# A Variational Quantum Attack for AES-like Symmetric Cryptography

**Preprint** · May 2022



**4 authors**, including:

L. Hanzo
University of Southampton
**2,859** PUBLICATIONS **61,618** CITATIONS

SEE PROFILE

# A Variational Quantum Attack for AES-like Symmetric Cryptography

ZeGuo Wang[1,2], ShiJie Wei[2,1*], Gui-Lu Long[1,2,3,4*] & Lajos Hanzo[5]

[1]*State Key Laboratory of Low-Dimensional Quantum Physics and Department of Physics, Tsinghua University, Beijing* 100084*, China;*

[2]*Beijing Academy of Quantum Information Sciences, Beijing* 100193*, China;*

[3]*Beijing National Research Center for Information Science and Technology and School of Information Tsinghua University, Beijing* 100084*, China;*

[4]*Frontier Science Center for Quantum Information, Beijing* 100193*, China;*

[5]*Department of Electronics and Computer Science (ECS), Southampton SO17 1BJ, United Kingdom*

**Abstract** We propose a variational quantum attack algorithm (VQAA) for classical AES-like symmetric cryptography, as exemplified the simplified-data encryption standard (S-DES). In the VQAA, the known ciphertext is encoded as the ground state of a Hamiltonian that is constructed through a regular graph, and the ground state can be found using a variational approach. We designed the ansatz and cost function for the S-DES's variational quantum attack. It is surprising that sometimes the VQAA is even faster than Grover's algorithm as demonstrated by our simulation results. The relationships of the entanglement entropy, concurrence and the cost function are investigated, which indicate that entanglement plays a crucial role in the speedup.

**Keywords** S-DES, VQA, ansatz, cost function, optimization

## 1 Introduction

Security of information plays an important role in defense, in the economy and in people's livelihood [1,2]. At the time of writing typically asymmetric cryptography, such as RSA [3], is used for transmitting the secret key and symmetric cryptography, such as the Advanced Encryption Standard (AES) [4], is employed

* Corresponding author (email: weisj@baqis.ac.cn, gllong@tsinghua.edu.cn)

for encrypting data. With the development of quantum computers [5–8], more and more attention is paid to the security analysis of classical cryptography under quantum attacks.

Shor's algorithm [9] is capable of decrypting RSA cryptography in polynomial time [10], which seriously threatens the security of asymmetric cryptography. For symmetric cryptography, Grover's algorithm [11–13] can find the key in a set having $N$ entries by only evaluating on the order of $\sqrt{N}$ entries. In Ref [14–17], the efficient quantum implementations of AES and Data Encryption Standard (DES) are proposed by relying on less quantum resources, such as qubits, quantum gates and circuit depths.

At the time of writing we are in the noisy intermediate-scale quantum (NISQ) era [18] when quantum computing systems are characterized by low number of qubits, low fidelity and shallow quantum circuits. Under these restrictions, various classical-quantum hybrid algorithms, including the variational quantum algorithm (VQA) [19,20], and the Quantum Approximate Optimization Algorithm (QAOA) [21] have been proposed. These hybrid algorithms have significant advantages in solving combinatorial optimization [22] and Hamiltonian ground state problems [23]. Briefly, VQA has found applications both in quantum chemistry [24, 25], as well as in quantum machine learning [26–28], and in quantum finance [29, 31] etc. However, there is a paucity of research on the employment of VQA in classical symmetric cryptography attacks. We fill this knowledge-gap by conceiving a quantum attack scheme based on VQA for AES-like symmetric cryptography. In our design, the parameterized quantum circuit (PQC) operates on the key space, and the cost function is designed according to the known ciphertext. We will show by our simulations that the VQAA on average uses the same order of search-space queries as Grover's algorithm. However in some cases, it is even faster than Grover's algorithm, which is really surprising. We also investigated the relationship between the entanglement entropy, concurrence and the cost function, and it was found that the speedup attained is related to the entropy, which is not unexpected, because the entropy by definition represents the specific degree of surprise upon revealing a particular problem solution/outcome.

The paper is organized as follow. Section 2 briefly reviews the structure of symmetric cryptography and the S-DES technique. Section 3 describes the VQAA process in our work. In this part, the cost function, the ansatzes, and the classical optimization algorithms are designed. Then, the optimization results and the entanglement entropy, as well as the associated concurrence are presented and analyzed. Finally, a summary is provided.

## 2 Symmetric cryptography

The symmetric cryptography, including AES and DES, encrypts and decrypts the data using the same key. For AES [4], there are four operations, AddRoundKey, SubBytes, ShiftRows and MixColumns, to replace and substitute the data. For DES [32], the encryption mainly includes the IP operation, the $f$ function and the SWAP operation. We use the S-DES as an example for characterizing the VQAA for simplicity. The specific process of S-DES is as follows.

### 2.1 S-DES

The input of S-DES is an 8-bit plaintext and a 10-bit key. The output is an 8-bit ciphertext. We can simply express the encryption process as a composite of functions

$$\text{Ciphertext} = \text{IP}^{-1} \circ f_{K_2} \circ \text{SW} \circ f_{K_1} \circ \text{IP}[\text{Plaintext}], \tag{1}$$

where IP is the initial replacement, the function $f_K$ includes the replacement and substitution operations, SW is a swap function and $K_1, K_2$ are the sub-keys in the first and second round of encryption, respectively. The decryption represents the inversion of encryption in the form of

$$\text{Plaintext} = \text{IP}^{-1} \circ f_{K_1} \circ \text{SW} \circ f_{K_2} \circ \text{IP}[\text{Ciphertext}]. \tag{2}$$

The encryption process is shown in Figure 1 and the associated details are given in Appendix A.

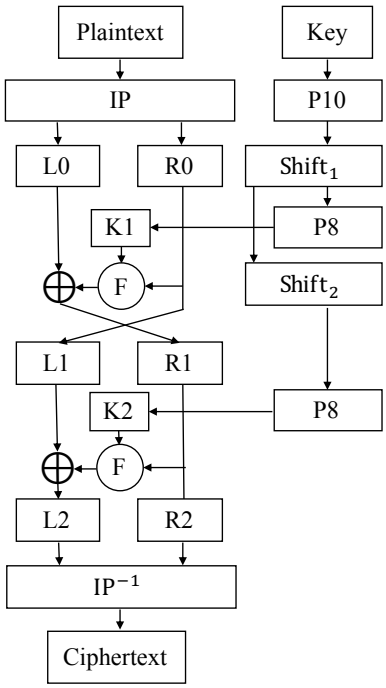



**Figure 1** The encryption process of S-DES.

## 3 VQAA for symmetric cryptography

The main idea of our VQAA is shown in Figure 2. Based on a pair of known ciphertext and plaintext, the associated Hamiltonian is designed, whose ground state is the ciphertext. The parameterized quantum circuit gives a linear combination of all possible keys. After the symmetric cryptography operations, we have a linear combination of all the ciphertext corresponding to the known plaintext, associated with all possible keys. Then the variational process is started to find the Hamiltonian associated with the lowest

energy, which contains the corresponding key.

We now demonstrate how to carry out a quantum attack on the S-DES using the VQAA as an example. The plaintext and ciphertext are represented by 8-bit quantum states. Firstly, we construct the Hamiltonian whose ground state corresponds to the ciphertext. The detailed construction of the Hamiltonian is assumed to be given. Secondly, the key space is encoded into an adjustable quantum state by a parameterized quantum circuit which is also known as ansatz. Next, the output of the parameterized quantum circuit is used as a key to encrypt the known plaintext based on the S-DES and then the superposition of ciphertexts is obtained. Finally, we measure the superposition of ciphertexts and forward the result to the classical optimization algorithm. Using the optimization algorithm, we adjust the input parameters of the parameterized quantum circuit to arrange for the superposition ciphertext state to have a considerable overlap with the known ciphertext. When the result of measurement is the known ciphertext, the key space also collapses to the required key state. In the VQAA of S-DES, the key space and the data space contain 10-qubit and 8-qubit strings respectively, and the 'Symmetric Cryptography' block of Figure 2 is substituted by the 'S-DES' module whose quantum implementation can be found in Ref [17].

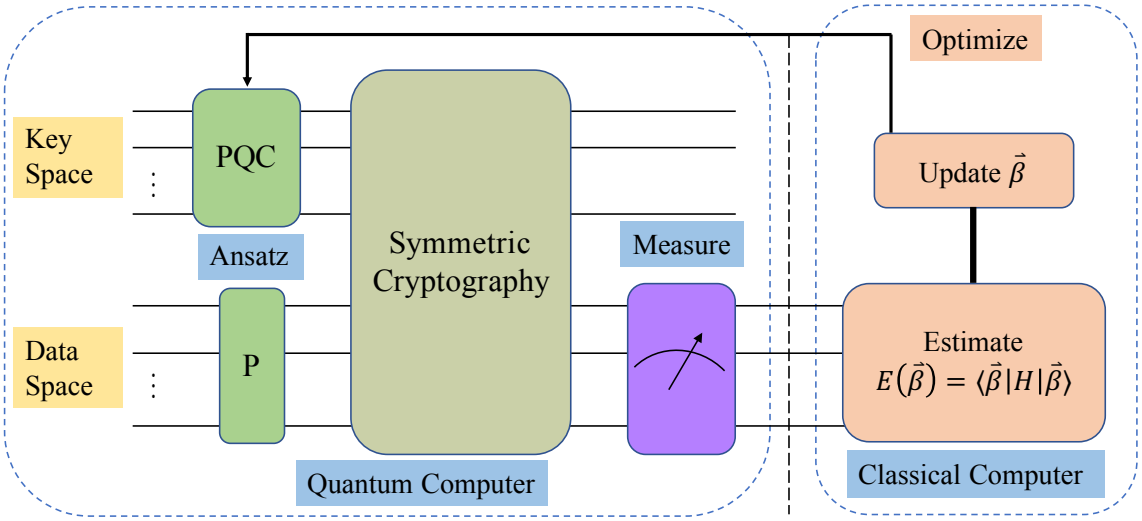



**Figure 2** Schematic diagram of the quantum attack on S-DES using VQAs, where $\vec{\beta}$ represents the parameters of PQC.

### 3.1 The construction of cost function

In order to encode the known ciphertext into a Hamiltonian ground state, we use each bit as a node to construct regular graphs. For an 8-node network, we can construct an $n$-regular ($n$=1,2,...,7) graph. The value of the $i$-th node is denoted by $V(i)$, which is the value of the $i$-th bit. If there is a pair of nodes $i$ and $j$ in the graph that are connected, we add the term $w_{ij}Z_iZ_j$ into the Hamiltonian, where $Z$ is the Pauli-$Z$ operator, $i, j \in \{0, 1, ..., 7\}$. The coefficient $w_{ij}$ is determined by $V(i)$ and $V(j)$, which takes the

form

$$w_{ij} = \begin{cases} 1 & \text{if } V(i) \neq V(j), \\ -1 & \text{if } V(i) = V(j). \end{cases} \tag{3}$$

Additionally, all of the single-qubit operators $\sum_{i=0}^{7} t_i Z_i$ are added into the Hamiltonian, where $t_i$ is defined as

$$t_i = \begin{cases} 0.5 & \text{if } V(i) = 1, \\ -0.5 & \text{if } V(i) = 0. \end{cases} \tag{4}$$

Then the energy level of these seven Hamiltonians is analyzed. The results are shown in Table 1. The 'Ratio' represents the ratio of the energy level differences between the ground state and the first excited state to the total dynamical range of the energy levels.

**Table 1** The energy range of seven Hamiltonians, where 'reg' represents regular.

| | 1-reg | 2-reg | 3-reg | 4-reg | 5-reg | 6-reg | 7-reg |
|---|---|---|---|---|---|---|---|
| Ground energy | -8 | -12 | -16 | -20 | -24 | -28 | -32 |
| Highest energy | 4 | 8 | 8 | 9 | 12 | 8 | 4 |
| The first excited energy | -6 | -7 | -9 | -12 | -16 | -20 | -24 |
| Ratio | 0.1667 | 0.2500 | 0.2917 | 0.2759 | 0.2222 | 0.2222 | 0.2222 |

Instinctively, the higher the ratio, the easier it is to distinguish the global minimum. Because a large ratio implies that the gradient changes rapidly in the vicinity of the minimum value, the optimization path tends to lean more toward the neighborhood of the minimum. The simulations in Appendix B prove our prediction, when $n = 3$, the optimization works best. The 3-regular graph we use is shown in Figure 3. The number of items in the Hamiltonian is $3 \times 8/2 + 8 = 20$, which is a polynomial whose order is determinated by the number of nodes.

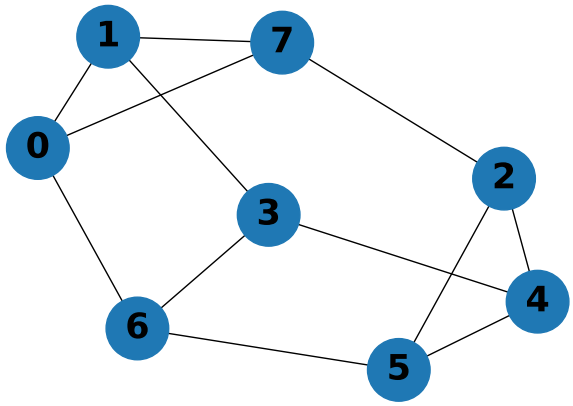



**Figure 3** The 3-regular graph used in Hamiltonian construction.

The Hamiltonian defined in Figure 3 is

$$\begin{aligned} H = {} & w_{01}Z_0Z_1 + w_{06}Z_0Z_6 + w_{07}Z_0Z_7 + w_{13}Z_1Z_3 + w_{17}Z_1Z_7 \\ & + w_{24}Z_2Z_4 + w_{25}Z_2Z_5 + w_{27}Z_2Z_7 + w_{34}Z_3Z_4 + w_{36}Z_3Z_6 \\ & + w_{45}Z_4Z_5 + w_{56}Z_5Z_6 + \sum_{i=0}^{7} t_i Z_i. \end{aligned} \tag{5}$$

As described earlier, $w_{ij}$ and $t_i$ depend on the ciphertext. The cost function $E(\vec{\beta})$ is the expectation of the Hamiltonian,

$$E(\vec{\beta}) = \langle \vec{\beta} | \, H \, | \vec{\beta} \rangle \, , \tag{6}$$

where $|\vec{\beta}\rangle$ is the superposition of ciphertext state.

### 3.2 Ansatz

We have chosen six ansatzes in this work. The first two are denoted as the Y-Cx model, which are shown in Figure 4. The next two are denoted as the Y-Cy model which are shown in Figure 5, while the last two are denoted as the Y-Cz model which are shown in Figure 6. The six ansatzes can be divided into two categories, as shown in Figures 4, 5, 6(a) and Figures 4, 5, 6(b) respectively. They are denoted as A-ansatz and B-ansatz. Their differences are that there is a controlled $X$, $Y$, $Z$ gate at the right-most edge of Figure 4, 5, 6(a).

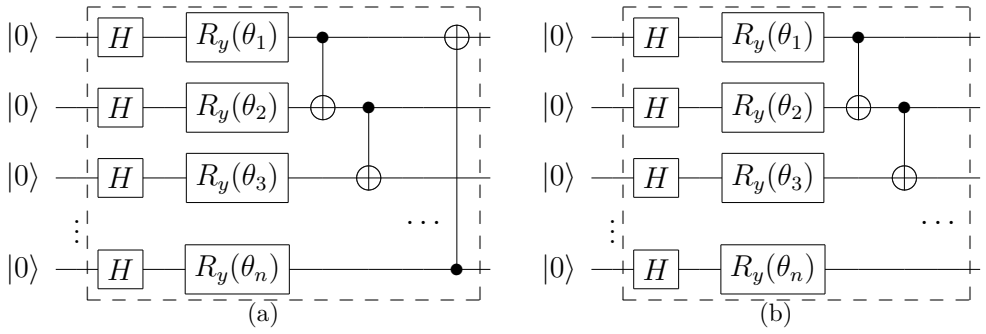



**Figure 4** The dashed box indicates a single Y-Cx circuit layer that can be repeated.

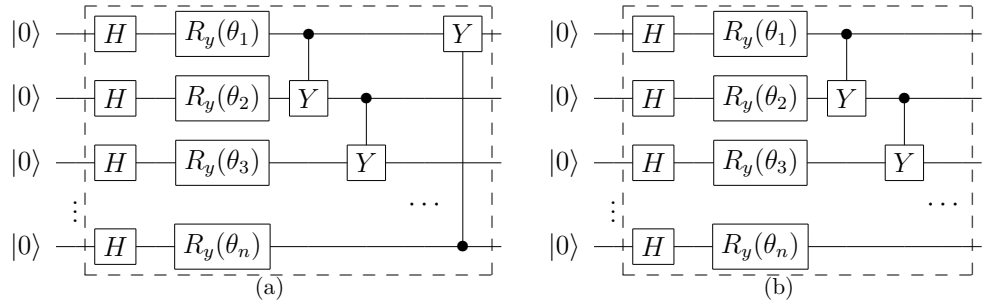



**Figure 5** The dashed box indicates a single Y-Cy circuit layer that can be repeated. Gate Y represents a Pauli-Y gate.

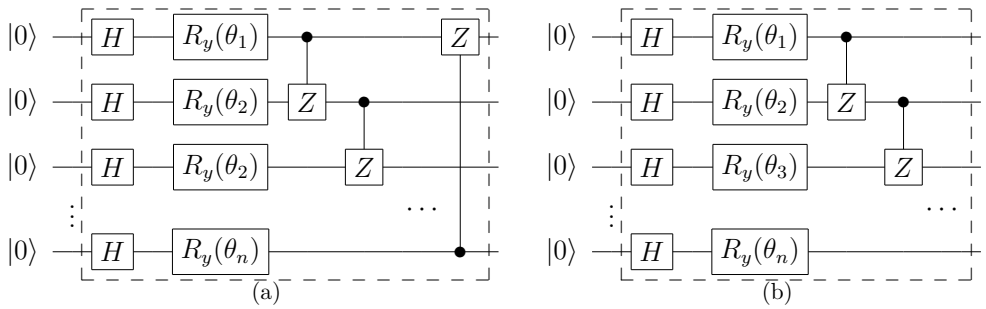



**Figure 6** The dashed box indicates a single Y-Cz circuit layer that can be repeated. Gate Z represents a Pauli-Z gate.

For an $n$-qubit system, a 1-layer ansatz requires $n$ parameters and their circuit depths are $n+2$ for an A-ansatz, and $n+1$ for a B-ansatz. The key space in S-DES contains 10 qubits, so it requires 10 parameters as its input for a 1-layer ansatz and its circuit depth is 12 or 11 for A or B respectively, which is far lower than the depth of S-DES's quantum implementation circuit. The initial state is prepared as the uniform superposition state.

### 3.3 Classical optimization algorithms

We use two methods to optimize the parameters, namely the Gradient Descent method and the Nelder-Mead (N-M) method [33] whose pseudo-codes and hyper-parameters are given in Appendix C. The cut-off condition is set as $-9$, which is the first excited energy. When the expectation of Hamiltonian is less than $-9$, the superposition cipher state has a large overlap with the known ciphertext (the ground state). When the measurement result is the known ciphertext, the key space collapses to the desired key state. Additionally, we have to set the restart condition for both two optimization algorithms. Explicitly, the VQAA will be restarted when the norm of the gradient is lower than 0.8 in the Gradient Descent method and when we have $f(x_N) - f(x_0) < 0.15$ for the N-M method. Furthermore, $f(x_N)$ and $f(x_0)$ are the maximum and the minimum expectation of the Hamiltonian from the $N+1$ points, respectively.

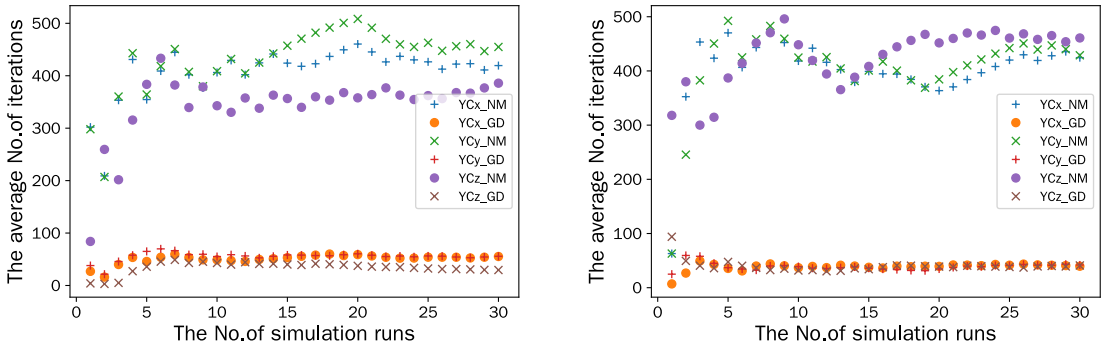
## 4 The optimization results

We now characterize the performance of the VQAA for the different combination of ansatzes and optimization algorithms using numerical simulations. The relationships between the cost function and the entanglement entropy, as well as the concurrence are presented.

### 4.1 The number of iterations

The hyper-parameters of the classical optimization algorithms are adjusted to the optimal value for the different ansatzes. The initial input parameters are the same in each simulation in which we use different classical optimization methods in order to search for the ground state. In each simulation, the key and plaintext are chosen randomly, at the same time and the ciphertext is determined. The range of the key is $[0, 2^{10}-1]$, and the range of both the plaintext and of the ciphertext are $[0, 2^8-1]$. All these values have to

be converted into binary strings and then prepared as quantum states. We performed thirty simulations, each with six experiments corresponding to a combination of three ansatzes and two classical optimization algorithms. We terminated the process if we failed to find key after $2^{10}$ measurements. We portray the average number of iterations in Figure 7(a) and Figure 7(b) for A-ansatz and B-ansatz, respectively. For the sake of reference, we gave all the number of iterations averaged over 1 to 30 simulations.




(a) A-ansatz: The average number of iterations over the number of simulations

(b) B-ansatz: The average number of estimations over the number of simulations

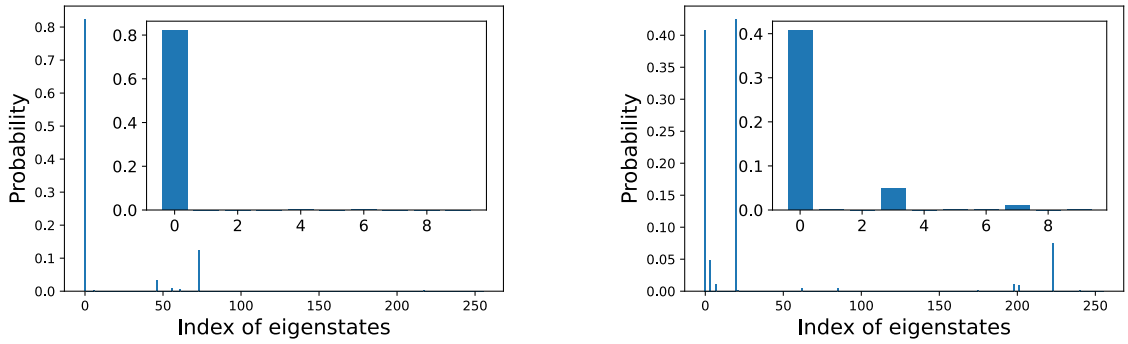
**Figure 7** The simulation results

**Table 2** The number of iterations in the A-ansatz (A) and B-ansatz (B), respectively.

| | | N-M | | | GD | | |
|---|---|---|---|---|---|---|---|
| | | Maximum | Minimum | Average | Maximum | Minimum | Average |
| A | Y-Cx | 694 | 28 | 419.40 | 94 | 2 | 55.27 |
| | Y-Cy | 691 | 28 | 454.93 | 94 | 3 | 55.67 |
| | Y-Cz | 692 | 21 | 385.83 | 94 | 2 | 29.50 |
| B | Y-Cx | 705 | 63 | 424.53 | 94 | 4 | 39.63 |
| | Y-Cy | 687 | 63 | 428.83 | 94 | 3 | 41.93 |
| | Y-Cz | 700 | 19 | 460.83 | 94 | 2 | 41.03 |

In Table 2, we have given the maximum, the minimum and the average number of iterations required for the six-ansatz and two classical optimization algorithm. It is apparent that the results for those using the Gradient Descent method are much better than those using the N-M method. The Gradient Descent method usually takes 40-50 iterations to obtain the key, whereas the N-M method takes more than 400 iterations. For the six-ansatz scenario, the Y-Cz ansatz's results are better.

When the process is convergent, the occupation probability of the target state is the highest. Figure 8 presents the probability distribution of the eigenstates under the Y-Cz(A) ansatz, when the cost function value is lower than the threshold -9. The $x$-axis represents the eigenstates, which are ordered from the ground state to the highest eigenstate. The $y$-axis represents the corresponding probability. The probability of ground states in Figure 8 is 0.82 and 0.41, respectively.




(a) (b)

**Figure 8** The probability distribution of eigenstates for the Y-Cz(A) ansatz

### 4.2 Convergence vs entanglement entropy/concurrence

Entanglement entropy and concurrence constitute a pair of popular entanglement metrics. In our work, the relationships between the cost function and the entanglement entropy as well as, concurrence are investigated. For a pure state $\rho_{AB} = |\Psi\rangle \langle \Psi|_{AB}$, the entanglement entropy is defined as follows,

$$\mathcal{S}(\rho_A) = -\operatorname{Tr}[\rho_A \log \rho_A] = -\operatorname{Tr}[\rho_B \log \rho_B] = \mathcal{S}(\rho_B), \tag{7}$$

while the concurrence is defined as:

$$\mathcal{C}_A(\rho) = \sqrt{2\left(1 - \operatorname{Tr}\rho_A^2\right)}, \tag{8}$$

where $\rho_A = \operatorname{Tr}_B(\rho_{AB})$ and $\rho_B = \operatorname{Tr}_A(\rho_{AB})$ represent the reduced density matrices for each partition. We opted for the equal partition, where partition A represents the first 5 qubits and partition B the next 5 qubits, when calculating the entropy/concurrence.

The results in Figure 9 are generated from the Y-Cz (A-ansatz) and the Gradient Descent method. There are four scenarios in Figure 9:

a) the process converges monotonically;

b) both EE & C first increase, and then gradually converge towards 0. This case represents a process where the search path is initially close to the target, but there is some initial divergence before convergence;

c) the sudden spike represents a restart. As we recall, we have set up a limit for the gradient to be below at which the iterations eventually are curtailed and then restarted again with a new set of parameters. Nevertheless, the process of searching for the target converges, and becomes successful;

d) the process does not converge at all even after several restarts.

As we see from Figure 9, the entanglement entropy and concurrence behave similarly to the cost function. A strong indication that entanglement is crucial for the success of VQAA.

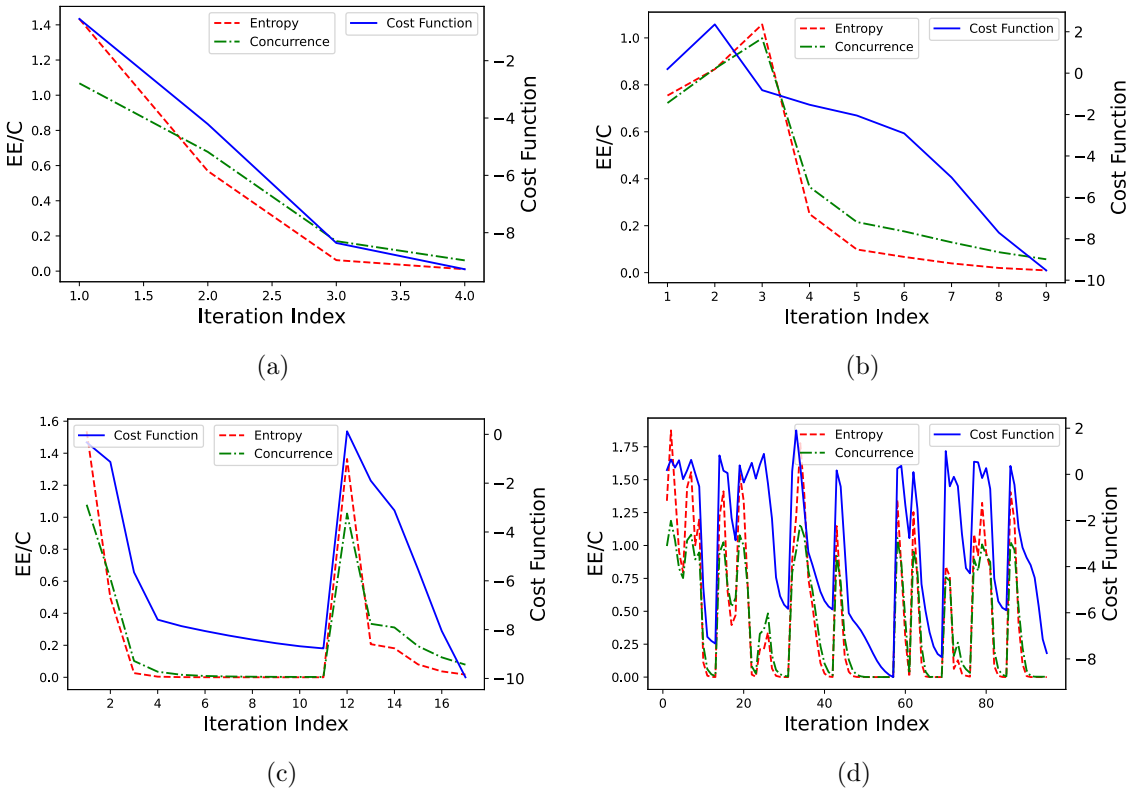



**Figure 9** The relationships between the entanglement entropy, concurrence and the cost function, where the y-label $EE$ and $C$ represent the entanglement entropy and concurrence respectively

## 5 Summary

In this work, we proposed a VQAA for orchestrating an attack in symmetric cryptography. We first constructed a cost function whose minimum corresponds to the Hamiltonian's ground state, which is the known ciphertext. The ground state is found by a variational quantum algorithm. Our simulations show that the Gradient Descent method is much faster than the N-M method. The result of Gradient Descent method is comparable to that of Grover's algorithm, and sometimes it is even faster than Grover's algorithm. It will be interesting to investigate further if this trend is also valid when the number of qubits becomes high. If this trend persists when the number of qubits is high, then it is a serious threat to the symmetric cryptography. There are still a lot of open issues, such as the employment of a better classical optimization algorithm, a better ansatz, and better initial parameters. Further studies will continue in the future.

**Acknowledgements** S.W. acknowledges the National Natural Science Foundation of China under Grants No.12005015; We acknowledge the support from the National Natural Science Foundation of China under Grants No.11974205, and No.11774197; the National Key Research and Development Program of China (2017YFA0303700); the Key Research and Development Program of Guangdong province (2018B030325002); Beijing Advanced Innovation Center for Future Chip (ICFC).

**Supporting information** Appendix A-C. The supporting information is available online at info.scichina.com and link.springer.com.

**References**

1 Feng D, Lian Y. Challenges to Cyberspace Security and Countermeasures. Bull Chin Acad Sci (in Chinese), 2021, 36(10): 1239-1245

2 You X, Wang C X, Huang J, et al. Towards 6G wireless communication networks: Vision, enabling technologies, and new paradigm shifts. Sci China Inf Sci, 2021, 64(1): 110301

3 Rivest R L, Shamir A, Adleman L. A method for obtaining digital signatures and public-key cryptosystems. Commun ACM, 1978, 21(2): 120-126

4 Joan D, Vincent R. The design of rijndael: AES-the advanced encryption standard. Inf Secur Cryptogr, 2002

5 Arute F, Arya K, Babbush R, et al. Quantum supremacy using a programmable superconducting processor. Nature, 2019, 574(7779): 505-510

6 Zhu Q, Cao S, Chen F, et al. Quantum computational advantage via 60-qubit 24-cycle random circuit sampling. Sci Bull, 2022, 67(3): 240-245

7 Chang C R, Lin Y C, Chiu K L, et al. The Second Quantum Revolution with Quantum Computers. AAPPS Bull, 2020, 30(1): 9-22

8 Kwek L C, Cao L, Luo W, et al. Chip-based quantum key distribution. AAPPS Bull, 2021, 31(1): 15

9 Shor P W. Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM Rev, 1999, 41(2): 303-332

10 Gidney C, Ekera M. How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits. Quantum, 2021, 5: 433

11 Grover L K. A fast quantum mechanical algorithm for database search. In: Proc Annu ACM Symp Theory Comput. 1996: 212-219

12 Long G L. Grover algorithm with zero theoretical failure rate. Phys Rev A, 2001, 64(2): 022307

13 Zhu Y, Wang Z, Yan B, et al. Robust quantum search with uncertain number of target states. Entropy, 2021, 23(12): 1649

14 Grassl M, Langenberg B, Roetteler M, et al. Applying Grover's algorithm to AES: quantum resource estimates. In: Post-Quantum Cryptography. Springer, 2016: 29-43

15 Zou J, Wei Z, Sun S, et al. Quantum circuit implementations of AES with fewer qubits. In: International Conference on the Theory and Application of Cryptology and Information Security. Springer, 2020: 697-726

16 Wang Z G, Wei S J, Long G L. A quantum circuit design of AES requiring fewer quantum qubits and gate operations. Front Phys, 2022, 17: 41501

17 Denisenko D, Nikitenkova M. Application of Grover's quantum algorithm for SDES key searching. J Exp Theor Phys, 2019, 128(1): 25-44

18 Preskill J. Quantum computing in the NISQ era and beyond. Quantum, 2018, 2: 79

19 Peruzzo A, Mcclean J, Shadbolt P, et al. A variational eigenvalue solver on a photonic quantum processor. Nat Commun, 2014, 5(1): 4213

20 Yung M H, Casanova J, Mezzacapo A, et al. From transistor to trapped-ion computers for quantum chemistry. Sci Rep, 2014, 4(1): 3589

21 Farhi E, Goldstone J, Gutmann S. A quantum approximate optimization algorithm. 2014. arXiv:1411.4028

22 Harrigan M P, Sung K J, Neeley M, et al. Quantum approximate optimization of non-planar graph problems on a planar superconducting processor. Nat Phys, 2021, 17(3): 332-336

23 Cervera-Lierta A, Kottmann J S, Aspuru-Guzik A. Meta-variational quantum eigensolver: Learning energy profiles of parameterized Hamiltonians for quantum simulation. PRX Quantum, 2021, 2(2): 020329

24 Mcardle S, Endo S, Aspuru-Guzik A, et al. Quantum computational chemistry. RMP, 2020, 92(1): 015003

25 Aspuru-Guzik A, Dutoi A D, Love P J, et al. Simulated quantum computation of molecular energies. Science, 2005, 309(5741): 1704-1707

26 Wei S J, Chen Y H, Zhou Z R, et al. A quantum convolutional neural network on NISQ devices. AAPPS Bull, 2022, 32(1): 2

27 Huang H L, Du Y, Gong M, et al. Experimental quantum generative adversarial networks for image generation. Phys Rev Appl, 2021, 16(2): 024051

28 Beer K, Bondarenko D, Farrelly T, et al. Training deep quantum neural networks. Nat Commun, 2020, 11(1): 808

29 Rebentrost P, Gupt B, Bromley T R. Quantum computational finance: Monte Carlo pricing of financial derivatives. Phys Rev A, 2018, 98(2): 022321

30 Tang H, Pal A, Wang T Y, et al. Quantum computation for pricing the collateralized debt obligations. Quantum eng, 2021, 3(4): e84

31 Egger D J, Gambella C, Marecek J, et al. Quantum computing for finance: state of the art and future prospects. IEEE J Quantum Electron, 2020, 1: 3101724

32 Tuchman W. A brief history of the data encryption standard. In: Internet besieged: countering cyberspace scofflaws. ACM Press/Addison-Wesley Publishing Co, 1997: 275-280

33 Nelder J A, Mead R. A simplex method for function minimization. Comput J, 1965, 7(4): 308-313

## Appendix A The details of S-DES

**The generation of sub-key**

Firstly, the initial key can be arranged as $(k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10})$. The sub-keys $K_1, K_2$ can be generated as

$$\begin{aligned} K_1 &= \text{P8}[\ \text{Shift}_1[\text{P10[key]}]], \\ K_2 &= \text{P8}[\ \text{Shift}_2[\ \text{Shift}_1[\text{P10[key]}]]]. \end{aligned} \tag{A1}$$

The replacement function P10 is defined as

$$\begin{aligned} &\text{P10}\,(k_1, k_2, k_3, k_4, k_5, k_6, k_7, k_8, k_9, k_{10}) \\ =&\,(k_3, k_5, k_2, k_7, k_4, k_{10}, k_1, k_9, k_8, k_6) \end{aligned} \tag{A2}$$

So the first bit of the output is the third bit of the input, the second bit of the output is the fifth bit of the input, etc. Then the first 5 bits and the last 5 bits are shifted to the left by one bit ($\text{shift}_1$), respectively. Next, we select 8 bits from the above 10 bits by P8 as the sub-key $K_1$. where P8 is expressed as

| P8 |
|---|
| 6 3 7 4 8 5 10 9 |

. (A3)

In Equation A3, the numbers represent the positions in the 10-bit string of Equation A2 and the figures in this section have the same meaning.

Finally, let us return to the 5-bit string pairs produced by the function $\text{Shift}_1$, then shift them to the left by two bits ($\text{Shift}_2$). Turning to the sub-key $K_2$ now, it is obtained by the operation P8 of Equation A3.

**The encryption**

For an 8-bit plaintext, the first equation is IP:

| IP |
|---|
| 2 6 3 1 4 8 5 7 |

. (A4)

This function retains the 8-bit information of the plaintext but rearranges it. The inversion function of IP is

| $\text{IP}^{-1}$ |
|---|
| 4 1 3 5 7 2 8 6 |

. (A5)

The most complex part of S-DES is the function $f_K$, which consists of the replacement and substitution operations. To be specific, $f_K$ is expressed as

$$f_K(\text{L}, \text{R}) = (\text{L} \oplus F(\text{R}, K), \text{R}), \tag{A6}$$

where L and R are the left 4 bits and right 4 bits of the 8-bit input, $F$ is a mapping from 4 bits to 4 bits, $K$ is the sub-key, $\oplus$ is XOR operation. Assuming that the output of the function IP is 10111101, for some key $K$, we have $F(1101, K) = 1110$. Because we have $1011 \oplus 1110 = 0101$, this yields $f_K(10111101) = 01011101$.

Now we illustrate the map $F$. Its input is a 4-bit string: $(n_1, n_2, n_3, n_4)$. The first operation is its extension

$$\begin{array}{c} \\ \text{E/P} \\ \\ \begin{array}{cccccccc} 4 & 1 & 2 & 3 & 2 & 3 & 4 & 1 \end{array} \end{array}. \tag{A7}$$

which can be expressed as

$$\left|\begin{array}{ccc|c} n_4 & n_1 & n_2 & n_3 \\ & & & \\ n_2 & n_3 & n_4 & n_1 \end{array}\right|. \tag{A8}$$

Upon performing XOR with the 8-bit sub-key $K1 = (k_{11}, k_{12}, k_{13}, k_{14}, k_{15}, k_{16}, k_{17}, k_{18})$, we arrive at

$$\left|\begin{array}{ccc|c} n_4 \oplus k_{11} & n_1 \oplus k_{12} & n_2 \oplus k_{13} & n_3 \oplus k_{14} \\ & & & \\ n_2 \oplus k_{15} & n_3 \oplus k_{16} & n_4 \oplus k_{17} & n_1 \oplus k_{18} \end{array}\right|. \tag{A9}$$

It can be recorded as

$$\left|\begin{array}{ccc|c} p_{0,0} & p_{0,1} & p_{0,2} & p_{0,3} \\ & & & \\ p_{1,0} & p_{1,1} & p_{1,2} & p_{1,3} \end{array}\right|. \tag{A10}$$

The first 4 bits (the first row in Equation. (A10)) and the last 4 bits (the second row in Equation. (A10)) are imported into the S-box $S_0$ and S-box $S_1$ and the 2-bit output strings, respectively. The pair of S-boxes are as follows:

$$\mathrm{S}_0 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \left(\begin{array}{cccc} 1 & 0 & 3 & 2 \\ 3 & 2 & 1 & 0 \\ 0 & 2 & 1 & 3 \\ 3 & 1 & 3 & 2 \end{array}\right) \end{array}, \qquad \mathrm{S}_1 = \begin{array}{c} \\ 0 \\ 1 \\ 2 \\ 3 \end{array} \begin{array}{c} \begin{array}{cccc} 0 & 1 & 2 & 3 \end{array} \\ \left(\begin{array}{cccc} 0 & 1 & 2 & 3 \\ 2 & 0 & 1 & 3 \\ 3 & 0 & 1 & 0 \\ 2 & 1 & 0 & 3 \end{array}\right) \end{array}. \tag{A11}$$

The first and forth bits of the 4-bit input form a 2-bit binary number, which represents the S-box row, while the second and third bits represent the S-box column. The element determined by the row and column of S-box is the output, which is a 2-bit binary number.

Then the 4-bit output from $S_0$ and $S_1$ is ordered by the operation P4:

$$\begin{array}{c} \\ \text{P4} \\ \\ 2\ 4\ 3\ 1 \end{array}. \tag{A12}$$

The output of P4 is the result of the function $F$.

The function $f_K$ just changes the 4 bits on the left, while the Function SW is an exchange function, which swaps the left 4 bits and right 4 bits of the input, so the input of function $f_K$ in the second round is represented by four different bits.

## Appendix B The performance of different regular graphs

In Table B1, we show the average number of iterations for different regular graphs where the initial parameters are the same and the other parameters have been set as the proper values. The data is calculated by 15 simulations with the Y-Cz (A) ansatz and the gradient method.

**Table B1** The number of iterations for different regular graphs

| | 1-reg | 2-reg | 3-reg | 4-reg | 5-reg | 6-reg | 7-reg |
|---|---|---|---|---|---|---|---|
| Learning rate | 0.72 | 0.84 | 1.08 | 1.44 | 1.92 | 2.40 | 2.88 |
| Restart condition | 0.53 | 0.62 | 0.80 | 1.07 | 1.42 | 1.78 | 2.13 |
| Maximum | 94 | 94 | 94 | 94 | 94 | 94 | 94 |
| Minimum | 4 | 2 | 4 | 2 | 2 | 1 | 14 |
| Average | 53.27 | 33.07 | 31.13 | 32.73 | 34.87 | 44.93 | 65.60 |

When the norm of gradient is less than the value of Restart condition, the optimization will be restarted. From the results of Table B1, we find the 3-regular graph performs best.

## Appendix C Two classical optimization algorithms

**Algorithm C1** Gradient Descent method

**Require:** Initial point $x_0$, the function $f$, the learning rate r, the cut-off condition xerr.

times=0
Calculate len=length($x_0$)
**for** $ii$ in range(1024) **do**
  Let cost = $f(x_0)$
  times=times+1
  **if** costf<xerr **then**
    break
  **end if**
  Initialize a zero vector Gd, the length is len
  **for** $i$ in range(len) **do**
    Let $x \leftarrow x_0$
    Change the $i$-th component of $x$: $x_i = x_i + 0.01$
    $\text{cost}' = f(x)$
    times=times+1
    The $i$-th component of Gd: $\text{Gd}_i = (\text{cost}' - \text{cost})/0.01$
  **end for**
  Generate a random number $r_0$ in range $[0, 1]$
  $x_0 = x_0 - (\text{r}/|\text{cost}| + \log(\text{times})/\text{times} * r_0) * \text{Gd}$
  **if** $|\text{Gd}| < 0.8$ **then**
    Initialize a $x_0$ randomly
  **end if**
**end for**

**return** $x$

* The learning rates in the A-ansatz are set to 0.72, 0.72, 1.08 for the Y-Cx model, Y-Cy model, Y-Cz model respectively; the learning rates in the B-ansatz are set to 0.72, 0.76, 0.94 for the Y-Cx model, Y-Cy model, Y-Cz model respectively.

**Algorithm C2** N-M method

**Require:** Generating the other $N$ (the dimension of $x$) points $(x_1, ..., x_N)$ according to the initial point $x_0$. Let the $i$-th component in $x_i$ is $\alpha$ (the amplification factor) larger than the $i$-th component in $x_0$. If the $i$-th component is 0 in $x_0$, the $i$-th component in $x_i$ is set to 0.8, the cut-off condition xerr.

times=$N$+1
**while** times<1024 **do**
  Sort and rename these points $(x_i)$ in ascending order according to the value of $f(x_i)$, the larger $i$, the larger $f(x_i)$.
  **if** $f(x_0) \leqslant$ xerr **then**
    Break
  **end if**
  **if** $f(x_N) - f(x_0) < 0.15$ **then**
    Restart
  **end if**
  Calculate the average of the first $N$ points $m = \frac{1}{N}\sum_{i=0}^{N-1} x_i$
  Calculate the reflect point $r = 2m - x_N$
  times=times+1
  **if** $f(x_1) \leqslant f(r) < f(x_{N-1})$ **then**
    $x_N = r$

Continue
**end if**
**if** $f(r) < f(x_1)$ **then**
Calculate the expand point $s = m + 2(m - x_N)$
times=times+1
**if** $f(s) < f(r)$ **then**
$x_N = s$
Continue
**else**
$x_N = r$
Continue
**end if**
**end if**
**if** $f(x_{N-1}) \leqslant f(r) < f(x_N)$ **then**
$c_1 = m + (r - m)/2$
times=times+1
**if** $f(c_1) < f(r)$ **then**
$x_N = c_1$
Continue
**else**
$v_i = x_0 + (x_i - x_0)/2$
$x_i = v_i$; $(i = 1, ..., N)$
times=times+N
Continue
**end if**
**end if**
**if** $f(x_N) \leqslant f(r)$ **then**
$c_2 = m + (x_N - m)/2$
times=times+1
**if** $f(c_2) < f(x_N)$ **then**
$x_N = c_2$
Continue
**else**
$v_i = x_0 + (x_i - x_0)/2$
$x_i = v_i$; $(i = 1, ..., N)$
times=times+N
Continue
**end if**
**end if**
**end while**

**return** $x_0$

* The amplification factors in the A-ansatz are set to 2.7, 2.7, 2.8 for the Y-Cx model, Y-Cy model, Y-Cz model respectively; the amplification factors in the B-ansatz are set to 2.7, 2.7, 2.7 for the Y-Cx model, Y-Cy model, Y-Cz model respectively.