ of $H_0 = H(\lambda_0)$, we can quench the parameter lambda, by suddenly changing its value $\lambda_0 \to \lambda$. Then, in general, $|\psi_0\rangle$ will not be an eigenstate of $H(\lambda)$. The central object [4] in the study of DQPT is the amplitude

$$\mathcal{G}(t) = \langle\psi_0|\psi(t)\rangle = \langle\psi_0|e^{-iHt}|\psi(t)\rangle \tag{2.1}$$

and the associated probability $\mathcal{L}(t) = |\mathcal{G}(t)|^2$. The amplitude $\mathcal{G}(t)$ is formally identical to the canonical partion, so it is natural to think that in the thermodynamic limit it will be have the same functional dependence with respect to the system size N. Then,

$$\mathcal{G}(t) = e^{-Ng(t)} \tag{2.2}$$

where $g(t)$ plays the role of a complex free energy density. Since Quantum Phase transitions are defined as non-analytical points for some driving parameter (e.g. temperature) for the free energy density, we define the Dynamical Quantum Phase Transitions as points in time where the functions $g(t)$ becomes non analytical. Thanks to the return probability $\mathcal{L}(t)$, we define the most natural quantity to study DQPTs: the Loschmidt rate $\lambda(t) = 2Re[g(t)]$, defined by the relation

$$.\mathcal{L}(t) = e^{-N\lambda(t)} \tag{2.3}$$

As showed in [4], [1], [2] when $N < \infty$ we can find DQPTs by looking for "kinks" in the shape of the function $\lambda(t)$ that become *sharper* (in the sense their functional behaviour becomes similar to $\sim |t - t_c|$) in the limit $N \to \infty$. As explained also in [1] and [2], we can look for DQPTs by measuring the entanglement entropy of the system in time, since it appears that entanglement production is very closely related to DQPTs.

3 Simulations

We analyze the following model of a spin chain with long range interactions ([1], [2]):

$$\hat{H}(h) = H_0 + H_1 = -\sum_{i\neq j=1}^{N} V(i-j)\sigma_i^x\sigma_j^x - h\sum_{j=1}^{N}\sigma_j^z \tag{3.1}$$

where $V(i-j) \simeq |i-j|^{-\alpha}$. During the experiments [1], α and h can be considered as tunable parameters. Fixed α, the Hamiltonian is a function of h. We then initialize our system in one of the two ground states of $H_0 = H(h=0)$:

$$|\psi_0\rangle = |+,+,\ldots,+\rangle, \tag{3.2}$$

where, in the computational basis, $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$. At time $t=0$, we quench the parameter h to a finite value $h \neq 0$. The time evolution operator is then given by

$$\hat{U}(t) = e^{-iH(h)t} \tag{3.3}$$

and the evolution of the initial state is non-trivial since $|\psi_0\rangle$ is not an eigenstate of $H(h)$ when $h \neq 0$.

3.1 Time-step evolution

To simulate the Hamiltonian 3.1, we need to implement the unitary time evolution operator 3.3 in Cirq. First, we divide the time interval $[0,t]$ of finite steps of equal duration dt such that,

$$\hat{U}(t) = e^{-iH(h)t} = e^{-iH(h)dt[t/dt]} = (e^{-iH(h)dt})^M \tag{3.4}$$

, where $M = [t/dt]$ is the integer number of steps in the interval $[0,t]$ with spacing dt. If the time-step dt is choosed small enough we can approximate

$$(e^{-iH(h)dt}) = e^{-i(H_0+H_1)dt} = e^{-iH_0dt}e^{-iH_1dt} + o(dt) \approx e^{-iH_0dt}e^{-iH_1dt} \tag{3.5}$$

that can be easily proved using the Baker-Campbell-Hausdorff formula. Then,

$$\hat{U}(t) \simeq \hat{U}(Mdt) \approx \prod_{i=0}^{M-1} e^{-iH_0dt}e^{-iH_1dt}. \tag{3.6}$$

Now,

$$e^{-iH_0dt} = e^{i\sum_{i<j}V_{ij}\sigma_i^x\sigma_j^x dt} = \prod_{i=0}^{N-1}\prod_{j=0}^{i} e^{iV_{ij}\sigma_i^x\sigma_j^x dt}$$

$$e^{-iH_1dt} = e^{ih\sum_j \sigma_j^z dt} = \prod_{j=0}^{N-1} e^{ih\sigma_j^z dt}$$

since the operators in the sum in the exponential commute with each others. Finally,

$$\hat{U}(t) \approx \prod_{k=0}^{M-1}\left(\prod_{i=0}^{N-1}\prod_{j=0}^{i} e^{iV_{ij}\sigma_i^x\sigma_j^x dt}\right)\prod_{\ell=0}^{N-1} e^{ih\sigma_\ell^z dt} = \prod_{k=0}^{M-1}\hat{U}(dt). \tag{3.7}$$

3.2 Circuit initialization and Cirq implementation of the time-step evolution

To perform the simulation we need to *initialize* our circuit. In Cirq, this is done by defining an instance of the class "cirq.Circuit()". To initialize a set of N qubits in the $|\psi_0\rangle$ state, we do

```
qubits = cirq.LineQubit.range(N)
circuit = cirq.Circuit([cirq.H(q) for q in qubits])
```

Listing 1: Circuit initialization in Circuit.

where it is clear that "cirq.H(q)" is the *Hadamard* gate applied to the qubits q of the chain. Then, the time-step evolution operator $U(dt)$ is implemented in Cirq using the following code:

```
def U_Adt(N, h, dt, qubits, circuit) -> cirq.Circuit():
  """ Given a circuit and a set of qubits, return a quantum circuit with the U_Adt gate applied on all the qubits"""
  circuit.append([(cirq.rz(-2*h*dt)).on(q) for q in qubits])
  return circuit

def U_Bdt(N, J, alpha,  dt, qubits,v,circuit) -> cirq.Circuit():
  """  Given a circuit and a set of qubits and a set of coefficients v (numpy array NxN if there are N qubits in the circuit), return a quantum circuit with the U_Bdt gate applied on all the qubits"""
  for i, qi in enumerate(qubits):
    for j in range(i):
      circuit.append(cirq.XX(qi,qubits[j])**v[i][j])
  return circuit

def U_dt(N, J, h, alpha, qubits, dt, v, circuit = None) -> cirq.Circuit():
  """  Given a circuit and a set of qubits, return a quantum circuit with the U_dt gate applied on all the qubits"""
  if not circuit:
    circuit = cirq.Circuit()
  temp = U_Adt(N, h, dt, qubits,circuit)
  circuit = U_Bdt(N, J, alpha, dt, qubits, v, temp)
  return circuit
```

Listing 2: Implementation of the operator $U(dt)$ in Cirq.

where we defined $U_A(dt)$, $U_B(dt)$ such that $U(dt) = U_A(dt)U_B(dt)$. As an example, we show in figure 1 the quantum circuit that performs the initialization in the $|\psi_0\rangle$ state and a single step time evolution followed by a measurement in the X basis for N = 3 qubits.

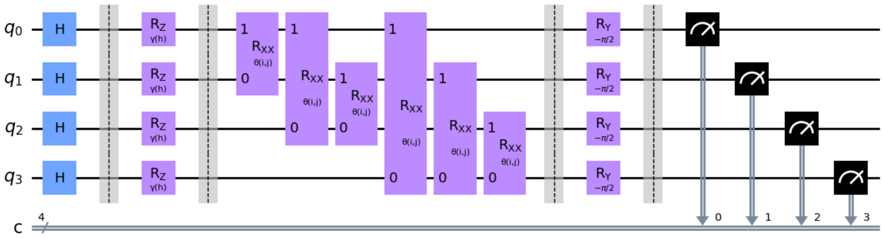

Figure 1: Example of a quantum circuit in which a single step evolution $U(dt)$ is perfomerd on $N = 3$ qubits initialized in the state $|+, +, \ldots, +\rangle$. The evolution is followed by a set of measurements in the X basis

.

3.3 Time evolution and results

3.3.1 Algorithm for time evolution

To replicate the figures in [1], we first simulated the time evolution of the expectation values of the average magnetization $\sigma_x(t)$ and the energy density $\varepsilon(t)$ for $|\psi(0)\rangle = |\psi_0\rangle$. The considered operators are defined as

$$\sigma_x(t) = \frac{1}{N}\sum_{i=1}^{N}\langle\psi(t)|\sigma_i^x|\psi(t)\rangle \tag{3.8}$$

$$\varepsilon(t) = \frac{1}{N}\langle\psi(t)|H_0|\psi(t)\rangle. \tag{3.9}$$

To compute the expectation value of a set of operators "ops" from the simulated wavefunction $|\psi(t)\rangle$, we wrote the following function that performs the expectation value calculation after each time-step evolution :

```python
def simulate_evals(operators, t_f, dt, initial_state, U_dt, u_dt_args qubits):
  """ Returns an array containing the expectation values of the operators in 'operators', for each time step in  time_steps = np.arange(0, t_f, dt) for a given time-step unitary evolution and initial state. """
  psi = initial_state/np.linalg.norm(initial_state)
  n = len(operators)
  out_array = np.zeros((n, time_steps.size), dtype = 'complex')
  for idx, t in enumerate(time_steps):
    circuit_dt = U_dt(*u_dt_args)
    result = simulator.simulate(circuit_dt, initial_state = psi)
    psi = result.state_vector()/np.linalg.norm(result.state_vector())
    evals = simulator.simulate_expectation_values(circuit_dt, observables = operators, initial_state = psi)
    out_array[:,idx] = evals
  return out_array
```

Listing 3: Simulation of expectation values in Cirq.

At first we did not considered the effects of decoherence.

3.3.2 Magnetization and energy density

We run the code above choosing $\alpha = 1.0$, $J/h = 0.5$ for different sizes N of the spin chain and for $t_f = 3.0/h, dt = 0.02/h$.

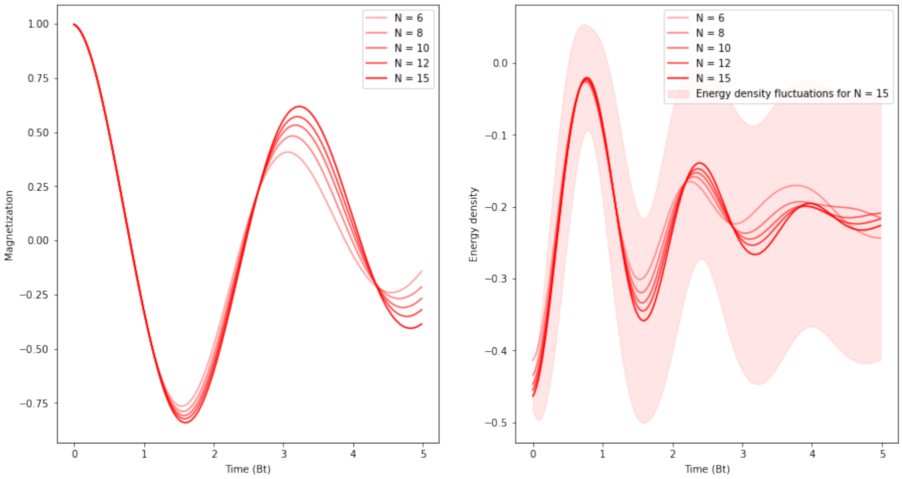

Figure 2: Average magnetization and energy density as function of time (measured in units of h) for different sizes of the system.

The results are plotted in **??**.We simulated the system up to N = 15. We did not go further since the running time for our simulation scales faster than an exponential as a function of the size of the system. This assertion can be motivated by looking at Figure**??** where the running time as function of the system size N is fitted against two models: the first one, exponential, a^N, the second one a^{b^N}. The estimated running time for, to say, $N = 20$ is approximately 8 hours.

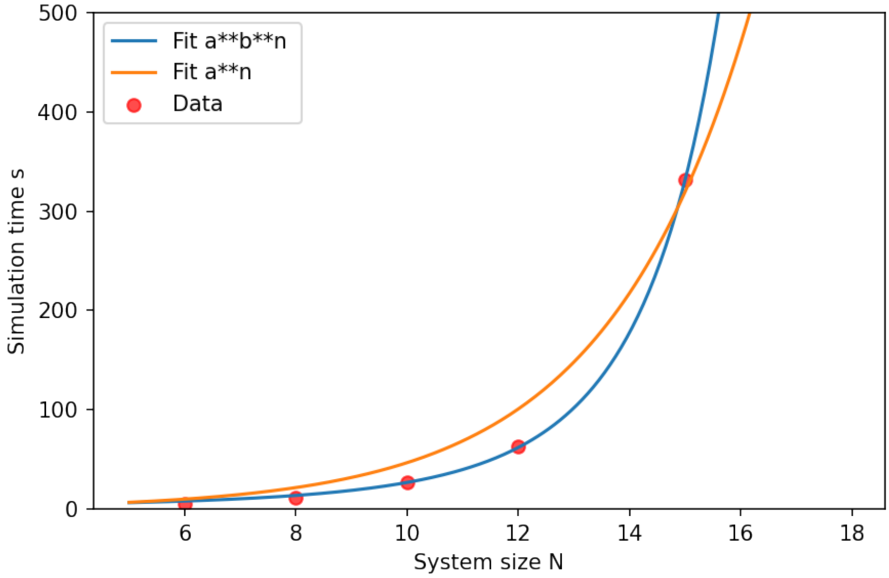

Figure 3: Running time of the simulation as function of the system size N.

3.3.3 Loschmidt rate and Entanglement entropy

With the same values of the physical parameters and for the same values of the system sizes, we simulated the time evolution of the Loschmidt rate $\lambda(t)$ and entanglement entropy of the system $\mathcal{S}(t)$, measured using the Renyi Entropy:

$$\mathcal{S}(t) = -\mathbf{Tr}\hat{\rho}_h^2 \tag{3.10}$$

where $\hat{\rho}_H$ is the reduced density matrix of half the qubit chain. The results are plotted in figure 4

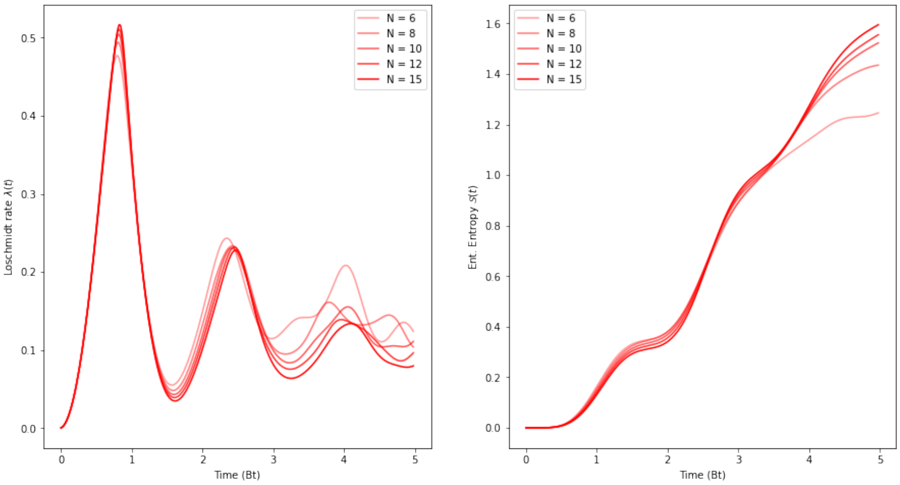

Figure 4: Loschmidt rate and Renyi Entropy as function of time for different sizes of the chain.

3.4 Summary

In figure 5 we summed the results of the simulation for size of the system $N = 8$. We can see how, as stated in [1], we have the "kink" in the Loschmidt rate approximately when the average magnetizations is zero and the energy density is maximal. The three functions show indeed the same period.

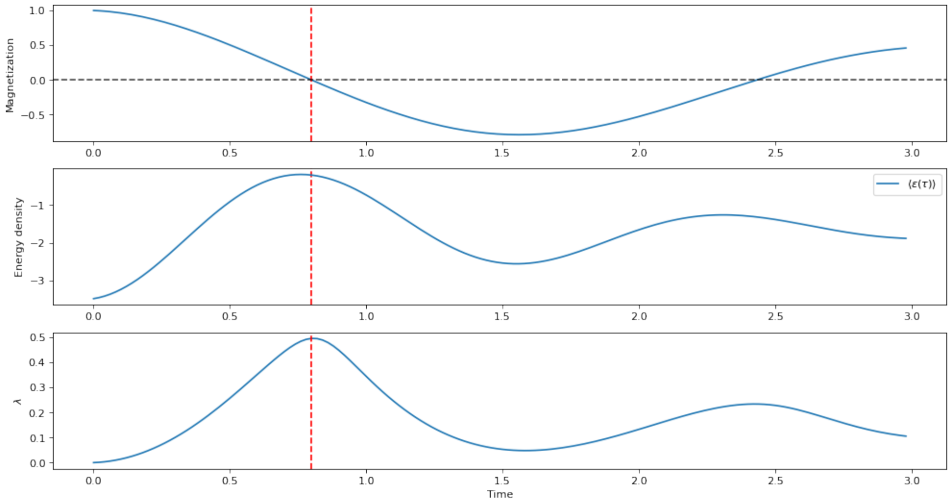

Figure 5: Caption

4 Measuring the system dynamics

After having simulated the system dynamics by calculating numerically the system's wavefunction at each time steps, we simulated the system dynamics by emulating what can be done with state of the art quantum hardware. This can be done in Cirq by using the cirq.Simulator.run(circuit, repetitions) which simulated $n = repetitions$ measurement of the input circuit and store the resuts histograms. The time evolution algorithm in this case is a little bit more tricky since each measurements collapses the system wavefunction in one of the eigenstates of the measurement basis. For this reason, before each measurement, we have to "copy" the state of the circuit. The code to perform the time-step evolution in presence of quantum measurements is the following one:

```
    circ = initial_circuit
    for i in range(0,time_steps.size):
    circuit_dt = U_dt(N, J, h, alpha, qubits, dt, v, circuit = circ)
    circuit_temp = circuit_dt.copy()
    if i % m == 0:
      circuit_dt.append([cirq.ry(-np.pi/2).on(q) for q in qubits])
      circuit_dt.append([cirq.measure(q, key = 'qubit {}, {}'.format(j,r)) for j,q in enumerate(qubits)])
      results = simulator.run(circuit_dt, repetitions = runs)
      """ do something to calculate exp. values or probabilities """
    circ = circuit_temp
```

Listing 4: Code to measure the average magnetization.

Note that we performed a measurement each $m = 5$ steps in time $dt = 0.02/h$. Also, we choose to perform the measurements in the time interval $[0, t_f]$ with $t_f = 3/h$.

4.1 Magnetization

The first quantity we measured was the average magnetization $\sigma_x(t)$ for $N = 6, 8, 10$ sizes of the system. The measure was performed in the following way: at each time t, we measured each qubit q in the X basis for $r \gg 1$ times. For each qubit we estimated $\langle\sigma_x^i\rangle\rangle = (n_1 - n_0)/r$ where $n_1(n_0)$ counts the number of time we got "$1(0)$" measuring the qubit i. The average magnetization $\sigma_x(t)$ was then calculated using the definition. The results are plotted in figure 6. The results are in agreement with the simulation shoved before. The code to perform the measurement of the average magnetization is the following:

```
    results = qsim_simulator.run(circuit_dt, repetitions = runs)
    for k,q in range(enumerate(qubits):
      hist = results.histogram(key = 'qubit {}'.format(k))
      sigma_x_avg.append((-hist[1] + hist[0])/runs)
    avg_magnetization_t.append(np.average(np.array(sigma_x_avg)))
```

Listing 5: Code to measure the average magnetization.

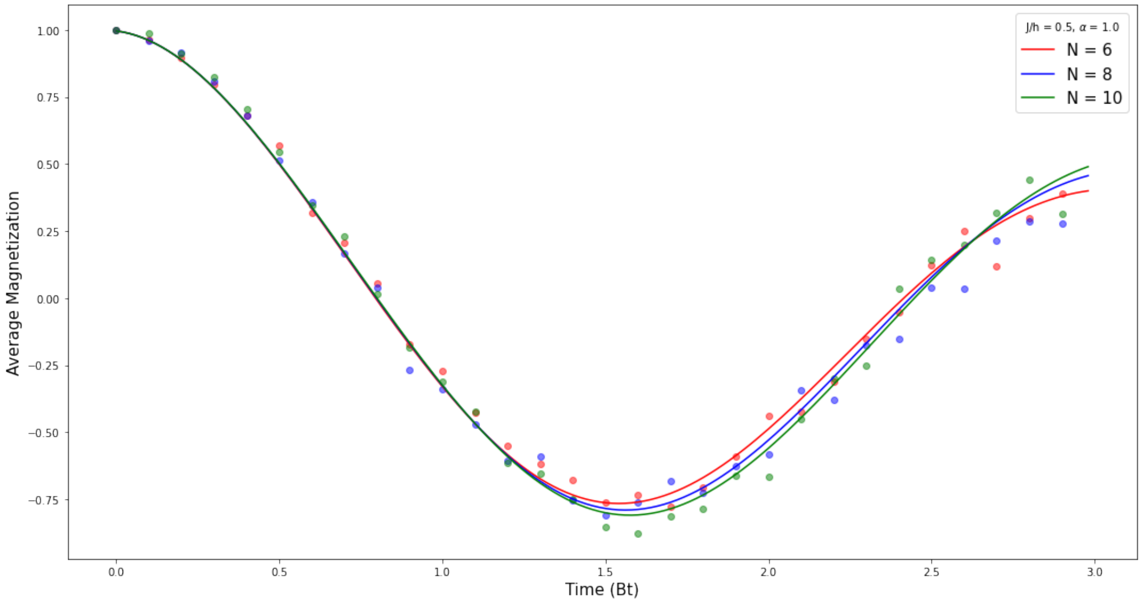

Figure 6: Measures of the Average magnetization (points) and simulated data for the same quantity (solid lines) for different sizes of the system. The parameters were $\alpha = 1.0$, $J/h = 0.5$ and $dt = 0.02/h$. Each points is the average of $runs = 1000$ measurements.

4.2 Loschmidt rate

To measure the Loschmidt rate $\lambda(t)$ at a given time t, we measure the qubits in the chain in the X basis $r \gg 1$ times . As usual, to perform the measurement in the X basis we perform a $-\pi/2$ rotation around the $yaxis$ of the Bloch's Sphere before applying the measurement gate (implemented by cirq.measure(q) for a given qubit q). For each measurement we save the obtained bitstring **s**. The probability to stay in the first ground state $|+, +, \dots, +\rangle$ is estimated by n_+/r where $n_+ =$ of bitstrings $(1, 1, \dots, 1)$. In the same way, the probability to stay in the second ground state is estimated via n_-/r where $n_- =$ of bitstrings $(0, 0, \dots, 0)$.The Loschmidt rate is then calculated using the definition

$$\lambda(t) = -\frac{1}{N} log(P_+ + P_-). \tag{4.1}$$

We measured the Loschmidt rate of the system dynamics for $N = 6, 8, 10$, keeping the parameters α, J and h fixed. The results are plotted in Figure 7. Each points is the average of $runs = 1000$ measurements, The results are in agreement with the data obtained from the simulation. The code wrote to perform the measurement of the Loschmidt ratio is the following one:

```
    results = simulator.run(circuit_dt, repetitions = runs)
    keys = ['qubit {}'.format(k) for k in range(N)]
    counts = results.multi_measurement_histogram(keys = keys)
    key0 = tuple( [0] * N )
    key1 = tuple( [1] * N )
    probability_0 = counts[key0]/runs
    probability_1 = counts[key1]/runs
    p_t[i] = probability_0 + probability_1
lambda[t] = -1/N*np.log(p_t)
```

Listing 6: Code to measure the Loschmidt rate.

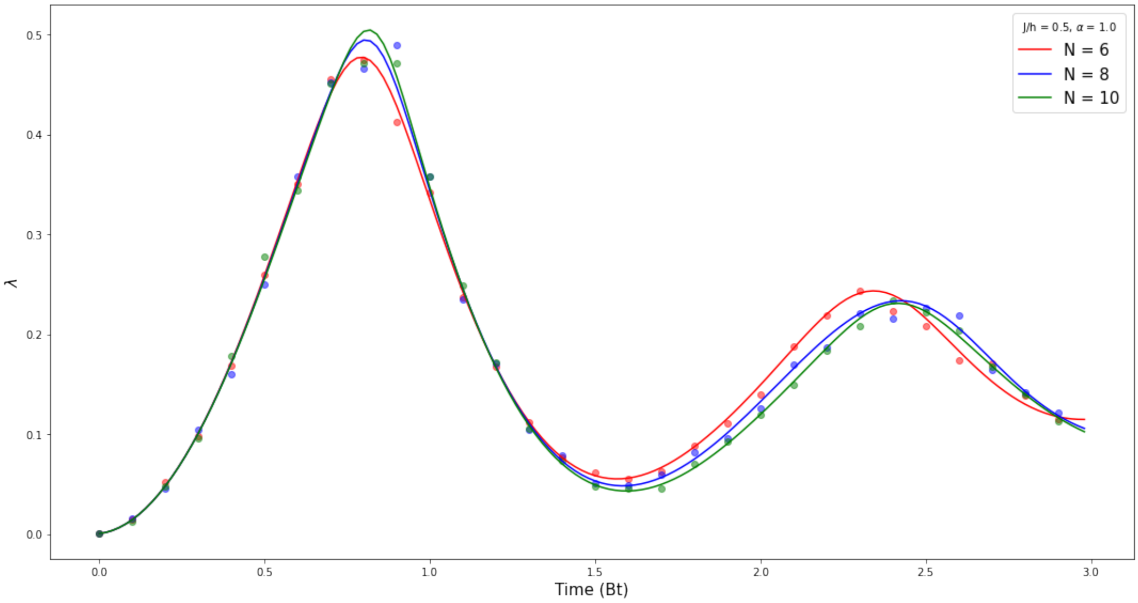

Figure 7: Measures of the Loschmidt rate (points) and simulated data for the same quantity (solid lines) for different sizes of the system. The parameters were $\alpha = 1.0$, $J/h = 0.5$ and $dt = 0.02/h$. Each points is the average of $runs = 1000$ measurements.

References

[1] P. Jurcevic, H. Shen, P. Hauke, C. Maier, T. Brydges, C. Hempel, B. P. Lanyon, M. Heyl, R. Blatt, and C. F. Roos, “Direct observation of dynamical quantum phase transitions in an interacting many-body system”, *arXiv*, Dec. 2016. DOI: 10.1103/PhysRevLett.119.080501. eprint: 1612.06902.

[2] B. Zunkovic, M. Heyl, M. Knap, and A. Silva, “Dynamical Quantum Phase Transitions in Spin Chains with Long-Range Interactions: Merging different concepts of non-equilibrium criticality”, *arXiv*, Sep. 2016. DOI: 10.1103/PhysRevLett.120.130601. eprint: 1609.08482.

[3] M. Heyl, “Dynamical quantum phase transitions: a review”, *arXiv*, Sep. 2017. DOI: 10.1088/1361-6633/aaaf9a. eprint: 1709.07461.

[4] A. Elben, B. Vermersch, C. F. Roos, and P. Zoller, “Statistical correlations between locally randomized measurements: a toolbox for probing entanglement in many-body quantum states”, *arXiv*, Dec. 2018. DOI: 10.1103/PhysRevA.99.052323. eprint: 1812.02624.