Enlarging the GKP stabilizer group for enhanced noise protection

Jonathan Pelletier[1,2] and Baptiste Royer[1,2]

[1]Département de Physique, Université de Sherbrooke, Sherbrooke, Qc. J1K 2R1, Canada
[2]Institut Quantique, Université de Sherbrooke, Sherbrooke, Qc. J1K 2R1, Canada

Encoding a qubit in a larger Hilbert space of an oscillator is an efficient way to protect its quantum information against decoherence. Promising examples of such bosonic encodings are the Gottesman-Kitaev-Preskill (GKP) codes. In this work, we investigate how redefining the stabilizer group of the GKP codes to include all operations with trivial action on the code space can contribute to the search for an optimal implementation of a logical circuit when it is affected by noise. We find the generators of the Gaussian stabilizer group, allowing us to search for different physical implementations of a Clifford operation. We then propose an algorithm that finds the optimal implementation of a given logical Clifford circuit on GKP codes, such that the state is less affected by loss errors during the computation. Finally, we demonstrate numerically, with logical randomized benchmarking, that such a compiler can increase the lifetime of square-GKP qubits while running Clifford circuits, compared to a random walk compiler.

1 Introduction

As we move toward and beyond advanced noisy intermediate-scale quantum computing, the roles of quantum error correction (QEC) and compilation techniques become increasingly important. QEC ensures that the errors caused by decoherence channels during a computation do not accumulate too quickly, while compilation techniques can reduce the rate at which they occur in the first place. The combination of both is essential to move toward fault-tolerant computations and useful applications. QEC consist of redundantly encoding quantum information in a larger quantum system such that small physical errors can be detected and corrected before the full system is affected by its spreading. Systems that enable bosonic encodings are promising platforms for QEC, where the Hilbert space of possibly many high-quality harmonic oscillator modes can be used to encode quantum information. The increase in the ratio of redundancy over the number of physical devices needed compared to other options, like qubit codes, is an encouraging perspective toward the scaling of such implementations in a hardware-efficient manner. Bosonic codes can be implemented in many architectures, such as cavity quantum electrodynamics [1, 2, 3, 4], optical modes [5, 6] and trapped ions [7, 8, 9], to cite only a few. Amplitude damping (bosonic loss) is the main type of error channel affecting those architectures, followed by dephasing and other higher-order non-linear processes. The code introduced by Gottesman, Kitaev and Preskill (GKP) [10] can correct small displacement errors, and has been observed to be very efficient to protect against amplitude damping, while remaining com-

petitive against other types of noise [11, 12]. This has created a lot of interest to study the limitations of GKP codes and potential ways to achieve universal quantum computation. For example, with the ability to apply arbitrary Clifford gates, it is possible to achieve universal computation with the addition of magic state injection [13]. This path to universal computation is particularly interesting for GKP codes because Clifford operations can be realized with Gaussian operations only, for which the spread of displacement error can be controlled. If we instead consider finite-energy GKP codes and the bosonic loss, the effect of the error channel becomes dependent on the particular Gaussian operations used to implement the logical operation. Solving this issue and finding good compiling algorithms for Clifford circuits on finite energy GKP codes is then crucial to achieve universal quantum computing with bosonic codes.

In the present work, we study this problem of finding an optimal implementation of a Clifford circuit on a multimode finite energy GKP encoding, while it is affected by bosonic loss. To attain this goal, we first identify all logically equivalent implementations of a given logical operation on the ideal code. It is parametrized by the stabilizer group, which we define here to be the (potentially non-abelian) group of all operations that stabilize the code space. This leads us to include the standard translations symmetries of GKP codes, but also an infinite number of non-commuting stabilizers. Focusing on Clifford operations, we restrict this group to Gaussian operations and define the Gaussian GKP stabilizer group. Along the way, we solve a quadratic matrix equation over the restricted ring of integer matrices to fully characterize the generators of the group. Moving on to an application on finite energy GKP, we find that each GKP codeword implementation can be described by a general Fock-type envelope that generalizes the concept of finite-energy GKP codewords [14]. For the sake of completeness, we include an updated version of the so-called small-Big-small (sBs) stabilization protocol for those codewords. We find that the logical information contained within a finite energy GKP state is independent of the envelope, and is only dependent on the underlying logical grid, which create redundancy on how we choose to implement a logical state. A consequence of the redundancy is that choosing a specific implementation of a gate is equivalent to choosing an implementation of a codeword, which is also equivalent to choosing an envelope. This enables us to find clear optimization criteria on Clifford circuits based on the characteristics of the bosonic loss channel and their effects on the envelope of the codewords. In the last part of this work, we propose an algorithm that compiles Clifford circuits using those criteria. Finally, we use logical randomized benchmarking to compare our compiler algorithm to a random walk compiler algorithm [15] and show the gain of using the former. In the appendices, we included supplementary results that are more technical, but still of interest for the community. In Sect. C, we present the generalization of the sBs protocol to any Fock-type envelope. In Sects. D to F, we included a derivation of the Wigner function of the codespace projector for any GKP code, the derivation of the Wigner function of any finite energy GKP grid state and a computation of the number of excition contained in a finite energy GKP grid state, respectively.

The paper is organized as follows. We start in Sect. 2 with useful definitions and notations on GKP codes, their representation and operations on harmonic oscillators. In Sect. 3, we present and characterize the Gaussian stabilizer group for GKP codes. It is followed by its application on finite energy GKP codes to find optimization criteria in Sect. 4. In Sect. 5, we introduce our compiler algorithm and analyze its performance. We conclude in Sect. 6.

2 Preliminaries and notation

In this section, we review the construction of multimode GKP codes as lattice codes [10, 16, 17, 18]. GKP codes are stabilizer codes which use the infinite-dimensional quantum Hilbert space of N bosonic modes to encode a finite-dimensional logical subspace. We denote the creation and annihilation operators for the k^{th} mode as $\hat{a}_k$ and $\hat{a}_k^\dagger$, respectively, with commutation relation $[\hat{a}_j, \hat{a}_k^\dagger] = \delta_{j,k}$ for $j, k = 1, ..., N$. We also make use of the quadrature coordinates $\hat{q}_k = (\hat{a}_k + \hat{a}_k^\dagger)/\sqrt{2}$ and $\hat{p}_k = -i(\hat{a}_k - \hat{a}_k^\dagger)/\sqrt{2}$ with $\hbar = 1$. We arrange these coordinates in a vector of operators $\hat{\boldsymbol{\xi}} = (\hat{\mathbf{q}}, \hat{\mathbf{p}})^{\mathrm{T}}$ in the $qqpp$ convention such that the commutation relation becomes $[\hat{\xi}_j, \hat{\xi}_k] = i\Omega_{jk}$ for $j, k = 1, ..., 2N$. The matrix

$$\Omega_{2N} = \begin{pmatrix} 0 & 1 \\ -1 & 0 \end{pmatrix} \otimes I_N \tag{1}$$

gives rise to the skew-symmetric bilinear form (symplectic form)

$$\omega(\mathbf{u}, \mathbf{v}) = \mathbf{u}^T \Omega \mathbf{v}, \tag{2}$$

defined for $\mathbf{u}, \mathbf{v} \in \mathbb{R}^{2N}$. We omit the subscript of Ω_{2N} when the dimension is clear from the context. We define the translation operator as

$$\hat{T}(\mathbf{v}) = \mathrm{e}^{-il\mathbf{v}^T \Omega \hat{\boldsymbol{\xi}}}, \tag{3}$$

from which the relation

$$\hat{T}(\mathbf{u})\hat{T}(\mathbf{v}) = \mathrm{e}^{-il^2 \mathbf{u}^T \Omega \mathbf{v}}\ \hat{T}(\mathbf{v})\hat{T}(\mathbf{u}) = \mathrm{e}^{\frac{-il^2}{2} \mathbf{u}^T \Omega \mathbf{v}}\ \hat{T}(\mathbf{u} + \mathbf{v}) \tag{4}$$

is implied by the commutation relation defined before. When using translations with units of $l = \sqrt{2\pi}$, the commutativity of two translation operators is tantamount to the symplectic form being an integer:

$$[\hat{T}(\mathbf{u}), \hat{T}(\mathbf{v})] = 0 \iff \omega(\mathbf{u}, \mathbf{v}) \in \mathbb{Z}. \tag{5}$$

These translation operators form representations of the Heisenberg group $H_{2N+1}(\mathbb{R})$ on the infinite Hilbert space of N oscillators. GKP codes are stabilizer codes which use this structure to embed one or multiple finite-dimensional representations of the Weyl-Heisenberg group on a finite-dimensional logical subspace using the translation operators. In more detail, the stabilizer of a GKP code is an abelian group generated by a set of $2N$ translation operators

$$\mathcal{S} = \left\langle \mathrm{e}^{2\pi i \mu_1} \hat{T}(\mathbf{s}_1), \mathrm{e}^{2\pi i \mu_2} \hat{T}(\mathbf{s}_2), ..., \mathrm{e}^{2\pi i \mu_{2N}} \hat{T}(\mathbf{s}_{2N}) \right\rangle, \tag{6}$$

with $\{\mathbf{s}_j\}$ a basis for $\mathbb{R}^{2N}$ and $\boldsymbol{\mu}$ the gauge vector. In most cases we take the trivial gauge $\boldsymbol{\mu} = 0$. The abelianity condition of $\mathcal{S}$ translate to the matrix

$$A = S\Omega S^T \tag{7}$$

being integral, where the matrix S is constructed from $\{\mathbf{s}_j\}$ as row vectors. We refer to A as the symplectic Gram matrix. The GKP stabilizer group is then isomorphic to a lattice $\Lambda \subset \mathbb{R}^{2N}$ generated by the matrix S with the elements given by

$$\Lambda = \{S^T \mathbf{a} \mid \mathbf{a} \in \mathbb{Z}^{2N}\}. \tag{8}$$

In the stabilizer formalism, we obtain the generalized logical Pauli operators as operators that commute with all of the stabilizers, but are not in the stabilizer: $\mathcal{C}(\mathcal{S})/\mathcal{S}$. Using the isomorphism between the stabilizer group and the lattice Λ, we can show that the Pauli operators coincide with the definition of the symplectic dual lattice Λ^*

$$\Lambda^* = \{\mathbf{u} \mid S\Omega\mathbf{u} \in \mathbb{Z}^{2N}\}. \tag{9}$$

A potential generator matrix for the symplectic dual lattice is

$$S^* = A^{-1}S, \tag{10}$$

such that we can also write

$$\Lambda^* = \{\Omega S^{-1}\mathbf{b} \mid \mathbf{b} \in \mathbb{Z}^{2N}\}. \tag{11}$$

As a result, the logical Pauli operators are each associated with one of the $\det(\Lambda) = \det(A) = d^2$ cosets of Λ^*/Λ, where d is the dimension of the finite-dimensional encoded Hilbert space.

An important type of unitary operation on oscillators is Gaussian operations, which are generated by Hamiltonians quadratic in mode operators

$$\hat{U}_M = \exp\left(\frac{-i}{2}\hat{\boldsymbol{\xi}}^T C\hat{\boldsymbol{\xi}}\right), \quad C = C^T, \tag{12}$$

where C is a symmetric matrix that specifies the generating Hamiltonian. The group of Gaussian unitaries can also be parametrized with symplectic matrices $M \in \mathrm{Sp}_{2N}(\mathbb{R})$, where the connection with Eq. (12) is $M = \exp(\Omega C)$. The unitaries $\hat{U}_M$ are representations of the metaplectic group, which is a double cover of the symplectic group. However, it is possible to lift those projective representations to ordinary representations by choosing the right representatives for each M. Doing so, the unitaries respect the homomorphism property

$$\hat{U}_N\hat{U}_M = \hat{U}_{NM}. \tag{13}$$

The action of Gaussian unitaries on mode operators is a linear application of the symplectic representative

$$U_M^\dagger\hat{\boldsymbol{\xi}}U_M = M\hat{\boldsymbol{\xi}}, \tag{14}$$

where we have an element-wise product on the left and matrix multiplication on the right. This implies the more general relation

$$U_M^\dagger f\left(\hat{\boldsymbol{\xi}}\right)U_M = f\left(M\hat{\boldsymbol{\xi}}\right), \tag{15}$$

for any function $f(\hat{\boldsymbol{\xi}})$ that can be represented as a power series of mode operators. Any symplectic matrix M can be written as a a product of two orthogonal matrices and a positive definite diagonal matrix via its Bloch-Messiah decomposition:

$$M = O_2\Gamma O_1, \tag{16}$$

where $O_i \in \mathrm{Sp}(2N,\mathbb{R}) \cap \mathrm{O}(2N) \cong U(N)$ and $\Gamma = D \oplus D^{-1}$ with D is a positive definite diagonal matrix of dimension N.

In this work, we simulate GKP states directly using their Wigner function, defined as

$$W(\boldsymbol{\xi}, \hat{\rho}) = \int_{\mathbb{R}^{2N}} d\boldsymbol{v}\, \mathrm{e}^{-i2\pi\boldsymbol{\xi}^T\Omega\boldsymbol{v}}\, \mathrm{Tr}\left[\hat{T}(\boldsymbol{v})\,\hat{\rho}\right]. \tag{17}$$

This version of the Wigner function is corrected for our specific version of the translation operator and the $l = \sqrt{2\pi}$ scaling for the units of $\boldsymbol{\xi}$. Consequently, the average of an operator $\hat{A}$ in phase-space is given by

$$\left\langle \hat{A} \right\rangle = \mathrm{Tr}\left(\hat{\rho}\hat{A}\right) = (2\pi)^{2N} \int_{\mathbb{R}^{2N}} d\boldsymbol{\xi}\, W\left(\boldsymbol{\xi}, \hat{\rho}\right) W\left(\boldsymbol{\xi}, \hat{A}\right). \tag{18}$$

When a general gaussian unitary acts on the density matrix $U_{M,\boldsymbol{\lambda}} : \hat{\rho} \to U_{M,\boldsymbol{\lambda}}\, \hat{\rho}\, U_{M,\boldsymbol{\lambda}}^{\dagger}$, the Wigner function is modified to

$$W\left(\boldsymbol{\xi}, U_{M,\boldsymbol{\lambda}}\, \hat{\rho}\, U_{M,\boldsymbol{\lambda}}^{\dagger}\right) = W\left(M^{-1}\left[\boldsymbol{\xi} - \boldsymbol{\lambda}\right], \hat{\rho}\right), \tag{19}$$

where $U_{M,\boldsymbol{\lambda}} = \hat{T}\left(\boldsymbol{\lambda}\right) \hat{U}_M$. An important class of states in harmonic oscillators are Gaussian states. They are the states that are fully described by their first and second moments:

$$\boldsymbol{\mu} = \left\langle \hat{\boldsymbol{\xi}} \right\rangle / \sqrt{2\pi}, \qquad \Sigma_{jk} = \frac{1}{4\pi} \left\langle \left\{ \hat{\boldsymbol{\xi}}_j - \sqrt{2\pi}\boldsymbol{\mu}_j, \hat{\boldsymbol{\xi}}_k - \sqrt{2\pi}\boldsymbol{\mu}_k \right\} \right\rangle, \tag{20}$$

where $\{\cdot,\cdot\}$ is the anticommutator. In phase-space, they are represented by gaussian Wigner functions

$$G_{\Sigma,\boldsymbol{\mu}}\left(\boldsymbol{\xi}\right) = \frac{\exp\left(-\frac{1}{2}\left(\boldsymbol{\xi} - \boldsymbol{\mu}\right)^T \Sigma^{-1} \left(\boldsymbol{\xi} - \boldsymbol{\mu}\right)\right)}{\sqrt{\left(2\pi\right)^{2N} \det\left(\Sigma\right)}}. \tag{21}$$

Finally, we define, on a bipartite qubit system jk, the generalized controlled Pauli operation of applying Pauli operator B on subsystem k if subsystem j is in the -1 eigenstate of Pauli operator A as

$$C\left(A_j, B_k\right) = \frac{I_j I_k + A_j I_k + I_j B_k - A_j B_k}{2}. \tag{22}$$

This symbol is symmetric in the sense that $C\left(A_j, B_k\right) = C\left(B_k, A_j\right)$. Using this notation, a controlled Z gate is noted as $C\left(Z_1, Z_2\right)$ and a controlled X gate with control qubit 1 and target qubit 2 is noted as $C\left(Z_1, X_2\right)$.

3 Gaussian stabilizer group

3.1 Generalized Stabilizer Groups

Qubit stabilizer codes use n qubits to encode $k \leq n$ qubits. The codespace $\mathcal{T}$ of these codes is the submanifold of the full Hilbert space $\left(\mathbb{C}^2\right)^n$ restricted by the elements of the stabilizer group $\mathcal{S}$ containing Pauli operators such that

$$\mathcal{T} = \left\{ |\psi\rangle : s\,|\psi\rangle = |\psi\rangle\ \forall s \in \mathcal{S} \right\}. \tag{23}$$

In the usual definition, the stabilizer group of a qubit code is generated by $n-k$ commuting elements, and it completely defines the code [19]. In this definition of the stabilizer group, it is taken to be abelian. However, it is possible to relax this assumption of abelianity and only ask that the stabilizers need to share the 2^k eigenvectors that describe T with eigenvalue 1. This is equivalent to asking that the action of the stabilizers on the codespace commute, but not necessarily the operators themselves:

$$S_j S_k\,|\psi\rangle = S_k S_j\,|\psi\rangle\ \ \forall\ \ |\psi\rangle \in T \implies [S_j, S_k] = 0. \tag{24}$$

The XS [20] and XP [21] formalisms are two examples where qubit codes were constructed from a non-abelian stabilizer group. In general, removing this restriction to abelian groups enables us to define a new stabilizer group that we call the unitary stabilizer :

$$\mathcal{S}_U = \{U \in U\left(2^n\right) \mid U\left|\psi\right\rangle = \left|\psi\right\rangle \forall \left|\psi\right\rangle \in T\}\,. \tag{25}$$

The stabilizer group described in [19] is, by definition, a subgroups of $\mathcal{S}_U$. Similar to the abelian elements, the unitary stabilizers can be used as a resource for stabilization, error mitigation, and potentially decoding. This notion of unitary stabilizer groups can also be used for continuous-variable codes such as the GKP encoding, in which case it becomes of infinite order. It is worth noting that, contrary to the abelian stabilizer, unitary stabilizers might mix error spaces or even entangle those subspaces together, which might be devastating for the preservation of information in the system. With the definition of Eq. (25), we can understand those symmetry operations as encoded identities that are implemented via a non-identity operation on the system.

A way to identify such symmetry of a code is to compare the group order of a logical operation with its physical implementation. If they have a mismatching order, then necessarily there will exist a symmetry that will be included in $\mathcal{S}_U$. For example, let g be a unitary operation on an arbitrary Hilbert space that implements a logical operator $\overline{g}$ of finite order $|\overline{g}|$. If g is an adequate representation of $\overline{g}$, then, necessarily,

$$g^{r|\overline{g}|} \in \mathcal{S}_U \quad \forall r \in \mathbb{Z}, \tag{26}$$

as $g^{|\overline{g}|}$ must implement a logical identity. An example of such operations for the square GKP code is the logical gate $\sqrt{\overline{H}}$ implemented via the unitary [18, 22]:

$$\mathrm{Imp}\left(\sqrt{\overline{H}}\right) = \exp\left(\frac{i\pi}{8}\hat{n}^2\right). \tag{27}$$

Since $\hat{n}$ has integer eigenvalues, this physical operation has order 16, while the underlying logical operation, $\sqrt{\overline{H}}$, has order 4. As a result, the four unitaries

$$\left\{\exp\left(\frac{ir\pi}{2}\hat{n}^2\right)\right\}_{r\in\mathbb{Z}_4}, \tag{28}$$

implement a logical identity on the square GKP code. For infinite-dimensional Hilbert spaces, the physical implementation of a logical operation might be of infinite order, resulting in the inclusion of an infinite subgroup in $\mathcal{S}_U$. This case will arise in our treatment of the GKP code with squeezing operations. There might even exist stabilizing operations that do not correspond to powers of non-trivial logical operations. As a whole, the rich structure of the unitary stabilizer group makes it impractical to try to find every element in $\mathcal{S}_U$. Following this thread of ideas, we restrict the scope of this paper to grid codes. We will also only be interested in a subgroup that is more easily implementable, *i.e.* Gaussian operations. For qubit stabilizer codes, the restriction of the unitary stabilizer group to Clifford operations has been studied [23, 24] to optimize the implementation of logical Clifford operations. This work extends this notion to GKP codes.

3.2 Gaussian Stabilizer Group

The elements of any generalized stabilizer group on grid codes need to share a specific property: they need to stabilize the Pauli subgrids $\mathcal{P}_j \subset \Lambda^*$ individually:

$$\mathcal{P}_j = \Lambda + \mathbf{p}_j \quad \mathbf{p}_j \in \Lambda^*, \tag{29}$$

with an index j for every logical Pauli string. By definition, $\mathcal{P}_0$ is Λ. This gives a new definition of the group as the intersection of all automorphisms of these subgrids

$$\mathcal{S}_U = \bigcap_j \mathrm{Aut}\left(\mathcal{P}_j\right) \subset \mathrm{Aut}\left(\Lambda^*\right). \tag{30}$$

For grid codes, an important subgroup of $\mathcal{S}_U$ is the Gaussian stabilizer group, which we define as the intersection of the unitary stabilizer group and the Gaussian unitaries

$$\mathcal{S}_G = \mathcal{S}_U \cap \{U_M | M \in \mathrm{Sp}_{2N}(\mathbb{R})\}. \tag{31}$$

Since any Gaussian unitary acts as an affine transformation on phase space, and stabilizers in $\mathcal{S}_G$ must be logical operations of the code, we know that each element of $\mathcal{S}_G$ admits the decomposition [25]

$$S_G \subset \mathrm{Aut}_\infty^S\left(\Lambda^*\right) = \mathrm{Aut}^S\left(\Lambda^*\right) \ltimes \Lambda^*. \tag{32}$$

Such affine transformations are characterized by two transformations: a fixed point symplectic transformation $M \in \mathrm{Aut}^S\left(\Lambda^*\right)$ and a translation by a vector of $\boldsymbol{\lambda}^* \in \Lambda^*$. We denote such transformations by $\mathcal{L}_{M,\lambda^*}$ with its application

$$\mathcal{L}_{M,\boldsymbol{\lambda}^*}\left(\boldsymbol{u}\right) = M\boldsymbol{u} + \boldsymbol{\lambda}^*. \tag{33}$$

With respect to the condition in Eq. (30), these morphisms describe an element of S_G when

$$\mathcal{L}_{M,\boldsymbol{\lambda}^*}\left(\mathcal{P}_j\right) = \mathcal{P}_j \quad \forall j. \tag{34}$$

Expanding the restrictriction in terms of the original lattice $\mathcal{P}_0$ moved by a vector of the dual lattice $\boldsymbol{p}_j$, we obtain a much simpler condition on M:

$$\mathcal{L}_{M,\boldsymbol{\lambda}^*}\left(\mathcal{P}_j\right) = \mathcal{L}_{M,\boldsymbol{\lambda}^*}\left(\mathcal{P}_0\right) + M\mathbf{p}_j = \mathcal{P}_0 + \mathbf{p}_j, \tag{35}$$

$$\left(M - I\right)\mathbf{p}_j \in \Lambda \quad \forall j. \tag{36}$$

Using both the lattice and the dual lattice definition in terms of their generator matrix (S and S^*), we conclude that elements of S_G must have a fixed-point transformation of the form

$$M = S^T\left(XA + I\right)S^{-T}, \tag{37}$$

where $X \in \mathrm{Mat}_{2N}\left(\mathbb{Z}\right)$ is a general matrix of integers and $\left(XA + I\right) \in \mathrm{SL}_{2N}\left(\mathbb{Z}\right)$ (see Sect. A). The matrix M is a valid solution only if it is also symplectic, which is not implied by Eq. (37). Enforcing the symplectic structure to M returns an equation for X

$$XAX^T - \left(X - X^T\right) = 0, \tag{38}$$

with A the symplectic Gram matrix that defines the lattice. From [10, 26], we know that for every integrally symplectic matrix $A = S\Omega S^T$, there is a change of basis $R \in \mathrm{SL}_{2N}\left(\mathbb{Z}\right)$ that transforms S and A as

$$S_D = RS, \quad A_D = RAR^T = \Omega_2 \otimes D, \tag{39}$$

such that D is a diagonal matrix of natural numbers that fix the type of lattice. Since A and A_D describe the same lattice Λ, the transformation M must be valid for both. As a result, if the pair (X_D, A_D) is a solution to Eq. (38), then so must be $\left(R^T X_D R, A\right)$. As a result, we only have to solve the equation

$$X_D A_D X_D^T - \left(X_D - X_D^T\right) = 0 \tag{40}$$

which is simpler since A_D is skew-symmetric with only N independent non-zero elements. The solutions of Eq. (40) form a group with a product $\circ$ defined as

$$X_D \circ Y_D = X_D + Y_D + X_D A_D Y_D, \tag{41}$$

with X_D, Y_D solutions to Eq. (40). The group product for solutions in Eq. (41) is also the group product for solutions in a non-canonical basis, such as the solutions of Eq. (38).

To fully describe the transformations M, we now proceed to find the generators of this group of solutions using the generators of the symplectic algebra $\mathfrak{sp}(2N, \mathbb{R})$. Looking at Eq. (37), we can deduce that XA must be similar in structure to an element that is a small deviation from the identity, which also respects the symplectic property. This is equivalent to saying that XA has the same structure as an element of the symplectic algebra $\mathfrak{sp}(2N, \mathbb{R})$. A basis for $\mathfrak{sp}(2N, \mathbb{R})$ can be chosen to be the union of two sets. The first set contains symmetric matrices

$$F_{j,k} = E_{j,k} + E_{k,j}, \qquad F_{j,j} = E_{j,j}, \tag{42}$$

and the second contains skew-symmetric matrices

$$G_{j,k} = E_{j,k} - E_{k,j}, \tag{43}$$

with $E_{j,k}$ the matrix with 1 at position (j, k) and 0 elsewhere. We use those basis elements to search for solutions of Eq. (40). Starting with the symmetric set, we fix the solution to $X_D = \alpha F_{j,k}$ with $k \geq j$ and try to solve for $\alpha \in \mathbb{Z}$. This type of matrix is a non-zero solution of Eq. (40) only if

$$A_{D,j,k} = 0. \tag{44}$$

This condition is validated for all values of α and all pairs of indices (j, k), except for combinations of the form $(j, k) = (j, N + j)$. We also find that the product rule in Eq. (41) is simplified to

$$\alpha_1 F_{j,k} \circ \alpha_2 F_{j,k} = (\alpha_1 + \alpha_2) F_{j,k} \tag{45}$$

when restricted to this infinite subgroup of solutions. This means that solutions $F_{j,k}$ generate all other solutions of the form $\alpha F_{j,k}$, $\alpha \in \mathbb{Z}$ via products with itself. By counting the number of valid (j, k) pairs, we obtain $2N^2$ generators of this type. Moving on to skew-symmetric solutions, we fix $X_D = \beta G_{j,k}$ with $\beta \in \mathbb{Z}$. This matrix is a solution only if

$$2 - \beta A_{D,j,k} = 0. \tag{46}$$

Because of the structure of A_D, this can only be validated for pairs of indices such that $(j, k) = (j, N + j)$, which complement the symmetric set of solutions. Additionally, the relation of Eq. (46) can also only be validated if the matrix A_D contains elements $A_{D,j,k} \in \{1, 2\}$. As such, we get two types of potential solutions: $(\beta = 1, A_{D,j,k} = 2)$ and $(\beta = 2, A_{D,j,k} = 1)$. Both correspond to an involution operation such that

$$\beta G_{j,k} \circ \beta G_{j,k} = 0. \tag{47}$$

Futhermore, only when $A_{D,j,k} = 1$, we get an order four solution: $X_D = G_{j,k} + F_{j,j} + F_{k,k}$ that squares to the solution of Eq. (46). In total, we have between $2N^2$ and $2N^2 + N$ generators of solutions depending on the number of diagonal elements of D that are either 1 or 2. This corresponds to the maximum number of generators possible, which means that we found all solutions to Eq. (40).

Now that the fixed-point part of the transformation M is determined, it only remains to fix $\boldsymbol{\lambda}^*$ to fully parametrize the transformation $\mathcal{L}_{M,\boldsymbol{\lambda}^*}$. The fixed-point transformation was obtained by fixing each Pauli grid to itself. However, this restriction does not restrain the phases that a Pauli representative can gain when the symplectic unitary is applied. This means that the symplectic unitary U_M is, in the general case, an implementation of a logical Pauli operator. To adjust this phase, we choose $\boldsymbol{\lambda}^*$ such that the whole transformation commutes with every Pauli representative $\mathbf{p}_j$ on the code space

$$\hat{T}\left(\boldsymbol{\lambda}^*\right) U_M \hat{T}\left(\mathbf{p}_j\right) |\psi\rangle = \hat{T}\left(\mathbf{p}_j\right) \hat{T}\left(\boldsymbol{\lambda}^*\right) U_M |\psi\rangle . \tag{48}$$

The Pauli representatives $\hat{T}\left(\mathbf{p}_j\right)$ might not commute in general with $\mathcal{L}_{M,\boldsymbol{\lambda}^*}$ as operators, but their action need to commute on the code space. Requiring that Eq. (48) is true entails that the condition is also true after a transformation $\mathbf{p}_j \to \mathbf{p}_j + \boldsymbol{\lambda} \, \forall \, \boldsymbol{\lambda} \in \Lambda$. Equation (48) leads to a set of $2N$ equations derived in appendix B that always have a solution $\boldsymbol{\lambda}^* \in \Lambda^*$. This choice fixes $\mathcal{L}_{M,\boldsymbol{\lambda}^*}$ up to a translation stabilizer of the code. In general, we only need to identify the action of U_M on the Pauli representative and inverse it with a translation by the corresponding $\boldsymbol{\lambda}^*$.

In the end, for each generator of solution X_k of Eq. (38), we get a gaussian stabilizer group generator $U_{M_k,\boldsymbol{\lambda_k}^*}$, that implements the transformations $\mathcal{L}_{M_k,\boldsymbol{\lambda_k}^*}$. We can now write the Gaussian stabilizer group as the group generated by the $2N$ translation stabilizers (indexed by j) and the $2N^2 + m$ Gaussian stabilizer generators $U_{M_k,\boldsymbol{\lambda_k}^*}$

$$\mathcal{S}_G = \left\langle \hat{T}(\mathbf{s}_j), U_{M_k,\boldsymbol{\lambda_k}^*} \right\rangle \tag{49}$$

with m the number of entries in D that are either 1 or 2.

3.3 Square GKP Gaussians Stabilizer

We now proceed to give an example of this group for the single-mode square GKP code and how it is related to the encoded Clifford group. We then construct the same group but for the combination of two square GKP codes before generalizing to N square GKP codes. For the single mode case, the generator matrix is $S = \sqrt{2} I_2$ and the symplectic Gram matrix is

$$A = 2\Omega_2, \tag{50}$$

which is already in the canonical form. To parametrize the Gaussian stabilizer group, we are looking for all symplectic matrices of the form

$$M = 2X\Omega_2 + I. \tag{51}$$

Following the analysis of the last section, we find three generators for the solutions of the Eq. (38). They are

$$X_{H^2} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \quad X_{Q^2} = \begin{bmatrix} 1 & 0 \\ 0 & 0 \end{bmatrix}, \quad X_{P^2} = \begin{bmatrix} 0 & 0 \\ 0 & 1 \end{bmatrix}, \tag{52}$$

with the corresponding symplectic matrices

$$M_{H^2} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix}, \quad M_{Q^2} = \begin{bmatrix} 1 & 2 \\ 0 & 1 \end{bmatrix}, \quad M_{P^2} = \begin{bmatrix} 1 & 0 \\ -2 & 1 \end{bmatrix}. \tag{53}$$

We indexed them as H^2, Q^2 and P^2 because of their logical action on the code, with Q denoting the phase gate in the logical X basis and P the phase gate in the logical Z

basis. The transformations in Eq. (53) implement the square of the logical operators $\overline{H}$, $\overline{\sqrt{X}}$ and $\overline{\sqrt{Z}}$, respectively. From each symplectic matrix of Eq. (53), the corresponding unitary operator that implement it are obtained by solving Eq. (12), which in this case yield $\overline{H}^2 = \exp\{-i\pi\hat{n}\}$, $\overline{Q}^2 = \exp\{-i\hat{p}^2\}$ and $\overline{P}^2 = \exp\{-i\hat{q}^2\}$, respectively. It now only remains to fix the translation part of the affine transformations. Because the unitary $\overline{Q}^2$ has the same action on the logical Pauli representatives as an $\overline{X}$ operation (which can be obtained by solving the equations of Sect. B), the affine transformation $\mathcal{L}_{M_{Q^2},\boldsymbol{\lambda}^*\in\mathcal{P}_x} = \overline{X}\,\overline{Q}^2$ is a generator of the gaussian stabilizer group. Similary, the transformation $\mathcal{L}_{M_{P^2},\boldsymbol{\lambda}^*\in\mathcal{P}_z} = \overline{Z}\,\overline{P}^2$ is also a generator. Finally, we find that $\overline{H}^2$ is itself a generator. The Gaussian stabilizer group of the single-mode square GKP code is

$$\mathcal{S}_{\square} = \left\langle \overline{X}^2, \overline{Z}^2, \overline{X}\,\overline{Q}^2, \overline{Z}\,\overline{P}^2, \overline{H}^2 \right\rangle. \tag{54}$$

We now move on to the combination of two square GKP codes. In this case, the Gaussian stabilizer group inherit two copies of the Gaussian stabilizer group described previously, one for each mode. There are still four new generators to find out of the ten in total. The symplectic Gram matrix now is $2\Omega_4$, and we are looking for symplectic matrices of the form

$$M = 2X\Omega_4 + I. \tag{55}$$

The four missing generators are related to the solutions $X = F_{j,k}$, $k \geq j$ that are not on the diagonal of any single block, given by $F_{1,2}, F_{2,3}, F_{3,4}$ and $F_{1,4}$. From those four solutions, we obtain the following four symplectic matrices

$$M_1 = \begin{bmatrix} 1 & 0 & 0 & 2 \\ 0 & 1 & 2 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \quad M_2 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ -2 & 1 & 0 & 0 \\ 0 & 0 & 1 & 2 \\ 0 & 0 & 0 & 1 \end{bmatrix}, \tag{56}$$

$$M_3 = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & -2 & 1 & 0 \\ -2 & 0 & 0 & 1 \end{bmatrix}, \quad \text{and} \quad M_4 = \begin{bmatrix} 1 & -2 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 2 & 1 \end{bmatrix}. \tag{57}$$

Their action on the Pauli operators $\overline{X}_j = \exp\left(-i\hat{p}_j\sqrt{\pi}\right)$ and $\overline{Z}_j = \exp\left(i\hat{q}_j\sqrt{\pi}\right)$, are given by the squares of the encoded controlled Pauli operators $\overline{C\left(X_1, X_2\right)}$, $\overline{C\left(Z_1, X_2\right)}$, $\overline{C\left(Z_1, Z_2\right)}$ and $\overline{C\left(X_1, Z_2\right)}$, in order from M_1 to M_4, respectively. The action of $\overline{C\left(A, B\right)}$ is to apply a $\overline{B}$ operation if the state is in a -1 eigenstate of $\overline{A}$. The square of those controlled encoded gates are implemented by the unitaries

$$\overline{C\left(X_j, X_k\right)}^2 = \exp\left(2i\hat{p}_j\hat{p}_k\right) \quad \overline{C\left(Z_j, X_k\right)}^2 = \exp\left(2i\hat{q}_j\hat{p}_k\right) \quad \overline{C\left(Z_j, Z_k\right)}^2 = \exp\left(-2i\hat{q}_j\hat{q}_k\right). \tag{58}$$

Any encoded generalized controlled Pauli gate is an order two logical gate, such that $\overline{C\left(A, B\right)}^2$ is in the Gaussian stabilizer group. As a result, we can write the full Gaussian stabilizer group as generated by the two sets of single-mode GKP Gaussian stabilizer generators and those four additional two-mode generators. Since the full group of symplectic operations is generated by two-body interactions, the Gaussian stabilizer group over N single-mode square GKP is found as

$$\mathcal{S}_{\square^{\otimes N}} = \left\langle \overline{X}_j^2, \overline{Z}_j^2, \overline{X}_j\,\overline{Q}_j^2, \overline{Z}_j\,\overline{P}_j^2, \overline{H}_j^2, \overline{C\left(X_j, X_k\right)}^2, \overline{C\left(Z_j, X_k\right)}^2, \overline{C\left(Z_j, Z_k\right)}^2 \right\rangle \tag{59}$$

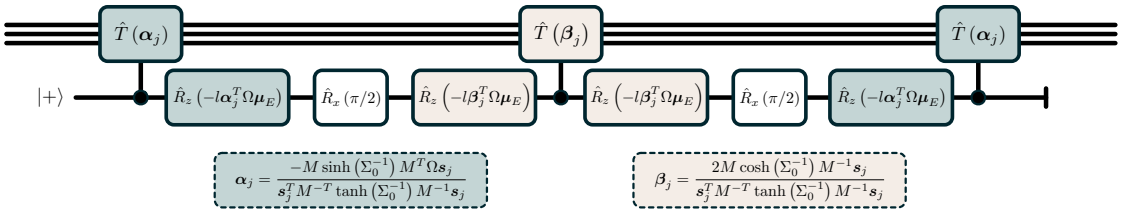


Figure 1: **Finite-energy GKP state stabilization.** Autonomous sBs stabilization circuit for a GKP state with envelope described by the covariance matrix $\Sigma_E = M\Sigma_0 M^T$ and position $\boldsymbol{\mu}_E$. The protocol uses engineered cooling of a GKP on N modes (represented by three wires) using a 2-level system as an ancilla (bottom wire) that is discarded at the end. The full stabilization is the result of cycling over all the stabilizer generators $\{\boldsymbol{s}_j\}$ of the code.

with $j,k \in \{1...N\}$ and $j \neq k$. Moreover, since this decomposition is based on the canonical form of the Gram matrix A, any combination of arbitrary one-mode GKP qubits leads to a Gaussian stabilizer group with the same structure and a very similar generator decomposition.

4 Noise protection of finite energy GKP codes

The construction of the Gaussian stabilizer group in Sects. 3.2 and 3.3 was tailored for GKP codes that span infinitely in phase space. In the following sections, we are interested in using this group as a resource to increase the performance of gate-based computation with a finite-energy GKP encoding. We show how a non-abelian stabilizer group on the infinite energy code becomes a logical isometry group on finite energy codewords. Based on arguments related to the noise channels, we propose optimization metrics to compare different implementations of Clifford gates.

4.1 Finite-energy GKP and Gaussian stabilizer

A logical state of the infinite energy GKP code defined by the generator matrix S is noted $|\Psi_S\rangle$. We construct a finite-energy logical state from $|\Psi_S\rangle$ by acting with a fock type damping operator E on it:

$$E\left(\Sigma_E,\mu_E\right)|\Psi_S\rangle \propto \exp\left(-\frac{1}{2}\left[\hat{\boldsymbol{\xi}}-\boldsymbol{\mu}_E\right]^T\Sigma_E^{-1}\left[\hat{\boldsymbol{\xi}}-\boldsymbol{\mu}_E\right]\right)|\Psi_S\rangle\,. \tag{60}$$

This damping operator can be understood as a Gaussian envelope described by a mean position $\boldsymbol{\mu}_E$ and a covariance matrix Σ_E. This state can be stabilized by a sequence of measurements of all normalized stabilizers

$$\hat{T}_{\Sigma_E,\mu_E}\left(\mathbf{s}_j\right) = E\left(\Sigma_E,\mu_E\right)\hat{T}\left(\mathbf{s}_j\right)E\left(\Sigma_E,\mu_E\right)^{-1}. \tag{61}$$

Another way of achieving stabilization of this state is by engineering an oscillator-bath interaction that cools down the oscillator's system toward the $+1$ eigenspace of each $\hat{T}_{\Sigma_E,\mu_E}\left(\mathbf{s}_j\right)$. One such protocol, called sBs [7, 14], was described in Ref. [18] for a multimode GKP code with an isometric envelope. The sBs protocol can be generalized to accommodate a more general envelope $E\left(\Sigma_E,\mu_E\right)$ presented here. The circuit that implements it, using only controlled displacements and qubit rotations, is presented in Fig. 1 and the full derivation is included in Sect. C. In an experimental setup, an initialization algorithm that prepares a logical state $|\Psi_{S,\Sigma,\mu}\rangle \propto E\left(\Sigma_E,\mu_E\right)|\Psi_S\rangle$ with an envelope that is centred and distributed equally in the quadratures of each mode is preferred, as we show

in the following sections. In other words, at the beginning of a computation, the ideal envelope is described by the parameters

$$\Sigma_0 = \varepsilon^{-1} \oplus \varepsilon^{-1} \qquad \boldsymbol{\mu}_E = \mathbf{0} \tag{62}$$

where ε is a diagonal matrix of dimension $N \times N$ with elements ε_j on its diagonal that describe the distribution of mode j. At initialization time, the envelope thus takes the form

$$E\left(\Sigma_0, \mathbf{0}\right) = \exp\left(-\sum_j \varepsilon_j \hat{n}_j\right). \tag{63}$$

The smaller the value of ε_j is, the closer the code is to its infinite energy counterpart. In the process of stabilizing the GKP codespace in an actual device, there exists an optimal value ε_j, limited by the amount of amplitude damping or other noises. For a unitary U that implements a general logical operator on the infinite energy codespace $U\left|\Psi_S\right\rangle = \left|\Phi_S\right\rangle$, the application of the same operator on a finite-energy state is equivalent to applying the gate perfectly on the logical state and modifying the envelope:

$$UE\left(\Sigma, \boldsymbol{\mu}\right)\left|\Psi_S\right\rangle = \left(UE\left(\Sigma, \boldsymbol{\mu}\right)U^\dagger\right)U\left|\Psi_S\right\rangle = E'\left(\Sigma, \boldsymbol{\mu}, U\right)\left|\Phi_S\right\rangle. \tag{64}$$

For a general unitary U, the new envelope E' is not Gaussian. However, in the case that U is a Gaussian unitary indexed by the symplectic matrix M followed by a translation in phase space by $\boldsymbol{\lambda}$, $U_{M,\boldsymbol{\lambda}}$, the new envelope E' remains Gaussian and is computed using Eq. (15):

$$U_{M,\boldsymbol{\lambda}} : E\left(\Sigma, \boldsymbol{\mu}\right) \to E\left(M\Sigma M^T, M\boldsymbol{\mu} + \boldsymbol{\lambda}\right). \tag{65}$$

As a result, no logical information is transferred to the envelope during a Clifford circuit computation since Clifford operations are implemented via Gaussian unitaries, . At each step, the logical information is contained only within the underlying infinite energy grid state $\left|\Psi_S\right\rangle$ or $\left|\Phi_S\right\rangle$. If the transformation $U_{M,\boldsymbol{\lambda}}$ describes a symmetry of the code, $U_{M,\boldsymbol{\lambda}}\left|\Psi_S\right\rangle = \left|\Psi_S\right\rangle$, the initial and final states both contain the same logical information, even if different envelopes describe the states. They are physically different, but logically equivalent. We've seen how to parametrize this set of logically equivalent states in Sect. 3. The set of Gaussian operations that are also isometries of a GKP code is the definition of the Gaussian stabilizer group of Eq. (49):

$$U_{M,\boldsymbol{\lambda}}E\left(\Sigma, \boldsymbol{\mu}\right)\left|\Psi_S\right\rangle = E\left(M\Sigma M^T, M\boldsymbol{\mu} + \boldsymbol{\lambda}\right)\left|\Psi_S\right\rangle \quad \forall U_{M,\boldsymbol{\lambda}} \in \mathcal{S}_G. \tag{66}$$

In the following, we show that the existence of an optimal state, and therefore, of an optimal circuit, is highly dependent on the noise model.

4.2 Noise model and state representative optimization

A quantum circuit defined by a sequence of gates $\{V_k\}$ and initial logical state $\left|\Psi_0\right\rangle$ can, equivalently, be fully caracterised by a sequence $\{\left|\Psi_k\right\rangle\}$ of states :

$$\ldots \xrightarrow{V_{k-1}} \left|\Psi_{k-1}\right\rangle \xrightarrow{V_k} \left|\Psi_k\right\rangle \xrightarrow{V_{k+1}} \ldots \tag{67}$$

This dual description of a circuit can be used as a tool to optimize how the sequence $\left\{\overline{V}_k\right\}$ is implemented, based on the properties of the state sequence. During a computation, some intermediate states might be more prone to errors, making it favourable to choose

an alternative circuit implementation and hence reducing the probability of having an error. In the context of quantum error correction, we aim to implement an encoded circuit $\left(\left|\overline{\Psi}_0\right\rangle, \left\{\overline{V}_k\right\}\right)$ onto a physical system :

$$\ldots \xrightarrow{\mathrm{Imp}\left(\overline{V}_{k-1}\right)} \left|\mathrm{Imp}\left(\overline{\Psi}_{k-1}\right)\right\rangle \xrightarrow{\mathrm{Imp}\left(\overline{V}_k\right)} \left|\mathrm{Imp}\left(\overline{\Psi}_k\right)\right\rangle \xrightarrow{\mathrm{Imp}\left(\overline{V}_{k+1}\right)} \ldots \tag{68}$$

An explicit physical implementation (Imp) is called a compilation strategy. In the typical quantum error correction code, the implementation of a codeword $\mathrm{Imp}\left(\overline{\Psi}_k\right)$ is unique, and so, only the gate compilation needs to be optimized. As presented in the last section, the case of GKP codes is different since different physical states with different envelopes describe the same logical information. This additional degree of freedom can be used to reduce the effect of noise on the logical information during the computation by choosing the logical states that are less affected by noise at each step. The optimization over all the different implementations can then be done in both the gates compilation picture or the states compilation picture. Since the best implementation is noise-dependent, we need to define a noise model and its effect on the GKP codewords as a function of the parameters of the envelope Σ_E and $\boldsymbol{\mu}_E$ in order to find an optimization criterion.

There are two main error channels for bosonic codes in harmonic modes. The more prominent one is bosonic loss $\mathcal{N}_L[\gamma]$ with its Kraus representation [27]:

$$\mathcal{N}_L[\gamma](\rho) = \sum_{k=0}^{\infty} \hat{L}_k \hat{\rho} \hat{L}_k^\dagger, \qquad \hat{L}_k = \sqrt{\frac{\gamma^k}{k!}} (1-\gamma)^{\hat{n}/2} \hat{a}^k. \tag{69}$$

Among other transformations, the main action of this channel on functions of phase-space is a radial contraction of every feature toward the origin by a factor of $\sqrt{\gamma}$, displacing by an amount $\sqrt{\gamma}\, r$ features at radius r. The dephasing channel is the second most impactful noise channel on bosonic modes, and is represented as [12]

$$\mathcal{N}_D[\gamma_\phi](\rho) = \frac{1}{\sqrt{2\pi\gamma_\phi}} \int_{\mathbb{R}} \mathrm{d}\phi\, e^{-\phi^2/2\gamma_\phi} e^{i\phi\hat{n}} \hat{\rho} e^{-i\phi\hat{n}}. \tag{70}$$

Its action is to smear phase-space functions in the angular direction, which can also be interpreted as a rotation in phase-space by a random angle sampled from a normal distribution. On average, this channel rotates the Wigner function of a state by an angle of amplitude $\sqrt{2/\pi}\,\gamma_\phi$. For a small dephasing rate γ_ϕ, this induces a displacement of $\sqrt{2/\pi}\,\gamma_\phi\, r$ for a feature positioned at radius r in phase-space. Because their action are in orthogonal directions, the pure loss and dephasing channel commute [27] and, therefore, the general noise channel can be described as a composition of both:

$$\mathcal{N}_{LD}[\gamma, \gamma_\phi] = \mathcal{N}_L[\gamma] \circ \mathcal{N}_D[\gamma_\phi] = \mathcal{N}_D[\gamma_\phi] \circ \mathcal{N}_L[\gamma]. \tag{71}$$

When both channels are active, the amplitude of the total translation of a feature at radius r is, on average,

$$\Delta\boldsymbol{\xi}(r) = r\sqrt{\gamma + \frac{2}{\pi}\gamma_\phi^2}. \tag{72}$$

Coming back to finite energy GKP codes, each feature of the Wigner function of a GKP state is a Gaussian (more details in Sect. 5.2). When both bosonic loss and dephasing are considered, each Gaussian that composes the full Wigner function will be translated on average by an amplitude of $\Delta\boldsymbol{\xi}(r)$ depending on its position at each application of the

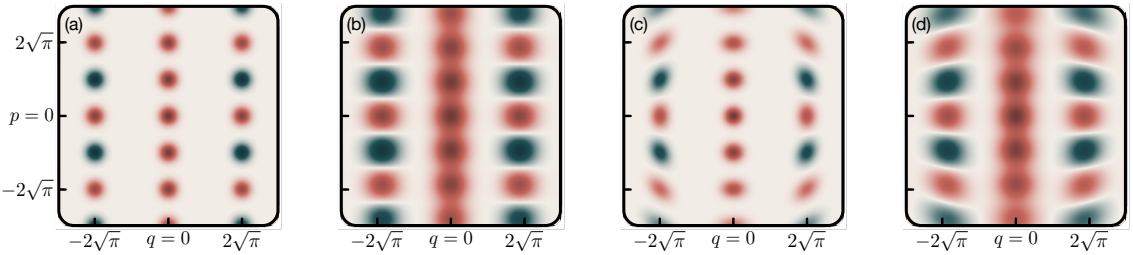


Figure 2: **Noise application on GKP states.** Effects of different error channels on the Wigner representation of the $|\overline{0}\rangle$ state of the square GKP code. (a) Unaffected state. (b) Amplitude damping with an average radial translation of $r\sqrt{\gamma}$. (c) Dephasing with an average absolute angular translation of $\sqrt{2/\pi}\, r\, \gamma_\phi$. (d) Combined amplitude damping and dephasing.

channel. The separate and combined effects of the two noise channels are shown in Fig. 2. When $\Delta\boldsymbol{\xi}(r)$ becomes of a length similar to the logical operators of the code, the logical error rate increases. It then becomes important to minimize the radial position of the full finite energy GKP state, which in turn suggests that the position of the envelope $\boldsymbol{\mu}_E$ should be minimized at every step of a computation. Accordingly, we compare different implementations with the metric

$$\mathrm{d}_\mu^2(\boldsymbol{\mu}_E) = \boldsymbol{\mu}_E^T \boldsymbol{\mu}_E. \tag{73}$$

More generally, the argument toward reducing the number of excitations in the harmonic oscillator can be used more broadly for most error channels on harmonic oscillators (e.g. bosonic loss, dephasing, spurious Kerr non-linearity), as their effects increase with excitation number. We thus compute the average number of excitations in a GKP state described by generator matrix S and envelope $(\Sigma_E, \boldsymbol{\mu}_E) = \left(M_E \Sigma_0 M_E^T, \boldsymbol{\mu}_E\right)$, where Σ_0 contains the symplectic eigenvalues on the diagonal after a Williamson decomposition [28]. This result takes the form of a ponderated sum of the excitation contribution over all the Gaussian peaks in phase-space:

$$\left\langle \hat{\boldsymbol{\xi}}^T \hat{\boldsymbol{\xi}} \right\rangle = \sum_{\boldsymbol{\chi}_E \in \Lambda_E} f(\boldsymbol{\chi}_E) \left(\boldsymbol{\chi}_E^T \boldsymbol{\chi}_E + \mathrm{Tr}\left[M_E \tanh\left(\Sigma_0^{-1}\right) M_E^T \right] \right), \tag{74}$$

where $f(\boldsymbol{\chi}_E)$ is the ponderation coefficient (see Sect. F for the full derivation). Λ_E is the set of peak positions altered by the envelope. For each $\boldsymbol{\lambda} \in \Lambda$, we have an associated $\boldsymbol{\chi}_E \in \Lambda_E$ such that

$$\boldsymbol{\chi}_E = M_E \,\mathrm{sech}\left(\Sigma_0^{-1}\right) M_E^{-1} \left[\boldsymbol{\lambda} - \boldsymbol{\mu}_E\right] + \boldsymbol{\mu}_E. \tag{75}$$

The result of Eq. (74) reinforces the criterion of Eq. (73) because the envelope position $\boldsymbol{\mu}_E$ that minimizes the contribution

$$\sum_{\boldsymbol{\chi}_E} f(\boldsymbol{\chi}_E)\, \boldsymbol{\chi}_E^T \boldsymbol{\chi}_E, \tag{76}$$

also minimizes $\mathrm{d}_\mu^2(\boldsymbol{\mu}_E)$. The second contribution to the excitation number is connected to the covariance matrix of the envelope

$$\sum_{\boldsymbol{\chi}_E} f(\boldsymbol{\chi}_E)\, \mathrm{Tr}\left[M_E \tanh\left(\Sigma_0^{-1}\right) M_E^T \right]. \tag{77}$$

The Williamson decomposition splits the covariance matrix Σ_E into the product of a symplectic matrix M_E and a positive definite diagonal matrix Σ_0: $\Sigma_E = M_E \Sigma_0 M_E^T$. The trace contribution in Eq. (77) is invariant under orthogonal matrix congruence, leaving only the squeezing part of M_E contributing non-trivially to $\left\langle \hat{\boldsymbol{\xi}}^T \hat{\boldsymbol{\xi}} \right\rangle$. The contribution increases with the amount of squeezing in M_E and is minimized when $M_E \in \mathrm{O}_{2N}(\mathbb{R})$ [29]. Altough our general goal is to reduce the excitation number, it is not the only goal of this optimisation. As discussed in Sect. 4.1, the optimal eigenvalues of Σ_0 depend on the quality of the oscillators and the frequency of error correction. Consequently, reducing the excitation contribution of one mode below its optimal value (at the expense of another mode) means that we are not using the full capacity of the encoding and that errors will lead to quick decoherence. In order to consider squeezing and anti-squeezing on the same footing, we introduce a metric on covariance matrices

$$\mathrm{d}_\Sigma^2\left(\Sigma_E, \Sigma_0\right) = \mathrm{Tr}\left[\ln^2\left(\sqrt{\Sigma_0^{-1}}\,\Sigma_E\,\sqrt{\Sigma_0^{-1}}\right)\right], \tag{78}$$

with Σ_E the covariance matrix of the envelope and Σ_0 the optimal covariance matrix from the Williamson decomposition [30]. If we consider that all modes are similar, the initial parameters of the envelope are the ones of Eq. (62), where $\Sigma_0 = \varepsilon^{-1} I$. Under this similarity assumption, minimizing the cost function

$$\mathrm{Tr}\left[M_E M_E^T\right] \geq 1, \tag{79}$$

would minimize the excitation number of Eq. (77), while minimizing the cost function

$$\mathrm{d}_\Sigma^2\left(M_E\left(\varepsilon^{-1} I\right) M_E^T, \varepsilon^{-1} I\right) = \mathrm{Tr}\left[\ln^2\left(M_E M_E^T\right)\right] \geq 0, \tag{80}$$

would minimize the distance between covariance matrices of Eq. (78). Thus, in the case where $\Sigma_0 = \varepsilon^{-1} I$, both metrics from Eq. (79) and Eq. (80) lead to the same optimization, which only depends on the squeezing contained in M_E.

Therefore, to reduce the impact of noise channels on GKP states, we aim to reduce the amount of displacement and squeezing that are created by the chosen physical implementation. More general schemes could be designed for multimode GKP by playing with the trade-off of sending errors to one mode to gain fidelity in another. Using such a sacrificial mode could be an interesting optimization technique, but it is outside of the scope of this work. Combining both criteria of Eqs. (73) and (78), we can now compare different physical implementations of a logical Clifford gate, and we are ready to give an explicit description of our compiler algorithm.

5 Gaussian stabilizer compiler

5.1 Gaussian stabilizer compiler

As described in the last section, a compiler algorithm selects an implementation given a logical circuit. When considering a physical representative U_0 for a logical operation on a code (we choose it closest to the identity for simplicity), $U_0\mathcal{S}$ is the group of all implementations of the same logical operation: $U_0 S \left|\overline{\Psi}\right\rangle = U_0 \left|\overline{\Psi}\right\rangle \; \forall\, S \in \mathcal{S}$, where $\mathcal{S}$ is the stabilizer group that can be implemented. Here, we choose $\mathcal{S}$ to be the gaussian stabilizer group. Any operation logically equivalent to U_0 can be written as

$$\mathrm{Imp}\left(\overline{U}\right) = U_0\, g_n, \tag{81}$$

where g_n correspond to the application of a sequence of at most n generators of $\mathcal{S}$ (or their inverse). We can look at g_n in Eq. (81) as a walk of length at most n on the Cayley graph of the Gaussian stabilizer group $\Gamma(\mathcal{S}_G)$, with generating set as in Eq. (49). A compilation strategy is then translated to a choice of walk on $\Gamma(\mathcal{S}_G)$. A simple compiling strategy is to choose a fixed implementation for a given operation, which we call below the constant compiler. Another compiling strategy, which we refer to as the random walk compiler, is to choose an implementation at random among the closest ones to the identity, by which we mean an implementation of the form of Eq. (81) with $n = 1$. This last strategy induces a random walk on the Cayley graph $\Gamma(\mathcal{S}_G)$ interleaved with the basic physical representatives $\{U_0\}$. As discussed in [31], such a random walk induce a Gaussian stabilizer group twirling, which will result in an approximate logical depolarizing channel. Both the constant and random walk compilers are suboptimal as the amount of squeezing and total displacement increases with the number of operations, as seen in Fig. 3. This remains true if we modify the random walk compiler to only sample from a smaller set of generators.

To achieve an optimized circuit implementation, the metrics in Eqs. (73) and (78) should be minimized, among all possible walks on $\Gamma(\mathcal{S}_G)$. However, because the Gaussian stabilizer is an infinite group, the metrics cannot be computed for all equivalent operations, and a cutoff needs to be defined. Because operations in $\mathcal{S}_G$ that are far from the identity generate more squeezing and displacement, they are less likely to be useful in the optimization. Therefore, we only consider operations made of short sequences of generators of the stabilizer group. With $\mathcal{S}_G$ finitely generated, we can compute both metrics in Eqs. (73) and (78) of all different walks on Γ up to a finite length n and define the optimal encoded Clifford implementation as the path that minimizes the metrics at each step. This process is described in algorithm 1. The algorithm takes as input the target operation U_0 and the parameters of the initial envelope (Σ_i and $\boldsymbol{\mu}_i$). After comparing the different paths g_n on Γ of length n, it returns the optimal operation U_{opt} and the final parameters of the envelope ($\Sigma_{f,\text{opt}}$ and $\boldsymbol{\mu}_{f,\text{opt}}$). We derived an example of the application of algorithm 1 in Sect. G. Below, we set the maximum length n to 2. In general, pass a certain threshold, increasing the walk length n will not necessarily increase the performance of the compiler. A lot of the times, only short walks on Γ are used and increasing n will not give better implementations of a gate. In algorithm 1, we choose to optimize the squeezing parameter first and then the displacement since squeezing is typically a larger limiter of the performance of error correction protocols. Below, we compare the performance of the Gaussian stabilizer compiler with the constant and random walk compilers and show that it outperforms both of them.

5.2 Performance of the GS compiler

A method to compare different implementations of gates at the encoded level is logical randomized benchmarking (LRB) [32]. Randomized benchmarking (RB) methods are efficient ways to characterize and compare gate set implementations, in a way that the effects of state preparation and measurement errors can be factored out [33, 34]. When applying sequences of gates from a group G to a system, the final state is affected compoundly by the decoherence that happens during each gate. The result of the following measurement is then a function of the error channel and the length of the sequence. When the gate implementations are multiplicity-free representations of the target group G and the error model is Markovian and time-independent, the survival probability is well described by a sum of decaying exponentials in the number of operations [35]. However, in the setup of LRB, the condition of multiplicity-free representations is often unfulfilled and non-exponential

Algorithm 1 Gaussian stabilizer compiler

Input: $U_0 = \hat{T}(\boldsymbol{\lambda}_0) U_{M_0}$, Σ_i, $\boldsymbol{\mu}_i$
  $\Sigma_{f,\text{opt}} \leftarrow M_0 \Sigma_i M_0^T$
  $\boldsymbol{\mu}_{f,\text{opt}} \leftarrow M_0 \boldsymbol{\mu}_i + \boldsymbol{\lambda}_0$
  $g_{\text{opt}} \leftarrow I$
  for all $g_n \in \{g_n\}$ **do**
    Compute its symplectic representation $U_0\, g_n = \hat{T}(\boldsymbol{\lambda}) U_M$
    if $d_\Sigma^2\left(M\Sigma_i M^T, \Sigma_0\right) < d_\Sigma^2\left(\Sigma_{f,\text{opt}}, \Sigma_0\right)$ **then**
      $\Sigma_{f,\text{opt}} \leftarrow M\Sigma_i M^T$
      $\boldsymbol{\mu}_{f,\text{opt}} \leftarrow M\boldsymbol{\mu}_i + \boldsymbol{\lambda}$
      $g_{\text{opt}} \leftarrow g_n$
    else if $d_\Sigma^2\left(M\Sigma_i M^T, \Sigma_0\right) = d_\Sigma^2\left(\Sigma_{f,\text{opt}}, \Sigma_0\right)$ **and** $d_\mu^2\left(M\boldsymbol{\mu}_i + \boldsymbol{\lambda}\right) < d_\mu^2\left(\boldsymbol{\mu}_{f,opt}\right)$ **then**
      $\Sigma_{f,\text{opt}} \leftarrow M\Sigma_i M^T$
      $\boldsymbol{\mu}_{f,\text{opt}} \leftarrow M\boldsymbol{\mu}_i + \boldsymbol{\lambda}$
      $g_{\text{opt}} \leftarrow g_n$
    end if
  end for
  return $U_0\, g_{\text{opt}}$, $\Sigma_{f,\text{opt}}$, $\boldsymbol{\mu}_{f,\text{opt}}$

behaviour can appear [36]. In some cases, the survival probability remains a monotonically non-increasing function of the sequence length and can still be used as a measure of performance, but without links to the average gate fidelity. To obtain a measure of performance of the GKP Gaussian compiler, we perform logical randomized benchmarking by sampling uniformly the encoded Clifford group. As encoded Clifford operations are represented by Gaussian operations that are not multiplicity-free, we do not expect to obtain pure exponential decay (at least not for small sequence length [35]). Furthermore, the Gaussian stabilizer compiler presented in algorithm 1 does not respect the Markovian condition as it selects gate implementations based on the previous ones in the circuit. Although Markovian noise is a necessary condition for classic RB methods, correlations between the gates, and so between noise applications, are exactly how we gain in performance against decoherence. By using LRB, we do not aim specifically to obtain an error rate for our gate set, but instead aim to compare different compilation strategies for the encoded circuit. An implementation strategy is better than another if, for all circuit lengths, the probability of getting the correct result is greater. Alternatively, it means that longer circuits can be implemented with the same success probability.

With the goal of doing an LRB analysis, we simulate GKP states by following the evolution of their Wigner function in phase-space with the tools developed in Ref. [37]. In this setup, the Wigner function of a finite energy GKP grid state is described as a sum of Gaussian functions in phase space:

$$W_{\text{GKP}}(\xi) = \sum_{m \in \mathcal{M}} c_m G_{\Sigma_m, \boldsymbol{\mu}_m}(\xi), \tag{82}$$

with $G_{\Sigma_m, \boldsymbol{\mu}_m}(\xi)$ defined in Eq. (21) and the set $\mathcal{M}$ indexing the Gaussian functions. The normalization condition reads $\sum_{m \in \mathcal{M}} c_m = 1$ for this representation. We have also generalized the results of [37] to obtain a representation of the form of Eq. (82) for any multimode GKP grid state with potentially squeezed and displaced envelope in Sects. D and E. For simplicity, we consider that the system is initialized in a way that all modes are

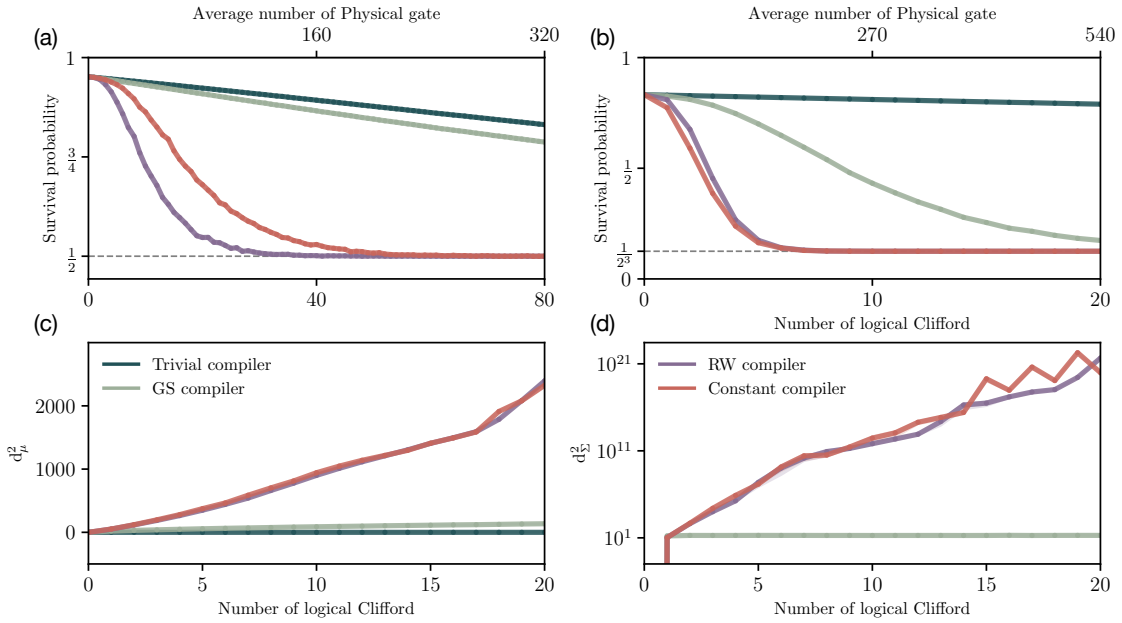


Figure 3: **Logical randomized benchmarking results.** Survival probability for the constant compiler (red), random walk (purple) and Gaussian stabilizer compiler (light green) on square GKP codes with varying sequence length. We have also included the average survival probability curve when no logical operation is applied (blue-green) as an indicator. The average survival probability curves are shown for (a) 1 and (b) 3 logical qubits with loss strength $\gamma = 0.999$ and $\gamma = 0.9999$, respectively. The dashed line indicates the survival probability for a fully mixed logical state. For the simulations included in the decay curve (b), we compute (c) the maximum position metric of Eq. (73) and (d) the maximum squeezing metric of Eq. (78) reached as a function of logical circuit length.

decoupled from each other $\Sigma_E = \varepsilon^{-1} \oplus \varepsilon^{-1}$ and $\boldsymbol{\mu}_E = \mathbf{0}$ as in Eq. (63). Those considerations lead to an initial state with with parameters

$$\Sigma_m = \frac{1}{4\pi} \coth\left(\Sigma_E^{-1}\right), \qquad \boldsymbol{\mu}_m = \operatorname{sech}\left(\Sigma_E^{-1}\right) \boldsymbol{\mu}_{S,m}, \tag{83}$$

where $\boldsymbol{\mu}_{S,m}$ is the position of each Dirac delta function in the infinite energy description of the state with stabilizer generator matrix S derived in Sect. D. This description of the state as a sum of Gaussian functions in phase space has many advantages. The most useful being that we can follow its evolution under all Gaussian operations and Gaussian noise channels by updating the parameters in Eq. (83). Under any Gaussian operation, the parameters of the envelope are updated like in Eq. (65). This rule can also be used for the dephasing channel of Eq. (70) if we use a Monte Carlo method to model this channel as a random rotation of each mode. For bosonic loss, the parameters are updated with

$$\Sigma_m \to \gamma \Sigma_m + \frac{1-\gamma}{4\pi} I, \qquad \boldsymbol{\mu}_m \to \sqrt{\gamma} \boldsymbol{\mu}_m, \tag{84}$$

when the loss parameter is γ. In summary, to apply a given Gaussian circuit, we find the representation of the circuit in terms of symplectic operation and apply those symplectic matrices directly onto the saved covariance matrices and peak positions.

At the end of the computation, we wish to apply a decoding gadget to know if a logical error occurs. To achieve this decoding, we compute the characteristic function of a given

Wigner function using the Fourier transform. This can be done analytically for any Wigner functions of the form of Eq. (82) due to the linearity of the transform. The result is again a sum of Gaussian functions that can be evaluated at any point. To complete the connection with GKP codes, we present another form of the characteristic function

$$\chi(\mathbf{v}) = \mathrm{Tr}\left[\hat{T}(\mathbf{v})\,\rho\right] = \left\langle \hat{T}(\mathbf{v}) \right\rangle. \tag{85}$$

This form is particularly useful for us because we can obtain the average value of a displacement operators, which include stabilizers of ideal GKP codes. Because we have access to the whole characteristic function analytically, we can efficiently compute an approximation of the projector onto the $+1$ eigenvalue subspace of the logical $\overline{Z}$ operator

$$\left\langle \Pi_{\overline{Z}=+1} \right\rangle = \sum_{\mathbf{s}\in\Lambda\cup\Lambda_z} \left\langle \hat{T}(\mathbf{s}) \right\rangle = \sum_{\mathbf{s}\in\Lambda\cup\Lambda_z} \chi(\mathbf{s}), \tag{86}$$

by truncating the sum. The value of Eq. (86) is then interpreted as the probability of having the correct result after the computation (based on an ideal characteristic function decoder), which we also defined as a survival probability. The definition of the projector can be generalized for the operator $\overline{Z}^{\otimes N}$ when multiple qubits are used, again described by a sum of the average value of displacement operators. Hence, we can perform GKP state preparation, state evolution under logical Clifford circuits and survival probability measurements efficiently.

To analyze the advantages of using the Gaussian stabilizer compiler over other compiler algorithms, we performed LRB simulations for 1,2 and 3 logical qubits encoded in a square GKP code. The LRB simulations consist of generating a large number of logical Clifford circuits varying in length and simulating the implementation of those circuits, obtained from a given compiler algorithm. By averaging the survival probability obtained over all random circuits of a given length, we obtain the average performance of a compiler at this length. The results for 1 and 3 qubits are showcased in Fig. 3 in panels a and b, respectively. As expected, the decay curves do not exhibit the regular exponential behaviour for short sequences, but rather a Gaussian behaviour. The Gaussian behaviour can be explained by the shape of the characteristic function of the finite-energy GKP states. This function, like its Wigner function, is composed of a sum of many Gaussian features. The effect of the noise channel is to shift the peaks of the Gaussian away from the measured points, leading to a reduction in the survival probability. For short sequences of gates, the resulting shape of the decay then corresponds to the shape of the characteristic function near the measured point. For longer logical circuits, logical Pauli errors become more important, and we recover the exponential decay behaviour. As discussed at the beginning of the section, even if the decay does not follow an exponential behaviour, we still obtain a lifetime ordering of the different compilation strategies.

Figure 3 demonstrates that the constant and random walk compilers have similar performance, while the Gaussian stabilizer compiler performs better. In this situation, we expected the constant and random compiler to perform similarly. Since the logical Hadamard gate is implemented via a $\pi/2$ phase-space rotation, it leads to an inversion of coordinates in phase-space, which means that the randomness from the circuit sampling step can also be understood as a random walk, leading to similar performance as the random walk compiler. We observe that in all cases, the Gaussian stabilizer compiler performs better than the other two compilers. We also observe that, as expected, both the displacement and

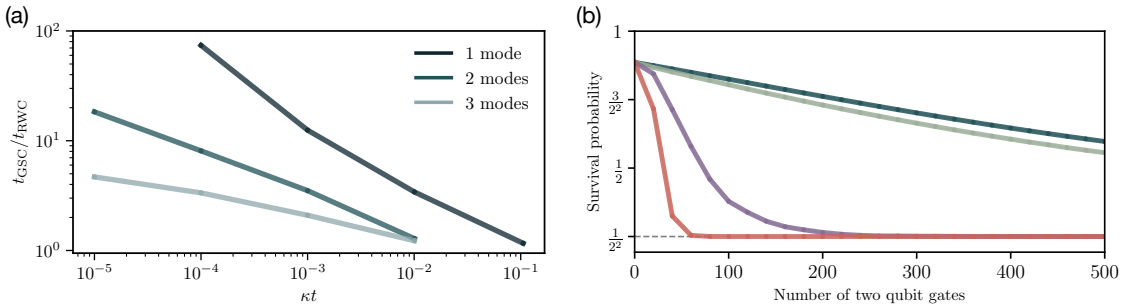


Figure 4: **Performance of the GS compiler with varying randomized benchmarking conditions.** (a) Lifetime amelioration factor between the Gaussian stabilizer compiler and random walk compiler as a function of loss rate (κt). (b) Results of two-qubit gates randomized benchmarking with the group $\langle \overline{CX}, \overline{CZ} \rangle$ and loss parameter $\gamma = 0.999$. The figure shows the survival probability for the constant compiler (red), random walk (purple) and Gaussian stabilizer compiler (light green) on two logical square GKP codes with varying sequence length.

squeezing metrics remain low during computations with the GS compiler. To extract a computation lifetime, we fit a Gaussian function on the survival probability $P(x)$

$$P(x) = A\mathrm{e}^{-\left(ax^2+bx\right)} + B, \tag{87}$$

where x is the number of logical gates applied. For all decay curves, we define the computation lifetime t as the number of logical operations that we need to apply to reduce the survival probability by a factor of $1/\mathrm{e}$:

$$P(t) = A\mathrm{e}^{-1} + B. \tag{88}$$

This definition of the lifetime allows us to compare both Gaussian and exponential fits on an equal footing. The amelioration factor, defined as the ratio of the GS compiler lifetime over the random walk compiler lifetime, is dependent on bosonic loss strength. As shown in Fig. 4 (a), we obtain that, at low loss rate, the GS compiler performs even better. When the loss rate is high, no compiler can help preserve the logical information and the computation fails with high probability in all cases. Another result from this simulation is that the amelioration factor seems to decrease with the number of modes. We explain this behaviour by the fact that, as we increase the number of modes and therefore the number of logical Clifford gates that we cannot simplify, the state spends a lot of time in moderatly squeezed state before we can actually apply any simplification. And so, as we increase the number of modes, we need to reduce the noise strength to achieve the same performance.

Finally, we performed similar LRB simulations, but now replacing the full Clifford group with two-qubit gates only. This scenario is relevant in a situation where a quantum compution is performed with magic state injection and Clifford gates. In this case, all single-qubit Clifford gates can be performed in a Pauli and Clifford frame, leaving only two-qubit gates to be implemented physically. As a result, only squeezing would play a role here, since all displacement operations would be removed. The result of a logical randomized benchmarking simulation with only two-qubit gates on two logical qubits is included in Fig. 4 (b). We find that the Gaussian stabilizer compiler still performs well in this setup compared to other compiling strategies.

As a final note, we remark that interleaving quantum error correction gadgets in GKP logical circuits will have an effect on the envelope and might affect the results presented

here. For example, error correction via state teleportation resets the envelope of the state, the sBs protocol stabilizes a centered envelope, and displacement twirling of the envelope reduce the logical space leakage [31, 38]. However, we believe that our Gaussian stabilizer compiler will remain useful with hybrid approaches depending on the actual implementation of error correction. We leave the analysis of combining quantum error correction schemes with the GS compiler to future work.

6 Conclusion

In this work, we have investigated how to leverage more symmetries of GKP codes to reduce the impact of noise at the logical level. We first presented how we can obtain more symmetries of a code by relaxing the restriction to abelian stabilizer groups. This new perspective enables us to find all symmetry operations related to an implementation of a logical operator. Restricting this set to Gaussian operations, we showed that it is possible to describe the Gaussian symmetry group with only a few generators for any lattice describing a GKP code. This new set of non-commuting stabilizers can be used similarly to the commuting ones, such as for state stabilization, quantum error correction and decoding. In future work, we aim to characterize more thoroughly this connection between non-abelian stabilizer groups and potential applications in quantum error correction.

Exploring the passage from infinite to finite energy GKP state, we found that infinite energy stabilizers become logical stabilizers, but not physical stabilizers. As a first application of GKP non-abelian stabilizer groups, we presented a compiler algorithm that uses those logical symmetries on the finite-energy version of GKP codes to reduce the impact of loss and dephasing. This compiler works by selecting an implementation of a logical operation that reduces as much as possible the amount of total displacement and squeezing. Doing so reduces the mean excitation number in the oscillator and, by consequence, reduces the impact of loss and dephasing. We conjecture that it should also increase the protection against other noise channels, such as Kerr non-linearity. We leave that analysis for future work. It would also be interesting to explore how non-Gaussian stabilizer operations can fit within this formalism and increase their fidelity.

We showed that we obtain an amelioration in computation lifetime while using the Gaussian stabilizer compiler, and that this amelioration increases with the reduction of loss strength. While the GS compiler only uses the GKP symmetries to recenter and unsqueezed the intermediate states of a computation, it seems probable that dynamical decoupling techniques at the compilation level could help symmetrize the effects of noise and further reduce its impact.

It is conceivable that GKP codes will be used as a base layer for a fault-tolerant quantum computer, with a top-layer qubit code [39]. To reach this goal, it is primordial to have good native operations on the base layer combined with a performant compilation strategy. In this paper, we have introduced one such strategy.

Acknowledgments

This research was undertaken thanks in part to funding from the Canada First Research Excellence Fund. This research was also funded by the National Science and Engineering Council with a Canada Graduate Research Scholarship – Doctoral, by the Army Research

Office under grant #W911NF2310045 and by a Université de Sherbrooke Research Excellence Scholarships.

Code Availability

The source code for the Gaussian stabilizer compiler and the simulations of logical randomized benchmarking is available at `https://github.com/Jopelxl/Gaussian-Stabilizer-Compiler.git`.

References

[1] V. V. Sivak, A. Eickbusch, B. Royer, S. Singh, I. Tsioutsios, S. Ganjam, A. Miano, B. L. Brock, A. Z. Ding, L. Frunzio, S. M. Girvin, R. J. Schoelkopf, and M. H. Devoret. "Real-time quantum error correction beyond break-even". Nature **616**, 50–55 (2023). arXiv:2211.09116.

[2] Dany Lachance-Quirion, Marc-Antoine Lemonde, Jean Olivier Simoneau, Lucas St-Jean, Pascal Lemieux, Sara Turcotte, Wyatt Wright, Amélie Lacroix, Joëlle Fréchette-Viens, Ross Shillito, Florian Hopfmueller, Maxime Tremblay, Nicholas E. Frattini, Julien Camirand Lemyre, and Philippe St-Jean. "Autonomous Quantum Error Correction of Gottesman-Kitaev-Preskill States". Physical Review Letters **132**, 150607 (2024).

[3] Benjamin L. Brock, Shraddha Singh, Alec Eickbusch, Volodymyr V. Sivak, Andy Z. Ding, Luigi Frunzio, Steven M. Girvin, and Michel H. Devoret. "Quantum error correction of qudits beyond break-even". Nature **641**, 612–618 (2025).

[4] Nissim Ofek, Andrei Petrenko, Reinier Heeres, Philip Reinhold, Zaki Leghtas, Brian Vlastakis, Yehan Liu, Luigi Frunzio, S. M. Girvin, L. Jiang, Mazyar Mirrahimi, M. H. Devoret, and R. J. Schoelkopf. "Extending the lifetime of a quantum bit with error correction in superconducting circuits". Nature **536**, 441–445 (2016).

[5] Shunya Konno, Warit Asavanant, Fumiya Hanamura, Hironari Nagayoshi, Kosuke Fukui, Atsushi Sakaguchi, Ryuhoh Ide, Fumihiro China, Masahiro Yabuno, Shigehito Miki, Hirotaka Terai, Kan Takase, Mamoru Endo, Petr Marek, Radim Filip, Peter van Loock, and Akira Furusawa. "Logical states for fault-tolerant quantum computation with propagating light". Science **383**, 289–293 (2024).

[6] M. V. Larsen, J. E. Bourassa, S. Kocsis, J. F. Tasker, R. S. Chadwick, C. González-Arciniegas, J. Hastrup, C. E. Lopetegui-González, F. M. Miatto, A. Motamedi, R. Noro, G. Roeland, R. Baby, H. Chen, P. Contu, I. Di Luch, C. Drago, M. Giesbrecht, T. Grainge, I. Krasnokutska, M. Menotti, B. Morrison, C. Puviraj, K. Rezaei Shad, B. Hussain, J. McMahon, J. E. Ortmann, M. J. Collins, C. Ma, D. S. Phillips, M. Seymour, Q. Y. Tang, B. Yang, Z. Vernon, R. N. Alexander, and D. H. Mahler. "Integrated photonic source of Gottesman–Kitaev–Preskill qubits". Nature **642**, 587–591 (2025).

[7] Brennan de Neeve, Thanh-Long Nguyen, Tanja Behrle, and Jonathan P. Home. "Error correction of a logical grid state qubit by dissipative pumping". Nature Physics **18**, 296–300 (2022).

[8] V. G. Matsos, C. H. Valahu, T. Navickas, A. D. Rao, M. J. Millican, X. C. Kolesnikow, M. J. Biercuk, and T. R. Tan. "Robust and Deterministic Preparation of Bosonic Logical States in a Trapped Ion". Physical Review Letters **133**, 050602 (2024).

[9] V. G. Matsos, C. H. Valahu, M. J. Millican, T. Navickas, X. C. Kolesnikow, M. J. Biercuk, and T. R. Tan. "Universal quantum gate set for Gottesman–Kitaev–Preskill logical qubits". Nature PhysicsPages 1–6 (2025).

[10] Daniel Gottesman, Alexei Kitaev, and John Preskill. "Encoding a qubit in an oscillator". Physical Review A **64**, 012310 (2001). arXiv:quant-ph/0008040.

[11] Kyungjoo Noh, Victor V. Albert, and Liang Jiang. "Quantum Capacity Bounds of Gaussian Thermal Loss Channels and Achievable Rates With Gottesman-Kitaev-Preskill Codes". IEEE Transactions on Information Theory **65**, 2563–2582 (2019).

[12] Peter Leviant, Qian Xu, Liang Jiang, and Serge Rosenblum. "Quantum capacity and codes for the bosonic loss-dephasing channel". Quantum **6**, 821 (2022). arXiv:2205.00341.

[13] Sergey Bravyi and Alexei Kitaev. "Universal quantum computation with ideal Clifford gates and noisy ancillas". Physical Review A **71**, 022316 (2005).

[14] Baptiste Royer, Shraddha Singh, and S. M. Girvin. "Stabilization of Finite-Energy Gottesman-Kitaev-Preskill States". Physical Review Letters **125**, 260509 (2020). arXiv:2009.07941.

[15] Ilan Tzitrin, J. Eli Bourassa, Nicolas C. Menicucci, and Krishna Kumar Sabapathy. "Progress towards practical qubit computation using approximate Gottesman-Kitaev-Preskill codes". Physical Review A **101**, 032315 (2020).

[16] Jim Harrington and John Preskill. "Achievable rates for the Gaussian quantum channel". Physical Review A **64**, 062301 (2001).

[17] Jonathan Conrad, Jens Eisert, and Francesco Arzani. "Gottesman-Kitaev-Preskill codes: A lattice perspective". Quantum **6**, 648 (2022). arXiv:2109.14645.

[18] Baptiste Royer, Shraddha Singh, and Steven M. Girvin. "Encoding qubits in multimode grid states". PRX Quantum **3**, 010335 (2022). arXiv:2201.12337.

[19] Daniel Gottesman. "Stabilizer Codes and Quantum Error Correction, Caltech Ph.D. thesis" (1997). arXiv:quant-ph/9705052.

[20] Xiaotong Ni, Oliver Buerschaper, and Maarten Van den Nest. "A non-commuting stabilizer formalism". Journal of Mathematical Physics **56**, 052201 (2015).

[21] Mark A. Webster, Benjamin J. Brown, and Stephen D. Bartlett. "The XP Stabiliser Formalism: A Generalisation of the Pauli Stabiliser Formalism with Arbitrary Phases". Quantum **6**, 815 (2022).

[22] Jérémie Boudreault, Ross Shillito, Jean-Baptiste Bertrand, and Baptiste Royer. "Using a Kerr interaction for GKP magic state preparation" (2025). arXiv:2507.09684.

[23] Narayanan Rengaswamy, Robert Calderbank, Swanand Kadhe, and Henry D. Pfister. "Logical Clifford Synthesis for Stabilizer Codes". IEEE Transactions on Quantum Engineering **1**, 1–17 (2020).

[24] Eric J. Kuehnke, Kyano Levi, Joschka Roffe, Jens Eisert, and Daniel Miller. "Hardware-tailored logical Clifford circuits for stabilizer codes" (2025). arXiv:2505.20261.

[25] Jonathan Conrad, Ansgar G. Burchards, and Steven T. Flammia. "Lattices, Gates, and Curves: GKP codes as a Rosetta stone" (2024). arXiv:2407.03270.

[26] Mao Lin, Christopher Chamberland, and Kyungjoo Noh. "Closest Lattice Point Decoding for Multimode Gottesman-Kitaev-Preskill Codes". PRX Quantum **4**, 040334 (2023).

[27] Christian Weedbrook, Stefano Pirandola, Raúl García-Patrón, Nicolas J. Cerf, Timothy C. Ralph, Jeffrey H. Shapiro, and Seth Lloyd. "Gaussian quantum information". Reviews of Modern Physics **84**, 621–669 (2012).

[28] Martin Houde, Will McCutcheon, and Nicolás Quesada. "Matrix decompositions in quantum optics: Takagi/Autonne, Bloch–Messiah/Euler, Iwasawa, and Williamson". Canadian Journal of Physics **102**, 497–507 (2024).

[29] Rajendra Bhatia and Tanvi Jain. "On symplectic eigenvalues of positive definite matrices". Journal of Mathematical Physics **56**, 112201 (2015). arXiv:1803.04647.

[30] Wolfgang Förstner and Boudewijn Moonen. "A Metric for Covariance Matrices". In Erik W. Grafarend, Friedrich W. Krumm, and Volker S. Schwarze, editors, Geodesy-The Challenge of the 3rd Millennium. Pages 299–309. Springer Berlin Heidelberg, Berlin, Heidelberg (2003).

[31] Jonathan Conrad, Jens Eisert, and Steven T. Flammia. "Chasing shadows with Gottesman-Kitaev-Preskill codes". Quantum **10**, 1973 (2026).

[32] Joshua Combes, Christopher Granade, Christopher Ferrie, and Steven T. Flammia. "Logical Randomized Benchmarking" (2017). arXiv:1702.03688.

[33] Joseph Emerson, Robert Alicki, and Karol Życzkowski. "Scalable noise estimation with random unitary operators". Journal of Optics B: Quantum and Semiclassical Optics **7**, S347 (2005).

[34] E. Knill, D. Leibfried, R. Reichle, J. Britton, R. B. Blakestad, J. D. Jost, C. Langer, R. Ozeri, S. Seidelin, and D. J. Wineland. "Randomized benchmarking of quantum gates". Physical Review A **77**, 012307 (2008).

[35] J. Helsen, I. Roth, E. Onorati, A.H. Werner, and J. Eisert. "General Framework for Randomized Benchmarking". PRX Quantum **3**, 020357 (2022).

[36] Athena Ceasura, Pavithran Iyer, Joel J. Wallman, and Hakop Pashayan. "Non-Exponential Behaviour in Logical Randomized Benchmarking" (2022). arXiv:2212.05488.

[37] J. Eli Bourassa, Nicolás Quesada, Ilan Tzitrin, Antal Száva, Theodor Isacsson, Josh Izaac, Krishna Kumar Sabapathy, Guillaume Dauphinais, and Ish Dhand. "Fast simulation of bosonic qubits via Gaussian functions in phase space". PRX Quantum **2**, 040315 (2021). arXiv:2103.05530.

[38] Mahnaz Jafarzadeh, Jonathan Conrad, Rafael N. Alexander, and Ben Q. Baragiola. "Logical channels in approximate Gottesman-Kitaev-Preskill error correction". Physical Review A **112**, 062413 (2025).

[39] Kyungjoo Noh, Christopher Chamberland, and Fernando G.S.L. Brandão. "Low-Overhead Fault-Tolerant Quantum Error Correction with the Surface-GKP Code". PRX Quantum **3**, 010315 (2022).

[40] David Mumford. "Tata Lectures on Theta I". Modern Birkhauser Classics. Birkhauser Boston. Boston, MA (2007).

[41] George A. Baker. "Formulation of Quantum Mechanics Based on the Quasi-Probability Distribution Induced on Phase Space". Physical Review **109**, 2198–2206 (1958).

[42] Howard J. Carmichael. "Statistical Methods in Quantum Optics 1". Springer Berlin Heidelberg. Berlin, Heidelberg (1999).

A Derivation of gaussian stabilizer

Starting from Eq. (36), we derive in this section the condition on M such that it defines a symplectic automorphism of the grid. In Sect. 3.2, we derived that the vector $(M-I)\,\boldsymbol{p}_j$ must be part of the grid Λ. A general element $\boldsymbol{p}_0$ of Λ can always be written as $\boldsymbol{p}_0 = S^T\mathbf{a}$ for $\mathbf{a}\in\mathbb{Z}^{2N}$. At the same time, a vector $\boldsymbol{p}_j$ of the dual grid can always be written as $\boldsymbol{p}_j = \Omega S^{-1}\mathbf{b}$ for $\mathbf{b}\in\mathbb{Z}^{2N}$. Equation (36) is modified to

$$(M-I)\,\Omega S^{-1}\mathbf{b} = S^T\mathbf{a}. \tag{89}$$

There always exist a matrix $X\in \mathrm{Mat}_{2N}(\mathbb{Z})$, potentially singular, such that $\mathbf{a} = -X\mathbf{b}$. Equation (89) then becomes

$$(M-I)\,\Omega S^{-1}\mathbf{b} = -S^TX\mathbf{b} \implies M = S^TXS\Omega + I. \tag{90}$$

But we know that $A = S\Omega S^T$, from which we have the form of Eq. (37)

$$M = S^T\left(XA+I\right)S^{-T}. \tag{91}$$

To force this matrix to be symplectic, we need that $M\Omega M^T = \Omega$. This condition becomes

$$S^T\left(XA+I\right)S^{-T}\Omega S^{-1}\left(A^TX^T+I\right)S = \Omega. \tag{92}$$

Using the relation $A = S\Omega S^T$ and the skew-symmetry of A, we arrive at the conclusion of Eq. (38):

$$XAX^T - X + X^T = 0. \tag{93}$$

B Fixing the translation part of Gaussian stabilizers

In this appendix, we derive the set of $2N$ equations obtained from Eq. (48) that need to be solve to obtain $\boldsymbol{\lambda}^*\in\Lambda^*$ in the transformation $\mathcal{L}_{M,\boldsymbol{\lambda}^*}$. Starting with the restriction from Eq. (48)

$$\hat{T}\left(\boldsymbol{\lambda}^*\right)U_M\hat{T}\left(\mathbf{p}_j\right)|\psi\rangle = \hat{T}\left(\mathbf{p}_j\right)\hat{T}\left(\boldsymbol{\lambda}^*\right)U_M|\psi\rangle, \tag{94}$$

we can permute the translation by the Pauli representative $\mathbf{p}_j$ with the other two operators on the left side, at the cost of a multiplication by M on the argument

$$\hat{T}\left(\boldsymbol{\lambda}^*\right)\hat{T}\left(M\mathbf{p}_j\right)U_M|\psi\rangle = \hat{T}\left(\mathbf{p}_j\right)\hat{T}\left(\boldsymbol{\lambda}^*\right)U_M|\psi\rangle, \tag{95}$$

and an additional phase

$$\exp\Big\{-2\pi i\boldsymbol{\lambda}^{*T}\Omega M\mathbf{p}_j\Big\}\hat{T}\left(M\mathbf{p}_j\right)\hat{T}\left(\boldsymbol{\lambda}^*\right)U_M\left|\psi\right\rangle=\hat{T}\left(\mathbf{p}_j\right)\hat{T}\left(\boldsymbol{\lambda}^*\right)U_M\left|\psi\right\rangle. \tag{96}$$

Because we are asking that $\mathcal{L}_{M,\boldsymbol{\lambda}^*}$ acts as a stabilizer, we can reduce the last equation to

$$\exp\Big\{-2\pi i\boldsymbol{\lambda}^{*T}\Omega M\mathbf{p}_j\Big\}\hat{T}\left(M\mathbf{p}_j\right)\left|\psi\right\rangle=\hat{T}\left(\mathbf{p}_j\right)\left|\psi\right\rangle. \tag{97}$$

At this point, we can reframe the argument of the translation by $M\mathbf{p}_j$ as a translation by $[M-I]\,\mathbf{p}_j+\mathbf{p}_j$

$$\exp\Big\{-2\pi i\boldsymbol{\lambda}^{*T}\Omega M\mathbf{p}_j\Big\}\hat{T}\left(\left[M-I\right]\mathbf{p}_j+\mathbf{p}_j\right)\left|\psi\right\rangle=\hat{T}\left(\mathbf{p}_j\right)\left|\psi\right\rangle, \tag{98}$$

and split it into two translation operators using Eq. (4)

$$\exp\Big\{-2\pi i\boldsymbol{\lambda}^{*T}\Omega M\mathbf{p}_j-\pi i\mathbf{p}_j^T\Omega\left[M-I\right]\mathbf{p}_j\Big\}\hat{T}\left(\mathbf{p}_j\right)\hat{T}\left(\left[M-I\right]\mathbf{p}_j\right)\left|\psi\right\rangle=\hat{T}\left(\mathbf{p}_j\right)\left|\psi\right\rangle. \tag{99}$$

With the definition of the matrix M, we know that $[M-I]\,\mathbf{p}_j\in\Lambda$ (see Eq. (36)) and so, for trivial gauge, we have

$$\exp\Big\{\pi i\mathbf{p}_j^T\left[M-I\right]^T S^{-1}A_{\diagdown}S^{-T}\left[M-I\right]\mathbf{p}_j\Big\}\hat{T}\left(\left[M-I\right]\mathbf{p}_j\right)\in\mathcal{S}, \tag{100}$$

where $A_{\diagdown}$ is the lower triangular part of the symplectic Gram matrix $A=S\Omega S^T$ and $\boldsymbol{\mu}$ is the gauge vector. With the identification of a stabilizer element in Eq. (100), we can reduce Eq. (99) to

$$\mathrm{e}^{-i\pi\left(2\boldsymbol{\lambda}^{*T}\Omega M\mathbf{p}_j+\mathbf{p}_j^T\Omega[M-I]\mathbf{p}_j+\mathbf{p}_j^T[M-I]^TS^{-1}A_{\diagdown}S^{-T}[M-I]\mathbf{p}_j\right)}\hat{T}\left(\mathbf{p}_j\right)\left|\psi\right\rangle=\hat{T}\left(\mathbf{p}_j\right)\left|\psi\right\rangle. \tag{101}$$

The solutions of this equation are the vectors $\boldsymbol{\lambda}^*$ for which the phase is the identity $\forall j$. What we get is $2N$ equations (one for each generator $\mathbf{p}_j$ of the dual lattice) of the form

$$2\boldsymbol{\lambda}^{*T}\Omega M\mathbf{p}_j+c_j=0\mod 2, \tag{102}$$

which can always be solve since the generators $\mathbf{p}_j$ are linearly independant.

C Generalized multimode sBs protocol

In this section, we derive a generalized sBs protocol to stabilize finite-energy GKP codewords with a general Fock-type envelope. We start by computing the normalized stabilizers of Eq. (61). To compute the effect of the normalization on the stabilizers, we only need to understand how it acts on the quadrature operators since

$$E\left(\Sigma_E,\mu_E\right)\hat{T}\left(\boldsymbol{v}\right)E\left(\Sigma_E,\mu_E\right)^{-1}=\mathrm{e}^{-il\mathbf{v}^T\Omega\,E(\Sigma_E,\mu_E)\,\hat{\boldsymbol{\xi}}\,E(\Sigma_E,\mu_E)^{-1}} \tag{103}$$

A general Gaussian envelope is described by a real symmetric matrix of covariance Σ_E and a vector of means $\boldsymbol{\mu}_E$. Any real symmetric matrix has a symplectic decomposition $\Sigma_E=M\Sigma_0M^T$, where $\Sigma_0=\varepsilon^{-1}\oplus\varepsilon^{-1}$ is diagonal and positive definite and M is a symplectic matrix. Using this decomposition, the envelope operator can be rewritten as

$$E\left(\Sigma_E,\mu_E\right)=\hat{T}\left(\boldsymbol{\mu}_E\right)U_M E\left(\Sigma_0,\mathbf{0}\right)U_M^{\dagger}\hat{T}\left(\boldsymbol{\mu}_E\right)^{\dagger}, \tag{104}$$

which means that the action of the envelope on the quadrature operators can be written as

$$E\left(\Sigma_E,\mu_E\right)\,\hat{\boldsymbol{\xi}}\,E\left(\Sigma_E,\mu_E\right)^{-1}=\hat{T}\left(\boldsymbol{\mu}_E\right)U_M E\left(\Sigma_0,\mathbf{0}\right)\left[M\hat{\boldsymbol{\xi}}+\boldsymbol{\mu}_E\right]E\left(\Sigma_0,\mathbf{0}\right)^{-1}U_M^{\dagger}\hat{T}\left(\boldsymbol{\mu}_E\right)^{\dagger}. \tag{105}$$

The envelope operator $E\left(\Sigma_0,\mathbf{0}\right)$ acts element-wise on each quadrature, leading to the simplification

$$E\left(\Sigma_E,\mu_E\right)\,\hat{\boldsymbol{\xi}}\,E\left(\Sigma_E,\mu_E\right)^{-1}=\hat{T}\left(\boldsymbol{\mu}_E\right)U_M\left[ME\left(\Sigma_0,\mathbf{0}\right)\hat{\boldsymbol{\xi}}E\left(\Sigma_0,\mathbf{0}\right)^{-1}+\boldsymbol{\mu}_E\right]U_M^{\dagger}\hat{T}\left(\boldsymbol{\mu}_E\right)^{\dagger}. \tag{106}$$

For a diagonal covariance matrix Σ_0, we get

$$\begin{aligned}E\left(\Sigma_0,\mathbf{0}\right)\hat{\boldsymbol{\xi}}E\left(\Sigma_0,\mathbf{0}\right)^{-1}&=\left[\cosh\left(\Sigma_0^{-1}\right)+i\sinh\left(\Sigma_0^{-1}\right)\Omega\right]\hat{\boldsymbol{\xi}},\\&\equiv h\left(\Sigma_0\right)\hat{\boldsymbol{\xi}},\end{aligned} \tag{107}$$

resulting in the full expression

$$E\left(\Sigma_E,\mu_E\right)\,\hat{\boldsymbol{\xi}}\,E\left(\Sigma_E,\mu_E\right)^{-1}=Mh\left(\Sigma_0\right)M^{-1}\left[\hat{\boldsymbol{\xi}}-\boldsymbol{\mu}_E\right]+\boldsymbol{\mu}_E. \tag{108}$$

Inserting this last equation into Eq. (103), we arrive at the expression of the finite-energy code stabilizers

$$\hat{T}_{\Sigma_E,\boldsymbol{\mu}_E}\left(\boldsymbol{s}_j\right)=\exp\left(-il\boldsymbol{s}_j^T\Omega\boldsymbol{\mu}_E\right)\exp\left(-il\boldsymbol{s}_j^T\Omega Mh\left(\Sigma_0\right)M^{-1}\left[\hat{\boldsymbol{\xi}}-\boldsymbol{\mu}_E\right]\right). \tag{109}$$

A state described by a translated envelope is stabilized by the same protocol as a centered one, but with a different gauge. Moving forward, we will omit this phase for simplicity and focus solely on finding an implementation for the second term. Following the derivation of Ref. [14], we compute the set of nullifiers

$$\begin{aligned}\hat{d}_j&=\frac{i}{\sqrt{2}}\ln\hat{T}_{\Sigma_E,\boldsymbol{\mu}_E}\left(\boldsymbol{s}_j\right) &(110)\\&=\frac{l}{\sqrt{2}}\boldsymbol{s}_j^T\Omega M\cosh\left(\Sigma_0^{-1}\right)M^{-1}\left[\hat{\boldsymbol{\xi}}_{\mathrm{mod}}-\boldsymbol{\mu}_E\right]+\frac{il}{\sqrt{2}}\boldsymbol{s}_j^T\Omega M\sinh\left(\Sigma_0^{-1}\right)\Omega M^{-1}\left[\hat{\boldsymbol{\xi}}-\boldsymbol{\mu}_E\right]\end{aligned}$$

that enable the stabilization of the desired manifold given the implementation of the unitary

$$\hat{U}_j=\mathrm{e}^{-i\Gamma_j\left[\hat{d}_j\hat{b}^{\dagger}+\hat{d}_j^{\dagger}\hat{b}\right]} \tag{111}$$

between the oscillators and a bath. The subscript on $\hat{\boldsymbol{\xi}}_{\mathrm{mod}}$ means that we take the quadrature operators modulo a translation by $lM\,\mathrm{sech}\left[\Sigma_0^{-1}\right]M^{-1}\boldsymbol{s}_j$, which comes from the branching of the logarithmic function. We choose to remove this condition on the quadrature operator, and instead apply this symmetry condition on the whole unitary $\hat{U}_j$. The cooling rate Γ_j (dimensionless) is then fixed by this condition. Under a few assumptions (see Ref. [14]), we modelize the bath as a qubit with $b=\frac{\sigma_x+i\sigma_y}{\sqrt{2}}$ that is discarded after each interaction. We also use a second-order Trotterization on the oscillator-bath interaction. This leads to an approximation of the unitary in Eq. (111):

$$\begin{aligned}\hat{U}_j\approx&\exp\left(\frac{-il\Gamma_j}{2}\boldsymbol{s}_j^T\Omega M\sinh\left(\Sigma_0^{-1}\right)\Omega M^{-1}\left[\hat{\boldsymbol{\xi}}-\boldsymbol{\mu}_E\right]\sigma_y\right) &(112)\\&\times\exp\left(-il\Gamma_j\boldsymbol{s}_j^T\Omega M\cosh\left(\Sigma_0^{-1}\right)M^{-1}\left[\hat{\boldsymbol{\xi}}-\boldsymbol{\mu}_E\right]\sigma_x\right) &(113)\\&\times\exp\left(\frac{-il\Gamma_j}{2}\boldsymbol{s}_j^T\Omega M\sinh\left(\Sigma_0^{-1}\right)\Omega M^{-1}\left[\hat{\boldsymbol{\xi}}-\boldsymbol{\mu}_E\right]\sigma_y\right).\end{aligned}$$

Asking for this unitary to be symmetric under the translation mentioned before results in the expression

$$\Gamma_j = \left(\boldsymbol{s}_j^T M^{-T} \tanh\left(\Sigma_0^{-1}\right) M^{-1} \boldsymbol{s}_j\right)^{-1} \tag{114}$$

for the cooling rate. This unitary can be cast in a circuit that is implemented only with controlled displacements and qubit rotations

$$\mathrm{C}\hat{T}\left(\boldsymbol{v}\right) = \exp\left(-il\boldsymbol{v}^T\Omega\hat{\boldsymbol{\xi}}\frac{\hat{\sigma}_z}{2}\right), \qquad R_x\left(\theta\right) = \exp\left(-i\theta\frac{\hat{\sigma}_x}{2}\right). \tag{115}$$

In this representation, we obtain the unitary operator

$$\hat{U}_j \approx \mathrm{C}\hat{T}\left(\boldsymbol{\alpha}_j\right)\hat{R}_z\left(l\boldsymbol{\alpha}_j^T\Omega\boldsymbol{\mu}_E\right)\hat{R}_x^\dagger\left(\frac{\pi}{2}\right)\hat{R}_z\left(-l\boldsymbol{\beta}_j^T\Omega\boldsymbol{\mu}_E\right)\mathrm{C}\hat{T}\left(\boldsymbol{\beta}_j\right) \tag{116}$$

$$\times\,\hat{R}_z\left(-l\boldsymbol{\beta}_j^T\Omega\boldsymbol{\mu}_E\right)\hat{R}_x\left(\frac{\pi}{2}\right)\hat{R}_z\left(l\boldsymbol{\alpha}_j^T\Omega\boldsymbol{\mu}_E\right)\mathrm{C}\hat{T}\left(\boldsymbol{\alpha}_j\right), \tag{117}$$

with small and big controlled displacements α_j and β_j, respectively defined as

$$\boldsymbol{\alpha}_j = -\frac{M\sinh\left(\Sigma_0^{-1}\right)M^T\Omega\boldsymbol{s}_j}{\boldsymbol{s}_j^T M^{-T}\tanh\left(\Sigma_0^{-1}\right)M^{-1}\boldsymbol{s}_j}, \quad \boldsymbol{\beta}_j = \frac{2M\cosh\left(\Sigma_0^{-1}\right)M^{-1}\boldsymbol{s}_j}{\boldsymbol{s}_j^T M^{-T}\tanh\left(\Sigma_0^{-1}\right)M^{-1}\boldsymbol{s}_j}. \tag{118}$$

This is the circuit presented in Fig. 1 of Sect. 4. A major difference with the sBs circuit of [14] is that the small and big displacements are not orthogonal for a general symplectic matrix M. Another difference is that, for non-zero $\boldsymbol{\mu}_E$, we get additional correction to the qubit rotations. Those rotations can be combined with the original $\pi/2$ rotations around the σ_x axis to keep the number of operations constant.

D Wigner function of GKP codes projectors

In this section, we derive the Wigner representation of the codespace projector of an arbitrary GKP code considering a gauge. This demonstration is inspired from appendix A in [17]. Our approach complete the demonstration for non-trivial gauges and non symplectically even lattice. As defined in [17, 18], an element of the grid $\boldsymbol{\lambda} \in \Lambda$ of Eq. (8) with generator matrix S is related to an element of the stabilizer group $\mathcal{S}$ up to a gauge $\boldsymbol{\mu} \in \mathbb{Z}_2^{2N}$ by the stabilizer equation:

$$\mathcal{S} = \left\{\nu_{\boldsymbol{\mu}}\left(\boldsymbol{\lambda}\right)\hat{T}\left(\boldsymbol{\lambda}\right)|\psi\rangle = |\psi\rangle\,,\ \forall\boldsymbol{\lambda}\in\Lambda\right\}, \tag{119}$$

where

$$\nu_{\boldsymbol{\mu}}\left(\boldsymbol{\lambda}\right) = \exp\left(i\pi\boldsymbol{\lambda}^T S^{-1}\left[A_{\triangleright}S^{-T}\boldsymbol{\lambda}+\boldsymbol{\mu}\right]\right), \tag{120}$$

and $A_{\triangleright}$ is the lower triangular part of the symplectic Gram matrix $A = S\Omega S^T$. After moving to the canonical basis $S_D = RS$ using the matrix R in Eq. (39), we have a simplified equation for the gauge

$$\nu_{\boldsymbol{\mu}}\left(\boldsymbol{\lambda}\right) = \exp\left(i\pi\boldsymbol{a}^T\left[A_{D\triangleright}\boldsymbol{a}+R\boldsymbol{\mu}\right]\right), \tag{121}$$

where we used $\boldsymbol{a} = S_D^{-T}\boldsymbol{\lambda} \in \mathbb{Z}^{2N}$. This transformation does not change the properties of the code, and we obtain the codespace projector as

$$\Pi_S = \sum_{\boldsymbol{\lambda}\in\Lambda}\nu_{\boldsymbol{\mu}}\left(\boldsymbol{\lambda}\right)\hat{T}\left(\boldsymbol{\lambda}\right) \tag{122}$$

The Wigner representation of this projector is then computed using the definition in Eq. (17)

$$W\left(\boldsymbol{\xi}, \Pi_S\right) = \int_{\mathbb{R}^{2N}} d\boldsymbol{v}\, e^{-i2\pi\boldsymbol{\xi}^T\Omega\boldsymbol{v}} \operatorname{Tr}\left[\hat{T}\left(\boldsymbol{v}\right)\Pi_S\right], \tag{123}$$

$$W\left(\boldsymbol{\xi}, \Pi_S\right) = \sum_{\boldsymbol{\lambda}\in\Lambda} \nu_{\boldsymbol{\mu}}\left(\boldsymbol{\lambda}\right) \int_{\mathbb{R}^{2N}} d\boldsymbol{v}\, e^{-i2\pi\boldsymbol{\xi}^T\Omega\boldsymbol{v}} \operatorname{Tr}\left[\hat{T}\left(\boldsymbol{v}\right)\hat{T}\left(\boldsymbol{\lambda}\right)\right]. \tag{124}$$

Since translation operators are trace orthogonal: $\operatorname{Tr}\left[\hat{T}\left(\boldsymbol{v}\right)\hat{T}\left(\boldsymbol{\lambda}\right)\right] = \delta^{2N}\left(\boldsymbol{v}+\boldsymbol{\lambda}\right)$, we obtain

$$W\left(\boldsymbol{\xi}, \Pi_S\right) = \sum_{\boldsymbol{\lambda}\in\Lambda} \nu_{\boldsymbol{\mu}}\left(\boldsymbol{\lambda}\right)\, e^{i2\pi\boldsymbol{\xi}^T\Omega\boldsymbol{\lambda}}. \tag{125}$$

$$W\left(\boldsymbol{\xi}, \Pi_S\right) = \sum_{\boldsymbol{a}\in\mathbb{Z}^{2N}} \exp\left(\pi i \boldsymbol{a}^T A_{D\diamondsuit}\boldsymbol{a} + 2\pi i\left[\frac{R}{2}\boldsymbol{\mu} - S_D\Omega\boldsymbol{\xi}\right]^T\boldsymbol{a}\right), \tag{126}$$

which can be rewritten in terms of a multivariable theta function [40]

$$W\left(\boldsymbol{\xi}, \Pi_S\right) = \frac{1}{\det\left(\Lambda\right)}\vartheta\left(\frac{R}{2}\boldsymbol{\mu} - S_D\Omega\boldsymbol{\xi}; A_{D\diamondsuit}\right). \tag{127}$$

We can go further in the description using the symmetry of the theta function in the second argument. Reminding ourselves that $A_D = \Omega_2 \otimes D$, only the components for which D_{kk} is odd contribute to a non-trivial phase. Defining the matrix $D_2 = (D \oplus D) \mod 2$, we can split the infinite sum into two part on $\boldsymbol{b} \in \mathbb{Z}^{2N}$ and $\boldsymbol{m} \in \mathbb{Z}_2^{2N}$:

$$\boldsymbol{a} = \left(I_{2N} + D_2\right)\boldsymbol{b} + D_2\boldsymbol{m}. \tag{128}$$

This manipulation separates the even and odd components of the vector $\boldsymbol{a}$, but only on the part for which the lattice is not symplectically even :

$$\boldsymbol{a}_k = \begin{cases} \boldsymbol{b}_k & \text{if } D_{kk} \mod 2 = 0 \\ 2\boldsymbol{b}_k + \boldsymbol{m}_k & \text{if } D_{kk} \mod 2 = 1 \end{cases} \tag{129}$$

Using this change of summation indexing, we find that the products $\left(I_{2N} + D_2\right) A_{D\diamondsuit} = A_{D\diamondsuit}\left(I_{2N} + D_2\right)$ is an even matrix in $2\mathbb{Z}^{2N\times 2N}$. This simplify the Wigner function of Eq. (126) to

$$\sum_{\boldsymbol{m}\in\mathbb{Z}_2^{2N}} e^{\left(\pi i \boldsymbol{m}^T D_2 A_{D\diamondsuit} D_2\boldsymbol{m} + 2\pi i\left[\frac{D_2 R}{2}\boldsymbol{\mu} - D_2 S_D\Omega\boldsymbol{\xi}\right]^T\boldsymbol{m}\right)} \sum_{\boldsymbol{b}\in\mathbb{Z}^{2N}} e^{2\pi i\left[\frac{(I+D_2)R}{2}\boldsymbol{\mu} - (I+D_2)S_D\Omega\boldsymbol{\xi}\right]^T\boldsymbol{b}}. \tag{130}$$

The first sum in $\boldsymbol{m}$, that we note $C\left(S, \boldsymbol{\mu}, \boldsymbol{\xi}\right)$, is finite and can be computed for a specified lattice. For the second series, we can compute it as a product of Poisson summations to find

$$\sum_{\boldsymbol{b}\in\mathbb{Z}^{2N}} e^{2\pi i\left[\frac{(I+D_2)R}{2}\boldsymbol{\mu} - (I+D_2)S_D\Omega\boldsymbol{\xi}\right]^T\boldsymbol{b}} = \frac{1}{\det\left(\tilde{\Lambda}\right)} \sum_{\tilde{\boldsymbol{\lambda}}^*\in\tilde{\Lambda}^*} \delta^{(2N)}\left(\boldsymbol{\xi} - \left[\tilde{\boldsymbol{\lambda}}^* - \frac{1}{2}S^*\boldsymbol{\mu}\right]\right) \tag{131}$$

with $S^* = \Omega S^{-1}$. We have also defined a partially rescale lattice $\tilde{\Lambda}$:

$$\tilde{\Lambda} = \left\{\left[\left(I + D_2\right)S_D\right]^T\boldsymbol{a} \,|\, \boldsymbol{a} \in \mathbb{Z}^{2N}\right\}, \tag{132}$$

and its symplectic dual lattice

$$\tilde{\Lambda}^* = \left\{ \Omega \left[(I + D_2) S_D \right]^{-1} \boldsymbol{a} \,|\, \boldsymbol{a} \in \mathbb{Z}^{2N} \right\}, \tag{133}$$

We finally arrive at the result for the Wigner function for the projector onto the codespace of a GKP code with generator matrix S and gauge $\boldsymbol{\mu}$:

$$W(\boldsymbol{\xi}, \Pi_S) = \frac{1}{\det\left(\tilde{\Lambda}\right)} C(S, \boldsymbol{\mu}, \boldsymbol{\xi}) \sum_{\tilde{\boldsymbol{\lambda}}^* \in \tilde{\Lambda}^*} \delta^{(2N)} \left(\boldsymbol{\xi} - \left[\tilde{\boldsymbol{\lambda}}^* - \frac{1}{2} S^* \boldsymbol{\mu} \right] \right). \tag{134}$$

For the final result of this section, we produce the Wigner function of a pure grid state $|\Psi_S\rangle$. Provided that we are looking at the projector over a pure GKP grid state $|\Psi_S\rangle$. Then, the Wigner function of the projector on $|\Psi_S\rangle$ is the Wigner function of the state itself. In this case, because the canonical symplectic Gram matrix is only Ω, $D_2 = I$ and Eq. (134) simplify to

$$W(\boldsymbol{\xi}, |\Psi_S\rangle) = \frac{C(S, \boldsymbol{\mu}, \boldsymbol{\xi})}{4^{2N}} \sum_{\boldsymbol{\lambda}^* \in \Lambda^*} \delta^{(2N)} \left(\boldsymbol{\xi} - \frac{1}{2} \left[\boldsymbol{\lambda}^* - S^* \boldsymbol{\mu} \right] \right), \tag{135}$$

with $\tilde{S} = 2S$. As a result, all grid states can be represented in phase-space as a sum of Dirac delta functions positioned at half the dual lattice vectors under a trivial gauge ($\boldsymbol{\mu} = \mathbf{0}$):

$$W(\boldsymbol{\xi}, |\Psi_S\rangle) \propto \sum_{\boldsymbol{\lambda}^* \in \Lambda^*} C\left(S, \mathbf{0}, \frac{\boldsymbol{\lambda}^*}{2} \right) \delta^{(2N)} \left(\boldsymbol{\xi} - \frac{\boldsymbol{\lambda}^*}{2} \right). \tag{136}$$

E Wigner function of multimode finite energy GKP codes states

As described in the main text, a general pure grid state of a finite energy GKP code is described by the infinite energy grid state and an envelope $E(\Sigma_E, \boldsymbol{\mu}_E)$:

$$\left| \Psi_{S, \Sigma_E, \boldsymbol{\mu}_E} \right\rangle = \hat{E}(\Sigma_E, \boldsymbol{\mu}_E) |\Psi_S\rangle. \tag{137}$$

In this section, we derive the Wigner representation of this finite energy grid state as a function of S, Σ_E and $\boldsymbol{\mu}_E$. The Wigner representation of this state has a simple decomposition as a star-product of Wigner functions of the envelope and the infinite energy state

$$W(\boldsymbol{\xi}, |\Psi_{S, \Sigma_E, \boldsymbol{\mu}_E}\rangle) = W\left(\boldsymbol{\xi}, \hat{E}\right) \star W(\boldsymbol{\xi}, |\Psi_S\rangle) \star W\left(\boldsymbol{\xi}, \hat{E}\right). \tag{138}$$

This star-product is defined as

$$A \star B := A \prod_{j=1}^{N} \exp\left(\frac{i}{2} \left[\overleftarrow{\partial}_{q_j} \vec{\partial}_{p_j} - \overleftarrow{\partial}_{p_j} \vec{\partial}_{q_j} \right] \right) B \tag{139}$$

with $\vec{\partial}_a$ the partial derivative with respect to a acting in the direction of the arrow. Another equivalent definition for the star-product is [41]

$$A(\boldsymbol{\xi}) \star B(\boldsymbol{\xi}) = 2^{2N} \int_{\mathbb{R}^{2N}} d\boldsymbol{\eta} d\boldsymbol{\nu} A(\boldsymbol{\xi} + \boldsymbol{\eta}) B(\boldsymbol{\xi} + \boldsymbol{\nu}) \, \mathrm{e}^{4\pi i \boldsymbol{\eta}^T \Omega \boldsymbol{\nu}}, \tag{140}$$

which has been rescaled according to our definition of the Wigner function in Eq. (17). This product is associative and bilinear. Combining the argument of bilinearity with the

result of the last section that a grid state has a Wigner function that decomposes as a sum of Dirac delta functions (See Sect. D), Eq. (138) reduces to

$$W\left(\boldsymbol{\xi}, |\Psi_{S,\Sigma_E,\boldsymbol{\mu}_E}\rangle\right) = \sum_{\boldsymbol{\lambda}^* \in \Lambda^*} c_{\boldsymbol{\lambda}^*} W\left(\boldsymbol{\xi}, \hat{E}\right) \star \delta^{(2N)}\left(\boldsymbol{\xi} - \frac{1}{2}\left[\boldsymbol{\lambda}^* - S^*\boldsymbol{\mu}\right]\right) \star W\left(\boldsymbol{\xi}, \hat{E}\right). \quad (141)$$

where $c_{\boldsymbol{\lambda}^*} = C\left(S, \boldsymbol{\mu}, \frac{\boldsymbol{\lambda}^*}{2}\right)/4^{2N}$. Then, we only need to compute each term of the sum:

$$W\left(\boldsymbol{\xi}, \hat{E}\right) \star \delta^{(2N)}\left(\boldsymbol{\xi} - \boldsymbol{\chi}\right) \star W\left(\boldsymbol{\xi}, \hat{E}\right). \quad (142)$$

To achieve this, we first compute the Wigner representation of the envelope. A general envelope $\hat{E}\left(\Sigma_E, \boldsymbol{\mu}_E\right)$ can always be decomposed as a displaced and squeezed Gaussian function

$$\hat{E}\left(\Sigma_E, \boldsymbol{\mu}_E\right) = \hat{E}\left(M_E \Sigma_0 M_E^T, \boldsymbol{\mu}_E\right) = \hat{T}\left(\boldsymbol{\mu}_E\right) U_{M_E} \hat{E}\left(\Sigma_0, \mathbf{0}\right) U_{M_E}^\dagger \hat{T}\left(\boldsymbol{\mu}_E\right)^\dagger, \quad (143)$$

where Σ_0 is the symplectic diagonalisation of Σ_E. This canonical form of the envelope is connected to a thermal state of temperature Σ_0^{-1} in multiple modes. They are explicitly related as

$$\hat{E}\left(\Sigma_0, \mathbf{0}\right) = \frac{1}{\sqrt{\det\left(I - \exp\left(-\Sigma_0^{-1}\right)\right)}} \hat{\rho}_{\text{thermal}}. \quad (144)$$

The full thermal state is the product of one-mode thermal states. Therefore, $\hat{\rho}_{\text{thermal}}$ is a product of Gaussian states with zero mean and covariance matrix $\left(2\overline{n}_k + 1\right) I/4\pi$ [27], where $\overline{n}_k$ is the mean number of excitation in mode k. Alternatively, we can use the connection $\overline{n}_k = \left[\exp\left(\varepsilon_k\right)\right]^{-1}$ between excitation number and temperature of the thermal state to find that its covariance matrix can also be written as $\coth\left(\varepsilon_k/2\right) I/4\pi$. Combining N single-mode thermal states result in a Gaussian function with a covariance matrix

$$\frac{1}{4\pi} \coth\left(\Sigma_0^{-1}/2\right). \quad (145)$$

Using this connection with the thermal state, we find that the Wigner representation of this simple envelope is given by

$$W\left(\boldsymbol{\xi}; \hat{E}\left(\Sigma_0, \mathbf{0}\right)\right) = \det\left(I - \exp\left(-\Sigma_0^{-1}\right)\right)^{-1/2} G_{\frac{1}{4\pi}\coth\left(\Sigma_0^{-1}/2\right), \mathbf{0}}\left(\boldsymbol{\xi}\right), \quad (146)$$

with G defined in Eq. (21). From this result, we can obtain the Wigner representation of a general envelope by using Eq. (19):

$$W\left(\boldsymbol{\xi}; \hat{E}\left(\Sigma_E, \boldsymbol{\mu}_E\right)\right) = \det\left(I - \exp\left(-\Sigma_0^{-1}\right)\right)^{-1/2} G_{\frac{1}{4\pi} M_E \coth\left(\Sigma_0^{-1}/2\right) M_E^T, \boldsymbol{\mu}_E}\left(\boldsymbol{\xi}\right). \quad (147)$$

To simplify the notation, we rewrite this last equation as

$$W\left(\boldsymbol{\xi}; \hat{E}\left(\Sigma_E, \boldsymbol{\mu}_E\right)\right) = \mathcal{N} G_{A, \boldsymbol{a}}\left(\boldsymbol{\xi}\right), \quad (148)$$

where $A = \frac{1}{4\pi} M_E \coth\left(\Sigma_0^{-1}/2\right) M_E^T$ and $\boldsymbol{a} = \boldsymbol{\mu}_E$. With this intermediate result, we can move on to compute Eq. (142). Starting with the left star-product, we have

$$W\left(\boldsymbol{\xi}, \hat{E}\right) \star \delta^{(2N)}\left(\boldsymbol{\xi} - \boldsymbol{\chi}\right) = 2^{2N} \mathcal{N} \int_{\mathbb{R}^{2N}} d\boldsymbol{\eta} d\boldsymbol{\nu}\, G_{A,\boldsymbol{a}}\left(\boldsymbol{\xi} + \boldsymbol{\eta}\right) \delta^{(2N)}\left(\boldsymbol{\xi} + \boldsymbol{\nu} - \boldsymbol{\chi}\right) \mathrm{e}^{4\pi i \boldsymbol{\eta}^T \Omega \boldsymbol{\nu}}. \quad (149)$$

We then use a change of variable $\tilde{\boldsymbol{\eta}} = \boldsymbol{\eta} + \boldsymbol{\xi}$

$$(149) = 2^{2N}\mathcal{N}\int_{\mathbb{R}^{2N}} d\boldsymbol{\nu}\, \delta^{(2N)}\left(\boldsymbol{\xi}+\boldsymbol{\nu}-\boldsymbol{\chi}\right) \mathrm{e}^{-4\pi i \boldsymbol{\xi}^T\Omega\boldsymbol{\nu}} \int_{\mathbb{R}^{2N}} d\tilde{\boldsymbol{\eta}}\, G_{A,\boldsymbol{a}}\left(\tilde{\boldsymbol{\eta}}\right) \mathrm{e}^{4\pi i \tilde{\boldsymbol{\eta}}^T\Omega\boldsymbol{\nu}}. \tag{150}$$

The integral over $\tilde{\boldsymbol{\eta}}$ can be identified as the characteristic function of a Gaussian distribution at the position $4\pi\Omega\nu$ to obtain

$$(149) = 2^{2N}\mathcal{N}\int_{\mathbb{R}^{2N}} d\boldsymbol{\nu}\, \delta^{(2N)}\left(\boldsymbol{\xi}+\boldsymbol{\nu}-\boldsymbol{\chi}\right) \mathrm{e}^{4\pi i[\boldsymbol{a}-\boldsymbol{\xi}]^T\Omega\boldsymbol{\nu}} \exp\left(\frac{-16\pi^2}{2}\boldsymbol{\nu}^T\Omega A\Omega^T\boldsymbol{\nu}\right). \tag{151}$$

We complete the computation of the product using the Dirac delta distribution inside the integral

$$(149) = 2^{2N}\mathcal{N}\mathrm{e}^{4\pi i[\boldsymbol{\xi}-\boldsymbol{a}]^T\Omega[\boldsymbol{\xi}-\boldsymbol{\chi}]} \exp\left(\frac{-16\pi^2}{2}\left[\boldsymbol{\xi}-\boldsymbol{\chi}\right]^T\Omega A\Omega^T\left[\boldsymbol{\xi}-\boldsymbol{\chi}\right]\right). \tag{152}$$

With the first star product computed, we now proceed to the second part

$$\begin{aligned}(142) =& 2^{2N}\mathcal{N}^2\int_{\mathbb{R}^{2N}} d\boldsymbol{\eta}d\boldsymbol{\nu}\, \mathrm{e}^{4\pi i[\boldsymbol{\xi}+\boldsymbol{\eta}-\boldsymbol{a}]^T\Omega[\boldsymbol{\xi}+\boldsymbol{\eta}-\boldsymbol{\chi}]} \times \\ & \exp\left(\frac{-16\pi^2}{2}\left[\boldsymbol{\xi}+\boldsymbol{\eta}-\boldsymbol{\chi}\right]^T\Omega A\Omega^T\left[\boldsymbol{\xi}+\boldsymbol{\eta}-\boldsymbol{\chi}\right]\right) G_{A,\boldsymbol{a}}\left(\boldsymbol{\xi}+\boldsymbol{\nu}\right) \mathrm{e}^{4\pi i\boldsymbol{\eta}^T\Omega\boldsymbol{\nu}}.\end{aligned} \tag{153}$$

Once again, we use a change of variable $\tilde{\boldsymbol{\nu}} = \boldsymbol{\nu} + \boldsymbol{\xi}$ and identify the characteristic function of a Gaussian distribution to compute a first integral. Combining two Gaussian functions and using the property $\omega\left(\boldsymbol{\nu},\boldsymbol{\nu}\right) = 0$ to simplify phase terms, we arrive at Eq. (154)

$$\begin{aligned}(142) =& 2^{2N}\mathcal{N}^2 \exp\left(\frac{-8\pi^2}{2}\left[\boldsymbol{\xi}-\boldsymbol{\chi}\right]^T\Omega A\Omega^T\left[\boldsymbol{\xi}-\boldsymbol{\chi}\right]\right) \mathrm{e}^{4\pi i[\boldsymbol{\xi}-\boldsymbol{a}]^T\Omega[\boldsymbol{\xi}-\boldsymbol{\chi}]} \times \\ & \int_{\mathbb{R}^{2N}} d\boldsymbol{\eta}\, \mathrm{e}^{4\pi i[\boldsymbol{\xi}+\boldsymbol{\chi}-2\boldsymbol{a}]^T\Omega\boldsymbol{\eta}} \exp\left(\frac{-32\pi^2}{2}\left[\boldsymbol{\eta}+\frac{\boldsymbol{\xi}-\boldsymbol{\chi}}{2}\right]^T\Omega A\Omega^T\left[\boldsymbol{\eta}+\frac{\boldsymbol{\xi}-\boldsymbol{\chi}}{2}\right]\right).\end{aligned} \tag{154}$$

The last exponential function in Eq. (154) is linked to a Gaussian distribution function through

$$\exp\left(\frac{-32\pi^2}{2}\left[\boldsymbol{\eta}+\frac{\boldsymbol{\xi}-\boldsymbol{\chi}}{2}\right]^T\Omega A\Omega^T\left[\boldsymbol{\eta}+\frac{\boldsymbol{\xi}-\boldsymbol{\chi}}{2}\right]\right) = \sqrt{\det\left(\frac{A^{-1}}{16\pi}\right)} G_{\frac{\Omega A^{-1}\Omega^T}{32\pi^2},\frac{\boldsymbol{\chi}-\boldsymbol{\xi}}{2}}\left(\boldsymbol{\eta}\right), \tag{155}$$

with

$$\det\left(\frac{A^{-1}}{16\pi}\right) = 2^{-4N}\det\left(\tanh\left(\Sigma_0^{-1}/2\right)\right), \tag{156}$$

once we insert back the initial form of A. With this identification, we can again use the definition of the characteristic function of a Gaussian distribution to compute this last integral

$$\begin{aligned}(142) =& \sqrt{\det\left(\tanh\left(\Sigma_0^{-1}/2\right)\right)}\mathcal{N}^2 \exp\left(\frac{-8\pi^2}{2}\left[\boldsymbol{\xi}-\boldsymbol{\chi}\right]^T\Omega A\Omega^T\left[\boldsymbol{\xi}-\boldsymbol{\chi}\right]\right) \mathrm{e}^{4\pi i[\boldsymbol{\xi}-\boldsymbol{a}]^T\Omega[\boldsymbol{\xi}-\boldsymbol{\chi}]} \times \\ & \int_{\mathbb{R}^{2N}} d\boldsymbol{\eta}\, \mathrm{e}^{i(-4\pi\Omega[\boldsymbol{\xi}+\boldsymbol{\chi}-2\boldsymbol{a}])^T\boldsymbol{\eta}} G_{\frac{\Omega A^{-1}\Omega^T}{32\pi^2},\frac{\boldsymbol{\chi}-\boldsymbol{\xi}}{2}}\left(\boldsymbol{\eta}\right).\end{aligned} \tag{157}$$

After a few manipulations, we get the simplified equation

$$\begin{aligned}(142) =&\sqrt{\det\left(\tanh\left(\Sigma_0^{-1}/2\right)\right)}\mathcal{N}^2\exp\left(\frac{-8\pi^2}{2}\left[\boldsymbol{\xi}-\boldsymbol{\chi}\right]^T\Omega A\Omega^T\left[\boldsymbol{\xi}-\boldsymbol{\chi}\right]\right)\times\\&\exp\left(\frac{-1}{2}\left[\boldsymbol{\xi}+\boldsymbol{\chi}-2\boldsymbol{a}\right]^T\frac{A^{-1}}{2}\left[\boldsymbol{\xi}+\boldsymbol{\chi}-2\boldsymbol{a}\right]\right).\end{aligned}\tag{158}$$

At this point, it is useful to combine both exponentials into a single Gaussian function by completing the square in the exponent. After inserting the definition of A and $\boldsymbol{a}$ into Eq. (158), we get

$$\begin{aligned}(142) =&\frac{\sqrt{\det\left(\tanh\left(\Sigma_0^{-1}/2\right)\right)}}{\det\left(I-\exp\left(-\Sigma_0^{-1}\right)\right)}\exp\left(\frac{-1}{2}\left[\boldsymbol{\chi}-\boldsymbol{\mu}_E\right]^T\left[\frac{M_E\coth\left(\Sigma_0^{-1}\right)M_E^T}{4\pi}\right]^{-1}\left[\boldsymbol{\chi}-\boldsymbol{\mu}_E\right]\right)\times\\&\exp\left(\frac{-1}{2}\left[\boldsymbol{\xi}-\boldsymbol{\chi}_E\right]^T\left[\frac{M_E\tanh\left(\Sigma_0^{-1}\right)M_E^T}{4\pi}\right]^{-1}\left[\boldsymbol{\xi}-\boldsymbol{\chi}_E\right]\right),\end{aligned}\tag{159}$$

where we have extracted the altered peak position

$$\boldsymbol{\chi}_E = M_E\,\mathrm{sech}\left(\Sigma_0^{-1}\right)M_E^{-1}\left[\boldsymbol{\chi}-\boldsymbol{\mu}_E\right]+\boldsymbol{\mu}_E.\tag{160}$$

Equation (159) tells us that the normalization process of adding a Fock-type envelope to a GKP state Wigner function maps each Dirac distribution of the sum in Eq. (141) to a Gaussian function of amplitude described by the first line of Eq. (159), centered around $\boldsymbol{\chi}_E$ and with covariance matrix $M_E\tanh\left(\Sigma_0^{-1}\right)M_E^T/4\pi$. This describes entirely the Wigner function of any finite-energy GKP state with a Fock-type envelope. Especially, Eq. (159) reduces to the result of Ref. [37] when $M_E = I$ and $\boldsymbol{\mu}_E = \mathbf{0}$. If we set $\Sigma_0 = \alpha^{-1}I$ and take the limit $\alpha \to 0^+$, we recover the infinite energy state. In this limit, the peaks position are unaltered $\boldsymbol{\chi}_E \to \boldsymbol{\chi}$, each peak is infinitely squeezed with an infinite amplitude because $\tanh(\Sigma_0) \to 0$ and the envelope becomes infinitely large because $\coth(\Sigma_0) \to \infty$.

F Average number of excitation in GKP states

In this section, we aim to obtain an expression for the average number of excitations in a GKP state with generator matrix S and an envelope described by covariance matrix Σ_E and position in phase-space $\boldsymbol{\mu}_E$. In Sect. E, we have derived the Wigner function of such a state, and in the following, we are going to use this result. From our definition of the Wigner function

$$W(\boldsymbol{\xi},\hat{\rho}) = \int_{\mathbb{R}^{2N}} d\boldsymbol{v}\,\mathrm{e}^{-i2\pi\boldsymbol{\xi}^T\Omega\boldsymbol{v}}\,\mathrm{Tr}\left[\hat{T}(\boldsymbol{v})\,\hat{\rho}\right],\tag{161}$$

we can derive the expression for the average excitation number [42]

$$\left\langle\hat{\boldsymbol{\xi}}^T\hat{\boldsymbol{\xi}}\right\rangle = \mathrm{Tr}\left[\hat{\boldsymbol{\xi}}^T\hat{\boldsymbol{\xi}}\hat{\rho}\right] = (2\pi)^{2N}\int_{\mathbb{R}^{2N}} d\boldsymbol{\xi}\,\boldsymbol{\xi}^T\boldsymbol{\xi}W(\boldsymbol{\xi},\hat{\rho}),\tag{162}$$

for a state described by the Wigner function $W(\boldsymbol{\xi},\hat{\rho})$. With this intermediate result, we are ready to compute it with the Wigner function of GKP states from Eq. (159). To simplify

the demonstration, we express the Wigner function of GKP states as

$$W\left(\boldsymbol{\xi}, \hat{\rho}_{\mathrm{GKP}}\right) = \sum_{\boldsymbol{\chi}\in\mathcal{M}} c_{\boldsymbol{\chi}} \exp\left(\frac{-1}{2}\left[\boldsymbol{\xi}-\boldsymbol{\chi}_E\right]^T B^{-1}\left[\boldsymbol{\xi}-\boldsymbol{\chi}_E\right]\right), \tag{163}$$

with $\mathcal{M} = \left(\Lambda^* - S^*\boldsymbol{\mu}\right)/2$ from Eq. (135), $\boldsymbol{\chi_E}$ from Eq. (160) and $B = M_E \tanh\left(\Sigma_0^{-1}\right) M_E^T/4\pi$. This leads us to compute the integral

$$\left\langle \hat{\boldsymbol{\xi}}^T \hat{\boldsymbol{\xi}} \right\rangle = (2\pi)^{2N} \sum_{\boldsymbol{\chi}\in\mathcal{M}} c_{\boldsymbol{\chi}} \int_{\mathbb{R}^{2N}} d\boldsymbol{\xi}\, \boldsymbol{\xi}^T\boldsymbol{\xi} \exp\left(\frac{-1}{2}\left[\boldsymbol{\xi}-\boldsymbol{\chi}_E\right]^T B^{-1}\left[\boldsymbol{\xi}-\boldsymbol{\chi}_E\right]\right). \tag{164}$$

This is an integral that we can compute analytically using a change of variable and adding a source term. We reframe the problem as solving the integral

$$\int_{\mathbb{R}^{2N}} d\boldsymbol{\xi}\, \left[\boldsymbol{\xi}+\boldsymbol{\chi}_E\right]^T \left[\boldsymbol{\xi}+\boldsymbol{\chi}_E\right] \exp\left(\frac{-1}{2}\boldsymbol{\xi}^T B^{-1}\boldsymbol{\xi} + \boldsymbol{J}^T\boldsymbol{\xi}\right)\Big|_{\boldsymbol{J}=\boldsymbol{0}}. \tag{165}$$

With this trick, we can move the polynomial term out of the integral by changing the variable for derivatives in $\boldsymbol{J}$

$$\left[\frac{\partial}{\partial \boldsymbol{J}} + \boldsymbol{\chi}_E\right]^T \left[\frac{\partial}{\partial \boldsymbol{J}} + \boldsymbol{\chi}_E\right] \int_{\mathbb{R}^{2N}} d\boldsymbol{\xi}\, \exp\left(\frac{-1}{2}\boldsymbol{\xi}^T B^{-1}\boldsymbol{\xi} + \boldsymbol{J}^T\boldsymbol{\xi}\right)\Big|_{\boldsymbol{J}=\boldsymbol{0}}, \tag{166}$$

and we obtain a multivariate Gaussian integral. The result of this integral is

$$(2\pi)^N \sqrt{\det\left(B\right)} \left[\frac{\partial}{\partial \boldsymbol{J}} + \boldsymbol{\chi}_E\right]^T \left[\frac{\partial}{\partial \boldsymbol{J}} + \boldsymbol{\chi}_E\right] \exp\left(\frac{1}{2}\boldsymbol{J}^T B \boldsymbol{J}\right)\Big|_{\boldsymbol{J}=\boldsymbol{0}}, \tag{167}$$

and we can reduce it to

$$(2\pi)^N \sqrt{\det\left(B\right)} \left(\boldsymbol{\chi}_E^T\boldsymbol{\chi}_E + \mathrm{Tr}\left[B\right]\right), \tag{168}$$

which is the contribution to energy from one Gaussian peak of the GKP Wigner function. Putting everything together and doing simplifications on prefactors, we arrive at the result that the average number of excitations in a GKP state described by the generator matrix S and envelope $(\Sigma_E, \boldsymbol{\mu}_E) = \left(M_E\Sigma_0 M_E^T, \boldsymbol{\mu}_E\right)$ is

$$\begin{aligned}\left\langle \hat{\boldsymbol{\xi}}^T \hat{\boldsymbol{\xi}} \right\rangle =& \pi^{2N} \sqrt{\det\left(I + \tanh\left(\Sigma_0^{-1}\right)\right)} \sum_{\boldsymbol{\chi}\in\mathcal{M}} c_{\boldsymbol{\chi}} \left(\boldsymbol{\chi}_E^T\boldsymbol{\chi}_E + \mathrm{Tr}\left[M_E \tanh\left(\Sigma_0^{-1}\right) M_E^T\right]\right) \\ &\times \exp\left(\frac{-1}{2}\left[\boldsymbol{\chi}-\boldsymbol{\mu}_E\right]^T \left[\frac{M_E \coth\left(\Sigma_0^{-1}\right) M_E^T}{4\pi}\right]^{-1} \left[\boldsymbol{\chi}-\boldsymbol{\mu}_E\right]\right).\end{aligned} \tag{169}$$

As a final remark, we can also obtain the number of excitations in a sub-system (in a single mode, for example) by restricting the scalar product and trace in Eq. (168) to be only over elements of this sub-system and rescaling the prefactors accordingly.

G Application of algorithm 1

In this section, we work out the optimization process of algorithm 1 for the toy logical Clifford circuit on one qubit:

$\Sigma_0 = I, \mu_0 = 0$ —[$\overline{P}$]—[$\overline{P}$]—[$\overline{P}$]—[$\overline{P}$]—, (170)

applied on a square GKP qubit. This circuit, although trivial in practice, is ideal for demonstrating the mechanics of the algorithm. As inputs of algorithm 1, we use the parameters $\Sigma_0 = I$ and $\mu_0 = 0$, as they simplify the computations and make the demonstration more intuitive. For those parameters, both metrics (Eq. (73) and Eq. (80)) are zero.

The first task of the compiler is to identify all potential implementations of the targeted circuit. For the optimization of the circuit in (170), only the stabilizer group generated by two elements $\mathcal{S} = \langle Z^2, P^2 Z \rangle$ is useful, where we used the definitions of Sect. 3.3 for the operations. At first order of the generators, this group contains the elements

$$\mathcal{S}_1 = \left\{ I, Z^2, Z^{-2}, P^2 Z, P^2 Z^{-1}, P^{-2} Z, P^{-2} Z^{-1}, P^2 Z^3, P^{-2} Z^{-3} \right\}, \tag{171}$$

where every element of the set is the product of the generators at a power of either $-1, 0$ or 1. By multiplying the set $\mathcal{S}_1$ by a representative of $\overline{P}$ (i.e. P), we then obtain the different implementations of the logical gate $\overline{P}$ at first order

$$P\mathcal{S}_1 = \left\{ P, PZ^2, PZ^{-2}, P^3 Z, P^3 Z^{-1}, P^{-1} Z, P^{-1} Z^{-1}, P^3 Z^3, P^{-1} Z^{-3} \right\}. \tag{172}$$

For the purpose of this demonstration, we retained only the elements with unit magnitude power for all operations

$$\left\{ P, P^{-1} Z, P^{-1} Z^{-1} \right\}. \tag{173}$$

The result of the optimization is not affected by the simplification in this specific case. The second task is to identify the parameters $(M, \boldsymbol{\lambda})$ of each implementation following the form $U = \hat{T}(\boldsymbol{\lambda}) U_M$ described in algorithm 1. Based on the derivations of Sect. 3.3, we find the parametrization

$$\left\{ \left(\begin{bmatrix} 1 & 0 \\ -1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 0 \end{bmatrix} \right), \left(\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1/\sqrt{2} \end{bmatrix} \right), \left(\begin{bmatrix} 1 & 0 \\ 1 & 1 \end{bmatrix}, \begin{bmatrix} 0 & -1/\sqrt{2} \end{bmatrix} \right) \right\} \tag{174}$$

for the operations $\{P, P^{-1} Z, P^{-1} Z^{-1}\}$, respectively. Before applying the algorithm, we observe that the set of matrices

$$M(n) = \begin{bmatrix} 1 & 0 \\ -n & 1 \end{bmatrix} \tag{175}$$

form a group for which

$$M(n) M(m) = M(n+m) \qquad \text{and} \qquad M(n) M(n)^T = \begin{bmatrix} 1 & -n \\ -n & n^2 + 1 \end{bmatrix}. \tag{176}$$

When we compute the metric Eq. (80) for this set of matrices, we get

$$\mathrm{Tr}\left[\ln^2 \left(M(n) M(n)^T \right) \right] = \ln^2 \left(\frac{n^2 + 2 - n\sqrt{n^2+4}}{2} \right) + \ln^2 \left(\frac{n^2 + 2 + n\sqrt{n^2+4}}{2} \right), \tag{177}$$

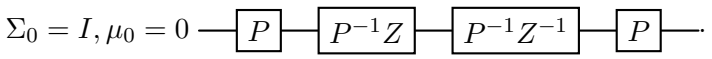
which is symmetric in n and increasing with the magnitude of n. This is useful because the combination of elements of Eq. (174) is all part of this set. With all the tools in hand, we can now complete the application of algorithm 1.

In the first round of optimization, we try to optimize the first gate in the logical circuit (170). Initially, the envelope parameters are $\Sigma_0 = I$ and $\boldsymbol{\mu}_0 = \mathbf{0}$. At the squeezing level,

all implementations in Eq. (174) are equivalent (the squeezing metric of Eq. (177) is the same for $n = 1$ and $n = -1$). Therefore, we turn to the displacement metric to make the discrimination. The second and the third operations both contain displacement, while the first does not. It means that the first implementation of $\overline{P}$ by P is optimal and that after the gate, the envelope parameters are

$$\Sigma = \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix} \qquad \text{and} \qquad \boldsymbol{\mu} = \mathbf{0}. \tag{178}$$

We move on to the second gate in circuit (170). We have the choice between implementing the squeezing part of the gate with $M(1)$ or $M(-1)$. While the options are the same as for the first implementation, the optimal choice might be different because the input state is different. Choosing $M(1)$ for the second implementation lead to the metric of Eq. (177) with $n = 2$, while choosing $M(-1)$ reverse the squeezing action of the first implemented gate and lead to the same metric with $n = 0$. Therefore, the choice $M(-1)$ is optimal. Both options with $M(-1)$ in Eq. (174) have the same cost for the displacement metric, which means that we can choose at random between the two (we take $\overline{P} = P^{-1}Z$). Following the same rules for the third and fourth implementations, we find that the optimized implementation of circuit (170) is



(179)

The output of the algorithm 1 for the chosen circuit shows that its action is to first remove as much squeezing as possible, and then recenter the envelope. We observe this mechanism between the first and second implementations, where the P operation is reversed and replaced with an equivalent Z operation. The same logic is applied to the third gate implementation, where the addition of squeezing is necessary, but where we can also remove a unit of displacement in the Z direction. In the general case, the computation is more tedious, and it is difficult to apply algorithm 1 analytically. However, the principles remain the same.