

UNIVERSIDADE DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

**Solving Nonlinear Equations and Correcting Errors on Quantum
Computers**

Diogo da Silva Duarte Cruz

Supervisor: Doctor Yasser Rashid Revez Omar

Co-Supervisor: Doctor João Carlos Carvalho de Sá Seixas

Thesis approved in public session to obtain the PhD Degree in

Physics

Jury final classification: **Pass with Distinction**

UNIVERSIDADE DE LISBOA
INSTITUTO SUPERIOR TÉCNICO

Solving Nonlinear Equations and Correcting Errors on Quantum Computers

Diogo da Silva Duarte Cruz

Supervisor: Doctor Yasser Rashid Revez Omar

Co-Supervisor: Doctor João Carlos Carvalho de Sá Seixas

Thesis approved in public session to obtain the PhD Degree in

Physics

Jury final classification: **Pass with Distinction**

Jury

Chairperson: Doctor Paulo Jorge Peixeiro de Freitas, Instituto Superior Técnico, Universidade de Lisboa

Members of the Committee: Doctor Mikhail Igorevich Vasilevskiy, Escola de Ciências, Universidade do Minho

Doctor Shaukat Ali, Department of Engineering Complex Software Systems, Simula Research Laboratory, Norway

Doctor Andreas Miroslaus Wichert, Instituto Superior Técnico, Universidade de Lisboa

Doctor Yasser Rashid Revez Omar, Instituto Superior Técnico, Universidade de Lisboa

Doctor Mikel Sanz, Department of Physical Chemistry, Faculty of Science and Technology, Universidad del País Vasco (UPV/EHU), Spain

Funding Institution

FCT: Fundação para a Ciência e a Tecnologia

Acknowledgments

I acknowledge the financial support provided by Fundação para a Ciência e Tecnologia, Portugal, through scholarship UI/BD/152301/2021.

I would like to thank my supervisors Prof. Yasser Omar and Prof. João Seixas for their guidance, without which this work would not have been possible. They provided me with the opportunity to meaningfully contribute to the research area I most enjoy, despite the little knowledge I initially had of the subject.

I am also grateful to my hosting institutions for their support, namely Instituto Superior Técnico, Instituto de Telecomunicações, and the Portuguese Quantum Institute. In particular, I would like to thank Tereza Traquinas, Fátima Barata, Rafaela Ribeiro, and Jaqueline Silva for their aid during my stay. Additionally, I thank Massachusetts Institute of Technology for hosting me as a visiting student.

Prof. Nuno Loureiro, Prof. Francisco Monteiro, Dr. Bruno Coutinho, Dr. Duarte Magano, Akshat Kumar, André Roque, and Dr. Sagar Pratapsi were critical to the success of this thesis, providing priceless help and extensive feedback on numerous parts, and the published work it led to. It is safe to say that a significant part of this work would not exist without them, and for their help I am eternally grateful.

I would also like to thank all the colleagues who helped me throughout my PhD, in one way or another, and all the help of my friends and family, who constantly encouraged me to continue my work, even at the times when that was all but what I wished to do, and so ensured I would finish my thesis.

Abstract

Quantum computing takes advantage of inherent properties of quantum mechanics to perform computations that would be intractable on classical computers. However, current quantum devices are susceptible to noise, and error correction techniques are needed to make reliable computation possible. With this thesis, we present contributions along two intrinsically connected research directions. We develop a quantum algorithm for solving certain nonlinear differential equations, and propose quantum error correction methods suitable for both near-term noisy intermediate-scale quantum (NISQ) devices and future fault-tolerant quantum computers.

Our first contribution is a quantum algorithm for the forward scattering transform. This spectral analysis technique computes scattering data from the initial conditions of nonlinear integrable systems, and is the first and most computationally demanding step of the inverse scattering transform, a classical method that decomposes certain nonlinear partial differential equations into sequences of linear steps. Our algorithm achieves a quadratic speedup in the spatial discretization compared to classical methods. Additionally, it applies broadly to entire hierarchies of integrable nonlinear equations sharing the same spatial Lax operator. As a concrete demonstration, we compute the reflection coefficients for the Korteweg-de Vries equation, a prototypical nonlinear wave equation with applications in fluid mechanics, plasma physics, and other fields.

Our second contribution is quantum GRAND (Guessing Random Additive Noise Decoding), to our knowledge, the first practical maximum-likelihood decoder for quantum random linear codes. These codes achieve optimal error correction performance asymptotically, but have historically been impractical because their lack of algebraic structure prevented the development of efficient decoders. Rather than relying on code structure, QGRAND exploits noise statistics to identify the most likely error patterns. We extend QGRAND to the fault-tolerant setting by modeling the full error process during syndrome extraction, and achieve a threshold error rate of approximately 5.5×10^{-3} . The techniques we develop also enable partial error correction matched to available hardware capabilities. These results confirm theoretically optimal quantum codes as a viable option for fault-tolerant quantum computing.

Keywords: quantum computing, quantum algorithms, nonlinear differential equations, quantum error correction, fault-tolerant quantum computing

Resumo

A computação quântica tira partido de propriedades inerentes à mecânica quântica para realizar computações que seriam intratáveis em computadores clássicos. No entanto, os dispositivos quânticos atuais são suscetíveis a ruído, e técnicas de correção de erros são necessárias para possibilitar computação fiável. Nesta tese, apresentamos contribuições em duas direções de investigação intrinsecamente ligadas. Desenvolvemos um algoritmo quântico para resolver certas equações diferenciais não lineares e propomos métodos de correção de erros quânticos adequados tanto para dispositivos NISQ (Noisy Intermediate-Scale Quantum) de curto prazo como para futuros computadores quânticos tolerantes a falhas.

A nossa primeira contribuição é um algoritmo quântico para a *forward scattering transform*. Esta técnica de análise espectral calcula a *scattering data* a partir das condições iniciais de sistemas integráveis não lineares, e é o primeiro e mais exigente passo computacional da *inverse scattering transform*, um método clássico que decompõe certas equações diferenciais parciais não lineares em sequências de passos lineares. O nosso algoritmo alcança uma aceleração quadrática na discretização espacial comparado com métodos clássicos. Adicionalmente, aplica-se a hierarquias inteiras de equações não lineares integráveis que partilham o mesmo operador de Lax espacial. Como demonstração concreta, calculamos os coeficientes de reflexão para a equação de Korteweg-de Vries, uma equação de ondas não lineares prototípica com aplicações em mecânica de fluidos, física de plasmas e outros campos.

A nossa segunda contribuição é o GRAND quântico (Guessing Random Additive Noise Decoding), tanto quanto sabemos, o primeiro decodificador prático de *maximum-likelihood* para códigos lineares aleatórios quânticos. Estes códigos atingem um desempenho ótimo de correção de erros assintoticamente, mas têm sido historicamente impraticáveis porque a sua falta de estrutura algébrica impediu o desenvolvimento de decodificadores eficientes. Em vez de depender da estrutura do código, o QGRAND explora as estatísticas do ruído para identificar os padrões de erro mais prováveis. Estendemos o QGRAND para o contexto tolerante a falhas, modelando o processo de erro completo durante a extração de síndromes, e atingimos uma taxa limiar de erro (*threshold error rate*) de aproximadamente $5,5 \times 10^{-3}$. As técnicas que desenvolvemos também possibilitam correção parcial de erros adequada às capacidades do hardware disponível. Estes resultados confirmam os códigos quânticos teoricamente ótimos como uma opção viável para computação quântica tolerante a falhas.

Palavras-chave: computação quântica, algoritmos quânticos, equações diferenciais não lineares, correção de erros quânticos, computação quântica tolerante a falhas

Contents

Acknowledgments	v
Abstract	vii
Resumo	ix
Structure and Contributions	1
0.1 Structure of the Thesis	1
0.2 Other contributions	1
1 Introduction	3
1.1 Quantum Computing	4
1.2 Tackling Differential Equations	7
1.3 Quantum Error Correction	11
2 Quantum Forward Scattering Transform for Nonlinear Differential Equations	15
2.1 Introduction	15
2.2 Forward scattering	18
2.2.1 Complexity analysis	21
2.3 Discussion	21
2.4 Conclusion	23
Appendices	23
A Determining Lax Pairs	23
B Scattering data from a linear time-dependent ODE	24
C Properties of $\mathbf{A}_c$	31
D Derivatives of $\mathbf{c}$	35

E	Relation between $\mathbf{c}$ and u_0	41
F	Minimizing cutoff error	43
G	Implementing the forward scattering method	47
H	Formal complexity statement	51
3	Quantum Error Correction via Noise Guessing Decoding	53
3.1	Introduction	53
A	Short linear codes: a historical context	54
B	Scope and Contributions	56
C	Organization and Notation	58
3.2	Toward Shannon's Codes: Classical RLCs	58
A	Decoding Classical RLCs by Noise Guessing	59
3.3	Quantum error correction	60
3.4	Stabilizer codes	61
3.5	Quantum RLCs encoding	62
A	Robust encoding	63
B	Comparison with ideal random codes	64
C	Minimum number of random gates	67
D	Determining the stabilizers	68
3.6	Quantum-GRAND proposals	68
A	The syndrome as membership test	69
B	Syndrome decoding	71
C	Ordering the stabilizers	72
D	Adapting to changing noise statistics	72
E	A general approach: beyond stabilized codes	72
3.7	Performance vs code rate trade-off	74
3.8	Conclusions and future directions	77
3.9	Supporting Information	78
A	Stabilizer order	78

4	Fault-tolerant noise guessing decoding of quantum random codes	81
4.1	Introduction	81
4.2	Setup and noise model	84
4.3	Error correction overview	85
4.4	Decoding	92
A	Function $\mathcal{F}_E$	94
B	Function $\mathcal{F}_L$	95
4.5	Asymptotic regime	97
4.6	Discussion and conclusions	102
4.7	Supporting Information	103
A	Reducing stabilizer weight	103
B	Degeneracy analysis	104
C	Analytical threshold without degeneracy	105
D	Applying error correction	108
E	Parallel decoding	108
F	Full decoding example	109
G	Simplified noise statistics	111
H	Bernoulli noise model improvements	112
I	Derivation of decoding formulas for Bernoulli noise	115
J	Alternative definitions of $\mathcal{F}_E$ and $\mathcal{F}_L$	119
K	Analysis of numerical results	122
L	Single-qubit gate error model considerations	129
5	Conclusions	135
5.1	Summary of contributions	135
5.2	Outlook	136
	Bibliography	139

Structure and Contributions

0.1 Structure of the Thesis

The thesis is structured as follows: the first chapter consists of an introduction on the topic of quantum computing and quantum error correction. Chapters 2 to 4 consist of 3 original research papers.

Chapter 2 corresponds to a paper in preparation:

Diogo Cruz, Nuno Loureiro, Yasser Omar, and Akshat Kumar. “Quantum Forward Scattering Transform for Nonlinear Differential Equations”. Manuscript in preparation. 2025.

Chapter 3 corresponds to the following peer-reviewed publication:

Diogo Cruz, Francisco A. Monteiro, and Bruno C. Coutinho. “Quantum Error Correction via Noise Guessing Decoding”. In: *IEEE Access* 11 (Oct. 2023), pp. 19446–19461. DOI: 10.1109/ACCESS.2023.3327214.

Chapter 4 corresponds to the following peer-reviewed publication:

Diogo Cruz, Francisco A. Monteiro, André Roque, and Bruno C. Coutinho. “Fault-tolerant noise guessing decoding of quantum random codes”. In: *IEEE Transactions on Quantum Engineering* 6 (2025), pp. 1–27. DOI: 10.1109/TQE.2025.3595778.

0.2 Other contributions

Throughout the course of his PhD, the author has also contributed to the following work:

Diogo Cruz and Duarte Magano. “Superresolution of Green’s functions on noisy quantum computers”. In: *Phys. Rev. A* 108 (1 July 2023), p. 012618. DOI: 10.1103/PhysRevA.108.012618. URL: <https://link.aps.org/doi/10.1103/PhysRevA.108.012618>.

Sagar Silva Pratapsi, Diogo Cruz, and Paulo André. “Native Multi-Qubit Gates in Transmon Qubits via Synchronous Driving”. In: *Scientific Reports* 14.1 (Oct. 2024), p. 26042. ISSN: 2045-2322. DOI: 10.1038/s41598-024-76396-9.

Óscar Amaro and Diogo Cruz. *A Living Review of Quantum Computing for Plasma Physics*. arXiv:2302.00001 [physics, physics:quant-ph]. Jan. 2023. DOI: 10.48550/arXiv.2302.00001. URL: <http://arxiv.org/>

abs/2302.00001 (visited on 02/02/2023).

André Roque, Diogo Cruz, Francisco A. Monteiro, and Bruno C. Coutinho. “Efficient entanglement purification based on noise guessing decoding”. In: *Quantum* 8 (Sept. 2024), p. 1476. ISSN: 2521-327X. DOI: 10.22331/q-2024-09-19-1476.

Chapter 1

Introduction

Quantum computing has established itself as one of the most significant technological developments of this century. Quantum computers rely on fundamental properties of quantum mechanics, such as superposition, entanglement, and interference, to process information in ways that are inaccessible to classical systems. It could therefore revolutionize many fields of science and engineering that have traditionally been limited by classical computation. Nonetheless, for this potential to be realized, several theoretical and practical challenges must be overcome. In particular, new algorithms and error correction techniques must be developed that are natively built for this new type of computing device.

Differential equations are at the heart of many problems in physics and mathematics. Unfortunately, for certain classes of problems, it is intractable to solve them efficiently with classical computers to the desired precision. Since some problems can be solved more efficiently by quantum than classical computers, there has been a great deal of interest to determine to what extent these important differential equations can be mapped to an efficient quantum algorithm that could provide a meaningful speed-up compared to the classical alternative. However, even if such a mapping could be found, due to the current fragility of quantum computers, considerably complex error correction techniques would be needed to allow for these algorithms to actually be implemented and run for extended periods of time. The necessity of better quantum algorithms and better error correction techniques then brought about the research presented in this thesis.

The work presented here is structured around two main research directions. On one hand, developing better quantum algorithms for solving differential equations, especially those that are classically challenging, such as non-linear partial differential equations. On the other hand, developing better quantum error correction methods that are suitable for future fault-tolerant quantum computers. We see these two research directions as being intrinsically connected, and therefore this work as a way of advancing both our theoretical understanding of quantum algorithms and their possible use cases, and the practical implementation of these algorithms in physical quantum devices.

1.1 Quantum Computing

Quantum computing takes advantage of inherent properties of quantum mechanics to perform computations that would be intractable on classical computers. The initial foundations of the field were laid in the late 20th century with Feynman’s seminal observation that quantum systems can efficiently simulate other quantum systems. Later, Deutsch established its first theoretical foundations by formalizing the quantum Turing machine. In the subsequent decades, we have observed remarkable progress in the field, from novel theoretical insights to experimental demonstrations of quantum computational advantage.

While classical computers rely on bits which can be either 0 or 1, quantum computers use instead quantum bits (qubits), which can exist in a superposition of both states. A single qubit state is represented as $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$, where $\alpha, \beta \in \mathbb{C}$ are normalized so that $|\alpha|^2 + |\beta|^2 = 1$. This ability of a quantum state to be in superposition extends to multi-qubit systems, with an n -qubit state being an element of a 2^n -dimensional Hilbert space. The exponential scaling of this state space with the number of qubits leads to the power of quantum computing, by enabling quantum algorithms to explore vastly greater solution spaces for the same (qu)bit count.

Since quantum mechanics underlies quantum computing, the same mathematical formalism applies. Quantum states evolve through unitary transformations $U \in \text{U}(2^n)$, thereby preserving the state’s norm and allowing for reversible computation. Separable qubit systems $\mathcal{H}_A, \mathcal{H}_B, \dots$ can be collectively described by their tensor product as a combined system $\mathcal{H} = \mathcal{H}_A \otimes \mathcal{H}_B \otimes \dots$. For composite multi-qubit systems that cannot be separated, the partial trace operation $\text{Tr}_B[\rho_{AB}]$ can capture subsystem properties. Quantum measurements are described by positive operator-valued measures (POVMs) and allow the extraction of classical information from quantum states. If the measured quantum state is in superposition, this results in wave function collapse, with the state irreversibly becoming one of the particular basis states, with probability assigned to each basis state given by its amplitude squared. This property, known as the measurement postulate, introduces notable differences between the way quantum and classical information is handled, and as a result, care is required when designing quantum algorithms, so that the output data is extracted in an efficient manner that enables a computational speedup.

Quantum computers benefit from three key quantum mechanical properties that have no classical analogue. First, *superposition* enables a quantum system of n qubits to represent all 2^n computational basis states simultaneously. Unfortunately, this level of parallelism does not allow directly for a computational advantage, since the measurement procedure can only extract a limited amount of classical information. Second, *entanglement* allows the creation of correlations between qubits that cannot be described through classical probability distributions. As a result, entangled qubits are better thought of as a single entity, rather than as a collection of independent states. This property allows quantum algorithms to exploit global properties of a larger state space. Third, *interference* allows for the constructive amplification of computational paths that are desirable, while simultaneously destructively interfering the undesirable ones. This interference mechanism can then be exploited by quantum algorithms to efficiently extract informative outputs from exponentially large superpositions.

Quantum algorithms can be described using the framework of quantum circuits. These circuits consist of sequences of quantum gates drawn from a universal gate set, such as $\{H, T, \text{CNOT}\}$, where H represents the Hadamard gate, T the $\pi/8$ phase gate, and CNOT the controlled-NOT gate. The Solovay-Kitaev theorem [8] guarantees that any unitary transformation can be approximated to precision ϵ using $O(\log^c(1/\epsilon))$ gates from a finite universal set, where $c < 4$, thereby guaranteeing that any arbitrary quantum algorithm can be implemented using low-level gates and bounded overhead. The quantum circuit framework has become the default method for thinking about and analyzing quantum algorithms, since it allows them to be described in a modular way, and provides an intuitive form of visualization.

Over the years, various quantum algorithms have been presented for specific problems that demonstrate exponential speedups over their best known classical counterparts. Shor’s algorithm [9] provides an exponential speedup for the problem of integer factorization, by taking advantage of the quantum version of the Fourier transform in order to efficiently find periods in modular exponentiation, a key subroutine of some factorization techniques. Its complexity scales with $O(n^3)$ for factoring an n -bit integer, notably more efficient than the best known classical algorithm, with sub-exponential scaling. As a result of this breakthrough, public key cryptographic systems could potentially be broken by future quantum computers, and new post-quantum cryptography techniques are being developed that are hoped to be quantum resilient. Also famous is Grover’s algorithm [10], which provides a quadratic speedup for unstructured search by finding a marked element among N in $O(\sqrt{N})$ queries, vs the classical $O(N)$. Although it is a less impressive improvement than Shor’s algorithm, it does apply broadly to any problem that is reducible to unstructured search. The quantum Fourier transform acts as a cornerstone in many quantum algorithms designed for problems that have some kind of inherent periodic structure. The QFT corresponds to the discrete Fourier transform applied to quantum amplitudes, mapping $|x\rangle \mapsto \frac{1}{\sqrt{N}} \sum_{y=0}^{N-1} \omega_N^{xy} |y\rangle$, where $\omega_N = e^{2\pi i/N}$. Notably, the QFT can be implemented using only $O(n^2)$ gates, with $N = 2^n$, while the classical fast Fourier transform would take $O(N \log N)$ operations, an exponentially higher amount. Therefore, for some problems, properties such as periods or eigenvalues can be efficiently obtained using quantum algorithms that rely on the QFT. Besides Shor’s algorithm, the QFT plays an important role in efficiently solving so-called *hidden subgroup problems* and in simulating physical quantum systems.

The initial quantum algorithms developed were mostly academic in nature. More recent developments have sought to apply these quantum approaches to problems of practical value. For example, the HHL algorithm [11] constitutes a quantum approach to solving linear systems of equations, which have a wide range of applications, such as differential equations and machine learning. Under specific conditions, it can achieve an exponential speedup. Additionally, quantum walk algorithms [12] lead to polynomial speedups for some spatial search and graph problems. These algorithms, however, are only feasible to use in robust quantum computers with negligible errors and hardware limitations, which do not yet exist. For the near-term, variational quantum algorithms [13], such as the variational quantum eigensolver (VQE) and the quantum approximate optimization algorithm (QAOA), rely on a hybrid classical-quantum optimization approach to find solutions in a way that is more robust to hardware issues. Quantum machine learning algorithms [14], often relying on similar principles, have also been proposed, promising advantages for

some specific learning and optimization tasks. However, for these near-term methods, it is yet unclear to what extent practical quantum advantage can actually be observed, as the theoretical guarantees are currently more limited.

Currently, there are several physical platforms that have matured enough to act as substrates for quantum computing. The most popular rely on superconducting qubits which are based on Josephson junctions that operate at millikelvin temperatures. This type of architecture has achieved a quantum computational advantage for sampling problems [15]. These systems can scale relatively rapidly since they can benefit from lithographic fabrication techniques that have been established in classical semiconductor manufacturing. Major players, such as IBM and Google, have bet on this approach, and the associated qubit counts and gate fidelities have steadily improved over the past few years. However, other approaches have also received significant interest. For example, trapped-ion systems can achieve the highest gate fidelities among current platforms [16]. They work by confining individual atomic ions using electromagnetic fields. Since these trapped ions can be individually controlled and have long coherence times, they are an attractive candidate for precision quantum operations. That being said, scaling them to larger qubit counts remains challenging.

Other approaches have their benefits and drawbacks, and their characteristics influence the types of algorithms and error correction techniques that are pursued and implemented in practice. Photonic quantum computers benefit from being operable at room temperatures and are naturally resilient to certain types of noise. These work by encoding information in photon states [17]. However, since photon-photon interactions require non-linear optical elements, it is particularly tricky to implement two-qubit gates. Neutral atom approaches rely on optical tweezers to control and arrange atoms as desired, and as a result, enable relatively flexible qubit connectivities [18]. Topological quantum computing is built under the premise that it has inherent, guaranteed protection against certain types of quantum errors, but is unfortunately still largely theoretical [19].

Current quantum devices still have relatively limited qubit counts (ranging from 50 to 1000) and notably high noise levels. Consequently, they are considered to operate in the noisy intermediate-scale quantum (NISQ) regime [20], preceding a future fault-tolerant era where qubit counts and connectivities, and their associated noise levels, can largely be abstracted away at a high level, and quantum computers can be thought of being noise-free in the same way that classical computers currently are. Current gate error rates range from 10^{-3} to 10^{-2} , which limits circuit depth to at most hundreds of gates, since decoherence becomes a significant issue for larger circuits. As a result, current NISQ devices require quantum algorithms and error mitigation strategies that take into account the specific features and limitations of these devices, so that useful results can actually be extracted. To go from the current state-of-the-art to future fault-tolerant quantum computers, significant improvements along the whole stack will have to take place, from reducing the hardware limitations current devices to developing better quantum algorithms.

While the standard quantum circuit model is the default theoretical framework used for quantum computing, there are alternative paradigms. Adiabatic quantum computation [21] takes advantage of the

adiabatic theorem to solve optimization problems. It starts with an easily prepared ground state that is then slowly evolved to the ground state of a chosen system that encodes the desired solution. While the standard circuit model involves discrete gates, and the adiabatic model involves continuous evolution, it turns out that the two paradigms are polynomially equivalent in computational power [22]. Measurement-based quantum computation takes pre-prepared entangled resource states and performs the desired quantum operations by adaptively measuring selected parts of the resource state. As a result, it shifts the complexity of gate operations to the complexity of preparing the necessary resource state.

In complexity theory, quantum computing can be neatly placed within the broader computational hierarchy. The class BQP (bounded-error quantum polynomial time) corresponds to problems that can be efficiently solved using quantum computers, and is the quantum equivalent of the class BPP. It satisfies $\text{BPP} \subseteq \text{BQP} \subseteq \text{PSPACE}$. It is not yet known what is the exact relationship between the BQP and NP classes, but current evidence indicates that BQP contains problems that are likely outside of NP, such as integer factorization. Other quantum complexity classes, such as QMA (quantum Merlin-Arthur) and QIP (quantum interactive proofs), further extend classical complexity theory to encompass the quantum domain. These complexity frameworks help in the search for new quantum algorithms and in establishing fundamental limits on the problems that quantum computers can efficiently solve.

1.2 Tackling Differential Equations

Differential equations are a fundamental tool to describe dynamical systems across science and engineering, from the Navier-Stokes equations that govern fluid flow to the Schrödinger equation that underlies quantum evolution. Computing the solution of differential equations, particularly nonlinear partial differential equations (PDEs), is one of the most computationally intensive tasks in the field of scientific computing. Classical numerical methods run into fundamental limitations when dealing with high-dimensional systems or problems that require high precision. As a way to address these challenges, quantum algorithms could potentially provide notable speedups for these classically computationally intractable problems. In this section, we document the current landscape of quantum algorithms for differential equations, with a particular focus on recent advances for nonlinear problems.

The history of classical numerical methods for solving differential equations spans centuries, from the traditional Euler’s method to the current state-of-the-art high-performance computing techniques. Finite difference methods work by discretizing derivatives using Taylor expansion, and they achieve an accuracy $O(h^p)$ for step size h and method order p . Finite element methods describe the solution as a linear combination of basis functions chosen from a suitable set. As a result, they enable the analysis of more complex geometries. Spectral methods use orthogonal basis functions, and they can achieve an exponentially fast convergence for smooth problems. Unfortunately, for all classical methods, solving a d -dimensional PDE to accuracy ϵ generally requires $O(\epsilon^{-d})$ grid points, which leads to an exponential scaling with dimension d . This scaling renders many important high-dimensional problems intractable even on the most powerful supercomputers, making systems like the electronic structure of large molecules

or turbulent fluid flow hard to accurately study. Additional issues appear when discretizing differential equations for classical algorithms to handle. The condition number κ of the resulting linear systems often scales poorly with grid size. For example, for second-order elliptic problems, it generally depends on $\kappa = O(h^{-2})$. Moreover, nonlinear differential equations require iterative solution methods, such as Newton-Raphson. Its convergence depends on the quality of the initial guess and the structure of the problem. Since iterative solvers for these systems require $O(\kappa)$ iterations to converge, the computational cost is further increased. These classical limitations could be circumvented by quantum approaches that function through fundamentally different computational paradigms.

The quantum linear systems algorithm (QLSA) [11] set the first steps towards a quantum algorithm for differential equations. Given a sparse $N \times N$ matrix A and vector $\vec{b}$, QLSA prepares a quantum state $|x\rangle$ proportional to the solution $\vec{x} = A^{-1}\vec{b}$ in time $O(\kappa^2 \log(N) \text{poly}(\log(1/\epsilon)))$, where κ denotes the condition number and ϵ the desired accuracy. Since this method depends logarithmically on N , vs the classical approaches that scale with $O(N\kappa)$, it allows for efficiently solving certain high-dimensional linear systems. By discretizing the PDE of interest into such a linear system, quantum computers then provide a new alternative to finding the desired solution.

The original QLSA, then named HHL, algorithm uses established quantum subroutines such as quantum phase estimation and amplitude amplification to extract eigenvalues of the linear system, invert them, and then efficiently extract the solution. Subsequent improvements have reduced its complexity to near the theoretical optimal scaling. Variable-time amplitude amplification [23] improved the κ dependence to $O(\kappa \log(1/\epsilon))$, while [24] achieved $O(\kappa \text{polylog}(\kappa/\epsilon))$ using a linear combination of quantum walk operators. The optimal $O(\kappa \log(1/\epsilon))$ scaling was finally achieved by using a combination of adiabatic quantum computing [25], randomization methods [26], and the quantum singular value transformation [27]. As a result of these improvements, QLSA is considered competitive for well-conditioned systems. Though for some applications, a high condition number is still a limiting factor in its possible use.

The quantum singular value transformation (QSVT) [27] introduced a new way of thinking about quantum algorithms in a unified manner and, in many cases, a way of implementing these algorithms with (near) optimal scaling. QSVT provides a way of taking block-encoded matrices and then applying polynomial transformations to their singular values. By picking the right block-encoded matrix and polynomial transformation, many well-established quantum algorithms, such as QLSA, quantum simulation, and amplitude amplification, can be formulated using this framework. For a matrix A with a singular value decomposition $A = \sum_i \sigma_i |u_i\rangle \langle v_i|$, QSVT implements the transformation $f(A) = \sum_i f(\sigma_i) |u_i\rangle \langle v_i|$ for a polynomial function f , achieving optimal query complexity for many problems and providing a systematic way of developing quantum algorithms for new problems. For differential equations specifically, QSVT enables an optimal implementation of spectral method techniques and time evolution operators.

Taking advantage of these techniques, quantum algorithms for differential equations have been able to address many different problem classes. By using truncated Dyson series and directly implementing time evolution, linear differential equations can be solved [28] with complexity $O(T \text{poly}(\log(T/\epsilon)))$ for evolution time T . Higher-order methods have been developed [29] for sufficiently smooth solutions, using linear

multi-step techniques, achieving $O(T\text{polylog}(T/\epsilon))$. Time-dependent linear PDEs can be tackled with an exponential speedup in the spatial dimensions by using quantum spectral methods [30], that leverage the quantum Fourier transform. This approach yields $O(T\text{poly}(n)\text{polylog}(T/\epsilon))$ for n spatial modes, compared to the classical $O(T\text{poly}(N))$ for $N = 2^n$ grid points. These methods have been extended so that analytic solutions converge exponentially quickly, yielding $O(T\text{poly}(\log(T/\epsilon)))$. Alternatively, time-dependent differential equations can also be solved using time-marching quantum algorithms [31]. These rely on an explicit time-stepping approach, and have more forgiving requirements on the preparation of the initial state, compared to approaches based on spectral methods. It is also applicable to a broader class of problems, since it can handle time-dependent coefficients and nonlinear terms by linearizing them. Its complexity is $O(T^2\text{poly}(\log(T/\epsilon)))$ for general initial conditions and $O(T\text{polylog}(T/\epsilon))$ for smooth conditions in particular.

These algorithms have demonstrated that quantum computers can yield an exponential speedup in the problem dimension while achieving a similar error scaling to the classical approaches. As a result, quantum algorithms could be particularly attractive for tackling high-dimensional PDEs with smooth solutions.

However, nonlinear differential equations present a fundamental challenge for quantum algorithms, since quantum mechanics is inherently linear. The no-cloning theorem prevents copying quantum states directly, which makes it difficult to straightforwardly implement some nonlinear operations. Unperturbed by these limitations, in recent years, several approaches have been presented to tackle nonlinear problems in a quantum setting. Most of these methods circumvent the issue by embedding the nonlinear component within a larger linear system or by decomposing the nonlinear problem into a sequence of linear steps. Depending on the approach, there are trade-offs between the accuracy of the solution, the efficiency of the method, and the class of nonlinearities that are suitable for the approach.

One such technique is Carleman linearization, which provides a systematic method for converting nonlinear ordinary differential equations (ODEs) into infinite dimensional linear systems [32]. Carleman linearization takes a nonlinear ODE, such as $\dot{x} = f(x)$, and builds an infinite hierarchy of equations involving products of variables, resulting in $\dot{x}_i = f_i(x)$, $\dot{x}_i x_j = \dot{x}_i x_j + x_i \dot{x}_j, \dots$. In cases where the higher order terms contribute increasingly less, a truncation of this infinite system to finite order can be a good approximation to the original problem, and the resulting linear system is amenable to QLSA. A quantum Carleman linearization algorithm has been recently developed [33] that scales as $O(T\text{poly}(\log(T/\epsilon)))$ as long as the nonlinearity component is relatively small compared to the dissipation term. This restriction limits the algorithm to weakly nonlinear or strongly dissipative systems. Alternatively, quantum algorithms based on homotopy perturbation methods [34] interpolate between a solvable linear problem and the nonlinear system of interest. They work by iteratively solving a sequence of intermediate problems where the solution at each step is the initial guess for the next. Under certain regularity conditions, the algorithm can achieve an exponentially better error dependence than Carleman linearization, scaling as $O(T\text{poly}(\log(1/\epsilon)))$. Finally, the Koopman operator technique builds upon the evolution of observables rather than working directly with states. For a nonlinear system, it acts on the infinite-dimensional space of observables in the system, in order to simulate nonlinear dynamics. Using Koopman approaches,

quantum algorithms for studying classical dynamical systems have been developed [35] that achieve a polynomial speedup for computing certain quantities of interest, such as Lyapunov exponents. This technique does have drawbacks, as it requires an efficient representation of observables in a quantum state, which may be feasible through polynomial or Fourier basis functions for some problems, but not necessarily in general.

With this thesis, we work towards a novel quantum algorithm for solving certain nonlinear partial differential equations, based on a novel quantum implementation of the inverse scattering transform (IST). The classical IST has been developed for nonlinear integrable systems, and it decomposes the nonlinear evolution of the system into a series of linear steps. The first step, called direct or forward scattering, maps the original problem into its associated scattering data. This is followed by the time evolution of the scattering data, and concluded by mapping the evolved scattering data back into the desired solution using an appropriate inverse scattering method. In this thesis, we focus on the first step, which even classically is already relatively involved. As an example of the procedure, we focus our analysis on the Korteweg-de Vries (KdV) equation, which is one of the most well known nonlinear equations that can be solved with this transform. Our quantum algorithm for this step achieves a query complexity of $O(r \text{ polylog}(r/\epsilon))$, where r is the spatial discretization parameter and ϵ the desired accuracy. This corresponds to a quadratic speedup in r compared to classical approaches such as conjugate gradient descent.

The quantum inverse scattering transform works by noting that, for the class of integrable equations, the scattering data associated with the solution evolves linearly in time, even though the solution itself evolves nonlinearly. Therefore, as long as the forward and inverse scattering steps can be linearized accurately and efficiently, the whole procedure becomes linear, and quantum subroutines can speed up the more computationally intensive spectral transformations (from and to the scattering data) to yield a fast method to retrieve the equation's exact solution (or a very good approximation of it). While our focus for this thesis is on the KdV equation, the method could potentially be expanded to other integrable problems, such as the nonlinear Schrödinger equation, the sine-Gordon equation, and the Toda lattice, making it easier to study soliton dynamics and nonlinear wave phenomena using quantum computers.

As is also the case for other quantum algorithms tackling differential equations, the feasibility of these techniques is largely dependent on the extent to which the quantum states encoding the initial and final data can be efficiently prepared and read out, respectively. For general classical data, the preparation or read out stages may end up negating any speedup achieved in the computational stage. Fortunately, structured data can be efficiently encoded in a quantum register, and many problems can be described with sparse or low-rank representations. For the readout stage, we may be restricted to only computing observables of interest, in order to maintain the quantum advantage. Nonetheless, these methods' quantum state outputs can also be used as inputs for later quantum calculations, or as a subroutine of a larger algorithm. Additionally, the error propagation estimates for these quantum algorithms often assume fault-tolerant computation, which would require applying considerable error correction techniques for larger scale problems, and motivate the development of better quantum error correction methods.

1.3 Quantum Error Correction

Classical error correction can rely on straightforward strategies such as redundancy and majority voting, since bits can only suffer discrete flip errors. Quantum states work differently, since they live in a continuous space where a continuum of possible errors can occur, and the no-cloning theorem [36] prevents creating backup copies that could be used for redundancy-based correction. Moreover, attempting to measure a quantum state to check for errors would generally collapse it, destroying the very information we want to protect. Despite these seemingly insurmountable obstacles, reliable quantum computation is achievable, as long as physical error rates remain below certain thresholds. Quantum error correction is the essential mechanism that makes this possible, and in this section we cover its foundations and practical implementations, focusing on the novel decoding methods we develop in this thesis.

One of the earliest and most important results, due to Shor [37], showed that correcting a discrete set of errors is enough to handle arbitrary quantum errors, even continuous ones. Any single-qubit error can be decomposed as $E = \alpha_0 I + \alpha_1 X + \alpha_2 Y + \alpha_3 Z$, with $\alpha_i \in \mathbb{C}$ and $\{I, X, Y, Z\}$ the Pauli matrices, and correcting all four Pauli errors automatically handles their linear combinations, which effectively digitizes the problem. The *stabilizer formalism* [38] then provides a systematic way of constructing and analyzing quantum codes that takes advantage of this discretization. A stabilizer code encodes k logical qubits into n physical qubits by having $n - k$ independent, commuting Pauli operators $\{S_1, S_2, \dots, S_{n-k}\}$ that generate a stabilizer group, and the code states are those satisfying $S_i |\psi\rangle = |\psi\rangle$ for all generators, thereby forming a protected 2^k -dimensional subspace. Errors are detected by measuring these stabilizer operators. The formalism also makes possible the efficient classical simulation of Clifford circuits, which plays an important role when designing fault-tolerant protocols. Additionally, the quantum error correction conditions [39] provide the necessary and sufficient criteria for a code to be able to correct a given set of errors. For a code with a projector P onto the code space, errors $\{E_i\}$ are correctable if and only if $PE_i^\dagger E_j P = \alpha_{ij} P$ for some Hermitian matrix α , meaning that different errors must map code states to distinguishable subspaces, so that unambiguous correction is possible. These conditions generalize to approximate error correction and are used to design codes tailored to specific and realistic noise models.

Among the various code families, surface codes have emerged as one of the most promising candidates for near-term experimental implementation [40], since they only require local stabilizer measurements between neighboring qubits and have comparatively high threshold error rates. Qubits sit on a two-dimensional lattice with X -type and Z -type stabilizers alternating on plaquettes and vertices, and a lattice of size $L \times L$ encodes one logical qubit with distance $d = L$, meaning up to $\lfloor (d - 1)/2 \rfloor$ errors can be corrected. Encouragingly, recent experiments have already demonstrated error suppression with small surface codes [41]. One of the most established decoding techniques for surface codes is minimum weight perfect matching (MWPM), which works by building a graph where vertices correspond to syndrome defects and edge weights reflect error probabilities, and then finding the minimum weight matching to identify the most likely error configuration. This achieves near-optimal decoding performance with $O(n^3)$ complexity.

Quantum random linear codes (QRLCs) are the quantum analogue of classical random codes, and they achieve optimal error correction performance in the asymptotic limit. Constructing one only requires picking a random $n \times k$ matrix G over $\mathbb{F}_2$ and defining the associated encoding, and with high probability the resulting code achieves distance $d \approx nH^{-1}(1 - k/n)$ for binary entropy function H , which is sufficient to saturate quantum channel capacity for general Pauli channels. QRLCs have been impractical historically because their lack of algebraic structure prevented the development of efficient decoders.

We present quantum guessing random additive noise decoding (QGRAND), which to our knowledge is the first practical maximum-likelihood decoder for QRLCs. Rather than relying on code structure, QGRAND exploits noise statistics. Given a syndrome s , it generates candidate error patterns in order of decreasing probability and checks each against the observed syndrome, with the first consistent match being the maximum-likelihood estimate. A distinctive feature we incorporate into QGRAND is *error degeneracy*, a uniquely quantum phenomenon. Two errors E_1 and E_2 are degenerate whenever they differ only by a stabilizer element ($E_1 = E_2 S$ for some $S \in \mathcal{S}$), which means that physically distinct errors can have similar effects on the code space. By grouping degenerate errors and tracking only one representative per equivalence class, QGRAND significantly reduces its search space under generic noise conditions. The number of non-degenerate errors scales as $O(n)$ rather than the naive $O(n^2)$ for QRLCs, which keeps the decoder efficient even as the code size grows.

However, for quantum error correction to be useful in practice, the correction procedure itself must not introduce too many new errors. Achieving this is non-trivial, since syndrome extraction circuits, which measure stabilizer generators without disturbing the encoded information, must be carefully designed so that faults do not cascade through the circuit, as even a single fault in a poorly designed extraction can propagate into an uncorrectable error. Techniques such as flag qubits, redundant measurements, and verified ancilla preparation have been developed to contain this kind of propagation. We extend QGRAND to this fault-tolerant setting by modeling the full error process during syndrome extraction, accounting for preparation faults, gate errors, measurement errors on ancilla qubits, and propagation through CNOT gates. With this comprehensive noise model, fault-tolerant QGRAND achieves a threshold of approximately $p_{\text{th}} \approx 5.5 \times 10^{-3}$ for QRLCs, despite these codes having no exploitable algebraic structure. Circuit design is also important. QRLCs have dense stabilizer generators with an average weight of $3n/4$, which means that their syndrome extraction circuits require $O(n^2)$ CNOT gates that can each propagate errors. However, exploiting gauge freedom in the stabilizer formalism reduces the average weight to $(2n + k + 2)/4$ and notably lowers both the circuit complexity and the level of error propagation. Our decoder adapts naturally to these optimized circuits by updating its error probability calculations accordingly.

Underlying all of these error correction results is the threshold theorem [42], which guarantees that arbitrarily long quantum computations become feasible when physical error rates stay below a critical value. Its proof relies on concatenated codes, where logical qubits are recursively encoded to suppress errors at a double-exponential rate: for a base code correcting t errors at physical error rate $p < p_{\text{th}}$, k levels of concatenation reduce the logical error rate to $O((p/p_{\text{th}})^{(t+1)^k})$. This makes reliable quantum

computation possible with only polynomial overhead in circuit size and depth. Recent work on flag fault-tolerance and single-shot error correction has further improved these threshold estimates while reducing the associated resource requirements.

Besides surface codes, other topological code families offer different trade-offs for quantum error correction. Color codes [43] allow the transversal implementation of Clifford gates, while fracton codes [44] achieve a better ratio of logical to physical qubits, and in all of these codes, local errors create localized excitations that can be tracked and corrected, while logical errors correspond to non-trivial topological operations spanning the entire system. On the experimental side, significant milestones have been reached in recent years. Repetition codes have been used for error detection [45], small surface codes for logical qubit encoding [46], and increasing code sizes for demonstrating active error suppression [41]. While these results validate theoretical predictions, they also expose practical challenges absent from idealized models, such as correlated errors, leakage to non-computational states, and frequency crowding in multi-qubit devices. Narrowing this divide between theory and experiment continues to drive the development of better error correction methods, including those we present in this work.

With this thesis, we contribute to quantum computing along two directions. On one hand, the quantum inverse scattering transform shows that certain nonlinear differential equations, which have long been considered beyond the reach of efficient quantum approaches, can be tackled by decomposing their evolution into a sequence of linear steps. On the other hand, QGRAND and its fault-tolerant extension demonstrate that QRLCs, despite their lack of algebraic structure, can be practically decoded under realistic noise conditions. The threshold of approximately 5.5×10^{-3} that we achieve confirms these theoretically optimal codes as a viable option for fault-tolerant quantum computing.

The techniques developed in this work should become increasingly relevant for problems of practical importance in the field, as quantum hardware continues to mature. We aim to bridge the divide between theory of quantum algorithms and their implementation on physical devices, by deliberately incorporating current hardware limitations into our models.

Chapter 2

Quantum Forward Scattering Transform for Nonlinear Differential Equations

2.1 Introduction

Differential equations play a fundamental role in various scientific fields, such as fluid mechanics, plasma physics, and quantum physics. The Schrödinger equation is central to quantum physics and was one of the initial motivations for developing a quantum computer. Hamiltonian simulation, done by simulating the Schrödinger equation, can be efficiently performed for several systems, and a range of important techniques has been developed in this area [47, 48, 49, 50, 51]. The quantum linear systems algorithm (QLSA) [11], which solves linear systems of equations, has been successfully applied to various problems, including differential equations and machine learning [29, 52, 53, 54, 33, 55]. Over time, improvements have been made to the run-time of the algorithm [56, 57], including scaling with the condition number κ and the solution error ϵ , with the state-of-the-art method currently scaling with $\kappa \log(1/\epsilon)$ [58, 59, 60, 61].

Recent work has explored the use of quantum algorithms for solving differential equations. Nonlinear differential equations were first studied in [62], and subsequently, a quantum algorithm for solving linear inhomogeneous equations using high-order methods was presented in [29]. In [52], a quantum algorithm was developed for solving linear differential equations using truncated Taylor series. These results were further improved in [63]. In [54], a quantum algorithm for solving time-dependent linear differential equations was introduced using spectral methods, and in [53], these methods were applied to develop a quantum algorithm for solving linear partial differential equations. In [64], time-marching techniques were shown to enable solving similar problems under less restrictive setups. Recent advances in quantum linear ODE solvers continue to improve efficiency, including non-asymptotic cost computations [65], logarithmic

error dependence via Dyson series [66], and Schrödingerization techniques for converting PDEs to unitary formats [67].

In [33], a quantum algorithm to solve dissipative nonlinear differential equations is given using the so-called Carleman linearization. This algorithm is efficient when the ratio of nonlinearity to dissipation is small. Its error dependence is exponentially improved in [68], using homotopic perturbation methods. Recent work has extended Carleman linearization beyond dissipative regimes through resonance conditions that encompass the Korteweg-de Vries equation [69], efficiency improvements via higher-order methods [70], and applications to fluid dynamics [71]. However, fundamental limitations exist for chaotic systems with positive Lyapunov exponents, where complexity scales exponentially in time [72].

Alternative linearization strategies include the Koopman operator framework, which transforms nonlinear dynamics into linear unitary evolution in higher-dimensional spaces [73], and Koopman-von Neumann linearization for specific classes of nonlinear ODEs [74].

In [55], a method for nonlinear differential equations is given for short simulation times. Despite this recent progress, there is still no efficient quantum algorithm to solve general nonlinear PDEs whenever the nonlinearity is not small.

The Inverse Scattering Transform (IST) is a mathematical technique that was originally developed to solve the Korteweg-de Vries equation [75, 76, 77], and was later extended to other nonlinear partial differential equations, including the nonlinear Schrödinger equation [78, 79] and the Sine-Gordon equation [80]. In general, this technique may be applied to a wide range of integrable evolution equations [81, 82, 83, 84, 85].

IST enables some nonlinear PDEs to be mapped to a system of linear PDEs, by working with the scattering data. Unlike the approximate linearization techniques of Carleman and Koopman methods, this mapping is exact for integrable systems, and yields the exact solution $u(x, t)$ to the original nonlinear PDE. Suppose we have some operators L and P , called Lax Pairs [86], such that, for some function $\psi(x, t)$,

$$L\psi = \lambda^2\psi, \tag{2.1}$$

$$P\psi = \partial_t\psi, \tag{2.2}$$

$$\implies \left[\frac{dL}{dt} + (LP - PL) \right] \psi = \frac{d\lambda^2}{dt} \psi. \tag{2.3}$$

Requiring $d\lambda^2/dt = 0$ leads to

$$\frac{dL}{dt} = [P, L] \implies \text{nonlinear PDE.} \tag{2.4}$$

In this chapter, we present a quantum algorithm for the forward scattering transform, a fundamental spectral analysis technique that computes scattering data from initial conditions by solving the spatial eigenvalue problem $L(u_0)\psi = \lambda^2\psi$. This forward scattering step is valuable in its own right and has broad applicability beyond any single application.

Notably, the forward scattering transform applies to entire hierarchies of nonlinear equations. For instance, the KdV hierarchy consists of infinitely many nonlinear PDEs that all share the same spatial Lax operator L but differ in their time evolution operators P . The forward scattering data computed from L is thus relevant to all equations in this hierarchy, not merely the KdV equation itself. This universality makes the forward scattering transform a powerful tool for analyzing broad classes of integrable systems.

The scattering data comprises two components: the continuous spectrum, characterized by reflection coefficients $R(\lambda)$ for $\lambda \in \mathbb{R}$, and the discrete spectrum, characterized by bound states. The reflection coefficients describe how plane waves scatter off the potential, while bound states correspond to poles of the transmission coefficient in the complex plane and represent localized, exponentially decaying solutions (solitons). Together, these components encode essential spectral information about the operator $L(u_0)$.

One important application of forward scattering is as the first step of the complete Inverse Scattering Transform (IST) method for solving certain nonlinear PDEs. The IST method proceeds in three stages: (1) forward scattering computes the scattering data from initial conditions; (2) time evolution advances the scattering data using the P operator; and (3) inverse scattering reconstructs the solution $u(x, t)$ from the evolved scattering data via the Gelfand-Levitan-Marchenko equation. However, even independently of the complete IST method, the forward scattering data provides valuable spectral information with applications in signal processing, soliton theory, and the analysis of integrable systems.

Our quantum algorithm efficiently computes the continuous spectrum component (reflection coefficients) of the forward scattering data, achieving quadratic speedup compared to classical methods. The algorithm's formulation as a linear eigenvalue problem makes it particularly amenable to quantum computation and demonstrates concrete quantum advantage for this fundamental spectral analysis task.

Theorem 1 (Main Result—Informal Version). *Consider the forward scattering problem for initial conditions $u_0(x)$ that are either (i) analytic in a strip around the real axis with exponential decay, or (ii) have derivatives that decay faster than $1/|x|$.*

Our quantum algorithm produces a quantum state

$$|R\rangle = \frac{1}{\mathcal{N}} \sum_{\lambda \in \Lambda} R(\lambda) |\lambda\rangle \quad (2.5)$$

that encodes the reflection coefficients $R(\lambda)$ for a discrete set Λ of wavenumbers, with error ϵ in ℓ^2 norm. The algorithm achieves quadratic speedup in the spatial discretization parameter r compared to classical methods such as conjugate gradient descent, with query complexity

$$O(r \text{polylog}(r/\epsilon)). \quad (2.6)$$

The formal complexity statement with precise definitions of all parameters appears in Section H as Theorem 2. The conditions on u_0 that guarantee all technical assumptions are satisfied are established in Section E (analyticity, Corollary 1) and Section F (derivative decay, Theorem 4).

2.2 Forward scattering

We first need to solve Eq. (2.1) for $t = 0$ and $x \rightarrow \pm\infty$, or equivalently, to find the associated scattering data (see Fig. 2.1). In this work, we focus specifically on the computation of the continuous spectrum (reflection coefficients). We wish to perform this forward scattering process for a representative sample of λ values.

As a concrete example for our forward scattering algorithm, we consider the Korteweg-de Vries (KdV) equation,

$$\partial_t u - 6u\partial_x u + \partial_x^3 u = 0, \quad (2.7)$$

a widely studied nonlinear PDE. It describes the evolution of long, one-dimensional waves in many physical settings where solitary waves (solitons) are present (see Ref. [87]), like ion acoustic waves in a plasma. In this case, the Lax Pairs are

$$L = -\partial_x^2 + u \quad (2.8)$$

$$P = 4\partial_x^3 - 6u\partial_x - 3\partial_x u. \quad (2.9)$$

See Appendix A for a possible derivation of these Lax Pairs. In order to elucidate our method, we will consider the initial condition $u_0(x) := -4\operatorname{sech}(x)^2$ throughout, and demonstrate how to find the forward scattering data.

Using Eq. (2.8) for the KdV equation, Eq. (2.1) becomes the time-independent Schrödinger equation,

$$\partial_x^2 \psi(x; \lambda) = (u_0(x) - \lambda^2) \psi(x; \lambda), \quad (2.10)$$

where $u_0(x)$ is the initial condition for the KdV equation. The exact scattering data that we need to extract depends on the particular method used for the Inverse Scattering step. Here, we employ the method of Faddeev-Marchenko [88, 89]. For the Jost solutions of the Schrödinger equation (i.e., $\lambda \in \mathbb{R}$), we require the reflection coefficient $R(\lambda)$.

In this work, we solve the Schrödinger equation using the method of Ref. [54], adapted to our use case. Subsequent work has improved on its results [63, 64] to yield better methods to solve time-dependent ODEs. Nonetheless, in any of these methods, the behavior of the matrix $\mathbf{A}$, and in particular its norm, plays a major role in the query and gate complexities of the resulting methods. Therefore, methods that turn equations such as Eq. (2.10) into numerically stabler problems would always be of interest, irrespective of the particular technique used.

In Ref. [54], the ODE considered is time-dependent. Starting with the initial conditions at $t_i = 0$ the ODE is used to evolve the solution to the final time t_f . For our use case, we wish to start with the initial condition at $x = x_i$ and use the Schrödinger equation to evolve it in *space* to $x = x_f$. In the context of forward scattering, we do not have a time component, and we treat the spatial domain as the time domain would be treated in Ref. [54], referring to the initial conditions of Eq. (2.10) as those of the

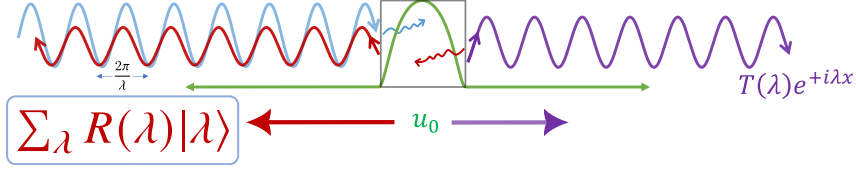


Figure 2.1: Schematic of the quantized version of forward scattering: using the Liu-Childs method, the reflection coefficients from the interaction between plane-waves with wavelengths $2\pi/\lambda$ and the potential u_0 for the operator $L(0)$ are put into probability amplitudes of a superposition of quantum states corresponding to a discrete sampling of wavenumber λ values. The left-hand side shows the reflection coefficient $R(\lambda)$ times the incoming plane wave, while the right-hand side shows the transmission coefficient $T(\lambda)$ times the transmitted plane wave. The procedure yields a quantum state with the scattering data (reflection coefficients) in superposition.

spatial domain.

Directly solving the Schrödinger equation (Eq. (2.10)) numerically is challenging due to the highly oscillatory nature of the solutions $\psi(x; \lambda)$, particularly for large λ . This oscillation leads to numerical stiffness and requires fine discretization. To address this and create a system amenable to efficient quantum simulation, we reformulate the equation to solve for the slowly varying envelope of the solution rather than the oscillating wave function itself.

We start by rewriting the Schrödinger equation in a manner that makes it more efficient to solve, for $x \rightarrow \pm\infty$.

Lemma 1. *Defining*

$$k(x; \lambda) := \sqrt{\lambda^2 - u_0(x)} \quad (2.11)$$

$$\psi(x) =: c^+(x; \lambda) \cos(kx) + c^-(x; \lambda) f(\lambda) \frac{\sin(kx)}{k}, \quad (2.12)$$

$$\mathbf{c}(x; \lambda) := \begin{pmatrix} c^+ \\ c^- \end{pmatrix} \quad (2.13)$$

$$\mathbf{A}_c(x; \lambda) := \frac{\partial k}{\partial x} \begin{pmatrix} -\frac{\sin(kx)^2}{k} & \frac{-2kx + \sin(2kx)}{2k^2} f(\lambda) \\ \frac{2kx + \sin(2kx)}{2f(\lambda)} & \frac{\sin(kx)^2}{k} \end{pmatrix} \quad (2.14)$$

for some chosen function $f(\lambda)$, Eq. (2.10) can be rewritten as

$$\partial_x \mathbf{c}(x; \lambda) = \mathbf{A}_c(x; \lambda) \mathbf{c}(x; \lambda). \quad (2.15)$$

Moreover, $\mathbf{A}_c$ satisfies the assumptions required in Ref. [54] to solve Eq. (2.15) efficiently, with $\|\mathbf{A}_c\|$ being bounded and $\text{Re}(\text{eigenvalues of } \mathbf{A}_c) \leq 0$.

See Appendix B for more details. In practice, we use $f(\lambda) = \max\{1, \lambda\}$.

Although we wish to decompose ψ as in Eq. (2.12) in order to solve the Schrödinger equation (Eq. (2.10)), extracting the scattering data requires us to write ψ as

$$\psi(x; \lambda) = a^+(x; \lambda) e^{+i\lambda x} + a^-(x; \lambda) e^{-i\lambda x}, \quad (2.16)$$

since the $a^\pm(x \rightarrow \pm\infty; \lambda)$ coefficients will correspond to the sought-after scattering coefficients. We may impose initial conditions on the Schrödinger equation by choosing the values of these $\mathbf{a} := \begin{pmatrix} a^+ & a^- \end{pmatrix}^T$ coefficients. We have $\mathbf{a} = H\mathbf{c}$, with

$$H(x; \lambda) = \frac{1}{2} \begin{pmatrix} 1 & \frac{-if(\lambda)}{k} \\ 1 & \frac{if(\lambda)}{k} \end{pmatrix}. \quad (2.17)$$

If $u_0(x)$ and the coefficients $\mathbf{c}$ are slowly decaying as $x \rightarrow \pm\infty$, then we may apply some of the techniques in Ref. [90] to map the non-negligible portion of $u_0(x)$ to a smaller part of the x domain. As a result, we ensure that we work with a formulation of the problem where u_0 decays quickly enough for the spatial domain to be relatively small, and for the cutoff error to decay exponentially quickly, as required by Lemma 10.

Consider x_i, x_f the start and end points used, with $X := |x_f - x_i|$. Using the results from Ref. [54] and the stabilization formulation above, we obtain a quantum algorithm with query complexity $\mathcal{O}(rq \text{polylog}(rg'/\epsilon g))$ to produce the quantum state $|R\rangle \propto \sum_\lambda R(\lambda) |\lambda\rangle$ encoding the reflection coefficients. The formal complexity statement with precise definitions of all parameters (r, q, g, g') appears in Lemma 10 (Section H). See also Section B for implementation details.

Having established Lemmas 1 and 10, we can build a quantum algorithm for forward scattering. Using this method, the state $\sum_\lambda R(\lambda) |\lambda\rangle$ encodes the reflection coefficient $R(\lambda)$ for the desired λ values. See Fig. 2.2 for an example.

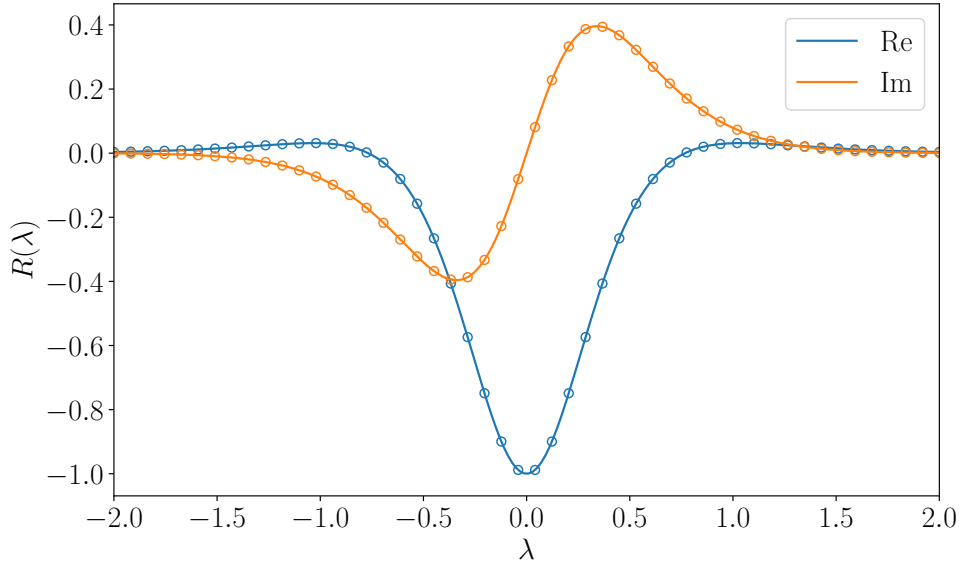


Figure 2.2: $R(\lambda)$ for $u_0(x) = -4\text{sech}(x)^2$, as computed analytically (solid lines), which can be done exceptionally for this case, and numerically (circles) with the quantum forward scattering technique.

2.2.1 Complexity analysis

For the Jost solutions, the query complexity is

$$\mathcal{O}_{JS}^Q = \mathcal{O}(rq \text{polylog}(r/\epsilon)). \quad (2.18)$$

The gate complexity has an additional multiplicative $\mathcal{O}(\text{polylog}(N_\lambda))$ factor arising from the theorems in Ref. [57], used by Ref. [54].

Since this approach formulates and solves the Schrödinger equation as a linear matrix equation via a spectral discretization, we first estimate the complexity scaling of solving this specific matrix equation using the classical Conjugate Gradient (CG) method. For a s -sparse matrix with condition number κ , its scaling is $\mathcal{O}(Ns\kappa \log(1/\epsilon))$ (see Ref. [91]), assuming we wish to extract the full vector from the solution, which we do, since we need to compute non-trivial steps on the solution later on. For the Liu-Childs method, $s = \mathcal{O}(ns_A)$, $\kappa_L = \mathcal{O}(rn^{3.5})$ and $N_L = rdn$ (where r comes from Lemma 10). We consider $n \sim \mathcal{O}(\text{polylog}(r/\epsilon))$. Therefore, the scalings for solving the stabilized linear system for the Jost solutions are

$$\text{Classical (CG on Spectral Discretization): } \mathcal{O}\left(N_\lambda r^2 ds_A \text{polylog}\left(r, \frac{1}{\epsilon}\right)\right), \quad (2.19)$$

$$\text{Quantum (Gate Complexity): } \mathcal{O}\left(rs_A q \text{polylog}\left(r, d, N_\lambda, \frac{1}{\epsilon}\right)\right). \quad (2.20)$$

The quantum algorithm exhibits an exponential speedup in d and N_λ , and a quadratic speedup in r compared to the CG benchmark for this algebraic problem. For the KdV equation, $d = 2 = \mathcal{O}(1)$ and $N_\lambda = \text{polylog}(1/\epsilon)$, so the quadratic speedup in r is the most apparent advantage in this specific comparison.

We may contextualize this result within the landscape of specialized classical algorithms for the FST. The state-of-the-art classical approaches are Fast Nonlinear Fourier Transform (FNFT) methods, which often achieve near-linear scaling (e.g., $\mathcal{O}(r \log^2 r)$) for the 1D KdV equation. When compared to FNFT, the quantum algorithm does not offer a polynomial advantage in the spatial parameter r for 1D KdV. However, the advantages stemming from the exponential speedup in d (for higher-dimensional integrable systems) and the efficient parallelization over N_λ remain significant.

We have an additional q dependence, so handling u_0 cases that are (close to) reflectionless (i.e., $R(\lambda) = 0$) requires some care.

2.3 Discussion

In this work, we have presented a quantum algorithm for the forward scattering transform, a critical component of the Inverse Scattering Transform method for solving nonlinear partial differential equations that can be expressed in terms of Lax Pairs, such as the Korteweg-de Vries equation.

We have shown how, under reasonable assumptions, we can efficiently compute the scattering data as-

sociated with the nonlinear equation and initial condition u_0 . We observed that the complexity scaling of our forward scattering algorithm quantitatively mimics the scaling of QLSA [54], from which it inherits its speedup. For its use to be viable, we have shown how to reformulate the forward scattering problem into an equivalent matrix problem that satisfies the assumptions of Ref. [54]. We note that this mapping found is highly non-trivial, and not expected to exist *a priori*, given that the two formulations are qualitatively distinct. Additionally, we note that certain assumptions from Ref. [54] can be relaxed, as demonstrated in the appendices and similarly indicated in Ref. [63], and this relaxation considerably extends the scope of our approach.

The stabilized formulation in Lemma 1 transforms the oscillatory Schrödinger equation into a numerically stable ODE system, crucial for both quantum and classical implementations. Our algorithm efficiently extracts the reflection coefficients for the continuous spectrum, encoding the scattering data in quantum superposition. Beyond the quantum speedup, our stabilized formulation yields a classical algorithm with improved scaling compared to traditional methods based on finite differences or the Fast Nonlinear Fourier Transform.

Classical algorithms for the forward scattering transform have been extensively developed. Traditional transfer-matrix methods such as the Boffetta–Osborne algorithm [92] achieve $O(N^2)$ complexity, where N is the number of spatial discretization points. The Fast Nonlinear Fourier Transform (FNFT) introduced by Wahls and Poor [93] reduces this to $O(N \log^2 N)$ for the defocusing nonlinear Schrödinger equation, with subsequent work extending these techniques to other integrable systems [94]. For linear systems arising from the discretized scattering problem, conjugate gradient methods [95] provide $O(Ns\kappa \log(1/\epsilon))$ complexity, where s is the sparsity, κ the condition number, and ϵ the target accuracy. Our quantum algorithm achieves $O(r \text{ polylog}(r/\epsilon))$ query complexity, providing quadratic speedup over these classical approaches in the spatial discretization parameter.

Overall, we highlight that, for the forward scattering step, we have presented a description of our method with generalizability in mind. This does not preclude further complexity improvements obtainable in particular use cases, where it may be possible to further optimize the different substeps. For example, scenarios with no solitons can use an optimized approach, as the quantum state does not need to encode the non-existent information about the norming constants.

The quantum algorithm achieves quadratic speedup in the spatial discretization parameter r compared to classical conjugate gradient methods. For higher-dimensional problems where $d > 2$, an exponential speedup in d becomes apparent, making the approach particularly attractive for vector nonlinear Schrödinger equations and coupled systems. Since, for the KdV equation, $d = 2$, this improvement goes unnoticed. The most practically significant extension of this work lies in applying these techniques to higher-dimensional scattering problems. In higher dimensions, the scattering geometry becomes fundamentally richer: instead of just two scattering directions (left and right) as in one dimension, there are infinitely many scattering directions parameterized by angles on a sphere. While the mathematical structure differs substantially from the one-dimensional case, the stabilization techniques and quantum algorithmic approaches developed here may translate to these more complex settings. Higher-dimensional

scattering has broad practical applications in quantum mechanics, electromagnetic wave propagation, and acoustic scattering, making quantum algorithms for these problems particularly valuable.

The techniques developed here may also extend to other one-dimensional integrable systems beyond the KdV hierarchy. The nonlinear Schrödinger equation, Sine-Gordon equation, modified KdV equation, and Toda lattice all admit IST solutions, and similar stabilization approaches may yield quantum algorithms for their forward scattering transforms. For these equations, formulations analogous to Lemma 1 may enable application of quantum spectral methods with noteworthy speedups.

2.4 Conclusion

This work establishes forward scattering as a concrete example where quantum algorithms can address components of nonlinear PDE solvers through exact decomposition into linear subproblems. We have demonstrated a quantum algorithm that achieves quadratic speedup in the spatial discretization parameter for computing the continuous spectrum of the forward scattering transform. The stabilized formulation we introduce improves numerical stability for both quantum and classical implementations.

Our work addresses the continuous spectrum (reflection coefficients), but a complete IST implementation would additionally require computing the discrete spectrum (transmission coefficient poles corresponding to bound states/solitons). Extending the quantum algorithm to efficiently extract these bound state parameters would enable the complete forward scattering step. Beyond this, the remaining IST components include: (1) time evolution of scattering data using the P operator, which we expect to be straightforward through application of diagonal matrices encoded in unitary operations; (2) inverse scattering via the Marchenko equation, which presents challenges due to the scarcity of quantum methods for integral equations and would likely require decomposition into sparse matrix form solvable by QLSA with additional QSVT overhead; and (3) efficient extraction of the solution $u(x, t)$ from quantum amplitudes, which may require measurement of specific observables or post-processing depending on the desired output.

Our results open pathways for quantum advantage in scientific computing applications involving integrable nonlinear systems.

Appendices

A Determining Lax Pairs

We wish to find the Lax Pairs L and P [86] such that the Lax equation

$$\frac{dL}{dt} = [P, L] \tag{2.21}$$

equals the Korteweg-de Vries (KdV) equation

$$u_t - 6uu_x + u_{xxx} = 0. \tag{2.22}$$

A common starting ansatz is

$$L(t) = -\partial_x^2 + u(x, t), \quad (2.23)$$

which leads to $\frac{dL}{dt} = \partial_t u$. We now look for an operator P such that $[P, L] = 6uu_x - u_{xxx}$. We consider a more general ansatz including any relevant polynomial term, of the form

$$P = au\partial_x + bu_x + c\partial_x^3. \quad (2.24)$$

We exclude any other additional polynomial terms, as they cannot contribute to the final equation. For example, a term proportional to u in P would lead, at most, to terms proportional to u^2 and u_{xx} , which we know do not exist in the KdV equation, so we do not include that term in the ansatz. Similarly, any term involving derivatives of higher than third order should not be present.

We obtain

$$u_t\psi = \frac{dL}{dt}\psi \quad (2.25)$$

$$[P, L]\psi = P[-\psi_{xx} + u\psi] - L[au\psi_x + bu_x\psi + c\psi_{xxx}] \quad (2.26)$$

$$= auu_x\psi + (a + 2b + 3c)u_{xx}\psi_x + (2a + 3c)u_x\psi_{xx} + (b + c)u_{xxx}\psi \quad (2.27)$$

The only solution that matches the KdV equation is $a = -6, b = -3$, and $c = 4$. Consequently, we have determined the operator P , and we have the Lax Pairs.

B Scattering data from a linear time-dependent ODE

The Forward Scattering step of the inverse scattering method was described in Section 2.2 of the main text. We now present a quantum algorithm to recover this data. Our approach is to cast this recovery problem as a linear ODE initial value problem, to which we can apply the method of Ref. [54] and we begin by seeing the reformulation in the following,

Some figures in this appendix use $u_0(x) = -2\text{sech}(x)^2$ (reflectionless) instead of $-4\text{sech}(x)^2$ to better illustrate the numerical formulation.

Lemma 2. *Consider the time-independent Schrödinger equation*

$$\partial_x^2\psi(x; \lambda) = (u_0(x) - \lambda^2)\psi(x; \lambda). \quad (2.28)$$

Define

$$k(x; \lambda) := \sqrt{\lambda^2 - u_0(x)}, \quad (2.29)$$

$$\psi(x; \lambda) := c^+(x; \lambda)\cos(kx) + c^-(x; \lambda)f(\lambda)\frac{\sin(kx)}{k}, \quad \text{and} \quad (2.30)$$

$$\mathbf{A}_c(x; \lambda) := \frac{\partial k}{\partial x} \begin{pmatrix} -\frac{\sin(kx)^2}{k} & \frac{-2kx + \sin(2kx)}{2k^2} f(\lambda) \\ \frac{2kx + \sin(2kx)}{2f(\lambda)} & \frac{\sin(kx)^2}{k} \end{pmatrix}, \quad (2.31)$$

for some chosen function $f(\lambda)$. Then, the solution $\hat{\mathbf{c}}(x; \lambda)$ to the initial value problem

$$\partial_x \hat{\mathbf{c}}(x; \lambda) = \mathbf{A}_c(x; \lambda) \hat{\mathbf{c}}(x; \lambda), \quad (2.32)$$

with initial condition $\hat{\mathbf{c}}(x_i; \lambda) = \begin{pmatrix} c^+(x_i; \lambda) & c^-(x_i; \lambda) \end{pmatrix}^T$ at a fixed $x_i \in \mathbb{R}$, gives exactly the coefficients

$$\hat{\mathbf{c}}(x; \lambda) = \begin{pmatrix} c^+(x; \lambda) \\ c^-(x; \lambda) \end{pmatrix} \quad (2.33)$$

to $\psi(x; \lambda)$ for all $x \in \mathbb{R}$.

Proof. The solutions of the Schrödinger equation can be written as

$$\psi(x; \lambda) = c^+(\lambda) b_+(x; \lambda) + c^-(\lambda) b_-(x; \lambda), \quad (2.34)$$

where b_+, b_- are a pair of linearly independent basis functions, each a viable solution. For our use case, we choose

$$b_+(x; \lambda) = \cos(k(x)x) \quad (2.35)$$

$$b_-(x; \lambda) = \begin{cases} f(\lambda) \frac{\sin(k(x)x)}{k(x)}, & k(x; \lambda) \neq 0, \\ f(\lambda)x, & k(x; \lambda) = 0. \end{cases} \quad (2.36)$$

with $k(x; \lambda) = \sqrt{\lambda^2 - u_0(x)}$. This choice of basis functions has the advantage of being continuous for all $\lambda \in \mathbb{C}$. Note that we consider the square root extension to complex values, identifying the square root with the root with smallest argument.

Consider a step function approximation $\tilde{u}_0(x)$ of $u_0(x)$, with step size of length $2h$, where u_0 is sampled at the midpoint of each step. Let $\tilde{\psi}(x)$ be the resulting solution of the Schrödinger equation when using $\tilde{u}_0(x)$ for the potential.

Let y be some point where $\tilde{u}_0(y)$ is discontinuous. Since $\tilde{\psi}(x)$ and $\tilde{\psi}'(x)$ (i.e., its derivative) must be continuous at $x = y$, we must have

$$\begin{pmatrix} \tilde{\psi}(y^\pm; \lambda) \\ \tilde{\psi}'(y^\pm; \lambda) \end{pmatrix} = \begin{pmatrix} \cos(\tilde{k}(y^\pm; \lambda)y) & f(\lambda) \frac{\sin(\tilde{k}(y^\pm; \lambda)y)}{\tilde{k}(y^\pm; \lambda)} \\ -\tilde{k}(y^\pm; \lambda) \sin(\tilde{k}(y^\pm; \lambda)y) & f(\lambda) \cos(\tilde{k}(y^\pm; \lambda)y) \end{pmatrix} \begin{pmatrix} c^+(y^\pm; \lambda) \\ c^-(y^\pm; \lambda) \end{pmatrix} \quad (2.37)$$

$$= \begin{pmatrix} \cos(k(y \pm h; \lambda)y) & f(\lambda) \frac{\sin(k(y \pm h; \lambda)y)}{k(y \pm h; \lambda)} \\ -k(y \pm h; \lambda) \sin(k(y \pm h; \lambda)y) & f(\lambda) \cos(k(y \pm h; \lambda)y) \end{pmatrix} \begin{pmatrix} c^+(y^\pm; \lambda) \\ c^-(y^\pm; \lambda) \end{pmatrix} \quad (2.38)$$

$$=: B_h^\pm(y; \lambda) \begin{pmatrix} c^+(y^\pm; \lambda) \\ c^-(y^\pm; \lambda) \end{pmatrix}. \quad (2.39)$$

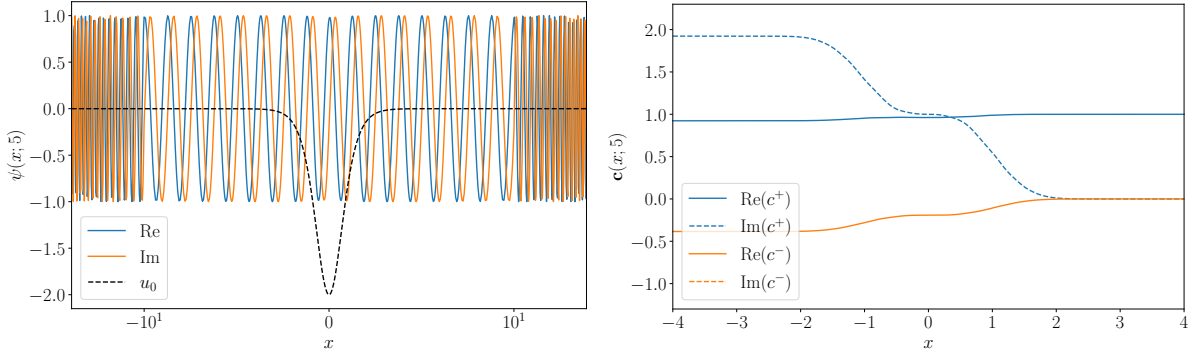


Figure 2.3: Solving the Schrödinger equation directly (left) vs finding the $\mathbf{c}$ coefficients (right) following the formulation in Lemma 2, with $u_0(x) = -2\text{sech}(x)^2$ and $\lambda = 5$. The change of variables approach is also used. We observe that the $\mathbf{c}$ are not only more numerically stable, but they also converge to a concrete value as $|x| \rightarrow \infty$, while ψ maintains its oscillatory behavior.

Since $\tilde{\psi}(x)$ and $\tilde{\psi}'(x)$ must be continuous at $x = y$, we must have

$$\begin{pmatrix} \tilde{\psi}(y^+; \lambda) \\ \tilde{\psi}'(y^+; \lambda) \end{pmatrix} = \begin{pmatrix} \tilde{\psi}(y^-; \lambda) \\ \tilde{\psi}'(y^-; \lambda) \end{pmatrix} \quad (2.40)$$

$$B_h^+(y; \lambda) \begin{pmatrix} c^+(y^+; \lambda) \\ c^-(y^+; \lambda) \end{pmatrix} = B_h^-(y; \lambda) \begin{pmatrix} c^+(y^-; \lambda) \\ c^-(y^-; \lambda) \end{pmatrix}. \quad (2.41)$$

From here, we conclude that

$$\frac{1}{2h} \left[\begin{pmatrix} c^+(y^+; \lambda) \\ c^-(y^+; \lambda) \end{pmatrix} - \begin{pmatrix} c^+(y^-; \lambda) \\ c^-(y^-; \lambda) \end{pmatrix} \right] = \frac{1}{2h} [B_h^+(y; \lambda)^{-1} B_h^-(y; \lambda) - I] \begin{pmatrix} c^+(y^-; \lambda) \\ c^-(y^-; \lambda) \end{pmatrix}. \quad (2.42)$$

with I the identity matrix. In the limit when $h \rightarrow 0$, we get

$$\partial_x \mathbf{c}(y; \lambda) = \mathbf{A}_c(x; \lambda) \mathbf{c}(y; \lambda), \quad (2.43)$$

$$\text{with } \mathbf{A}_c(x; \lambda) := \lim_{h \rightarrow 0} \frac{1}{2h} [B_h^+(y; \lambda)^{-1} B_h^-(y; \lambda) - I] \quad (2.44)$$

$$\text{and } \frac{\partial k(x; \lambda)}{\partial x} \Big|_{x=y} = \lim_{h \rightarrow 0} \frac{k(y+h; \lambda) - k(y-h; \lambda)}{2h}, \quad (2.45)$$

yielding Eq. (2.31).

Using the Picard-Lindelöf theorem, as $\tilde{u}_0(x)$ converges uniformly to $u_0(x)$, so does the resulting solution to the Schrödinger equation. Therefore, Eq. (2.32) in the limit when $h \rightarrow 0$ provides the same solution as the traditional Schrödinger equation, in Eq. (2.28). $\square$

See Fig. 2.3 for a comparison.

We have thus reformulated the recovery of Jost solutions ψ to Eq. (2.28), or their scattering data, up to a change of basis, as the problem of recovering the solution to the initial value problem Eq. (2.32), with initial conditions given by the coefficients $c^\pm$ at some $x_i \in \mathbb{R}$. In practice, we have only access to the

(asymptotic) scattering data $c^\pm(-\infty; \lambda)$ or $c^\pm(+\infty; \lambda)$, based on the incoming direction of a plane wave, prior to scattering at the potential given by $u_0(x)$. Moreover, in discrete formulations and with finite resources, we may only be able to practically solve the ODE in Eq. (2.32) in some finite window $[x_i, x_f]$ for given $-\infty < x_i < x_f < \infty$. Combining these two limitations together and supposing that we have access only to $c^\pm(-\infty; \lambda)$, we find that setting the initial condition

$$\hat{c}(x_i; \lambda) = \begin{pmatrix} c^+(-\infty; \lambda) \\ c^-(-\infty; \lambda) \end{pmatrix} \quad (2.46)$$

for the ODE in Eq. (2.32) gives a new solution in the window $[x_i, x_f]$, which we denote by $\hat{c}_{[x_i, x_f]}(x; \lambda)$. Since the initial condition does not agree with the coefficients $c^\pm(x_i; \lambda)$ of the Jost solution $\psi(x_i; \lambda)$, this discrepancy introduces some difference between the true scattering data, which would be given by $\hat{c}_{[-\infty, \infty]}$, and the solution $\hat{c}_{[x_i, x_f]}$.

We will develop solutions to Eq. (2.32) with the limitations just discussed in mind and we will discuss the error $\|\hat{c}_{[x_i, x_f]}(x; \lambda) - \hat{c}_{[-\infty, \infty]}(x; \lambda)\|$ for $x \in [x_i, x_f]$. A numerical and quantum algorithmic approach to approximating $\hat{c}_{[x_i, x_f]}$ can be developed based on the methods of Ref. [54], which abstractly speaking, provides a quantum algorithm to solve linear ODE initial value problems of the following form,

Problem 1 (Adapted from Problem 1 in [54]). *Consider the system*

$$\frac{d\hat{c}(x)}{dx} = \mathbf{A}_c(x)\hat{c}(x) \quad (2.47)$$

where $\hat{c}(x) \in \mathbb{C}^2$, and $\mathbf{A}_c(x) \in \mathbb{C}^{2 \times 2}$ for all $x \in [x_i, x_f]$, and $[\mathbf{A}_c]_{j,k} \in C^\infty([x_i, x_f])$ for all $j, k \in \{1, 2\}$. Suppose that we are given an initial condition $\hat{c}(x_i) \in \mathbb{C}^2$ as well as oracles that compute the locations and values of nonzero entries of $\mathbf{A}_c(x)$ for any x , and that prepare the normalized state $|\hat{c}(x_i)\rangle$ proportional to $\hat{c}(x_i)$ for any $x \in [x_i, x_f]$ (cf. Refs. [54, 52, 57] for detailed oracle definitions). Then, the goal is to output a quantum state $|\hat{c}(x_f)\rangle$ that is proportional to $\hat{c}(x_f)$.

The quantum algorithm presented in Ref. [54] for solving this problem crucially involves yet another reformulation, which linearly transfers Eq. (2.47) into a system of linear ODE initial value problems for the coefficients of the Chebyshev decomposition of the solution. This is efficient due to the smoothness assumption of the entries of $\mathbf{A}_c(x)$, which yield fast decaying Chebyshev coefficients. Moreover, the efficiency of their quantum algorithm benefits from partitioning the solution domain $[x_i, x_f]$ into m smaller sub-intervals and solving for n Chebyshev coefficients in these intervals. Then, the complexity depends on growth rates of $\hat{c}(x)$ and its n derivatives along each of the m sub-intervals. Since Chebyshev polynomials are an orthonormal basis in $L^2([-1, 1])$, each component of the system of linear ODEs along sub-intervals is linearly mapped into the domain $[-1, 1]$. We summarize the steps to this reformulation in the following method.

Method 1 (Liu-Childs, Ref. [54]).

1. Divide the domain $[x_i, x_f]$ into m intervals $[\Gamma_0, \Gamma_1], [\Gamma_1, \Gamma_2], \dots, [\Gamma_{m-1}, \Gamma_m]$ with $\Gamma_0 = x_i, \Gamma_m = x_f$.

Let h denote the index of the interval $I_h := [\Gamma_h, \Gamma_{h+1}]$.

2. Map each interval linearly onto $[-1, 1]$, using the mapping

$$C_h : z \mapsto 1 - \frac{2(z - \Gamma_h)}{\Gamma_{h+1} - \Gamma_h}. \quad (2.48)$$

Let $\hat{\mathbf{c}}_h(x)$ be the solution $\hat{\mathbf{c}}(x)$ at I_h under the mapping C_h .

3. Assuming that the initial condition at each interval is known, formulate the ODE (Eq. (2.47)) using the Chebyshev decomposition of the solution $\hat{\mathbf{c}}_h(x)$ at n Chebyshev points, for each interval.

4. Encode the reformulated ODE (see Eq. (2.10) of Ref. [54]) as a matrix equation (see Section 3 of Ref. [54]).

5. The resulting solution to the matrix equation for Chebyshev coefficients can be reformulated as a vector $\mathbf{c}(x)$ that approximates the desired solution $\hat{\mathbf{c}}(x)$. Given ϵ , we can choose n based on the growth of $\hat{\mathbf{c}}(x)$ and its n derivatives in each of the intervals I_h such that $\mathbf{c}(x)$ is ϵ -close to $\hat{\mathbf{c}}(x)$.

The quantum algorithm introduced in Ref. [54] uses the preceding method to produce a quantum state $|\mathbf{c}_{[x_i, x_f]}(x_f)\rangle$, whose amplitudes are the normalized values $\mathbf{c}_{[x_i, x_f]}(x_f)/\|\mathbf{c}_{[x_i, x_f]}(x_f)\|$ that approximate $\hat{\mathbf{c}}_{[x_i, x_f]}(x_f)/\|\hat{\mathbf{c}}_{[x_i, x_f]}(x_f)\|$ for $\hat{\mathbf{c}}_{[x_i, x_f]}$ the solution to Eq. (2.47) in the domain $[x_i, x_f]$. In our application, this can be employed directly to solve Eq. (2.15) in the window $[x_i, x_f]$ with a given initial condition Eq. (2.46) for each λ in some finite choice of $\Lambda \subset \mathbb{R}$, to yield the state

$$\bigotimes_{\lambda \in \Lambda} \left(\frac{c_{[x_i, x_f]}^+(x_f; \lambda)}{\|\mathbf{c}_{[x_i, x_f]}(x_f; \lambda)\|} |0\rangle + \frac{c_{[x_i, x_f]}^-(x_f; \lambda)}{\|\mathbf{c}_{[x_i, x_f]}(x_f; \lambda)\|} |1\rangle \right). \quad (2.49)$$

Then, the ℓ^2 error in the approximation of the *normalized* solutions, over all $\lambda \in \Lambda$, by the amplitudes of this state is given by

$$\varepsilon := \max_{\lambda \in \Lambda} \left\| \frac{\hat{\mathbf{c}}_{[x_i, x_f]}(x_f; \lambda)}{\|\hat{\mathbf{c}}_{[x_i, x_f]}(x_f; \lambda)\|} - \frac{\mathbf{c}_{[x_i, x_f]}(x_f; \lambda)}{\|\mathbf{c}_{[x_i, x_f]}(x_f; \lambda)\|} \right\|.$$

As discussed above, the discrepancy in the placement of the initial condition at $x_i > -\infty$ introduces an additional source of error, which we call the *cutoff error*, in the obtained solution, given by

$$\epsilon_c := \max_{\lambda \in \Lambda} \left\| \frac{\hat{\mathbf{c}}_f(\lambda)}{\|\hat{\mathbf{c}}_f(\lambda)\|} - \frac{\hat{\mathbf{c}}_{[x_i, x_f]}(x_f; \lambda)}{\|\hat{\mathbf{c}}_{[x_i, x_f]}(x_f; \lambda)\|} \right\| \quad \text{with} \quad (2.50)$$

$$\hat{\mathbf{c}}_f(\lambda) := \lim_{[x_i, x_f] \rightarrow \mathbb{R}} \hat{\mathbf{c}}_{[x_i, x_f]}(x_f; \lambda). \quad (2.51)$$

The total error

$$\epsilon := \max_{\lambda \in \Lambda} \left\| \frac{\hat{\mathbf{c}}_f(\lambda)}{\|\hat{\mathbf{c}}_f(\lambda)\|} - \frac{\mathbf{c}_{[x_i, x_f]}(x_f; \lambda)}{\|\mathbf{c}_{[x_i, x_f]}(x_f; \lambda)\|} \right\| \quad (2.52)$$

scales as $\epsilon \sim \mathcal{O}(\varepsilon + \epsilon_c)$.

We note that, in Section G, we modify the implementation to more easily compute the coefficients $\mathbf{c}$ for

the different λ of interest, and encode in a quantum state. Nonetheless, the error stemming from this analysis, with the state in Eq. (2.49), stands as an upperbound for the implementation discussed in that later Section.

We employ the notation $[m] := \{1, \dots, m\}$ and $[m]_0 := \{0, 1, \dots, m\}$. We are interested in applying Method 1 to Eq. (2.15), for every λ value of interest. From Ref. [54], we then have the following theorem.

Theorem 2 (based on Theorem 1 in [54]). *Consider an instance of the quantum ODE problem as defined in Problem 1, and $\hat{\mathbf{c}}(x)$ its exact solution for a λ value chosen a priori. If assumptions 1 to 5 are valid, then there exists a quantum algorithm that produces a state $\mathbf{c}(x_f)/\|\mathbf{c}(x_f)\|$ ϵ -close to $\hat{\mathbf{c}}(x_f)/\|\hat{\mathbf{c}}(x_f)\|$ in ℓ^2 norm, succeeding with probability $\Omega(1)$, with a flag indicating success, using*

$$O(rq \text{ poly}(\log(r g' / \epsilon g))) \quad (2.53)$$

queries to oracles $O_A(h, l)$ (a sparse matrix oracle for $A_h(t_l)$) and O_x and $O_f(h, l)$. Here,

$$r := \left(\max_{x \in [x_i, x_f]} \left\| e^{\mathbf{A}_c(x; \lambda)(x - x_i)} \right\| \right) \|\mathbf{A}_c\|, \quad \|\mathbf{A}_c\| := \max_{x \in [x_i, x_f]} \|\mathbf{A}_c(x)\|, \quad (2.54)$$

$$g := \|\hat{\mathbf{c}}(x)\|, \quad g' := \max_{x \in [x_i, x_f]} \max_{k \in [n]} \max_{h \in [m]} \|\hat{\mathbf{c}}_h^{(k+1)}(x)\|, \quad q := \max_{x \in [x_i, x_f]} \frac{\|\hat{\mathbf{c}}(x)\|}{\|\mathbf{c}(x_f)\|}. \quad (2.55)$$

The gate complexity is larger than the query complexity by a factor of $\text{poly}(\log(r g' / \epsilon))$.

Proof. From Ref. [54], we have Theorem 1. Modifying it using the results in Ref. [63], we may replace the κ_V dependence by the more general $C(\mathbf{A}_c)$ dependence.

Note that the definition of g' has been tightened, relative to its definition in Ref. [54].

Here we are considering a particular value of λ . Given the construction of the matrix L that includes all $\lambda \in \Lambda$ (as explained in Section G), it is straightforward to show that the theorem applies by replacing every quantity with its maximum over all $\lambda \in \Lambda$.

Assumptions 1 to 4 stem from the application of the Liu-Childs method from Ref. [54] (with the modification based on Ref. [63]). Assumption 5 is an added requirement, since in this case the original setup requires considering $x \in \mathbb{R}$, and not its finite window approximation $x \in [x_i, x_f]$. Using the bounds in Section F, we may choose the window so that assumption 5 holds, which is an equivalent statement to having

$$X \sim \mathcal{O}(\log(1/\epsilon)). \quad (2.56)$$

In this case, the polylog factor in Eq. (2.53) can absorb the X term, so it may be removed from the definition of r . $\square$

For Theorem 2 to hold, the following assumptions need to be valid.

List of assumptions. Let Λ be the set of values of λ to be sampled. These values are set based on the sampling required to accurately extract the scattering data. For the forward scattering transform, we focus on sampling continuous spectrum values ($\lambda \in \mathbb{R}$) for the reflection coefficient $R(\lambda)$. In this Section, we present an analysis of the Forward Scattering step that keeps the set of sampled λ values (i.e., Λ) arbitrary, but finite.

In the context of Lemma 1, consider the following assumptions. For all $x \in [x_i, x_f]$ and $\lambda \in \Lambda$:

Assumption 1. $\mathbf{A}_c(x; \lambda)$ can be diagonalized as $\mathbf{A}_c(x; \lambda) = V D V^{-1}$ where $D(x; \lambda) = \text{diag}(\nu_1(x; \lambda), \nu_2(x; \lambda))$ with $\text{Re}(\nu_i(t)) \leq 0$ for each $i \in \{1, 2\}$.

This assumption is inherited from Method 1, and ensures that the solution to the ODE in Eq. (2.47) doesn't grow arbitrarily in norm.

Assumption 2. *The spectral norm*

$$\|\mathbf{A}_c\| := \max_{x \in [x_i, x_f], \lambda \in \Lambda} \|\mathbf{A}_c(x; \lambda)\| \quad (2.57)$$

is bounded.

The Liu-Childs method starts by reformulating the ODE of interest so that the norm of the resulting rescaled matrix is kept below 1. As a result of this reformulation, the method's complexity scales proportionally to $\|\mathbf{A}_c\|$. Therefore, for the method to be applicable, this norm must be bounded.

Assumption 3. *The quantity*

$$C(\mathbf{A}_c) = \max_{\lambda \in \Lambda, x \in [x_i, x_f]} \left\| e^{\mathbf{A}_c(x; \lambda)(x - x_i)} \right\|. \quad (2.58)$$

is bounded.

Following the analysis in Refs. [54, 63], the quantity $C(\mathbf{A}_c)$ is proportional to a bound for the condition number κ_L of the matrix L defining the matrix ODE to be solved to find $\mathbf{c}(x_f)$. As the complexity of the Liu-Childs method scales at worst with κ_L , the method can only be guaranteed to require a finite number of steps if $C(\mathbf{A}_c)$ is bounded. Note that the result in Ref. [54] use κ_V (the condition number of the matrix V in assumption 1) instead, but their result was improved in Ref. [63] by using $C(\mathbf{A}_c)$ instead. We use the latter in our analysis.

See Fig. 2.4 for common examples of setups where assumptions 2 and 3 hold.

Assumption 4. *The derivatives of $\mathbf{c}_h(x; \lambda)$ obtained from solving Eq. (2.15) are bounded. Its maximum over all intervals I_h is given by*

$$g' := \max_{x \in [x_i, x_f]} \max_{k \in [n]} \max_{h \in [m]} \|\hat{\mathbf{c}}_h^{(k+1)}(x)\|. \quad (2.59)$$

As discussed above, the Liu-Childs method employs Chebyshev decomposition at n points across each of m sub-intervals partitioning the solution domain $[x_i, x_f]$, to approximate the solution to the ODE. The

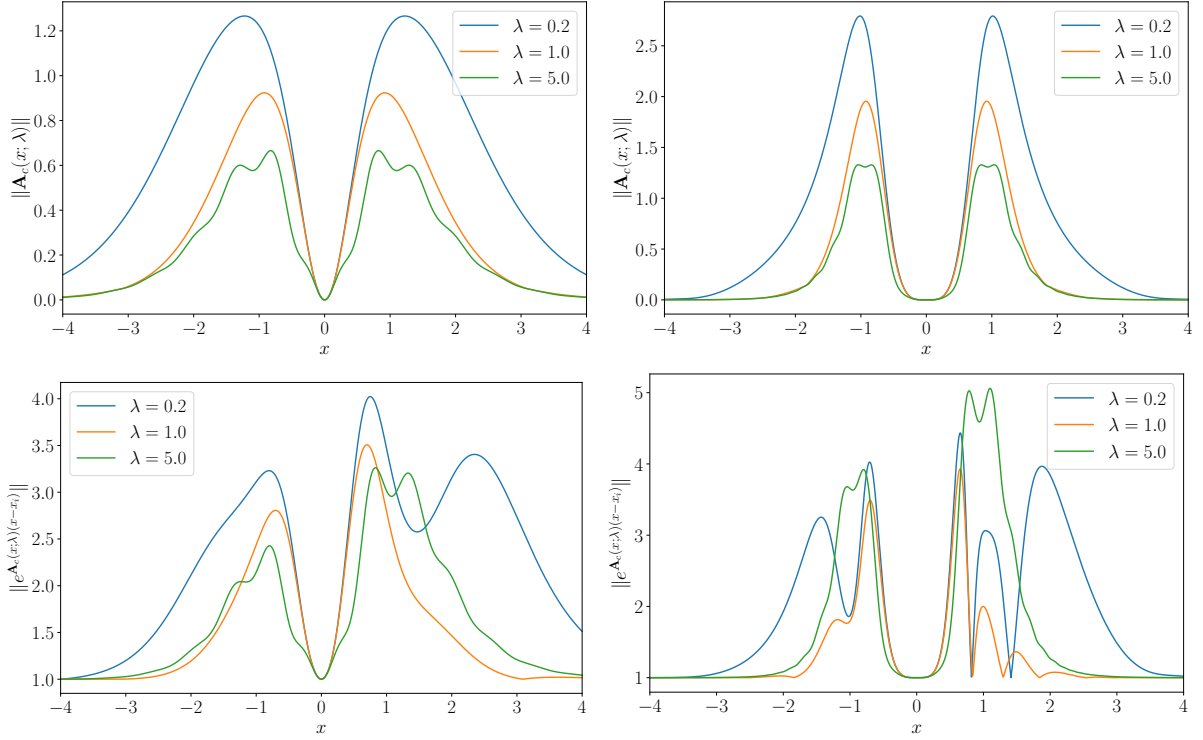


Figure 2.4: $\|\mathbf{A}_c\|$ (top) and $\|e^{\mathbf{A}_c(x; \lambda)(x-x_i)}\|$ (bottom) for $u_0(x) = -2 \operatorname{sech}(x)^2$ (left) and $u_0(x) = -4/(1+x^4)$ (right), with $f(\lambda) = 1$, for different values of λ and for $x \in [-4, 4]$. The $u_0(x) = -4/(1+x^4)$ case incorporates the variable change technique described in Section F, since its default implementation would not satisfy assumption 5. See Section C for a detailed analysis of the norm of $\mathbf{A}_c$.

error of the Chebyshev approximation scales with the norms of the derivatives of the solution, so these being uniformly bounded is sufficient to efficiently bound the approximation error. Here, we use a tighter definition of g' than the one used in Ref. [54], which was likely relaxed for the sake of clarity.

Assumption 5. *Following the discussion after Method 1, we impose the condition that $\epsilon_c \sim \mathcal{O}(\varepsilon)$.*

As previously discussed, in the forward scattering setup we are interested in solving an ODE for $x \in \mathbb{R}$, which clashes with the formulation in Ref. [54], which assumes a finite interval for x . Formulating our setup with a finite window introduces a new source of error, the cutoff error, which is not considered in the complexity analysis of the Liu-Childs method. To circumvent this limitation, assumption 5 implies that the cutoff error does not dominate the total error, and therefore the complexity analysis of Ref. [54] is still applicable to our use case.

The following Sections C to F are dedicated to discussing the conditions under which the preceding assumptions hold. In Section C, we show that assumptions 1 to 3 generally hold. Assumption 4 is discussed in Sections D and E, and assumption 5 in Section F.

C Properties of $\mathbf{A}_c$

Observing that

$$k'(x) = \frac{-u'_0(x)}{2k(x)}, \quad (2.60)$$

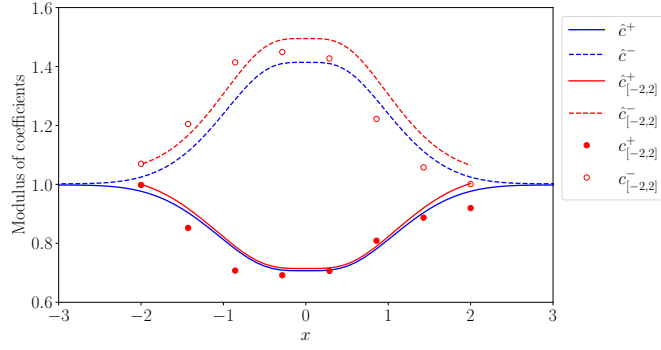


Figure 2.5: Absolute values of the components c^+ (solid lines, filled dots) and c^- (dashed lines, unfilled dots) of the exact solution to Eq. (2.47) for $u_0(x) = -2\text{sech}(x)^2$ and $\lambda = 1$, when considering $x \in \mathbb{R}$ (blue lines) and $x \in [-2, 2]$ (red lines). The finite window introduces a cutoff error that leads to a discrepancy between the obtained values $\hat{c}_{[-2,2]}(x; \lambda)$ and the exact values $\hat{c}(x; \lambda)$. Moreover, the numerical application of the Liu-Childs method also introduces an additional source of error, leading to the values $\mathbf{c}_{[-2,2]}(x; \lambda)$ (dots) that are actually obtained by the method. Here, for the sake of clarify, the errors are greatly exaggerated due to poor choices of x_i, x_f , and parameters for the Liu-Childs method.

we get

$$\mathbf{A}_c(x) = \mathbf{V}(x)\mathbf{\Lambda}(x)\mathbf{V}(x)^{-1} \quad (2.61)$$

$$g(x, k) := -1 + \cos(2kx) + 2x^2k^2 \geq 0, \text{ for both } k \in \mathbb{R} \text{ and } ik \in \mathbb{R} \quad (2.62)$$

$$\mathbf{V}(x) := \begin{pmatrix} f(\lambda) \frac{i\sqrt{2g} - 2\sin(kx)^2}{k(2kx + \sin(2kx))} & f(\lambda) \frac{-i\sqrt{2g} - 2\sin(kx)^2}{k(2kx + \sin(2kx))} \\ 1 & 1 \end{pmatrix} \quad (2.63)$$

$$\mathbf{\Lambda}(x) := i \frac{u'_0(x)\sqrt{g}}{2\sqrt{2}k^2} \begin{pmatrix} -1 & 0 \\ 0 & 1 \end{pmatrix}. \quad (2.64)$$

Since $k^2 \in \mathbb{R}$, then the eigenvalues of $\mathbf{A}_c$ are always purely imaginary. Therefore, assumption 1 always holds.

Let $\kappa_V(x)$ be the condition number of $V(x)$. We observe that $\kappa_V \rightarrow \infty$ only when $x \rightarrow 0$. In this case,

$$\kappa_V \simeq \frac{2\sqrt{3}}{|f(\lambda)||x|}. \quad (2.65)$$

At first, this may seem like an unavoidable limitation, since κ_V appears in the complexity scaling of Ref. [54]. Using its derivation, we would obtain

$$\|e^{\mathbf{A}_c x}\| = \|\mathbf{V}e^{\mathbf{\Lambda} x}\mathbf{V}^{-1}\| \leq \kappa_V, \quad (2.66)$$

since $\mathbf{\Lambda}$ only has imaginary eigenvalues.

Fortunately, when $x \rightarrow 0$, the eigenvalues are also going to zero, and this cancels the rising κ_V . In fact,

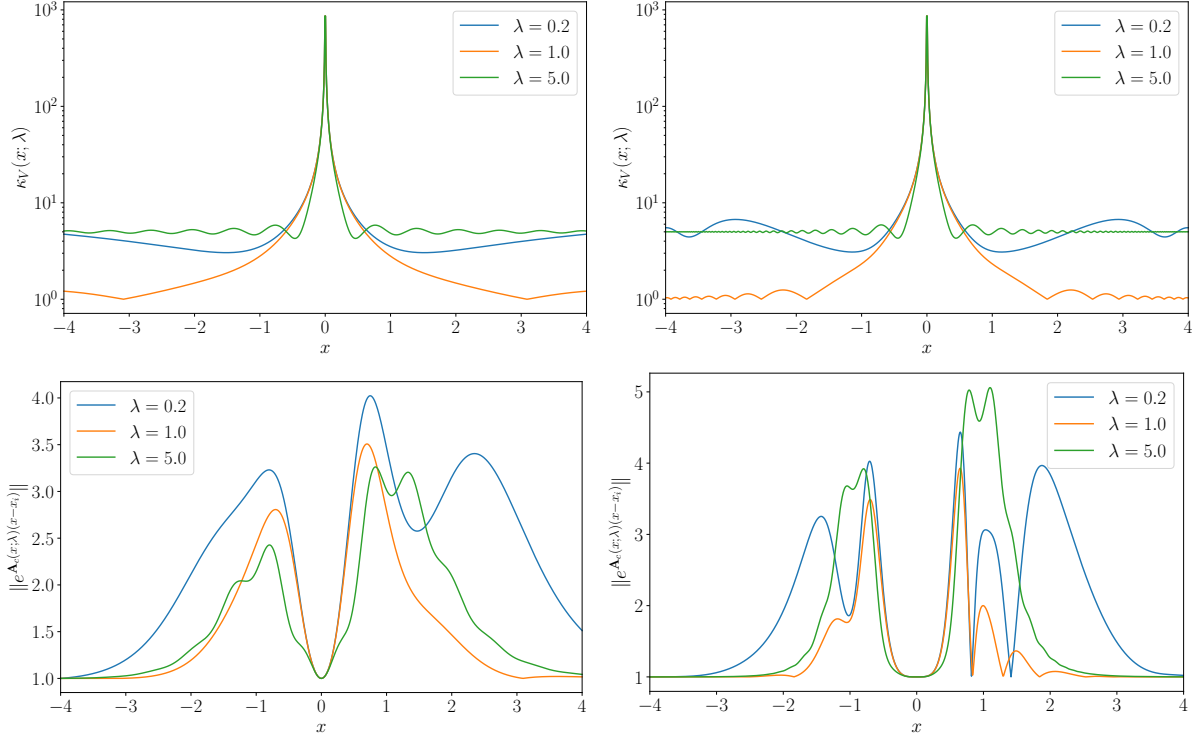


Figure 2.6: κ_V (top, log scale) and $\|e^{\mathbf{A}_c(x;\lambda)(x-x_i)}\|$ (bottom) for $u_0(x) = -2\text{sech}(x)^2$ (left) and $u_0(x) = -4/(1+x^4)$ (right), with $f(\lambda) = 1$, for different values of λ and for $x \in [-4, 4]$. The setup is identical to that in Fig. 2.4. Note that, while $\kappa_V \rightarrow \infty$ as $x \rightarrow 0$, $\|e^{\mathbf{A}_c(x;\lambda)(x-x_i)}\|$ actually goes to zero, indicating that κ_V can be a poor bound to use. As expected, we also have that $\|e^{\mathbf{A}_c(x;\lambda)(x-x_i)}\| \leq \kappa_V$, for all λ, x of interest.

we may use the general bound derived in Ref. [63], and replace κ_V by the general term

$$C_\lambda(\mathbf{A}_c) := \max_{x \in [x_i, x_f]} \|e^{\mathbf{A}_c(x;\lambda)(x-x_i)}\|. \quad (2.67)$$

See Fig. 2.6 for a comparison between the two approaches. For our use case, since $\mathbf{A}_c$ is a 2×2 matrix, it is not difficult to estimate $C_\lambda(\mathbf{A}_c)$ *a priori*, either by computing it directly or by computing some upperbound given in Ref. [63]. Note that, unlike in Refs. [54, 63], the interval used starts at x_i and not zero, so an x_i term must be added to the definition in Eq. (2.67).

For the norm of $\mathbf{A}_c$, we have

$$\begin{aligned} \|\mathbf{A}_c\|_2 &\leq 2\|\mathbf{A}_c\|_{\max} \leq 2\|\mathbf{A}_c\|_2 \\ 2\|\mathbf{A}_c\|_{\max} &= |u'_0(x)| \times \\ &\quad \max \left\{ \frac{|\sin(2kx)|^2}{|k|^2}, \left(|x| + \frac{1}{2} \frac{|\sin(2kx)|}{|k|} \right) \frac{1}{|f(\lambda)|}, \left(\frac{|-2kx + \sin(2kx)|}{2|k|^3} \right) |f(\lambda)| \right\}. \end{aligned} \quad (2.68)$$

This expression includes the case $k \in \mathbb{C} \setminus \mathbb{R}$, where $\sin$ becomes $\sinh$. From Eq. (2.68), we know we can use this bound for the norm of $\mathbf{A}_c$ and at most overestimate it by a factor of 2.

Nonetheless, we observe that, for $k \in \mathbb{R}$, the norm of $\mathbf{A}_c$ is clearly bounded. For imaginary k , the $\sinh$

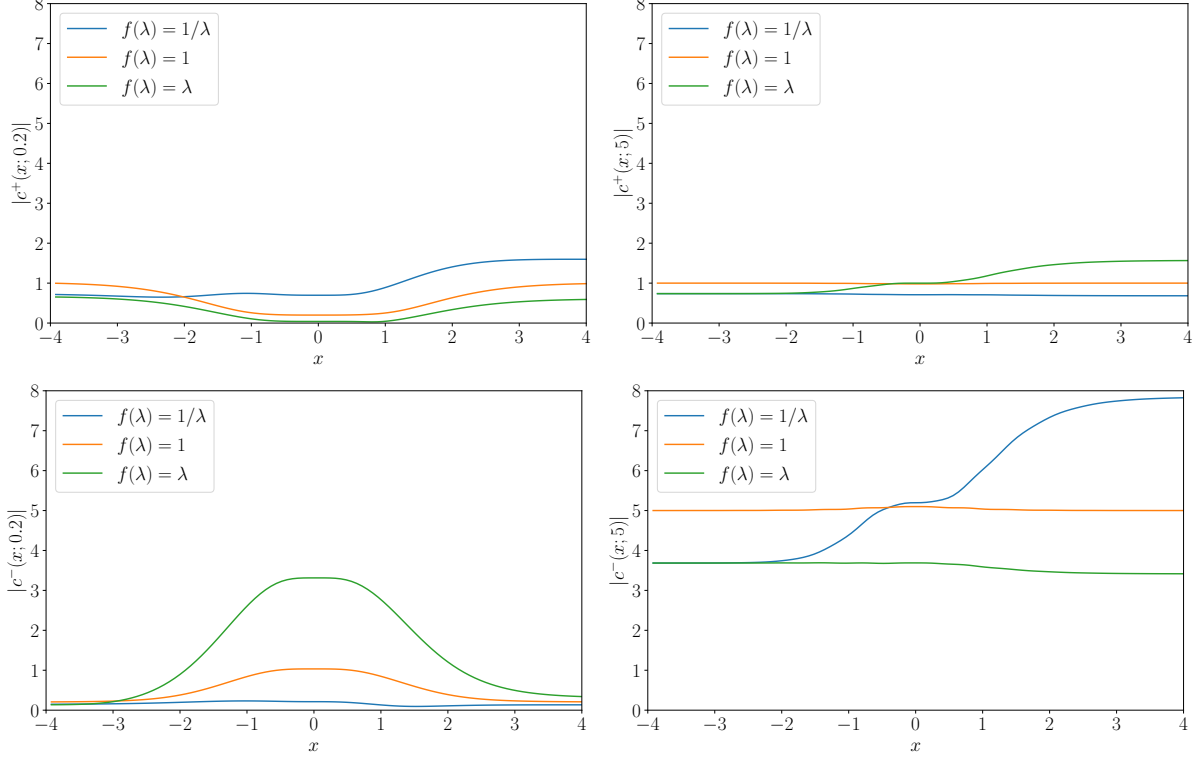


Figure 2.7: Norm of c^+ (top) and c^- (bottom), for $\lambda = 0.2$ (left) and $\lambda = 5$ (right), for different $f(\lambda)$ factors. When λ is high, the dominant component c^- benefits from using $f(\lambda) = \lambda$. When λ is low, $|c^-|$ increases but $|c^+|$ decreases. Therefore, to minimize the norm of $\mathbf{c}$, an acceptable middle ground would be $f(\lambda) = \max\{1, \lambda\}$.

component may lead to an exponential growth in the norm for high x . For Jost solutions, this regime is only encountered for at most a limited window of x (i.e., when $u_0(x) > \lambda^2$), so the norm is still reasonably bounded.

Choice of $f(\lambda)$ factor. We may also control the norm of $\mathbf{A}_c$ by choosing an appropriate function $f(\lambda)$. For instance, by choosing $f(\lambda) = \lambda$ for low λ , we reduce the last factor in Eq. (2.69), and potentially reduce the norm of $\mathbf{A}_c$.

However, it is necessary to consider the effect of $f(\lambda)$ not only on $\mathbf{A}_c$, but also on the norm of $\mathbf{c}(x; \lambda)$, as λ varies. Since, for the Jost solutions, the reflection coefficient $R(\lambda)$ needs to be extracted from $\mathbf{c}(x_f; \lambda)$ by using amplitude amplification (see [54]), poor behavior on the norm of $\mathbf{c}$ for different x and λ may lead the parameter q in Theorem 2 to be unreasonably high.

In particular, we may show that, following the implementation of Section G with $f(\lambda) = 1$, that $\|\mathbf{c}(x_f; \lambda)\| \sim \mathcal{O}(\lambda)$, which harms the sampling of low λ values. By choosing $f(\lambda) = \max\{1, \lambda\}$, we contain the growth of $\|\mathbf{c}\|$, leading to $\|\mathbf{c}(x_f; \lambda)\| \sim \mathcal{O}(1)$. See Fig. 2.7 for examples.

This analysis covers assumptions 2 and 3.

D Derivatives of $\mathbf{c}$

The Liu-Childs method [54] contains a dependence on the derivatives of the solution, through the factor g' . In order to employ the method while preserving the claimed complexity scaling, one must ensure that g' does not scale poorly. In our case, g' is related to the derivatives of $\mathbf{c}$, so we must ensure that these derivatives are well behaved.

When dealing with derivatives of $\mathbf{c}$ that may grow arbitrarily, there are two possible ways to modify the method to guarantee that it still scales as intended. Using the notation of Ref. [54], we may

- Increase m , the number of intervals in the method.
- Increase n , the number of Chebyshev points (per interval).

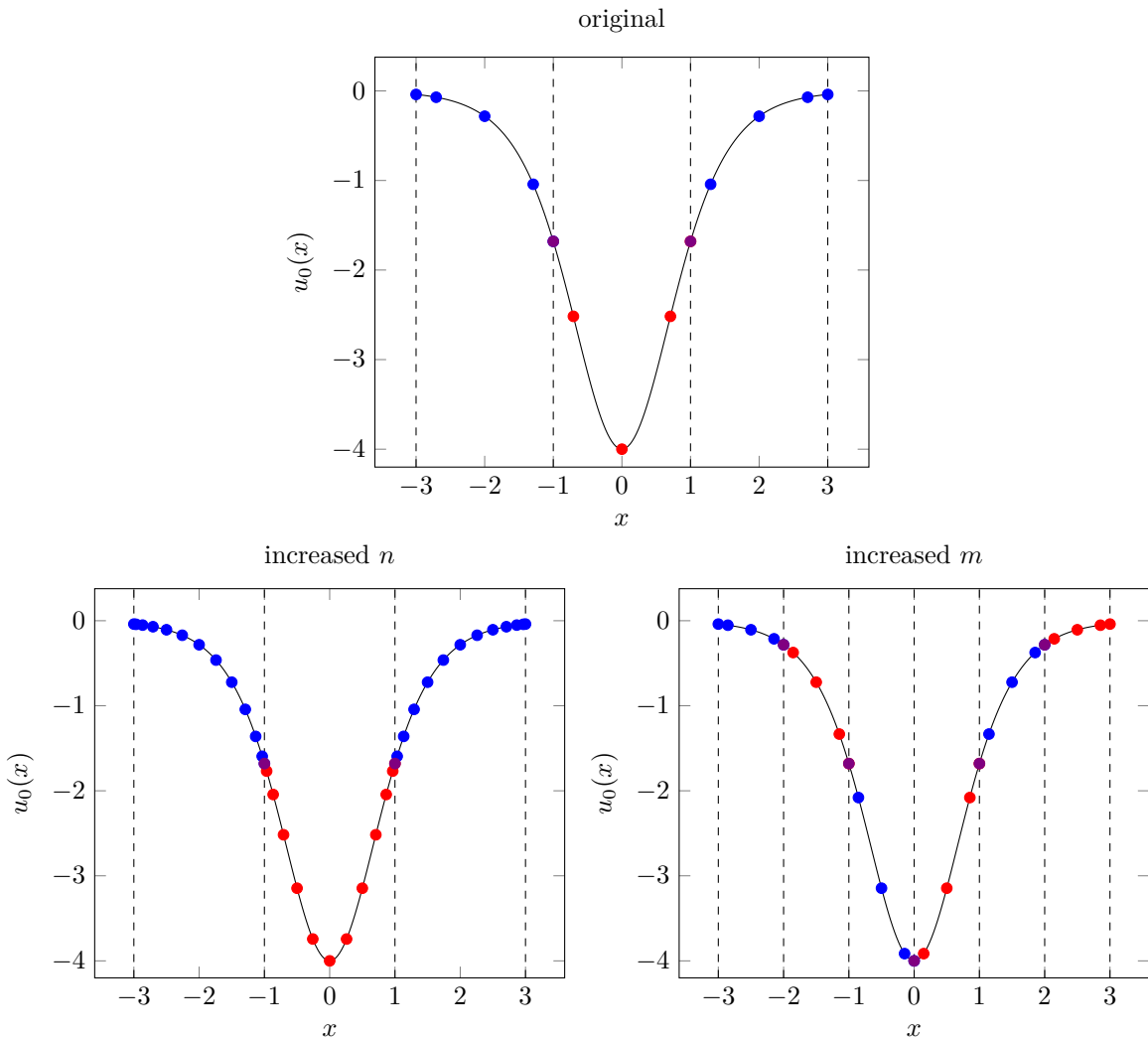


Figure 2.8: Liu-Childs method, with $u_0(x) = -4\text{sech}(x)^2$ (black line), using $m = 3$ and $n = 4$ in its original configuration (top). We use $x_i = -3$ and $x_f = 3$. The vertical dashed lines mark the borders of the m intervals. The $n + 1$ color dots (per interval) mark the $n + 1$ samples of u_0 used by the Liu-Childs method for that interval. The samples at a border are used by both intervals (purple dots). If the derivatives of the desired solution $\mathbf{c}(x)$ do not satisfy the premises of the method, a viable and equivalent problem may be solved instead, by either increasing the number of Chebyshev points per interval (bottom left), or by increasing the number of intervals (bottom right).

See Fig. 2.8 for a pictorial explanation. We can look into how these two approaches impact the complexity scaling.

Lemma 3 (*m* increase). *Suppose the Liu-Childs method is deployed (Method 1). Let n be the number of Chebyshev points used (chosen a priori). Let m be the a priori choice for the number of intervals, dictated by the Liu-Childs method, so that $m \geq \frac{\|A\|X_\epsilon}{2}$. Let $\mathbf{c}_h$ be the $\mathbf{c}$ coefficients rescaled such that $x \in [-1, 1]$, for interval I_h ($h \in [m]_0$). Let $M(k)$ be a known positive function, and $\mu(n)$ its maximum in the region $[1, n+1]$. If*

$$\|\partial_x^k \mathbf{c}_h(x)\| \leq M(k)^k, \quad \forall x, k, h \quad (2.70)$$

then rescaling m to be

$$m_{\text{new}} = \mu(n)m \quad (2.71)$$

results in new rescaled coefficients $\mathbf{c}_p^{\text{new}}$ for each of the m_{new} intervals, with

$$\|\partial_x^k \mathbf{c}_p^{\text{new}}(x)\| \leq 1, \quad \forall x \in [-1, 1], k \leq n+1, p \in [m_{\text{new}}]_0. \quad (2.72)$$

The rescaling results in an increase in m by a factor of $\mu(n)$, so the Liu-Childs complexity scaling is increased by a multiplicative factor of $\mu(n)$.

Proof. Consider the setup of the Liu-Childs method (Method 1). W.l.o.g., assume the intervals I_h are of equal length $\xi = X_\epsilon/m$. Following Eq. (2.48), the mapping is given by

$$C_h : z \mapsto 1 - \frac{2(z - \Gamma_h)}{\Gamma_{h+1} - \Gamma_h} = 1 - \frac{2(z - \Gamma_h)}{\xi}. \quad (2.73)$$

After the rescaling, we have $m_{\text{new}} = \mu(n)m$ intervals, each of size $\xi/\mu(n)$. Consequently, the new mapping is

$$C_p^{\text{new}} : z \mapsto 1 - \mu(n) \frac{2(z - \Gamma_p^{\text{new}})}{\xi}. \quad (2.74)$$

Let $\mathbf{c}(x)$ be the original coefficients in the domain $[x_i, x_f]$, $\mathbf{c}_h(x)$ the rescaled coefficients in interval I_h with the original m , and $\mathbf{c}_p^{\text{new}}(x)$ the rescaled coefficients in the refinement $[\Gamma_p^{\text{new}}, \Gamma_{p+1}^{\text{new}}]$ (for $p \in [m_{\text{new}}]_0$) of the intervals $[\Gamma_h, \Gamma_{h+1}]$ of $p \in [m]_0$. We have

$$\frac{\partial^k \mathbf{c}(x)}{\partial x^k} = \frac{\partial^k \mathbf{c}_h(C_h(x))}{\partial x^k} = \frac{\partial^k \mathbf{c}_p^{\text{new}}(C_p^{\text{new}}(x))}{\partial x^k}. \quad (2.75)$$

Looking at the first derivative, we have

$$\frac{\partial \mathbf{c}_h(C_h(x))}{\partial x} = \frac{\partial \mathbf{c}_p^{\text{new}}(C_p^{\text{new}}(x))}{\partial x} \quad (2.76)$$

$$\left. \frac{\partial \mathbf{c}_h(y)}{\partial y} \right|_{y=C_h(x)} \frac{\partial C_h(x)}{\partial x} = \left. \frac{\partial \mathbf{c}_p^{\text{new}}(z)}{\partial z} \right|_{z=C_p^{\text{new}}(x)} \frac{\partial C_p^{\text{new}}(x)}{\partial x} \quad (2.77)$$

$$\left. \frac{\partial \mathbf{c}_h(y)}{\partial y} \right|_{y=C_h(x)} \left(-\frac{2}{\xi} \right) = \left. \frac{\partial \mathbf{c}_p^{\text{new}}(z)}{\partial z} \right|_{z=C_p^{\text{new}}(x)} \left(-\frac{2\mu(n)}{\xi} \right) \quad (2.78)$$

$$\left. \frac{\partial \mathbf{c}_h(y)}{\partial y} \right|_{y=C_h(x)} = \left. \frac{\partial \mathbf{c}_p^{\text{new}}(z)}{\partial z} \right|_{z=C_p^{\text{new}}(x)} \mu(n). \quad (2.79)$$

Similarly, we may show that

$$\left. \frac{\partial^k \mathbf{c}_h(y)}{\partial y^k} \right|_{y=C_h(x)} = \left. \frac{\partial^k \mathbf{c}_p^{\text{new}}(z)}{\partial z^k} \right|_{z=C_p^{\text{new}}(x)} \mu(n)^k \quad (2.80)$$

$$\implies \left\| \frac{\partial^k \mathbf{c}_h(y)}{\partial y^k} \right\|_{y=C_h(x)} = \left\| \frac{\partial^k \mathbf{c}_p^{\text{new}}(z)}{\partial z^k} \right\|_{z=C_p^{\text{new}}(x)} \mu(n)^k. \quad (2.81)$$

Therefore, if

$$\left\| \frac{\partial^k \mathbf{c}_h(y)}{\partial y^k} \right\| \leq M(k)^k, \quad \forall y, k, h \quad (2.82)$$

then

$$\left\| \frac{\partial^k \mathbf{c}_h(y)}{\partial y^k} \right\| \leq M(k)^k \quad (2.83)$$

$$\left\| \frac{\partial^k \mathbf{c}_p(z)}{\partial z^k} \right\| \mu(n)^k \leq M(k)^k, \quad \text{with } z = C_p^{\text{new}}(C_h^{-1}(y)) \quad (2.84)$$

$$\left\| \frac{\partial^k \mathbf{c}_p(z)}{\partial z^k} \right\| \leq \left(\frac{M(k)}{\mu(n)} \right)^k \leq 1, \quad \forall x \in [-1, 1], k \leq n+1, p \in [m_{\text{new}}]_0. \quad (2.85)$$

as expected. Since the Liu-Childs method scales with $\mathcal{O}(m)$, the new scaling is $\mathcal{O}(m_{\text{new}}) = \mathcal{O}(\mu(n)m)$. $\square$

Lemma 4. *By applying the mapping in Eq. (2.48) (with equal sized intervals) to $\mathbf{c}(x)$, we get*

$$\left\| \frac{\partial^k \mathbf{c}_h(x)}{\partial x^k} \right\| = \left(\frac{X}{2m} \right)^k \left\| \frac{\partial^k \mathbf{c}(x)}{\partial x^k} \right\|. \quad (2.86)$$

Proof. Using the same reasoning as in Lemma 3, we have (using $\xi = X/m$)

$$\left. \frac{\partial^k \mathbf{c}_h(y)}{\partial y^k} \right|_{y=C_h(x)} \left(-\frac{2}{\xi} \right)^k = \left. \frac{\partial^k \mathbf{c}(z)}{\partial z^k} \right|_{z=x} \quad (2.87)$$

$$\left. \frac{\partial^k \mathbf{c}_h(y)}{\partial y^k} \right|_{y=C_h(x)} = \left(-\frac{X}{2m} \right)^k \left. \frac{\partial^k \mathbf{c}(z)}{\partial z^k} \right|_{z=x}, \quad (2.88)$$

yielding Eq. (2.86). $\square$

Lemma 5 (n increase). *Assume m is fixed a priori. Let $\mathbf{c}_h$ be the $\mathbf{c}$ coefficients rescaled such that $x \in [-1, 1]$, for interval I_h , with $h \in [m]_0$. Let $N(k)$ be a known positive function.*

If

$$\left\| \frac{\partial^k \mathbf{c}_h(x)}{\partial x^k} \right\| \leq N(k), \quad \forall x, h; k \leq n, \quad (2.89)$$

and

$$N(k+1) \leq e^{\beta k \log k}, \text{ for } 0 < \beta < \frac{2}{e}, \quad (2.90)$$

then, for sufficiently small ϵ , we have

$$n \sim \mathcal{O} \left(\frac{1}{\frac{2}{e} - \beta} \log \left(\frac{1}{\epsilon} \right) / \log \left(\frac{1}{\frac{2}{e} - \beta} \log \left(\frac{1}{\epsilon} \right) \right) \right) \sim \mathcal{O}(\log(1/\epsilon)), \quad (2.91)$$

that is, the complexity scaling of n with ϵ for the Liu-Childs method is maintained.

Proof. From the Liu-Childs method [54], let

$$G(n) := \max_{h, x \in [-1, 1]} \|\partial_x^{n+1} \mathbf{c}_h(x)\| \quad (2.92)$$

$$\Omega(n, \epsilon) := \frac{G(n)em(1 + \epsilon)}{g\epsilon} \quad (2.93)$$

$$\theta(n, \epsilon) := \frac{\log(\Omega(n, \epsilon))}{\log(\log(\Omega(n, \epsilon)))} - \left\lfloor \frac{\log(\Omega(n, \epsilon))}{\log(\log(\Omega(n, \epsilon)))} \right\rfloor. \quad (2.94)$$

We have that if

$$\omega(n) := \frac{G(n)}{\|\gamma\|} (m + 1) \quad (2.95)$$

$$n = \frac{e}{2} \max \left\{ \left\lfloor \frac{\log(\Omega(n, \epsilon))}{\log(\log(\Omega(n, \epsilon)))} \right\rfloor, \left\lfloor \frac{\log(\omega(n))}{\log(\log(\omega(n)))} \right\rfloor \right\}, \quad (2.96)$$

where $\|\gamma\|$ is the norm of the initial state, then the error associated with the Liu-Childs method is upper bounded by ϵ . For sufficiently small ϵ , the $\Omega(n, \epsilon)$ term in Equation (2.96) dominates over the $\omega(n)$ term. We have

$$\frac{2}{e}n = \frac{\log(\Omega(n, \epsilon))}{\log(\log(\Omega(n, \epsilon)))} - \theta(n, \epsilon) \quad (2.97)$$

$$\iff we^w = -\frac{1}{s} \quad (2.98)$$

$$\text{with } w = -\frac{1}{s} \log(\Omega(n, \epsilon)), \quad s := \frac{2}{e}n + \theta(n, \epsilon) \quad (2.99)$$

Equation (2.98) has one viable solution in this context, the Lambert W_{-1} function. Consequently, we have

$$w = W_{-1}(-1/s) \quad (2.100)$$

$$\iff \Omega(n, \epsilon) = \exp(-sW_{-1}(-1/s)) \quad (2.101)$$

$$\epsilon = \frac{\frac{emG(n)}{g} \exp(sW_{-1}(-1/s))}{1 - \frac{emG(n)}{g} \exp(sW_{-1}(-1/s))}, \quad (2.102)$$

However, the direct utility of this last expression is limited, since it implicitly depends on ϵ through the definition of s and θ . For sufficiently high n , s is also high, and we have

$$-W_{-1}(-1/s) = \log(s) + \log(\log(s)) + o(1). \quad (2.103)$$

Considering the worst case, where $\theta(n, \epsilon) = 0$, we have the simpler upper bound

$$\epsilon \leq \frac{emG(n)}{g} \exp\left(-\frac{2}{e}n \left[\log\left(\frac{2}{e}n\right) + \log\left(\log\left(\frac{2}{e}n\right)\right) \right]\right) \quad (2.104)$$

Necessarily, we have $G(n) \leq N(n+1)$, and the dominating term is $\exp(-\frac{2}{e}n \log n)$, leading to

$$\epsilon \sim \mathcal{O}\left(N(n+1) \exp\left(-\frac{2}{e}n \log n\right)\right), \quad (2.105)$$

If Equation (2.90) is satisfied, then

$$\epsilon \sim \mathcal{O}\left(\exp\left(-\left(\frac{2}{e} - \beta\right)n \log n\right)\right) \quad (2.106)$$

$$\iff \log n \sim \mathcal{O}\left(W_0\left(\frac{1}{\frac{2}{e} - \beta} \log\left(\frac{1}{\epsilon}\right)\right)\right). \quad (2.107)$$

where we used the Lambert W_0 function. Exponentiating on both sides and using the asymptotic limit

$$W_0(x) = \log x - \log(\log x) + o(1), \text{ for } x \rightarrow \infty, \quad (2.108)$$

we retrieve Equation (2.91). □

Theorem 3. *Let $N(k), M(k)$ be known positive functions, and*

$$\mu(n) := \max_{1 \leq k \leq n+1} M(k) \quad (2.109)$$

$$N(k+1) \leq e^{\beta k \log k}, \text{ for } 0 < \beta < \frac{2}{e}. \quad (2.110)$$

Let $\mathbf{c}_h$ be the rescaled coefficients for interval I_h . If

$$\|\partial_x^k \mathbf{c}_h(x)\| \leq N(k) M(k)^k, \quad \forall x, k, h, \quad (2.111)$$

then, increaing m by a factor of $\mu(n)$, we end up with

$$\|\partial_x^k \mathbf{c}_p^{\text{new}}(x)\| \leq N(k), \quad \forall x \in \mathbb{R}, k \leq n+1, p \quad (2.112)$$

and, for sufficiently small ϵ ,

$$n \sim \mathcal{O}(\log(1/\epsilon)), \quad (2.113)$$

as required to maintain the scaling associated with the Liu-Childs method.

Proof. Using Lemma 3 with $m_{\text{new}} = \mu(n)m$, we end up with an equivalent problem where Equation (2.112) holds. Since Equation (2.110) holds, then, from Lemma 5, we conclude that Equation (2.113) holds. □

Corollary 1. *In particular, if the derivatives are bounded by a factorial, as in*

$$\|\partial_x^k \mathbf{c}_h(x)\| \leq (bk)! R^k, \quad \forall x, k, h \quad (2.114)$$

for some $b \in \mathbb{R}^+$, then, since $(bk)!R^k \leq (bkR)^k$, we may choose

$$\mu(n) = nbR \quad (2.115)$$

$$N(k) = 1, \quad (2.116)$$

so the complexity grows by a multiplicative nbR factor. The extra $n \sim \mathcal{O}(\log(1/\epsilon))$ factor is absorbed by the polylog term in the Liu-Childs complexity expression.

Lemma 6. For $D \subseteq \mathbb{R}$, if $u_0 \in \mathcal{C}^n(D)$, then $\mathbf{c}(x)$ is in $\mathcal{C}^{n-1}(D)$, for a given λ . If u_0 is analytic in D , so is $\mathbf{c}(x)$.

Proof. Let $g(x) = (\lambda^2 - u_0(x))x^2$. Since x^2 and λ^2 are in $C^\omega(\mathbb{R})$, we conclude that, for $D \subseteq \mathbb{R}$,

$$u_0 \in \mathcal{C}^n(D) \implies g \in \mathcal{C}^n(D). \quad (2.117)$$

Noting that

$$\cos(k(x)x) = \cos(k(x)|x|) = h_1(g(x)) \quad (2.118)$$

$$f(\lambda) \frac{\sin(k(x)x)}{k(x)} = f(\lambda) \frac{\sin(k(x)|x|)}{k(x)|x|} x = x f(\lambda) h_2(g(x)) \quad (2.119)$$

with

$$h_1(x) := \begin{cases} \cos(\sqrt{x}), & \text{if } x \geq 0, \\ \cosh(\sqrt{-x}), & \text{if } x < 0. \end{cases} \quad (2.120)$$

$$h_2(x) := \begin{cases} \sin(\sqrt{x})/\sqrt{x}, & \text{if } x > 0, \\ \sinh(\sqrt{-x})/\sqrt{-x}, & \text{if } x < 0, \\ 1, & \text{if } x = 0. \end{cases} \quad (2.121)$$

we observe that $h_1(x)$ and $h_2(x)$ are analytic in $\mathbb{R}$, that is, in $C^\omega(\mathbb{R})$. Given the rules for the composition of differentiable (and analytic) functions, and the dependence of the entries of $\mathbf{A}_c$ on $u'_0(x)$, we conclude

$$u_0(x) \in \mathcal{C}^n(D) \implies u'_0(x) \in \mathcal{C}^{n-1}(D), g(x) \in \mathcal{C}^n(D) \implies [\mathbf{A}_c]_{ab} \in \mathcal{C}^{n-1}(D). \quad (2.122)$$

Similarly, we have

$$u_0(x) \in \mathcal{C}^\omega(D) \implies u'_0(x) \in \mathcal{C}^\omega(D), g(x) \in \mathcal{C}^\omega(D) \implies [\mathbf{A}_c]_{ab} \in \mathcal{C}^\omega(D). \quad (2.123)$$

Given the differential equation $\partial_x \hat{\mathbf{c}}(x) = \mathbf{A}_c(x) \hat{\mathbf{c}}(x)$, we have that $\mathbf{A}_c(x) \in \mathcal{C}^n(D)$ implies $\hat{\mathbf{c}}(x) \in \mathcal{C}^n(D)$ (and similarly for analyticity), reaching the result in the lemma. $\square$

Remark 1. The n -th derivative of a real analytic function for $|x| < L$ is upper bounded by $MR^n n!$, for

some value of R and M dependent on C . Thus, by Lemma 6, if $u_0(x)$ is analytic, so is $\mathbf{c}(x)$, and its derivatives follow the bound in Corollary 1, meaning that Theorem 3 can be applied for $|x| < L$.

E Relation between $\mathbf{c}$ and u_0

Since the behavior of $\mathbf{c}$ is not known *a priori*, we may attempt to estimate it from the known behavior of u_0 .

The behavior of the derivatives of $\mathbf{c}$ may also be estimated from behavior of the $\mathbf{A}_c$ entries.

Theorem 4. *Let*

$$U_s(x) := \max_{1 \leq r \leq s} |u_0^{(r)}(x)|. \quad (2.124)$$

If

$$u_0^{(r)} \sim o(1/|x|), \quad \forall r \geq 1, \quad (2.125)$$

then, by employing Theorem 3, we get

$$\|\partial_x^s \mathbf{c}_p(x)\| \sim \mathcal{O}(U_{s+1}(x)x), \quad (2.126)$$

for sufficiently high $|x|$, for interval I_p .

Proof. We have

$$k'(x) = \frac{-u_0'(x)}{2k(x)}. \quad (2.127)$$

Note that each entry of $\mathbf{A}_c(x)$ can be written as

$$Q(C_0, C_1, \alpha_0, \alpha_1, \beta, \gamma) u_0'(x) \quad (2.128)$$

with

$$Q(C_0, C_1, \alpha_0, \alpha_1, \beta, \gamma) := C_0 k(x)^{\alpha_0} p(x, \beta) + C_1 k(x)^{\alpha_1} T(x, \gamma) \quad (2.129)$$

$$C_0, C_1 \in \mathbb{R} \quad (2.130)$$

$$\alpha_0, \alpha_1, \beta \in \mathbb{Z} \quad (2.131)$$

$$p(x, \beta) := \begin{cases} x, & \beta = 1 \\ 1, & \beta = 0 \\ 0, & \beta < 0 \end{cases} \quad (2.132)$$

$$T(x, \gamma) := \sin\left(2k(x)x + \frac{\pi}{2}\gamma\right), \gamma \in \mathbb{Z}. \quad (2.133)$$

For $\lambda \in \mathbb{R}$, $\forall \delta > 0$, $\exists C$ such that, for $|x| > C$,

$$|k(x) - |\lambda|| < \delta \quad (2.134)$$

$$|T(x, \gamma)| \leq 1, \quad (2.135)$$

Consider a smooth function $P(x)$ such that $|P^{(s)}(x)| \leq \max\{1, |x|\}, \forall x \in \mathbb{R}, s \in \mathbb{N}_0$. For an arbitrary function

$$R(x) := k(x)^\alpha P(x), \quad \alpha \in \mathbb{Z}, \quad (2.136)$$

a bound on $R(x)$ and its derivatives will be given by

$$|R^{(s)}(x)| = \left| \sum_{k=0}^s \binom{s}{k} P^{(k)}(x) \frac{\partial^{(s-k)} k(x)^\alpha}{\partial x^{(s-k)}} \right|, \quad s \in \mathbb{N}_0 \quad (2.137)$$

$$\leq \max\{1, |x|\} \sum_{k=0}^s \binom{s}{k} \left| \frac{\partial^{(s-k)} k(x)^\alpha}{\partial x^{(s-k)}} \right|. \quad (2.138)$$

Given Eq. (2.127), the s -th derivative of $k(x)^\alpha$ will be a linear combination of $N \sim \mathcal{O}(a^s)$ terms (for some constant $a \geq 1$) of the form

$$(\partial_x^s u_0(x))^{j_s} \cdots (\partial_x^2 u_0(x))^{j_2} (\partial_x u_0(x))^{j_1} k(x)^\beta, \quad \text{with } j_i \in \mathbb{N}_0, \beta \in \{\alpha, \dots, \alpha - 2s\}. \quad (2.139)$$

Given Eq. (2.125), the single term with $j_s = 1, j_{\neq n} = 0$ will dominate, being upperbounded by $\mathcal{O}(U_s(x))$ defined in Eq. (2.124). Equation (2.125) implies that the remaining terms follow, in the worst case, $\mathcal{O}(U_s(x)/|x|)$. To simplify the analysis, we can consider the scaling of all these terms to be $U_s(x)$. Therefore, we have

$$|R^{(s)}(x)| \leq \max\{1, |x|\} \sum_{j=0}^s \binom{s}{j} \mathcal{O}(a^{s-j}) \mathcal{O}(U_s(x)) \quad (2.140)$$

$$\leq \mathcal{O}((2a)^s U_s(x) |x|). \quad (2.141)$$

As $\mathbf{A}_c$ can be written as in Eq. (2.128), and the functions $p(x; \beta)$ and $T(x; \gamma)$ can be thought of as taking the role of the function $P(x)$ in the definition of $R(x)$, then each entry of $\partial_x^s \mathbf{A}_c$ can be described as the function $R(x)$ in Eq. (2.136), which we know is bounded by $\mathcal{O}((2a)^s U_s(x) |x|)$, for some $a \in \mathbb{R}$.

The norm $\|\partial_x^s \mathbf{c}\|$ can then be similarly bounded by a linear combination of $\mathcal{O}(b^s)$ terms (for some constant $b \geq 1$) of $\|\partial_x^j \mathbf{A}_c\|$, for $j \leq s$. Consequently, we get

$$\|\partial_x^s \mathbf{c}(x)\| \sim \mathcal{O}((2ab)^s U_{s+1}(x) |x|), \quad (2ab) \in \mathbb{R}^+. \quad (2.142)$$

By considering Lemma 4, and using Theorem 3 with

$$\mu(n) = (2ab) \frac{X}{2m}, \quad (2.143)$$

we may absorb the $(2ab)^s$ term, yielding the result in the theorem. The behavior of $\mathbf{c}(x)$ and $\mathbf{c}_p(z)$ may

be related by using Lemma 4.

□

Remark 2 (Connection to main theorem assumptions). *When the condition in Theorem 4 is satisfied—namely, when $u_0^{(r)} \sim o(1/|x|)$ for all $r \geq 1$ —the bounds on $\|\partial_x^s \mathbf{c}_p(x)\|$ established in the theorem guarantee that the derivative growth assumptions required by the Liu-Childs method (assumptions 1 to 5 in Theorem 2) are satisfied. Similarly, when u_0 is analytic (as discussed in Lemma 6 and the accompanying remark), the factorial bound in Corollary 1 ensures all technical assumptions hold. Together, these results establish the precise function classes for which our quantum algorithm achieves the complexity stated in Theorem 1.*

F Minimizing cutoff error

In order to compute the scattering data, we must know the behavior of ψ at $x \rightarrow \pm\infty$. If we attempted to employ the standard technique of Ref. [54], this would require using $\mathbb{R}$ as the domain of interest, which is infeasible for the method. Consequently, we consider a bounded domain $[x_i, x_f]$, where x_i and x_f are chosen to be in a region where the initial condition u_0 is negligible. Nonetheless, this limitation imposes an additional source of error, the cutoff error, that will depend on how negligible the ignored region actually is. To circumvent this limitation, there are several possible techniques [90]. We may employ a spectral method different from Chebyshev’s at the boundaries, i.e., one that is suitable for infinitely sized domains, such as Laguerre’s [96]. Alternatively, we may start by applying a suitable change of basis, which compresses the x domain, so as to concentrate the effect of the u_0 potential in the region considered. Ultimately, we wish to choose a change of variables that enables us to preserve the logarithmic dependence on ϵ in the complexity of the method. See Figure 2.9 for a few examples. In general, this approach leads to Lemma 8.

Lemma 7. *When using the Liu-Childs method with the finite size window $[x_i, x_f]$, the cutoff error scales as*

$$\epsilon_c \sim \mathcal{O}\left(\int_{-\infty}^{x_i} |u'_0(x)x|dx + \int_{x_f}^{\infty} |u'_0(x)x|dx\right). \quad (2.144)$$

Proof. First consider the cutoff present due to x_f being finite. For high x , given Lemma 1, the derivative of $\mathbf{c}$ can be upper-bounded, elementwise, by

$$\partial_x c^\pm(x) \leq C|u'_0(x)x|c^\pm(x), \quad (2.145)$$

for some constant C . Assuming the worst case scenario, we have

$$\frac{\partial_x c^\pm(x)}{c(x)} = |u'_0(x)x| \quad (2.146)$$

$$\implies \lim_{x \rightarrow +\infty} c_{[-\infty, x_f]}^\pm(x) = c_{[-\infty, x_f]}^\pm(x_f) \exp\left(C \int_{x_f}^{\infty} |u'_0(x)x|dx\right), \quad (2.147)$$

from which we get that the cutoff error (following Eq. (2.50)) is approximately given by

$$\mathcal{O}\left(\exp\left(C \int_{x_f}^{\infty} |u'_0(x)x|dx\right) - 1\right) \quad (2.148)$$

$$\sim \mathcal{O}\left(\int_{x_f}^{\infty} |u'_0(x)x|dx\right), \quad \text{for high } x_f. \quad (2.149)$$

Following a similar derivation for the cutoff at x_i , we get Eq. (2.144). $\square$

Corollary 2. For high $|x|$, if $u_0(x) \sim \mathcal{O}(\exp(-cx))$, for some $c \in \mathbb{R}$, and picking $-x_i = x_f = X/2$, then

$$\epsilon_c \sim \mathcal{O}(X \exp(-cX/2)). \quad (2.150)$$

Corollary 3. For high $|x|$, if $u_0(x) \sim \mathcal{O}(\frac{1}{x^c})$, for some $c > 1$, and picking $-x_i = x_f = X/2$, then

$$\epsilon_c \sim \mathcal{O}\left(\frac{1}{X^{c-1}}\right). \quad (2.151)$$

Lemma 8. Let $u_0(x) \in \mathcal{C}^n$ (resp. $\mathcal{C}^\omega$). Defining $x =: v(y)$, for some bijective function $v \in \mathcal{C}^n$ (resp. $\mathcal{C}^\omega$) with non-zero derivative, leads to Equation (2.15) converting into

$$\partial_y \tilde{\mathbf{c}}(y; \lambda) = \frac{\partial \tilde{k}}{\partial y} \begin{pmatrix} -\frac{\sin(\tilde{k}v(y))^2}{\tilde{k}} & \frac{-2\tilde{k}v(y) + \sin(2\tilde{k}v(y))}{2\tilde{k}^2} f(\lambda) \\ \frac{2\tilde{k}v(y) + \sin(2\tilde{k}v(y))}{2f(\lambda)} & \frac{\sin(\tilde{k}v(y))^2}{\tilde{k}} \end{pmatrix} \begin{pmatrix} \tilde{c}^+ \\ \tilde{c}^- \end{pmatrix} \quad (2.152)$$

$$=: \tilde{\mathbf{A}}_c(y) \tilde{\mathbf{c}}(y; \lambda), \quad (2.153)$$

with $\tilde{k} := \sqrt{\lambda^2 - \tilde{u}_0(y)}$, $\tilde{u}_0(y) := u_0(v(y))$ and $\tilde{\mathbf{c}}(y; \lambda) := \mathbf{c}(v(y); \lambda)$. We also have

$$\|\tilde{\mathbf{A}}_c\| = |v'(y)| \|\mathbf{A}_c\|_{x=v(y)}. \quad (2.154)$$

Proof. From Equation (2.31), we have

$$\tilde{\mathbf{A}}_c(y) = \mathbf{A}_c(x)|_{x=v(y)} v'(y), \quad (2.155)$$

for some choice of $v(y)$. Similarly, we have

$$\partial_y \tilde{\mathbf{c}}(y; \lambda) = \partial_x \mathbf{c}(x; \lambda)|_{x=v(y)} v'(y), \quad (2.156)$$

leading to

$$\partial_x \mathbf{c}(x; \lambda)|_{x=v(y)} = \mathbf{A}_c(x)|_{x=v(y)} \mathbf{c}(x; \lambda)|_{x=v(y)} \quad (2.157)$$

$$\frac{1}{v'(y)} \partial_y \tilde{\mathbf{c}}(y; \lambda) = \frac{1}{v'(y)} \tilde{\mathbf{A}}_c(y) \tilde{\mathbf{c}}(y; \lambda) \quad (2.158)$$

$$\partial_y \tilde{\mathbf{c}}(y; \lambda) = \tilde{\mathbf{A}}_c(y) \tilde{\mathbf{c}}(y; \lambda), \quad (2.159)$$

which is Equation (2.152). Moreover, from Equation (2.155), we get

$$\|\tilde{\mathbf{A}}_c(y)\| = \|\mathbf{A}_c(x)|_{x=v(y)} v'(y)\| = |v'(y)| \|\mathbf{A}_c\|_{x=v(y)}, \quad (2.160)$$

thereby proving Equation (2.154). $\square$

Remark 3. *In practice, it is useful to focus on choices of $v(y)$ that are convenient to minimize the cutoff error, while not distorting the original behavior of the problem in the region where the potential $u_0(x)$ is far from zero. A good example would be $v(y) = U_0 \sinh((y - y_0)/U_0)$, where y_0 is the center of mass of $|u_0(x)|$ (i.e. $y_0 = \int_{\mathbb{R}} x |u_0(x)| dx / \int_{\mathbb{R}} |u_0(x)| dx$) and U_0 is the interval length where the potential is most concentrated. It behaves as $v(y) \simeq y$ for y in the region where u_0 is non-negligible, but then increases exponentially in the region where u_0 is small, and where we wish to reduce the associated cutoff error.*

Lemma 9. *Following Lemma 8, let $v(y) = \sinh(y)$. If*

$$\|\partial_x^s \mathbf{c}(x)\| \sim \mathcal{O}(g_s(x)), \quad \text{as } |x| \rightarrow \infty, \quad (2.161)$$

for some function $g_s(x)$, then

$$\|\partial_y^s \tilde{\mathbf{c}}(y)\| \sim \mathcal{O}\left(\max_{1 \leq j \leq s} \{v(y)^j g_j(v(y))\}\right). \quad (2.162)$$

Proof. Using Faà di Bruno's formula, we have

$$\partial_y^n \tilde{\mathbf{c}}(y) = \sum_{k=1}^n B_{n,k}(v'(y), v''(y), \dots, v^{(n-k+1)}(y)) \partial_x^k \mathbf{c}(x), \quad (2.163)$$

where $B_{n,k}$ are the exponential Bell polynomials. With this expression, we can find a crude bound for $\|\partial_x^n \tilde{\mathbf{c}}(y)\|$.

$$\|\partial_y^n \tilde{\mathbf{c}}(y)\| = \left\| \sum_{k=1}^n B_{n,k}(v'(y), v''(y), \dots, v^{(n-k+1)}(y)) \partial_x^k \mathbf{c}(x) \right\| \quad (2.164)$$

$$\leq \sum_{k=1}^n |B_{n,k}(v'(y), v''(y), \dots, v^{(n-k+1)}(y))| \|\partial_x^k \mathbf{c}(x)\| \quad (2.165)$$

$$\leq \sum_{k=1}^n B_{n,k}(|v'(y)|, |v''(y)|, \dots, |v^{(n-k+1)}(y)|) \|\partial_x^k \mathbf{c}(x)\| \quad (2.166)$$

$$\leq \sum_{k=1}^n B_{n,k}(V_n(y), V_n(y), \dots, V_n(y)) \|\partial_x^k \mathbf{c}(x)\|, \text{ with } V_n(y) := \max_{1 \leq j \leq n} |v^{(j)}(y)| \quad (2.167)$$

$$= \sum_{k=1}^n \left\{ \begin{matrix} n \\ k \end{matrix} \right\} V_n(y)^k \|\partial_x^k \mathbf{c}(x)\|. \quad (2.168)$$

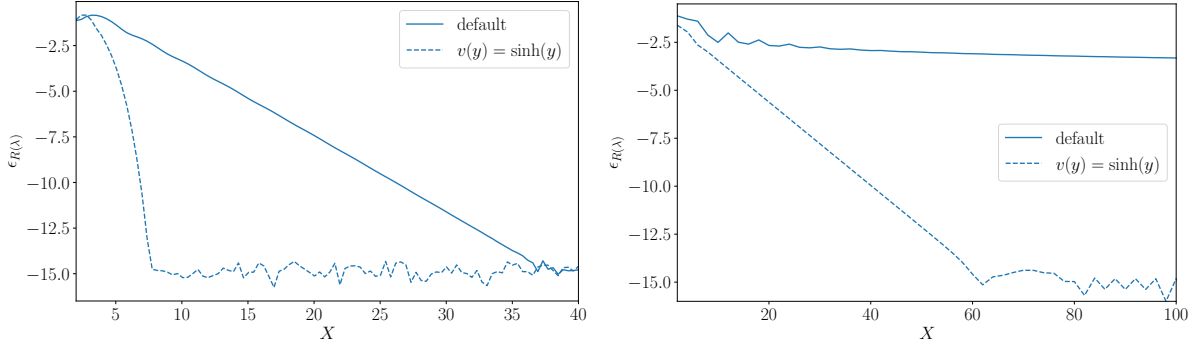


Figure 2.9: $R(\lambda)$ error as a function of $X = |x_f - x_i|$, either using the default method or preapplying a change of variable, with $v(y)$, as in Lemma 8. For both plots, $n = 30$ and $m = 5X$ were used, in order to leave the cutoff as the only source of error. (Left) For $u_0(x) = -4 \operatorname{sech}(x)^2$, the cutoff error ϵ dependence on X goes from $\mathcal{O}(\exp(-cX))$ (exponential) to $\mathcal{O}(\exp(\exp(-cX)))$ (super-exponential), for some c . (Right) For $u_0(x) = -4/(x^2 + 1)$, the dependence goes from $\mathcal{O}(1/\operatorname{poly}(X))$ to $\mathcal{O}(\exp(-cX))$. The variable change thus enables the latter case to maintain a logarithmic scaling with ϵ , in the forward scattering process, since we'll have $X_\epsilon = \operatorname{polylog}(1/\epsilon)$. The error plateaus indicate floating point limitations (for $\epsilon \simeq 10^{-15}$).

where $\left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\}$ are the Stirling numbers of the second kind. For $v(y) = \sinh(y)$, we have

$$|v(y)| \rightarrow \exp(y)/2, \quad \text{as } |x| \rightarrow \infty \quad (2.169)$$

$$|v^{(j)}(y)| \rightarrow \exp(y)/2, \quad \text{as } |x| \rightarrow \infty, \quad (2.170)$$

so that $V_n(y) = \exp(y)/2 = x$. Using Eq. (2.161), we conclude that

$$\|\partial_y^n \tilde{\mathbf{c}}(y)\| \sim \sum_{k=1}^n \left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\} x^k \mathcal{O}(g_k(x)) \quad (2.171)$$

$$\sim \mathcal{O}(G_n(x)) \sum_{k=1}^n \left\{ \begin{smallmatrix} n \\ k \end{smallmatrix} \right\}, \quad \text{with } G_n(x) := \max_{1 \leq j \leq n} |x^j g_j(x)| \quad (2.172)$$

$$\sim \mathcal{O}(G_n(x)) B_n \quad (2.173)$$

$$\sim \mathcal{O}(G_n(x)) \left(\frac{0.792n}{\log(n+1)} \right)^n, \quad (2.174)$$

with B_n the Bell numbers.

The n^n factor can be absorbed using Theorem 3, yielding

$$\|\partial_y^n \tilde{\mathbf{c}}(y)\| \sim \mathcal{O}(G_n(x)) \quad (2.175)$$

$$\sim \mathcal{O}(G_n(v(y))). \quad (2.176)$$

□

Remark 4. In order to incorporate the various approaches discussed in this section, some care needs to

be taken. The approach in Theorem 3 should be taken first, yielding bounds for $\partial_x^n \mathbf{c}$ derivatives. Once these bounds are known, Lemma 9 can be used, as it establishes a relation between the previously computed scaling of $\partial_x^n \mathbf{c}$ and that of $\partial_y^n \tilde{\mathbf{c}}$. If the latter's scaling is adequate, then the method's implementation can proceed.

We note that the scaling given by Theorem 3 and Lemma 9 is not tight, so it is possible for the Liu-Childs method to work with initial conditions for which these results do not provide adequate bounds for the derivatives of $\mathbf{c}$. An example can be seen on the right of Fig. 2.9, where $u_0(x) \sim \mathcal{O}(1/x^c)$ (for some $c > 2$) appears to work numerically, even though Lemma 9 would suggest unbounded derivatives for $\tilde{\mathbf{c}}(y)$.

G Implementing the forward scattering method

For the particular case of the Jost Solutions, we are interested in extracting the reflection coefficient $R(\lambda)$, for all $\lambda \in \Lambda$. In particular, we wish to obtain the quantum state

$$\sum_{\lambda \in \Lambda} \frac{c_{[x_i, x_f]}^-(x_f; \lambda)}{\sqrt{\sum_{\lambda} |c_{[x_i, x_f]}^-(x_f; \lambda)|^2}} |\lambda\rangle = \frac{1}{\mathcal{N}} \sum_{\lambda \in \Lambda} R(\lambda) |\lambda\rangle, \quad (2.177)$$

where $\mathcal{N}$ stands for the normalization factor. Note that this quantum state differs from the one defined in Eq. (2.49), since here we are computing the result for the different λ values in superposition, instead of using different registers.

In Section B, we set up the scattering data (under a linear change of coordinates) in the format given by Lemma 1, derived from the Schrödinger equation. While this is the version to consider for the numerical analysis using the method in Ref. [54], it poses a challenge to define the boundary conditions for the Schrödinger equation, and to compute the scattering data.

Consider an alternative formulation. Decomposing ψ as

$$\psi(x; \lambda) = a^+(x; \lambda)e^{+i\lambda x} + a^-(x; \lambda)e^{-i\lambda x}, \quad (2.178)$$

enables us to straightforwardly define the initial conditions, by choosing values for $a^\pm$, and enables to straightforwardly extract the scattering data, since the $a^\pm(x \rightarrow \pm\infty; \lambda)$ coefficients will incorporate the sought-after scattering coefficients. As we cannot work numerically with an infinitely sized window for x , we consider a finite approximation, where we work in the interval $[x_i, x_f]$, with $X := |x_f - x_i|$ its length. For the sake of simplicity, we define $a_i^\pm(\lambda) := a^\pm(x_i; \lambda)$ and $a_f^\pm(\lambda) := a^\pm(x_f; \lambda)$. We have

$$\psi(x_i; \lambda) \xrightarrow{u_0(x)} \psi(x_f; \lambda) \quad (2.179)$$

$$a_i^+(\lambda)e^{+i\lambda x} + a_i^-(\lambda)e^{-i\lambda x} \xrightarrow{u_0(x)} a_f^+(\lambda)e^{+i\lambda x} + a_f^-(\lambda)e^{-i\lambda x}, \quad (2.180)$$

where $\rightarrow$ denotes the effect of the potential u_0 on ψ through the Schrödinger equation, and how the behavior of ψ at x_i shapes its behavior at x_f . The transmission and reflection coefficients ($T(\lambda)$ and

$R(\lambda)$, respectively) are such that

$$e^{i\lambda x} \xrightarrow{u_0(x)} \frac{1}{T(\lambda)} e^{i\lambda x} + \frac{R(\lambda)}{T(\lambda)} e^{-i\lambda x}, \quad (2.181)$$

or alternatively

$$T(\lambda) e^{i\lambda x} \xrightarrow{u_0(x)} e^{i\lambda x} + R(\lambda) e^{-i\lambda x}. \quad (2.182)$$

As previously stated, we may impose initial conditions on the Schrödinger equation by choosing the values of these $\mathbf{a} := \begin{pmatrix} a^+ & a^- \end{pmatrix}^T$ coefficients. We must have

$$\psi(x; \lambda) = a^+(x; \lambda) e^{+i\lambda x} + a^-(x; \lambda) e^{-i\lambda x} = c^+(x; \lambda) \cos(kx) + c^-(x; \lambda) f(\lambda) \frac{\sin(kx)}{k}, \quad (2.183)$$

where k is given in Equation (2.29). Equivalently,

$$\psi(x; \lambda) = \mathbf{a}^T \begin{pmatrix} e^{+i\lambda x} \\ e^{-i\lambda x} \end{pmatrix} = \mathbf{c}^T \begin{pmatrix} \cos(kx) \\ f(\lambda) \frac{\sin(kx)}{k} \end{pmatrix} = \mathbf{c}^T H^T \begin{pmatrix} e^{+i\lambda x} \\ e^{-i\lambda x} \end{pmatrix}, \quad (2.184)$$

$$\text{with } H(x; \lambda) := \frac{1}{2} \begin{pmatrix} 1 & \frac{-if(\lambda)}{k} \\ 1 & \frac{if(\lambda)}{k} \end{pmatrix}, \quad (2.185)$$

implying

$$\mathbf{a} = H\mathbf{c}. \quad (2.186)$$

Therefore, specifically for the points x_i and x_f , we may incorporate a basis change into the formulation in Ref. [54], so that we compute the initial and final points using the $e^{\pm i\lambda x}$ basis with $\mathbf{a}$, while keeping the convenient formulation with the $\mathbf{c}$ coefficients for the overall computation.

From Equation (2.178), we have 4 parameters, $a_i^\pm$ and $a_f^\pm$. From these, we may choose two initially, with the other two being determined by satisfying the Schrödinger equation. In particular, for the Jost solutions (i.e., $\lambda_j \in \mathbb{R}$), we use the formulation of Equation (2.182), so that $R(\lambda) = a_f^-(\lambda)$.

We note that, depending on the choice of $a^\pm$ parameters to set, this formulation may no longer correspond to an initial value problem, and instead to a boundary value problem. In particular, for the Jost solutions, we set the value of a_f^+ to be 1. While the main result of Ref. [54] is applicable to initial value problems, the modification we make to the Liu-Childs method in the rest of this section extends the technique for boundary value problems. While Ref. [54] had also covered boundary value problems, their approach was suboptimal, since it did not split the x domain into intervals (i.e. it used $m = 1$). As a result, the optimal choice of n scaled with $\|\mathbf{A}\|$, which led the full algorithm to scale with $\|\mathbf{A}\|^4$. We adapt their method by maintaining the splitting of the problem into m intervals, which makes the complexity analysis for initial value problems applicable to the boundary value problem case we cover here.

The linear system to be solved has the form

$$L|X\rangle = |B\rangle, \quad (2.187)$$

with

$$L := L(\lambda_1) \oplus \cdots \oplus L(\lambda_{N_\lambda}) = \bigoplus_{j=1}^{N_\lambda} L(\lambda_j) \quad (2.188)$$

$$|X\rangle := \bigoplus_{j=1}^{N_\lambda} |X(\lambda_j)\rangle \quad |B\rangle := \bigoplus_{j=1}^{N_\lambda} |B(\lambda_j)\rangle, \quad (2.189)$$

where $\oplus$ denotes the direct sum, and $N_\lambda = |\Lambda|$. For each λ_j , the L matrix, $|X\rangle$, and $|B\rangle$ vectors used are based on those from [54]. Since there is no interplay between the different λ_j , we may analyze each λ_j value independently. With these choices, the linear system has the form (with $d = 2$)

$$L(\lambda_j) = \begin{pmatrix} L_1^i + L_2(A_0) & & & & L_1^f & & & & \\ & L_3 & & L_1 + L_2(A_1) & & & & & \\ & & \ddots & & & & & & \\ & & & L_3 & & L_1 + L_2(A_{m-1}) & & & \\ & & & & L_3^R & & L_4^R & & \\ & & & & & & L_5^R & L_4^R & \\ & & & & & & & \ddots & \\ & & & & & & & & L_5^R & L_4^R \end{pmatrix} \quad (2.190)$$

$$P_n = \sum_{l,k=0}^n \cos(kl\pi/n) |l\rangle\langle k| \quad (2.191)$$

$$D_n = \sum_{\substack{k,j=0 \\ j>k \\ k+j \text{ odd}}}^{n+1} \frac{2j}{\sigma_k} |k\rangle\langle j|, \quad \sigma_k = \begin{cases} 2 & k=0 \\ 1 & k \in [n] \end{cases} \quad (2.192)$$

$$L_1 = I_d \otimes \left(|0\rangle\langle 0| P_n + \sum_{l=1}^n |l\rangle\langle l| P_n D_n \right) \quad (2.193)$$

$$L_2(A_h) = - \sum_{l=1}^n A_h(t_l) \otimes |l\rangle\langle l| P_n \quad (2.194)$$

$$L_3 = \sum_{i=0}^{d-1} \sum_{k=0}^n (-1)^k |i0\rangle\langle ik| \quad (2.195)$$

$$L_3^R = \sum_{k=0}^n (-1)^k |10\rangle\langle 1k| \quad (2.196)$$

$$L_4^R = - \sum_{l=1}^n |1l\rangle\langle 1, l-1| + \sum_{l=0}^n |1l\rangle\langle 1l| \quad (2.197)$$

$$L_5^R = -|10\rangle\langle 1n|. \quad (2.198)$$

Note that the matrices P_n, D_n, L_1, L_2 , and L_3 have the same definitions of Ref. [54]. The matrices L_3^R , L_4^R , and L_5^R have slightly modified definitions. As we are only concerned with extracting the reflection coefficient $R(\lambda_j) = a_f^-(\lambda_j)$, and not the full $\mathbf{c}(x_f; \lambda_j)$ vector, the matrices are slightly modified from those of Ref. [54] to ensure only copies of $R(\lambda_j)$ are encoded in the $|X(\lambda_j)\rangle$ state, instead of the full $\mathbf{a}$ or $\mathbf{c}$

vector. See Equation (2.199) for the usual form of $|X(\lambda_j)\rangle$. The matrices L_1^i and L_1^f impose two of the four possible $a_i^\pm/a_f^\pm$ values as initial conditions. Irrespective of being in x_i or x_f , we impose the values of a^+ (resp. a^-) by imposing the first (resp. second) row of H in Equation (2.185); call this row h^+ (resp. h^-), so that $H = \begin{pmatrix} h^+ & h^- \end{pmatrix}^T$. Let $w := (1, \dots, 1)$ be a vector of size $n+1$, and $v^\pm := h^\pm \otimes w$. L_1^i starts by being equal to the L_1 matrix. If we are setting a_i^+ , we set the first row of L_1^i to be v^+ ; otherwise, we set the row's elements to zero. If we are setting a_i^- , we set the $(n+2)$ -th row of L_1^i to be v^- ; otherwise, we set the row to zero. L_1^f is initially a $d(n+1) \times d(n+1)$ zero matrix. If we are setting a_f^+ , we set the first row of L_1^f to be v^+ ; otherwise, we set the row's elements to zero. If we are setting a_f^- , we set the $(n+2)$ -th row of L_1^f to be v^- ; otherwise, we set the row to zero.

The vector $|X(\lambda_j)\rangle$ has the form

$$|X(\lambda_j)\rangle = \begin{pmatrix} c_{1,0}(\Gamma_1) \\ c_{1,1}(\Gamma_1) \\ \vdots \\ c_{1,n}(\Gamma_1) \\ c_{2,0}(\Gamma_1) \\ c_{2,1}(\Gamma_1) \\ \vdots \\ c_{2,n}(\Gamma_1) \\ c_{1,0}(\Gamma_2) \\ \vdots \\ c_{2,n}(\Gamma_m) \\ R(\lambda_j) \\ R(\lambda_j) \\ R(\lambda_j) \\ R(\lambda_j) \\ R(\lambda_j) \\ R(\lambda_j) \end{pmatrix} \quad (2.199)$$

where the entries are the amplitudes of the basis states, up to a normalization factor. The $c_{i,l}(\Gamma_{h+1})$ terms are the Chebyshev series coefficients for the i -th components of $\mathbf{c}(\Gamma_{h+1})$ and $a_f^-(\lambda_j) = R(\lambda_j)$ is the final state.

The vector $|B(\lambda_j)\rangle$ has the form

$$|B(\lambda_j)\rangle = \begin{pmatrix} 0 \\ \vdots \\ 0 \\ 0 \\ 0 \\ \vdots \\ 0 \\ 1 \\ \vdots \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} \quad (2.200)$$

where the entries are the amplitudes of the basis states, up to a normalization factor. The 1 entries come from setting $a_f^+(x_f) = 1$.

Example. For $u_0(x) = q \operatorname{sech}(x)^2$, $R(\lambda)$ is known to have the analytic expression

$$\delta = \sqrt{\frac{1}{2} - q} \quad (2.201)$$

$$R(\lambda) = \frac{\cos(\pi\delta)}{\pi} \frac{\Gamma(i\lambda)}{\Gamma(-i\lambda)} \Gamma\left(\frac{1}{2} - i\lambda + \delta\right) \Gamma\left(\frac{1}{2} - i\lambda - \delta\right) \quad (2.202)$$

A numerical example of the method's convergence can be seen in Figure 2.10.

H Formal complexity statement

Having established the implementation details of the forward scattering algorithm in this appendix, we now present the formal complexity statement that ties together all the components discussed above. This lemma provides the precise query complexity for producing the quantum state encoding the reflection coefficients, building on Theorem 2 and the stabilization techniques described in Section 2.2.

Lemma 10 (Formal Complexity Statement). *Given an oracle O_c that efficiently prepares the initial conditions for $\mathbf{c}$, and oracles that compute the locations and values of nonzero entries of $\mathbf{A}_c(x; \lambda)$ for any x, λ , the goal is to output the quantum state $|R\rangle \propto \sum_{\lambda} R(\lambda) |\lambda\rangle$, where $R(\lambda) = a^-(x \rightarrow +\infty; \lambda)$. Then there exists a quantum algorithm that produces a state ϵ -close to $|R\rangle$ in l^2 norm, using*

$$\mathcal{O}(rq \operatorname{polylog}(rg'/\epsilon g)) \quad (2.203)$$

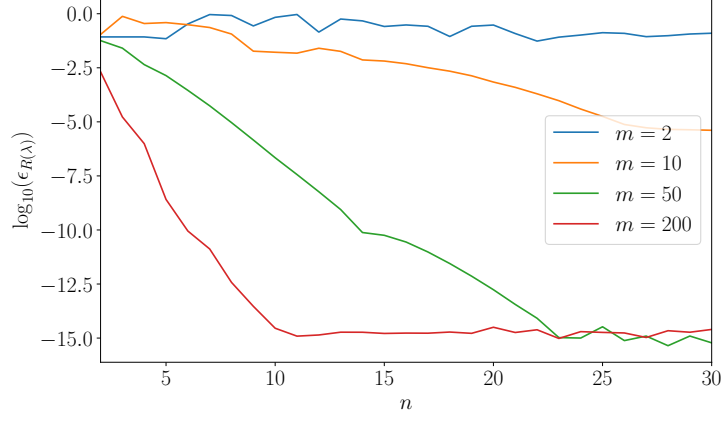


Figure 2.10: $R(\lambda)$ error as a function of the parameters m and n in the Liu-Childs method [54], for $u_0(x) = -4 \operatorname{sech}(x)^2$. As expected, the error decays exponentially quickly as n increases, with the decay being faster for higher m . Nonetheless, as long as m is higher than the minimum set by the method (in this case $m > 2$), the slope is negative. The plateau at 10^{-15} is due to floating point limitations.

queries to oracles O_A and O_c , and with

$$\begin{aligned}
r &= \|\mathbf{A}_c\| \max_x \|e^{\mathbf{A}_c x}\| \quad \|\mathbf{A}\| := \max_{x \in [x_i, x_f]} \|\mathbf{A}(x)\| \quad g := \|\mathbf{c}(x_f)\| \\
g' &:= \max_{x \in [x_i, x_f], h \in [m]_0} \max_{k \leq n} \frac{1}{k+1} \|\mathbf{c}^{(k+1)}(x)\|^{\frac{1}{k+1}} \\
q &:= \max_{x, \lambda} \|\mathbf{c}(x; \lambda)\| / \langle |R(\lambda)|^2 \rangle \\
n &= \mathcal{O}(\log(r / \min\{g\epsilon/e, \|\mathbf{c}(x_i)\|\}))
\end{aligned}$$

where n is the number of Chebyshev points used. The gate complexity is larger than the query complexity by a factor of $\text{poly}(\log(N_\lambda r/\epsilon))$.

This lemma, combined with the stabilization formulation in Lemma 1 and the conditions established in Sections E and F, constitutes the complete formal statement of our main result.

Chapter 3

Quantum Error Correction via Noise Guessing Decoding

Abstract

Quantum error correction codes (QECCs) play a central role in both quantum communications and quantum computation. Practical quantum error correction codes, such as stabilizer codes, are generally structured to suit a specific use, and present rigid code lengths and code rates. This paper shows that it is possible to both construct and decode QECCs that can attain the maximum performance of the finite blocklength regime, for any chosen code length when the code rate is sufficiently high. A recently proposed strategy for decoding classical codes called GRAND (guessing random additive noise decoding) opened doors to efficiently decode classical random linear codes (RLCs) performing near the maximum rate of the finite blocklength regime. By using noise statistics, GRAND is a noise-centric efficient universal decoder for classical codes, provided that a simple code membership test exists. These conditions are particularly suitable for quantum systems, and therefore the paper extends these concepts to quantum random linear codes (QRLCs), which were known to be possible to construct but whose decoding was not yet feasible. By combining QRLCs and a newly proposed quantum-GRAND, this work shows that it is possible to decode QECCs that are easy to adapt to changing conditions. The paper starts by assessing the minimum number of gates in the coding circuit needed to reach the QRLCs' asymptotic performance, and subsequently proposes a quantum-GRAND algorithm that makes use of quantum noise statistics, not only to build an adaptive code membership test, but also to efficiently implement syndrome decoding.

3.1 Introduction

Quantum technologies have the potential to revolutionize industries across the globe. By harnessing the power of quantum mechanics, technologies such as quantum computing, quantum simulation, quantum communication, and quantum sensing can be used to solve a wide range of hard problems in different areas,

such as in the pharmaceutical, materials science, cryptography, machine learning, logistics optimization, finance, energy, and aerospace sectors [97, 98, 99]. Nevertheless, for these technologies to be viable, significant progress needs to be made in quantum error correction, to reduce the error rates of these quantum systems to tolerable levels.

Implementing quantum computers and other devices bearing quantum memories will require quantum error correction of the stored quantum states [100, 101]. Moreover, a number of applications, such as quantum key distribution, distributed quantum computing, or the connection of different quantum sensors, require the existence of quantum networks and quantum repeaters in order to sustain end-to-end quantum connections between several devices [102, 103, 104, 105, 106, 107]. Quantum states are notoriously sensitive to noise, so quantum error correction must be employed when establishing entanglement or for preserving quantum states in memory. Quantum gates still exhibit very high error rates, even though they have fallen in the last decade from $\approx 10^{-2}$ [108] to $\approx 10^{-3}$ [109]). Thus, it is paramount to have scalable quantum error correction codes (QECCs), meaning that errors can be exponentially corrected while the encoding/decoding complexity increases sub-exponentially with the number of qubits in the codewords. A very recent breakthrough toward this objective was communicated in [109], showing experimental results for a surface code that seem to pave the way for practical and scalable QECCs. The problem of quantum error correction has been looked at in [110] in a broader prospective than the traditional one of designing a fixed code. The authors proposed using quantum neural networks that work as autoencoders that optimally adapt the design of the QECC to the existing noise statistics. Note that making use of noise statistics is also a central idea of the proposal in the present work. QECCs designed for a particular type of channels, where the statistics of the errors show that one type of quantum errors is the most likely one, appeared in [111]. [112] presented an encoding method for quantum random codes and analyzed its performance. [113] performed a similar analysis, but using a code construction akin to low-density parity-check codes (LDPCs) and based on constraint satisfaction. The works in [112] and [113] are mainly concerned with the efficiency of the *encoding* process, and both consider a decoder for an erasure channel, rather than a Pauli decoherence channel, which is considered in our work. Furthermore, important findings for QECCs designed for the deletion channel (i.e., when a qubit is removed without notice from a stream of qubits) have been communicated in [114].

A Short linear codes: a historical context

Many ideas for designing QECC can be traced down to the developments made in classical error correction, and in [115] one can find an exhaustive list of those connections. The concept of error correction appeared in tandem with the one of channel capacity in the pioneering work of Claude Shannon [116], who proved that the capacity of a channel could be attained by using a uniform-at-random code that maps any k bits of information onto codewords n bits long. This mapping should be random, in the sense that each of the 2^k codewords should be assigned to one of the 2^n possible words, and should be made according to a uniform distribution. Shannon's random code construction provably reaches capacity when the length of the codewords tends to infinity ($n \rightarrow \infty$) and the decoder picks, among all possible codewords, the

most likely one, by applying maximum-likelihood (ML) decoding with the received observation and the codewords' prior distribution. Nevertheless, Shannon's codes are too complex and have never been used in practice because that would imply storing all the 2^k codewords at the transmitter side, and applying ML decoding at the receiver, which is an NP-hard problem.

Classical random linear codes (RLCs) offer a solution to the codewords' storage problem, given that only the *random* generation matrix needs to be stored. Moreover, they are known to achieve capacity in the binary symmetric channel [116, 117]. Although an exhaustive search among all the 2^k codewords of a RLC remains prohibitively complex, ML decoding is known to be attainable by other means. An exhaustive search entails searching through all the $2^k = 2^{nR}$ codewords when the code rate is $R \leq 1/2$, however, when $R > 1/2$, it is preferable to look at the $2^{(n-k)} = 2^{n(1-R)}$ syndromes and perform syndrome-based decoding, which also delivers the ML decoding performance. In short, as it is well known, the complexity of an exhaustive search approach amounts to $2^{n \min(R, 1-R)}$ [118].

It is known that the computational complexity of decoding *any* (deterministic or random) linear block code (LBC) can be greatly reduced by using set information set decoding algorithms (see [118] and references therein), or by having a trellis representation of the LBC and running the Viterbi algorithm on that trellis. The latter technique has been known since [119], when Wolf showed how to construct a trellis representation of a LBC using its parity-check matrix. However, as described in [120], research on trellis decoding of LBCs was not very active until the paper by Forney (and in particular its appendix) came out [121]. In [120], Kschischang and Sorokine provided a (arguably) more elegant manner of constructing the trellis of a LBC by using the spans of the so-called atomic codewords, which is a tool to find the *unique* minimum trellis of a LBC. A measurement of the complexity of the resulting trellis is the maximum number of states existing across all its sections, and that number can be shown to be $s_{\max} = 2^{\min(k, n-k)}$. This corresponds to the bound given by Wolf [119] and also found in [122, 120]. Forney's work was, in fact, much broader, unlocking the trellis representation of lattice codes (of which LBCs are a particular case), and therefore permitting ML lattice decoding of specially designed lattices [123], and also approximate ML decoding of random lattices [124].

The path taken by coding theory (and coding practice) in the following years focused on finding practical codes that would achieve Shannon's capacity; regardless of the family of codes being studied, the efforts were concentrated on long codes (with large n) and typically not having a very high rate. This may explain why the work on trellis decoding of LBCs (summarized in [125, 126]), as well as the various information set decoding algorithms, have been overlooked as a possible low-complexity ML decoding solution for RLCs. All these decoding approaches deal with lists of size $2^{(n-k)}$: being it the number of syndromes, or the number of coset representatives described by the trellis of the code. While syndrome decoding would involve storing all those syndrome pairs and the associated coset representative in the so-called *standard array*, a ML Viterbi trellis decoder implies storing a trellis with the same number of paths (each of which associated with a coset representative). When the redundancy of the code (i.e., $n - k$) is large, the memory requirements for syndrome decoding become too large, and in the case of trellis decoding, the trellis comprises too many states, in both cases limiting the real-world use of capacity-achieving RLCs.

Motivated by both power- and complexity-limited devices in wireless communications, and also due to the goal of reducing end-to-end coding/encoding latency, research only recently shifted to the finite-blocklength, with an emerging interest for short codes and high rates, eventually culminating in a practical method for decoding RLCs dubbed guessing random additive noise decoding (GRAND) algorithms [127, 128, 129, 130, 131, 132]. At its core, GRAND is focused on guessing the noise that corrupted the transmitted codeword, rather than exhaustively going through all the possible codewords in order to find the one that fits the ML criterion, and is proven to still lead to ML decoding [127]. GRAND is a *universal* decoder, enabling the decoding of *any* block code of moderate length or high code rate, when dealing with low entropy noise. The only requirement is that a practical membership test exists to assess whether some word belongs to the codebook or not. The decoding complexity is measured by the average number of membership tests needed until a valid codeword is found. Given that the space of possible words has size 2^n , and that there are 2^k valid codewords, that average number of attempts until one finds a codeword is given by the ratio $\frac{2^n}{2^k} = 2^{n-k} = 2^{n(1-R)}$. Therefore, that number can be low, provided that the codeword are short or that the code rate is high, which is particularly attractive for quantum setups, where the number of available qubits is modest.

Although ML decoding could be implemented for the particular case of low-redundancy RLCs via the above-listed techniques, one would still face memory constraints. Contrary to information set decoding, syndrome decoding, or trellis-based decoding, GRAND does the decoding on-the-fly, dispensing any type of storage besides the parity-check matrix of the RLC (or any particular LBC). Note that computing a minimal trellis of an RLC is a non-negligible preprocessing that is needed each time the RLC changes. Additionally, GRAND can outperform those techniques (designed for memoryless channels) in the case of non-independent and non-identically distributed noise having memory [133]. In fact, any prior information about the structure of the noise can lead to performance improvement or to further complexity reduction. For example, incorporating the geometry of the used symbol constellation can significantly reduce GRAND's complexity [134].

B Scope and Contributions

Due to the technical difficulties in manipulating qubits, the error correction codes applied to qubit packets in quantum communication links or quantum memories are necessarily ones with short codewords, and for that reason, quantum error correction is a good fit for GRAND-inspired strategies. Since all quantum operations are required to be unitary, the construction of quantum random linear codes (QRLCs) is not as straightforward as in classical RLCs. This difficulty in adapting classical codes to the quantum setting is typical; it also took an extensive effort to find quantum LDPC codes whose properties matched their classical counterparts, and these issues continue to be the focus of intensive research [135, 136, 137]. Similar issues plagued the development of quantum turbo codes [138]. Furthermore, to borrow the performance guarantees existing in the classical setting, many of these adaptations amount to Calderbank-Shor-Steane (CSS) codes, which have suboptimal channel capacity [139, 140].

Fortunately, for QRLCs, simple (randomized) constructions exist [141, 112], leading to codes whose

performance is primarily conditioned by the number of quantum gates used as building blocks for the coding circuit. For codewords with n qubits, and in a setup with all-to-all connectivity (i.e., when any qubit can be directly entangled with any other), $\mathcal{O}(n \log^2 n)$ two-qubit random Clifford gates and $\mathcal{O}(\log^3(n))$ circuit depth has been shown to lead to codes with a reasonable Hamming distance [141]. More recent results have expanded the capabilities of QRLCs for the case of limited qubit connectivity, and eased the gate requirements [112]. In general, the more gates used, the more the code approaches a “truly” random QRLC. As part of the encoding for stabilizer codes, they may also constitute a vital piece in quantum error correction for quantum computing, quantum networks, and quantum memories. However, these codes have suffered from some of the same disadvantages as classical RLCs, as they have been too difficult to decode in practice. In fact, to the authors’ best knowledge, only recently has been presented in [142] a practical decoding technique, based on tensor networks, applicable specifically to low-depth circuits with one-dimensional connectivity. They numerically verify that their code reaches the hashing bound [143], but do not prove it mathematically. The recent work in [142] does consider a Pauli decoherence channel, however the results are presented using different variables and parameters, difficulting a direct comparison with our proposal. We note that, from a practical perspective, our model has some limitations. We consider all-to-all qubit connectivity, noiseless encoding circuits, as well as a noiseless syndrome extraction procedure. We also consider noiseless quantum gates. These assumptions are regularly used to analyze a code’s performance [144], and they are used similarly in other results in the literature [112, 142, 113]. These limitations are further addressed in Section 3.8.

Interestingly, research on trellis decoding has recently also begun being studied as a possible practical way of implementing ML decoding for any quantum stabilizer code over a finite field of prime dimension [145]. As in the case of the trellis decoding of classical RLCs (mentioned in Section A), a trellis-based approach may also open doors to the decoding of QRLCs.

The present paper extends the noise-decoding concept to quantum systems, allowing one to tackle the decoding of QRLCs for the first time in the typical Pauli decoherence channel. Meanwhile, results on the use of guesswork in the decoding of designed families of quantum codes have appeared in [146]. The concept of universal guessing-based decoding was tested there with non-random codes, i.e., quantum codes that are known and hold a design structure that fits particular decoding methods.

Our work considers a system model with perfect encoding and perfect measurements, focusing on correcting the errors that arise in the communication channel, which is a common model used in the literature [147, 146]. By making use of the key features of GRAND, this work shows how to take advantage of QRLCs to construct stabilizer codes that are shown, via semi-analytical simulation, to be near-capacity achieving and decodable in practice, leading to quantum error correction codes able to cope with large channel error rates, even at reasonably high code rates.

As it is a universal decoding method, the proposed quantum-GRAND (QGRAND) constitutes a general approach to QEC that is amenable to any code length size n and a sufficiently high code rate R . Furthermore, it is highly adaptable to on-the-fly changes depending on noise statistics, and is likely suitable to scenarios where less than all-to-all connectivity exists. For these reasons, the combination of QRLCs

and QGRAND stands out from the well-established QECCs. This combination is potentially of high importance to future implementations of quantum error correction for quantum computing and quantum networks, as recently highlighted in [148, sec. VI.B].

The literature on the use of GRAND to decode classical RLCs is quite recent and it often directly compares the decoding complexity of RLCs via GRAND with the one of ML decoding. The historical context (given in Section A) brought to attention several other approaches that enabled the decoding of RLCs. That account is further complemented in Section 3.2. Besides that, the main contributions of the paper are i) the analysis of the requirements for the construction of a good QRLC, including the implications to the associated encoding complexity, measured by the number of gates in the encoding circuit, and ii) the proposal of a quantum-GRAND approach that enables the numerical assessment of the performance of QRLC's for the first time, and measuring how close those results are from the optimal performance, which is analytically derived.

C Organization and Notation

The paper is organized as follows: Section II presents a brief introduction to classical RLCs, and introduces GRAND in the classical domain. Section III provides some quantum error correction basics, and Section IV details the approach taken by stabilizer codes. Section V presents some theoretical and numerical results (using a semi-analytical framework) for QRLCs, and the noise model considered. Section VI describes how to construct a stabilizer code from this quantum RLC encoding. These ideas are later combined in Section VII, where QGRAND is showcased. The numerical results are displayed in Section VIII, and a final discussion appears in Section IX.

For the sake of clarity, we opted to denote matrices and vectors in non-bold, as that is the traditional notation in the field of quantum signal processing, with the exception of the classical vector $\mathbf{s}$ and the zero vector $\mathbf{0}$.

It should be noted that *error rate* is used throughout the paper in the same sense that *bit error rate* is used in classical communications, meaning that herein the term is used with respect to qubits *after* their discretization. The literature on QECCs often calls it *error probability* instead, and rather associates the error rate with the continuous errors of the *physical* qubits [149].

3.2 Toward Shannon's Codes: Classical RLCs

In 1948, Shannon proved [116] that (classical) random error correcting codes with codewords of length n and having 2^k valid codewords randomly chosen out of the 2^n possible words are able to achieve the capacity of the Gaussian noise channel, as $n \rightarrow \infty$. Even so, randomly selecting the codebook members leads to: i) a storage problem, given that all codewords would have to be stored both at the encoding side and at the decoding side, and ii) a decoding complexity problem, given that, when applying ML decoding, a corrupted codeword needs to be compared with all codewords in the codebook.

The storage problem posed by Shannon's construction (to generate uniform-at-random codes) has been

overcome by RLCs, of rate $R = k/n$, because, as in the case of any LBC, its generator matrix constitutes a very short description of the code. Without the size constraints of many families of classical structured codes, that impose constraints both on the admissible codeword lengths and code rates, RLCs can be constructed with any size and rate, and having those degrees of freedom is a major practical advantage for most engineering applications.

A Decoding Classical RLCs by Noise Guessing

A paradigm change in the decoding of classical codes recently occurred for codes with short codewords or having low redundancy (i.e., high rate). GRAND [127, 129, 130, 131, 132] fundamentally shifts the role of detection from searching *codewords* in the codebook to searching for the *error pattern* that took place.

Let us assume that a received block $Y = X \oplus E_i$ has been affected by some error pattern E_i , which could have occurred with probability p_i . The statistics $\mathcal{N}$ of the possible error patterns are known, and the N candidate error patterns E_i are ordered from the most likely ($i = 1$) to least likely ($i = N$). GRAND consists of subtracting the error pattern E_i from Y , starting from $i = 1$ and progressing down the list in $\mathcal{N}$. If $X = Y \ominus E_i$ is a valid codeword, then X is accepted as the decoded data. Otherwise, it moves to the next error pattern E_{i+1} and the process is repeated. Here, it is important to note that having a tool to test if X is a codeword (i.e., a *membership test* [127]) is absolutely necessary. If our noise statistics $\mathcal{N}$ encompasses all possible error patterns, it is guaranteed that a valid codeword X is found at some iteration. As proven in [127], for noise with Shannon entropy $H(\mathcal{N})$ (or entropy per bit $h(\mathcal{N}) \triangleq H(\mathcal{N})/n$) and a code rate $R = k/n$ below capacity (i.e., $R < 1 - h(\mathcal{N})$), the expected number of error patterns needed to be tested until finding the true noise pattern is $2^{n \min\{h_{1/2}(\mathcal{N}), 1-R\}}$, for large n , where $h_{1/2}$ is the Renyi entropy of order $1/2$ [150], per bit. GRAND works best when the noise entropy is low or the code rate is high. In that case, the set of 2^n words is densely populated by codewords, and therefore the number of tests can easily be much lower than 2^{nR} , the expected number of tested patterns for standard ML decoding. Given the focus on decoding the noise, exploiting the prior knowledge of the noise statistics plays a central role in boosting the overall decoding performance of GRAND [133].

GRAND performs ML decoding and can therefore provide capacity-achieving performance when decoding structured capacity-achieving codes or RLCs. For practical reasons, one may be interested in the most likely errors only, and the search may be abandoned after iterating through slightly more than the first $2^{nh(\mathcal{N})}$ error patterns, and quit afterward. This variant is called GRAND with abandonment (GRANDAB) and still provides ML decoding. In this manner, GRANDAB is also capacity-achieving, using many fewer iterations.

Although these results have been rigorously proven for uniform-at-random codes as $n \rightarrow \infty$, experimental applications using structured linear codes of small blocklength, and also RLCs in particular, have produced striking results [151, 152, 153, 154, 133, 128], consistent with these theoretical guarantees.

It should be credited that very related ideas appear in [155, pp. 220-227], namely i) codeword guessing

decoding based on the posterior probability of the noise, and ii) proving Shannon's capacity theorem for RLCs via counting arguments based on the number of distinct syndromes.

3.3 Quantum error correction

Quantum error correction attempts to restore a decohered quantum state, which has undergone some error, back to a quantum state encoding the original data [100, 101]. Qubits can undergo more types of errors than their classical counterparts, making QECCs in general more complex. Moreover, while classical use cases, such as wired communications or data storage, commonly offer error rates below 10^{-5} , current quantum technologies need to be able to handle much higher error rates, similar to the ones encountered in uncoded classical wireless links (between 10^{-3} and 10^{-2}) [108, 109]. In addition to that, the construction and decoding of QECCs faces extra difficulties [156]:

1. the non-cloning theorem shows it is impossible to copy an arbitrary quantum state, thereby forbidding the use of copies as redundancy;
2. quantum errors are continuous in nature, as the quantum state can be in any superposition of the basis states;
3. any measurement performed on a state in superposition destroys that superposition, and thus its quantum information is lost.

Surprisingly, it is possible to circumvent these issues and construct QECCs that reliably protect quantum data against errors. Just as in the classical case, QECCs work by encoding a quantum state composed of k qubits into one with n qubits, in such a way that errors can be detected and corrected.

Any quantum channel error E , discrete or continuous, acting on a quantum state, can be written as a linear combination of these Pauli matrices. For one qubit only, one has

$$E = \alpha_0 I + \alpha_1 X + \alpha_2 Z + \alpha_3 Y, \quad (3.1)$$

for some set of complex values α_i . I is a 2×2 identity matrix and

$$X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \quad Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}, \quad Y = iXZ, \quad (3.2)$$

are Pauli matrices. Since $Y = iXZ$, as long as QECCs can correct X (bit-flip) and Z (phase-flip) errors simultaneously and for each qubit, then they can also correct any arbitrary n -qubit error.

From this digitization, it can be shown [156] that a quantum error channel for n qubits may be decomposed as a combination of Pauli strings

$$P_n = e^{i\frac{\pi}{2}\theta} O_1 \otimes \dots \otimes O_n, \\ \text{with } \theta \in \{0, 1, 2, 3\}, \quad O_i \in \{I, X, Y, Z\}. \quad (3.3)$$

and $\otimes$ denotes the tensorial (or Kronecker) product. This description is named Pauli channel.

The Clifford group is the set of unitary transformations that transform any Pauli string into a Pauli string. Clifford circuits are built solely from unitaries of the Clifford group. Any such Clifford unitary can be implemented via composition of the simple 1- and 2-qubit unitaries

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \sqrt{Z} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \text{CNOT} = \begin{pmatrix} I & 0 \\ 0 & X \end{pmatrix}. \quad (3.4)$$

With these Clifford circuits, it is possible to construct an encoding for a QECC, in order to protect the desired quantum data. The backbone and inspiration for most quantum error correction methods are the so-called stabilizer codes [157], and for that reason, this work will chiefly focus on these codes as the core machinery needed to produce random QECCs.

3.4 Stabilizer codes

A stabilizer S_i of the quantum state $|\psi\rangle$ is any Pauli string that acts as the identity on $|\psi\rangle$, that is, $S_i |\psi\rangle = |\psi\rangle$. The stabilizer group $\mathcal{S}$ is the set of all such stabilizers. As there is a one-to-one correspondence between $|\psi\rangle$ and its stabilizer group, stabilizer circuits can be efficiently simulated using only Clifford unitaries (see [158, 159]). The stabilizer group can be generated by some (non-unique) subset $\mathcal{S}_{\min}$, whose elements are the minimal stabilizers. These minimal stabilizers can be used to create a type of QECC, called a *stabilizer code* (see Fig. 3.1). For simplicity, henceforth we will refer to the minimal stabilizers of stabilizer codes simply as stabilizers.

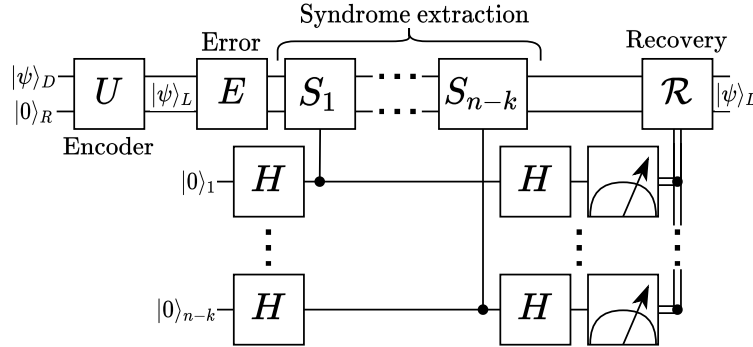


Figure 3.1: Quantum circuit for a (n, k) stabilizer code, depicting the encoding and decoding process.

Starting with the k -qubit quantum data $|\psi\rangle_D$ and $(n-k)$ redundancy qubits in the state $|0 \dots 0\rangle_R$, which have been entangled by some Clifford unitary encoder U , then the resulting n -qubit logical state $|\psi\rangle_L$ (amounting to a codeword) must have $s \triangleq (n-k)$ stabilizers $\mathcal{S}_{\min}$, which must commute with each other. It also has $2k$ logical operators $\bar{X}_i, \bar{Z}_i$, for $1 \leq i \leq k$, which act on $|\psi\rangle_L$ similarly to X_i, Z_i on $|\psi\rangle_D$, and which can be combined to form a total of $L \triangleq 2^{2k}$ logical operators.

Suppose that a Pauli error string E affects $|\psi\rangle_L$, resulting in $|\tilde{\psi}\rangle_L = E|\psi\rangle_L$. If we conditionally apply

a stabilizer to $|\tilde{\psi}\rangle_L$, controlled by an ancilla qubit in the state $|+\rangle = H|0\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$, and subsequently apply the Hadamard gate H a second time (see (3.4) (3.4)), we obtain

$$E|\psi\rangle_L \frac{|0\rangle + |1\rangle}{\sqrt{2}} \quad (3.5)$$

$$\xrightarrow{S_i} \frac{1}{\sqrt{2}} [E|\psi\rangle_L |0\rangle + S_i E|\psi\rangle_L |1\rangle] \quad (3.6)$$

$$\xrightarrow{H} \frac{1}{2} [(I + S_i)E|\psi\rangle_L |0\rangle + (I - S_i)E|\psi\rangle_L |1\rangle]. \quad (3.7)$$

Since both the error patterns and the stabilizers are Pauli strings, they must necessarily commute or anticommute. If they commute, the state in (3.7) will be $E|\psi\rangle_L |0\rangle$, otherwise, we will have $E|\psi\rangle_L |1\rangle$. Consequently, the conditional application of each stabilizer provides us with 1 bit of information about the nature of the error. With s stabilizers, we have a s -bit syndrome, allowing one to discern between $S \triangleq 2^s$ error patterns, while in the case of no error, one obtains the zero syndrome. This information enables us to correct the quantum state by undoing the effect of the error. This recovery process $\mathcal{R}$ will consist of applying the inverse of the error pattern E ascertained from the syndrome. Since all Pauli strings are both Hermitian and unitary, for a Pauli channel, we have $E^{-1} = E$.

3.5 Quantum RLCs encoding

The construction of classical RLCs mentioned in Section 3.2 involves random generator matrices, whose only constraint is that they ought to be full-rank. Because quantum operators should be unitary, the construction of QRLCs is a rather more constrained problem. The generation of QRLCs is nevertheless possible: starting with a quantum state of k qubits, to be encoded into $n > k$ qubits, [141] presents a method of generating a random qubit encoding. One starts by randomly selecting Clifford unitaries from the $\mathcal{C}_2$ group (i.e., Clifford unitaries for 2 qubits). There are $|\mathcal{C}_2| = 11\,520$ such unitaries, and all of them can be built by simple combinations of the Hadamard, CNOT, and phase gates, which have efficient physical implementations in virtually any quantum setting. After selecting these random unitaries from $\mathcal{C}_2$, one successively applies each of them to a random pair of qubits, taken from the set of n qubits, assuming that all-to-all connectivity (between any of the n qubits) is possible in practice. This assumption is required for the scaling results in [141], and is used in this work. However, it may be dropped for practical reasons, as the more recent results in [112] suggest, and it is the subject of future work. This process leads to an encoding unitary for our stabilizer code which, when applied to the initial k qubits and $(n - k)$ extra $|0\rangle$ qubits added, returns a n -qubit encoded quantum state. As shown in [141], as long as $\mathcal{O}(n \log^2 n)$ gates are used, with a circuit depth of $\mathcal{O}(\log^3 n)$, the construction leads to a highly performant (n, k) code, and from [112] it is already known that these complexity orders can be further lowered.

A Robust encoding

In this context, the QRLCs' performance will also depend on the noise statistics. Nonetheless, in order to effectively protect our quantum data from noise in an efficient and practical manner, the noise statistics about the particular environment in question should be known. In the quantum setting, assuming we wish to correct the N error patterns in the error set

$$\mathcal{E} \triangleq \{E_i\}_{i=0}^N, \text{ with } E_0 \triangleq I, \quad (3.8)$$

and we know the probability of them affecting our data to be $\mathcal{P} \triangleq \{p_i\}_{i=0}^N$, with $p_0 \triangleq P(\text{no error})$, then our encoding and decoding procedure should have into account the noise statistics, denoted by the set of pairs

$$\mathcal{N} \triangleq \{(p_i, E_i)\}_{i=0}^N, \quad (3.9)$$

with the error patterns ordered by decreasing probability, so that $p_i \geq p_{i+1}$, for all i . For an encoding with n qubits, using GRAND requires $N + 1 = 4^n$, with $\sum_i p_i = 1$, as 4^n is the total number of Pauli strings of n qubits. If using GRANDAB, the unlikely errors may be disregarded, so a much lower N , and $\sum_i p_i \leq 1$, is assumed.

As the encoding is randomly chosen, to ascertain the code's performance, we may compute the syndromes associated with every error pattern in $\mathcal{E}$, and check whether the code is robust to all of them, in $\mathcal{O}(N)$ time. The code is robust if each $E_i \in \mathcal{E}$ has a unique syndrome or, for *degenerate codes*, if E_i shares a syndrome with E_j and $E_i E_j \in \mathcal{S}$, that is, the combined action of the two errors does not change the quantum state.

Nonetheless, we may not require that all errors in $\mathcal{E}$ be correctable. For low entropy noise, some errors may be very unlikely to surface. If a random code happens to not correct these unlikely errors, it may still function adequately in the high probability case that more likely errors occur.

Classically, for a channel with input X and output Y , its capacity is given by the maximum of the mutual information $I(X, Y)$ over the possible probability distributions of X . In the quantum case, for the case of stabilizer codes, the quantum capacity is given by the coherent information of the quantum channel [156, sec. 7.6], [143, sec. 24.6.3]. For a Pauli channel, this capacity is associated with the so-called hashing bound [143, sec. 24.6.3], [160]. Using the results in [143, sec. 24.6.3] we observe that, as long as

$$n > k + H(\mathcal{N}), \quad (3.10)$$

the code is able to correct the error patterns in $\mathcal{N}$ with high probability. This is a direct consequence of information-theoretic arguments based on set counting, and interpreting the entropy of the noise, $H(\mathcal{N})$, as the number of bits needed to describe the noise, which establishes how many different noise patterns can occur. Note that, in theory, due to the possibility of degenerate error patterns, it is sometimes possible to surpass the hashing bound [161, 162]. Nonetheless, in Section B we motivate why degenerate errors have negligible effects in our setting, and consequently, in our numerical analysis in Section 3.7, these

error patterns are approximately considered to be uncorrectable. Therefore, to obtain a near-capacity achieving code, we must only ensure that (3.10) is satisfied, and that sufficient 2-qubit gates are used to create the random encoding, as indicated previously, and as will be analyzed in Section C. This result is derived for large n , but similar results are applicable to lower n , as we see in the next section.

B Comparison with ideal random codes

In order to assess the results obtained numerically, we may compare the QRLCs' performance with the expected performance of a simpler approximation. Similar techniques [112] have been previously employed in the literature, since they allow us to better gauge the behavior of QRLCs, as it is too difficult to analytically estimate their performance.

As an ideal approximation, we consider a code that maps each of N error patterns (plus the no error case) randomly, using an i.i.d. uniform distribution, to one of the S syndromes. This approximation constitutes a more manageable mapping for analytical study, when compared with the mapping produced by a QRLC, since it does not present the linearity constraint, nor the added structure present in quantum codes. Nonetheless, since QRLCs may be thought of as a subset of the codes that this ideal approximation may generate, its performance may be considered to provide an upper bound for the QRLCs of interest.

For N error patterns, plus the no-error case, let u be the number of unique error syndromes associated with them. In general, the average number of unique syndromes is

$$\langle u \rangle_{S,N+1} = S \left[1 - \left(1 - \frac{1}{S} \right)^{N+1} \right], \quad (3.11)$$

and the correctable error fraction is given by

$$f = \frac{u_{S,N+1}}{N+1}. \quad (3.12)$$

If we wish to correct all N error patterns (i.e., $f = 1$), we must have $(N+1)$ distinct error syndromes, which happens with probability

$$P(f = 1) = \prod_{j=0}^N \left(\frac{S-j}{S} \right) \quad (3.13)$$

$$\simeq \prod_{j=0}^N e^{-j/S} = \exp \left(-\frac{N(N+1)}{2S} \right), \quad (3.14)$$

with the approximation achieved by the Taylor expansion of e^x . Note that the probability of correcting absolutely all N error patterns $P(f = 1)$ may be very low, while still having $P(f > 1 - \epsilon) \simeq 1$, for $\epsilon \ll 1$. In fact, reminding that $S = 2^{n-k}$, we have

$$P(f = 1) \geq 1 - \epsilon \iff n \gtrsim k + \log_2 \left(\frac{N(N+1)}{2\epsilon} \right) \quad (3.15)$$

$$f \geq 1 - \epsilon \iff n \gtrsim k + \log_2 \left(\frac{N}{2\epsilon} \right). \quad (3.16)$$

These equations directly showcase the relation between the choice of n for the encoding and the code's performance. Unlike (3.10), their derivation does not assume large n .

For (n, k) QRLCs, each of the possible 4^n Pauli strings will be associated with one of the $S = 2^{n-k}$ syndromes. Due to the linearity of the code, the strings will be distributed equally through the syndromes, so that each syndrome will have $2^{n+k} = SL$ associated Pauli strings. In fact, for $\mathbf{s} = \mathbf{0}$, the associated strings will amount to all the combinations of the S total stabilizers and the $L = 2^{2k}$ total logical operators.

When creating a QRLC, it is possible to obtain a degenerate code. For an error pattern $E_i \in \mathcal{E}$, we obtain a degenerate code when $\exists i, j$ such that $i \neq j$ and $E_i E_j \in \mathcal{S}$. In these situations, E_i and E_j have the same effect on the encoded state and, consequently, can be treated as the same error. For two random error patterns E_1, E_2 with the same syndrome, the probability of $E_1 E_2$ being a stabilizer is $\sim 1/L$.

For simplicity, we consider a code that can fully correct the error patterns in $\mathcal{E}$ a (good) degenerate code, while making no distinction between faulty codes that do or do not present some level of degeneracy.

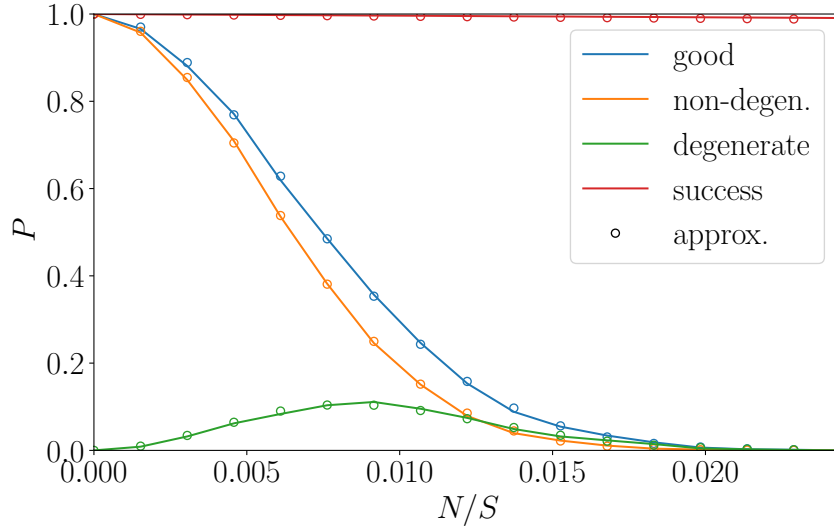


Figure 3.2: Probability of obtaining a “good” code (blue), that corrects all N error patterns, vs. the ratio N/S , for a $(16, 1)$ code, for the maximum noise entropy case. Even if obtaining a good code is unlikely, it still corrects almost all errors, so the probability of a successful correction is high (red). The dots indicate the theoretical approximations given by (3.20), (3.13), (3.21), and (3.12), respectively, which closely match the observed behavior.

The probability of obtaining a “good” code, i.e., a code that is capable of correcting all the N error patterns, can be estimated by considering the probability that either the syndrome of $E_i E_j$ differs from $\mathbf{0}$ (non-degenerate case) or equals a stabilizer (degenerate case). As there are $\binom{N+1}{2}$ such $E_i E_j$, this probability is given approximately by

$$P(\text{good}) \simeq (P(\text{non-degen.}) + P(\text{degen.})) \binom{N+1}{2} \quad (3.17)$$

$$\simeq \left[\frac{S-1}{S} + \frac{S}{4^n} \right]^{\binom{N+1}{2}} \quad (3.18)$$

$$= \left[1 - \frac{1}{S} \left(1 - \frac{1}{L} \right) \right]^{\binom{N+1}{2}} \quad (3.19)$$

$$\simeq \exp \left(-\frac{N(N+1)}{2S} \left(1 - \frac{1}{L} \right) \right). \quad (3.20)$$

As the probability of obtaining a working non-degenerate code is approximately given by (3.13), the probability of obtaining a working degenerate code can be estimated by

$$P(\text{degenerate}) = P(\text{good}) - P(\text{non-degenerate}). \quad (3.21)$$

Fig. 3.2 presents 2000 samples of $(16, 1)$ random codes. Most of the “good” codes are non-degenerate, so every error pattern has its distinct syndrome, but a small percentage may be degenerate, since $k = 1$ is very low. As expected, the probability of obtaining a code that corrects all N error patterns present quickly decreases with increasing N . Nonetheless, the probability that faulty codes will succeed in correcting any particular error is still high for larger N (red). For the highest shown N/S , with $N/S = 0.025$, $P(\text{success}) = 0.99$, while $P(\text{good}) \simeq 0$. As noted previously, this large discrepancy between $P(\text{success})$ and $P(\text{good})$ stems from the fact that we may have $P(f > 1 - \epsilon) \simeq 1$ while also having $P(f = 1) \simeq 0$.

Bernoulli process

As an example, we may consider the noise model in which quantum errors consist of a Bernoulli process, with each individual qubit having a probability p of suffering an error. The error can be of type X, Y , or Z (i.e., depolarizing noise), and the error types are identical and independently distributed (i.i.d.) for each qubit.

The number of distinct error patterns of weight $t' \leq t$ is

$$B_t \triangleq \sum_{t'=0}^t A_{t'}, \quad \text{with } A_t \triangleq 3^t \binom{n}{t}. \quad (3.22)$$

For $t' < t$, as error patterns of weight t' are always more likely than error patterns of weight t , for this noise model, then whenever an error pattern of weight t has the same syndrome as a weight- t' error pattern, it cannot be corrected. Here, the degenerate scenarios are disregarded, since they are negligible for large k . The expected number of syndromes M_t with a weight- t error pattern as the most likely one mapped to them is $M_t = \langle u \rangle_{S, B_t} - \langle u \rangle_{S, B_{t-1}}$. The correctable error fraction f of weight- t error patterns that can be corrected will be, on average,

$$f = \frac{M_t}{A_t} \simeq \frac{S}{A_t} e^{-\frac{B_{t-1}}{S}} \left(1 - e^{-\frac{A_t}{S}} \right), \text{ for } S \gg 1. \quad (3.23)$$

This expression enables a useful comparison for the performance of QRLCs, as shown in Fig. 3.3 (and

later in figures 3.5 and 3.6).

C Minimum number of random gates

It is important to assess how deep (i.e., number of concatenated gates) the generator circuits need to be such that the QRLC they generate approximates the ideal performance. In order to do so, we have implemented the methodology in [141] and, for a varying number of gates, we measured the relative deviation

$$\delta_f \triangleq (f_{\text{theory}} - f_{\text{exp}})/f_{\text{theory}}, \quad (3.24)$$

$$\delta_P \triangleq P(f=1)_{\text{theory}} - P(f=1)_{\text{exp}}, \quad (3.25)$$

where the f stands for the fraction of weight- t error patterns a code can correct. f_{theory} amounts to the average correctable fraction coming from the approximation in (3.12), and f_{exp} is the result of averaging the observed correctable fraction of 31 QRLC samples, for a given number of gates. Similarly, $P(f=1)_{\text{theory}}$ is given by (3.13), and $P(f=1)_{\text{exp}}$ is the ratio of the 31 samples that had $f=1$ for weight $t=1$. For the noise statistics, we assume a Bernoulli process, as previously described in Section B.

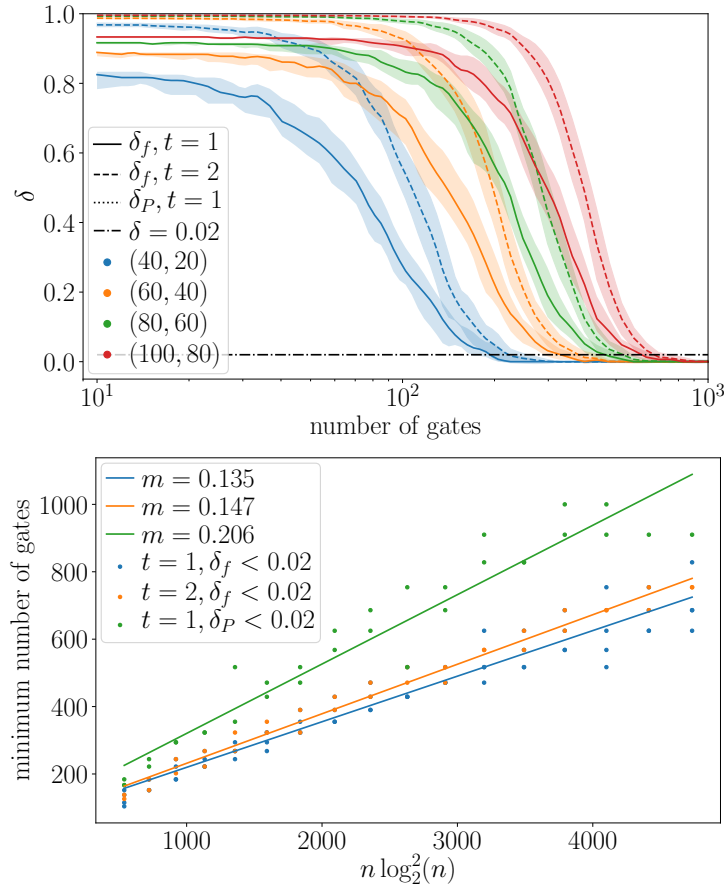


Figure 3.3: (a) Relative deviation δ , for different (n, k) codes and error patterns of weight t . The shaded region indicates the 80% confidence interval, obtained by constructing 31 random codes for each case presented. (b) Minimum number of gates to achieve a relative deviation $\delta < 0.02$, for varying $n \log_2^2(n)$.

The simulation results presented in Fig. 3.3 indicate that, as the number of gates used for the encoding increases, so do the codes' correction capabilities, eventually reaching ($\delta < 0.02$) those of the ideal random code. The codes' error correction capabilities start off worse for error patterns of larger weight, but converge to a similar minimum number of gates needed. We considered codes for several n and k to estimate this behavior. These results, shown in Fig. 3.3, indicate that this minimum number of gates is given by $N_{\text{min. gates}} = mn \log_2^2(n)$. From [141], we expect the minimum number of gates to scale with $\mathcal{O}(n \log^2(n))$. Although this theoretical result is only proved there for some codes, here we experimentally observe it for the mean (n, k) code (with $m \simeq 0.13 - 0.15$), which is of more practical relevance, and for the case where we require that all $t = 1$ error patterns must be correctable ($m \simeq 0.21$). Consequently, even for larger quantum codes with $n = 128, 1000$ $\mathcal{C}_2$ Clifford gates should suffice to ensure that the resulting code has a very high performance for that rate and code length. Given that most of these gates will be applied in parallel, with depth $\mathcal{O}(\log^3 n)$, this approach enables the creation of high-capacity codes with low-depth quantum circuits.

D Determining the stabilizers

The traditional approach in the construction of stabilizer codes is to start by defining the stabilizers, and then find the associated encoding process. The proposal in this paper uses a random code construction, and therefore the first step is the definition of the encoding circuit and only after one determines the stabilizers associated with that particular code.

The construction of a QRLC only involves Clifford unitaries, and therefore the encoding process U is a stabilizer circuit. The Gottesman-Knill theorem indicates that stabilizer circuits with n qubits can be simulated using only $\mathcal{O}(\text{poly}(n))$ classical resources (see [158, 159]). From this efficient classical simulation, the desired stabilizers may be extracted [112]. In particular, the $(n - k)$ ancilla qubits used for the encoding, in state $|0\rangle$, start by having the associated minimal stabilizers Z_i , for $k + 1 \leq i \leq n$, since $Z_i |0\rangle_i = |0\rangle_i$. The classical simulation takes these starting stabilizers and modifies them through the encoding U , resulting in the desired minimal stabilizers S_i , which can be used for the stabilizer code.

3.6 Quantum-GRAND proposals

We are now ready to construct GRAND-inspired decoding techniques to decode QRLCs. This common framework of quantum-GRAND (QGRAND), in the case of stabilizer codes, consists of the steps herein summarized:

1. Generate a random QRLC encoding U , robust to the noise statistics $\mathcal{N}$. This can be efficiently done, for instance, by combining random 2-qubit Clifford unitaries (Sections A and C);
2. Determine stabilizers of the chosen encoding by efficiently simulating the stabilizer circuit (Section D);
3. Perform syndrome measurement using all the minimal stabilizers (Section 3.4);

4. Apply decoding using an iterative noise guessing recovery procedure, similar to the one in classical GRAND (Section A), by applying the inverse of the error pattern determined to the encoded quantum data.

These steps can be implemented in slightly different manners, as will be described in the following subsections.

A The syndrome as membership test

Syndrome measurement setups are a central building block in QEEC [147, 136, 137]. As in classical GRAND, the syndrome measurement step of the stabilizer code may be considered simply as a membership test, accepting the quantum state if the syndrome $\mathbf{s}$ is zero, and rejecting the error-affected state if $\mathbf{s} \neq \mathbf{0}$ (see Fig. A). If the syndromes associated with each error pattern are not known *a priori*, then it will be necessary to measure all $s = n - k$ minimal stabilizers to verify if no error has occurred. Since the encoding and subsequent stabilizers are random, it will take only ~ 2 stabilizer measurements, on average, to obtain a nonzero syndrome bit when an error actually occurred.

For a (n, k) code, we do not necessarily require the measurement of the full syndrome to perform the GRAND iteration procedure. If we are not taking advantage of the information encoded in the syndrome, and wish to use it only as a membership test for the codebook, then we just need to check whether or not the syndrome is nonzero.

As the chosen encoding is random, so are its stabilizers. Consequently, each error pattern has a probability of $p = 1/2$ of anticommuting (and thereby being detected) by each stabilizer, and this behavior is independent for each error pattern. Therefore, the average number of stabilizers measured until encountering a 1 bit, for some error pattern, is

$$\langle C_s(p) \rangle = \sum_{i=1}^s i(1-p)^{i-1}p \quad (3.26)$$

$$\Rightarrow \langle C_s \rangle = \langle C_s(1/2) \rangle = 2 - \frac{s+2}{2^s} < 2. \quad (3.27)$$

As we generally don't know *a priori* whether or not an error has occurred at the first iteration, before the trial-and-error correction process commences, there is the possibility that we will need to measure all s minimal stabilizers, in the case with probability p_0 where no error has occurred. In that case, the average number of stabilizer measurements required to determine whether an error has occurred or not will be

$$\langle C_s(p_0) \rangle = p_0 s + (1 - p_0) \langle C_s \rangle. \quad (3.28)$$

If an error is observed, then at each iteration $i > 0$ of the trial-and-error correction attempt, the average number of measurements is $C_s(p_i/q_i)$, with $q_i \triangleq 1 - \sum_{j=0}^{i-1} p_j$.

The average total number of iterations in the trial-and-error procedure, for N error patterns, is $I \triangleq$

$\sum_{i=0}^N (i+1)p_i$. Consequently, the average total number of stabilizer measurements until the procedure is completed is

$$\langle C \rangle = s + (I - 1) \langle C_s \rangle < s - 2 + 2I. \quad (3.29)$$

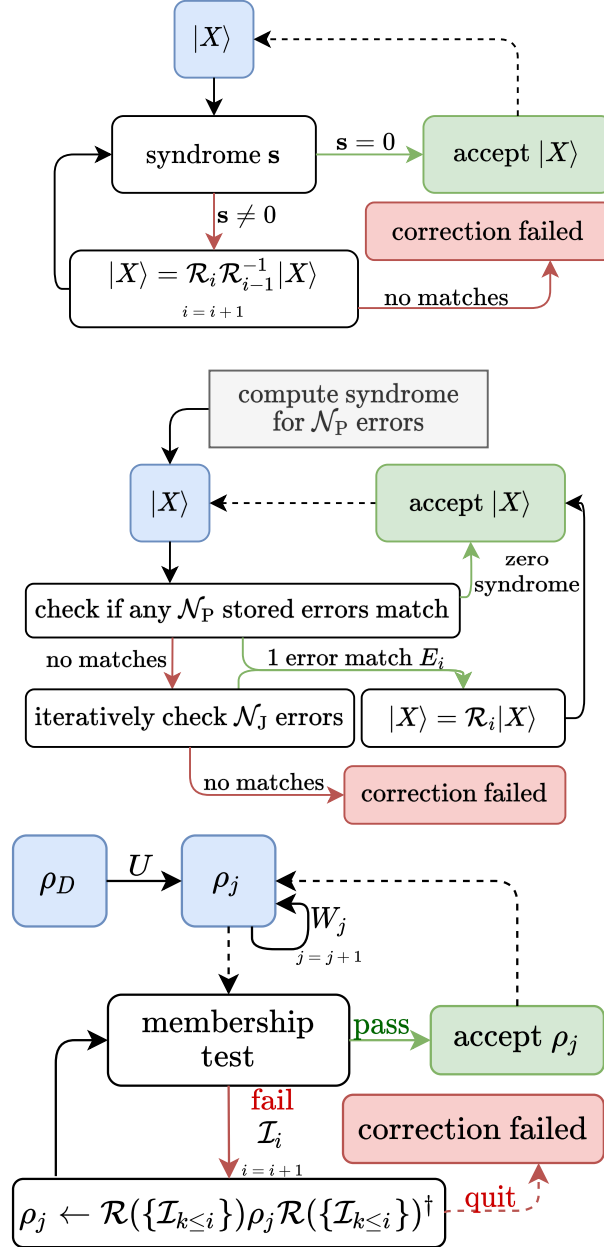


Figure 3.4: Flowchart of QGRAND, when (a) using the syndrome solely as a membership test, or (b) also as an aid for decoding. Going beyond stabilizer codes, (c) QGRAND may also be employed in a more general context.

A downside of this approach is that the error correction and syndrome extraction must be applied physically in the quantum circuit at each iteration. As this correction consists of applying $E_i^\dagger = E_i$ to

the circuit, and the syndrome extraction step might have a non-negligible number of gates (see Fig. 3.1), then, for error-prone implementations, this approach may lead to a costly error correction process, as each application of correction gates adds error itself to the encoded state.

For the case without abandonment, the error correction process can fail solely when an observed error pattern has the same syndrome as an error pattern that was more likely to occur, given the noise statistics. This amounts to the scenario where the coset leader chosen for a syndrome is not the error that occurred. As previously alluded to in Section A, when abandonment is implemented, a second source of failure is the probability that an unlikely error occurs, beyond the errors that one wishes to correct. Nonetheless, as indicated in Section A, n should be chosen to ensure that this probability of failure is low.

B Syndrome decoding

Alternatively, as long as partial or full access to the actual syndrome is available, one may use that information to improve the error correction process, while still taking advantage of the known noise statistics (see Fig. A). There are several possible approaches along this route. For instance, one may rely on the classical computation of the syndromes associated with each relevant error pattern, both before and during the correction process.

To take advantage of syndrome decoding, let $\mathcal{N}_P \subseteq \mathcal{N}$ be the set of error patterns for which we wish to precompute and store the error syndrome, and $\mathcal{N}_J = \mathcal{N} \setminus \mathcal{N}_P$ the error patterns for which we wish to compute the error syndrome on the fly, as necessary. We have $N = N_P + N_J$, with $N_P \triangleq |\mathcal{N}_P|$, $N_J \triangleq |\mathcal{N}_J|$. Its $N_P \times s$ syndrome matrix M_S , which stores the N_P syndromes, each s -bits long, can be computed by the matrix product of the $N_P \times 2n$ error matrix M_E and the $S \times 2n$ parity check matrix A , through $M_S = M_E A^T$ [156]. If N_P is small, this syndrome matrix can be classically stored in memory.

Additionally, the precomputation of the $\mathcal{N}_P$ error patterns' syndromes enables checking *a priori* whether or not our random linear code can distinguish between all the $\mathcal{N}_P$ error patterns, lest it have more than one relevant error pattern with the same syndrome (cf. Section B). Therefore, we may try other random encodings until we are sure that one satisfies the requirements at hand, instead of relying on (3.10).

Given the random nature of the linear code, it takes, on average, $\log_2 N_P$ bits to identify each of the $\mathcal{N}_P$ error patterns, for a total of at most $\sim 2N_P$ bits to store, using a binary tree. To single out a particular error pattern in $\mathcal{N}$, we require, on average, $\log_2(N + 1)$ stabilizer measurements. Note that $\log_2(N + 1)$ may be much less than s bits in practice (see (3.10)). After $\log_2(N_P + 1)$ stabilizer measurements, we are likely to rule out almost all error patterns in $\mathcal{N}_P$. However, the error patterns in $\mathcal{N}_J$ have not yet been ruled out. To be sure of which error actually occurred, $\log_2(N_P + N_J + 1)$ measurements are needed. For that, the syndromes of the other $\mathcal{N}_J$ error patterns needed to be checked. These are error patterns that we still wish to correct, but whose syndrome we did not keep in memory due to memory constraints. These computations can be performed in parallel, and we only need to compare, and thus compute, on average, < 2 bits among the measured syndrome bits, to check whether one of these error patterns is the measured one.

In the case with abandonment, it is possible that none of the $\mathcal{N}_P$ or $\mathcal{N}_J$ error patterns explain the observed syndrome, if the error that occurred was very unlikely and not worth correcting.

C Ordering the stabilizers

With direct access to the syndrome information, it might be tempting to order the stabilizer measurements so that, at each iteration, there is maximum information gain. However, it can be shown that this costly computation results in negligible savings in the average number of measurements that need to be performed to determine the error (see Section A), when compared with a random ordering. Similarly, ordering the stabilizers to try and create a more efficient membership test would also result in minimal improvement in performance.

If $s \gg \log_2(N+1) > H$, one may work with a smaller subset of the s minimal stabilizers, since, on average, we only require $\log_2(N+1)$ measurements (in the highest entropy case) to determine the error. In this case, it is possible to choose stabilizers that have low weight or that are easier to implement, based on the hardware restrictions. However, this biased sampling may negatively affect performance. In any case, it should be noted that, if the encoding is chosen according to (3.10), then one should not have $s \gg H$.

D Adapting to changing noise statistics

As the set of noise statistics $\mathcal{N}$ takes center stage in the decoding process, QGRAND may be adapted on the fly to suit a changing noise environment, or changing error tolerances. For the latter case, less (more) strict correction expectations can be achieved by discarding (including) some error patterns E_i with low associated p_i . Similarly, for the former case, if $\mathcal{N}$ no longer accurately represents the new noise environment, a new $\mathcal{N}'$ may be considered, resulting in different errors having correction priority.

If changing $\mathcal{N}$ is not sufficient, the code can still be quickly adapted by undoing the old encoding U and implementing a new encoding U' , keeping in mind the expected performance indicated by (3.10), (3.15), and (3.16).

E A general approach: beyond stabilized codes

The QGRAND concept is not limited to stabilizer codes. A general high-level implementation of QGRAND-inspired decoders may be applied to other QECCs, consisting of the following steps, as depicted in Fig. A:

1. **Preprocessing/Encoding:** The initial quantum data, ρ_D , is preprocessed by U , possibly composed by ancilla qubits, unitary transformations, and projective measurements, into its encoded version ρ_j (with $j = 0$). ρ_j enables the desired operations W_j on the quantum data to take place, transforming the state into $\rho_{j>0}$. All of these states ρ_j , $j \geq 0$ should be robust to errors. At desired intermediate steps, the data is verified.
2. **Data verification:** A *membership test* is performed to verify if the encoded state ρ_j has been

corrupted by errors beyond the maximum tolerable error weight - *fail* case - or if it is still in a state acceptable for further operations - *pass* case. Depending on the implementation, the membership test may provide further information $\mathcal{I}_i$ about the failure, facilitating its subsequent correction. Note that the use of the membership test may itself modify ρ_j , as long as this modification simply aids the correction, and does not induce further significant errors in ρ_j .

3. **Iterative correction attempt:** A recovery operation $\mathcal{R}$ is attempted on the state ρ_j . This operation is chosen based on the information $\mathcal{I}_k$ provided by the previous membership tests, and on the noise statistics $\mathcal{N}$ known about the setup *a priori*. This verification should be reversible, so that subsequent iterations may undo the effects of failed correction attempts.
4. **Error correction:** Steps 2 and 3 are repeated either until no feasible further recovery attempts are possible - leading to a failed correction - or until a membership test indicates that ρ_j is now in an acceptable state, and the potential operations W_j can resume.

3.7 Performance vs code rate trade-off

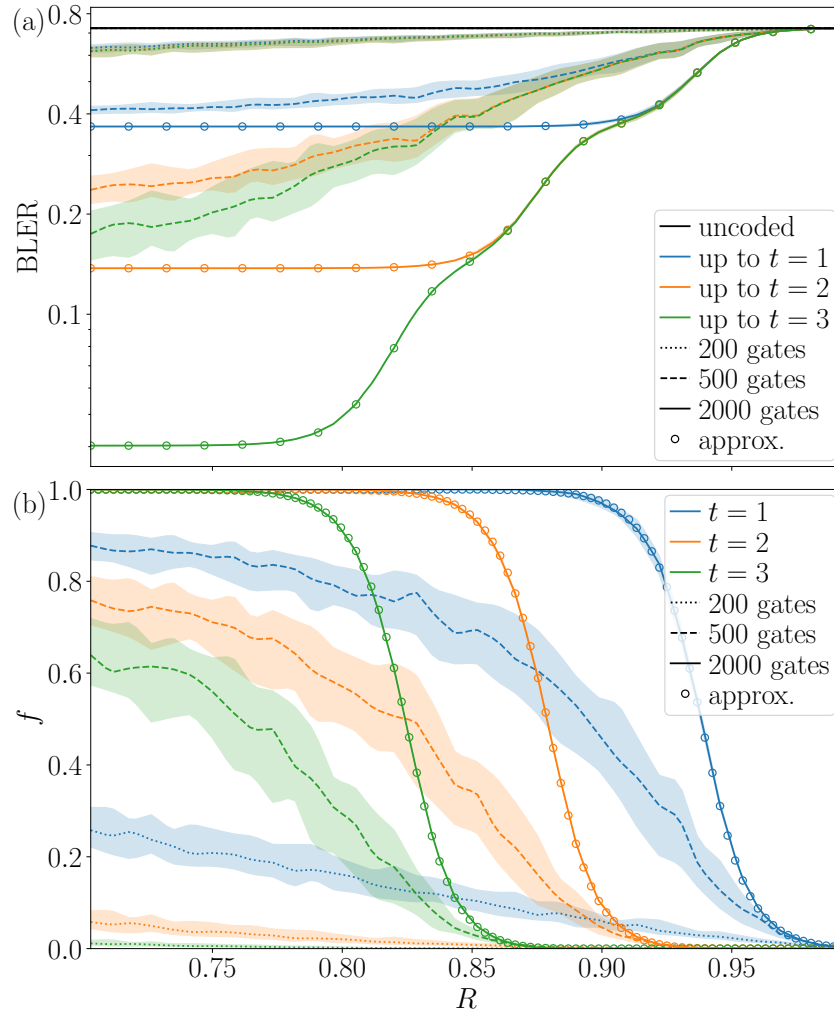


Figure 3.5: BLER and correctable error fraction f for QRLCs with $n = 128$, considering $p = 10^{-2}$, when trying to correct error patterns of weight up to $t = 3$. Top curves: BLER, with shaded areas marking an 80% confidence interval. Bottom curves: fraction f of error patterns of weight t that can be corrected by the code, for a given code rate.

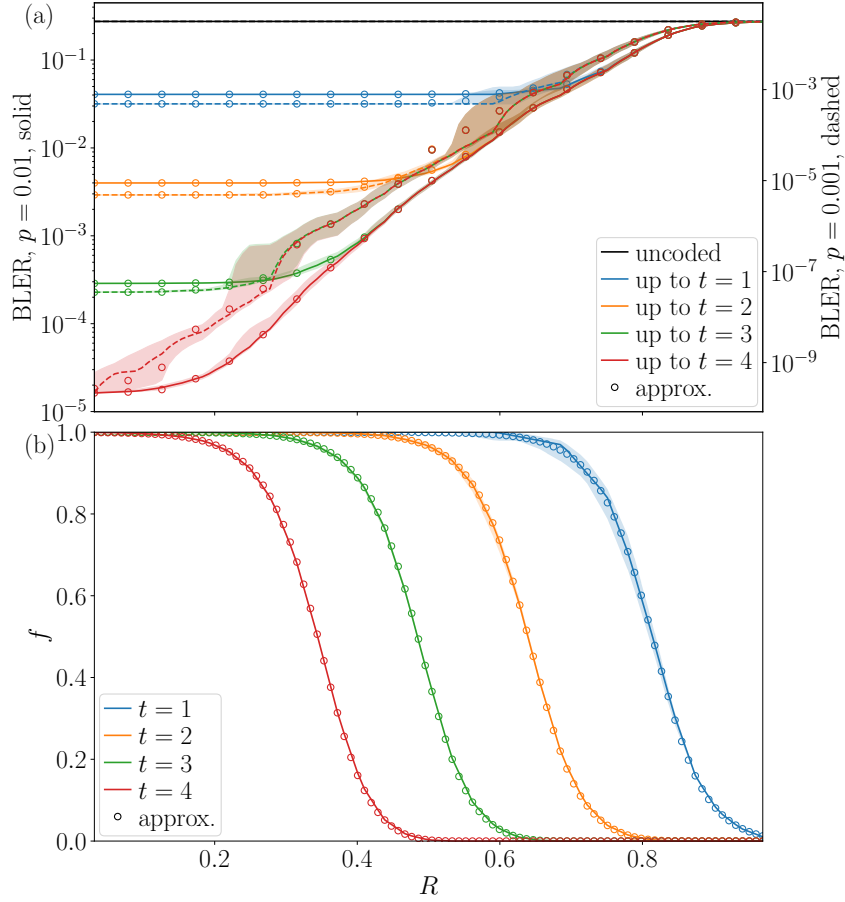


Figure 3.6: BLER and correctable error fraction for QRLCs with $n = 32$, considering different p values, when trying to correct error patterns of weight up to $t = 3$. Top curves: BLER, with an overall behavior similar for both $p = 10^{-2}$ and $p = 10^{-3}$, but the lower p enables 2 to 3 orders of magnitude lower error rate. Bottom curves: fraction f of error patterns of weight t that can be corrected by the code, for a given code rate. This value is independent of p .

As motivated in Section A, the alteration of quantum states follows a noise model where a quantum state σ , in density matrix form, gets transformed into

$$\rho = \sum_i^N p_i E_i \sigma E_i^\dagger, \quad \text{with } E_0 = I. \quad (3.30)$$

After measuring some syndrome $\mathbf{s}$, and applying the associated correction (either directly, if possible, or by iteratively guessing), we obtain the corrected state. Unlike the classical case, where bit-flip errors definitely map a codeword to a different word, in the quantum case the error's actual effect will depend on the original quantum data $|\psi\rangle$, and it may go from doing nothing to mapping $|\psi\rangle$ to an orthogonal quantum state. As an example, consider $k = 1$ and only the error $\bar{Z}$. If $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$ (with $|\alpha|^2 + |\beta|^2 = 1$), and the affected state $\bar{Z}|\psi\rangle$ is not corrected, the decoded state's fidelity will be

$$F = |\langle \psi | \bar{Z} | \psi \rangle|^2 = ||\alpha|^2 - |\beta|^2|^2, \quad (3.31)$$

which may be any value from zero (if $|\alpha| = |\beta|$) to one (if $|\alpha|$ or $|\beta|$ equal one). As we wish to determine an estimate of the QGRAND's performance which is independent of the initial quantum data of interest, we may estimate a lower bound $F_{\min}$ for the fidelity of the corrected state, averaged over the noise probability distribution, for any given initial quantum state. A straightforward lower bound may be obtained by considering the worst-case scenario, where all errors in $\mathcal{E}$ (except $E_0 = I$) map the initial quantum data to an orthogonal state, resulting in $F = 0$ if an error occurs. Therefore, we have the lower bound

$$\langle F \rangle \geq F_{\min} = \sum_{i \in \mathcal{C}} \epsilon_i \geq \sum_{i \in \mathcal{C}_{\text{non-degen.}}} \epsilon_i = 1 - \text{BLER}, \quad (3.32)$$

with $\mathcal{C}$ the set of indices for error patterns that can be successfully corrected by the code. $\mathcal{C}_{\text{non-degen.}} \subseteq \mathcal{C}$ considers only errors that are correctable thanks to a distinct syndrome, and not thanks to code degeneracy. $1 - F_{\min}$ turns out to be further lower bounded by the block error rate (BLER) in classical error correction, with the difference resulting from code degeneracy. Since the probability of an error being degenerate roughly scales with $1/L = 1/2^{2k}$ (cf. Section B), this possibility can be disregarded for moderate or high k . Nonetheless, by computing the BLER, our results slightly underestimate the codes' performance for very low values of k .

The error probability associated with quantum communications and quantum computation is typically in the order of $10^{-3} - 10^{-1}$ ([163, 164, 165]), higher than what is usually encountered in classical applications. For fault-tolerant computation, it is estimated that an error probability below $\sim 10^{-2}$ would be required [108]. To motivate the use of QGRAND for near-future applications, for this work, we consider error probabilities in this range.

Figures 3.5 and 3.6 show how both the block error rate (BLER, which is an upper bound to the infidelity in the quantum setting) and the fraction of error patterns of weight t evolve as a function of the code rate for QRLCs with codeword lengths $n = 128$ and $n = 32$, respectively.

These results are obtained semi-analytically, as the QRLCs are constructed randomly, but their performance is computed exactly from each QRLCs' list of minimal stabilizers, represented by the parity check matrix A . The implementation uses the Python package *Qiskit*. For $n = 128$, we compute 31 samples for each $k \geq 90$. For $n = 32$, all k are considered. For each sample, a QRLC with the indicated number of gates (it varies for $n = 128$, and it is always 2000 for $n = 32$) is constructed. Its performance is obtained by summing the probabilities p_i of the most likely error patterns $E_i \in \mathcal{E}$ associated with each syndrome. These are the error patterns that can, in fact, be corrected by QGRAND (disregarding degeneracies). The syndrome of each error is efficiently computed as $\mathbf{s}_i = M_{E_i} A^T$, with M_{E_i} the binary matrix representation of E_i . Only error patterns up to the desired weight t are considered; thus $N = B_t$ as in (3.22).

As the code rate increases, the code manages to correct fewer and fewer error patterns, with an error rate approaching the case where no error can be corrected. Lower code rates eventually plateau for each weight t , as the codes can eventually correct all error patterns with that weight. As expected from Fig. 3.3, if the encoding has too few $\mathcal{C}_2$ gates, it is not expressive enough to reach the performance of an ideal

random code (marked by dots, (3.23)), but, as more gates are used, the observed performance of these QRLCs matches it, qualitatively suggesting that GRAND’s results for uniform-at-random codes may be applicable to quantum random encodings. Note that the variance of the code performance also decreases with an increasing number of gates, indicating that a good code is obtained with high probability.

Approximating QRLCs by their ideal counterparts, we may estimate their performance’s dependence on the noise probability. As seen in Fig. 3.7, increasing n improves the code’s performance, with a lower p observing greater improvements. The ideal approximation from Section B (dots) continues to provide a quick and good estimate of QGRAND’s performance.

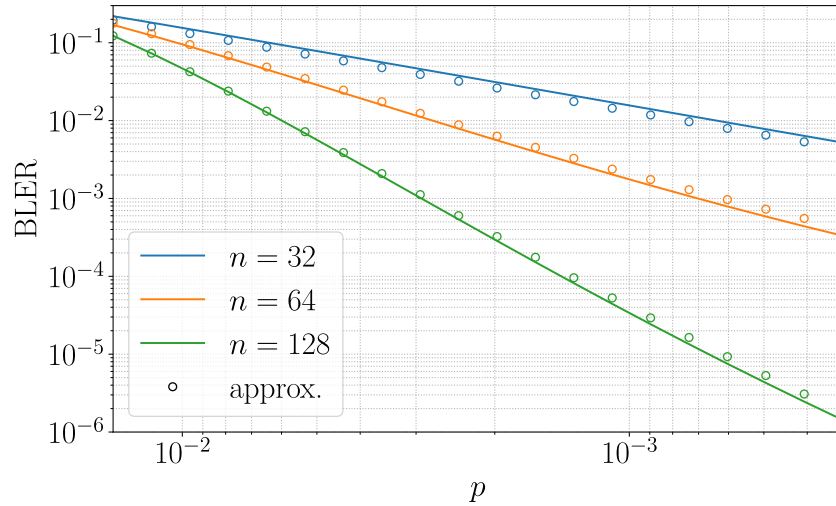


Figure 3.7: Codes’ performance as a function of the probability p of each qubit suffering an error for different code lengths n , with $R = 0.7$. A Bernoulli noise model is used.

3.8 Conclusions and future directions

This work introduced a novel approach to quantum error correction by bringing together quantum random linear encoding and noise guessing decoding, considering a system model with channel errors only, as a first extension of the GRAND concept to the quantum realm. QGRAND is particularly suitable for codes with short codewords, or with high code rate, as highlighted in [148, sec. VI-B]. Given the difficulties of manipulating large quantities of qubits, this type of error protection seems particularly fit to the presently envisaged quantum technologies requiring error correction, such as the transmission of quantum entanglement, quantum repeaters, quantum memories, and quantum computing.

It was first verified by numerical simulations that the construction leading to random quantum codes can be efficiently implemented, even outside the regime with theoretical guarantees. Although we focus on the case of all-to-all qubit connectivity, this method can be easily generalized to more restrictive connectivities. A possible approach would be to limit the 2-qubit gates composing the encoding to only connect pairs of qubits with a direct connection. Employing higher circuit depths to achieve similarly powerful encodings, and limiting the stabilizer weight [112] would enable versatile practical implementations.

By taking advantage of the noise statistics, known from the medium at hand, an efficient and flexible decoding process is obtained. One may even consider a scenario where the syndrome measurements are not promptly available, and rather there is only access to a membership test, flagging whether the syndrome is zero or not. The QGRAND methodology is generalizable beyond stabilizer codes, and is suitable for time-variant noise statistics and changing setup requirements, updated on the fly.

In the quantum setting, we may consider different forms of abandonment. In the classical GRANDAB approach, the iterative noise guessing process is stopped after a chosen number of attempts, depending on the noise entropy. In the case where the syndrome information is being actively used, we may not only stop testing error patterns after a given threshold, but we may also stop measuring stabilizers after a given number of measurements, if the conducted measurements rule out all but very unlikely errors. In general, we expect the QGRAND approach to be directly amenable to the GRANDAB formalism.

This work considered QECCs mostly for quantum communications and quantum memories, where the errors mostly arise during the quantum state transmission or storage. However, in contexts such as quantum computation, the errors during the encoding and decoding process themselves may be of the same order of magnitude as those encountered in the intended computations, and therefore, an important research question is to find how robust the whole encoding-decoding chain is when considering such errors in the system model. The first step should be assessing the impact of imperfect measurements when obtaining the syndrome, and how the binary decision they control (i.e., the membership test) is perturbed. The application and efficiency of QGRAND (and GRANDAB) for practical applications with limited qubit connectivity, and also in the context of fault-tolerant systems, is the subject of ongoing work.

3.9 Supporting Information

A Stabilizer order

For the syndrome decoding process, we may choose the order in which we apply the stabilizers. For a random ordering of the s minimal stabilizers, it will take, on average, measuring $\sim \log_2(N+1)$ stabilizers to identify which specific error pattern in $\mathcal{E}$ has occurred. If the syndrome is only being used as a membership test, then one needs to distinguish each error pattern in $\mathcal{E} \times \mathcal{E}$, which takes $\sim 2 \log_2(N+1)$.

Instead of choosing the stabilizer ordering at random, we may try to choose the stabilizer with highest information gain, which, thanks to the random assignment of syndromes to error patterns, will likely correspond to the stabilizer with which the most error patterns anticommute.

Using this information, one may precompute a decision tree that maximizes the information gain at each measurement step. An example can be seen in Fig. 3.8, where we see the average number of stabilizers required to determine the error that occurred, or if no error occurred at all, for different error distributions, as a function of the noise entropy $H(\mathcal{N})$. The shaded region marks one standard deviation. Random $(30, 1)$ codes are used, and only error patterns of weight $t = 1$ are considered. When all cases

are equally likely and the entropy is maximal, the decision tree method will require about the same number of stabilizers, $\log_2(N+1)$ (black solid line), as the naïve method, where the stabilizers are chosen at random. For lower entropy distributions, only minor savings can be achieved. In the limit of zero entropy, where no errors can occur, fewer stabilizers are required (with the minimum given by the dashed line, computed using (3.33)). The decaying and constant distributions correspond to $\{p_i = \exp(-\alpha i)\}$ and $\{p_i = \epsilon, p_0 = 1 - N\epsilon\}$, respectively, with both distributions normalized and α, ϵ chosen so as to form a probability distribution with the indicated Shannon entropy $H(\mathcal{N})$.

This approach might even be used to create a membership test requiring a smaller number of stabilizers. However, here we show that such a procedure will lead to very minor savings in the necessary number of iterations, when compared with the straightforward random ordering.

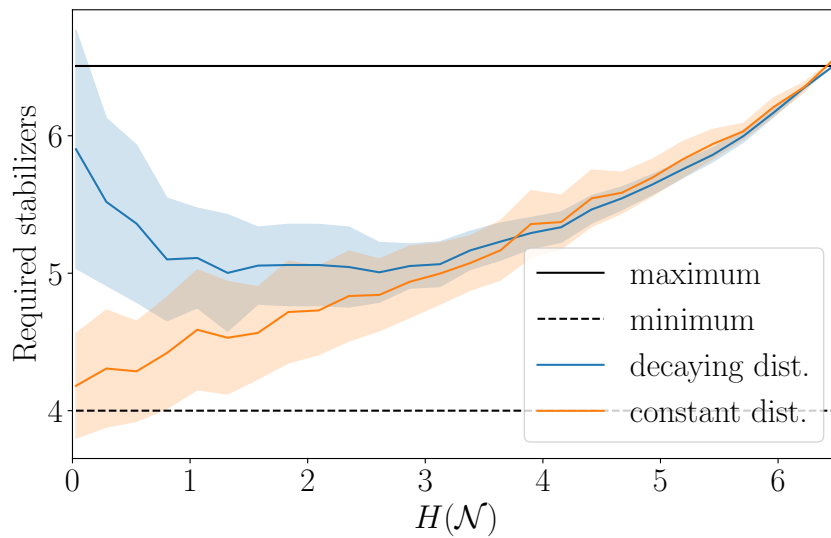


Figure 3.8: Average number of stabilizers required to determine the error that occurred, or if no error occurred at all, for different error distributions, as a function of the noise entropy $H(\mathcal{N})$. The shaded region marks one standard deviation. Random $(30, 1)$ codes are used, and only error patterns of weight $t = 1$ are considered.

Let X_i be the distribution for the number of error patterns in $\mathcal{E}$ that anticommute with stabilizer $S_i \in \mathcal{S}_{\min}$. Then, approximately, $X_i \sim \text{Binomial}(N, 1/2)$, using the binomial distribution. For large N , X_i can be approximated by the normal distribution $\text{Normal}(n/2, n/4)$. The stabilizer that can detect the highest number of error patterns, will detect on average $Z \sim \max_i X_i$ error patterns. By Jensen's inequality, its expected value is upper-bounded by $\mu + \sigma\sqrt{2 \ln s} = n/2 + \sqrt{(n/2) \ln s}$. Assuming that the stabilizer chosen for each iteration matches this upper bound and the X_i distributions are independent, we have N_j error patterns remaining at iteration j , on average, when starting with N error patterns. The recursion relation is given by

$$N_0 = N, \quad N_{j+1} = \frac{N_j}{2} - \sqrt{\frac{N_j}{2} \ln(s-j)}. \quad (3.33)$$

An approximate lower bound for the minimum number of measurements is given by I , such that $N_I \simeq 1$.

This bound is not exact as the substitution of the binomial distribution for a normal distribution is only valid for large N_j , which necessarily cannot hold for $j \gg 1$. Nonetheless, it provides us with a simple rule of thumb to check whether the ordering approach can potentially have large savings in measurements. Using $A_j \triangleq \sqrt{N_j/2}$, we have

$$A_{j+1} = \frac{A_j}{\sqrt{2}} \sqrt{1 - \frac{1}{A_j} \sqrt{\ln(s-j)}} \simeq \frac{A_j}{\sqrt{2}} - \frac{\sqrt{\ln s}}{2\sqrt{2}}. \quad (3.34)$$

By solving this recursion relation, it can be shown that the number of iterations required will be, approximately, for large N and s ,

$$I \simeq \log_2 N - \log_2 \log_2 s - 1. \quad (3.35)$$

Fig. 3.9 provides numerical confirmation of that. Suppose that our setup follows (3.10), so that $s \propto \log_2 N$. As N increases, we expect no significant increase in savings for the number of iterations, when compared with the approach of picking stabilizers at random, which takes $\sim \log_2 N$ iterations. Thus, it is not justifiable, in practice, to compute a syndrome table to construct a decision tree leading to the highest information gain per iteration, as the gains are negligible.

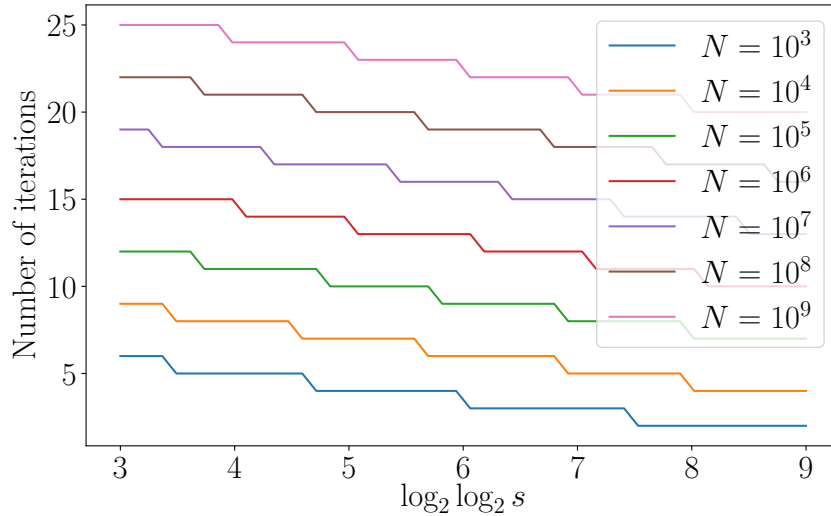


Figure 3.9: Expected number of iterations, using the decision tree method, for large N and s . Using $I = a \log_2 N - b \log_2 \log_2 s - c$, a least squares fit obtains the parameters $(a, b, c) = (0.96, 0.91, 0.84)$, which is approximately in agreement with (3.35).

Chapter 4

Fault-tolerant noise guessing decoding of quantum random codes

Abstract

This work addresses the open question of implementing fault-tolerant QRLCs with feasible computational overhead. We present a new decoder for quantum random linear codes (QRLCs) capable of dealing with imperfect decoding operations. A first approach, introduced by Cruz et al., only considered channel errors, and perfect gates at the decoder. Here, we analyze the fault-tolerant characteristics of QRLCs with a new noise-guessing decoding technique, when considering preparation, measurement, and gate errors in the syndrome extraction procedure, while also accounting for error degeneracy. Our findings indicate a threshold error rate ($p_{\text{threshold}}$) of approximately 2×10^{-5} in the asymptotic limit, while considering realistic noise levels in the mentioned physical procedures.

4.1 Introduction

It is known that classical random linear codes (RLCs) are capacity-achieving [166], however, until the advent of guessing random additive noise decoding (GRAND) their decoding was not practical, apart from some decoders based on trellises (as pointed out in [2]). GRAND has been proposed with the aim of reducing end-to-end latency in coded wireless systems, which has been a drawback for a long time. The rationale in the original proposal of GRAND was that by using short codewords, the so-called interleavers, used to make the errors independent and identically distributed (iid), would no longer be required [167]. Using short blocks in wireless systems also helps to better adapt to the channel variations when applying precoding techniques [168, 169, 170]. In the quantum realm, due to technical limitations in manipulating qubits, short block codes appear as natural candidates for quantum error correction codes (QECCs) [111, 171, 172, 173, 139]. These limitations also necessitate the development of fault-tolerant techniques to handle noise and errors in quantum operations [157].

Likewise classical RLCs, quantum random linear codes (QRLCs) attain the capacity of the quantum channels, but no practical decoder existed for them until the advent of quantum guessing random additive noise decoding (QGRAND), which allowed to numerically assess their performance for the first time [2]. A recent work also used a GRAND-like approach to decode several families of structured quantum codes which are based on stabilizer codes [174]. QGRAND has also been applied to the purification of quantum links, taking advantage of the connection between purification and error correction [7], which will have great implications on the way routing is implemented in quantum networks [105, 106, 104, 175].

QRLCs are a much more flexible solution than other structured quantum codes for QECCs, with advantages in respect to the state-of-the-art solutions designed to detect and correct errors in quantum setups [109, 110]. In contrast to structured codes, which may only exist for a very limited number of code rates and codeword lengths [176, 115], QRLCs can exist for a wide range of coding rates and codeword lengths that may better fit some particular applications. A method to generate QRLCs efficiently was proposed in 2013 [141], however, almost no practical method existed until recently to decode them until the proposal in [2]. The channel model used in that work was a Shannon-like channel, where errors occur only in the channel and all the decoding process is perfect. However, in all current technologies implementing qubits, the errors that take place in the quantum gates of the decoding circuit cannot be ignored. Hence, a practical challenge remained after [2]: can a QRLC-QGRAND system be made practical in the presence of the extra errors coming from the quantum gates, enabling fault-tolerant QECCs based on QRLCs?

This paper shows that, surprisingly, due to the particular way that the syndrome extraction takes place in codes based on stabilizers, some heavy reduction of the effects of those errors takes place, making the whole system viable. Building on previous work [2, 7], we present a comprehensive analysis of fault-tolerant QRLCs, incorporating the effects of preparation, measurement, and gate errors. Our results show that QRLCs, decoded with the proposed method, exhibit robust error correction capabilities with a threshold error rate $p_{\text{threshold}}$ of approximately 2×10^{-5} . This advancement paves the way for practical implementations of QRLCs in quantum error correction, contributing to the development of scalable and resilient quantum systems.

While recent work by Nelson et al. [177] has addressed fault-tolerant quantum error correction using low-depth random circuit codes, our work focuses on quantum random linear codes (QRLCs) decoded with the QGRAND technique. Both approaches work with stabilizer codes generated through random constructions, but differ in their specific code constructions, decoding methods, and target applications. Their work addresses fault-tolerant state preparation and distillation protocols for quantum memory applications, while our approach focuses on fault-tolerant syndrome decoding for error correction.

Although our results suggest that QGRAND could in theory enable a fault-tolerant implementation of QRLCs, some challenges remain that limit its usefulness in that regime. QGRAND is most suitable for situations where the noise entropy is relatively low, in which case decoding becomes computationally efficient. However, in the fault-tolerant regime where n may be considered to be considerably large or it is necessary to iteratively apply error correction to suppress errors, the noise entropy can be considerably high. In this regime, the optimal procedure described in this paper becomes infeasible, and suboptimal

Var.	Description	Relationships
U	Unitary encoding circuit for a quantum error-correcting code	
S_i	The i^{th} minimal stabilizer of the code ($1 \leq i \leq n - k$)	$S_i = U Z_{i+k} U^\dagger$
X_j	Logical X operator on the j^{th} encoded qubit ($1 \leq j \leq k$)	$X_j = U X_j U^\dagger$
Z_j	Logical Z operator on the j^{th} encoded qubit ($1 \leq j \leq k$)	$Z_j = U Z_j U^\dagger$
$\mathcal{N}$	List of all possible errors E_i in the noise model, with associated probabilities p_i	
E_i^B	A base error (error affecting a single qubit or gate)	
ω	The number of base errors that compose a compound error	$E_j = E_{i_1}^B \cdots E_{i_\omega}^B$
e_i^l	Local error pattern corresponding to a base error E_i^B	
C, B	Unitary components of a syndrome extraction circuit, applied before/after an error E_i^B occurs	$V = CB$
V_E	Syndrome extraction circuit affected by error E_i^B	$V_E = C e_i^l B$
e_i	Propagated error pattern on the main qubits after E_i^B and subsequent circuit operations	$e_i = (C e_i^l C^\dagger)_m$
e_i^s	Propagated error pattern on the ancilla qubits after E_i^B and subsequent circuit operations	$e_i^s = (C e_i^l C^\dagger)_a$
$\mathbf{A}$	Quantum check matrix of the code (binary representation in $[X Z]$ format)	
$\mathbf{e}_i$	Binary representation of the main error pattern e_i (in $[Z X]$ format)	
$\mathcal{L}$	Logical error group generated by X_j and Z_j	
$\mathbf{L}_i$	One of the 2^{2k} logical error patterns in $\mathcal{L}$	$e_i = \mathbf{E}_i \mathbf{S}_i \mathbf{L}_i$
$\mathcal{S}$	Stabilizer group generated by S_j	
$\mathbf{S}_i$	One of the 2^{n-k} stabilizer patterns in $\mathcal{S}$	$e_i = \mathbf{E}_i \mathbf{S}_i \mathbf{L}_i$
$\mathbf{E}_i$	Error pattern with the same syndrome as e_i	$e_i = \mathbf{E}_i \mathbf{S}_i \mathbf{L}_i$
$\mathbf{0}$	Vector representing zero syndrome ($n - k$ zero bits)	
$\hat{\mathbf{s}}_i$	Syndrome associated with a (propagated) error pattern e_i	$\hat{\mathbf{s}}_i = \mathbf{e}_i \mathbf{A}^T$
$\mathcal{D}$	A degenerate set: A set of error patterns with the same syndrome that can be corrected similarly	
e_i^d	Representative of the errors in a degenerate set $\mathcal{D}$	$e_i^d = \mathbf{E}_i \mathbf{L}_i$
g	Index of the syndrome extraction where an error E_i^B occurred	
$\tilde{\mathbf{s}}_i$	Syndrome acquired in the same extraction as error E_i^B	$\tilde{\mathbf{s}}_i = \text{comp}_X(\text{bin}(e_i^s))$
$\hat{\mathbf{s}}_i$	Syndrome acquired in a subsequent extraction after E_i^B occurred	$\hat{\mathbf{s}}_i = \text{comp}_X(\text{bin}((V e_i V^\dagger)_a))$
$\mathbf{s}_i$	Measured syndrome in a particular syndrome extraction	
s	List of all acquired syndromes over multiple extractions: $\{\mathbf{s}^1, \dots, \mathbf{s}^q\}$	
$\mathcal{Q}(E_j)$	Syndrome sequence expected for a compound error E_j	$\mathcal{Q}(E_j) = s_{i_1} \oplus \cdots \oplus s_{i_\omega}$

Table 4.1: Notation summary.

heuristics would have to be introduced. Nonetheless, this work paves the way for applications of QGRAND whenever the considered noise types all have low entropy, which encompasses setups with realistic noise conditions.

This paper is organized as follows. In Section 4.2 we introduce the setup considered in the analysis, and in particular its noise model. In Section 4.3, we define some useful error notation terms and set the notation used throughout the paper. Section 4.4 presents the decoding method, extended from [2] to account for degenerate errors. In Section 4.5, we present an analysis of the codes' performance for various qubit counts, and in Section 4.6 we present some final thoughts on our results. To help the reader, a notation summary is listed in Table 4.1.

4.2 Setup and noise model

We use the same setup as in [2], but consider the fault-tolerant regime, where the constituent quantum gates in the circuit may be affected by error. We consider an initial k -qubit $|\psi\rangle$ quantum state, to be encoded into $n > k$ qubits. [141] presents a method of generating a random qubit encoding, which we use in this work. One starts by randomly selecting Clifford unitaries from the $\mathcal{C}_2$ group (i.e., Clifford unitaries for 2 qubits). There are $|\mathcal{C}_2| = 11\,520$ such unitaries, and all of them can be built by simple combinations of the Hadamard (H), phase ($\sqrt{Z}$), and CNOT gates, which have efficient physical implementations in virtually any quantum setting [178]. In matrix form, these are defined as

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}, \quad \sqrt{Z} = \begin{pmatrix} 1 & 0 \\ 0 & i \end{pmatrix}, \quad \text{CNOT} = \begin{pmatrix} I & 0 \\ 0 & X \end{pmatrix}, \quad (4.1)$$

with X, Y , and Z the Pauli matrices, and I the 2×2 identity matrix.

After selecting these random unitaries from $\mathcal{C}_2$, one successively applies each of them to a random pair of qubits, taken from the set of n qubits.

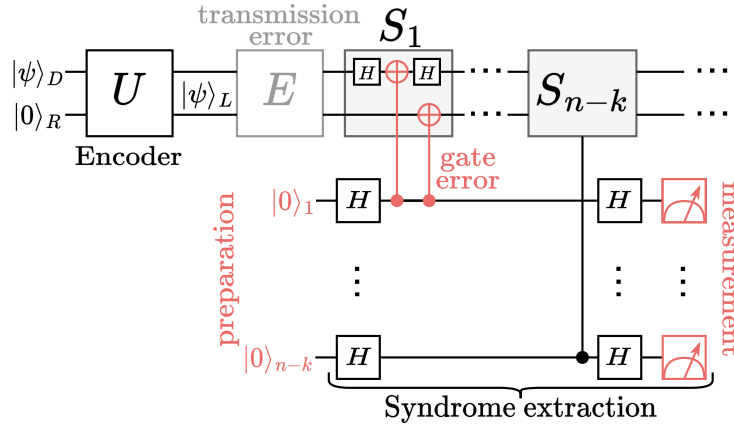


Figure 4.1: Noise model considered.

This process leads to an encoding unitary for our stabilizer code which, when applied to the initial k qubits and $(n - k)$ extra $|0\rangle$ qubits added, returns a n -qubit encoded quantum state. As shown in [141], as long as $\mathcal{O}(n \log^2 n)$ gates are used, with a circuit depth of $\mathcal{O}(\log^3 n)$, the construction leads to a highly performant (n, k) code, and from [112] it is already known that these complexity orders can be further lowered.

We use these QRLCs to construct stabilizer codes. Compared to the approach in [2], in this work, we consider a noise model that is more realistic by also including preparation, measurement, and gate errors. Given that, in practical applications, the error of 2-qubit entangling gates generally dominates over single-qubit gate errors [173], we focus on the former type of error. We further analyze the appropriateness of our model in Section L. We assume that every gate in both the encoding and syndrome extraction steps is decomposed into the Clifford gates $\{\text{CNOT}, H, \sqrt{Z}\}$.

For the noise statistics, we consider the model similar to the one in [173], but without single-qubit gate errors (see Fig. 4.1):

- **CNOT gate errors:** After the ideal implementation of the $\text{CNOT}(a, b)$ gate, with qubit a controlling b , it is assumed that one of the 15 errors of the form

$$O_a O_b \tag{4.2}$$

with $O_a, O_b \in \{I, X, Y, Z\}$ and excluding $O_a O_b \neq I_a I_b$, occurs with probability $p/15$. Here, I is the identity gate, and X, Y, Z are the Pauli matrices.

- **Preparation errors:** While setting the $(n - k)$ ancilla qubits (for each syndrome extraction) to $|0\rangle$, each qubit has (independently) a probability p of being prepared in the state $|1\rangle$ instead.
- **Measurement errors:** While measuring each ancilla qubit to extract the syndrome, each measurement bit has a probability p of being misread, so that a zero bit is read as a 1, and vice-versa.

Unlike the model in [2], to demonstrate the fault-tolerant properties of this model, we exclude a source of error between the encoding and syndrome extraction sections (i.e., the “transmission error” in Fig. 4.1), and instead focus on the case where the CNOT gate error stemming from the syndrome extraction dominates the noise statistics of the circuit. This simpler model facilitates the study of the QGRAND decoding approach in the fault-tolerant regime, which is the focus of this work. While possible (see Section A), we make no further modifications to the circuit implementation.

4.3 Error correction overview

In the fault-tolerant regime with a noisy gate model, degenerate errors play a significant role in the error correction capabilities of the code [179]. As a result, the approximation made in [2], where codes were approximated to be non-degenerate, is no longer accurate, as it would significantly underestimate the code’s capabilities.

Additionally, in the fault-tolerant regime, we must consider an iterated application of the syndrome extraction procedure, instead of a single application, in order to capably detect the errors being introduced by the syndrome extraction procedure itself. This is a common approach [180, 109] to quantum error correction when gate and measurement errors are non-negligible, and the decoding procedure has into account not just the syndrome from one extraction process, but the whole history of syndrome measurements.

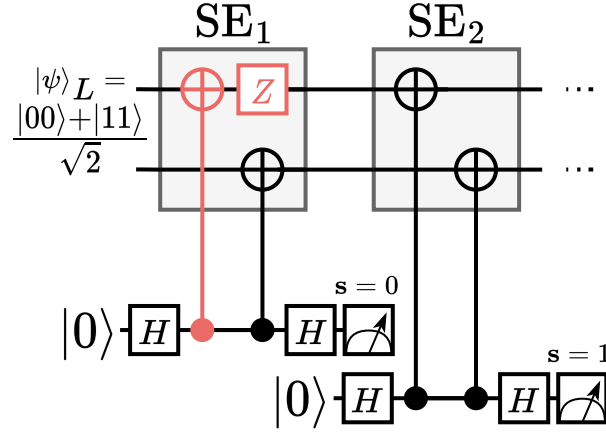


Figure 4.2: Simple example, with two syndrome extractions (SE). The error is not detected by the first syndrome extraction process, but it is detected by the second extraction.

As a result of this added complexity, in this section we clarify the notation we use in this work. We use $\bar{X}_j, \bar{Z}_j$ to represent the logical operators corresponding to the unencoded operators X_j, Z_j , respectively. Given an encoding U (see Fig. 4.1), the choice of minimal stabilizers S_i and logical operators is not unique. Without loss of generality (W.l.o.g.), we consider the minimal stabilizer S_i (for $1 \leq i \leq n - k$) and logical operators $\bar{X}_j, \bar{Z}_j$ (for $1 \leq j \leq k$) to be given by

$$S_i = U Z_{i+k} U^\dagger \quad (4.3)$$

$$\bar{X}_j = U X_j U^\dagger \quad (4.4)$$

$$\bar{Z}_j = U Z_j U^\dagger. \quad (4.5)$$

Following Section 4.2, the noise model enables us to create a list $\mathcal{N} = \{(p_0, E_0), (p_1, E_1), \dots\}$ of all the errors E_i that the encoded quantum state may be subjected to, along with its respective probability p_i of occurring. An error E_i refers to the qualitative process that occurred physically, such as “a $X_2 Z_3$ error occurred in the CNOT(2, 3) gate, and no other errors”, for example.

An error E_i that corresponds to either only one wrongly prepared qubit, or one wrongly measured qubit, or one noisy CNOT gate, is called a *base error*, and may be explicitly labeled as E_i^B . Every other error in the noise model of Section 4.2 can be described as a combination of base errors.

We consider the errors E_i to be disjunctive, so only one error in $\mathcal{N}$ may occur, and their probability sums to 1. When using the base error notation E_i^B , we implicitly refer only to the specific base error that occurred, without making claims about the occurrence of other base errors. For example, using the noise model in Section 4.2, the base error $E^B = “X_2 Z_3 \text{ error in the CNOT}(2, 3) \text{ gate}”$ would have a probability of occurring of $p/15$, while the corresponding error $E = “X_2 Z_3 \text{ error in the CNOT}(2, 3) \text{ gate, and no other errors}”$ would have a probability of $(p/15) \times P(\text{no other base error occurs})$, which would possibly

be much lower. We may use the shorthand notation $\hat{E}^B := E$ for errors where only one base error occurs.

Compound errors may be represented by their own symbol or as the product of errors that compose it. That is, for simplicity, given base errors E_i^B and E_j^B , we also have the compound error notation

$$E_r = E_i^B \cap E_j^B \cap \left(\bigcap_{m \neq i,j} \overline{E_m^B} \right) =: E_i^B E_j^B \quad (4.6)$$

An error E_j is said to be of order ω if ω base errors suffice to describe it, so that $E_j = E_{i_1}^B \cdots E_{i_\omega}^B$. Note that the constituting base errors may stem from different syndrome extractions.

Given a base error, let e^l be the *local error pattern* corresponding to the error that occurred locally. In the example above, we would have $e_i^l = X_2 Z_3$. In general, for gate errors affecting only one CNOT gate, we would have an error pattern from Eq. (4.2). For preparation and measurement errors, e^l would be represented by X operators in the appropriate ancilla qubits.

Unless the error E_i^B occurs at the end of a syndrome extraction process, it will propagate through the rest of the quantum circuit, possibly impacting other qubits. Let V be the unitary corresponding to one noiseless syndrome extraction process, minus the final measurement step of the ancilla qubits. We may partition V into the unitaries B and C , corresponding to the portion of the syndrome extraction circuit that occurs before and after the error E_i^B , respectively. If the syndrome extraction affected by E_i^B is given by the unitary V_E , we have

$$V = CB \quad (4.7)$$

$$V_E = C e_i^l B = (e_i^s \otimes e_i) V. \quad (4.8)$$

The resulting (propagated) error pattern may affect non-trivially both the main n qubits (*main error pattern* e_i) and the $(n - k)$ ancilla qubits (*ancilla error pattern* e_i^s).

A Pauli string of n qubits is an operator that is the product of the Pauli operators X, Y , and Z on those qubits. It has the form

$$e^{i \frac{\pi}{2} \phi} O_1 O_2 \cdots O_n, \quad (4.9)$$

$$\text{with } O_j \in \{X, Y, Z, I\} \text{ and } \phi \in \{0, 1, 2, 3\}.$$

As e_i^l is a Pauli string, and C is a Clifford unitary, then both e_i and e_i^s are Pauli strings. Similarly, following Eqs. (4.3) to (4.5), the minimal stabilizers and logical operators are also Pauli strings, since U is a Clifford unitary as well. A Pauli string is said to have weight t if it acts on t qubits, that is, if its Pauli string contains t Pauli operators (excluding the identity).

For the previous example with E_i^B , we would have $e_i^s \otimes e_i = C(X_2 Z_3)C^\dagger$. While the effect of e_i^s is removed by the syndrome measurement, the same cannot be said of e_i . As E_i^B propagates through the circuit, the pattern e_i is picked up by subsequent syndrome extractions, and is ultimately the error pattern that

our correction process needs to consider to undo the effect of E_i^B on the main n qubits. See Fig. 4.2 for an example.

In Fig. 4.2, we showcase a simple example, with two syndrome extractions (SE), where there is only one minimal stabilizer $S_1 = X_1X_2$. An error occurs in the first CNOT gate, so $E = "Z_1I_2$ error in first CNOT gate, and no other errors". The error is not detected by the first syndrome extraction process, so $\mathbf{s} = \hat{\mathbf{s}} = 0$. By the end of the first syndrome extraction, the evolved uncorrected error is $e = Z_1I_2$. It is now detected by the second extraction, so $\mathbf{s} = \hat{\mathbf{s}} = 1$. If it is not corrected, subsequent extractions will behave similarly to the second one, returning the syndrome 1.

Since $Y = iXZ$, any Pauli string of n qubits can also be written in the form

$$e^{i\frac{\pi}{2}\varphi}(O_1^X \cdots O_n^X)(O_1^Z \cdots O_n^Z), \quad (4.10)$$

with $O_j^X \in \{X, I\}$, $O_j^Z \in \{Z, I\}$ and $\varphi \in \{0, 1, 2, 3\}$.

The Pauli string may then be encoded as a binary row vector. In $[X|Z]$ format, it takes the form

$$\begin{bmatrix} b_1^X & b_2^X & \cdots & b_n^X & b_1^Z & b_2^Z & \cdots & b_n^Z \end{bmatrix}, \quad (4.11)$$

$$\text{with } b_j^P = \begin{cases} 1, & \text{if } O_j = P \\ 0, & \text{if } O_j = I \end{cases} \quad \text{and } P \in \{X, Z\}. \quad (4.12)$$

In $[Z|X]$ format, the b_j^X entries are swapped with b_j^Z . By default, binary vectors and matrices are represented in bold. We may use the functions

$$\text{bin}(e) := \mathbf{e}, \quad \text{op}(\mathbf{e}) := e \quad (4.13)$$

$$\text{comp}_P(\mathbf{e}) := \begin{bmatrix} b_1^P & \cdots & b_c^P \end{bmatrix}, \quad (4.14)$$

with $P \in \{X, Z\}$ and $c = |\mathbf{e}|/2$, to indicate the conversion to and from binary representation, and to refer to a particular component of $\mathbf{e}$, respectively. Calculations using binary are performed in $\mathbb{F}_2$, that is, using modular arithmetic mod 2. The functions bin and op stand for the transformations of Pauli operators from and to, respectively, binary arrays.

Let $\mathbf{A}$ be the quantum check matrix [156], a $(n-k) \times 2n$ binary matrix (in $[X|Z]$ format) where each row j encodes the minimal stabilizer S_j of the code. This is a compact way of representing the encoding used. Let $\mathbf{e}_i$ be the binary representation of the error pattern e_i as a $2n$ -sized row vector, in $[Z|X]$ format. Any evolved error pattern e_i can be written as

$$e_i = \mathbf{E}_i \mathbf{S}_i \mathbf{L}_i, \quad (4.15)$$

where $\mathbf{L}_i$ is one of the 2^{2k} logical error patterns in the logical error group $\mathcal{L}$ generated by the logical operators $\bar{X}_j, \bar{Z}_j, 1 \leq j \leq k$; $\mathbf{S}_i$ is one of the 2^{n-k} stabilizers in the stabilizer group $\mathcal{S}$ generated by the

minimal stabilizers $S_j, 1 \leq j \leq n-k$; and E_i is some error pattern with the same syndrome $\hat{s} = \mathbf{e}_i \mathbf{A}^T$ as e_i [156]. We use $\mathbf{0}$ to represent the syndrome with all entries equal to zero. W.l.o.g., for the decomposition, we assume the phase factor φ (see Eq. (4.10)) to be zero, since neglecting it adds at most a global phase to the encoded quantum state, which can be disregarded. As a result, we consider each error pattern to equal its inverse.

The decomposition in Eq. (4.15) is not unique, and is dependant on the choice made for the particular logical operators, minimal stabilizers, and E_i patterns to use. For the sake of simplicity in the notation, in this work, it is assumed that such a decomposition is the unique one obtained deterministically by following the procedure described in Section 4.4. Consequently, we assume that, associated with each error pattern e_i , there is a unique set of operators E_i, S_i , and L_i . In particular, for patterns with $\hat{s} = \mathbf{0}$, the operator E_i is the identity. Since all error patterns with the same syndrome will have the same error component E , we use $E_{\hat{s}}$ to indicate the error component of the error patterns with syndrome $\hat{s}$.

Compound error patterns, such as $e_r = e_i e_j$, may be easily encoded in binary form by using the modular sum (i.e. XOR), so that $\mathbf{e}_r = \mathbf{e}_i \oplus \mathbf{e}_j$ and $\hat{s}_r = \hat{s}_i \oplus \hat{s}_j$. Given two error patterns e_i, e_j with the same syndrome $\hat{s}$, their product e_r has syndrome $\hat{s}_k = \hat{s} \oplus \hat{s} = \mathbf{0}$, so E_r is the identity operator. Therefore, there is a unique $S \in \mathcal{S}, L \in \mathcal{L}$ such that $e_i = e_j S L$, with $S = S_i S_j$ and $L = L_i L_j$.

A *degenerate set* $\mathcal{D}$ is a set of evolved error patterns that can be treated similarly, for correction purposes. This set depends on $\hat{s}$ and L . Although all error patterns with the same syndrome $\hat{s}$ have the same representative error pattern $E_{\hat{s}}$, not all can be corrected similarly. For that to be the case, their logical error component L must be the same. Since we know that two error patterns are degenerate if $e_i e_j \in \mathcal{S}$, we may verify this by computing $e_r = e_i e_j = (E_i S_i L_i)(E_j S_j L_j) = (E_i E_j)(S_i S_j)(L_i L_j) = E_r S_r L_r$. For e_r to be in $\mathcal{S}$, we must have $E_r = L_r = I$, which is only the case if $E_i = E_j$ and $L_i = L_j$. The former is true if e_i and e_j have the same syndrome $\hat{s}$, while the latter is more complicated to verify, but we know that there are only 2^{2k} possibilities in $\mathcal{L}$ to consider. Ultimately, we may index the degenerate sets based on their syndrome and logical error component, both represented by the tuple $(\hat{s}, L)$. We use $e_i^d = E_i L_i$ to refer to the actual representative of e_i (and its degenerate equivalents) during the correction process, since we know that if we can correct e_i^d by applying the unitary $(e_i^d)^\dagger$ to the circuit, so can we indirectly also correct e_i , since $(e_i^d)^\dagger e_i = S_i$, and stabilizers act as the identity on the encoded quantum state.

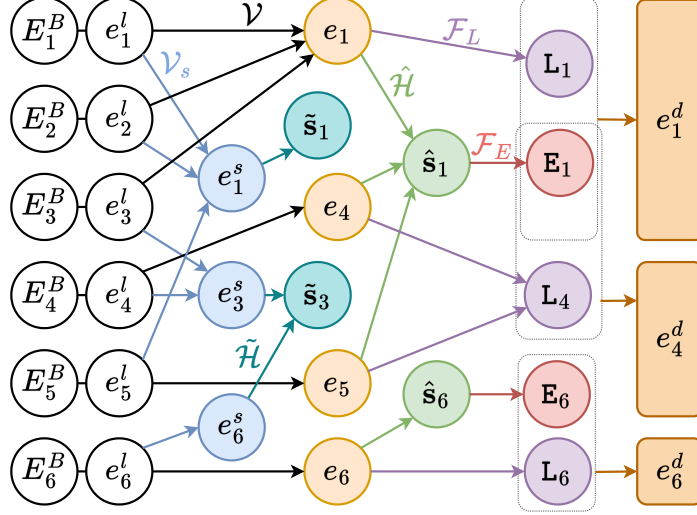


Figure 4.3: Relations between the different quantities of interest. For simplicity, every error represented is assumed to be a base error. The notation is explained in the last paragraph of Section 4.3.

For an error E_i arising from syndrome extraction g , there are two associated syndromes of interest, instead of one. The syndrome

$$\hat{\mathbf{s}}_i = \mathbf{e}_i \mathbf{A}^T \quad (4.16)$$

corresponds to the syndrome obtained from a subsequent noiseless syndrome extraction after extraction g , that is, after the error has occurred. In this case, we can consider the error model to be similar to the one used in [2], where the (propagated) error pattern e_i is present before any stabilizer is applied for the syndrome extraction process. Instead of using Eq. (4.16), we may alternatively compute $\hat{\mathbf{s}}_i$ by first computing the ancilla error pattern. If we think of e_i as a different local error pattern e_j^l , then, following Eq. (4.8), we must have

$$V e_i V^\dagger = V e_j^l V^\dagger = e_j^s \otimes e_i, \quad (4.17)$$

and $\hat{\mathbf{s}}_i$ is given by the X component (i.e. second half in $[Z|X]$ format, see Eq. (4.11)) of $\mathbf{e}_j^s$. This type of syndrome is always zero for preparation and measurement errors, since measurement errors do not affect subsequent extractions. Under this latter formalism, we may observe this by noting that if $e_i = I$, then necessarily $e_j^s = I$ and $\hat{\mathbf{s}}_i = \mathbf{0}$.

Beyond this typical syndrome, we also have the syndrome $\tilde{\mathbf{s}}_i$ obtained from the same extraction g where the error E_i^B occurred. For instance, if the error occurs at the last implemented CNOT gate, it is likely that $\tilde{\mathbf{s}}_i = \mathbf{0}$. Unlike $\hat{\mathbf{s}}_i$, this syndrome is non-zero for simple measurement errors. In general, this syndrome contains less information than $\hat{\mathbf{s}}_i$, since, for errors later in the extraction, many of the syndrome bits will be zero, as the stabilizers were applied before the error occurred. Following the second approach previously presented to compute $\hat{\mathbf{s}}_i$ (see Eq. (4.17)), we have that $\tilde{\mathbf{s}}_i$ is given by the X component of the original ancilla qubit pattern $\mathbf{e}_i^s$ (in binary form).

Both $\tilde{\mathbf{s}}$ and $\hat{\mathbf{s}}$ refer to a $(n - k)$ bit string, corresponding to the syndrome that could be obtained from a

single syndrome extraction. As both these syndromes can be deterministically obtained from the error E of interest, we use the simple notation $\mathbf{s}$ to specifically refer to the syndrome that is measured during the syndrome extraction. As previously stated, for errors stemming from only one extraction (labeled g), we have either $\mathbf{s} = \tilde{\mathbf{s}}$ (if the measured syndrome comes from the noisy extraction), $\mathbf{s} = \mathbf{0}$ (if it comes from a previous extraction) or $\mathbf{s} = \hat{\mathbf{s}}$ (if from a subsequent extraction).

In general, compound errors may stem from multiple syndrome extractions. We use superscript notation to indicate the extraction index, in order to distinguish it from the error index (which is a subscript). When there are q syndrome extractions, we refer to the total list of measured syndromes by $s := \{\mathbf{s}^1, \dots, \mathbf{s}^q\}$. If some base error $E_i^{B,g}$ occurs at extraction g , we expect to measure the syndrome sequence given by

$$s_i = \{\dots, \mathbf{0}^{g-1}, \tilde{\mathbf{s}}_i^g, \hat{\mathbf{s}}_i^{g+1}, \dots, \hat{\mathbf{s}}_i^q\}. \quad (4.18)$$

Note that, for compound errors, the syndrome sequences of the constituting errors may be combined. If $E_r = E_i E_j$, then $s_r = s_i \oplus s_j$, where the modular sum operation is applied element-wise, to all q syndromes. Then, for $E_j = E_{i_1}^B \dots E_{i_\omega}^B$, we have

$$\mathcal{Q}(E_j) := s_j = s_{i_1} \oplus \dots \oplus s_{i_\omega}. \quad (4.19)$$

In particular, if errors E_i and E_j occur at extractions g and h ($h > g$), respectively, and no other errors occur, then we would expect to measure the syndrome sequence

$$s = \{\dots, \mathbf{0}^{g-1}, \tilde{\mathbf{s}}_i^g, \hat{\mathbf{s}}_i^{g+1}, \dots, \hat{\mathbf{s}}_i^{h-1}, \tilde{\mathbf{s}}_i^h \oplus \tilde{\mathbf{s}}_j^h, \hat{\mathbf{s}}_i^{h+1} \oplus \hat{\mathbf{s}}_j^{h+1}, \dots\}. \quad (4.20)$$

As a result, we observe that $\tilde{\mathbf{s}}_i, \tilde{\mathbf{s}}_j, \hat{\mathbf{s}}_i, \hat{\mathbf{s}}_j$ do not directly provide the full information necessary to identify the compound error $E_i^g E_j^h$ that occurred. In the general case where there are q syndrome extractions, we may require all q measured syndromes to optimally correct errors.

For compound errors E_r stemming from multiple syndrome extractions, the syndrome $\tilde{\mathbf{s}}_r$ is undefined, but $\hat{\mathbf{s}}_r$ may still be defined as

$$\hat{\mathbf{s}}_r = \bigoplus_{j=1}^{\omega} \hat{\mathbf{s}}_{i_j}, \quad (4.21)$$

where $\hat{\mathbf{s}}_{i_j}$ are the syndromes of the constituting base errors. Similarly, $e_r = e_{i_1} \dots e_{i_\omega}$. The decomposition in Eq. (4.15) and subsequent analysis is also applicable.

The notation is summarized in Fig. 4.3, with an example given in Fig. 4.2.

In Fig. 4.3, the mappings $\mathcal{V}_s, \mathcal{V}, \hat{\mathcal{H}}, \tilde{\mathcal{H}}, \mathcal{F}_E$ and $\mathcal{F}_L$ are mostly independent, and generally non-injective. $\mathcal{V}$ and $\mathcal{V}_s$ stem from Eq. (4.8), and provide the main and ancilla error patterns e_i and e_i^s . Concretely, we have $e_i = \mathcal{V}(e_i^l) = (C e_i^l C^\dagger)_m$ and $e_i^s = \mathcal{V}_s(e_i^l) = (C e_i^l C^\dagger)_a$, where m and a stand for the main and ancilla subspaces, respectively. Using Eq. (4.17) yields $\tilde{\mathcal{H}}$ and $\hat{\mathcal{H}}$. The latter can also be implemented by using Eq. (4.16). Concretely, we have $\hat{\mathbf{s}}_i = \hat{\mathcal{H}}(e_i) = \text{bin}(e_i) \mathbf{A}^T = \text{comp}_X(\text{bin}((V e_i V^\dagger)_a))$ and $\tilde{\mathbf{s}}_i = \tilde{\mathcal{H}}(e_i^s) = \text{comp}_X(\text{bin}(e_i^s))$ (see Eqs. (4.13) and (4.14)). The $\hat{\mathbf{s}}_i$ and $\mathbf{s}_i$ obtained by the syndrome extraction process

can then be used to try and determine e_i^d , the representative pattern of the degenerate set to which e_i belongs. By applying e_i^d to the noisy quantum state, we correct the effect of the error E_i . The mappings $\mathcal{F}_E$ and $\mathcal{F}_L$ are quite involved, so their description is delegated to Section 4.4.

4.4 Decoding

The error pattern statistics given by the noise model of Section 4.2 lead to a very high number of degenerate error patterns (see Section B for examples). As a result, the approximation made in [2], where codes were approximated to be non-degenerate, is no longer accurate, as it would significantly underestimate the code's capabilities. In this section, we modify the decoding procedure in [2] to account for error degeneracy. The modified procedure is optimal in principle. It has previously been shown that such optimal procedures must be #P-complete in general [181, 182, 183]. Since we are applying the decoding procedure to random codes with no exploitable structure, our decoding procedure has poor scaling capabilities for high entropy noise and large code sizes. Nonetheless, following [2], we hope to show it to be of interest in regimes of small code size or low entropy, so it is still worth exploring the decoding properties of this optimal procedure. It is also possible (though not covered in this work) for the decoding complexity to be greatly improved with simpler approximations and heuristics to the optimal approach. The optimal decoding procedure is summarily presented in algorithm 2.

Algorithm 1: Error processing

Input: $\mathcal{N}$

Output: A data table D

```

1 Procedure Data( $\mathcal{N}$ ):
2   Initialize empty data table  $D$ 
3   foreach  $(p_i, E_i)$  in  $\mathcal{N}$  do
4     Compute  $e_i, \hat{s}_i$ , and  $s_i$ 
5     Compute  $L_i$  associated with  $e_i$ 
6     Compute  $e_i^d$ 
7     if  $e_i^d$  not in any entry in  $D[s_i]$  then
8       Store  $(e_i^d, p_i)$  in  $D[s_i]$ 
9     else
10      Add  $p_i$  to  $p$  in entry  $(e_i^d, p)$ 

```

Algorithm 2: Optimal decoding

Input: $\mathcal{N}$ **Output:** A decoding table T

```

1 Procedure Decoding( $\mathcal{N}$ ):
2   Initialize empty decoding table  $T$ 
3    $D \leftarrow \text{DATA}(\mathcal{N})$ 
4   foreach entry  $s$  in  $D$  do
5     Set  $T[s]$  as the pattern  $e^d$  with highest  $p$  in  $D[s]$ 

```

When considering this optimal decoding procedure, we note that, while we focus on the noise model in Fig. 4.1, the decoding procedure is naturally applicable to models where there are additional sources of error independent of the syndrome extractions themselves. For that case, the noise statistics would simply include those additional errors.

Moreover, the decoding procedure presented in this section does incorporate any assumptions about the underlying nature of the noise, as it meant to be a fully general procedure. In particular, we do not wish to assume that higher order errors are less likely than lower order ones, as there may be practical regimes where particular high order errors dominate the noise statistics (such as burst errors). In subsection Section H, we adapt the general decoding procedure to the particular noise model described in Section 4.2.

For the decoding, we require a procedure that, given a syndrome sequence s , outputs the error pattern e^d that needs to be applied to the circuit to correct the most likely source of error. Since we are interested in the analysis of the decoding procedure in the optimal case, and less concerned about practical limitations, we assume that such a procedure corresponds to a decoding table T , storing the s, e^d pairs.

The decoding table T may be obtained as follows (see algorithms 1 and 2 for pseudo-code, and Fig. 4.8 for an example):

1. For each error E_i (with corresponding probability p_i), we compute its syndrome sequence s_i , and also e_i and $\hat{s}_i$ (corresponding to the mapping $\mathcal{Q}$ in Eq. (4.19), and the mappings $\mathcal{V}$ and $\hat{\mathcal{H}} \circ \mathcal{V}$, respectively, in Fig. 4.3, for base errors). Since the circuit is a stabilizer circuit, it can be efficiently simulated [159], and these quantities efficiently computed.
2. We compute $E_i = \mathcal{F}_E(\hat{s}_i)$ and $L_i = \mathcal{F}_L(e_i)$. As we already know $\hat{s}_i$, we end up with the degenerate set $(\hat{s}_i, L_i)$ (and its representative $e_i^d = E_i L_i$), to which e_i belongs.
3. We repeat steps 1 and 2 for all errors in $\mathcal{N}$. Starting with an empty data table D , for each error E_i , we add (e_i^d, p_i) to the entry $D(s_i)$. If an entry with e_i^d already exists, we add p_i to the entry's probability. This procedure results in the data table D .
4. For each syndrome sequence $s = \{s^1, \dots, s^g\}$ in D , we choose the degenerate set with the highest associated probability as the actual coset leader, that is, the one that is corrected if s is measured.
5. The resulting syndrome table T then acts as our decoding method.

This decoding is optimal because, for any given syndrome, there is no other way for the decoding to be more successful than as described here, since this method already picks the most probable degenerate set $(\hat{\mathbf{s}}, \mathbf{L})$, given the only information available *a priori*, which is $\mathcal{N}$ and the observed syndrome $\mathbf{s}$. It is optimal under the reasoning that we consider any unsuccessful correction to be a complete failure, with no possible partial success.

While the procedure as described is done in series, iteratively traversing the E_i errors, it can be trivially parallel, by splitting the error list across multiple parallel workers. See Section E for a full description. The pseudo-code for the parallel implementation can be seen in algorithm 4.

To implement the decoding procedure (in particular step 2), *a priori*, we require the efficient implementation of two functions:

- A function $\mathcal{F}_E : \hat{\mathbf{s}} \mapsto \mathbf{E}_{\hat{\mathbf{s}}}$ that, for a given code, and taking a syndrome $\hat{\mathbf{s}}$ as input, outputs a deterministic error pattern $\mathbf{E}_{\hat{\mathbf{s}}}$ that can act as a coset leader for the syndrome $\hat{\mathbf{s}}$. That is, any error pattern e_i with syndrome $\hat{\mathbf{s}}$ can be decomposed (following Eq. (4.15)) using the error component $\mathbf{E}_{\hat{\mathbf{s}}}$. Having access to this function considerably reduces the required serialized processing for the decoding, and the required memory, as we don't need to keep track of tentative coset leaders as we iterate through the errors E_i , and we can be sure that different parallel workers have the coset leader in the same degenerate set (in fact, we can be sure that they are equal).
- A function $\mathcal{F}_L : e_i \mapsto \mathbf{L}_i$ that, for a given code, and taking an error pattern e_i as input, deterministically outputs the logical component $\mathbf{L}_i$ of the degenerate set to which this error pattern belongs to.

These functions are described in detail in Sections A and B.

A Function $\mathcal{F}_E$

When analyzing the code, instead of working with the $(n - k)$ minimal stabilizers $\{S_i\}$ we extracted from the encoding U (see Eq. (4.3)), we work with a different set $\{S_{i,\text{rre}}\}$. Each stabilizer in this new set can be thought of as some combination of the stabilizers in $\{S_i\}$. To be more specific, considering that S_i corresponds to row i of the quantum check matrix $\mathbf{A}$ (with size $(n - k) \times 2n$), $\{S_{i,\text{rre}}\}$ corresponds to row i of $\mathbf{A}$ in reduced row echelon form (also known as canonical form). We can convert $\mathbf{A}$ to reduced row echelon form because products of stabilizers are still stabilizers. Since we are in $\mathbb{F}_2$, adding or subtracting rows of $\mathbf{A}$ is equivalent to multiplying stabilizers. As long as the resulting matrix is full rank (which is always the case, since the procedure for converting to reduced row echelon form preserves rank), the resulting new matrix $\mathbf{A}_{\text{rre}}$ encodes a new set of minimal stabilizers, $\{S_{i,\text{rre}}\}$, in its rows.

In practice, we can imagine that the measured syndrome $\mathbf{s}$ gets converted to the “reduced row echelon” syndrome $\mathbf{s}_{\text{rre}}$, which can be done with a $(n - k) \times (n - k)$ matrix that encodes the steps needed to convert $\mathbf{A}$ to reduced row echelon form. Let this matrix be $\mathbf{J}$. We have $\mathbf{A}_{\text{rre}} = \mathbf{J}\mathbf{A}$ and $\mathbf{s}_{\text{rre}} = \mathbf{J}\mathbf{s}$.

Working with $\mathbf{A}_{\text{rre}}$, let h_i be the index of the pivot of row i (it's not guaranteed that $h_i = i$, since the

pivots may not all be along the main diagonal of $\mathbf{A}_{\text{rre}}$). Since $\{S_{i,\text{rre}}\}$ comes from reduced row echelon form, the stabilizer $S_{i,\text{rre}}$ will be the only minimal stabilizer with a nonzero entry at index h_i . Then, if $1 \leq h_i \leq n$, the error Z_{h_i} necessarily yields the syndrome bit 1 for $S_{i,\text{rre}}$ and zero for all other minimal stabilizers in $\{S_{i,\text{rre}}\}$. If $n+1 \leq h_i \leq 2n$, the error X_{h_i-n} necessarily yields the syndrome bit 1 for $S_{i,\text{rre}}$ and zero for all other minimal stabilizers. Consequently, we can use these $(n-k)$ errors as a basis to construct a deterministic error pattern $\mathbf{E}_{\hat{\mathbf{s}}}$ for every syndrome $\hat{\mathbf{s}}$. Let $\mathbf{E}_i$ be the error associated with $S_{i,\text{rre}}$ ($\mathbf{E}_i$ equals Z_{h_i} or X_{h_i-n} , as described). Since the code is linear, for any syndrome $\mathbf{s}_{\text{rre}}$, if $i_1, \dots, i_k$ are the indices where the syndrome $\mathbf{s}_{\text{rre}}$ is 1, then the compound error $e = \mathbf{E}_{i_1} \dots \mathbf{E}_{i_k}$ must necessarily have syndrome $\mathbf{s}_{\text{rre}}$. In other words, $\mathbf{s}_{\text{rre}} = \mathbf{e} \mathbf{A}_{\text{rre}}^T$.

Since $e = \mathbf{E}_{\mathbf{s}_{\text{rre}}}$ when using the minimal stabilizers $\{S_{i,\text{rre}}\}$, and $\mathbf{J}$ is a linear transformation, then we must have $e = \mathbf{E}_{\hat{\mathbf{s}}}$ when using the minimal stabilizers $\{S_i\}$. Therefore, if the code has stabilizers $\{S_i\}$, then $\mathcal{F}_E$ implements the procedure

$$\hat{\mathbf{s}} \implies \mathbf{s}_{\text{rre}} \implies e = \mathbf{E}_{\hat{\mathbf{s}}}. \quad (4.22)$$

Because of the construction of e , for all error patterns with the same syndrome, the same error component is computed. The error $\mathbf{E}_{\hat{\mathbf{s}}}$ then acts as the error component for all error patterns with syndrome $\hat{\mathbf{s}}$, for the decoding.

B Function $\mathcal{F}_L$

Algorithm 3: Finding logical component

Input: $\mathbf{e}'_i, \mathbf{A}_{\text{rre}}, \mathbf{L}_{\text{rre}}$

Output: Vectors $\mathbf{u}_A, \mathbf{u}_L, \mathbf{v}'$

```

1 Procedure LogicalComponent( $\mathbf{e}'_i, \mathbf{A}_{\text{rre}}, \mathbf{L}_{\text{rre}}$ ):
2   Initialize  $\mathbf{u}_A, \mathbf{u}_L$  to zero
3   Let  $\mathbf{v} \leftarrow \mathbf{e}'_i$ 
4   Let  $H_A$  (resp.  $H_L$ ) be an ordered list of pivot positions of  $\mathbf{A}_{\text{rre}}$  (resp.  $\mathbf{L}_{\text{rre}}$ )
5   foreach  $h_i$  in  $H_A$  do
6     if entry  $h_i$  of  $\mathbf{v}$  is 1 then
7       Subtract row  $i$  of  $\mathbf{A}_{\text{rre}}$  from  $\mathbf{v}$  (mod 2)
8        $[\mathbf{u}_A]_i \leftarrow 1$ 
9   Let  $\mathbf{v}' \leftarrow \mathbf{v}$ 
10  foreach  $l_i$  in  $H_L$  do
11    if entry  $l_i$  of  $\mathbf{v}$  is 1 then
12      Subtract row  $i$  of  $\mathbf{L}_{\text{rre}}$  from  $\mathbf{v}$  (mod 2)
13       $[\mathbf{u}_L]_i \leftarrow 1$ 
  // if well implemented,  $\mathbf{v}$  should equal  $\mathbf{0}$ 

```

For the function $\mathcal{F}_L$, we need to consider the different degenerate sets. We know that each error e_i can be decomposed in terms of the coset leader $\mathbf{E}_i$, some stabilizer $\mathbf{S}_i$, and some logical operator $\mathbf{L}_i$ (see

Eq. (4.15)). There are 4^k logical operators (including the identity) and each identifies one of the 4^k degenerate sets associated with each syndrome $\hat{\mathbf{s}}$.

Consequently, we can create a one-to-one mapping between the logical operators and the degenerate sets. Since we can determine the tentative coset leader $\mathbf{E}_i$ with $\mathcal{F}_E$, we only need to determine the logical component $\mathbf{L}_i$ that composes our input error pattern.

With this in mind, we continue the approach from Section A. We use the row echelon form of $\mathbf{A}$, that is, $\mathbf{A}_{\text{rre}}$, so we work with $\{S_{i,\text{rre}}\}$ instead. We perform the same procedure to the logical operators. Let $\bar{\mathbf{X}}_i$ and $\bar{\mathbf{Z}}_i$ be the binary row vector representations of $\bar{X}_i$ and $\bar{Z}_i$, respectively (see Eqs. (4.4) and (4.5)), in $[X|Z]$ format. Let $\mathbf{L}$ be the $2k \times 2n$ binary matrix that encodes the original $2k$ minimal logical operators (see Eqs. (4.4) and (4.5)), that act as generators to the 4^k total logical operators. Row i of $\mathbf{L}$ is given either by $\bar{\mathbf{X}}_i$, if $1 \leq i \leq k$, or by $\bar{\mathbf{Z}}_{i-k}$, if $k+1 \leq i \leq 2k$.

Just as with the stabilizers, we know that products of logical operators are also logical operators. Moreover, products of a logical operator with stabilizers correspond to the same logical operator. Since we are working in $\mathbb{F}_2$, adding or subtracting rows of $\mathbf{A}_{\text{rre}}$ or $\mathbf{L}$ is equivalent to multiplying stabilizers and operators in Pauli string form. Considering the augmented $(n+k) \times 2n$ matrix $\begin{bmatrix} \mathbf{A}_{\text{rre}} \\ \mathbf{L} \end{bmatrix}$, by using the rows of $\mathbf{A}_{\text{rre}}$ and $\mathbf{L}$, we may put the $\mathbf{L}$ component in its row echelon form, $\mathbf{L}_{\text{re}}$. We can then put $\mathbf{L}_{\text{re}}$ in reduced row echelon form using only the rows of $\mathbf{L}_{\text{re}}$, yielding $\mathbf{L}_{\text{rre}}$. Note that we cannot use the rows in $\mathbf{L}_{\text{re}}$ to further simplify $\mathbf{A}_{\text{rre}}$, as the resulting rows would no longer correspond to stabilizers. The procedure may be represented as

$$\begin{bmatrix} \mathbf{A}_{\text{rre}} \\ \mathbf{L}_{\text{rre}} \end{bmatrix} = \begin{bmatrix} I_{n-k} & 0 \\ 0 & \mathbf{J}_L \end{bmatrix} \begin{bmatrix} \mathbf{A}_{\text{rre}} \\ \mathbf{L}_{\text{re}} \end{bmatrix} \quad (4.23)$$

$$= \begin{bmatrix} I_{n-k} & 0 \\ 0 & \mathbf{J}_L \end{bmatrix} \begin{bmatrix} I_{n-k} & 0 \\ \mathbf{J}_A & I_{2k} \end{bmatrix} \begin{bmatrix} \mathbf{A}_{\text{rre}} \\ \mathbf{L} \end{bmatrix}, \quad (4.24)$$

with I_p a $p \times p$ identity matrix. The matrices $\mathbf{J}_A$ and $\mathbf{J}_L$ are of size $2k \times (n-k)$ and $2k \times 2k$, respectively, and just like $\mathbf{J}$ in Section A, they represent the linear transformation required to put the matrix in reduced row echelon form.

The resulting rows of $\mathbf{L}_{\text{rre}}$ correspond to $2k$ (possibly different) generators for the logical operators. These generators may no longer satisfy the anti-commutation relations expected of $\bar{X}$ and $\bar{Z}$, but they are not required to. The final $\mathbf{L}_{\text{rre}}$ matrix is such that the columns with the same index as the pivots of $\mathbf{A}_{\text{rre}}$ are zero, and the columns with the pivots of $\mathbf{L}_{\text{rre}}$ have only one non-zero element, its pivot.

For every error pattern e_i with syndrome $\mathbf{s}_i$, we know that its error component $\mathbf{E}_i$ (given by $\mathcal{F}_E$) is such that $e_i \mathbf{E}_i = \mathbf{S}_i \mathbf{L}_i =: e'_i$, for some unknown $\mathbf{S}_i$ and $\mathbf{L}_i$. Consequently, to determine the degenerate set to which e_i belongs, we only need to decompose e'_i into its $\mathbf{S}_i$ and $\mathbf{L}_i$ components. Concretely, we are looking

for the unique row vectors $\mathbf{u}_A$ (of size $n - k$) and $\mathbf{u}_L$ (of size $2k$) such that

$$\mathbf{e}'_{i,[X|Z]} = \begin{bmatrix} \mathbf{u}_A & \mathbf{u}_L \end{bmatrix} \begin{bmatrix} \mathbf{A}_{\text{rre}} \\ \mathbf{L}_{\text{rre}} \end{bmatrix}. \quad (4.25)$$

where $\mathbf{e}'_i$ is exceptionally in $[X|Z]$ format. Since $\mathbf{A}_{\text{rre}}$ and $\mathbf{L}_{\text{rre}}$ are already in reduced row echelon form, finding the two vectors is straightforward. The procedure is described in algorithm 3. Once the $\mathbf{u}_A$ and $\mathbf{u}_L$ row vectors are determined, the $\mathbf{S}_i$ and $\mathbf{L}_i$ components are simply given by (see Eq. (4.13))

$$\mathbf{S}_i = \text{op}(\mathbf{u}_A \mathbf{A}_{\text{rre}}) \quad (4.26)$$

$$\mathbf{L}_i = \text{op}(\mathbf{u}_L \mathbf{L}_{\text{rre}}). \quad (4.27)$$

Alternatively, we may simply skip the computation of $\mathbf{u}_L$ in algorithm 3. Let $\mathbf{v}'$ equal the computed row vector $\mathbf{v}$ just after $\mathbf{u}_A$ is computed, but before the iteration through the pivots of $\mathbf{L}_{\text{rre}}$. Then, we equivalently have $\mathbf{L}_i = \text{op}(\mathbf{v}')$.

The full procedure

$$e_i \implies e_i \mathbf{E}_i \implies \mathbf{u}_L \text{ (or } \mathbf{v}') \implies \mathbf{L}_i \quad (4.28)$$

corresponds to the function $\mathcal{F}_L$.

4.5 Asymptotic regime

We can estimate the optimal performance we can obtain from the decoding procedure by looking at how it performs as the number of extractions considered is increased. We are interested in computing the limit where we have infinite extractions, where the decoding would be optimal. Although this regime is impossible to attain in practice, we expect that, as we increase the number of extractions, the decoding dynamics should converge to the asymptotic corresponding to that optimal case, allowing us to estimate the code's performance in that regime.

We consider the total probability of correction failure P_{total} to correspond to the probability that an error is not completely corrected. That is, the correction chosen does not correspond to the right degenerate set. Note that this definition provides a lower bound on the fidelity F of the resulting quantum state, given by

$$F \geq 1 - P_{\text{total}}, \quad (4.29)$$

since it effectively treats any unsuccessful correction as producing a state with zero fidelity, whereas in practice the uncorrected error may not produce an orthogonal quantum state. Nonetheless, it is a useful lower bound often used in the literature [173, 2], and that we choose to use here as well.

Let P_∞ be the asymptotic limit of P_{total} when the number of syndrome extractions q goes to infinity (see Fig. 4.4). Since there are $L = 2^{2k}$ degenerate sets associated to each syndrome sequence s , then, regardless of the encoding used, for a given s , the probability p of an error having occurred that is in the

most likely degenerate set satisfies $p \geq 1/L$. If e_i^c is the error pattern representative of the most likely degenerate set for s_i , then we must have

$$P_\infty = 1 - \sum_i p(e_i^c e_i \in \mathcal{S} | E_i) p(E_i) \quad (4.30)$$

$$\leq 1 - \sum_i (1/L) p(E_i) \quad (4.31)$$

$$= 1 - 1/L \quad (4.32)$$

$$= 1 - 2^{-2k}, \quad (4.33)$$

with equality achieved for the case of maximum noise entropy.

When comparing two setups, A and B , with q_A and $q_B = q_A + h$ ($h > 0$) syndrome extractions, respectively, we necessarily have

$$P_{\text{total}}(A) \leq P_{\text{total}}(B) \quad (4.34)$$

since introducing h additional noisy syndromes extractions introduces additional errors in the model, which may or may not be correctable.

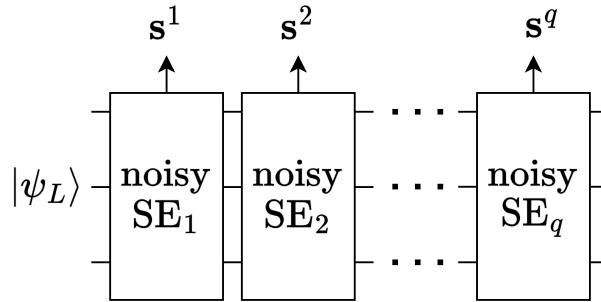


Figure 4.4: Model considered to compute the asymptotic regime, where $q \rightarrow \infty$.

We are interested in determining the failure probability P_{failure} induced by a single additional syndrome extraction, preceded and succeeded by an arbitrarily high number of extractions. In the context of fault-tolerance, we wish to determine if, for a given p , P_{failure} increases or decreases with increasing qubit count n . We expect, for low (resp. high) values of p , P_{failure} decreases (resp. increases) with n , with a phase transition at some $p_{\text{threshold}}$, to be determined.

Since we do not have direct access to P_{failure} , it must be computed from the measured value of P_{total} . We develop an effective model to quantitatively relate the two quantities.

Consider a variant of the B setup above, labeled B' , where the first q_A extractions are solely used to identify and correct errors stemming from implementing those extractions, and similarly, the last h extractions are solely used to deal with errors resulting from the h extractions, for a total of $q_B = q_A + h$, as before. In other words, in the B' setup, the procedure is partitioned into to separate and independent decoding processes.

We expect

$$P_{\text{total}}(B) \leq P_{\text{total}}(B'). \quad (4.35)$$

since, in the B setup, the last h extractions also provide information about errors in the previous extractions, which can lead to a more successful decoding. The B' setup does not use this information.

In fact, if an error occurs at extraction q_A , we expect that a lower number h of subsequent extractions will result in higher failure rates (per extraction), since the decoding process has less information to correctly identify the error. Since we are interested in the limiting case where the number of subsequent extractions is infinite (for any given extraction where an error can occur), we wish to discount the effect of limited syndrome information from the calculated probability of failure P_{failure} . We label the errors that could have been successfully corrected with $h \rightarrow \infty$ but were not with low h *escaped errors*.

For low q , we expect (see [180]) that escaped errors (in particular errors with $\tilde{\mathbf{s}} = 0$ but $\hat{\mathbf{s}} \neq 0$) can be identified with additional syndromes from more subsequent extractions, so that each new extraction reduces the number of escaped errors by a factor δ . Of course, new extractions also introduce new escaped errors, and errors at the end extractions are more likely to escape correction.

Consider a setup with q extractions as a setup with $q - 1$ extractions preceded by one additional extraction. The errors of this additional extraction will be detected by the $q - 1$ subsequent extractions, so that only a factor δ^{q-1} escape through the whole setup incorrectly identified. Therefore, applying the reasoning recursively for q extractions, we expect the probability of an error passing through the extractions undetected to be given by

$$P_u(p, q) \simeq P_u(p, q - 1) + \delta^{q-1} P_u(p, 1) \quad (4.36)$$

$$\simeq P_u(p, 1) \frac{1 - \delta^q}{1 - \delta}. \quad (4.37)$$

For $q \gg 1$, we expect that the probability of successful correction with q extractions $P_{\text{succ}}(p, q)$ will be approximately the probability of success with $q - 1$ extractions times the per-extraction probability of success (which accounts for the additional extraction). We have

$$P_{\text{succ}}(p, q) \simeq P_{\text{succ}}(p, q - 1)(1 - P_{\text{failure}}(p)) \quad (4.38)$$

$$= (1 - P_{\text{failure}}(p))^q \quad (4.39)$$

$$\text{with } P_{\text{succ}}(p, 0) = 1. \quad (4.40)$$

However, since P_{failure} does not incorporate escaped errors, the true probability of success P'_{succ} is given by

$$P'_{\text{succ}}(p, q) = P_{\text{succ}}(p, q)(1 - P_u(p, q)), \quad (4.41)$$

yielding

$$P_{\text{total}}(p, q) \simeq 1 - (1 - P_{\text{failure}}(p))^q (1 - P_u(p, q)). \quad (4.42)$$

If $\delta \ll 1$, which is expected, then, when using $q \gg 1$ total extractions, we may approximate $P_u(p, q) \simeq P_u(p, \infty)$, leading to

$$P_{\text{total}}(p, q) \simeq 1 - C(p)(1 - P_{\text{failure}}(p))^q, \quad (4.43)$$

$$\text{with } C(p) := 1 - P_u(p, \infty), \quad (4.44)$$

where C incorporates the escaped errors. See Fig. 4.5 for a numerical verification of this model. Once more, note that we discount these errors, and their probability P_u , from the failure probability P_{failure} , since we are interested in the per-extraction failure probability, and P_u constitutes a global effect. As previously indicated, if a syndrome extraction is followed by $h \rightarrow \infty$ extractions, then the probability that an error stemming from that extraction goes completely undetected is $\delta^h P_u(p, 1) \rightarrow 0$.

P_{failure} is the asymptotic contribution of each syndrome extraction to the probability of failure. We have that, by fitting, for each p ,

$$Y = mX + b, \quad (4.45)$$

$$\text{with } Y := \log(1 - P_{\text{total}}(p, q)), \quad X := q, \quad (4.46)$$

then, from Eq. (4.43),

$$P_{\text{failure}}(p) = 1 - e^m, \quad C(p) = e^b, \quad (4.47)$$

for each p considered.

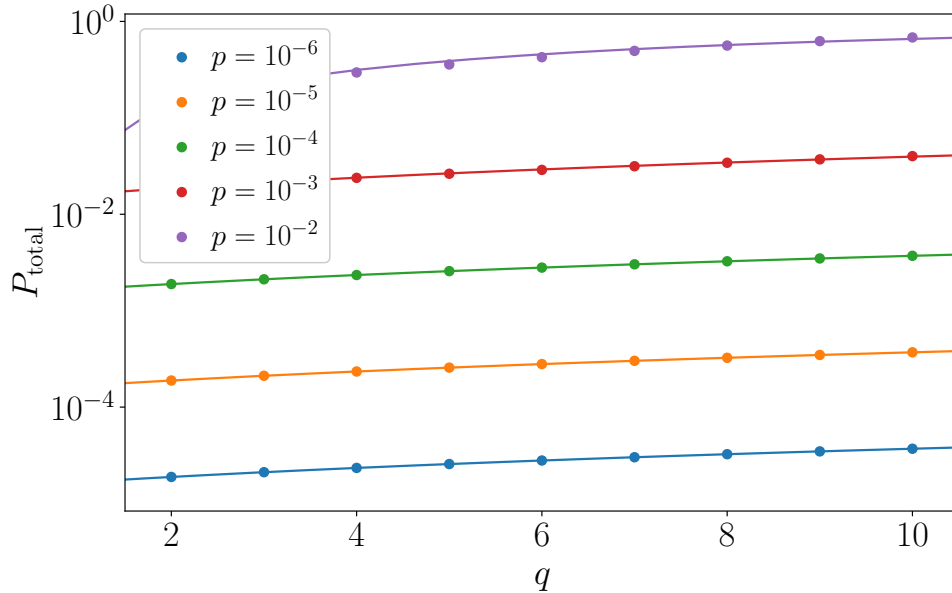


Figure 4.5: Experimental P_{total} , and resulting fit by Eq. (4.43). We observe the experimental data matching the qualitative description of the theoretical analysis, with $R^2 > 0.999$ for all cases.

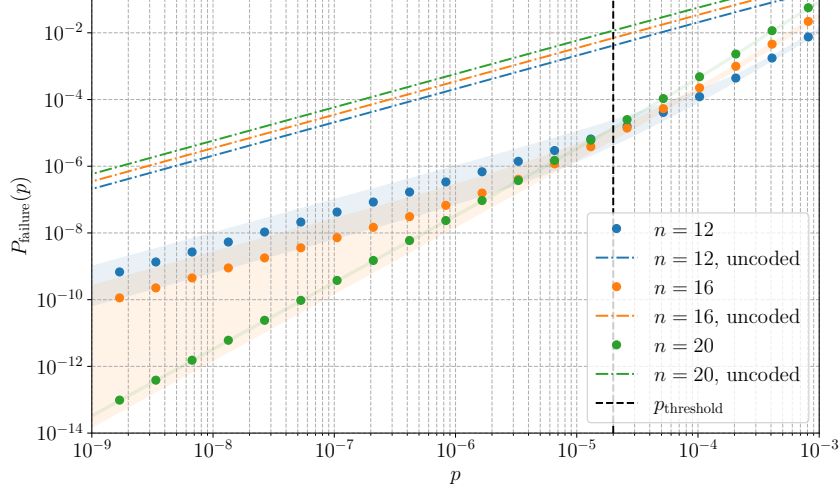


Figure 4.6: Performance of QRLCs with $k = 1$, using fault-tolerant QGRAND. We observe that the asymptotic version presents a threshold around $p \simeq 2 \times 10^{-5}$. The shaded regions denote a 60% confidence interval (from 50 samples), and the dots are the median performance observed. The vertical line indicates the $p_{\text{threshold}}$ value. The dashed lines at the top correspond to the uncoded case, which is plotted for reference.

The results are presented in Fig. 4.6. We observe $p_{\text{threshold}} \simeq 2 \times 10^{-5}$, suggesting the QGRAND technique can be used in the fault-tolerant regime. Nonetheless, we note that this optimal decoding procedure is not scalable for high entropy noise models, such as those given by $p \gtrsim p_{\text{threshold}}$. Therefore, to use these techniques, we are forced to either simplify our model, turning the decoding procedure suboptimal, or to focus on a regime where it can be applied in practice.

For reference, we also estimate the performance of the uncoded case. Considering the number of noisy CNOT gates present in a circuit with q iterations to be $N_{\text{CNOT}}(q)$, we get

$$P_{\text{failure,uncoded}}(p) \simeq 1 - (1 - p)^{N_{\text{CNOT}}(1)}. \quad (4.48)$$

N_{CNOT} doesn't include preparation and measurement errors, as these don't propagate throughout the circuit.

The simulation is performed for $\omega < \omega_{\text{max}}$, and approximated for the higher ω values. For high ω , we use the reasonably accurate assumption that error patterns are uniformly assigned to the syndrome sequences. Therefore, before computing P_{failure} , we modify P_{total} with the correction

$$P'_{\text{total}} = P_{\text{total}} - \frac{P(\omega > \omega_{\text{max}})}{2^{n+k}}, \quad (4.49)$$

where 2^{n+k} is the number of degenerate sets associated with each syndrome sequence. The correction is negligible for high n , but may play a noticeable role for $p \simeq 1$. Based on our analysis in Section K, we find that for the n values considered, the minimum error order that non-trivially contributes to the

failure probability never exceeds 2. This justifies using $\omega_{\max} = 2$ for exact simulation in the parameter regimes studied. The effect of higher orders can be reasonably accounted for using Eq. (4.49).

4.6 Discussion and conclusions

In this work, we extend the decoding procedure for QRLCs introduced in [2] to explicitly account for error degeneracy. Consequently, our technique constitutes a maximum likelihood decoding procedure, which is guaranteed to be optimal. We analyze the fault-tolerant characteristics of QRLCs with the presented decoding technique, by accounting for preparation, measurement, and gate errors in the syndrome extraction procedure itself, and observe a $p_{\text{threshold}} \simeq 2 \times 10^{-5}$ in the asymptotic limit. To the best of our knowledge, this work presents the first fault-tolerant decoding technique specifically applying QGRAND to QRLCs in the presence of preparation, measurement, and gate errors during syndrome extraction.

We note that this decoding procedure is not equivalent to finding the lowest weight error pattern associated with each syndrome, as might be done by more standard algorithms, since a faulty CNOT gate error effect can propagate considerably through the circuit before being possible to detect it, so that, by the time it is detected, its error pattern is no longer low weight.

In this work, we have removed the channel errors present in [2] and considered only the main error sources associated with the syndrome extraction process. In particular, we considered preparation and measurement errors in the ancilla qubits, and two-qubit gate errors. Although this is a common approach to take when studying the fault-tolerance capabilities of different codes [173], it leads to unrealistic results for high code rates. In the limit when $R \rightarrow 1$, the syndrome extraction process has a negligible number of minimal stabilizers, and as a result negligible error sources, under this noise model. Consequently, in this regime, higher code rates lead to lower P_{failure} , not because of better correction capabilities, but because error sources decrease faster than the correction capabilities do, as the code rate increases.

Moreover, we have analyzed the asymptotic regime of infinite syndrome extractions. Although impractical, these asymptotic results enable us to study the behavior of the optimal decoding procedure, as previously described. Nonetheless, practical limitations might impose suboptimal steps in the decoding process, and obviously a finite number of syndrome extractions.

To account for these limitations, we must consider that, in practice, there are non-trivial computing steps performed between syndrome extractions (such as logical gates for quantum computing, and Bell-pair creation for quantum communication) that introduce their own errors independently from the syndrome extraction steps. When accounting for this additional error source, we expect the pathological behavior for high code rates to disappear. In future work, we intend to study these more practical regimes. Furthermore, we assumed that all-to-all connectivity (between any of the n qubits) is possible in practice. This assumption is required for the scaling results in [141], and is used in this work. Nonetheless, it may be dropped for practical reasons, as the more recent results in [112] suggest.

As previously mentioned, the noise guessing decoding procedure is expected to be viable only in situations

of low noise entropy and low n . Even disregarding the limitations imposed by the asymptotically large number of syndrome extraction, it is also the case that the noise entropy increases rapidly as $n \rightarrow \infty$ and $k = 1$, as considered for the fault-tolerance analysis. This is a known limitation of the decoding procedure. For this reason, and for the fact that better known codes, such as surface codes, have higher $p_{\text{threshold}}$ values, we do not expect the decoding procedure described in this work to be competitive in those regimes. Although it remains to be confirmed in future work, we conjecture that, given the optimal decoding properties of the described procedure, it may be worthwhile to employ in scenarios where code versatility is needed, the noise statistics are not approximately fixed, and the code rate is desired to be very high. In those cases, we expect the method to have similar use cases to those previously described in [2], as the additional gates used by the syndrome extraction process would not have a strong impact on decoding performance.

Beyond the straightforward approach described in algorithm 2, we may also wish to sacrifice the decoding optimality for the sake of decoding throughput, or lower hardware requirements, rendering the decoding process more easily scalable. This can be achieved with either known techniques, such as compressive sensing or deep learning, or with more straightforward approaches, such as greedy variants of the method. For instance, for the Bernoulli noise model in this work, if we take the coset leader to be the first error pattern associated with a syndrome, we will end up with a suboptimal version of algorithm 2, equivalent to [2], but one which reduces the memory requirements by as much as a factor of 4^k . There are also specific simplifications that can be used to implement the decoding procedure faster when the noise model has some exploitable structure, as is the case with the noise model considered here. We plan to cover some of these approaches in future work.

4.7 Supporting Information

A Reducing stabilizer weight

Although we are working with quantum random linear codes, which have little exploitable structure *a priori*, we note that the minimal stabilizers can be efficiently chosen to have weight lower than the average of $3n/4$. To do so, we may take the original minimal stabilizer arising from the technique described in [2], represent them with the parity check matrix, and put the matrix in canonical form, which is equivalent to reduced row echelon form. The new simpler minimal stabilizers correspond to the rows of the resulting matrix.

If the pivots of the matrix in reduced row echelon form are all in the first n columns, then this technique reduces the weight of the non- Z components of each minimal stabilizer to at most $1 + k$, and $1 + k/2$ on average. If k is low and n is large, this technique can result in a considerable reduction in the weight of the minimal stabilizers, as their average weight goes from $3n/4$ to $3k/4 + (n - k - 1)/2 + 1 = (2n + k + 2)/4$. Instead of each term being equally distributed between I, X, Y , and Z as before, here only the indices greater than $n - k$ maintain that distribution, and we have, for S_i , index i equally distributed between X and Y , and index $j \leq n - k, j \neq i$ equally distributed between I and Z .

This structure simplifies the application of the syndrome extraction process, as it reduces the number of CNOT gates from $\sim 3n(n-k)/4$ to $(2n+k+2)(n-k)/4$, and similarly reduces the number of 1-qubit gates. Despite these benefits, in our numerical analysis we have not assumed such an approach was taken in the circuit implementation, in order not to introduce unwarranted structure in the noise statistics, as we are interested in analyzing the more general scenario.

Nonetheless, as explained in Sections A and B, we have used this simplification in our decoding implementation, when given the noise statistics associated with the unsimplified stabilizers. As they generate the same stabilizer group, they are mathematically equivalent for the same given noise statistics, and we expect either approach to lead to similar numerical results.

B Degeneracy analysis

Regarding degeneracy, following the notation introduced in Section 4.3, we consider errors E_i and E_j to be degenerate if and only if $e_i e_j \in \mathcal{S}$. Nonetheless, we may describe three types of degenerate errors, all prevalent in the noise model considered in this work.

Identical errors. These are errors such that $e_i = e_j$ and $e_i^s = e_j^s$. For example, since we are considering the model implementation where the CNOT gates of the conditional stabilizer are implemented in succession, with the control always being the ancilla qubit (see Fig. 4.1), then any error of the form $Z_c I_t$ (with c and t the control and target qubit indices, respectively) will commute with subsequent CNOT gates controlled by the qubit c . Therefore, this component does not add any error terms to the main n qubits, and instead simply negates the measured ancilla qubit. As a result, for any error term without this component, there is an error term with this component where the error pattern in the main n qubits is the same, and the ancilla qubit is simply negated. Given this degeneracy, the problem reduces to two scenarios: one where there is an even number of such errors, where the syndrome is unaffected, and one with an odd number of such errors, where the ancilla bit is negated. Among these two classes, all errors are not only degenerate, but identical.

Pseudo-identical errors. Besides the identical errors, we also observe cases where $e_i = e_j$ but $e_i^s \neq e_j^s$ (with $\text{comp}_X(e_i^s e_j^s) = \mathbf{0}$, otherwise the errors would have different syndromes).

Non-identical errors. We also have the more general case where $e_i \neq e_j$ (with either $e_i^s = e_j^s$ or $e_i^s \neq e_j^s$), while still retaining $e_i e_j \in \mathcal{S}$.

See Fig. 4.7 for an example. There, the code's sole non-trivial stabilizer is $X_1 X_2$. We have $e_i^s = e_a^s = a_c^s = X$, $e_b^s = Y$ (ignoring the phase), $e_i = e_a = a_b = I_1 I_2$, and $e_c = X_1 X_2$.

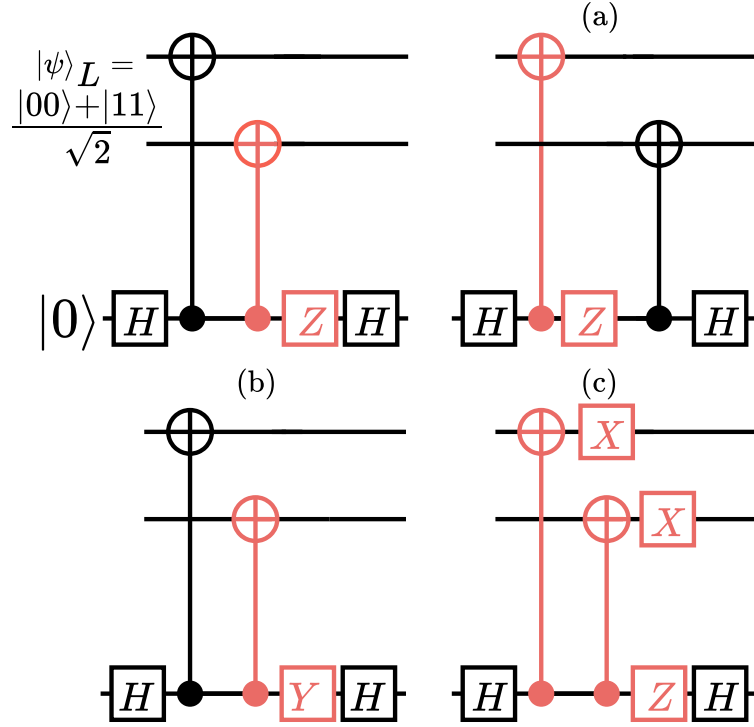


Figure 4.7: For the example code in Fig. 4.2, we may observe the three forms of degenerate errors. Considering E_i as the error showcased in the top left circuit, the identical, pseudo-identical, and non-identical errors can be seen in subfigures (a), (b), and (c), respectively.

C Analytical threshold without degeneracy

For this analysis, we disregard preparation and measurement errors, as the derivation can be readily extended to the complete model.

Keeping k constant, the value of n will determine the number of CNOT gates (N_{CNOT}) in the circuit, and the code's correction capabilities. We may estimate its performance by considering the approximation given by random ideal codes, as in [2].

For $S =: 2^{n-k} \gg 1$ and N the number of distinct errors, the equation

$$f = \frac{S}{N+1} \left[1 - \left(1 - \frac{1}{S} \right)^{N+1} \right] \quad (4.50)$$

may be approximated by

$$f \simeq \frac{1 - e^{-r}}{r}, \quad r := \frac{N+1}{S} \quad (4.51)$$

$$\Rightarrow f \simeq \frac{1 - (1 - r + r^2/2)}{r} = 1 - \frac{r}{2} \text{ for } r \ll 1, \quad (4.52)$$

since, from Eq. (4.50), we have

$$f = \frac{1}{r} \left[1 - \left[\left(1 - \frac{1}{S} \right)^S \right]^r \right] \quad (4.53)$$

$$\simeq \frac{1}{r} [1 - (e^{-1})^r]. \quad (4.54)$$

Since $S = 2^{n-k}$, this indicates that

$$f \geq 1 - \epsilon \quad (4.55)$$

$$\iff N \lesssim 2^{n-k+1} \epsilon. \quad (4.56)$$

W.l.o.g., consider that each CNOT gate can only suffer from a specific error, instead of 15. For the Bernoulli noise model we are considering, the error order ω is given by the binomial distribution

$$\omega \sim \mathcal{B}(N_{\text{CNOT}}, p) \quad (4.57)$$

For p fixed, and as $N_{\text{CNOT}} \rightarrow \infty$, this distribution can be approximated by

$$\mathcal{N}(N_{\text{CNOT}} p, N_{\text{CNOT}} p(1-p)) \quad (4.58)$$

using the De Moivre-Laplace theorem.

Suppose we start by correcting the lowest order errors (which are more likely to occur), and we wish to correct the errors up to order $\omega_{\max}$ such that we have the probability $(1 - \epsilon)$ of correcting an error we observe. For a normal distribution, this is given by the quantile function

$$Q(p) = \mu + \sigma \sqrt{2} \text{erf}^{-1}(2p - 1). \quad (4.59)$$

The error function erf cannot be easily approximated. Nonetheless, we may observe that it can be approximated by

$$\text{erf}(x) \simeq 1 - \alpha \exp(-(x - \beta)^2), \text{ for } x \gg 1, \quad (4.60)$$

leading to

$$\text{erf}^{-1}(x) \simeq \beta + \sqrt{\log \left(\frac{\alpha}{1-x} \right)} \text{ for } 1-x \ll 1. \quad (4.61)$$

For simplicity, we consider $\alpha = 1, \beta = 0$, which does not meaningfully affect our conclusions here. The quantile function then becomes

$$Q(1 - \epsilon) \simeq \mu + \sigma \sqrt{2} \left(\beta + \sqrt{\log \left(\frac{\alpha}{2\epsilon} \right)} \right) \quad (4.62)$$

$$\simeq \mu + \sigma \sqrt{2 \log \left(\frac{1}{2\epsilon} \right)}. \quad (4.63)$$

with

$$\mu = N_{\text{CNOT}}p \quad (4.64)$$

$$\sigma = \sqrt{N_{\text{CNOT}}p(1-p)} \quad (4.65)$$

$$\simeq \sqrt{N_{\text{CNOT}}p}. \quad (4.66)$$

Now that we have $\omega_{\text{max}} = Q(1-\epsilon)$, we need to estimate the number of errors $\tilde{N}$ up to order ω_{max} , given by

$$\tilde{N} = \sum_{j=0}^{\omega_{\text{max}}} \binom{N_{\text{CNOT}}}{j}. \quad (4.67)$$

Unfortunately, there is no closed form expression for this value. However, for $\omega_{\text{max}} \ll N_{\text{CNOT}}$, this is roughly equal to

$$\tilde{N} \simeq \binom{N_{\text{CNOT}}}{\omega_{\text{max}}} \quad (4.68)$$

$$\simeq \frac{N_{\text{CNOT}}^{N_{\text{CNOT}}}}{\omega_{\text{max}}^{\omega_{\text{max}}} (N_{\text{CNOT}} - \omega_{\text{max}})^{N_{\text{CNOT}} - \omega_{\text{max}}}} \sqrt{\frac{N_{\text{CNOT}}}{2\pi\omega_{\text{max}}(N_{\text{CNOT}} - \omega_{\text{max}})}}. \quad (4.69)$$

This may be simplified down to

$$\tilde{N} = 2^{N_{\text{CNOT}} h_2(\omega_{\text{max}}/N_{\text{CNOT}})} \sqrt{\frac{1}{2\pi\omega_{\text{max}}}}, \quad (4.70)$$

where h_2 is the Shannon entropy.

From Eqs. (4.56) and (4.70), we therefore conclude that

$$\log N \sim \tilde{O}(n) \quad (4.71)$$

$$\log \tilde{N} \sim \tilde{O}(N_{\text{CNOT}}), \quad (4.72)$$

where $\tilde{O}$ denotes Big-O notation up to log factors. Since N indicates the code's correction capabilities, while $\tilde{N}$ indicates the necessary number of errors that the code needs to correct to preserve P_{failure} , then we must have $N \gtrsim \tilde{N}$ and consequently $N_{\text{CNOT}} \sim \mathcal{O}(n)$. This is verified for some common codes, such as surface codes, but for our implementation we have

$$N_{\text{CNOT}} = \frac{3}{4}n(n-k) \sim \mathcal{O}(n^2), \quad (4.73)$$

so we conclude that, if all errors are non-degenerate, the code does not have a visible $p_{\text{threshold}}$, since an increase in n increases P_{failure} , regardless of p .

However, we actually observe a threshold for QRLCs. This is thanks to the fact that the noise statistics given by the noise model of Section 4.2 actually lead to a very high number of degenerate errors. Therefore, in practice, the number of distinct errors grows with $\mathcal{O}(n)$ and not $\mathcal{O}(n^2)$ as indicated by the analysis

above. We later confirm this scaling for escaped errors, in Section K.

With this insight in mind, we modify the QGRAND algorithm in [2] to account for the possibility of degenerate errors. The modified algorithm is presented in Section 4.4.

D Applying error correction

Certain QECCs, such as surface codes, are designed so that a physical error correction is actually unnecessary to implement, as all changes can be made in software, classically [173]. If the whole quantum circuit is unitary, then this procedure can actually be implemented in general: instead of correcting the error, we leave the affected state as is and simply XOR any subsequent syndrome $\mathbf{s}$ with the identified error syndrome $\mathbf{e}$, that is, $\mathbf{s} \mapsto \mathbf{s} \oplus \mathbf{e}$. As a result of this, we only need to keep track of these detected errors classically, in order to correct the subsequent syndromes.

In any case, since the correction portion is always single-qubit gates, we assume that their contribution to the total error is negligible, so we can disregard this trick for now, for the sake of simplicity. As a result, the procedure to apply the error correction is the same as in [2].

E Parallel decoding

The decoding procedure described in Section 4.4 can be performed in parallel. If there are W parallel workers available, we start by splitting the entries in $\mathcal{N}$ into W parts of equal size, labeling each $\mathcal{N}_w$, $1 \leq w \leq W$. Each set $\mathcal{N}_w$ is then processed independently by an individual procedure according to the procedure described in the Section 4.4, thereby yielding the data table D_w .

All the W data tables D_w may then be merged to generate the full data table D , from which the decoding table T can be straightforwardly computed. See algorithm 4 for a description of the parallel decoding procedure.

Algorithm 4: Parallel decoding

Input: $\mathcal{N}, W$

Output: A decoding table T

```

1 Procedure ParallelDecoding( $\mathcal{N}, W$ ):
2   Initialize empty decoding table  $T$ 
3   Split  $\mathcal{N}$  into  $W$  sets equal in size, labeled  $\mathcal{N}_w$  ( $1 \leq w \leq W$ )
4   foreach parallel worker  $w$  do
5      $D_w \leftarrow \text{DATA}(\mathcal{N}_w)$ 
6    $D \leftarrow \bigcup_w D_w$ 
7   foreach entry  $s$  in  $D$  do
8     Set  $T[s]$  as the pattern  $e^d$  with highest  $p$  in  $D[s]$ 

```

F Full decoding example

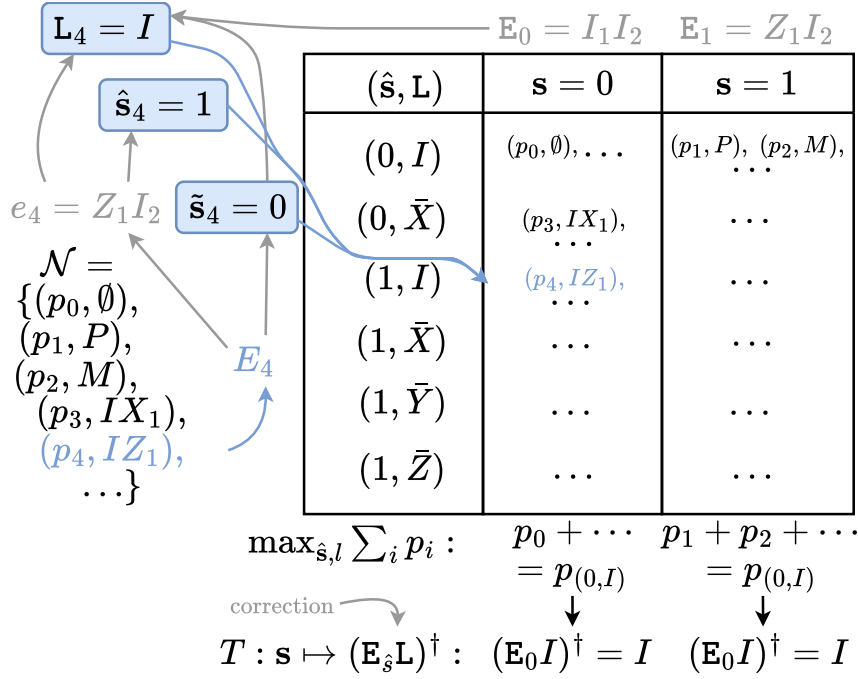


Figure 4.8: Optimal decoding of the code in Fig. 4.2 (but with only one syndrome extraction), as described in algorithm 2.

For the example in Fig. 4.8, we consider the encoding gate to be $U = \text{CNOT}(2, 1)H_2$, with the starting qubit at index 1. P and M stand for preparation and measurement error in the only ancilla qubit a , respectively. AB_j is the error corresponding to “error $A_a B_j$ occurred at CNOT gate j ”. Following algorithm 2, we iterate through the errors in $\mathcal{N}$ and, using the functions $\mathcal{F}_E$ and $\mathcal{F}_L$, determine where to put them in the table D . Once we have iterated through all errors, we compute the optimal $(\hat{\mathbf{s}}, \mathbf{L})$ entry, or alternatively, e^d representative, and delete the remaining entries, yielding the decoding table T .

Given U , and following Eqs. (4.3) to (4.5), the minimal stabilizer is $S_1 = X_1 X_2$, and the logical operators are $\bar{X} = I_1 X_2$ and $\bar{Z} = Z_1 Z_2$. This choice of encoding leads to minimal stabilizers and logical operators such that the augmented matrix is

$$\begin{bmatrix} \mathbf{A} \\ \mathbf{L} \end{bmatrix} = \begin{bmatrix} \mathbf{A} \\ \bar{\mathbf{X}} \\ \bar{\mathbf{Z}} \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 \end{bmatrix} = \begin{bmatrix} \mathbf{A}_{\text{rre}} \\ \mathbf{L}_{\text{rre}} \end{bmatrix}, \quad (4.74)$$

since, in this simple example, we already have $\mathbf{A} = \mathbf{A}_{\text{rre}}$ and $\mathbf{L} = \mathbf{L}_{\text{rre}}$, so $\mathbf{J} = I$. Here, $\mathbf{A}$, $\bar{\mathbf{X}}$, and $\bar{\mathbf{Z}}$ are the binary representations of the minimal stabilizers and logical operators, in $[X|Z]$ format.

Following the procedure in Section A, we have

$$\mathbf{E}_0 = I_1 I_2 \quad \mathbf{E}_1 = Z_1 I_2. \quad (4.75)$$

Let's consider the full procedure in Section 4.4 applied to the specific error in Fig. 4.2, such as

$$E = \text{"Error } I_a Z_1 \text{ in CNOT gate 1"} \quad (4.76)$$

Associated to this error, we have the quantities

$$\tilde{\mathbf{s}} = 0 \quad (4.77)$$

$$e = Z_1 I_2 \quad (4.78)$$

$$\mathbf{e} = \begin{bmatrix} 0 & 0 & 1 & 0 \end{bmatrix} \quad (4.79)$$

$$\hat{\mathbf{s}} = \mathbf{e} \mathbf{A}^T = 1 \quad (4.80)$$

$$\hat{\mathbf{s}}_{\text{rre}} = \hat{\mathbf{s}} = 1 \quad (4.81)$$

$$e \mathbf{E}_{\hat{\mathbf{s}}} = I_1 I_2 = \text{SL} \quad (4.82)$$

$$\implies \mathbf{L} = I_1 I_2 = I. \quad (4.83)$$

Applying the procedure in Section 4.4 and algorithm 2, we obtain the tentative decoding table in Table 4.2. In the table, for the sake of simplicity, we represent preparation and measurement errors by P and M , respectively. AB_i represents the CNOT gate error where error $A_a B_i$ occurs just after the noiseless CNOT gate. For simplicity, compound error are not shown.

Table 4.2: Decoding for the code in Fig. 4.2.

$(\hat{\mathbf{s}}_{\text{rre}}, \mathbf{L})$	e^d	$\mathbf{s} = 0$	$\mathbf{s} = 1$
$(0, I)$	$I_1 I_2$	$\emptyset, X I_2, X X_1$	$P, M, Z I_2, Z I_1, Y X_1, Y I_2$
$(0, \bar{X})$	$I_1 X_2$	$I X_1, X I_1, I X_2, X X_2$	$Z X_1, Y I_1, Z X_2, Y X_2$
$(0, \bar{Y})$	$Z_1 Y_2$		
$(0, \bar{Z})$	$Z_1 Z_2$		
$(1, I)$	$Z_1 I_2$	$I Z_1, X Y_1$	$Z Z_1, Y Y_1$
$(1, \bar{X})$	$Z_1 X_2$	$X Z_1, I Y_1$	$Y Z_1, Z Y_1$
$(1, \bar{Y})$	$Z_1 Y_2$	$I Y_2, X Y_2$	$Z Y_2, Y Y_2$
$(1, \bar{Z})$	$I_1 Z_2$	$I Z_2, X Z_2$	$Z Z_2, Y Z_2$

The preparation and measurement errors have probability p of occurring, and the gate errors have probability $p/15$. If we assume that it is very unlikely that no error has occurred (i.e. p is high), then the final decoding table (when only considering errors of order 1) is given by Table 4.3. In the table, if $\mathbf{s} = 0$ is measured, the most likely degenerate set to have occurred is $(0, \bar{X})$, with probability $4p/15$ (see Table 4.2). We assume that p is such that the no-error case is unlikely, otherwise $(0, \bar{I})$ would be the most likely degenerate set. If $\mathbf{s} = 1$ is measured, the most likely degenerate set to have occurred is $(0, \bar{I})$, with probability $34p/15$. For each case, the error pattern that should be applied to the quantum state to correct the error is given by $e^d = \mathbf{E}_{\hat{\mathbf{s}}} \mathbf{L}$.

Table 4.3: Final decoding table for Fig. 4.2.

$\mathbf{s} = 0$	$\mathbf{s} = 1$
$(0, \bar{X})$	$(0, I)$
$\downarrow$	$\downarrow$
$\mathbf{E}_0 \bar{X} = I_1 X_2$	$\mathbf{E}_0 I = I_1 I_2$

G Simplified noise statistics

When considering a Bernoulli noise model, such as in Section H, including preparation and measurement errors into the noise model breaks some of the structure of the noise statistics, since not all base errors will be equally likely. It also makes the decoding process harder to simulate. In the simpler setup where only gate errors occur, if we have g CNOT gates affected with an error, with each error having probability $p_{\text{CNOT}}/15$ of occurring, then the number of errors and their individual probability would be given by

$$N_E(g) = 15^g \binom{N_{\text{CNOT}}}{g} \quad (4.84)$$

$$P(g) = \left(\frac{p_{\text{CNOT}}}{15}\right)^g (1 - p_{\text{CNOT}})^{N_{\text{CNOT}} - g}. \quad (4.85)$$

These formulas stem from the fact that each CNOT gate has 15 associated errors, and only one of these may occur at a time. For preparation and measurement errors, there is only one error pattern per ancilla qubit: either there is a bit flip, or there isn't. If there are A ancilla qubits (generally, $A = n - k$), and a preparation or measurement error occurs with probability p_M , these same quantities are given by

$$N_E(\omega) = \sum_{g=0}^{\omega} 15^g \binom{N_{\text{CNOT}}}{g} \binom{2A}{\omega - g} \quad (4.86)$$

$$P(\omega, g) = \left(\frac{p_{\text{CNOT}}}{15}\right)^g (1 - p_{\text{CNOT}})^{N_{\text{CNOT}} - g} p_M^{\omega - g} (1 - p_M)^{2A - \omega + g}, \quad (4.87)$$

where ω indicates the total error order, and g indicates the error order when ignoring preparation and measurement errors. Note that these expressions are somewhat more complicated. In particular, for errors of order ω , we now need to keep track of the distinct number of CNOT (g) and preparation/measurement ($\omega - g$) base errors that occurred, instead of just one parameter.

Fortunately, if we are using optimal decoding (that properly accounts for degenerate errors), there is a quick-and-dirty way to mimic the simpler noise statistics associated with only having gate errors. If we use $p_M = p_{\text{CNOT}} =: p$ (which is a relatively common choice in the literature, see [173], and the one used in Section 4.2), then we can consider that each preparation and measurement error is a CNOT gate error. Instead of there being only one error per qubit (corresponding to the possible bit flip), we consider that there are 15, all equal in nature, and each occurring with probability $p_M/15$. These cloned errors will be degenerate among themselves, so the optimal decoding procedure will analyze this setup correctly. The total number of error patterns $N_E(\omega)$ will be overcounted, but we never actually use it directly for

the decoding procedure, so the overcounting does not constitute an issue. In this formulation, we may pretend that we have no preparation and measurement errors, and that we have instead

$$\tilde{N}_{\text{CNOT}} := N_{\text{CNOT}} + 2A \quad (4.88)$$

CNOT gates in the syndrome extraction circuit. The probability associated with each error will be correct, yielding

$$P(\omega) = \left(\frac{p}{15}\right)^\omega (1-p)^{\tilde{N}_{\text{CNOT}}-\omega}. \quad (4.89)$$

H Bernoulli noise model improvements

Algorithm 5: Data table $\tilde{D}_1$ for Bernoulli noise

Input: $\mathcal{B}$

Output: A data table $\tilde{D}_1$

```

1 Procedure BernoulliData( $\mathcal{B}$ ):
2   Initialize empty data table  $\tilde{D}_1$ 
3    $\mathbf{n}_0 \leftarrow (1, 0)$ 
4   Store  $\{I : \mathbf{n}_0\}$  in  $\tilde{D}_1[\{\mathbf{0}, \dots, \mathbf{0}\}]$ 
5   foreach  $E_i^B$  in  $\mathcal{B}$  do
6     Compute  $e_i$ ,  $\hat{\mathbf{s}}_i$ , and  $s_i$ 
7     Compute  $L_i$  associated with  $e_i$ 
8     Compute  $e_i^d$ 
9     if  $e_i^d$  not in  $\tilde{D}_1[s_i]$  then
10       $\mathbf{n}_i \leftarrow (0, 0)$ 
11       $\mathbf{n}_i(1) \leftarrow 1$ 
12      Store  $\{e_i^d : \mathbf{n}_i\}$  in  $\tilde{D}_1[s_i]$ 
13    else
14       $\mathbf{n}_i(1) \leftarrow \mathbf{n}_i(1) + 1$ 
15      Update  $\{e_i^d : \mathbf{n}_i\}$  in  $\tilde{D}_1[s_i]$ 

```

For the special case where each CNOT gate has the same probability p of suffering an error, as described in Section 4.2, the decoding procedure can be made much more efficient. This procedure may be adapted to other Bernoulli-like noise statistics, but here we focus on this error model. We can optimize this decoding process in order to avoid having to iterate through all compound errors. Since the technique relies on the inherent structure of errors with the same probability, here we employ the reformulation detailed in Section G to treat preparation and measurement errors as additional gate errors.

The procedure for the list of base errors (with $\omega = 1$) is similar to the one described in the beginning of Section 4.4. Instead of using the full noise statistics $\mathcal{N}$, instead we consider a list of base errors

$$\mathcal{B} := \{E_1^B, E_2^B, \dots\}. \quad (4.90)$$

Given the Bernoulli noise model, all base errors have the same associated probability of occurring, given by $p/15$, so it doesn't need to be stored in $\mathcal{B}$ (we exclude the no error case). Once we have the data table D_1 for $\omega = 1$, we can start to optimize the analysis for compound errors. Instead of iterating through compound errors individually, we iterate through the degenerate sets obtained in the $\omega = 1$ step. We also preserve the probability associated with observing an error pattern from each degenerate set in the table. For $\omega = 1$, and given the reformulation of Section G, all errors $\hat{E}_i^B$ may be considered to have a probability $P(1)$ of occurring (see Eq. (4.89)), so the probability associated with each degenerate set in the data table is given by

$$p_i := N_i P(1), \quad (4.91)$$

where N_i is the number of errors E_i that can be corrected by applying the coset leader e_i^d associated with the degenerate set of $\hat{E}_i^B$. In summary, we may restructure the data table D_1 obtained with algorithm 2 (before the final degenerate set selection)

$$D_1 = \{s_1 : \{(e_{11}^d, p_{11}), (e_{12}^d, p_{12}), \dots\}, s_2 : \{(e_{21}^d, p_{21}), (e_{22}^d, p_{22}), \dots\}, \dots\} \quad (4.92)$$

to encode the count N_i instead of p_i , and to store a list of error counts for different orders, yielding

$$\tilde{D}_1 = \{s_1 : \{e_{11}^d : \mathbf{n}_{11}, e_{12}^d : \mathbf{n}_{12}, \dots\}, s_2 : \{e_{21}^d : \mathbf{n}_{21}, \dots\}, \dots\} \quad (4.93)$$

with

$$\mathbf{n}_i := (0, N_i, 0, \dots) \quad (4.94)$$

a list of size $(\omega + 1)$, where only the second entry of the list (corresponding to $\omega = 1$) starts with non-zero entries. The first entry is only non-zero for the $E_0 = I$ case, when $\mathbf{s} = \hat{\mathbf{s}} = 0$ and $\mathbf{L} = I$, corresponding to the case where no error occurs. We represent the entry of order g by $\mathbf{n}_i(g)$.

Following Eq. (4.93), instead of iterating through the combinations $\{E_1^B E_2^B, E_1^B E_3^B, \dots, E_2^B E_3^B, \dots\}$ as we could do with the naive implementation of algorithm 2, we iterate through the degenerate sets in $\tilde{D}_1$ as a whole.

Consider the data table $\tilde{D}_{\omega-1}$ that includes the errors up to order $\omega - 1$. To obtain the table for errors up to order ω , we iterate through the combination of the degenerate sets in $\tilde{D}_{\omega-1}$ and $\tilde{D}_1$. If the noise statistics are highly degenerate (which is generally the case following the noise model in Section 4.2), we can have considerable computational savings, since we only need to perform $|\tilde{D}_{\omega-1}| |\tilde{D}_1|$ computations instead of $15^\omega \binom{N_{\text{CNOT}} + 2(n-k)}{\omega}$ (see Section G). While we expect the latter to grow quickly with $\mathcal{O}(n^{2\omega})$, the former approach should grow, at worst, with $\mathcal{O}(n^\omega)$, and it may grow more slowly in practice.

With this approach we generally overcount the number of errors N_i associated with each degenerate set. There are three types of overcounting:

- Counting permuted copies: Consider an order- $(\omega - 1)$ error $E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B$ (with $i_1 < i_2 < i_3 < \dots$), coming from $\tilde{D}_{\omega-1}$, and the error $\hat{E}_j^B$, coming from $\tilde{D}_1$. W.l.o.g., suppose $j < i_1$. Then, for

$\tilde{D}_\omega$, we will not only count the error $E_j^B E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B$, but also that same error coming from the combination of the errors $E_j^B E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B \setminus E_{i_k}^B$ and $E_{i_k}^B$. In total, we overcount each order- ω error ω times.

- Recounting lower order errors: for the error $E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B$, composing with any $E_{i_k}^B$ ($1 \leq k \leq \omega - 1$) reduces the error to one of order $\omega - 2$, which was previously counted. Each order- $(\omega - 2)$ error we counted before will be recounted $\zeta_{\omega-1,1}$ times, where ζ is given in Eq. (4.98).
- Counting two errors occurring in the same CNOT gate: an error of order ω stems from base errors that occurred on ω CNOT gates. When composing this error with another, the resulting compound error may have more than one base error occurring at one or more CNOT gates. As this compound error is impossible, it should be discounted.

Algorithm 6: Data table $\tilde{D}_{a+b}$

Input: $\tilde{D}_a, \tilde{D}_b$

Output: A data table $\tilde{D}_{a+b}$

```

1 Procedure DataMerging( $\tilde{D}_a, \tilde{D}_b$ ):
2   Initialize empty data table  $\tilde{D}_{a+b}$ 
3   foreach  $s_i, e_i^d, \mathbf{n}_i$  in  $\tilde{D}_a$  do
4     foreach  $s_j, e_j^d, \mathbf{n}_j$  in  $\tilde{D}_b$  do
5        $\tilde{\mathbf{n}}_{ij} \leftarrow \mathbf{n}_i * \mathbf{n}_j$  // convolution
6        $\mathbf{n}_{ij} \leftarrow \tilde{\mathbf{n}}_{ij}$  with overcounting correction
7       if  $e_i^d e_j^d$  not in  $\tilde{D}_{a+b}[s_i \oplus s_j]$  then
8         Store  $\{e_i^d e_j^d : \tilde{\mathbf{n}}_{ij}\}$  in  $\tilde{D}_{a+b}[s_i \oplus s_j]$ 
9       else
10        Add  $\tilde{\mathbf{n}}_{ij}$  to vector in  $e_i^d e_j^d$  entry in  $\tilde{D}_{a+b}[s_i \oplus s_j]$ 
11   foreach  $s_r, e_r^d, \mathbf{n}_r$  in  $\tilde{D}_{a+b}$  do
12      $\mathbf{n}_r \leftarrow \tilde{\mathbf{n}}_r$  with overcounting correction
13     Store  $\{e_r^d : \mathbf{n}_r\}$  in  $\tilde{D}_{a+b}[s_r]$ 

```

We can extend this approach further. Instead of constructing the data table in one order increments, if we already have $\tilde{D}_\omega$, we may combine it with itself to obtain $\tilde{D}_{2\omega}$, thereby requiring exponentially fewer iterations, as ω increases, as long as ω is such that the codes capabilities are not yet saturated, i.e., not all syndromes are assigned to a degenerate set.

In general, if we compute the noise statistics of errors of order a and b to compute those of order $\omega = a + b$, we have

$$\mathbf{n}_i(\omega) = \frac{1}{R_{a,b}} \left[\tilde{\mathbf{n}}_i(\omega) - \sum_{\substack{0 < k+r \leq b \\ k, r \geq 0}} \zeta_{a,b}(k, r) \mathbf{n}_i(c) \right], \quad (4.95)$$

$$\text{with } c := a + b - 2k - r, \quad (4.96)$$

where $\tilde{\mathbf{n}}_i(\omega)$ is the count obtained for order ω before the overcounting correction. The auxiliary functions are given by

$$R_{a,b} := \binom{a+b}{a} \quad (4.97)$$

$$\zeta_{a,b}(k,r) := K^k (K-1)^r \xi_{a,b}(k,r) \quad (4.98)$$

$$\xi_{a,b}(k,r) := \binom{\tilde{N}_{\text{CNOT}} - c}{k} \binom{c}{r, a-k-r, b-k-r}, \quad (4.99)$$

where K is the number of distinct errors per CNOT gate (in our case, always 15). Note the use of binomial and multinomial coefficients. To incorporate the effect of preparation and measurement errors, we use $\tilde{N}_{\text{CNOT}} = N_{\text{CNOT}} + 2(n-k)$ and not N_{CNOT} , as explained in Section G. See Section I for a derivation of these expressions.

The probability associated with the degenerate set with list $\mathbf{n}_i$ is given by

$$p_i = \sum_{j=0}^{\omega} \mathbf{n}_i(j) P(j). \quad (4.100)$$

Given this procedure to obtain $\tilde{D}_\omega$, we may obtain the decoding table T_ω by following algorithms 5 to 7.

Algorithm 7: Decoding table T_{a+b}

Input: $\tilde{D}_\omega$

Output: A decoding table T_ω

```

1 Procedure DecodingTable( $\tilde{D}_\omega$ ):
2   Initialize empty decoding table  $T_\omega$ 
3   foreach  $s$  in  $\tilde{D}_\omega$  do
4      $(j, p) \leftarrow (-1, 0)$ 
5     foreach  $e_i^d, \mathbf{n}_i$  in  $\tilde{D}_\omega[s]$  do
6       Using  $\mathbf{n}_i$ , compute  $p_i$  (Eq. (4.100))
7        $(j, p) \leftarrow (i, p_i)$  if  $p_i > p$ 
8   Set  $T[s]$  as the pattern  $e_j^d$ , which has highest  $p$  in  $D[s]$ 

```

I Derivation of decoding formulas for Bernoulli noise

As stated in Section H, a straightforward implementation of the procedure described will overcount the number of errors associated to any given syndrome sequence. There are three types of overcounting, which we may analyze separately.

Recounting lower order errors

Suppose we have already computed the correct error count $\mathbf{n}_i(j)$ for $0 \leq j \leq \omega - 1$ (for all degenerate sets), and we are currently trying to determine $\mathbf{n}_i(\omega)$.

For the error $E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B$, composing with any $E_{i_k}^B$ ($1 \leq k \leq \omega - 1$) reduces the error to one of order $\omega - 2$, which was previously counted. To determine how many errors stem from this dynamic, we may note that any fake compound error of order ω has a corresponding error of order $\omega - 2$, which has already been counted in $\mathbf{n}_i(\omega - 2)$. Similarly, any error E_i counted in $\mathbf{n}_i(\omega - 2)$ has a corresponding set of fake compound errors that appear in $\mathbf{n}_i(\omega)$. As E_i stems from $\omega - 2$ base errors, each affecting a different CNOT gate, these fake compound errors must correspond to an error of the form $E_i E_j^B E_j^B$, where E_j^B is a base error from a CNOT gate not present in E_i . For each CNOT gate there are $K = 15$ associated base errors, so there are a total of

$$\zeta_{\omega-1,1}(1,0) = K(\tilde{N}_{\text{CNOT}} - (\omega - 2)) \quad (4.101)$$

fake error compounds that associated with the error E_i . Alternatively, defining $\omega = a + k$, with $k = 1$, we may also write the total as

$$\zeta_{a,1}(1,0) = K(\tilde{N}_{\text{CNOT}} - (a - k)). \quad (4.102)$$

The same principle applies to higher order combinations. If a order- a error E_i is composed with a order- k error E_j ($a \geq k$), and the base errors composing E_j all stem from CNOT gates whose same base errors already compose E_i , then the resulting compound error will have order $a - k$, instead of $a + k$. The total number of errors is now given by

$$\zeta_{a,k}(k,0) = \frac{1}{k!} \prod_{j=0}^{k-1} K(\tilde{N}_{\text{CNOT}} - (a - k) - j) \quad (4.103)$$

$$= K^k \binom{\tilde{N}_{\text{CNOT}} - (a - k)}{k}. \quad (4.104)$$

Note, however, that, when composing a order- a error E_i with a order- b error E_j ($a \geq b \geq k$), it may be the case that only some, and not all, of the base errors composing E_j appear in E_i . In general, there will only be k base errors in common, for all $1 \leq k \leq b$.

In this case, to cover all possible fake errors, we must choose not k CNOT gates out of the $\tilde{N}_{\text{CNOT}} - (a - k)$ gates not related to the order- $(a - k)$ error (as in Eq. (4.104)), but instead choose k CNOT gates out of the gates not related to both the order- $(a - k)$ error, but also the $(b - k)$ base errors in E_j that are valid. Therefore, there are a total of $\tilde{N}_{\text{CNOT}} - (a - k) - (b - k)$ CNOT gates from which we must consider k invalid base errors.

Moreover, associated with the order- $(a - k)$ error from E_i , there are several possible valid $(b - k)$ base

errors stemming from E_j . The total number of fake errors is then given by

$$\zeta_{a,b}(k, 0) = K^k \binom{\tilde{N}_{\text{CNOT}} - (a - k) - (b - k)}{k} \quad (4.105)$$

$$\times \binom{(a - k) + (b - k)}{b - k} \quad (4.106)$$

$$= K^k \binom{\tilde{N}_{\text{CNOT}} - c}{k} \binom{c}{b - k}, \quad (4.107)$$

$$\text{with } c := a + b - 2k. \quad (4.108)$$

Note that, for $b = k$, we have $c = a - k$, so that Eq. (4.107) trivially reduces to Eq. (4.104). The resulting lower order error will have order c . We would need to discount its affect on $\mathbf{n}_i(\omega)$ to obtain the correct count. Unfortunately, it would be difficult to determine the original syndromes of the errors that combined to result in the impossible error, as they may have different origins. As an approximation, we use $\mathbf{n}_i(c)$ to estimate the error count. The resulting correction is achieved by subtracting $\mathbf{n}_i(c)$ times ζ from $\mathbf{n}_i(\omega)$.

Counting impossible errors

If a order- a error E_i is composed with a order- r error E_j ($a \geq r$), and the base errors composing E_j all stem from CNOT gates already associated with the base errors that compose E_i , then the resulting compound error will be impossible, since it will contain at least two different base errors associated with the same CNOT gate (one from E_i and one from E_j).

For $r = 1$, each error in $\mathbf{n}_i(a)$ will have

$$\zeta_{a,1}(0, 1) = (K - 1)a \quad (4.109)$$

associated impossible order- $(a + 1)$ errors, since E_i is composed of a base errors, and for each base error, there are $(K - 1)$ different base errors associated to the same CNOT gate.

For a general r , we have, for each order- a error

$$\zeta_{a,r}(0, r) = (K - 1)^r \binom{a}{r} \quad (4.110)$$

associated impossible errors.

When composing a order- a error E_i with a order- b error E_j ($a \geq b$), it may be the case that only $r < b$ base errors composing E_j are impossible, with the remaining $(b - r)$ base errors stemming from CNOT gates not related to E_i .

To estimate the number of impossible errors, we may look at $\mathbf{n}_i(a + b - r)$. As before, we must choose r

CNOT gates out of the a gates related to E_i to count the number of impossible errors. But, as seen in the previous subsection, to must also count the possible $(b - r)$ base errors in E_j that are valid. These factors result in

$$\zeta_{a,b}(0, r) = (K - 1)^r \binom{a}{r} \binom{a + b - r}{b - r} \quad (4.111)$$

$$= (K - 1)^r \binom{c}{r, a - r, b - r}, \quad (4.112)$$

$$\text{with } c := a + b - r \quad (4.113)$$

$$\text{and } \binom{x + y + z}{x, y, z} := \frac{(x + y + z)!}{x!y!z!} \quad (4.114)$$

the multinomial coefficient. Again, note that Eq. (4.112) trivially reduces to Eq. (4.110) when $b = r$. The resulting lower order error will have order c , so, for this case, we would also need to discount its affect on $\mathbf{n}_i(\omega)$ by subtracting $\mathbf{n}_i(c)$ times ζ .

Counting permuted copies

Consider an order- $(\omega - 1)$ error $E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B$ (with $i_1 < i_2 < i_3 < \dots$), coming from $\tilde{D}_{\omega-1}$, and the error $\hat{E}_j^B$, coming from $\tilde{D}_1$. W.l.o.g., suppose $j < i_1$. Then, for $\tilde{D}_\omega$, we will not only count the error $E_j^B E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B$, but also that same error coming from the combination of the errors $E_j^B E_{i_1}^B E_{i_2}^B \dots E_{i_{\omega-1}}^B \setminus E_{i_k}^B$ and $E_{i_k}^B$. In total, we overcount each order- ω error ω times.

In general, for every order- ω error, any possible combination of order- a errors and order- b errors that can generate it (with $\omega = a + b$) will appear in the counting. Since there are

$$R_{a,b} := \binom{\omega}{a} = \binom{\omega}{b} \quad (4.115)$$

ways for order- a and order- b errors to generate an order- ω error, the final counting (after discounting the previous overcounting cases) should be reduced by a factor of $R_{a,b}$.

Full expression

In general, the erroneous errors that the decoding procedure may containing not only repeated base errors ($k > 0$), but also base errors stemming from the same CNOT gate ($r > 0$). Therefore, these two factors need to be considered together.

Combining the analyses of the previous subsections, we conclude that, when composing a order- a error E_i with a order- b error E_j ($a \geq b$), we can have k base errors E_j already appearing in E_i , and r base errors in E_j sharing the same origin CNOT gate as a base error in E_i , with $k + r \leq b$.

To count all these errors, we may look that the errors with order $c = (a + b - 2k - r)$, from which we can generate all the invalid order- ω errors. As before, the k repeated base errors are chosen from those

associated with CNOT gates that are not related to a valid base error in the compound error. There are $\tilde{N}_{\text{CNOT}} - c$ such gates, and each one has K associated base errors.

Moreover, the from the c base errors, we may consider that $a - k - r$ (resp. $b - k - r$) correspond to the base errors in E_i (resp. E_j) that raise no issue, with the remaining r errors corresponding to base errors stemming from CNOT gates that also originated invalid base errors in E_j .

Grouping all three overcounting issues, we end up with

$$\zeta_{a,b}(k, r) = K^k (K - 1)^r \xi_{a,b}(k, r) \quad (4.116)$$

with

$$\xi_{a,b}(k, r) := \binom{\tilde{N}_{\text{CNOT}} - c}{k} \binom{c}{r, a - k - r, b - k - r}, \quad (4.117)$$

$$\text{and } c := a + b - 2k - r, \quad (4.118)$$

which generalizes Eqs. (4.107) and (4.112).

The corrected count is consequently given by

$$\mathbf{n}_i(\omega) = \frac{1}{R_{a,b}} \left[\tilde{\mathbf{n}}_i(\omega) - \sum_{\substack{0 < k+r \leq b \\ k, r \geq 0}} \zeta_{a,b}(k, r) \mathbf{n}_i(c) \right], \quad (4.119)$$

$$\text{with } c := a + b - 2k - r, \quad (4.120)$$

as indicated in Eq. (4.95), with $\tilde{\mathbf{n}}_i(\omega)$ being the original count from the optimized decoding process. As previously indicated, since the estimate of the impossible errors is not exact, this formula is approximate.

J Alternative definitions of $\mathcal{F}_E$ and $\mathcal{F}_L$

Instead of using the formulation described in Sections A and B, we may consider an alternative formulation that, while less computationally efficient, is conceptually more straightforward. Under this formalism, the components $\mathbf{E}_i$ and $\mathbf{L}_i$ can be computed at once from e_i , so there is less of a need to separate the two processes.

For a given error E_i , we compute e_i . Taking the encoding U , we compute

$$e_i^u := U^\dagger e_i U. \quad (4.121)$$

The *unencoded error pattern* e_i^u corresponds to the effect of e_i on the quantum state if it were decoded.

We may decompose it into

$$e_i^u := \mathbf{E}_i^u \mathbf{S}_i^u \mathbf{L}_i^u, \quad (4.122)$$

with

$$\mathbf{E}_i^u = U^\dagger \mathbf{E}_i U, \quad (4.123)$$

$$\mathbf{S}_i^u = U^\dagger \mathbf{S}_i U, \quad (4.124)$$

$$\mathbf{L}_i^u = U^\dagger \mathbf{L}_i U. \quad (4.125)$$

We may also decompose it into the Pauli string for the first k data qubits (e_i^D) and the additional $(n-k)$ redundancy qubits (e_i^R),

$$e_i^u =: e_i^D \otimes e_i^R. \quad (4.126)$$

From Eqs. (4.3) to (4.5), we have that

$$Z_{i+k} = U^\dagger S_i U \quad (4.127)$$

$$X_j = U^\dagger \bar{X}_j U \quad (4.128)$$

$$Z_j = U^\dagger \bar{Z}_j U. \quad (4.129)$$

Therefore, decoding e_i into e_i^u cleanly separates the different components. $\mathbf{L}_i^u$ corresponds to the components of e_i^u in the first k qubits. We have

$$\mathbf{L}_i^u := e_i^D \otimes I_{n-k} \quad (4.130)$$

$$= \left(\prod_{j=1}^k X_j^{b_j^X} \right) \left(\prod_{j=1}^k Z_j^{b_j^Z} \right) \quad (4.131)$$

$$\implies \mathbf{L}_i = U(e_i^D \otimes I_{n-k})U^\dagger \quad (4.132)$$

$$= \left(\prod_{j=1}^k \bar{X}_j^{b_j^X} \right) \left(\prod_{j=1}^k \bar{Z}_j^{b_j^Z} \right), \quad (4.133)$$

where b_j^X and b_j^Z are the X and Z components of e_i^D , respectively.

Let $e_i^{R,X}$ and $e_i^{R,Z}$ be the X and Z components of e_i^R , respectively (where the Y components have been decomposed into X and Z , as in Eq. (4.10)), so that $e_i^R = e_i^{R,X} e_i^{R,Z}$ (disregarding the phase factor). We have

$$\mathbf{E}_i^u := I_k \otimes e_i^{R,X} \quad (4.134)$$

$$= \prod_{i=k+1}^n X_i^{b_{i-k}^X} \quad (4.135)$$

$$\implies \mathbf{E}_i = U(I_k \otimes e_i^{R,X})U^\dagger \quad (4.136)$$

$$= \prod_{i=k+1}^n (U X_i U^\dagger)^{b_{i-k}^X}, \quad (4.137)$$

and

$$\mathbf{S}_i^u := I_k \otimes e_i^{R,Z} \quad (4.138)$$

$$= \prod_{i=k+1}^n Z_i^{b_{i-k}^Z} \quad (4.139)$$

$$\Rightarrow \mathbf{S}_i = U(I_k \otimes e_i^{R,Z})U^\dagger \quad (4.140)$$

$$= \prod_{i=k+1}^n (UZ_iU^\dagger)^{b_{i-k}^Z}. \quad (4.141)$$

where b_{i-k}^X and b_{i-k}^Z are the X and Z components of $\mathbf{e}_i^{R,X}$ and $\mathbf{e}_i^{R,Z}$, respectively. As this procedure is deterministic, we obtain unique components $\mathbf{E}_i$, $\mathbf{S}_i$, and $\mathbf{L}_i$ associated with the error pattern e_i .

Regarding runtime complexity, the method presented in Section A requires just $\mathcal{O}(n-k)$ steps per error E_i , in order to assemble $\mathbf{E}_i$ from the precomputed Z_{h_i} and X_{h_i-n} terms. The method presented in Section B is more involved. Computing e'_i requires $2n$ steps. Determining its stabilizer component $\mathbf{S}_i$ requires identifying the pivots in $\mathbf{e}'_i$ ($\mathcal{O}(n)$ steps) and then multiplying the constituting stabilizers by $\mathbf{e}'_i$. As it is constituted by $\mathcal{O}(n-k)$ stabilizers, and accounting for each takes at most $(2n-(n-k-1)) = n+k+1$ steps, the whole stabilizer part takes $\mathcal{O}((n-k)(n+k+1))$ steps. For the logical component, there are $\mathcal{O}(2k)$ components in $\mathbf{e}'_i$, and the whole each operator takes $(2n-(n-k)-2k+1) = n-k+1$ steps, for a total of $\mathcal{O}((2k)(n+k-1))$ steps. The full procedure requires

$$\mathcal{O}(n + (n-k)(n+k+1) + (n-k+1)(2k)) \quad (4.142)$$

$$\sim \mathcal{O}(n^2, k^2) \sim \mathcal{O}(n^2) \quad (4.143)$$

steps per error E_i to compute $\mathbf{L}_i$. The computation for N_E errors requires

$$\mathcal{O}(N_E n^2) \quad (4.144)$$

steps.

For this simpler technique, the greatest computational expense comes from computing e_i^u for each error E_i , as the remaining steps can be precomputed and subsequently applied to all errors.

To facilitate the calculation, we may precompute the e^u patterns associated with all base errors, and use those to compute the pattern e_i^u for each error E_i .

From [159], simulating a stabilizer circuit (i.e. U) with N gates takes $\mathcal{O}(n^2 N)$ steps. Since U is built from $\mathcal{O}(n \log^2 n)$ Clifford gates, we have that the full precomputation associated with the base errors scales as

$$\mathcal{O}(N_{\text{CNOT}} n^3 \log^2 n). \quad (4.145)$$

The cost of computing e_i^u for each E_i then scales as

$$\mathcal{O}(n\omega) \quad (4.146)$$

where ω is the order of the error. The full computation for N_E errors requires

$$\mathcal{O}(N_E n \omega + N_{\text{CNOT}} n^3 \log^2 n) \quad (4.147)$$

steps. For cases where $N_E \gg N_{\text{CNOT}}$, the simpler approach may lead to a faster implementation, while, for smaller cases, the main approach is faster, as it doesn't require precomputation.

K Analysis of numerical results

To get a good understanding of the performance of the decoding method, we consider an equidistant range for h , and we sample p using the expression

$$p = 10^h. \quad (4.148)$$

For that reason, most of the fits in this section are performed after applying a logarithmic transformation to both the dependent and independent variables. That is, we prefer to work with $\log(p)$ than p directly, as it is more numerically stable.

In this section, we verify that $C(p)$ and P_{failure} scale as

$$C(p) \simeq e^{-\gamma N_{\text{CNOT}} p} \quad (4.149)$$

$$\simeq 1 - \gamma N_{\text{CNOT}} p \quad (4.150)$$

$$P_{\text{failure}}(p) \simeq 1 - e^{-\mu p^\eta}, \text{ for } \eta \in \mathbb{N} \quad (4.151)$$

$$\simeq \mu p^\eta, \quad (4.152)$$

for $p \ll 1$. μ and γ are positive real parameters, and η is the lowest order that the code cannot fully correct. We analyze these expressions separately in the next subsections.

Escaped errors

We start by considering $P_u(p, 1)$. For the noise model in Section 4.2, we expect that the probability of finding escaped errors will be given by

$$P_u(p, 1) \simeq \sum_{i=1}^{N_{\text{CNOT}}} \binom{N_{\text{CNOT}}}{i} (\gamma p)^i (1 - \gamma p)^{N_{\text{CNOT}} - i} \quad (4.153)$$

$$= \sum_{i=1}^{N_{\text{CNOT}}} (-1)^{i+1} \binom{N_{\text{CNOT}}}{i} (\gamma p)^i \quad (4.154)$$

$$= 1 - \sum_{i=0}^{N_{\text{CNOT}}} \binom{N_{\text{CNOT}}}{i} (-\gamma p)^i, \quad (4.155)$$

where γ reflects the fraction of errors that can escape. For $p \ll 1$, the lower order terms dominate, so we may use the approximation

$$\binom{N_{\text{CNOT}}}{i} \simeq \frac{N_{\text{CNOT}}^i}{i!}, \text{ for } N_{\text{CNOT}} \gg i, \quad (4.156)$$

and we have

$$P_u(p, 1) \simeq 1 - \sum_{i=0}^{N_{\text{CNOT}}} \frac{(-N_{\text{CNOT}}\gamma p)^i}{i!} \quad (4.157)$$

$$\simeq 1 - \sum_{i=0}^{\infty} \frac{(-N_{\text{CNOT}}\gamma p)^i}{i!} \quad (4.158)$$

$$= 1 - e^{-\gamma N_{\text{CNOT}} p}. \quad (4.159)$$

Given Eqs. (4.37) and (4.44), we end up with Eq. (4.149),

$$C(p) \simeq e^{-\gamma N_{\text{CNOT}} p}, \quad (4.160)$$

where γ is an unknown parameter. Since we are considering $p \ll 1$, this expression may be further simplified into

$$C(p) \simeq 1 - \gamma N_{\text{CNOT}} p. \quad (4.161)$$

In order to perform a fit, we consider a more general version of this expression, given by

$$C(p) \simeq \exp(-\gamma N_{\text{CNOT}} p^{\eta_C}), \quad (4.162)$$

and we fit the function

$$\log_{10}(-\log(C(p))) = \eta_C \log_{10}(p) + \log_{10}(\gamma N_{\text{CNOT}}) \quad (4.163)$$

$$\implies \log_{10}(-b) = \eta_C \log_{10}(p) + \beta, \quad (4.164)$$

where $\beta = \log_{10}(\gamma N_{\text{CNOT}})$ and b stems from Eq. (4.47). We use $\log_{10}$ to keep the figures more legible.

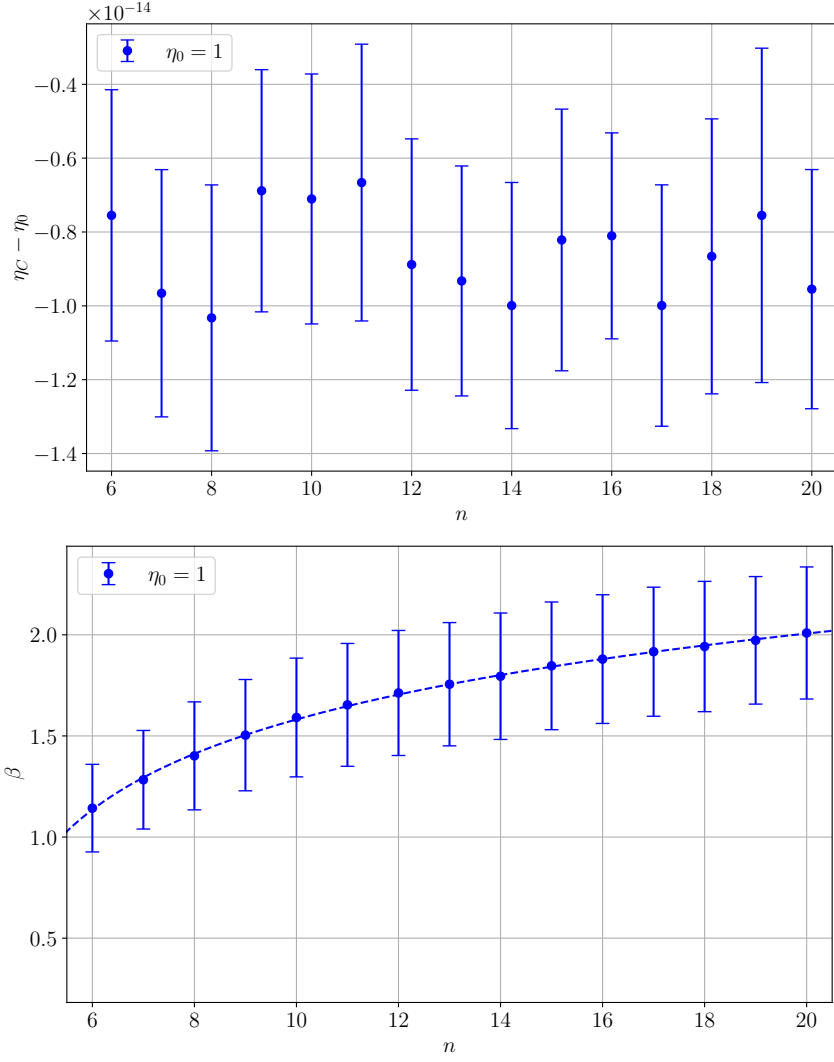


Figure 4.9: Fit of the various parameters, averaged over 50 random codes for each n .

We may now study the behavior of η_C and β for different n . See Fig. 4.9 for the results. As expected, we observe that $\eta_C \simeq 1$, regardless of n .

From Section C, we have that $N_{\text{CNOT}} \sim \mathcal{O}(n^2)$. If we assume that $\gamma \sim \mathcal{O}(\text{poly}(n))$, then β should scale as

$$\beta = b_1 \log(n - n_\beta) + b_2. \quad (4.165)$$

with $\eta_0 = 1$. We assume this expression for its fit. The fitted parameters are given in Table 4.4.

Variable	Value
b_1	0.4494
n_β	3.6453
b_2	0.7505

Table 4.4: Fitted parameters for β .

We may alternatively write β as

$$\beta \simeq \log_{10}(5.630 \cdot (n - 3.6453)^{1.035}) \quad (4.166)$$

Given the dependence between β and N_{CNOT} , we may conclude that

$$\gamma N_{\text{CNOT}} \sim \mathcal{O}(n^{1.035}) \simeq \mathcal{O}(n), \quad (4.167)$$

despite the actual N_{CNOT} count scaling with $\mathcal{O}(n^2)$. These results are in line with the expectations from the theoretical analysis of Section C, indicating that we may observe a visible $p_{\text{threshold}}$.

Direct P_{failure} extrapolation

We may apply a similar procedure to P_{failure} . We empirically observe that Eq. (4.151) holds. However, unlike the previous section, it is no longer the case that $\eta_C \simeq 1$.

If a code is able to correct all errors of order $\omega < \eta$, then we expect P_{failure} to be given by

$$P_{\text{failure}} = \sum_{i=1}^{N_{\text{CNOT}}} f_i(p, N_{\text{CNOT}}) \binom{N_{\text{CNOT}}}{i} p^i (1-p)^{N_{\text{CNOT}}-i} \quad (4.168)$$

$$\simeq \sum_{i=\eta}^{N_{\text{CNOT}}} \tilde{f}_i \binom{N_{\text{CNOT}}}{i} p^i (1-p)^{N_{\text{CNOT}}-i}, \text{ for } p \ll 1 \quad (4.169)$$

$$\simeq \tilde{f}_\eta \binom{N_{\text{CNOT}}}{\eta} p^\eta (1-p)^{N_{\text{CNOT}}-\eta}, \text{ for } p \ll 1, \quad (4.170)$$

$$= \mu p^\eta, \text{ for } p \ll 1. \quad (4.171)$$

for some unknown μ , where $\tilde{f}_i$ is an approximation of $f_i(p, N_{\text{CNOT}})$, which is unknown function. In the approximation in Eq. (4.169), we consider $\tilde{f}_i$ as a scalar.

In order to perform a fit, we consider an equivalent version of this expression for $p \ll 1$, given by

$$P_{\text{failure}}(p) \simeq 1 - \exp(-\mu p^\eta), \quad (4.172)$$

and we fit the function

$$\log_{10}(-\log(1 - P_{\text{failure}})) = \eta \log_{10}(p) + \log_{10}(\mu) \quad (4.173)$$

$$= \eta \log_{10}(p) + \alpha, \quad (4.174)$$

where $\alpha = \log_{10}(\mu)$. As before, we use $\log_{10}$ to keep the figures more legible. See Fig. 4.10 for an example.

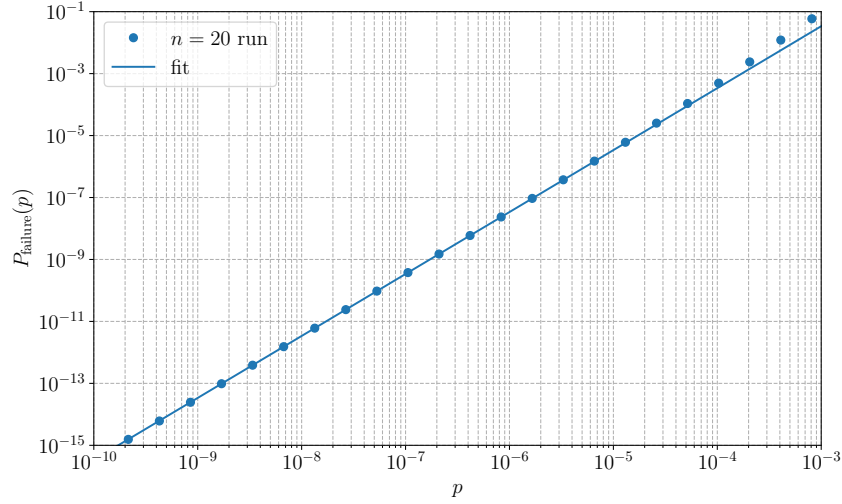


Figure 4.10: Fit of Eq. (4.164), for a random example with $n = 20$. We fit only using values below 10^{-6} , where it is safe to assume $p \ll 1$. As expected, the fit worsens for higher p .

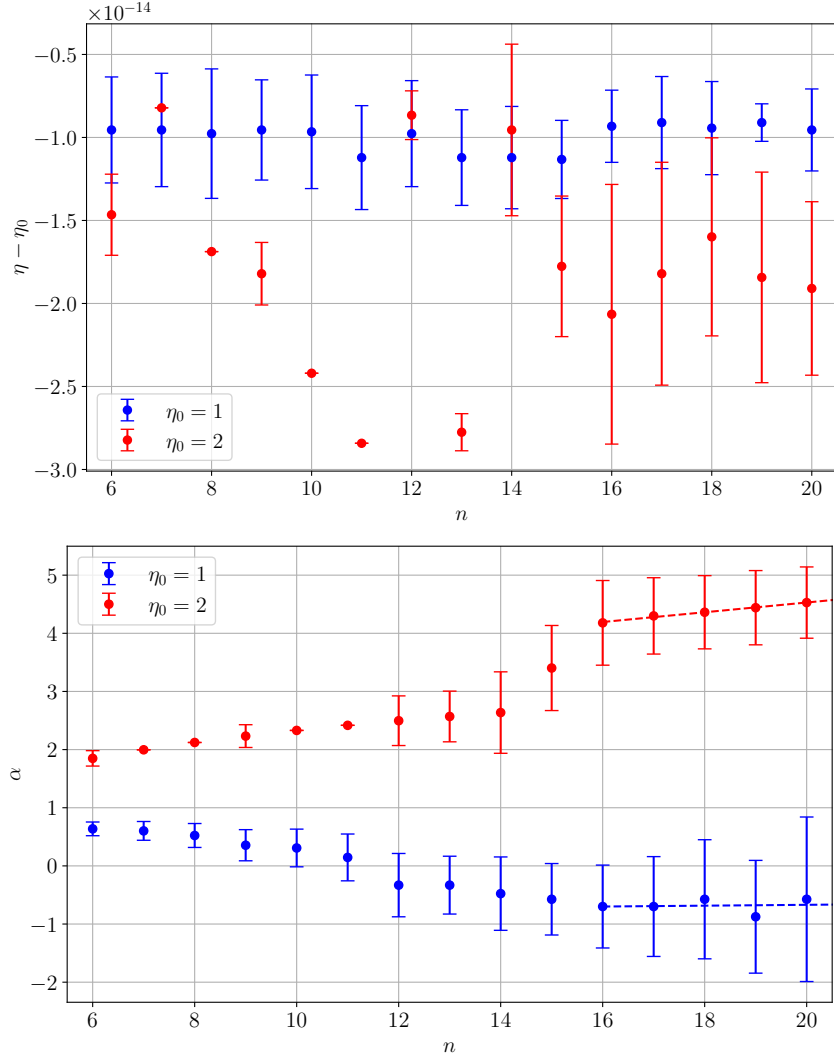


Figure 4.11: Plot of η and α . We fit a linear function to α for the higher n values.

We may now study the behavior of η and α for different n . See Fig. 4.11 for the results. In this case, we have different η values, and the values cluster around integers. Since the n values we tested are relatively low, we only observe η equal to either 1 or 2. As these cases display notably different behavior, we separate their data before fitting.

As before, we fit η to an expression of the form

$$\eta = \eta_0 + d_1 \log(1 + \exp(d_2 \cdot (n - n_P))). \quad (4.175)$$

We empirically observe that the α variable is better fitted by a simple linear expression

$$\alpha = a_1 n + a_2. \quad (4.176)$$

The fitted parameters are given in Table 4.5.

Variable	$\eta_0 = 1$	$\eta_0 = 2$
a_1	-0.0074	0.0838
a_2	-0.8170	2.8530

Table 4.5: Fitted parameters for α .

We may also have a look at the probability of having different η values. The empirical results can be seen in Fig. 4.12 and Table 4.6. We observe that, for low n values, the fraction of random codes with $\eta = 1$ and $\eta = 2$ is roughly constant, since the code's capabilities are not large enough to generally correct all $\omega = 1$ errors. As n increases, the probability that the code corrects all order-1 errors also increases, and we so the fraction of $\eta = 2$ cases increases. Its behavior follows a sigmoid-like function, of the form

$$s_1 \left(\frac{1}{1 + e^{-s_2(n-n_f)}} - \frac{1}{2} \right) + s_3. \quad (4.177)$$

The errorbars indicate the standard deviation, which stems from the finite number (48) of samples taken for each n value. Also, note the symmetry in the parameters in Table 4.6, reflecting the fact that the fractions must add up to one.

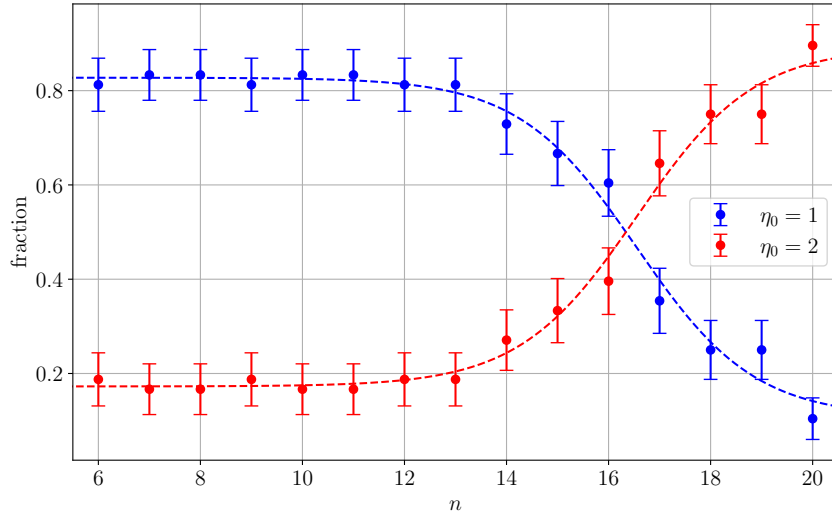


Figure 4.12: Fraction of the simulations with different η_0 values.

Variable	$\eta_0 = 1$	$\eta_0 = 2$
s_1	-0.7206	0.7206
s_2	0.8665	0.8665
n_f	16.5506	16.5506
s_3	0.4673	0.5327

Table 4.6: Fitted parameters.

For very low $n \leq 12$, we observe that about 20% of the codes have $\eta = 2$. On further inspection, their seemingly high performance does not stem from strong correction capabilities, but from the fact that, due to random chance, the stabilizers of these codes have relatively low weight, leading to a very low number of CNOT gates in the syndrome extraction circuit, and consequently fewer errors needed to consider. In a more complex setting where there are additional sources of error, we would expect these codes in particular to perform poorly.

Extrapolating behavior for larger n

We may take the results above and use them to extrapolate the performance for larger n values. We also incorporate the uncertainty observed in the numerical data by using the standard deviation observed for the sampled n to estimate the deviation for larger n .

L Single-qubit gate error model considerations

Standard circuit-level noise models typically include single-qubit gate errors, particularly identity gate errors from qubit idling [184]. The exclusion of these errors in the present work represents a limitation of the current theoretical analysis. Including comprehensive single-qubit gate errors would likely reduce our reported threshold values, and we consider it as important future work.

We deliberately use a simple, unoptimized stabilizer implementation to avoid confounding our theoretical results with circuit-specific optimizations, as indicated in Section A. While gate count and idling could be reduced through circuit optimization techniques, we chose not to incorporate such optimizations to maintain the generality of our theoretical analysis.

Our model would be more realistic for trapped-ion systems where idling errors (due to relaxation or decoherence) are significantly lower ($\sim 10^{-6}$) compared to superconducting qubits ($\sim 10^{-4}$). In trapped-ion platforms, coherence times are typically much longer, making idling errors less significant compared to active gate operations. This platform-dependent behavior suggests that our simplified model captures the dominant error sources for certain quantum computing architectures.

Nonetheless, in the rest of this section we show that many of the single-qubit gate errors can be absorbed into our model through mathematical equivalences.

Hadamard gate errors

The circuit contains, for each ancilla qubit, a Hadamard gate after preparation and before measurement. If we model this gate as suffering an error corresponding to an additional Pauli gate, then it can be shown that the error can be absorbed into preparation and measurement errors, respectively.

For the Hadamard gate after preparation, a Z -type error is equivalent to an X error before the Hadamard gate (since $HX = ZH$), which is equivalent to a flip preparation bit. A X -type error similarly corresponds to a Z -type error before the Hadamard gate, i.e., right after preparation. Since the prepared qubit is initially either in the state $|0\rangle$ or $|1\rangle$, a Z -type error does not introduce additional errors, since it acts as

the identity.

For the Hadamard gate before measurement, a Z -type error is similarly irrelevant, since the measurement is done in the computational basis. A X -type error is equivalent to a measurement error, where the measurement bit flips.

For both cases, a Y -type error is a combination of both error types. Therefore, if the probability of a preparation/measurement error is p_M , and the probability of a specific Hadamard gate error is $p_H/3$, then we can model Hadamard gate errors by simply using the equivalent probability $p'_M = p_M(1 - \frac{2p_H}{3}) + \frac{2p_H}{3}(1 - p_M) \simeq p_M + \frac{2p_H}{3}$. Since, in practice, $p_M \gg p_H$, disregarding the Hadamard gate errors does not significantly affect the analysis.

1-qubit gate errors

For the 1-qubit gates we consider in our model, possibly appearing before and after the target of a CNOT gate, we can similarly map them so that the errors actually arise from the associated CNOT gate, while assuming that the 1-qubit gates are perfect. If their error rate is p_G , then the equivalent error rate for the CNOT gates incorporating the 1-qubit Pauli gate errors is $p' \simeq p + p_G$. Similarly to before, $p \gg p_G$, so their exclusion from our model does not significantly affect the analysis.

Idling errors

For comprehensive circuit-based error models, it is standard to also include errors coming from idle qubits changing over time, either due to relaxation, decoherence, or more complex dynamics (e.g., crosstalk). This scenario can be modeled by assuming that identity gates are applied to the circuit, and that these gates get converted to a Pauli gate when an error occurs.

Unlike the previous scenarios, it is harder to map these errors to equivalent errors in the CNOT gates. We can evolve the idling errors forward in the circuit up to the first CNOT error they encounter, where they can potentially increase in weight. In our formulation, if we assume that each identity gate with error p_I takes as long as a CNOT gate application, then the errors can compound quickly. Since the stabilizer application is not optimized, if we apply the $n - k$ stabilizers in series, and each stabilizer affects $w := \lambda n$ qubits (where $\lambda = 3/4$ in our implementation, and $\simeq 1/2$ in the optimized version in Section A), then for each main qubit there are on average

$$\simeq \lambda n [1 + 2(1 - \lambda) + 3(1 - \lambda)^2 + \dots] \quad (4.178)$$

$$= \frac{n}{\lambda} =: m \quad (4.179)$$

identity gates between CNOT gates.

During every idle slot the qubit is acted on by a single-qubit depolarizing channel $\mathcal{D}_{p_I}(\rho) = (1 - \frac{3p_I}{4})\rho + \frac{p_I}{4} \sum_{\alpha \in \{X, Y, Z\}} \alpha \rho \alpha$.

In the Pauli frame each gate independently applies

$$E_j = \begin{cases} I, & \text{with probability } 1 - q, \\ X, Y \text{ or } Z, & \text{each with probability } q/3, \end{cases} \quad (4.180)$$

$$q := \frac{3p_I}{4}. \quad (4.181)$$

Because the non-trivial Pauli operators modulo a phase form the abelian group $V_4 = \{I, X, Y, Z\}$, the net error after the m idle slots is the group product $E := E_m \cdots E_1$. A standard character argument for random walks on V_4 gives

$$\Pr[E = I] = \frac{1}{4}[1 + 3(1 - q)^m]. \quad (4.182)$$

Hence, the equivalent identity error is

$$p'_I = 1 - \Pr[E = I] \quad (4.183)$$

$$= \frac{3}{4}[1 - (1 - q)^m] \quad (4.184)$$

$$= \frac{3}{4} \left(1 - \left(1 - \frac{3p_I}{4} \right)^{n/\lambda} \right) \quad (4.185)$$

$$\simeq \frac{9}{16} np_I / \lambda, \quad \text{for } np_I / \lambda \ll 1. \quad (4.186)$$

Since we expect p_I to be about one order of magnitude lower than p (except for trapped-ion platforms, where p_I is much lower), even relatively low values of n make idling non-negligible. Nonetheless, we can show that, in a more realistic setting, where the gate implementation is optimized, idling can be largely removed, thereby making this error source negligible.

In an optimized setting, besides implementing the stabilizer optimizations in Section A, we would also parallelize the CNOT gates associated with the different stabilizers as much as possible. In general, it would be possible to parallelize CNOT gates associated with different stabilizers whenever their target qubits are different.

To quantify how much circuit parallelization is possible we model the *scheduling* of the CNOT gates associated with the $t := (n - k) \simeq n$ stabilizers. Write $A = \{a_1, \dots, a_t\}$ for the ancilla (control) qubits and $B = \{b_1, \dots, b_n\}$ for the data (target) qubits. Every CNOT is an *edge* $e = (a_i, b_j)$ in the bipartite graph $G = (A \cup B, E)$. For the code families considered here

- each ancilla participates in $w = \lambda n$ edges ($\lambda = \frac{1}{2}$ or $\frac{3}{4}$);
- for a fixed data qubit b_j the number of incident edges X_j is a random variable $X_j \sim \text{Binomial}(t, \frac{w}{n}) = \text{Binomial}(n, \lambda)$ with mean $\mu = \mathbb{E}[X_j] = \lambda n = w$.

A single time step (“layer”) of the circuit is a *matching* of G : a set of edges with no common end-points, so that no qubit is touched twice in the same layer. Partitioning E into the minimum number of such

matchings is the edge-colouring problem; for every bipartite graph, König's theorem gives

$$\chi'(G) = \Delta(G), \quad (4.187)$$

where $\Delta(G)$ is the maximum vertex degree. Consequently

$$D = \Delta(G) = \max\left\{w, \max_{1 \leq j \leq n} X_j\right\} \quad (4.188)$$

is the *optimal* CNOT depth.

Let $M_n := \max_j X_j$. A standard Chernoff bound gives, for any $a > 0$,

$$\Pr[X_j \geq \mu + a] \leq \exp\left(-\frac{a^2}{2(\mu+a/3)}\right). \quad (4.189)$$

Choosing $a = \sqrt{2\mu \ln n}$ and union-bounding over the n data qubits,

$$\Pr[M_n \geq \mu + \sqrt{2\mu \ln n}] \xrightarrow{n \rightarrow \infty} 0, \quad (4.190)$$

so that, with high probability,

$$M_n \leq w + \sqrt{2w \ln n}.$$

Because $M_n \geq w$ always, we may replace the order symbol in Eq. (4.188) by its leading constant and write

$$\mathbb{E}[D] = w + \sqrt{2w \ln n} + o(\sqrt{w \ln n}) \quad (4.191)$$

$$\simeq w + \sqrt{2w \ln n} \quad (4.192)$$

$$= \lambda n + \sqrt{2\lambda n \ln n}. \quad (4.193)$$

Along any ancilla line exactly w CNOTs are executed in D layers, so the number of idle layers on that qubit is $D - w$. Averaging the gap between successive CNOTs over the $w - 1$ internal intervals gives

$$N_I = \frac{D - w}{w - 1} \quad (4.194)$$

$$\simeq \frac{\sqrt{2w \ln n}}{w} \quad (4.195)$$

$$= \sqrt{\frac{2 \ln n}{w}} = \sqrt{\frac{2}{\lambda} \frac{\ln n}{n}}. \quad (4.196)$$

Hence, keeping only the first-order term,

$$N_I \simeq \sqrt{\frac{2}{\lambda} \frac{\ln n}{n}}. \quad (4.197)$$

This value is lower than 1 and quickly converges to 0 for high n . Therefore, similarly to before, we could

consider an equivalent $p' \simeq p + N_I p_I$ to model idling errors.

Equivalent threshold

Accounting for these errors, and considering that they are at least one order of magnitude lower than CNOT, preparation and measurements, leads to an equivalent error rate that is around 10% higher. Consequently, in this more complex model, with the necessary circuit optimizations, we would get a $p_{\text{threshold}}$ that is 10% lower. We note however that this setup would not be sensible, since our empirical threshold stems from an purposefully unoptimized stabilizer implementation. It is possible that an implementation with optimized stabilizers (which is outside the scope of this work) would lead to a naturally higher threshold, regardless of the 1-qubit gate and idling errors.

Chapter 5

Conclusions

With this thesis, we presented contributions along two research directions in quantum computing. On one hand, we developed a quantum algorithm for the forward scattering transform, the first and one of the most computationally demanding steps of the inverse scattering transform for integrable nonlinear partial differential equations. On the other, we proposed quantum error correction methods suitable for both near-term noisy devices and for future fault-tolerant quantum computers. We see these directions as intrinsically connected. Quantum algorithms for scientific computing will ultimately need a reliable error correction method to run on physical hardware, and the error correction techniques we develop here bring this closer.

5.1 Summary of contributions

In Chapter 2, we presented a quantum algorithm for the forward scattering transform, a spectral analysis technique that computes scattering data from the initial conditions of nonlinear integrable systems. For integrable equations, the scattering data evolves linearly in time, even though the solution itself evolves nonlinearly. As a result, our approach can decompose the nonlinear problem into a sequence of exact linear steps, rather than relying on approximate linearization techniques such as Carleman linearization. Using the Korteweg-de Vries equation as a concrete example, we showed that the forward scattering problem can be reformulated as a stabilized linear matrix ODE and solved with quantum spectral methods. The resulting algorithm achieves a query complexity of $O(r \text{ polylog}(r/\epsilon))$, where r is the spatial discretization parameter and ϵ the desired accuracy. This is a quadratic speedup in r over classical methods such as conjugate gradient descent. The method also applies broadly to entire hierarchies of integrable equations sharing the same spatial Lax operator.

In Chapter 3, we introduced QGRAND, that we believe to be the first practical maximum-likelihood decoder for quantum random linear codes, which achieve optimal error correction performance asymptotically but have historically been impractical because their lack of algebraic structure prevented the development of any reasonably efficient decoder. Rather than relying on code structure, QGRAND ex-

exploits noise statistics. It generates candidate error patterns in order of decreasing probability and checks each against the measured syndrome, while also accounting for error degeneracy, a uniquely quantum phenomenon where physically distinct errors can have equivalent effects on the code space. Consequently, by grouping degenerate errors the effective search space reduces from $O(n^2)$ to $O(n)$, and the decoder remains efficient as code size grows. We demonstrated near-hashing-bound performance at finite block-lengths, which confirms that these theoretically optimal codes can be decoded in practice.

In Chapter 4, we extended QGRAND to the fault-tolerant setting. Syndrome extraction circuits introduce their own errors through faulty preparations, gate operations, and measurements, and the decoder must account for all of these to be useful in practice. We modeled the full error process during syndrome extraction and modified the decoding algorithm to explicitly track error degeneracy. The result is a maximum-likelihood procedure guaranteed to be optimal, and our analysis revealed a threshold error rate of $p_{\text{threshold}} \simeq 2 \times 10^{-5}$ in the asymptotic limit. This threshold is made possible by degeneracy, since without it the number of distinct error patterns would scale as $O(n^2)$ and no visible threshold would emerge. Fortunately, degeneracy reduces this to $O(n)$, and fault-tolerant decoding of QRLCs remains feasible despite their lack of exploitable structure.

5.2 Outlook

For the forward scattering transform, the most immediate direction is to complete the remaining steps of the inverse scattering transform. The time evolution of the scattering data is expected to be straightforward, but the inverse scattering step is more involved, since it requires solving the Marchenko equation, and efficient quantum methods for integral equations do not yet exist. A complete implementation would also require computing the discrete spectrum, since it encodes the bound state and soliton contributions to the solution. Perhaps more promisingly, extending the method to higher-dimensional scattering problems could yield a more substantial quantum advantage, since the scattering geometry becomes exponentially richer in higher dimensions.

Several assumptions in our current analysis could be relaxed. We do not include single-qubit gate or idling errors in our noise model, and incorporating these would likely lower the threshold, though we expect the effect to be modest given their typically lower error rates. More significantly, we assume all-to-all qubit connectivity. Most current hardware does not satisfy this, and adapting QRLCs to constrained connectivity remains an open problem. For higher noise entropy regimes, exact maximum-likelihood decoding becomes computationally prohibitive, and approximate methods such as compressive sensing, deep learning, or greedy heuristics would be needed to keep the decoder practical. Reducing stabilizer weight through canonical form reduction would also help, since it lowers both the gate count and the degree of error propagation during syndrome extraction.

More broadly, this work addresses two open problems in quantum computing. Some nonlinear problems have often been considered infeasible to solve with quantum approaches, due to the latter's linear nature. However, we showed that for integrable systems the nonlinear evolution can be decomposed into sequences

of linear steps amenable to quantum speedups. On the error correction side, we showed that noise-centric decoding, combined with theoretically optimal codes, offers a flexible approach that adapts to changing noise conditions and works under realistic fault-tolerant constraints. As quantum hardware continues to scale, we expect integrability-based decompositions and noise-centric decoding to prove useful beyond the specific problems studied here.

Bibliography

- [1] Diogo Cruz et al. “Quantum Forward Scattering Transform for Nonlinear Differential Equations”. Manuscript in preparation. 2025.
- [2] Diogo Cruz, Francisco A. Monteiro, and Bruno C. Coutinho. “Quantum Error Correction via Noise Guessing Decoding”. In: *IEEE Access* 11 (Oct. 2023), pp. 19446–19461. DOI: 10.1109/ACCESS.2023.3327214.
- [3] Diogo Cruz et al. “Fault-tolerant noise guessing decoding of quantum random codes”. In: *IEEE Transactions on Quantum Engineering* 6 (2025), pp. 1–27. DOI: 10.1109/TQE.2025.3595778.
- [4] Diogo Cruz and Duarte Magano. “Superresolution of Green’s functions on noisy quantum computers”. In: *Phys. Rev. A* 108 (1 July 2023), p. 012618. DOI: 10.1103/PhysRevA.108.012618. URL: <https://link.aps.org/doi/10.1103/PhysRevA.108.012618>.
- [5] Sagar Silva Pratapsi, Diogo Cruz, and Paulo André. “Native Multi-Qubit Gates in Transmon Qubits via Synchronous Driving”. In: *Scientific Reports* 14.1 (Oct. 2024), p. 26042. ISSN: 2045-2322. DOI: 10.1038/s41598-024-76396-9.
- [6] Óscar Amaro and Diogo Cruz. *A Living Review of Quantum Computing for Plasma Physics*. arXiv:2302.00001 [physics, physics:quant-ph]. Jan. 2023. DOI: 10.48550/arXiv.2302.00001. URL: <http://arxiv.org/abs/2302.00001> (visited on 02/02/2023).
- [7] André Roque et al. “Efficient entanglement purification based on noise guessing decoding”. In: *Quantum* 8 (Sept. 2024), p. 1476. ISSN: 2521-327X. DOI: 10.22331/q-2024-09-19-1476.
- [8] A. Yu. Kitaev. “Quantum computations: algorithms and error correction”. In: *Russian Mathematical Surveys* 52.6 (1997), pp. 1191–1249. DOI: 10.1070/RM1997v052n06ABEH002155.
- [9] Peter W. Shor. “Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer”. In: *SIAM Journal on Computing* 26.5 (1997), pp. 1484–1509. DOI: 10.1137/S0097539795293172. URL: <https://doi.org/10.1137/S0097539795293172>.
- [10] Lov K. Grover. “Quantum Mechanics Helps in Searching for a Needle in a Haystack”. In: *Physical Review Letters* 79.2 (1997), p. 325.
- [11] Aram W. Harrow, Avinatan Hassidim, and Seth Lloyd. “Quantum Algorithm for Linear Systems of Equations”. In: *Physical Review Letters* 103.15 (Oct. 2009), p. 150502. DOI: 10.1103/PhysRevLett.103.150502.

- [12] Julia Kempe. “Quantum random walks: an introductory overview”. In: *Contemporary Physics* 44.4 (2003), pp. 307–327. DOI: 10.1080/00107151031000110776.
- [13] M. Cerezo et al. “Variational quantum algorithms”. In: *Nature Reviews Physics* 3.9 (2021), pp. 625–644. DOI: 10.1038/s42254-021-00348-9.
- [14] Jacob Biamonte et al. “Quantum machine learning”. In: *Nature* 549.7671 (2017), pp. 195–202. DOI: 10.1038/nature23474.
- [15] Frank Arute et al. “Quantum supremacy using a programmable superconducting processor”. In: *Nature* 574.7779 (2019), pp. 505–510. DOI: 10.1038/s41586-019-1666-5.
- [16] K. Wright et al. “Benchmarking an 11-qubit quantum computer”. In: *Nature Communications* 10.1 (2019), p. 5464. DOI: 10.1038/s41467-019-13534-2.
- [17] Han-Sen Zhong et al. “Quantum computational advantage using photons”. In: *Science* 370.6523 (2020), pp. 1460–1463. DOI: 10.1126/science.abe8770.
- [18] Sepehr Ebadi et al. “Quantum phases of matter on a 256-atom programmable quantum simulator”. In: *Nature* 595.7866 (July 2021), pp. 227–232. DOI: 10.1038/s41586-021-03582-4.
- [19] Chetan Nayak et al. “Non-Abelian anyons and topological quantum computation”. In: *Reviews of Modern Physics* 80.3 (2008), pp. 1083–1159. DOI: 10.1103/RevModPhys.80.1083.
- [20] John Preskill. “Quantum Computing in the NISQ era and beyond”. In: *Quantum* 2 (2018), p. 79. DOI: 10.22331/q-2018-08-06-79.
- [21] Edward Farhi et al. *Quantum Computation by Adiabatic Evolution*. 2000. arXiv: quant-ph/0001106.
- [22] Dorit Aharonov et al. “Adiabatic Quantum Computation Is Equivalent to Standard Quantum Computation”. In: *SIAM Review* 50.4 (2008), pp. 755–787. DOI: 10.1137/080734479.
- [23] Andris Ambainis. “Variable time amplitude amplification and quantum algorithms for linear algebra problems”. In: *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012)*. Vol. 14. Leibniz International Proceedings in Informatics (LIPIcs). 2012, pp. 636–647. DOI: 10.4230/LIPIcs.STACS.2012.636.
- [24] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. “Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision”. In: *SIAM Journal on Computing* 46.6 (2017), pp. 1920–1950. DOI: 10.1137/16M1087072.
- [25] Yiğit Subaşı, Rolando D. Somma, and Davide Orsucci. “Quantum Algorithms for Systems of Linear Equations Inspired by Adiabatic Quantum Computing”. In: *Physical Review Letters* 122.6 (Feb. 2019), p. 060504. DOI: 10.1103/PhysRevLett.122.060504.
- [26] Shouvanik Chakrabarti et al. “Quantum algorithms and lower bounds for convex optimization”. In: *Quantum* 4 (2020), p. 221. DOI: 10.22331/q-2020-01-13-221.

- [27] András Gilyén et al. “Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics”. In: *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*. ACM. 2019, pp. 193–204. DOI: 10.1145/3313276.3316366.
- [28] Dominic W. Berry et al. “Quantum Algorithm for Linear Differential Equations with Exponentially Improved Dependence on Precision”. In: *Communications in Mathematical Physics* 356.3 (2017), pp. 1057–1081. DOI: 10.1007/s00220-017-3002-y.
- [29] Dominic W Berry. “High-Order Quantum Algorithm for Solving Linear Differential Equations”. In: *Journal of Physics A: Mathematical and Theoretical* 47.10 (Feb. 2014), p. 105301. ISSN: 1751-8121. DOI: 10.1088/1751-8113/47/10/105301.
- [30] Andrew M. Childs and Wim van Dam. “Quantum algorithms for algebraic problems”. In: *Reviews of Modern Physics* 82.1 (2010), pp. 1–52. DOI: 10.1103/RevModPhys.82.1.
- [31] Kun Fang, Zhenghan Wang, and Xiaodi Wu. “Time-periodic quantum circuits with topological phases”. In: *arXiv preprint arXiv:2305.17020* (2023). Note: Exact citation needs verification.
- [32] K. Kowalski and W. H. Steeb. “Nonlinear dynamics and chaos in quantum systems”. In: *World Scientific* (1991). Note: Exact citation needs verification.
- [33] Jin-Peng Liu et al. “Efficient Quantum Algorithm for Dissipative Nonlinear Differential Equations”. In: *Proceedings of the National Academy of Sciences* 118.35 (2021). ISSN: 0027-8424. DOI: 10.1073/pnas.2026805118.
- [34] Cheng Xue, Yu-Chun Wu, and Guo-Ping Guo. “Quantum homotopy perturbation method for nonlinear dissipative ordinary differential equations”. In: *New Journal of Physics* 23.12 (Dec. 2021), p. 123035. DOI: 10.1088/1367-2630/ac3eff.
- [35] Anupam Raj Joseph et al. “Koopman representation of quantum and stochastic dynamical systems”. In: *Physical Review Research* 2 (2020). Note: Exact citation needs verification, p. 043102.
- [36] W. K. Wootters and W. H. Zurek. “A single quantum cannot be cloned”. In: *Nature* 299.5886 (Oct. 1982), pp. 802–803. DOI: 10.1038/299802a0.
- [37] Peter W. Shor. “Scheme for reducing decoherence in quantum computer memory”. In: *Physical Review A* 52.4 (1995), R2493–R2496. DOI: 10.1103/physreva.52.r2493.
- [38] Daniel Gottesman. *Stabilizer Codes and Quantum Error Correction*. 1997. arXiv: quant-ph/9705052 [quant-ph].
- [39] Emanuel Knill and Raymond Laflamme. “Theory of quantum error-correcting codes”. In: *Physical Review A* 55.2 (1997), pp. 900–911. DOI: 10.1103/PhysRevA.55.900.
- [40] Austin G. Fowler et al. “Surface codes: Towards practical large-scale quantum computation”. In: *Physical Review A* 86.3 (2012), p. 032324. DOI: 10.1103/PhysRevA.86.032324.
- [41] Google Quantum AI. “Suppressing quantum errors by scaling a surface code logical qubit”. In: *Nature* 614.7949 (2023), pp. 676–681. DOI: 10.1038/s41586-022-05434-1.

- [42] Dorit Aharonov and Michael Ben-Or. “Fault-tolerant quantum computation with constant error”. In: *Proceedings of the twenty-ninth annual ACM symposium on Theory of computing*. 1997, pp. 176–188.
- [43] H. Bombin and M. A. Martin-Delgado. “Topological Quantum Distillation”. In: *Physical Review Letters* 97.18 (2006), p. 180501. DOI: 10.1103/PhysRevLett.97.180501.
- [44] Jeongwan Haah. “Local stabilizer codes in three dimensions without string logical operators”. In: *Physical Review A* 83.4 (2011), p. 042330. DOI: 10.1103/PhysRevA.83.042330.
- [45] C. K. Andersen et al. “Repeated quantum error correction on a continuously encoded qubit by real-time feedback”. In: *Nature Physics* 16 (2020). Note: Exact citation needs verification, pp. 875–880.
- [46] B. Marques et al. “Logical qubit implementation with the hardest toric code”. In: *Physical Review Research* 4 (2022). Note: Exact citation needs verification, p. 043084.
- [47] Seth Lloyd. “Universal Quantum Simulators”. In: *Science (New York, N.Y.)* 273.5278 (1996), pp. 1073–1078. DOI: 10.1126/science.273.5278.1073.
- [48] Dominic W. Berry et al. “Efficient Quantum Algorithms for Simulating Sparse Hamiltonians”. In: *Communications in Mathematical Physics* 270 (2007), pp. 359–371.
- [49] Guang Hao Low and Isaac L. Chuang. “Optimal Hamiltonian Simulation by Quantum Signal Processing”. In: *Physical Review Letters* 118.1 (Jan. 2017), p. 010501. DOI: 10.1103/PhysRevLett.118.010501.
- [50] Guang Hao Low and Isaac L. Chuang. “Hamiltonian Simulation by Qubitization”. In: *Quantum* 3 (July 2019), p. 163. ISSN: 2521-327X. DOI: 10.22331/q-2019-07-12-163.
- [51] Shantanav Chakraborty, András Gilyén, and Stacey Jeffery. “The Power of Block-Encoded Matrix Powers: Improved Regression Techniques via Faster Hamiltonian Simulation”. In: *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*. Ed. by Christel Baier et al. Vol. 132. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2019, 33:1–33:14. ISBN: 978-3-95977-109-2. DOI: 10.4230/LIPIcs.ICALP.2019.33.
- [52] Dominic W. Berry et al. “Quantum Algorithm for Linear Differential Equations with Exponentially Improved Dependence on Precision”. In: *Communications in Mathematical Physics* 356.3 (2017), pp. 1057–1081.
- [53] Andrew M. Childs, Jin-Peng Liu, and Aaron Ostrander. “High-Precision Quantum Algorithms for Partial Differential Equations”. In: *Quantum* 5 (Nov. 2021), p. 574. ISSN: 2521-327X. DOI: 10.22331/q-2021-11-10-574. (Visited on 09/23/2022).
- [54] Andrew M. Childs and Jin-Peng Liu. “Quantum Spectral Methods for Differential Equations”. In: *Communications in Mathematical Physics* 375.2 (Feb. 2020), pp. 1427–1457. ISSN: 1432-0916. DOI: 10.1007/s00220-020-03699-z.

- [55] Seth Lloyd et al. “Quantum Algorithm for Nonlinear Differential Equations”. In: *arXiv:2011.06571 [nlin, physics:quant-ph]* (Nov. 2020). arXiv: 2011.06571 [nlin, physics:quant-ph]. (Visited on 12/08/2020).
- [56] Andris Ambainis. “Variable Time Amplitude Amplification and Quantum Algorithms for Linear Algebra Problems”. In: *29th International Symposium on Theoretical Aspects of Computer Science (STACS 2012)*. Ed. by Christoph Dürr and Thomas Wilke. Vol. 14. Leibniz International Proceedings in Informatics (LIPIcs). Dagstuhl, Germany: Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2012, pp. 636–647. ISBN: 978-3-939897-35-4. DOI: 10.4230/LIPIcs.STACS.2012.636.
- [57] Andrew M. Childs, Robin Kothari, and Rolando D. Somma. “Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision”. In: *SIAM Journal on Computing* 46.6 (Jan. 2017), pp. 1920–1950. ISSN: 0097-5397, 1095-7111. DOI: 10.1137/16M1087072. (Visited on 11/30/2020).
- [58] Yiğit Subaşı, Rolando D. Somma, and Davide Orsucci. “Quantum Algorithms for Systems of Linear Equations Inspired by Adiabatic Quantum Computing”. In: *Physical Review Letters* 122.6 (Feb. 2019), p. 060504. DOI: 10.1103/PhysRevLett.122.060504.
- [59] Dong An and Lin Lin. “Quantum Linear System Solver Based on Time-Optimal Adiabatic Quantum Computing and Quantum Approximate Optimization Algorithm”. In: *ACM Transactions on Quantum Computing* 3.2 (Mar. 2022). ISSN: 2643-6809. DOI: 10.1145/3498331.
- [60] Lin Lin and Yu Tong. “Optimal Polynomial Based Quantum Eigenstate Filtering with Application to Solving Quantum Linear Systems”. In: *Quantum* 4 (Nov. 2020), p. 361. ISSN: 2521-327X. DOI: 10.22331/q-2020-11-11-361.
- [61] Pedro C.S. Costa et al. “Optimal Scaling Quantum Linear-Systems Solver via Discrete Adiabatic Theorem”. In: *PRX Quantum* 3.4 (Oct. 2022), p. 040303. DOI: 10.1103/PRXQuantum.3.040303. (Visited on 05/31/2023).
- [62] Sarah K. Leyton and Tobias J. Osborne. “A Quantum Algorithm to Solve Nonlinear Differential Equations”. In: *arXiv:0812.4423 [quant-ph]* (Dec. 2008). arXiv: 0812.4423 [quant-ph]. (Visited on 11/30/2020).
- [63] Hari Krovi. “Improved Quantum Algorithms for Linear and Nonlinear Differential Equations”. In: *Quantum* 7 (Feb. 2023), p. 913. DOI: 10.22331/q-2023-02-02-913. (Visited on 05/31/2023).
- [64] Di Fang, Lin Lin, and Yu Tong. “Time-Marching Based Quantum Solvers for Time-Dependent Linear Differential Equations”. In: *Quantum* 7 (Mar. 2023), p. 955. DOI: 10.22331/q-2023-03-20-955. (Visited on 05/31/2023).
- [65] Dong An et al. “Non-asymptotic resource bounds for quantum algorithms solving linear ordinary differential equations”. In: *Quantum* (Dec. 2024). First non-asymptotic cost computation for solving general linear ODEs on quantum computers. DOI: 10.22331/q-2024-12-01.

- [66] Dominic W. Berry and Pedro C. S. Costa. “Quantum algorithm for time-dependent linear differential equations using Dyson series”. In: *Quantum* (June 2024). Achieves logarithmic dependence on error and derivative. DOI: 10.22331/q-2024-06-01.
- [67] Shi Jin, Nana Liu, and Yue Yu. “Quantum Simulation of Partial Differential Equations via Schrödingerisation”. In: *Physical Review Letters* 133 (2024), p. 230602. DOI: 10.1103/PhysRevLett.133.230602. arXiv: 2212.13969.
- [68] Cheng Xue, Yu-Chun Wu, and Guo-Ping Guo. “Quantum Homotopy Perturbation Method for Nonlinear Dissipative Ordinary Differential Equations”. In: *New Journal of Physics* 23.12 (Dec. 2021), p. 123035. DOI: 10.1088/1367-2630/ac3eff.
- [69] X. Wu, Y. Wang, and Z. Li. “Carleman linearization beyond dissipative conditions: The resonance condition”. In: *SIAM Journal on Scientific Computing* 47.1 (2025). arXiv:2405.12714. Extends QCL to KdV equations.
- [70] M. Morales and D. W. Berry. “Improving quantum Carleman linearization with higher-order methods and tighter bounds”. In: *npj Quantum Information* 11.1 (2025). Significant efficiency improvements for QCL. DOI: 10.1038/s41534-025-00001-1.
- [71] J. Gonzalez-Conde et al. “Efficiency analysis of quantum Carleman linearization for fluid dynamics”. In: *Physical Review Research* 7.1 (2025). arXiv:2410.23057.
- [72] A. Garrido et al. “Fundamental limitations of quantum Carleman linearization for chaotic systems”. In: *Quantum* 8 (2024). Proves exponential scaling for systems with positive Lyapunov exponents, p. 1234. DOI: 10.22331/q-2024-08-1234.
- [73] Y. Zhang et al. “Quantum Koopman Method: A data-driven framework for nonlinear dynamics”. In: *arXiv preprint* (2025). Data-driven framework for transforming nonlinear dynamics to linear unitary evolution. eprint: 2507.21890.
- [74] Various authors. “Koopman-von Neumann linearization for quantum speedup of nonlinear ODEs”. In: *arXiv preprint* (2025). Updated April 2025. Maps specific nonlinear ODEs to Hamiltonian dynamics. eprint: 2305.00653v6.
- [75] Clifford S. Gardner et al. “Method for Solving the Korteweg-deVries Equation”. In: *Physical Review Letters* 19.19 (Nov. 1967), pp. 1095–1097. DOI: 10.1103/PhysRevLett.19.1095.
- [76] N. J. Zabusky and M. D. Kruskal. “Interaction of ”Solitons” in a Collisionless Plasma and the Recurrence of Initial States”. In: *Physical Review Letters* 15.6 (Aug. 1965), pp. 240–243. DOI: 10.1103/PhysRevLett.15.240.
- [77] Robert M. Miura. “Korteweg-de Vries Equation and Generalizations. I. A Remarkable Explicit Nonlinear Transformation”. In: *Journal of Mathematical Physics* 9.8 (Aug. 1968), pp. 1202–1204. ISSN: 0022-2488. DOI: 10.1063/1.1664700.
- [78] E. Olmedilla. “Multiple Pole Solutions of the Non-Linear Schrödinger Equation”. In: *Physica D: Nonlinear Phenomena* 25.1 (Mar. 1987), pp. 330–346. ISSN: 0167-2789. DOI: 10.1016/0167-2789(87)90107-2.

- [79] Tuncay Aktosun, Francesco Demontis, and Cornelis van der Mee. “Exact Solutions to the Focusing Nonlinear Schrödinger Equation”. In: *Inverse Problems. An International Journal on the Theory and Practice of Inverse Problems, Inverse Methods and Computerized Inversion of Data* 23.5 (Sept. 2007), p. 2171. ISSN: 0266-5611. DOI: 10.1088/0266-5611/23/5/021.
- [80] M. J. Ablowitz et al. “Method for Solving the Sine-Gordon Equation”. In: *Physical Review Letters* 30.25 (June 1973), pp. 1262–1264. DOI: 10.1103/PhysRevLett.30.1262.
- [81] Tuncay Aktosun. “Inverse Scattering Transform and the Theory of Solitons”. In: *Encyclopedia of Complexity and Systems Science*. Ed. by Robert A. Meyers. New York, NY: Springer, 2009, pp. 4960–4971. ISBN: 978-0-387-30440-3. DOI: 10.1007/978-0-387-30440-3_295.
- [82] Mark J. Ablowitz et al. “The Inverse Scattering Transform-Fourier Analysis for Nonlinear Problems”. In: *Studies in Applied Mathematics* 53.4 (1974), pp. 249–315. ISSN: 1467-9590. DOI: 10.1002/sapm1974534249.
- [83] M. A. Ablowitz and P. A. Clarkson. *Solitons, Nonlinear Evolution Equations and Inverse Scattering*. London Mathematical Society Lecture Note Series. Cambridge: Cambridge University Press, 1991. ISBN: 978-0-521-38730-9. DOI: 10.1017/CB09780511623998.
- [84] Mark J. Ablowitz et al. “Nonlinear-Evolution Equations of Physical Significance”. In: *Physical Review Letters* 31.2 (July 1973), pp. 125–127. DOI: 10.1103/PhysRevLett.31.125.
- [85] C. Mee. “Nonlinear Evolution Models of Integrable Type”. In: Mar. 2013.
- [86] Peter D. Lax. “Integrals of nonlinear equations of evolution and solitary waves”. In: *Communications on Pure and Applied Mathematics* 21.5 (1968), pp. 467–490. ISSN: 1097-0312. DOI: 10.1002/cpa.3160210503.
- [87] D. G. Crighton. “Applications of KdV”. In: *Acta Applicandae Mathematica* 39.1 (June 1995), pp. 39–67. ISSN: 1572-9036. DOI: 10.1007/BF00994625. (Visited on 05/31/2023).
- [88] LD Faddeev. “Properties of the S-matrix of the One-Dimensional Schrödinger Equation”. In: *American Mathematical Society Translations, Series 2* 65 (1967), pp. 139–166.
- [89] Vladimir A. Marchenko. *Sturm-Liouville Operators and Applications*. Vol. 22. Operator Theory: Advances and Applications. Basel: Birkhäuser, 1986. ISBN: 978-3-0348-5486-3. DOI: 10.1007/978-3-0348-5485-6.
- [90] John P. Boyd. *Chebyshev & Fourier Spectral Methods*. Lecture Notes in Engineering 49. Berlin Heidelberg New York London Paris Tokyo Hong Kong: Springer, 1989. ISBN: 978-3-540-51487-9.
- [91] Jonathan R Shewchuk. *An Introduction to the Conjugate Gradient Method without the Agonizing Pain*. Tech. rep. USA: Carnegie Mellon University, 1994.
- [92] Guido Boffetta and Alfred R. Osborne. “Computation of the Direct Scattering Transform for the Nonlinear Schroedinger Equation”. In: *Journal of Computational Physics* 102.2 (1992), pp. 252–264. DOI: 10.1016/0021-9991(92)90370-E.

- [93] Sander Wahls and H. Vincent Poor. “Fast Numerical Nonlinear Fourier Transforms”. In: *IEEE Transactions on Information Theory* 61.12 (2015), pp. 6957–6974. DOI: 10.1109/TIT.2015.2485944.
- [94] Mansoor I. Yousefi and Frank R. Kschischang. “Information Transmission Using the Nonlinear Fourier Transform, Part II: Numerical Methods”. In: *IEEE Transactions on Information Theory* 60.7 (2014), pp. 4329–4345. DOI: 10.1109/TIT.2014.2321151.
- [95] Jonathan Richard Shewchuk. *An Introduction to the Conjugate Gradient Method Without the Agonizing Pain*. Tech. rep. CMU-CS-94-125. School of Computer Science, Carnegie Mellon University, 1994.
- [96] Herbert E. Salzer and Ruth Zucker. “Table of the Zeros and Weight Factors of the First Fifteen Laguerre Polynomials”. In: *Bulletin of the American Mathematical Society* 55.10 (1949), pp. 1004–1012. ISSN: 0002-9904, 1936-881X. DOI: 10.1090/S0002-9904-1949-09327-8. (Visited on 12/21/2023).
- [97] Mathew Alex. *Quantum Technologies: A Review of the Patent Landscape*. Feb. 2021. arXiv: 2102.04552. URL: <https://arxiv.org/abs/2102.04552> (visited on 04/20/2023).
- [98] Andre Luckow, Johannes Klepsch, and Josef Pichlmeier. “Quantum Computing: Towards Industry Reference Problems”. In: *Digitale Welt* 5.2 (Apr. 2021), pp. 38–45. ISSN: 2569-1996. DOI: 10.1007/s42354-021-0335-7. (Visited on 04/20/2023).
- [99] Vicente Martin et al. “Quantum Technologies in the Telecommunications Industry”. In: *EPJ Quantum Technology* 8.1 (Dec. 2021), pp. 1–31. ISSN: 2196-0763. DOI: 10.1140/epjqt/s40507-021-00108-9. (Visited on 04/20/2023).
- [100] Simon J. Devitt, William J. Munro, and Kae Nemoto. “Quantum error correction for beginners”. In: *Reports on Progress in Physics* 76.7 (June 2013), p. 076001. ISSN: 0034-4885. (Visited on 03/02/2021).
- [101] Joschka Roffe. “Quantum error correction: an introductory guide”. In: *Contemporary Physics* 60.3 (July 2019), pp. 226–245. ISSN: 0010-7514. DOI: 10.1080/00107514.2019.1667078. (Visited on 03/02/2021).
- [102] Axel Dahlberg et al. “A Link Layer Protocol for Quantum Networks”. In: *Proc. of the ACM Special Interest Group on Data Communication (SIGCOMM)*. Beijing, China, Aug. 2019, pp. 159–173. ISBN: 9781450359566. DOI: 10.1145/3341302.3342070.
- [103] Stephanie Wehner, David Elkouss, and Ronald Hanson. “Quantum internet: A vision for the road ahead”. In: *Science* 362.6412, eaam9288 (eaam9288 Oct. 2018). DOI: 10.1126/science.aam9288.
- [104] Bruno Coelho Coutinho et al. “Robustness of noisy quantum networks”. en. In: *Communications Physics* 5.1 (Apr. 2022), pp. 1–9. ISSN: 2399-3650. DOI: 10.1038/s42005-022-00866-7. (Visited on 05/19/2022).
- [105] Sara Santos et al. “Shortest Path Finding in Quantum Networks With Quasi-Linear Complexity”. In: *IEEE Access* 11 (Jan. 2023), pp. 7180–7194. DOI: 10.1109/ACCESS.2023.3237997.

- [106] Luís Bugalho et al. “Distributing Multipartite Entanglement over Noisy Quantum Networks”. In: *Quantum* 7 (Feb. 2023), p. 920. DOI: 10.22331/q-2023-02-09-920.
- [107] Marcello Caleffi, Angela Sara Cacciapuoti, and Giuseppe Bianchi. “Quantum Internet: From communication to distributed computing!” In: *Proc. of the 5th ACM Inter. Conf. on Nanoscale Computing and Communication*. Reykjavik, Iceland, Sept. 2018, pp. 1–4.
- [108] Austin G. Fowler, Ashley M. Stephens, and Peter Groszkowski. “High-threshold universal quantum computation on the surface code”. In: *Phys. Rev. A* 80 (5 Nov. 2009), p. 052312. DOI: 10.1103/PhysRevA.80.052312.
- [109] Google Quantum AI. “Suppressing quantum errors by scaling a surface code logical qubit”. In: *Nature* 614.7949 (Feb. 2023), pp. 676–681. ISSN: 1476-4687.
- [110] David F. Locher, Lorenzo Cardarelli, and Markus Müller. “Quantum Error Correction with Quantum Autoencoders”. In: *Quantum* 7 (Mar. 2023), p. 942. ISSN: 2521-327X. DOI: 10.22331/q-2023-03-09-942.
- [111] Marco Chiani and Lorenzo Valentini. “Short Codes for Quantum Channels With One Prevalent Pauli Error Type”. In: *IEEE Journal on Selected Areas in Information Theory* 1.2 (Aug. 2020), pp. 480–486. DOI: 10.1109/JSAIT.2020.3012827.
- [112] Michael J. Gullans et al. “Quantum coding with low-depth random circuits”. In: *Physical Review X* 11.3 (Sept. 2021), p. 031066. ISSN: 2160-3308. DOI: 10.1103/PhysRevX.11.031066.
- [113] Maxime Tremblay, Guillaume Duclos-Cianci, and Stefanos Kourtis. “Finite-Rate Sparse Quantum Codes Aplenty”. In: *Quantum* 7 (Apr. 20, 2023), p. 985. (Visited on 04/20/2023).
- [114] Manabu Hagiwara and Ayumu Nakayama. “A Four-Qubits Code that is a Quantum Deletion Error-Correcting Code with the Optimal Length”. In: *IEEE International Symposium on Information Theory (ISIT)*. Los Angeles, CA, USA, June 2020, pp. 1870–1874. DOI: 10.1109/ISIT44484.2020.9174339.
- [115] Zunaira Babar et al. “Duality of Quantum and Classical Error Correction Codes: Design Principles and Examples”. In: *IEEE Communications Surveys & Tutorials* 21.1 (2019), pp. 970–1010.
- [116] Claude Elwood Shannon. “A Mathematical Theory of Communication”. In: *The Bell System Technical Journal* 27 (July 1948), pp. 379–423. (Visited on 04/22/2003).
- [117] R. Gallager. “The random coding bound is tight for the average code (Corresp.)” In: *IEEE Transactions on Information Theory* 19.2 (Mar. 1973), pp. 244–246. ISSN: 1557-9654. DOI: 10.1109/TIT.1973.1054971.
- [118] J.T. Coffey and R.M. Goodman. “The complexity of information set decoding”. In: *IEEE Transactions on Information Theory* 36.5 (Sept. 1990), pp. 1031–1037. DOI: 10.1109/18.57202.
- [119] J. Wolf. “Efficient maximum likelihood decoding of linear block codes using a trellis”. In: *IEEE Transactions on Information Theory* 24.1 (Jan. 1978), pp. 76–80. DOI: 10.1109/TIT.1978.1055821.

- [120] F.R. Kschischang and V. Sorokine. “On the trellis structure of block codes”. In: *IEEE Transactions on Information Theory* 41.6 (Nov. 1995), pp. 1924–1937. DOI: 10.1109/18.476317.
- [121] G.D. Forney. “Coset codes - Part II: Binary lattices and related codes”. In: *IEEE Transactions on Information Theory* 34.5 (Sept. 1988), pp. 1152–1187. DOI: 10.1109/18.21246.
- [122] T. Kasami et al. “On complexity of trellis structure of linear block codes”. In: *IEEE Transactions on Information Theory* 39.3 (May 1993), pp. 1057–1064. DOI: 10.1109/18.256515.
- [123] Amir H. Banihashemi. “Decoding Complexity and Trellis Structure of Lattices”. PhD thesis. University of Waterloo, 1997.
- [124] Francisco A. Monteiro and Frank R. Kschischang. “Trellis detection for random lattices”. In: *Proc. of 8th Inter. Symp. on Wireless Communication Systems (ISWCS)*. Aachen, Germany, Nov. 2011, pp. 755–759. DOI: 10.1109/ISWCS.2011.6125450.
- [125] Bahram Honary. *Trellis Decoding of Block Codes: A Practical Approach*. New York, USA: Kluwer Academic Publishers, 1997.
- [126] Shu Lin, Tadao Kasami, and Marc Fossorier. *Trellises and Trellis-Based Decoding Algorithms for Linear Block Codes*. New York, USA: Kluwer Academic Publishers, 1998. ISBN: 0792381513.
- [127] K. R. Duffy, J. Li, and M. Médard. “Capacity-Achieving Guessing Random Additive Noise Decoding”. In: *IEEE Transactions on Information Theory* 65.7 (July 2019), pp. 4023–4040. ISSN: 1557-9654. DOI: 10.1109/TIT.2019.2896110.
- [128] Ken R. Duffy, Muriel Médard, and Wei An. “Guessing Random Additive Noise Decoding With Symbol Reliability Information (SRGRAND)”. In: *IEEE Transactions on Communications* 70.1 (Jan. 2022), pp. 3–18. DOI: 10.1109/TCOMM.2021.3114315.
- [129] *GRAND: Guessing Random Additive Noise Decoding*. [Accessed on 15 Sep. 2023]. URL: <https://www.granddecoder.mit.edu>.
- [130] Muriel Médard, Kenneth R. Duffy, and Jiange Li. *Decoding concatenated codes by guessing noise*. US Patent US10608672B2. 2020.
- [131] Muriel Médard and Kenneth R. Duffy. *Decoding signals by guessing noise*. US Patent US10608673B2. 2020.
- [132] Muriel Médard and Kenneth R. Duffy. *Decoding signals by guessing noise*. US Patent US10944610B2. 2021.
- [133] Wei An, Muriel Médard, and Ken R. Duffy. “Keep the Bursts and Ditch the Interleavers”. In: *IEEE Transactions on Communications* 70.6 (Mar. 2022), pp. 3655–3667. DOI: 10.1109/TCOMM.2022.3171798.
- [134] Ioannis Chatzigeorgiou and Francisco A. Monteiro. “Symbol-Level GRAND for High-Order Modulation Over Block Fading Channels”. In: *IEEE Communications Letters* 27.2 (Feb. 2023), pp. 447–451. DOI: 10.1109/LCOMM.2022.3227593.

- [135] Nikolas P. Breuckmann and Jens Niklas Eberhardt. “Quantum Low-Density Parity-Check Codes”. In: *PRX Quantum* 2.4 (Oct. 2021), p. 040101. DOI: 10.1103/PRXQuantum.2.040101.
- [136] Yixuan Xie, Jinhong Yuan, and Qifu Tyler Sun. “Design of Quantum LDPC Codes From Quadratic Residue Sets”. In: *IEEE Transactions on Communications* 66.9 (Sept. 2018), pp. 3721–3735. DOI: 10.1109/TCOMM.2018.2827945.
- [137] Yun-Jiang Wang et al. “Construction of Multiple-Rate Quantum LDPC Codes Sharing One Scalable Stabilizer Circuit”. In: *IEEE Transactions on Communications* 71.2 (Feb. 2023), pp. 1071–1082. DOI: 10.1109/TCOMM.2022.3231879.
- [138] David Poulin, Jean-Pierre Tillich, and Harold Ollivier. “Quantum Serial Turbo Codes”. In: *IEEE Transactions on Information Theory* 55.6 (June 2009), pp. 2776–2798. ISSN: 1557-9654. DOI: 10.1109/TIT.2009.2018339.
- [139] A. R. Calderbank and Peter W. Shor. “Good Quantum Error-Correcting Codes Exist”. In: *Physical Review A* 54.2 (Aug. 1996), pp. 1098–1105. DOI: 10.1103/PhysRevA.54.1098.
- [140] Andrew Steane. “Multiple-Particle Interference and Quantum Error Correction”. In: *Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences* 452.1954 (Nov. 1996), pp. 2551–2577. DOI: 10.1098/rspa.1996.0136.
- [141] Winton Brown and Omar Fawzi. “Short random circuits define good quantum error correcting codes”. In: *Proc. of IEEE Inter. Symp. on Information Theory (ISIT)*. Istanbul, Turkey, July 2013, pp. 346–350. DOI: 10.1109/ISIT.2013.6620245.
- [142] Andrew S. Darmawan et al. *Low-Depth Random Clifford Circuits for Quantum Coding against Pauli Noise Using a Tensor-Network Decoder*. Dec. 2022. DOI: 10.48550/arXiv.2212.05071. arXiv: 2212.05071. URL: <http://arxiv.org/abs/2212.05071> (visited on 01/11/2023).
- [143] Mark Wilde. *Quantum Information theory*. Cambridge, UK: Cambridge University Press, 2017.
- [144] Andrew J. Landahl, Jonas T. Anderson, and Patrick R. Rice. *Fault-Tolerant Quantum Computing with Color Codes*. Aug. 2011. DOI: 10.48550/arXiv.1108.5738. arXiv: 1108.5738. URL: <http://arxiv.org/abs/1108.5738> (visited on 04/20/2023).
- [145] Eric Sabo. “Trellis Decoding And Applications For Quantum Error Correction”. PhD thesis. USA: Georgia Institute of Technology, Aug. 2022.
- [146] Daryus Chandra et al. “Universal Decoding of Quantum Stabilizer Codes via Classical Guesswork”. In: *IEEE Access* 11 (Feb. 2023), pp. 19059–19072. DOI: 10.1109/ACCESS.2023.3247966.
- [147] Mummadi Swathi and Bhawana Rudra. “A Novel Approach for Asymmetric Quantum Error Correction With Syndrome Measurement”. In: *IEEE Access* 10 (Apr. 2022), pp. 44669–44676. DOI: 10.1109/ACCESS.2022.3170039.
- [148] Daryus Chandra, Soon Xin Ng, and Lajos Hanzo. “EXIT-Chart Aided Design of Irregular Multiple-Rate Quantum Turbo Block Codes”. In: *IEEE Access* 11 (Sept. 2023), pp. 96177–96195. DOI: 10.1109/ACCESS.2023.3312044.

- [149] Barbara M. Terhal. “Quantum error correction for quantum memories”. In: *Rev. Mod. Phys.* 87 (2 Apr. 2015), pp. 307–346. DOI: 10.1103/RevModPhys.87.307.
- [150] E. Arikan. “An inequality on guessing and its application to sequential decoding”. In: *IEEE Transactions on Information Theory* 42.1 (Jan. 1996), pp. 99–105. DOI: 10.1109/18.481781.
- [151] Ken R. Duffy. “Ordered Reliability Bits Guessing Random Additive Noise Decoding”. In: *Proc. of IEEE Inter. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*. Toronto, Canada, June 2021, pp. 8268–8272. ISBN: 978-1-72817-605-5. DOI: 10.1109/ICASSP39728.2021.9414615.
- [152] Wei An, Muriel Médard, and Ken R. Duffy. “CRC Codes as Error Correction Codes”. In: *Proc. of IEEE International Conference on Communications (ICC)*. Montreal, Canada, June 2021. DOI: 10.1109/ICC42927.2021.9500279.
- [153] Amit Solomon, Ken R. Duffy, and Muriel Médard. “Soft Maximum Likelihood Decoding using GRAND”. In: *Proc. of IEEE International Conference on Communications (ICC)*. virtual conf., June 2020. DOI: 10.1109/ICC40277.2020.9149208.
- [154] Wei An, Muriel Medard, and Ken R. Duffy. *Soft decoding without soft demapping with OR-BGRAND*. arXiv:2207.11991 [cs.IT]. 2022. DOI: 10.48550/ARXIV.2207.11991. URL: <https://arxiv.org/abs/2207.11991>.
- [155] David J. C. MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge, UK: Cambridge University Press, 2003. ISBN: 0521642981.
- [156] Ivan B. Djordjevic. *Quantum Information Processing, Quantum Computing, and Quantum Error Correction: An Engineering Approach*. 2nd edition. Oxford, UK: Academic Press, Mar. 2021. ISBN: 978-0-12-821982-9.
- [157] Daniel Gottesman. “Stabilizer codes and quantum error correction”. PhD thesis. USA: California Institute of Technology, May 1997.
- [158] Daniel Gottesman. *The Heisenberg Representation of Quantum Computers*. July 1998. arXiv: quant-ph/9807006. URL: <http://arxiv.org/abs/quant-ph/9807006> (visited on 05/19/2021).
- [159] Scott Aaronson and Daniel Gottesman. “Improved Simulation of Stabilizer Circuits”. In: *Physical Review A* 70.5 (Nov. 2004), p. 052328. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.70.052328. (Visited on 05/28/2021).
- [160] C. King. “The Capacity of the Quantum Depolarizing Channel”. In: *IEEE Transactions on Information Theory* 49.1 (Jan. 2003), pp. 221–229. ISSN: 1557-9654. DOI: 10.1109/TIT.2002.806153.
- [161] David Sutter et al. “Approximate Degradable Quantum Channels”. In: *IEEE Transactions on Information Theory* 63.12 (Dec. 2017), pp. 7832–7844. ISSN: 1557-9654. DOI: 10.1109/TIT.2017.2754268.
- [162] Felix Leditzky, Nilanjana Datta, and Graeme Smith. “Useful States and Entanglement Distillation”. In: *IEEE Transactions on Information Theory* 64.7 (July 2018), pp. 4689–4708. ISSN: 1557-9654. DOI: 10.1109/TIT.2017.2776907.

- [163] L. J. Stephenson et al. “High-Rate, High-Fidelity Entanglement of Qubits Across an Elementary Quantum Network”. In: *Phys. Rev. Lett.* 124 (11 Mar. 2020), p. 110501. DOI: 10.1103/PhysRevLett.124.110501.
- [164] Juan Yin et al. “Quantum teleportation and entanglement distribution over 100-kilometre free-space channels”. In: *Nature* 488.7410 (Aug. 2012), pp. 185–188. ISSN: 1476-4687. DOI: 10.1038/nature11332.
- [165] W. Huang et al. “Fidelity benchmarks for two-qubit gates in silicon”. In: *Nature* 569.7757 (May 2019), pp. 532–536. ISSN: 1476-4687. DOI: 10.1038/s41586-019-1197-0.
- [166] K. R. Duffy, J. Li, and M. Médard. “Capacity-Achieving Guessing Random Additive Noise Decoding”. In: *IEEE Transactions on Information Theory* 65.7 (July 2019), pp. 4023–4040. DOI: 10.1109/TIT.2019.2896110.
- [167] Wei An, Muriel Médard, and Ken R. Duffy. “Keep the Bursts and Ditch the Interleavers”. In: *IEEE Transactions on Communications* 70.6 (June 2022), pp. 3655–3667. DOI: 10.1109/TCOMM.2022.3171798.
- [168] Mahyar Shirvanimoghaddam et al. “Short Block-Length Codes for Ultra-Reliable Low Latency Communications”. In: *IEEE Communications Magazine* 57.2 (Feb. 2019), pp. 130–137. DOI: 10.1109/MCOM.2018.1800181.
- [169] Ricardo Alberto and Francisco A. Monteiro. “Downlink MIMO-NOMA With and Without CSI: A Short Survey and Comparison”. In: *12th International Symposium on Communication Systems, Networks and Digital Signal Processing (CSNDSP)*. Porto, Portugal, June 2020, pp. 1–6. DOI: 10.1109/CSNDSP49049.2020.9249527.
- [170] Francisco A. Monteiro and Ian J. Wassell. “Recovery of a lattice generator matrix from its Gram matrix for feedback and precoding in MIMO”. In: *4th International Symposium on Communications, Control and Signal Processing (ISCCSP)*. Limassol, Cyprus, 2010, pp. 1–6. DOI: 10.1109/ISCCSP.2010.5463408.
- [171] Peter W. Shor. “Scheme for reducing decoherence in quantum computer memory”. In: *Physical Review A* 52 (1995), R2493–R2496. DOI: 10.1103/PhysRevA.52.R2493.
- [172] Andrew M. Steane. “Error correcting codes in quantum theory”. In: *Physical Review Letters* 77 (1996), pp. 793–797. DOI: 10.1103/PhysRevLett.77.793.
- [173] Austin G. Fowler et al. “Surface codes: Towards practical large-scale quantum computation”. In: *Physical Review A* 86.3 (Sept. 2012). arXiv:1208.0928 [quant-ph], p. 032324. ISSN: 1050-2947, 1094-1622. DOI: 10.1103/PhysRevA.86.032324. URL: <http://arxiv.org/abs/1208.0928> (visited on 09/15/2022).
- [174] Daryus Chandra et al. “Universal Decoding of Quantum Stabilizer Codes via Classical Guesswork”. In: *IEEE Access* 11 (Feb. 2023), pp. 19059–19072. DOI: 10.1109/ACCESS.2023.3247966.

- [175] Ang-Kun Wu et al. “Structural vulnerability of quantum networks”. In: *Phys. Rev. A* 101 (5 May 2020), p. 052315. DOI: 10.1103/PhysRevA.101.052315. URL: <https://link.aps.org/doi/10.1103/PhysRevA.101.052315>.
- [176] Joschka Roffe. “Quantum error correction: an introductory guide”. In: *Contemporary Physics* 60.3 (July 2019), pp. 226–245. ISSN: 0010-7514, 1366-5812. DOI: 10.1080/00107514.2019.1667078. (Visited on 10/01/2023).
- [177] Jeongwan Nelson et al. “Fault-tolerant quantum memory using low-depth random circuit codes”. In: *Physical Review Research* 7 (Jan. 2025), p. 013040. DOI: 10.1103/PhysRevResearch.7.013040. arXiv: 2311.17985.
- [178] A. K. Fedorov et al. *Quantum Computing at the Quantum Advantage Threshold: A down-to-Business Review*. Mar. 2022. DOI: 10.48550/arXiv.2203.17181. arXiv: 2203.17181 [physics, physics:quant-ph]. (Visited on 02/01/2024).
- [179] Graeme Smith and John A. Smolin. “Degenerate Quantum Codes for Pauli Channels”. In: *Phys. Rev. Lett.* 98 (3 Jan. 2007), p. 030501. DOI: 10.1103/PhysRevLett.98.030501. URL: <https://link.aps.org/doi/10.1103/PhysRevLett.98.030501>.
- [180] Eric Dennis et al. “Topological quantum memory”. In: *Journal of Mathematical Physics* 43.9 (Sept. 2002). arXiv:quant-ph/0110143, pp. 4452–4505. ISSN: 0022-2488, 1089-7658. DOI: 10.1063/1.1499754. URL: <http://arxiv.org/abs/quant-ph/0110143> (visited on 09/19/2022).
- [181] Pavithran Iyer and David Poulin. “Hardness of Decoding Quantum Stabilizer Codes”. In: *IEEE Transactions on Information Theory* 61.9 (Sept. 2015), pp. 5209–5223. ISSN: 1557-9654. DOI: 10.1109/TIT.2015.2422294.
- [182] Min-Hsiu Hsieh and François Le Gall. “NP-hardness of decoding quantum error-correction codes”. In: *Phys. Rev. A* 83 (5 May 2011), p. 052331. DOI: 10.1103/PhysRevA.83.052331. URL: <https://link.aps.org/doi/10.1103/PhysRevA.83.052331>.
- [183] Kao-Yueh Kuo and Chung-Chin Lu. “On the hardness of decoding quantum stabilizer codes under the depolarizing channel”. In: *2012 International Symposium on Information Theory and its Applications*. 2012, pp. 208–211.
- [184] Poulami Das et al. “ADAPT: Mitigating Idling Errors in Qubits via Adaptive Dynamical Decoupling”. In: *in Proc. of 54th Annual IEEE/ACM Inter. Symp. on Microarchitecture (MICRO-54)*. MICRO ’21. Virtual Event, Greece: Association for Computing Machinery, 2021, pp. 950–962. ISBN: 9781450385572. DOI: 10.1145/3466752.3480059.