# Instance Space Analysis of Quantum Algorithms

**Supervisors:** Prof. Kate Smith-Miles, Prof. Lloyd Hollenberg

**PhD Completion Seminar**

Vivek Katial

The University of Melbourne

2024-11-26

# Agenda

# Agenda

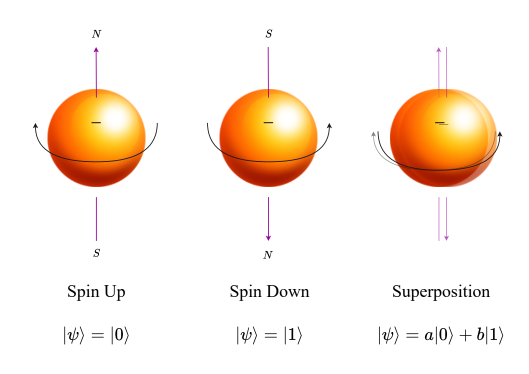
1. Introduction
2. Quantum Approximate Optimisation Algorithm (QAOA)
3. Instance Space Analysis
4. Evolving Instances for QAOA
5. Software for QAOA Parameter Initialisation
6. Conclusion and Future Work

# Introduction

# Why Quantum?

1. Quantum Computers can *theoretically solve* can solve some problems much faster than classical computers
2. What problems?
   - **Shor's algorithm** for factoring large numbers - could break RSA encryption [15]
   - **Grovers Search** - Quadratic speedup over classical search [7]
   - **Simulation of physical systems** - Quantum Chemistry, Material Science
3. What's the catch?
   - Hardware is **hard** - assuming no errors we need several 1000s of qubits
   - With current error rates - need millions of qubits + 100s of millions of gates
   - **NISQ** - Noisy Intermediate-Scale Quantum (NISQ) devices - 50-100 qubits, noisy, error-prone

# How does Quantum Computing work?




## Superposition

- Quantum bits (qubits) can exist in a superposition of states
- For $N = 2$ qubits: $|\psi\rangle \in \mathbb{C}^2 \otimes \mathbb{C}^2 = \mathbb{C}^4$
- Represented by a vector in a complex vector space $|\psi\rangle \in [\mathbb{C}^2]^{\otimes n}$
- For $N = 300$, there are $2^{300}$ possible states - more than the number of atoms in the universe!

# Noisy-intermediate Scale Quantum Era

- Currently we're in the NISQ-era of Quantum Computing
- Need to design algorithms that can run on NISQ-devices

Need to find algorithms that can:

1. Can run on small (100-1000 qubit devices)
2. Solve useful problems
3. Shouldn't require extensive error correction

This has led to the development of **Variational Quantum Algorithms (VQAs)** [9], which are hybrid algorithms specifically designed for NISQ devices

# Aims of the Thesis

- VQAs are a promising class of quantum algorithms tailored for NISQ devices
- Two prominent examples:
  1. **Quantum Approximate Optimisation Algorithm (QAOA)** [4]
  2. **Variational Quantum Eigensolver (VQE)**
- QAOA in particular is a low-depth algorithm designed to solve optimisation problems
  - Many problems can be mapped to a Hamiltonian and solved using QAOA (e.g. MaxCut, TSP, Vehicle Routing, 3SAT)

# MaxCut Problem

Partition a graph $G = (V, E)$ into two sets $S$ and $V \setminus S$ such that the number of edges between the two sets is maximised.

$$\max_{\mathbf{s}} \sum_{(i,j)\in E} w_{ij}(1 - z_i z_j)$$

where $z_i \in \{0, 1\}$ and $w_{ij} \in \mathbb{R}$ is the weight of edge $(i, j)$.

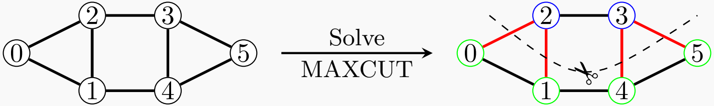



**Figure 1:** An example of a six-node MaxCut problem

Solution is a binary string $\mathbf{s} = (s_1, s_2, \ldots, s_n)$ where $s_i \in \{0, 1\}$ and the optimal objective value is $C_{\max}$ where $C_{\max} = \sum_{(i,j)\in E} w_{ij}$ for the edges in our **maximum cut**.

We map the MaxCut problem to a *Hamiltonian* for quantum optimisation.

## Classical Formulation

Objective:

$$\max_{\mathbf{s}} \sum_{(i,j)\in E} w_{ij}(1 - z_i z_j)$$

State Space:

$$\mathbf{s} \in \{0,1\}^n$$

Solution:

$$\mathbf{s}^* = \operatorname*{argmax}_{\mathbf{s}} \sum_{(i,j)\in E} w_{ij}(1 - z_i z_j)$$

$$\xrightarrow{\text{map}}$$

## Quantum Formulation

Objective:

$$H = \sum_{(i,j)\in E} w_{ij}(I - Z_i Z_j)$$

State Space:

$$|\psi\rangle \in \mathcal{H}_2^{\otimes n}$$

Solution:

$$|\psi_{\text{ground}}\rangle = \operatorname*{argmin}_{|\psi\rangle} \langle\psi|H|\psi\rangle$$

# Quantum Approximate Optimisation Algorithm (QAOA)

- QAOA prepares a parameterised “trial” (ansatz) state of the form:

$$\begin{aligned} |\psi(\theta)\rangle &= |\psi(\vec{\gamma}, \vec{\beta})\rangle \\ &= \prod_{j=1}^{p} e^{-i\beta_j \hat{H}_B} e^{-i\gamma_j \hat{H}_P} \left|+\right\rangle^{\otimes n} \end{aligned}$$

- Where $\hat{H}_P = \sum_{(i,j)\in E} w_{ij}(1 - \hat{Z}_i \hat{Z}_j)$ is the problem Hamiltonian and $\hat{H}_B = \sum_{i=1}^{n} \hat{X}_i$ is the mixing Hamiltonian.
- The parameters $\vec{\gamma} = (\gamma_1, \ldots, \gamma_p)$ and $\vec{\beta} = (\beta_1, \ldots, \beta_p)$ are optimised to minimise the expectation value of the problem Hamiltonian.

- The QAOA Ansatz Energy is given by taking the expectation value of the problem Hamiltonian with respect to the trial state:

$$\begin{aligned} F_p(\vec{\gamma}, \vec{\beta}) &= \langle + |^{\otimes n} \left( \prod_{j=1}^{p} e^{-i\beta_j \hat{H}_B} e^{-i\gamma_j \hat{H}_P} \right)^{\dagger} \hat{H}_P \left( \prod_{j=1}^{p} e^{-i\beta_j \hat{H}_B} e^{-i\gamma_j \hat{H}_P} \right) | + \rangle^{\otimes n} \\ &= \langle \psi(\vec{\gamma}, \vec{\beta}) | \hat{H}_P | \psi(\vec{\gamma}, \vec{\beta}) \rangle \end{aligned}$$

- The goal is to find the optimal parameters $\vec{\gamma}^*, \vec{\beta}^*$ that minimise the energy $F_p(\vec{\gamma}, \vec{\beta})$.

$$(\vec{\gamma}^*, \vec{\beta}^*) = \arg\min_{\vec{\gamma}, \vec{\beta}} F_p(\vec{\gamma}, \vec{\beta}), \qquad \alpha = \frac{F_p(\vec{\gamma}^*, \vec{\beta}^*)}{C_{\max}}$$

## Key Design Decisions

1. **Circuit Depth** $(p)$
   - Controls the expressivity of the ansatz
   - As $p \to \infty$ QAOA can find the exact solution
2. **Classical Optimiser**
   - Gradient-free: Nelder-Mead, COBYLA
   - Gradient-based: ADAM, SPSA
3. **Initial Parameters** $(\vec{\gamma}_0, \vec{\beta}_0)$
   - Various strategies for initialisation (e.g. TQA [11], INTERP [18])

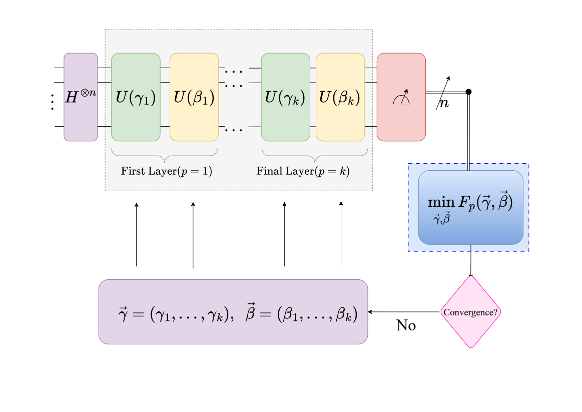



**Figure 2:** QAOA Circuit Architecture

# Current State of QAOA Research

## Key Findings

- Recent studies show optimal QAOA parameters for depth $p$ are transferable across a small class of instances [2]
- Optimal QAOA depth is instance-dependent [12]
- Algorithm performance heavily influenced by:
  - Choice of classical optimiser [5]
  - Initial parameter selection [18],[8],[14],[16]

## Current Limitations

- Narrow focus on specific instance classes ($d-$regular graphs, random graphs, no weights)
- Predominantly shallow depth circuits ($p \leq 3$) [12],[16]
- Limited understanding of instance feature impacts on QAOA performance and design decisions
- Lack of standardised parameter initialisation frameworks [1]

# Research Questions

## RQ1

How can we generate diverse MaxCut instances beyond current QAOA research?

## RQ2

How do instance characteristics influence key QAOA design decisions?

## RQ3

Can we develop methods to automatically optimise QAOA parameters based on instance features?

# QAOA Landscape Complexity & Instance Diversity

- **Landscape Evolution with Depth:**
  - Higher $p$ – more complex optimisation landscape [3]
- **Weight Distribution Impact:**
  - Cauchy distributions $\rightarrow$ more rugged landscapes [13]
  - Weight scaling heuristics show promise
  - Convergence to unweighted case under rescaling [16]
- **Practical Implications:**
  - More complex landscapes require more calls to the QPU, classical optimisers can falter $\rightarrow$more resource use!

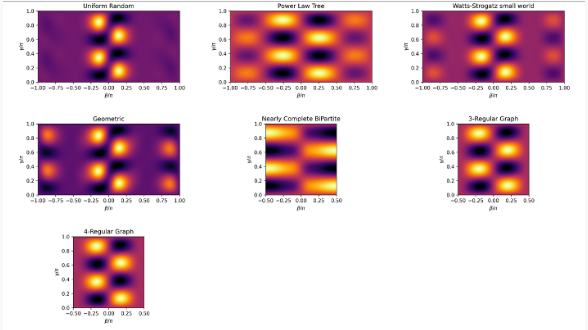



**Figure 3:** QAOA Landscape Variation Across MaxCut Instances (p=1, unweighted)

# Instance Space Analysis

- Based on the *No Free Lunch Theorem* [17]: Algorithms have strengths and weaknesses
- Identify **features** that differentiate instances from each other and influence algorithm performance
- Identifies which **algorithms** are best suited for which instances - instead of reporting "on-average" performance
- ISA projects instances onto a 2D space to visualise the strengths and weaknesses of algorithms
- Based on meta-data $\{\mathcal{I}, \mathcal{F}, \mathcal{A}, \mathcal{Y}\}$

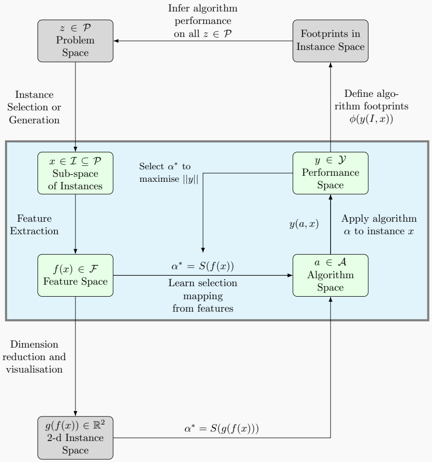
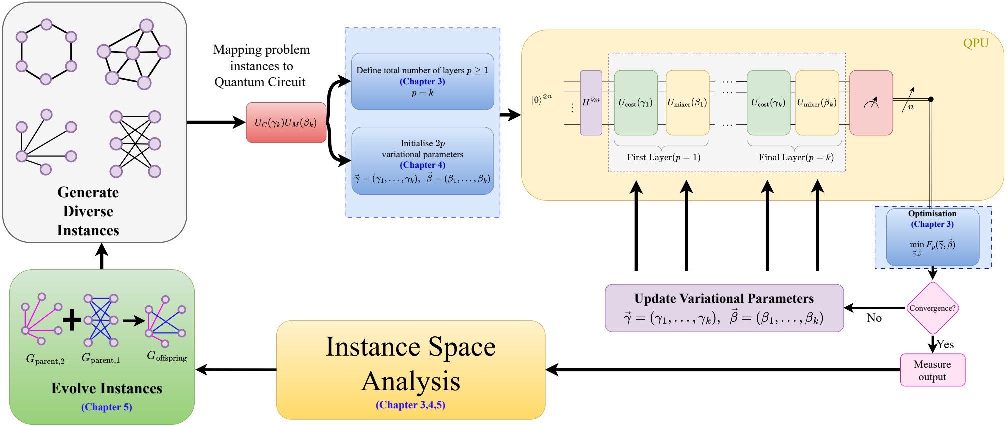



**Figure 4:** Instance space analysis workflow

# Our Approach



# Instances $\mathcal{I} \subseteq \mathcal{P}$

## IS I & II: Network Structures

- Random
- 3-regular
- 4-regular
- Geometric
- Watts-Strogatz
- Nearly complete bipartite

## IS III & IV: Weight Distributions

- $\mathcal{U}[0,1]$ (Uniform)
- $\mathcal{U}[-1,1]$ (Uniform)
- Exp$(\lambda)$ (Exponential)
- $\Gamma(k,\theta)$ (Gamma)
- $\mathcal{N}(\mu,\sigma^2)$ (Normal)
- LogNorm$(\mu,\sigma^2)$ (Lognormal)

---

**Total Instance Classes:** $7 \times 6 = 42$

# Features $\mathcal{F}$

## Structural Features

- Number of Edges, Nodes
- Bipartite Graph
- Clique Number
- Connected Graph
- Density
- Edge Connectivity
- Max, Min Degree
- Min. Dominating Set size
- Regular Graph
- Smallest Eigenvalue
- Vertex Connectivity

## Spectral Features

- Algebraic Connectivity
- Laplacian Largest Eigenvalue
- Ratio of Two Largest Laplacian Eigenvalues
- Ratio of Two Smallest Laplacian Eigenvalues

## Cycle & Path Features

- Acyclic Graph
- Average Distance
- Diameter
- Eulerian Graph
- Number of Components
- Planar Graph
- Radius

## Literature

- Distance-Regular Graph
- Group Size
- Number of Cut Vertices
- Number of Minimal Odd Cycles
- Number of Orbits
- Graph Entropy: $I(G) = \frac{1}{n} \sum_i |A_i| \log |A_i|$

## Weight Features

- Mean, Median, Standard Deviation, Variance
- Min/Max, IQE, Skewness, Kurtosis
- Coef. of Variation
- Weighted Avg Clustering
- Weighted Avg Path Length
- Weighted Diameter
- Weighted Radius
- Max Weighted Degree
- Min Weighted Degree

**Instance Space III and IV only**

---

**48 Features**

# Performance $y$

- **Why?** Most algorithms achieve good approximation ratios
- **Components:**
  - Function evaluations
  - Approximation ratio ($\alpha$)
- **Method to compute performance ($\kappa$):**
  1. Find $\alpha_{\max}$ (best ratio)
  2. Set $\alpha_{\text{acceptable}} = 0.95\alpha_{\max}$
  3. Count iterations ($\kappa$) to reach $\alpha_{\text{acceptable}}$
  4. If never reached, set penalty $\kappa = 10^5$

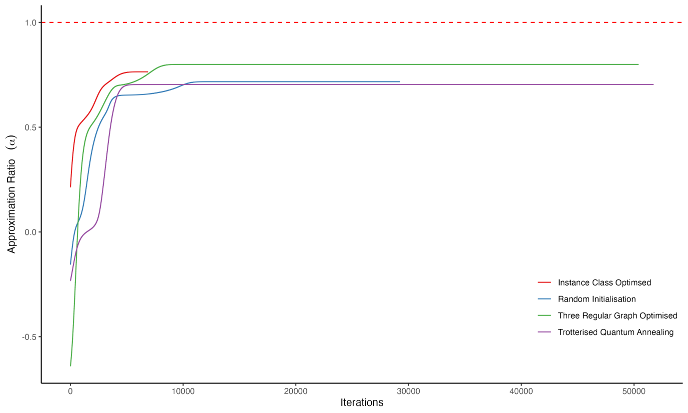



**Figure 5:** Approximation ratio $\alpha$ for various instances

# Algorithm Design Choices $\mathcal{A}$

## Design Choices Across Independent Studies

- **IS I:** Layer Depth ($p$)
  - How many layers are needed for different instances?
  - $p \in 2, 5, 10, 15, 20$
- **IS II:** Classical Optimiser Selection
  - Which optimiser requires fewer calls to the quantum device?
  - Nelder-Mead, Conjugate Gradient, Powell, SLSQP, L-BFGS-B
- **IS III/IV:** Initialisation Technique
  - Can we have faster convergence to optimal parameters?
  - Random, TQA, INTERP, Constant, QIBPI, Three-Regular

**Fixed meta-data**

- Features
- Instances
- Performance Measure

**All choices represent design decisions**

Each study isolates one key design choice while maintaining other parameters constant

# Instance Space III: Parameter Initialisation

**Input:** Number of nodes $N = 8$, Allowed Number of layers $L = 15$
**Output:** Median optimal parameters $\vec{\gamma}$ and $\vec{\beta}$ for each class
**Procedure** `GenerateGraphInstances()`:
  **for** $p \leftarrow 1$ **to** $L$ **do**
    **foreach** *graph type* $T$ *in* $T_1, \ldots, T_{42}$ **do**
      **for** $i \leftarrow 1$ **to** $100$ **do**
        Generate graph instance $G_{i,T}$ with $N$ nodes; `optimiseQAOAParameters(`$G_{i,T}$, $p$`)`;
      **end**
      `CalculateMedianParameters(`$T_p$`)`;
    **end**
  **end**
**Function** `optimiseQAOAParameters(`$G$, $p$`)`:
  **Input:** Graph instance $G$
  **Output:** Optimal parameters $(\vec{\gamma}_G, \vec{\beta}_G) = (\gamma_1, \ldots, \gamma_p), (\beta_1, \ldots, \beta_p)$
  Run QAOA on graph instance $G$ with random initialisation to find optimal parameters for depth $p$; **return** $(\vec{\gamma}_G, \vec{\beta}_G) = (\gamma_1, \ldots, \gamma_p), (\beta_1, \ldots, \beta_p)$;
**Function** `CalculateMedianParameters(`$T$`)`:
  **Input:** Graph type $T$, Set of parameters $\vec{\gamma}i, p, T$, $\vec{\beta}i, p, T$
  **Output:** Median parameters $\vec{\gamma}$median$, T$, $\vec{\beta}$median$, T$
  Calculate the median of the parameters $\gamma_{i,p,T}$ and $\beta_{i,p,T}$ for all instances of type $T$; Store median parameters in a parameter class corresponding to $T$; **return** $\vec{\gamma}$*median*$, T$, $\vec{\beta}$*median*$, T$;

**Algorithm 1:** Generate Initial QAOA Parameters using QIBPI

# QAOA Initialisation Strategies ($p = 15$)

**Random:**

- $\gamma_i \sim \mathcal{U}(-\pi, \pi)$
- $\beta_i \sim \mathcal{U}(-\pi/2, \pi/2)$

**TQA** [11]:

- $\gamma_i = i \cdot \Delta t / p$
- $\beta_i = (1 - i/p) \cdot \Delta t$

**CONSTANT** [6]:

- $\gamma_i = 0.2$
- $\beta_i = -0.2$

**FOURIER:** [18]

- Frequency domain approach
- $\gamma_i = \sum_k u_k \sin((k - \frac{1}{2})(i - \frac{1}{2})\frac{\pi}{p})$

**INTERP:** [18]

- Parameter interpolation
- $[\gamma_0^{(p+1)}]_i = \frac{i-1}{p}[\gamma_L^{(p)}]_{i-1}$

**QIBPI:**

- Uses QIBPI pre-optimised parameters

**3-Regular:**

- Uses 3-regular pre-optimised parameters

The projection transformation is given by the matrix $\mathbf{Z}$:

$$\mathbf{Z} = \begin{pmatrix} z_1 \\ z_2 \end{pmatrix} = \begin{pmatrix} -0.5225 & 0.2301 \\ -0.5939 & 0.7398 \\ 0.3977 & -0.2637 \\ -0.1423 & -0.2023 \\ -0.0091 & 0.5056 \\ 0.4226 & -0.019 \\ 0.0843 & 0.6528 \\ -0.0033 & -0.0937 \\ -0.2002 & -0.3513 \\ 0.3448 & -0.3839 \end{pmatrix}^{\top} \begin{pmatrix} \text{algebraic connectivity} \\ \text{average distance} \\ \text{clique number} \\ \text{diameter} \\ \text{maximum degree} \\ \text{maximum weighted degree} \\ \text{number of edges} \\ \text{radius} \\ \text{skewness weight} \\ \text{weighted average clustering} \end{pmatrix}$$

Plotting 4,200 instances across $\mathbb{R}^2$.

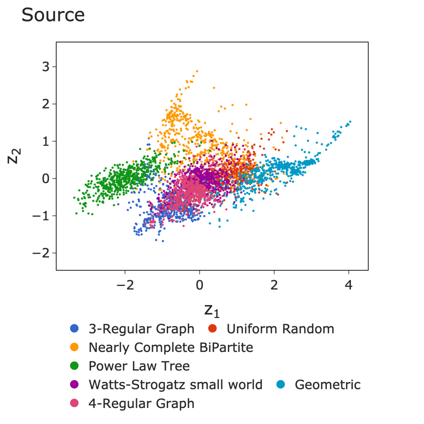
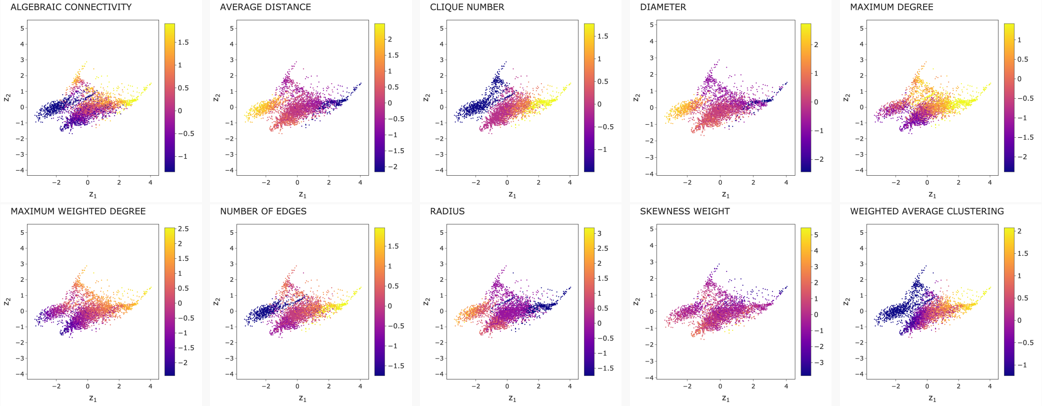


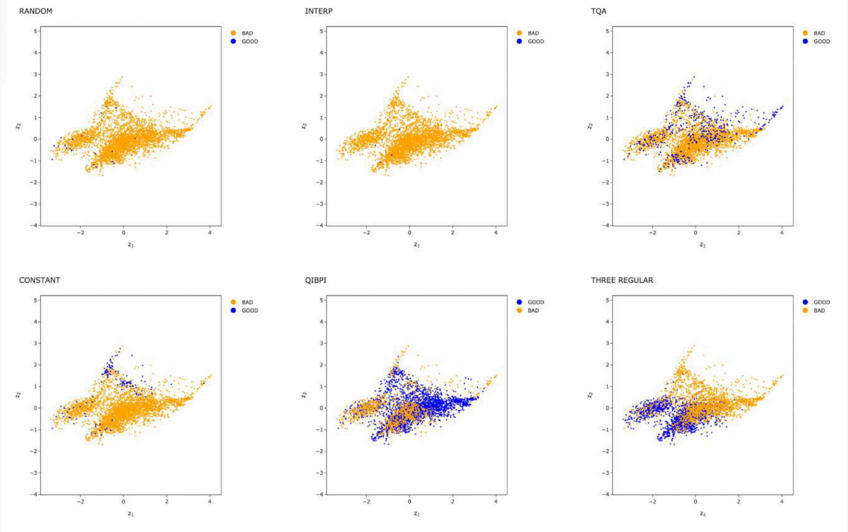

**Figure 6:** Source Distribution

# Instance Space III - Features

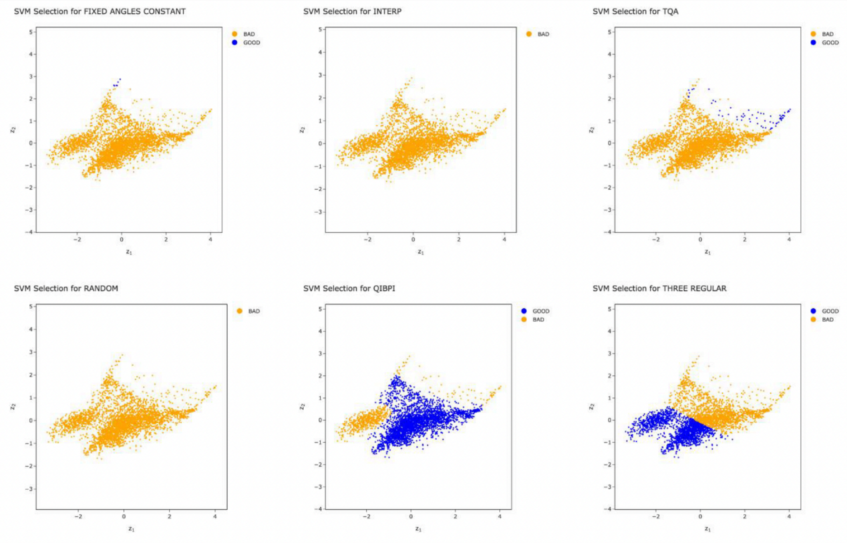




# Instance Space III - Performance



# Instance Space III - Performance



# Instance Space III - Algorithms

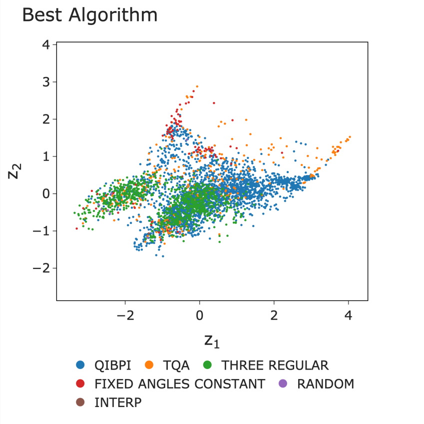



**Figure 7:** Best Algorithm

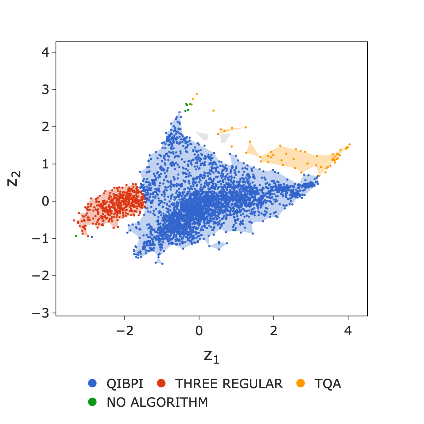



**Figure 8:** Algorithm Distribution

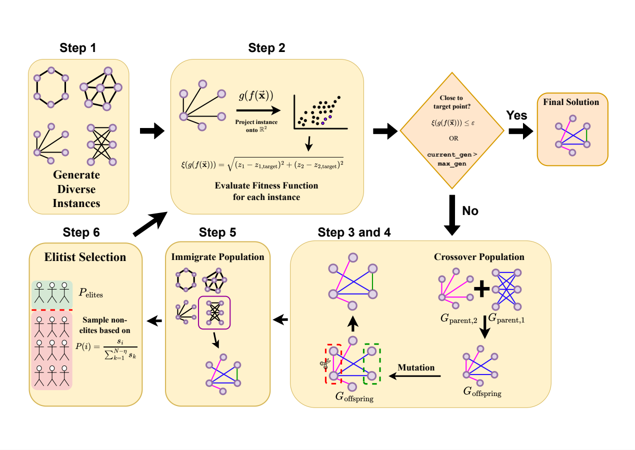
# Instance Space III - Summary

| Initialisation Strategy | $P_{\text{good}}$ (%) | CV Accuracy (%) | CV Precision (%) |
|---|---|---|---|
| CONSTANT | 3.20 | 96.70 | 33.3 |
| INTERP | 0.10 | 99.90 | – |
| QIBPI | 70.90 | 75.80 | 78.00 |
| Three-Regular | 44.80 | 78.50 | 73.80 |
| TQA | 11.10 | 90.10 | 87.50 |
| **Selector** | **77.90** | – | **78.10** |

**Table 1:** Performance metrics for various initialisation strategies

# Evolving Instances for QAOA

# Evolving Instances for QAOA



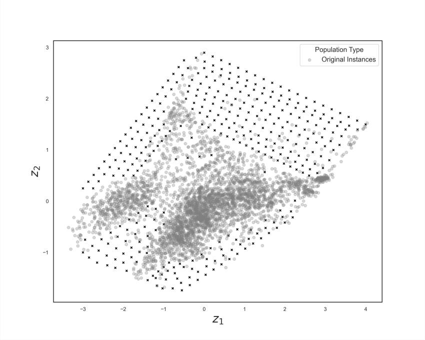




**Figure 9:** Gaps in Instance Space

We identify 360 target points in the instance space and evolve instances to reach these targets.

# Evolving Instances for QAOA

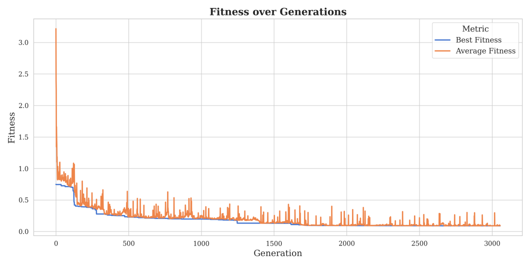



**Figure 10:** Fitness over Generations

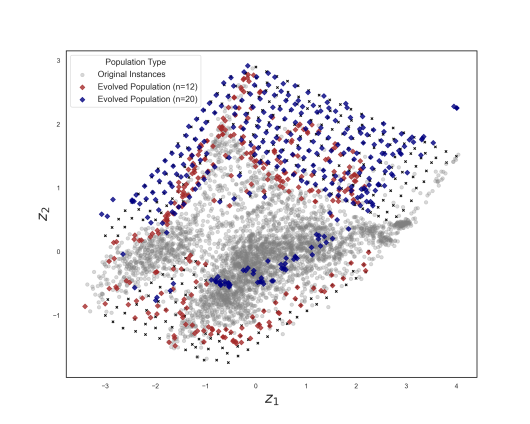
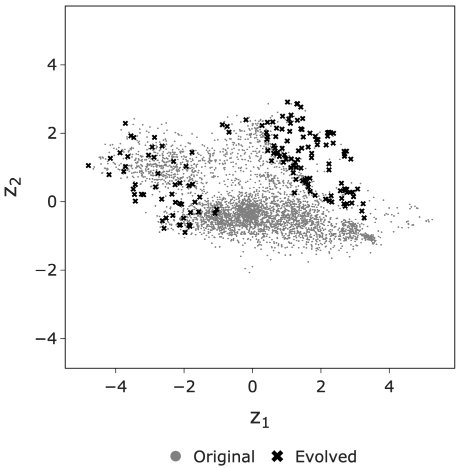
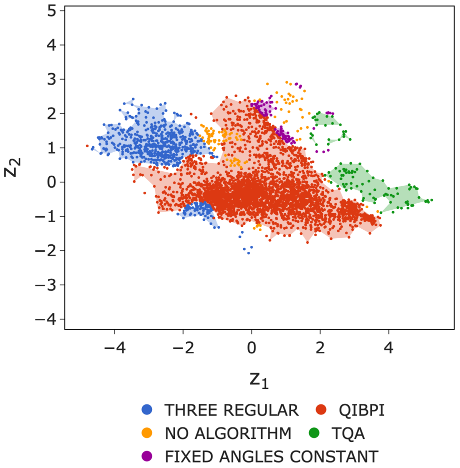



**Figure 11:** Gaps in Instance Space

## Source


4
2
0
−2
−4
$z_2$
−4
−2
0
2
4
$z_1$
Original
Evolved


5
4
3
2
1
0
−1
−2
−3
−4
$z_2$
−4
−2
0
2
4
$z_1$
THREE REGULAR
QIBPI
NO ALGORITHM
TQA
FIXED ANGLES CONSTANT

# Software for QAOA Parameter Initialisation

# Problem and Motivation

- Parameter initialisation significantly impacts QAOA's performance and convergence
- Current challenges:
  - Overwhelming number of initialisation techniques
  - Implementations are coupled with the various quantum libraries (e.g. Cirq, Qiskit, PyQuil)
  - Lack of standardization and comparison across different techniques

# QAOA-GamBet: A Fast API



Key Features:

- Consolidates multiple strategies (Random, QIBPI, QAOAKit, TQA, Constant)
- RESTful API for easy integration
- Modular and extensible
- Built-in validation and error handling
- Automatic documentation generation

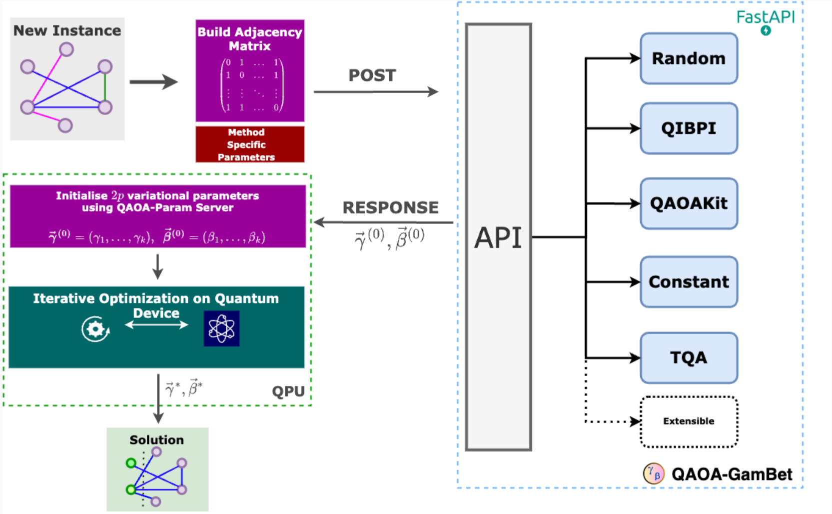



**Figure 12:** FastAPI Workflow

# Example Usage

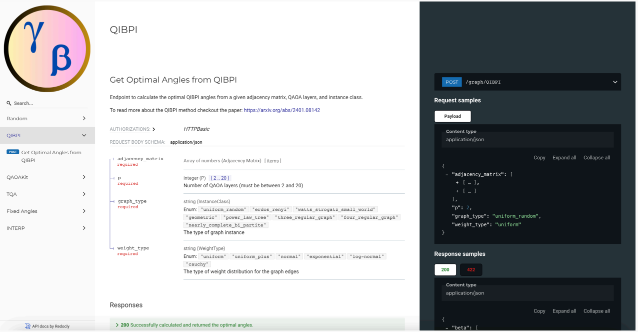
## Request

```python
# Prepare the request data
data = {
    "adjacency_matrix": adjacency_matrix,
    "p": 2,
    "graph_type": "three_regular_graph",
    "weight_type": "uniform"
}

# Send a request to get optimal angles
response = requests.post(
    "http://localhost:5000/graph/QIBPI",
    json=data,
    auth=('username', 'password')
)
```

## Response

```json
{
  "beta": [0.1, 0.15, 0.2],
  "gamma": [0.3, 0.35, 0.4],
  "source": "QIBPI"
}
```




- Auto-generated documentation from docstrings
- Rendered using Swagger UI and redoc

# Conclusion

# Conclusion

- Key Contributions:
  - Expanded QAOA research scope with diverse test suite of MaxCut instances
  - Conducted Instance Space Analysis (ISA) to understand instance-feature impacts on QAOA design choices
  - Quantum Instance-based Parameter Initialisation (QIBPI)
  - FastAPI for QAOA Parameter Initialisation
- Instance Space Analysis:
  - Parameter transfer effective within instance classes, but not across
  - Gaps in instance space identified and applied instance evolution

# Future Work

- Expand analysis scope
  - Extend to larger problem sizes
  - Include more weight distributions and network structures
  - Apply ISA to other quantum algorithms (VQE, F-VQE)
- Bridge theory-practice gap
  - Test on real quantum hardware (e.g., IBM's 128-qubit machine [10])
  - Create ML-based approaches leveraging experimental data

# Thank You - Questions?

# Thank You - Questions?



- Link to QAOA-GamBet
- Link to ISA's

# References i


[1] Amira Abbas et al. **“Quantum Optimization: Potential, Challenges, and the Path Forward”.** In: *arXiv preprint arXiv:2312.02279* (2023).

[2] Fernando G. S. L. Brandão et al. **“For Fixed Control Parameters the Quantum Approximate Optimization Algorithm’s Objective Function Value Concentrates for Typical Instances”.** Version Submitted. In: *arXiv* (Aug. 2023). DOI: 10.48550/arXiv.1812.04170.

[3] Sergey Bravyi, David Gosset, and Robert König. **“Quantum advantage with shallow circuits”.** In: *Science* 362.6412 (2018), pp. 308–311. DOI: 10.1126/science.aar3106. eprint: https://www.science.org/doi/pdf/10.1126/science.aar3106. URL: https://www.science.org/doi/abs/10.1126/science.aar3106.

[4] Edward Farhi, Jeffrey Goldstone, and Sam Gutmann. ***A Quantum Approximate Optimization Algorithm.*** 2014. arXiv: 1411.4028 [quant-ph].

[5] Mario Fernández-Pendás et al. **"A study of the performance of classical minimizers in the Quantum Approximate Optimization Algorithm".** In: *Journal of Computational and Applied Mathematics* 404 (2022), p. 113388. ISSN: 0377-0427. DOI: https://doi.org/10.1016/j.cam.2021.113388. URL: https://www.sciencedirect.com/science/article/pii/S0377042721000078.

[6] Igor Gaidai and Rebekah Herrman. ***Performance Analysis of Multi-Angle QAOA for $p > 1$.*** 2023. arXiv: 2312.00200 [cs.ET].

[7] Lov K. Grover. **"A fast quantum mechanical algorithm for database search".** In: *Proceedings of the twenty-eighth annual ACM symposium on Theory of computing - STOC '96* (1996). DOI: 10.1145/237814.237866. URL: http://dx.doi.org/10.1145/237814.237866.

# References iii


[8] X. Lee et al. **"Parameters Fixing Strategy for Quantum Approximate Optimization Algorithm".** In: *2021 IEEE International Conference on Quantum Computing and Engineering (QCE)*. Los Alamitos, CA, USA: IEEE Computer Society, Oct. 2021, pp. 10–16. DOI: 10.1109/QCE52317.2021.00016. URL: https://doi.ieeecomputersociety.org/10.1109/QCE52317.2021.00016.

[9] Alberto Peruzzo et al. **"A variational eigenvalue solver on a photonic quantum processor".** In: *Nature Communications* 5.1 (2014), p. 4213. DOI: 10.1038/ncomms5213. URL: https://doi.org/10.1038/ncomms5213.

[10] Natasha Sachdeva et al. **"Quantum optimization using a 127-qubit gate-model IBM quantum computer can outperform quantum annealers for nontrivial binary optimization problems".** In: *arXiv preprint arXiv:2406.01743* (2024).

# References iv


[11] Stefan H. Sack and Maksym Serbyn. **"Quantum annealing initialization of the quantum approximate optimization algorithm".** In: *Quantum* 5 (July 2021), p. 491. ISSN: 2521-327X. DOI: 10.22331/q-2021-07-01-491. URL: https://doi.org/10.22331/q-2021-07-01-491.

[12] Ruslan Shaydulin et al. **"Classical symmetries and the quantum approximate optimization algorithm".** In: *Quantum Information Processing* 20 (2021), pp. 1–28.

[13] Ruslan Shaydulin et al. **"Parameter Transfer for Quantum Approximate Optimization of Weighted MaxCut".** In: *ACM Transactions on Quantum Computing* 4.3 (Apr. 2023), pp. 1–15. ISSN: 2643-6817. DOI: 10.1145/3584706. URL: http://dx.doi.org/10.1145/3584706.

[14] Ruslan Shaydulin et al. **"QAOAKit: A Toolkit for Reproducible Study, Application, and Verification of the QAOA".** In: *2021 IEEE/ACM Second International Workshop on Quantum Computing Software (QCS)*. IEEE, Nov. 2021. DOI: 10.1109/qcs54837.2021.00011. URL: https://doi.org/10.1109/qcs54837.2021.00011.

[15] P. W. Shor. **"Algorithms for quantum computation: discrete logarithms and factoring".** In: *Proceedings 35th Annual Symposium on Foundations of Computer Science*. 1994, pp. 124–134.

[16] Shree Hari Sureshbabu et al. **"Parameter Setting in Quantum Approximate Optimization of Weighted Problems".** In: *Quantum* 8 (Jan. 2024), p. 1231. ISSN: 2521-327X. DOI: 10.22331/q-2024-01-18-1231. URL: https://doi.org/10.22331/q-2024-01-18-1231.

[17] David H Wolpert and William G Macready. **"No free lunch theorems for optimization".** In: *IEEE transactions on evolutionary computation* 1.1 (1997), pp. 67–82.

[18] Leo Zhou et al. **"Quantum Approximate Optimization Algorithm: Performance, Mechanism, and Implementation on Near-Term Devices".** In: *Phys. Rev. X* 10 (2 June 2020), p. 021067. DOI: 10.1103/PhysRevX.10.021067. URL: https://link.aps.org/doi/10.1103/PhysRevX.10.021067.