Vol.2022-QS-5 No.33
2022/3/25

表面符号の評価に向けたPEPDO ansatzの解析

真鍋 秀隆[1,2,a)]　鈴木 泰成[2,3]　徳永 裕己[2]

概要：ノイズ下における量子誤り訂正符号の性能を調べるのは一般に難しい．本研究では，開放量子多体系向けのテンソルネットワーク ansatz である PEPDO(Projected Entangled Pair Density Operator) を用いた GKP 表面符号のシミュレーションを実行した．その結果，閾値 (threshold) を見るまでの規模・精度には至らなかったものの，PEPO を含む既存手法と比べて高速にシミュレートできることが分かった．これより，ノイズ付き量子回路上における量子アルゴリズムの解析に PEPDO を用いたテンソルネットワーク法が有用である可能性がある．

Analysis of PEPDO ansatz for the evaluation of Surface code

1. はじめに

量子誤り訂正符号とは，量子コンピュータの実行中に発生するエラーを検知しその誤りを訂正するための一連のプロトコルのことを指し，Shor のアルゴリズムのような複雑かつ誤りに敏感なタスクを実行するには量子誤り訂正を用いて誤り耐性量子計算を行う必要がある．これまで様々な量子誤り訂正符号が提唱されてきたが，その中で「性能の良い」，つまりなるべく少数のリソースで誤り耐性量子計算を実行できるような符号を見つけることで，今後の量子コンピュータの実用化を加速させることができる．

量子誤り訂正符号の性能を調べる手法として，古典コンピュータによる数値シミュレーションが多く使われている．例えばスタビライザー符号と呼ばれるクラスの符号に Pauli ノイズがかかるようなケースを考えると，これらは古典コンピュータで効率的にシミュレート可能で [1]，性能の指標である閾値 (threshold) などを推定することができる [2]．ところが，実際に実機でかかるようなノイズ下における量子誤り訂正符号の振る舞いを調べるのは難しい．一般に，外部と相互作用する量子多体系は開放量子多体系 (open quantum many-body system) と呼ばれ，系のサイズに対し取り得る Hilbert 空間は通常の量子多体系と比べても急速に増大してしまう．

[1] 京都大学大学院情報学研究科
[2] NTT コンピュータ&データサイエンス研究所
[3] JST さきがけ
[a)] manabe@acs.i.kyoto-u.ac.jp

量子多体系や開放量子多体系に対する数値計算手法は古くから研究されており，例えば量子モンテカルロ法やテンソルネットワーク法などがよく用いられている．特にテンソルネットワーク法 [3] は量子状態を小さなテンソルの積として直接保持する手法で，近年の高性能計算 (High Performance Computing,HPC) の進化に伴い急速に発展してきた．先行研究 [4] では，量子誤り生成符号の一種である表面符号について，その現実的なノイズ下における性能を PEPO(Projected Entangled Pair Operator) テンソルネットワークを用いて解析し，性能の指標となる閾値 (threshold) が報告されている．

本研究では，光子を用いた量子計算のための量子誤り訂正符号である GKP 表面符号 [5] について，PEPDO(Projected Entangled Pair Density Operator)[6] を用いたシミュレーションを実行した．その結果，閾値 (threshold) を見るまでの規模・精度には至らなかったものの，PEPO を含む既存手法と比べて量子誤り訂正符号の振る舞いを高速にシミュレートできることが分かった．

本研究報告の構成は以下の通りである．2 章でテンソルネットワーク法と GKP 表面符号について紹介する．3 章で PEPDO を用いた GKP 表面符号のシミュレート手法について説明し，4 章でシミュレート結果を示す．5 章で考察と今後の展望について記す．

2. Preliminary

2.1 テンソルネットワーク

テンソルネットワーク形式において，テンソルは小さ

ⓒ 2022 Information Processing Society of Japan

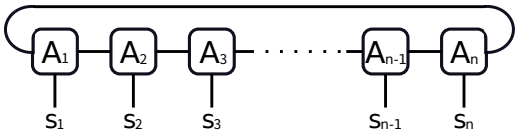

図 1 MPS のダイアグラム表示.

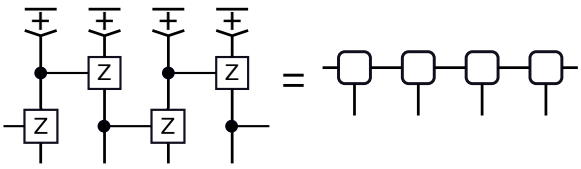

図 2 1 次元グラフ状態の MPS による表示.

なテンソルの積として記述される．例えば n 量子ビット系の波動関数 $|\psi_{s_1,s_2,\cdots s_n}\rangle$ に対し，波動関数を行列積状態 (matrix product state, MPS)[7], [8] で表すと，

$$|\psi_{s_1,s_2,\cdots s_n}\rangle = \sum_{\{s\}} \mathrm{Tr}\left[A_1^{(s_1)} A_2^{(s_2)} \cdots A_n^{(s_n)}\right] |s_1 s_2 \cdots s_n\rangle \tag{1}$$

となる．ただし $A_i^{(s_i)}$ は複素行列で，左右の行列と積が取れるように次元が設定されており，それらをボンド次元という．行列積状態では波動関数という大きなテンソルが小さなテンソル $\{A_i^{(s_i)}\}$ の積で表現されている．これをダイアグラム表示すると図 1 のようになる．

特に系の状態のエンタングルメントが低いときや，特別な構造を持つときには，波動関数をテンソルネットワーク形式により効率的に記述できる可能性がある．例えば測定型量子計算に用いられる一次元グラフ状態は

$$|G\rangle = \left(\prod_{e\in E} CZ_e\right) |+\rangle^{\otimes |V|} \tag{2}$$

と定義されるが，CZ ゲートは

$$CZ_{ij}^{IJ} = \sum_{k=1}^{2} A_{ki}^I B_{kj}^J \tag{3}$$

$$A_{ki}^I = \delta_{ki}^I \tag{4}$$

$$B_{1j}^J = I_j^J, \quad B_{2j}^J = Z_j^J \tag{5}$$

のように分解できるので，結局量子状態全体は図 2 のようにボンド次元が 2 の MPS として完全に記述できる．一量子ビットに対するユニタリ変換は MPS のボンド次元を増やさず，ボンド次元が定数である MPS の局所的な測定値は多項式時間で計算できるため，ここから直ちに一次元グラフ状態を用いた測定型量子計算は古典シミュレート可能であることがわかる．

他にも，例えば量子多体系における低エネルギー状態においてはエンタングルメントが比較的小さいことが期待されるため，テンソルネットワークの ansatz を用いて変分法による最適化を実行し基底状態を求めるアプローチが古くから開発されてきた [3]．また，データサイエンスの分野においては，画像データのピクセル間の相互情報量が距離が長くなると小さくなる局所性を利用して，データをテンソルネットワークで効率的に表現する手法も近年提案されている [9], [10].

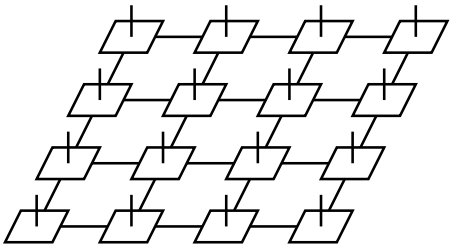

図 3 4×4 の PEPS.

ただし，系の状態がテンソルネットワークで効率的に表現できることと，その物理量の期待値やトレースの値が効率的に計算できることの間には大きなギャップが存在する．例えば図 3 に示す二次元系の状態に対する ansatz である PEPS(Projected Entangled Pair State)[8] では，例え系の状態を PEPS で効率的に表現できたとしても，その局所物理量を計算するのは系のサイズ $n \times n$ に対し指数時間 ($O(n^2 b^n)$) かかる．ただし b はボンド次元である．そのため，2 次元系の量子状態に対しテンソルネットワークによる計算を実行するためには，なんらかの近似を用いて必要なエンタングルメントの情報のみを取り出し，ボンド次元を抑える手法が必要となる [11].

2.1.1 開放量子多体系のためのテンソルネットワーク法

今まで紹介したテンソルネットワークの ansatz は全て純粋状態を表現するためのものであった．開放量子多体系 (open quantum many-body systems) を扱うためには密度演算子を記述する必要があり，一般に純粋状態のシミュレーションよりも難しい.

開放量子多体系の物理量を計算する数値計算手法として，モンテカルロ法により純粋波動関数をサンプリングする手法と，密度演算子そのものをテンソルネットワークとして記述する方法が知られている [6]．モンテカルロサンプリングによる手法では，密度演算子

$$\rho = \sum_i p_i |\psi_i\rangle \langle\psi_i| \tag{6}$$

に対し，純粋状態 $|\psi_i\rangle$ を確率 p_i でサンプリングし $|\psi_i\rangle$ に対する物理量を計算し，それらを平均することで最終的に密度演算子に対する所望の量を計算する．この手法では純粋状態に対する計算手法がそのまま使え，さらに大規模に並列化することができる．他方で，混合状態に対する物理量の計算を正確に行うためには多数のサンプリングが必要となる.

一方, テンソルネットワークを用いる手法では, 図 4, 5 に示す MPO(Matrix Product Operator) や PEPO(Projected

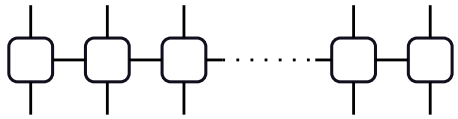

図 4 MPO.

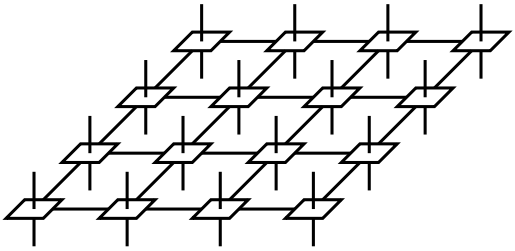

図 5 PEPO.

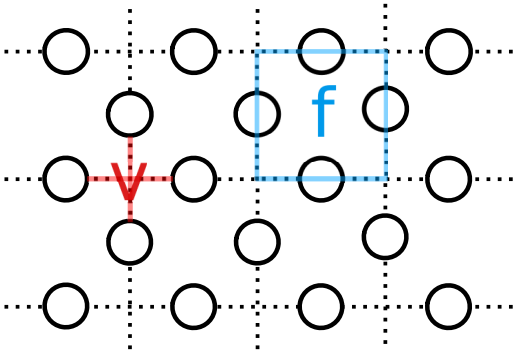

図 6 表面符号.

Entangled Pair Operator) のような ansatz を用いて密度演算子を表現する．モンテカルロ法に比べてサンプリングの手間はかからないが，純粋状態の ansatz と比べてより計算コストがかかってしまう．また，テンソルネットワークの ansatz 自体は密度演算子の正定置性やエルミート性などを保証しない．開放量子多体系の計算手法はモンテカルロ法・テンソルネットワーク法ともに今でも多く研究されている．

2.2 表面符号

表面符号 (surface code)[12] は量子誤り訂正符号の一つで，系にノイズがかかった際に適切に測定・訂正することで系の状態を元の論理状態に戻し，誤り耐性量子計算を実現することができる．

図 6 のような二次元多様体に埋め込まれたグラフ $G(V,E,F)$ を考える．ここで V は頂点，E は辺，F は面の集合である．量子ビットが全ての辺上に配置されているとする．表面符号のスタビライザー演算子を，各頂点 $v \in V$ と面 $f \in F$ に対し，

$$A_f = \prod_{i \in \partial f} Z_i \tag{7}$$

$$B_v = \prod_{j \in \delta v} X_j \tag{8}$$

として定義する．ただし ∂f は面 f に接する 4 つの辺を，δv は頂点 v に接続する 4 つの辺を指す．このとき論理状態を，全てのスタビライザー演算子の同時+1 固有状態

$$\forall f, \quad A_f |\Psi\rangle = |\Psi\rangle \tag{9}$$

$$\forall v, \quad B_v |\Psi\rangle = |\Psi\rangle \tag{10}$$

として定義する．同時+1 固有状態の張る空間は 2 次元複素空間となり，これを 1 論理量子ビット $\{|\bar{0}\rangle, |\bar{1}\rangle\}$ として扱う．

系にノイズがかかった際，各スタビライザー演算子を用いて射影測定 $\{K_0, K_1\}$：

$$K_0 = \frac{I + A_f}{2}, \quad K_1 = \frac{I - A_f}{2} \tag{11}$$

$$K_0 = \frac{I + B_v}{2}, \quad K_1 = \frac{I - B_v}{2} \tag{12}$$

を行うことでエラーを検知できる．これをシンドローム測定という．かかったノイズが作用した量子ビット数がある一定数未満なら，そのノイズを検出することができる．この定数を符号距離という．また，生じたノイズが作用する量子ビット数が符号距離の半分未満であれば，得られたシンドローム値をもとに適切にパウリ演算子を推定する（復号する）ことで，元の論理状態に戻すことができる．

2.2.1 GKP 符号

近年，光子を用いて計算する光量子コンピュータの開発が盛んに行われている [13]．連続量である光子を量子ビットとして用いるには，何らかの方法で連続量を 2 準位系としてエンコードするが，その際 photon loss などのエラーにある程度耐性のあるような符号である必要がある．

GKP 量子ビットは，そのような光子ベースの量子ビットを実現する符号の一つである [14], [15]．GKP 符号において，論理状態は

$$|0\rangle_{\mathrm{GKP}} = \sum_{m=-\infty}^{\infty} |2m\sqrt{\pi}\rangle_q, \tag{13}$$

$$|1\rangle_{\mathrm{GKP}} = \sum_{m=-\infty}^{\infty} |(2m+1)\sqrt{\pi}\rangle_q, \tag{14}$$

として定義される．ところがこれは物理的な状態ではないため，実際にはスクイージングパラメータ κ を用いて

$$|\tilde{0}\rangle_{\mathrm{GKP}} \propto e^{-\kappa^2 \hat{n}} |0\rangle_{\mathrm{GKP}} \tag{15}$$

$$|\tilde{1}\rangle_{\mathrm{GKP}} \propto e^{-\kappa^2 \hat{n}} |1\rangle_{\mathrm{GKP}} \tag{16}$$

のように近似的に論理状態を生成する．GKP 量子ビットに対する各演算は，生成消滅演算子 $\hat{a}^\dagger, \hat{a}$ と数演算子 $\hat{n}$ と，位置，運動量演算子

$$\hat{q} = (\hat{a}^\dagger + \hat{a})/\sqrt{2} \tag{17}$$

$$\hat{p} = i(\hat{a}^\dagger - \hat{a})/\sqrt{2} \tag{18}$$

を用いて

ⓒ 2022 Information Processing Society of Japan

$$\hat{X} = e^{-i\sqrt{\pi}\hat{q}} \tag{19}$$

$$\hat{Z} = e^{i\sqrt{\pi}\hat{q}} \tag{20}$$

$$\hat{H} = e^{i\frac{2}{\pi}\hat{n}} \tag{21}$$

$$\hat{\mathrm{CNOT}}^{j\to k} = e^{-i\hat{q}_j\hat{p}_k} \tag{22}$$

のようにかける．また，論理測定については位置演算子 $\hat{q}$ で測定し，それに mod $\sqrt{\pi}$ をとることで 0,1 状態を区別できる．非クリフォード操作についても GKP 量子ビットを用いて magic state preparation を実現できるため [16]，GKP 量子ビットはユニバーサルな計算が実行できることがわかる．

GKP 量子ビット単体で誤りレートを非常に小さく抑えるのには限界がある．そこで，GKP 量子ビットと表面符号を組み合わせることで，複数個の GKP 量子ビットで一つの論理量子ビットを生成し，誤り率をさらに下げることができる [5]．このような符号を GKP 表面符号という．

2.2.2 閾値定理

量子誤り訂正符号の性能の指標の一つとして閾値 (threshold) が知られている．閾値定理 [17] によると，量子誤り訂正符号に対し何らかの閾値 p_c が存在し，量子ゲートで発生するエラー率が $p < p_c$ であれば多項式時間で任意の精度の量子計算を実行することができる．よって，理論的には閾値 p_c の大きな符号ほど少ないリソースで誤り耐性量子計算を実行することができ，性能の良い符号といえる．

閾値 p_c を計算するのに，例えば depolarizing channel

$$\mathcal{N}_{DP}(\rho) = (1-\epsilon)I + \frac{\epsilon}{3}X\rho X + \frac{\epsilon}{3}Y\rho Y + \frac{\epsilon}{3}Z\rho Z \tag{23}$$

のように各量子ビットに確率的に Pauli ノイズがかかるモデルであれば，Stabilizer 形式により量子誤り訂正のプロセスが効率的にシミュレートでき [1]，得られた論理エラー率から符号距離に対する有限サイズスケーリングを考慮して転移点である閾値を推定することができる．ところが，現実にかかるような一般のノイズに対して閾値を効率的に推定するのは難しい．

2.3 テンソルネットワークを使った表面符号の評価

先行研究 [4] では，効率的にシミュレートできない現実的なノイズ下における表面符号について，PEPO を用いたテンソルネットワークシミュレーションにより閾値を推定している．例えば amplitude dampling ノイズ：

$$\mathcal{N}_{AD}(\rho) = \sum_{i=1}^{2} K_i \rho K_i \tag{24}$$

$$K_1 = |0\rangle\langle 0| + \sqrt{1-\gamma}\,|0\rangle\langle 0| \tag{25}$$

$$K_2 = \gamma\,|0\rangle\langle 1| \tag{26}$$

は Pauli ノイズではなく効率的にシミュレートできない．ノイズ全体の量子チャネルを $\mathcal{N}$，シンドローム値 s に対する表面符号のエラー検知の操作を $\mathcal{R}_s$，復号チャネルを $\mathcal{D}_s$ とおくと，量子誤り訂正のダイナミクスは

$$\mathcal{E} = \mathcal{D}_s \circ \mathcal{R}_s \circ \mathcal{N} \tag{27}$$

となる．この量子チャネルがもとの論理状態をなるべく保存していれば性能の良い符号といえる．つまり，何らかの距離関数 d を用いて

$$d(\mathcal{D}_s \circ \mathcal{R}_s \circ \mathcal{N}, I) \tag{28}$$

を計算できれば，それがそのままシンドローム s が出たときの論理エラーと解釈することができる．Choi-Jamiolkowski 同型対応を用いると，上記の量子チャネルは

$$C_{ij} = \mathrm{Tr}([P_i \otimes P_j][(\mathcal{E}\otimes I)(|\Psi^+\rangle\langle\Psi^+|)]) \tag{29}$$

により完全に記述され，式 (28) は計算できる．ただし $|\Psi^+\rangle$ は bell 状態である．あとは各シンドロームのサンプル重みで平均することで表面符号の論理エラー率が得られる．

このとき，ノイズは CPTP map になっておりノイズがかかった後の状態はもはや純粋状態ではない．そのため，論理エラー率の計算には密度演算子の時間発展を計算する必要があり，ナイーブに密度演算子を行列として保持すると計算量がすぐに爆発してしまう．

ここで，作り方から表面符号の論理状態は近接相互作用のみから作られることを考慮すると，図 5 の PEPO を用いることで系の密度演算子は比較的効率的に表現することが期待できる．先行研究 [4] では，PEPO を用いることで系のサイズ 9×17 の表面符号の論理エラー率を計算できることが報告されている．

3. Method

本章では，開放量子多体系向けのテンソルネットワークの ansatz の一つである PEPDO(Projected Entangled Pair Density Operator)[6] を紹介する．PEPDO を用いることで PEPO よりも高速に計算が出来る場合がある．また，そのノイズ付き量子回路シミュレーションへの応用，特に GKP 表面符号のシミュレートへの応用手法について説明する．

3.1 PEPDO

PEPDO は PEPO と同様に，密度演算子を表すためのテンソネットワークの anstaz である．PEPDO ansatz をダイアグラム記法を用いて示すと図 7 のようになる．PEPO を二重に重ねたような構造だが，上側のテンソルと下側のテンソルは互いに共役転置の関係にある．この構造により，PEPDO により表される密度演算子の正定置性が保証される．また，状態の混合の度合いが比較的弱い場合は内側のボンド次元が比較的小さくなり，PEPO より効率的な記述

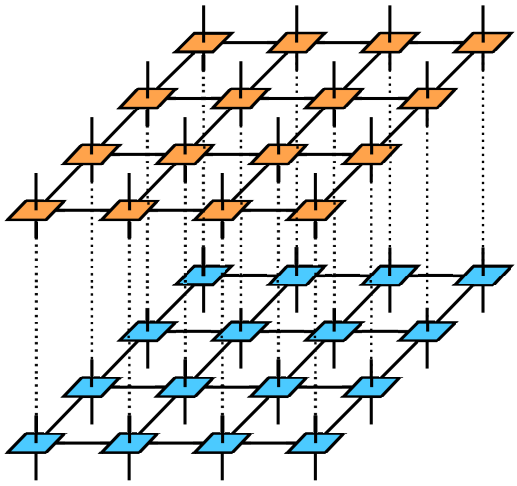

図 7 PEPDO．下側のテンソルは上側のテンソルに共役転置を取ったものである．

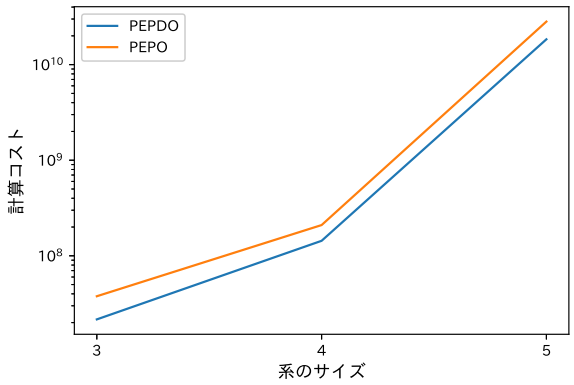

図 8 ボンド次元 4，内側のボンド次元 2 のときの PEPO,PEPDO の計算時間の比較．

ができる．さらに，内側のボンドを先に縮約をとることにより，PEPDO は等価な PEPO にいつでも変換できる．

ただし，PEPDO では状態を正定置性に縛っているために，ボンド次元を上げても PEPO に比べて表現能力が落ちてしまう可能性がある [18]．また，系の大きさを $n \times n$ とし，n を十分大きく取ったとき，状態のトレースを取るための計算量は PEPO と同じ $O(n^2(b^2)^n)$ になってしまう．

3.1.1 計算速度

図 8,9 に，系のサイズを $3\times3, 4\times4, 5\times5$，物理自由度を $d=2$ にした時の PEPO と PEPDO のトレースの時間計算量の見積もりを示す．ただし図 8 ではボンド次元 $b=4$，内側のボンド次元 $i=2$ に設定し，図 9 ではボンド次元 $b=8$，内側のボンド次元 $i=2$ に設定してある．また，PEPO の計算量とは，PEPDO と等価な PEPO についてその縮約をとったときの時間計算量を指す．時間計算量の見積もりについては，kahypar[19] などのテンソルの縮約順を自動的に発見する heuristic なアルゴリズムを用いた．

図から分かる通り，比較的小さな系においては PEPO より PEPDO のほうが時間計算量が小さくできることが分かる．また，ボンド次元と内側のボンド次元の比が大きいほどその差は顕著になる．

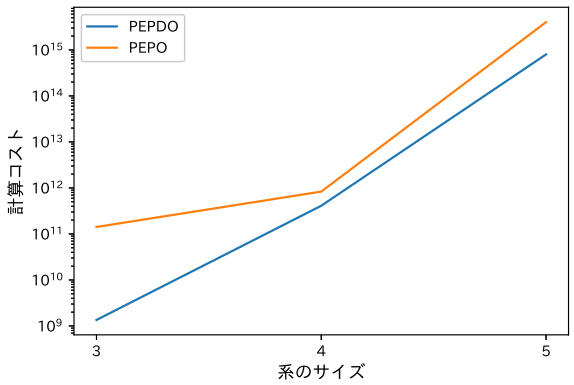

図 9 ボンド次元 8，内側のボンド次元 2 のときの PEPO,PEPDO の計算時間の比較．

3.2 GKP 表面符号の PEPDO によるシミュレーション

本節では，GKP 表面符号を PEPDO ansatz を用いたテンソルネットワークシミュレーション手法について説明する．

3.2.1 GKP 量子ビットの有限次元表現

GKP 量子ビットは連続量なので，そのままでは有限次元のテンソルで表現することができない．そこで今回は，論理 GKP 量子ビットを fock 基底で表現し，その上で有限の fock 基底で打ち切ることを試みる．近似 GKP 量子ビット (15)，(16) を fock 基底で表したときの係数は，$j=0,1$ に対し

$$\langle n|\tilde{j}\rangle \propto e^{-\kappa^2 n}\sum_{m=-\infty}^{\infty}\Psi_n((2m+j)\sqrt{\pi}) \tag{30}$$

となる [15]．ただし Ψ_n はエルミート多項式．この係数の値は n が増えるごとに指数関数的に減少するので，κ が大きい領域では少ない fock 基底でも近似論理 GKP 量子ビットを十分表現できることが期待できる．

3.2.2 GKP 量子ビットに対するノイズモデル

今回，系にかかるノイズとして photon loss：

$$\mathcal{N}(\rho)=\sum_{i=1}^{2}K_i\rho K_i \tag{31}$$

$$K_1=\sqrt{I-\gamma\hat{n}},\quad K_2=\sqrt{\gamma}a \tag{32}$$

を考える．ただし γ はノイズの強さを与えるパラメータである．photon loss は Pauli ノイズとして表現できないため，系のダイナミクスは効率的に追うことができない．ノイズのかかるタイミングは系を論理 0 状態に初期化した直後とする．

3.2.3 スタビライザー測定のテンソルネットワーク表現

有限 fock 次元で打ち切ることにより，近似 GKP 量子

ビットやそこにかかる論理演算子，ノイズモデルも全て有限次元のテンソルとして記述できる．よって後は，これらを組み合わせて GKP 表面符号の量子誤り訂正のプロセスをテンソルネットワーク，特に PEPDO 形式として表現する．

図 10 に，Z スタビライザー測定 (11) を実行し，位置演算子 $\hat{q}$ の固有ベクトル $|v_p\rangle$ に対応する固有値を測定したときのダイナミクスを表すテンソルネットワークを示す．スタビライザー測定は対応する頂点または面に補助量子ビットを置きそこに CNOT をかけて測定するといった操作で実行できるが，それらのダイナミクスは図 10 右の緑色のテンソルネットワークで表現できる．PEPDO に緑色のテンソルネットワークをかけても PEPDO ansatz に留まり続ける上，内側のボンド次元は増加しない．なお，実際にシンドロームをサンプルするためには，上記の操作を行った後，状態のトレースを計算することでそれぞれの固有値を測定する確率を計算する必要がある．

3.2.4 ノイズチャネルのテンソルネットワーク表現

photon loss は Kraus ランクが 2 の CPTP map で，ユニタリ作用素では記述できない．図 11 に，ノイズチャネルをかける操作のテンソルネットワーク表示を示す．図から分かるとおり，ノイズをかけた後でも PEPDO ansatz に留まるが，Kraus ランクの分だけ内側の bond 次元が増加する．

3.2.5 テンソルネットワーク近似

上の手法で GKP 表面符号をシミュレートするとき，各量子ビット間には 2 から 3 回程度，図 10 で表されるテンソルネットワークがかかることになる．このとき，例えば fock 基底を d 個とったとき，各量子ビット間の PEPDO の bond 次元は d^2 から d^3 となり，各テンソルの空間計算量が爆発してテンソルネットワークとして表現することすら難しくなってしまう．

そこで，テンソルネットワークの近似法を利用し，うまく精度を保ちつつボンド次元を落とすことを考える．本研究では，Simple Update(SU)[20] と呼ばれる簡単な手法と Full Enviromnent Truncation(FET)[21] と呼ばれる高級な手法を組み合わせて近似を実行した．

Simple update では，テンソルを更新する際に局所的に特異値分解 (SVD) を実行し，小さな特異値から順に所望のボンド次元まで削減する．図 12 に，Simple Update を PEPDO に適用する手順を示す．実際に行う際は QR 分解を混ぜることで特異値分解のコストを下げる必要がある．

Simple Update は計算量が低い反面，局所最適化しか行わないため局所的に最適な近似が大域的に最適であるとは限らない．Full Environment Truncation では，系の状態の内積が 1 であるという条件のもと，fidelity が最大になるように各ボンドに projection を挿入する．この問題を一般化固有値問題に帰着し iterative に最適化することで大域的に最適となるようにボンド次元を削減することができる．ただし，FET を実行するには full-environment，つまり注目しているボンド以外の全てのテンソルネットワークの縮約をとる必要があるため，計算量的には重い．本研究では，first trial として SU を実行したのち，FET を用いてより正確にボンド次元を切り捨てるという戦略をとっている．

4. Result

PEPDO を用いた GKP 表面符号のシミュレート結果を示す．

4.1 GKP 量子ビットの精度

テンソルネットワークで GKP 量子ビットなどの連続量量子ビットをシミュレートする際に有限の fock 基底で打ち切った．図 13，14 に，$\kappa = 0.3, 0.5$ の近似 GKP 量子ビットに対する，用いる fock 基底の数と fidelity の関係を示す．図からわかるように，$\kappa = 0.5$ のように粗い近似 GKP 量子ビットに対して必要な fock 基底の数は少ない一方，$\kappa = 0.3$ のようにより精度の高い GKP 量子ビットを扱うためには非常に沢山の fock 基底が必要となる．

一方で，そもそも近似 GKP 量子ビットを用いることによるエラーも発生する．図 15 に，$\kappa = 0.5$ の近似 GKP 量子ビットを用いたときの CNOT ゲートのエラー率を示す．fock 基底を増やしていきおよそ 10 個〜20 個程度でほぼ完全に近似 GKP 量子ビットを表せるが，κ を大きく設定することによる CNOT エラーの影響が強すぎるため，小さな外部ノイズに対する GKP 表面符号の性能を調べることが難しくなっている．図 16 に，$\kappa = 0.5$，fock 基底 10 個のときの近似 GKP 量子ビットを Wigner 関数により表示したものを示す．

4.2 シミュレーション結果

今回は，GKP 量子ビットの近似精度と計算量の兼ね合いから，photon loss に対する閾値を見るまでの精度・サンプル数に達することが困難であったため，単にノイズの影響下における GKP 表面符号のシンドローム値がどのように分布するかを見ることにする．

図 17 に，符号距離 3 の rotated-layout の GKP 表面符号の PEPDO シミュレーションを実行した際の，1 ラウンド後のシンドローム値の分布を示す．ただし，$\kappa = 0.5$ の近似 GKP 量子ビットを用い，外部ノイズとして $\gamma = 0.01$ の photon loss ノイズをかけ，fock 基底は 10 個で打ち切っている．また，テンソルネットワーク法において近似を適用しているが，今回のシミュレーションでは系の fidelity が十分高くなる（≥ 0.99）ようにボンド次元などを設定している．

全くエラーの乗らない状況では測定結果はすべて 0 となる．今回のシミュレーション結果をみると確かに測定結果

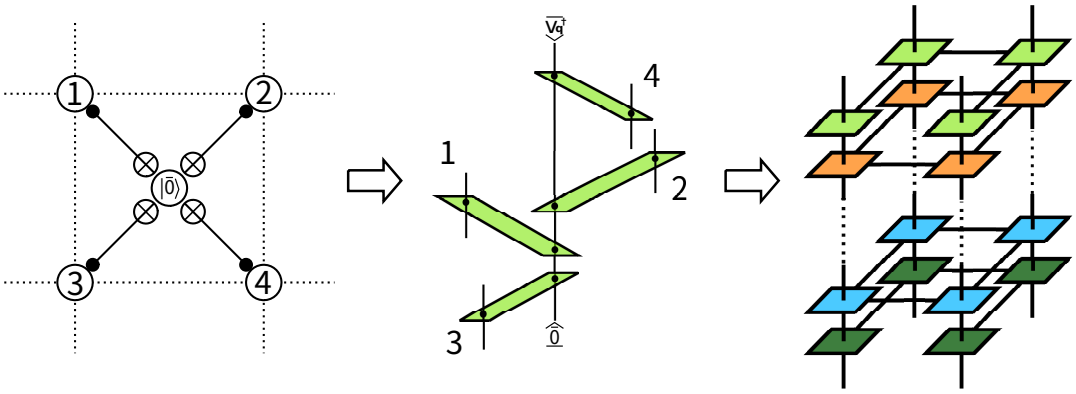

図 10 Z スタビライザー測定のテンソルネットワークによる表示.

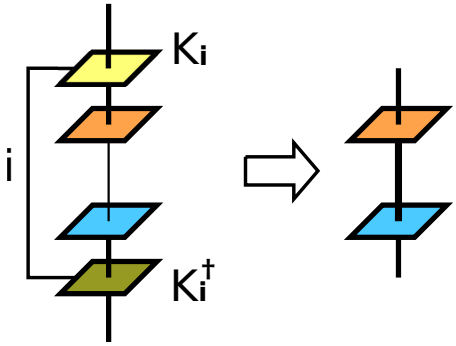

図 11 ノイズチャネルのテンソルネットワークによる表示.

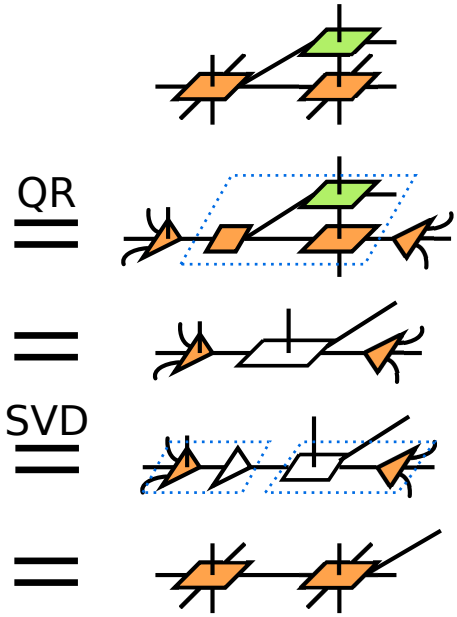

図 12 Simple Update によるボンド次元の切り捨て.

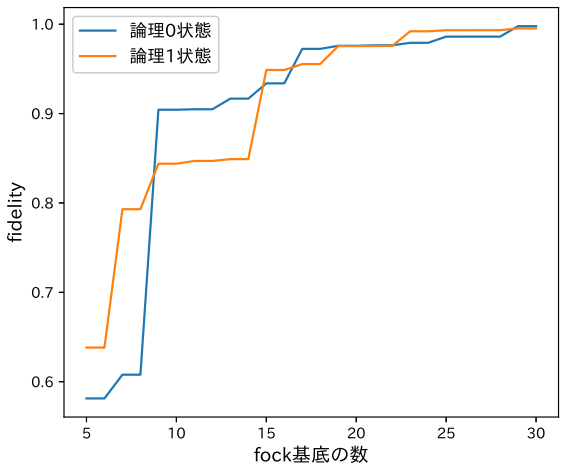

図 13 $\kappa = 0.3$ のときの用いる fock 基底の数と fidelity.

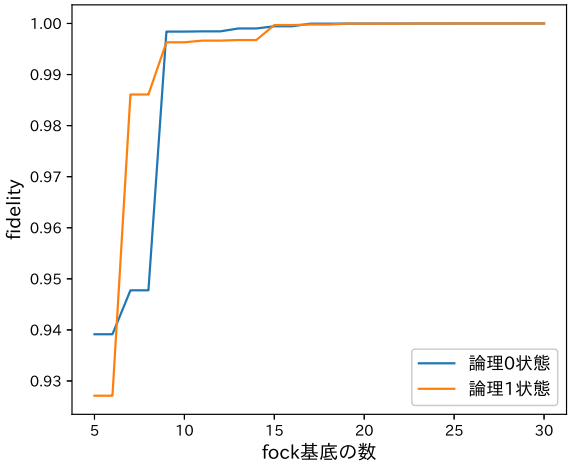

図 14 $\kappa = 0.5$ のときの用いる fock 基底の数と fidelity.

が 0 のほうに分布が寄っているが，CNOT ゲートのエラー率が高いことでかなり測定結果にエラーが乗ってしまい，測定結果が 1 となるシンドローム測定が多くなってしまっている.

4.3 シミュレーションにかかる計算時間

最後に，上記の GKP 表面符号シミュレーションにかかる時間について，図 18 に，テンソルネットワーク法 (PEPDO) を用いたときの計算時間と，状態ベクトルを用いた際の計算時間の見積もりを比較している．ただし，図中の状態ベクトル法 1 とは，系の混合状態を分解し 2^9 個の GKP 量子

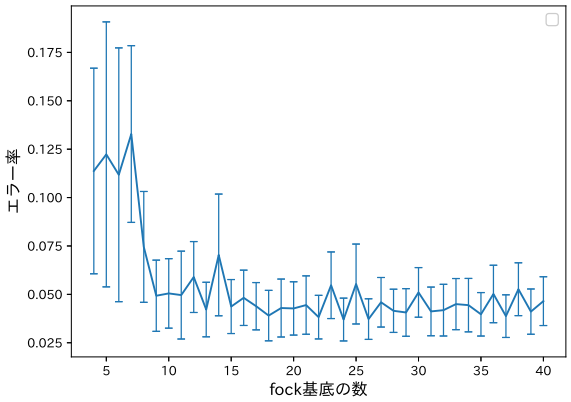

図 15 $\kappa = 0.5$ のときの用いる fock 基底の数と CNOT ゲートの精度.

ビットの状態ベクトルの和として計算する手法を意味し，状態ベクトル法 2 とは，ノイズレベルが十分小さいと仮定し，系の混合状態を 10 個の GKP 量子ビットの状態ベクトルの和として計算する手法を意味する．状態ベクトル法のグラフが途中で切れているのは，見積もりに Qulacs のベンチマークを用いた都合上，これ以上の fock 基底に対するベンチマークが存在していなかったことに由来する．

図から分かるとおり，状態ベクトル法では Fock 基底の数が増えるにつれて急激に計算時間が増加する一方，テンソルネットワーク法を用いた際はその増加は緩やかに抑えられている．ここから，GKP 表面符号のシミュレーションにおいて，GKP 量子ビット間のエンタングルメントが低い箇所を自動的に発見し切り捨てるテンソルネットワーク法の近似手法が効果的に機能していることがわかる．

5. 考察

本研究では，PEPDO を用いた小〜中規模のノイズ有り量子回路の古典シミュレーションを提案し，特に GKP 表面符号のシミュレーションに適用し数値実験を行った．その結果，threshold を見るまでの規模・精度には至らなかったものの，従来の手法と比べて非常に高速にシミュレートできることがわかった．

今後の課題としては，まず，GKP 表面符号について，より効率の良い GKP 量子ビットの有限次元表現を探すことが挙げられる．特に κ が非常に小さい領域においては，fock 基底で表すよりもむしろ論理状態とそれらに生成消滅演算子がかかった状態を直に基底として保持するほうが，必要な次元が低くなり threshold が見える領域までシミュレートできる可能性がある．

また，PEPDO を用いたテンソルネットワーク法を別の量子アルゴリズムの解析に用いることもできる．NISQ 時代の量子コンピュータではノイズの影響が避けられない．古典コンピュータによるシミュレーションにより，実機で発生するノイズがどの程度実行結果に影響を与えるかを調べ，さらにそれらノイズを緩和する新たな手がかりを得られるかもしれない．

表面符号の閾値を調べるという目的においては，PEPO や PEPDO など密度演算子を表すテンソルネットワーク法を用いるのではなく，ノイズに関してモンテカルロ法を用いて純粋状態をシミュレートするほうが計算量的に効率が良い可能性がある．純粋状態のシミュレートにもテンソルネットワーク法は使えるので，今後はテンソルネットワーク法とモンテカルロ法を組み合わせたノイズ付き量子回路シミュレーションの有効性について調査する必要がある．

謝辞 本研究は JST さきがけ（助成番号：No. JPMJPR1916）、内閣府ムーンショット（助成番号：No. JPMJMS2061）の助成の元で行いました。

参考文献

[1] Gottesman, D.: Stabilizer Codes and Quantum Error Correction, PhD Thesis (1997).
[2] Fowler, A. G., Mariantoni, M., Martinis, J. M. and Cleland, A. N.: Surface codes: Towards practical large-scale quantum computation, *Phys. Rev. A*, Vol. 86, p. 032324 (2012).
[3] Bridgeman, J. C. and Chubb, C. T.: Hand-waving and interpretive dance: an introductory course on tensor networks, *Journal of Physics A: Mathematical and Theoretical*, Vol. 50, No. 22, p. 223001 (2017).
[4] Darmawan, A. and Poulin, D.: Tensor-Network Simulations of the Surface Code under Realistic Noise, *Physical Review Letters*, Vol. 119 (2016).
[5] Noh, K. and Chamberland, C.: Fault-tolerant bosonic quantum error correction with the surface–Gottesman-Kitaev-Preskill code, *Phys. Rev. A*, Vol. 101, p. 012316 (2020).
[6] Weimer, H., Kshetrimayum, A. and Orús, R.: Simulation methods for open quantum many-body systems, *Rev. Mod. Phys.*, Vol. 93, p. 015008 (2021).
[7] Perez-Garcia, D., Verstraete, F., Wolf, M. M. and Cirac, J. I.: Matrix Product State Representations, Vol. 7, No. 5, p. 401–430 (2007).
[8] : A practical introduction to tensor networks: Matrix product states and projected entangled pair states, *Annals of Physics*, Vol. 349, pp. 117–158 (2014).
[9] Stoudenmire, E. and Schwab, D. J.: Supervised Learning with Tensor Networks, *Advances in Neural Information Processing Systems*, Vol. 29, Curran Associates, Inc., pp. 4799–4807 (2016).
[10] Convy, I., Huggins, W., Liao, H. and Whaley, K. B.: Mutual information scaling for tensor network machine learning, *Machine Learning: Science and Technology*, Vol. 3, No. 1, p. 015017 (2022).
[11] Ran, S.-J., Tirrito, E., Peng, C., Chen, X., Tagliacozzo, L., Su, G. and Lewenstein, M.: Tensor Network Contractions, *Lecture Notes in Physics* (2020).
[12] Fujii, K.: Quantum Computation with Topological Codes: from qubit to topological fault-tolerance (2015).
[13] Fukui, K. and Takeda, S.: Building a large-scale quantum computer with continuous-variable optical technologies, Vol. 55, No. 1, p. 012001 (2022).
[14] Grimsmo, A. L. and Puri, S.: Quantum Error Correction

ⓒ 2022 Information Processing Society of Japan

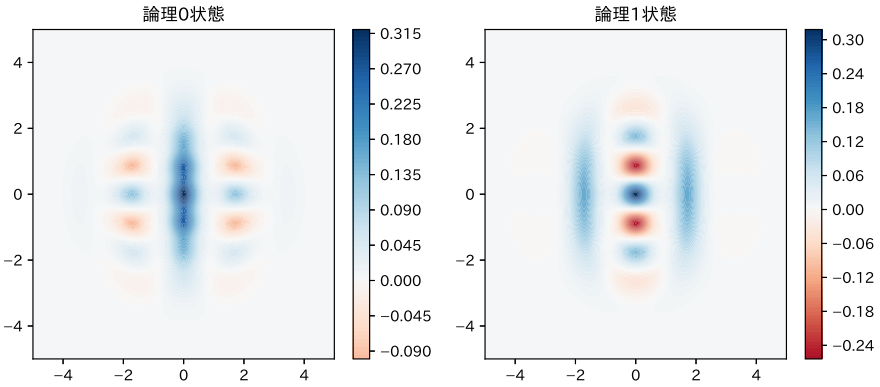

図 16 $\kappa = 0.5$, fock 基底 10 個のときの近似 GKP 量子ビットの Wigner 表示.

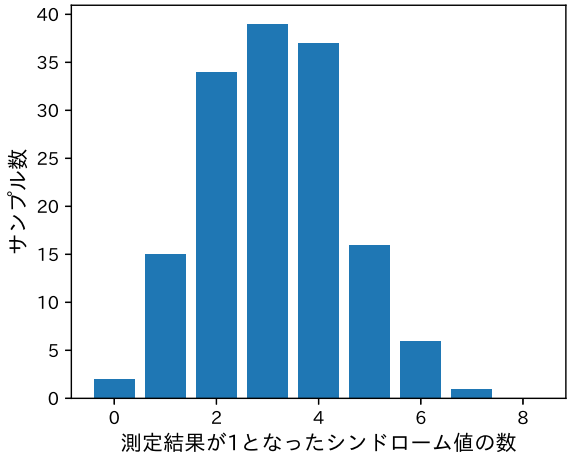

図 17 $\kappa = 0.5$, $\gamma = 0.01$ の GKP 表面符号の 1 ラウンドのシンドローム値の分布.

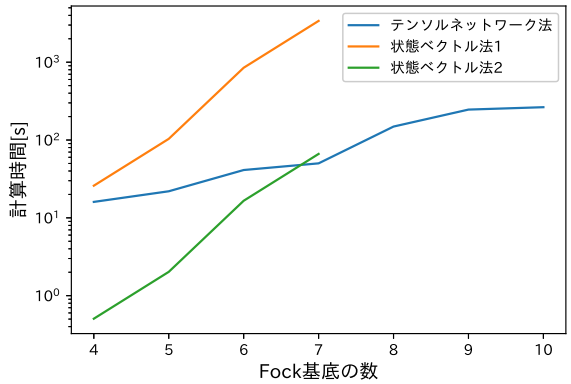

図 18 $\kappa = 0.5$, $\gamma = 0.01$, fock 基底 10 個の GKP 表面符号の 1 ラウンドのシミュレーション時間の比較.

with the Gottesman-Kitaev-Preskill Code, *PRX Quantum*, Vol. 2, No. 2 (2021).

[15] Terhal, B. M., Conrad, J. and Vuillot, C.: Towards scalable bosonic quantum error correction, *Quantum Science and Technology*, Vol. 5, No. 4, p. 043001 (2020).

[16] Baragiola, B. Q., Pantaleoni, G., Alexander, R. N., Karanjai, A. and Menicucci, N. C.: All-Gaussian Universality and Fault Tolerance with the Gottesman-Kitaev-Preskill Code, *Phys. Rev. Lett.*, Vol. 123, p. 200502 (2019).

[17] Aharonov, D. and Ben-Or, M.: Fault-Tolerant Quantum Computation With Constant Error Rate, Vol. 38 (1999).

[18] las Cuevas, G. D., Schuch, N., Pérez-García, D. and Cirac, J. I.: Purifications of multipartite states: limitations and constructive methods, *New Journal of Physics*, Vol. 15, No. 12, p. 123021 (2013).

[19] Gray, J. and Kourtis, S.: Hyper-optimized tensor network contraction, *Quantum*, Vol. 5, p. 410 (2021).

[20] Pang, Y., Hao, T., Dugad, A., Zhou, Y. and Solomonik, E.: Efficient 2D Tensor Network Simulation of Quantum Systems (2020).

[21] Evenbly, G.: Gauge fixing, canonical forms, and optimal truncations in tensor networks with closed loops, *Phys. Rev. B*, Vol. 98, p. 085155 (2018).

ⓒ 2022 Information Processing Society of Japan