Multistart Methods for Quantum Approximate Optimization

Ruslan Shaydulin, Clemson University
Ilya Safro, Clemson University
Jeffrey Larson, Argonne National Lab

Outline

- Quick intro to Quantum Computing
- QAOA definition and the challenge of QAOA parameter optimization
- Our 1st contribution — multistart methods applied to QAOA
- Our 2nd contribution — an extensive study of the performance of derivative-free methods for QAOA parameter optimization
- Our 3rd contribution — extend previous results on QAOA parameter reusing

Quantum Computing 101

- The state of a qubit is a vector in two-dimensional complex vector space span by two basis states $|1\rangle, |0\rangle$:

$$|\psi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha\begin{bmatrix}1\\0\end{bmatrix} + \beta\begin{bmatrix}0\\1\end{bmatrix}$$

- If we measure a qubit, we get 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$

- The state of an two-qubit system:

$$|\psi\rangle = \alpha_{00}|00\rangle + \alpha_{01}|01\rangle + \alpha_{10}|10\rangle + \alpha_{11}|11\rangle$$

Quantum Computing 101

- The state of a qubit is a vector in two-dimensional complex vector space span by two basis states $|1\rangle, |0\rangle$:

$$|\psi\rangle = \alpha |0\rangle + \beta |1\rangle = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix}$$

- If we measure a qubit, we get 0 with probability $|\alpha|^2$ and 1 with probability $|\beta|^2$

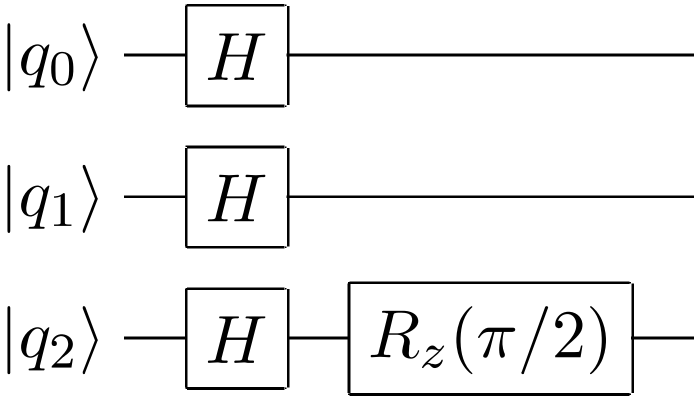
- The state of an n-qubit system:
- Note that we need 2^n complex numbers to describe an n-qubit system

$$\begin{aligned} |\psi\rangle \quad &= \quad \alpha_1 |0000000000000000000000000\rangle \\ &+ \quad \alpha_2 |0000000000000000000000001\rangle \\ &\quad\quad \ldots \\ &+ \quad \alpha_{2^n-1} |1111111111111111111111110\rangle \\ &+ \quad \alpha_{2^n} |1111111111111111111111111\rangle \end{aligned}$$

Quantum Computing 101

- Computation is performed by applying *gates* (unitary matrices):

$$H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}; \quad R_z(\theta) \equiv e^{-i\theta\hat{\sigma}^z/2} = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$$

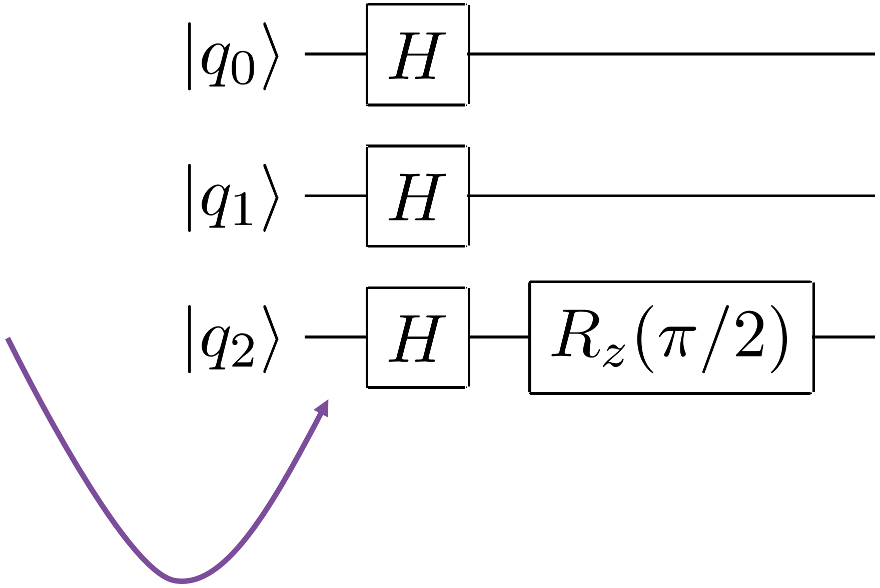


Quantum Computing 101

- Computation is performed by applying *gates* (unitary matrices):

$$H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}; \quad R_z(\theta) \equiv e^{-i\theta\hat{\sigma}^z/2} = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$$

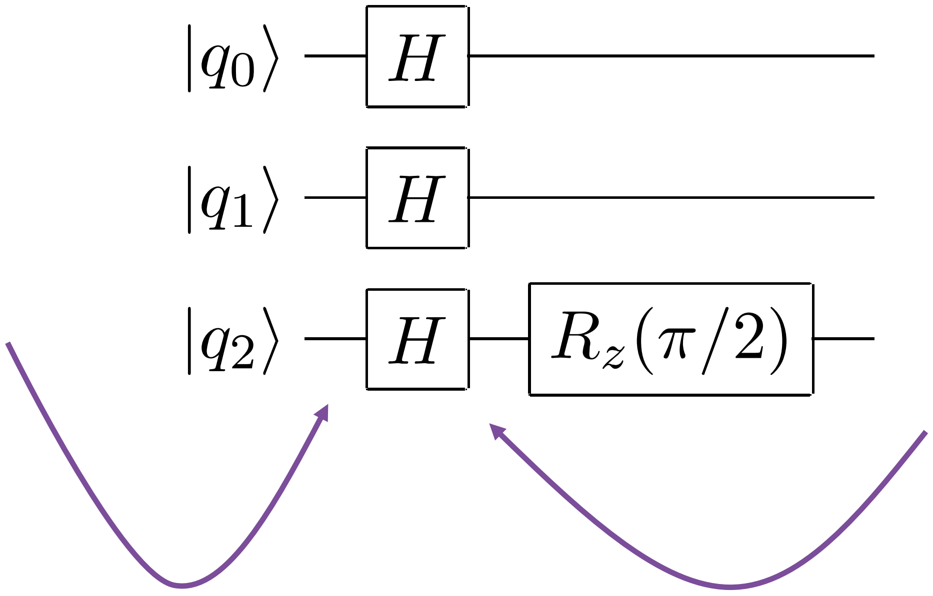
State: $$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$



Quantum Computing 101

$$H \equiv \frac{1}{\sqrt{2}}\begin{bmatrix}1 & 1\\ 1 & -1\end{bmatrix}; \quad R_z(\theta) \equiv e^{-i\theta\hat{\sigma}^z/2} = \begin{bmatrix}e^{-i\theta/2} & 0\\ 0 & e^{i\theta/2}\end{bmatrix}$$

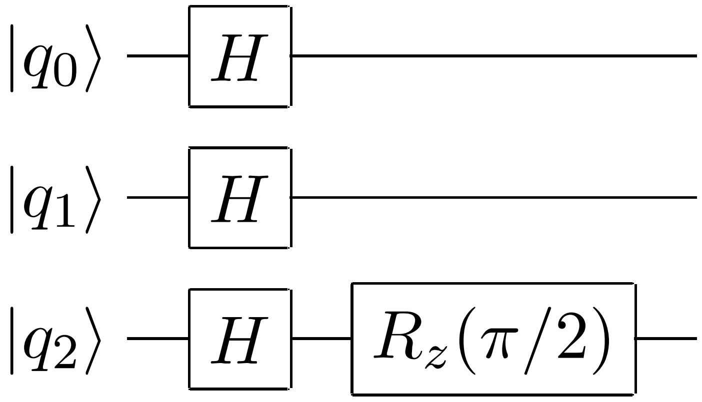
State: $\begin{bmatrix}\alpha_1\\ \alpha_2\\ \alpha_3\\ \alpha_4\\ \alpha_5\\ \alpha_6\\ \alpha_7\\ \alpha_8\end{bmatrix} = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix}$



$$\frac{1}{\sqrt{8}}\begin{bmatrix}1\\1\\1\\1\\1\\1\\1\\1\end{bmatrix}$$

Quantum Computing 101

$$H \equiv \frac{1}{\sqrt{2}} \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}; \quad R_z(\theta) \equiv e^{-i\theta \hat{\sigma}^z/2} = \begin{bmatrix} e^{-i\theta/2} & 0 \\ 0 & e^{i\theta/2} \end{bmatrix}$$



State: $$\begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \alpha_3 \\ \alpha_4 \\ \alpha_5 \\ \alpha_6 \\ \alpha_7 \\ \alpha_8 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

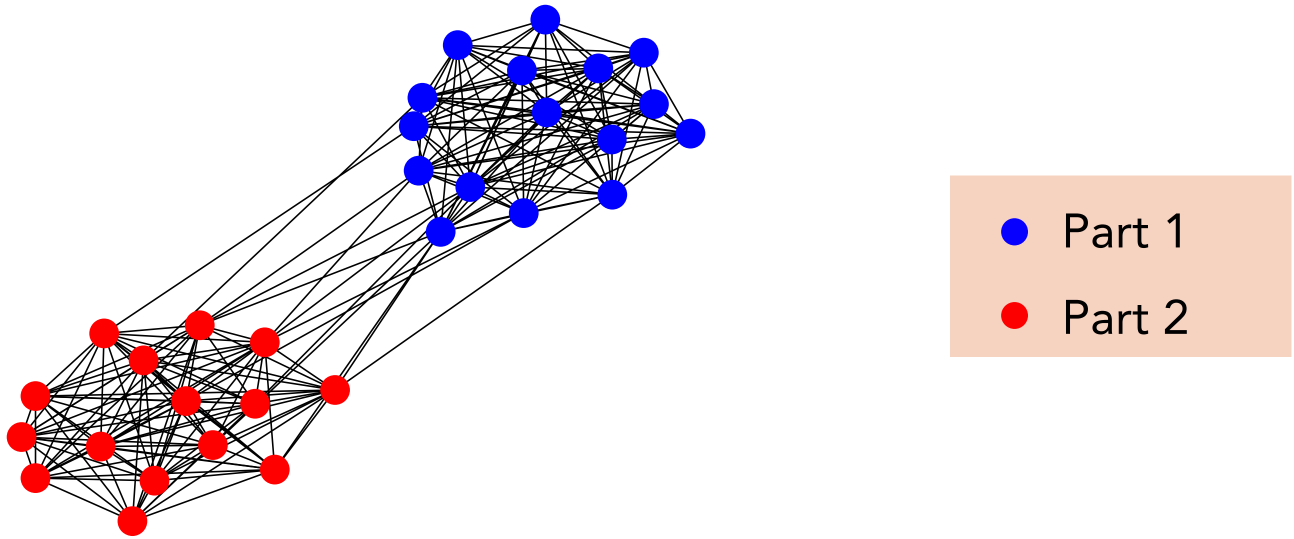
$$\frac{1}{\sqrt{8}} \begin{bmatrix} 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \\ 1 \end{bmatrix}$$

$$\frac{1}{\sqrt{8}} \begin{bmatrix} 1 \\ 1 \\ -1 \\ -1 \\ 1 \\ 1 \\ -1 \\ -1 \end{bmatrix}$$

Community Detection

Modularity maximization

- Also known as graph clustering
- Modularity is "the quality" of detected community structure in the network



Community Detection

Modularity maximization

- Modularity is "the number of edges falling within groups minus the expected number in an equivalent network with edges placed at random"
- Formally:

Actual number of edges

Expected number of edges

$$\begin{aligned} \underset{s}{\text{maximize}} \quad & \frac{1}{4|E|}\sum_{ij}(A_{ij} - \frac{k_i k_j}{2m}) s_i s_j \\ \text{subject to} \quad & s_i \in \{-1, +1\} \\ \text{where} \quad & k_i \text{ is degree of vertex } i, m = \frac{1}{2}\sum_i k_i \\ & A \text{ is an adjacency matrix of } G \end{aligned}$$

Community assignment of vertex i

NP-hard problem

Combinatorial Optimization (CO) on a quantum computer in one slide

CLEMSON UNIVERSITY

- Consider a CO problem on n variables:

$$\max_s \sum_{ij} B_{ij} s_i s_j + \sum_i h_i s_i, \qquad s_i \in \{-1, +1\}$$

- Notice that $\hat{\sigma}^z$ has eigenvalues -1, +1 with eigenvectors $|1\rangle, |0\rangle$
- We can construct the following 2^nx2^n Hermitian matrix (Hamiltonian) such that its eigenvector (eigenstate) with the largest eigenvalue (energy) corresponds to the solution of the original problem:

$$\sum_{ij} B_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z + \sum_i h_i \hat{\sigma}_i^z$$

Combinatorial Optimization (CO) on a quantum computer in one slide

- Consider a CO problem on n variables:

$$\max_s \sum_{ij} B_{ij} s_i s_j + \sum_i h_i s_i, \qquad s_i \in \{-1, +1\}$$

- Notice that $\hat{\sigma}^z$ has eigenvalues -1, +1 with eigenvectors $|1\rangle, |0\rangle$
- We can construct the following 2^nx2^n Hermitian matrix (Hamiltonian) such that its eigenvector (eigenstate) with the largest eigenvalue (energy) corresponds to the solution of the original problem:

$$\sum_{ij} B_{ij} \hat{\sigma}_i^z \hat{\sigma}_j^z + \sum_i h_i \hat{\sigma}_i^z$$

- Now all we have to do is prepare this eigenstate!

Quantum Approximate Optimization Algorithm (QAOA)

- QAOA prepares a parameterized "trial" (ansatz) state of the form:

$$
\begin{aligned}
|\psi(\boldsymbol{\theta})\rangle &= |\psi(\boldsymbol{\beta},\boldsymbol{\gamma})\rangle \\
&= e^{-i\beta_p \hat{H}_M} e^{-i\gamma_p \hat{H}_C} \cdots e^{-i\beta_1 \hat{H}_M} e^{-i\gamma_1 \hat{H}_C} \, |+\rangle^{\otimes n} .
\end{aligned}
$$

- Then a classical optimizer is used to vary the parameters $\boldsymbol{\beta},\boldsymbol{\gamma}$ to maximize:

$$f(\boldsymbol{\beta},\boldsymbol{\gamma}) = \langle\psi(\boldsymbol{\beta},\boldsymbol{\gamma})| \, \hat{H}_C \, |\psi(\boldsymbol{\beta},\boldsymbol{\gamma})\rangle .$$

Quantum Approximate Optimization Algorithm (QAOA)

- QAOA prepares a parameterized "trial" (ansatz) state of the form:

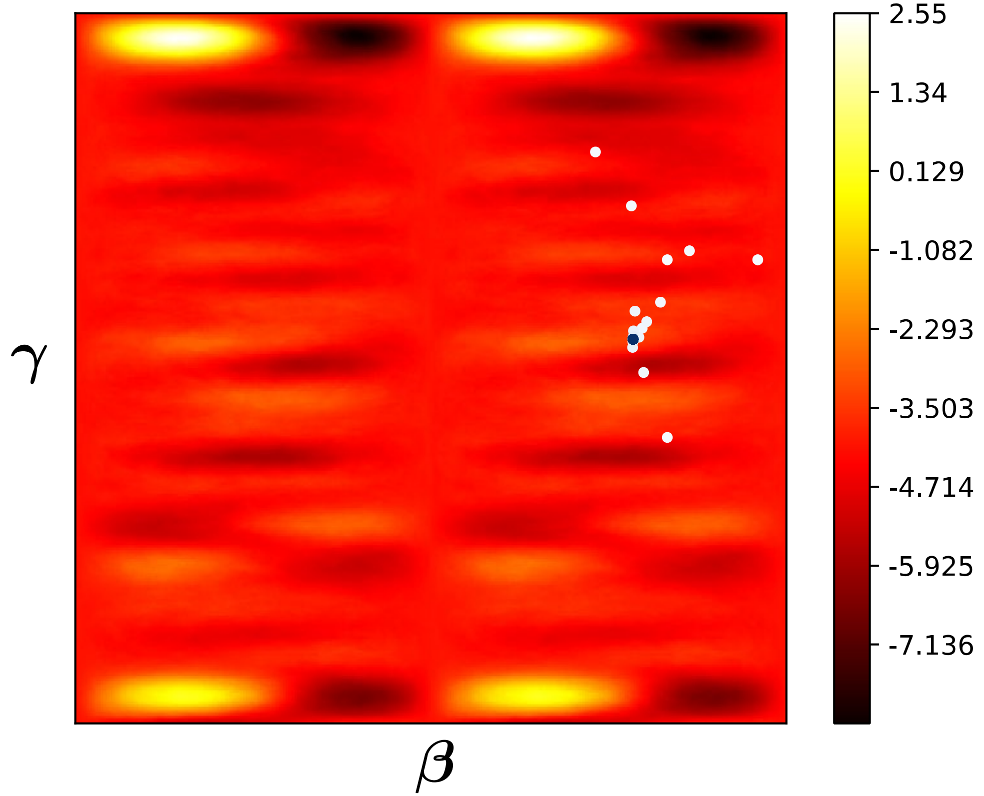
$$|\psi(\boldsymbol{\theta})\rangle = |\psi(\boldsymbol{\beta},\boldsymbol{\gamma})\rangle$$
$$= e^{-i\beta_p \hat{H}_M} e^{-i\gamma_p \hat{H}_C} \dots e^{-i\beta_1 \hat{H}_M} e^{-i\gamma_1 \hat{H}_C} |+\rangle^{\otimes n} .$$

- Note that for p → ∞ QAOA can *at least* exactly approximate adiabatic quantum evolution and can therefore find the exact optimal solution
- For small p, picture is more mixed, but there is some indication of the potential for quantum advantage

QAOA parameter optimization is hard

- The parameter space is highly nonconvex and contains many low-quality, nondegenerate local optima

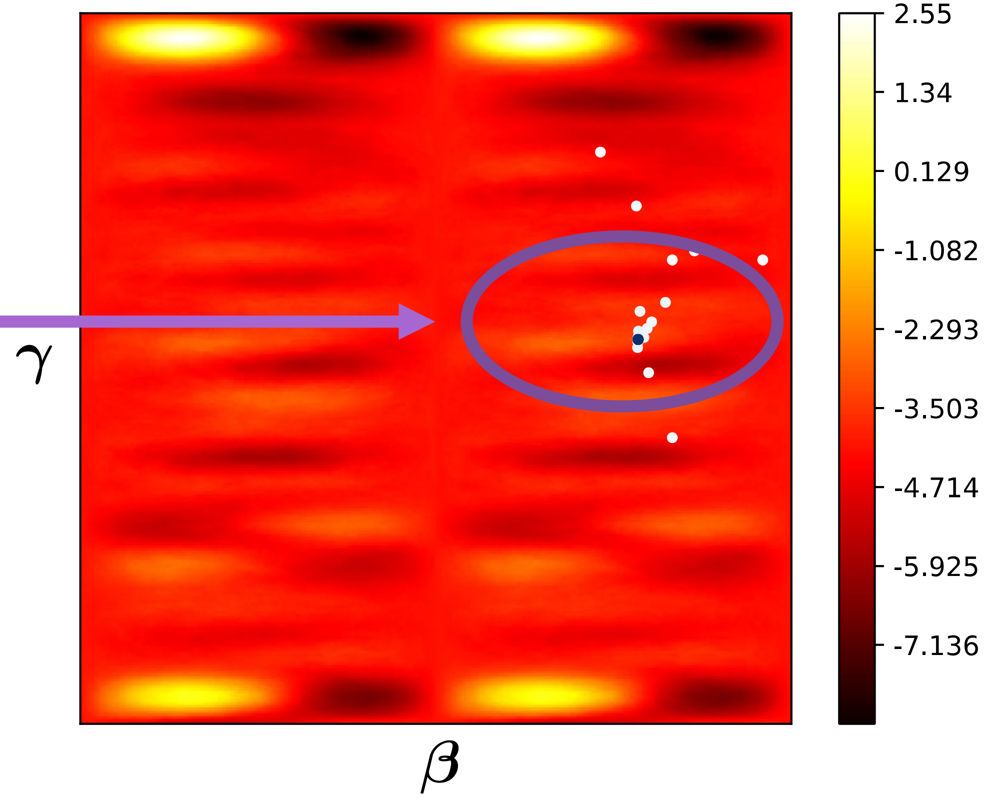
- Local optimizers get stuck in local optima



$$f(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \langle \psi(\boldsymbol{\beta}, \boldsymbol{\gamma}) | \hat{H}_C | \psi(\boldsymbol{\beta}, \boldsymbol{\gamma}) \rangle .$$

QAOA parameter optimization is hard

- The parameter space is highly nonconvex and contains many low-quality, nondegenerate local optima
- Local optimizers get stuck in local optima



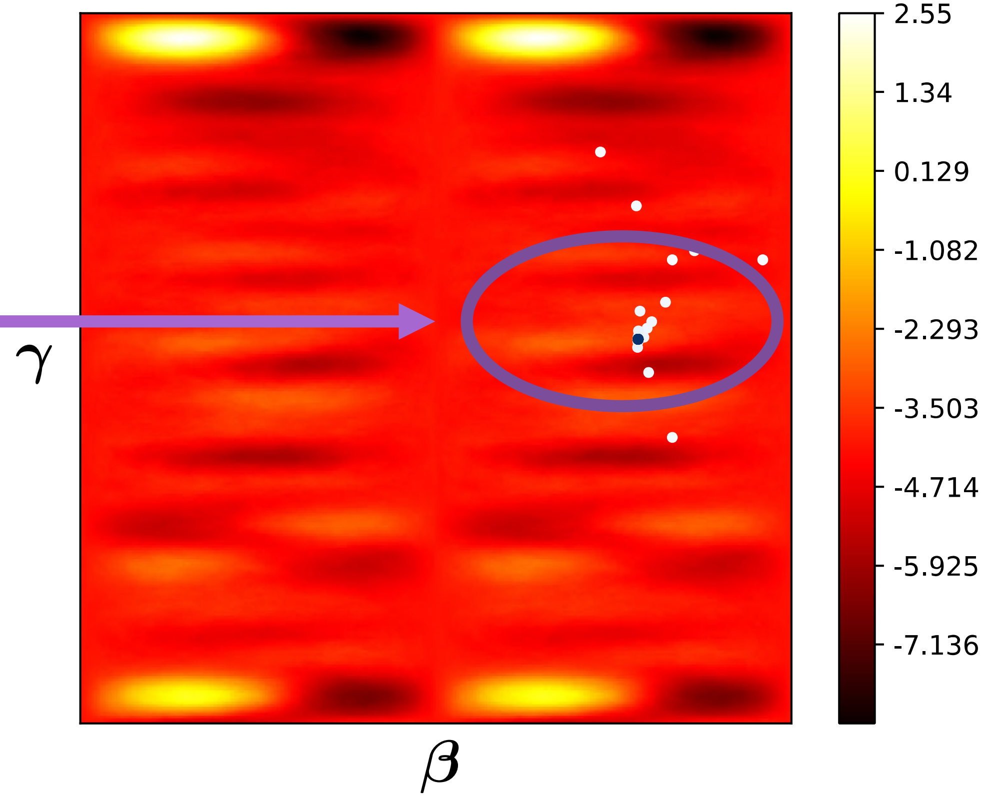
$$f(\boldsymbol{\beta}, \boldsymbol{\gamma}) = \langle \psi(\boldsymbol{\beta}, \boldsymbol{\gamma}) | \hat{H}_C | \psi(\boldsymbol{\beta}, \boldsymbol{\gamma}) \rangle .$$

QAOA parameter optimization is hard

- The parameter space is highly nonconvex and contains many low-quality, nondegenerate loc optima

- Local optimizers get stuck in local optima

- Our solution: multistart methods

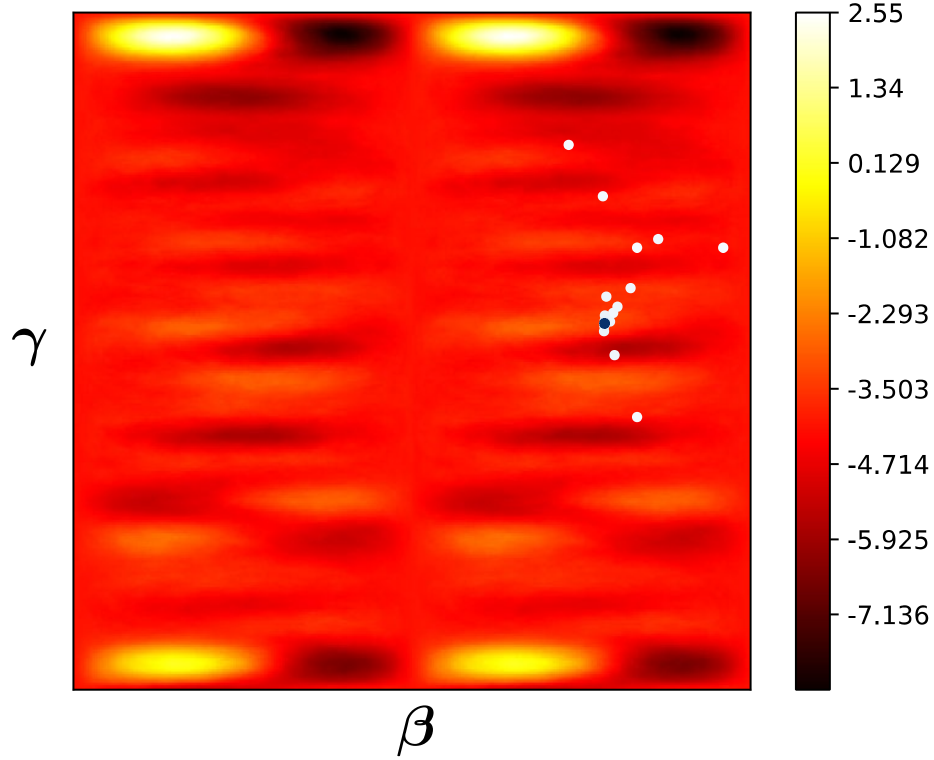


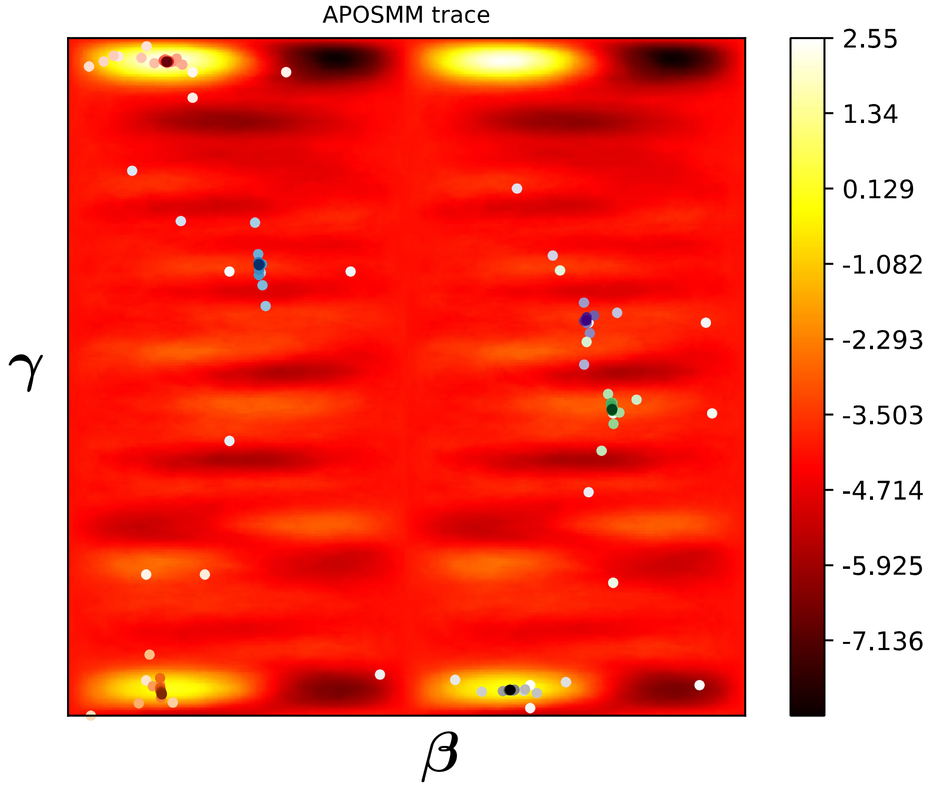
$$f(\boldsymbol{\beta},\boldsymbol{\gamma}) = \langle \psi(\boldsymbol{\beta},\boldsymbol{\gamma})|\, \hat{H}_C \,|\psi(\boldsymbol{\beta},\boldsymbol{\gamma})\rangle\,.$$

Multistart methods (APOSMM)

- Traditional approach: start local methods from different initial parameters
- Problem with traditional approach: the same optimum might be identified by multiple local optimization runs, resulting in unnecessary function evaluations
- APOSSM:
  - Starts runs from the points that do not have a better point within an algorithmically controlled neighborhood
  - Considers both the initially sampled points as well as the points generated by an ensemble of local optimization runs
- Note that APOSSM still needs a local optimizer – we choose BOBYQA as it performs best on our benchmarks

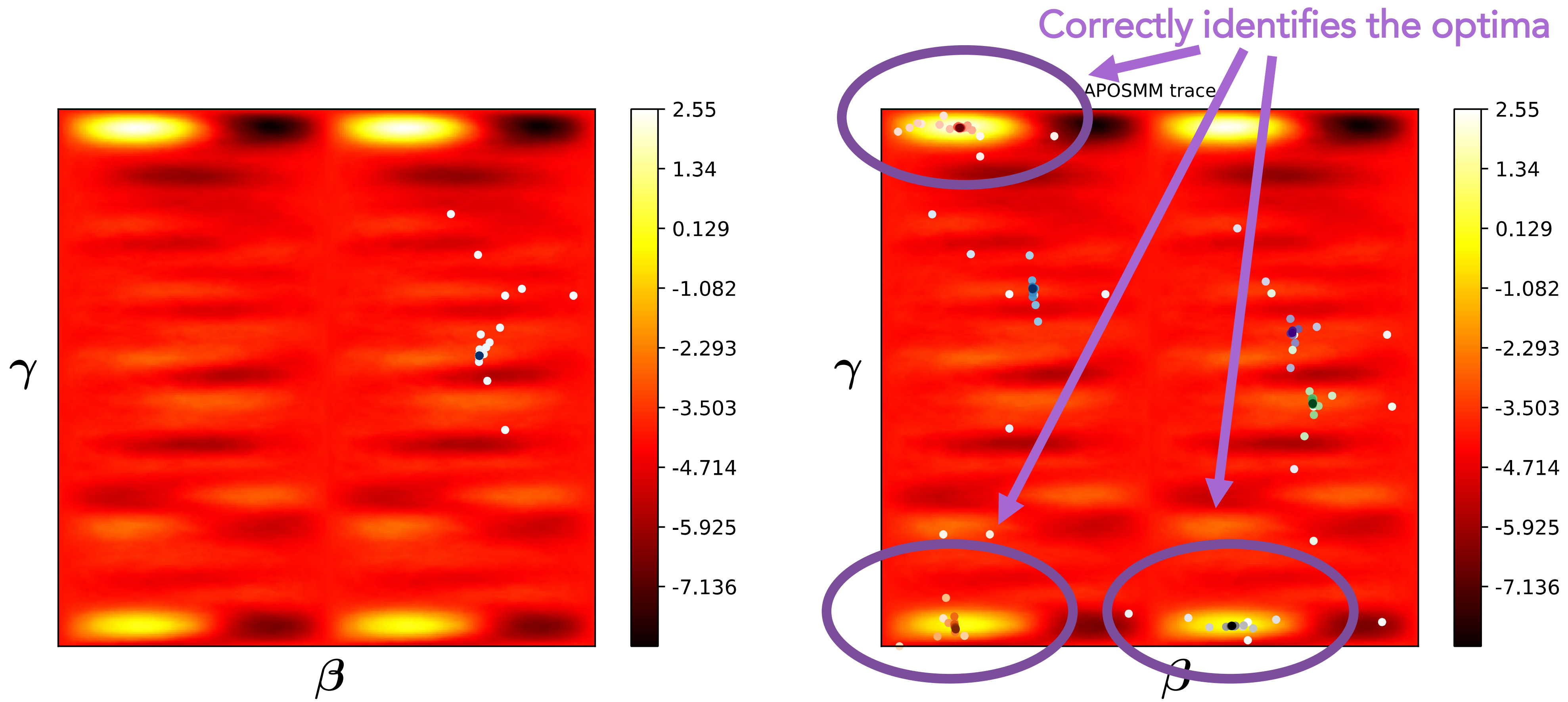
Multistart methods (APOSMM)





- APOSMM+BOBYQA identifies better optima with the same budget of function evaluations

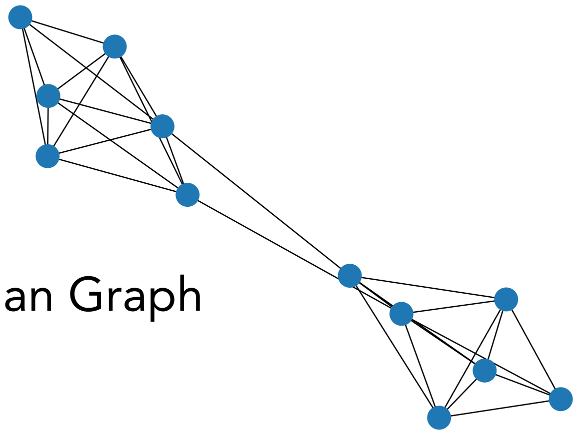
Multistart methods (APOSMM)



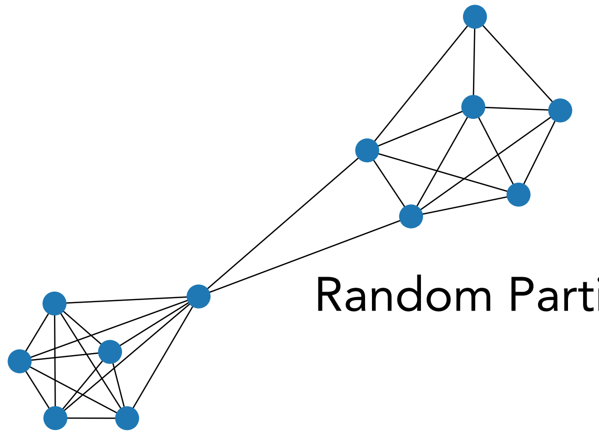
- APOSMM+BOBYQA identifies better optima with the same budget of function evaluations

Benchmark

- Modularity maximization on six synthetic graphs with community structure: three instances of connected caveman graph and three instances of random partition graph.
- All graphs have between 10 and 12 vertices
- Compare with 6 state-of-the-art derivative-free optimization methods
- All methods are given a budget of 1,000 function evaluations
- Each problem instance is run from 10 different starting points



Connected Caveman Graph



Random Partition Graph

Note on the choice of the budget of function evaluations

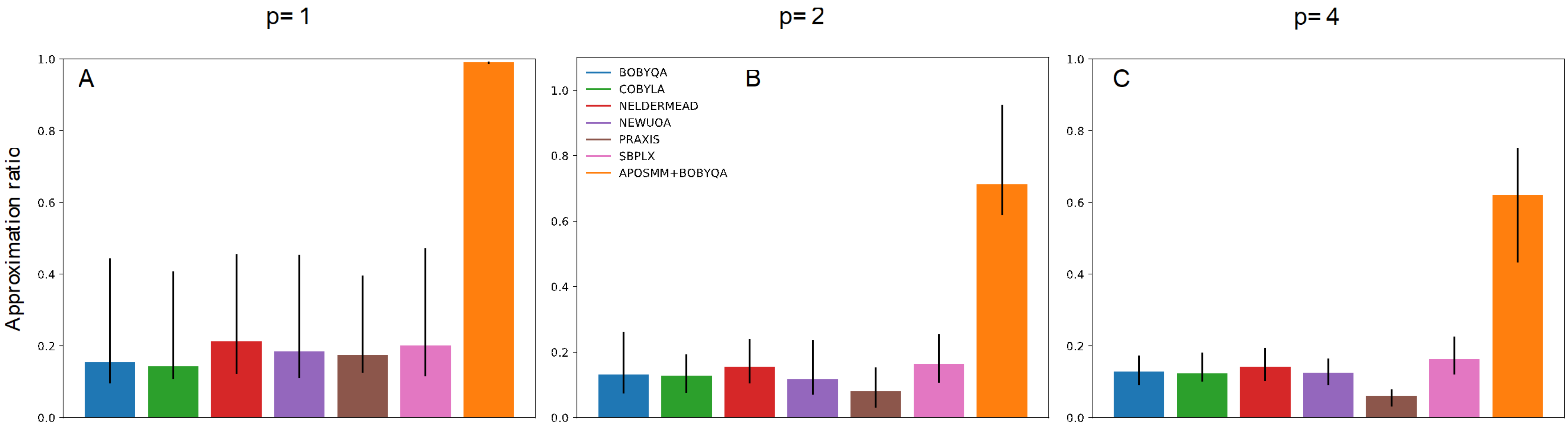
- We follow the estimates in Guerreschi et al, *Nature Scientific Reports* 2019
- Assume 1 millisecond for one "shot" (measurement of the quantum system)
- Assume 1,000 measurements needed for obtaining the statistics to calculate the objective function value
- Running time:

$$(\text{time per single measurement}) \times (1{,}000 \text{ measurements per evaluation}) \times (1{,}000 \text{ evaluations}) \approx 16 \text{ min}$$

- Note that the hardware is rapidly evolving, so it is impossible to project this numbers into the future with certainty

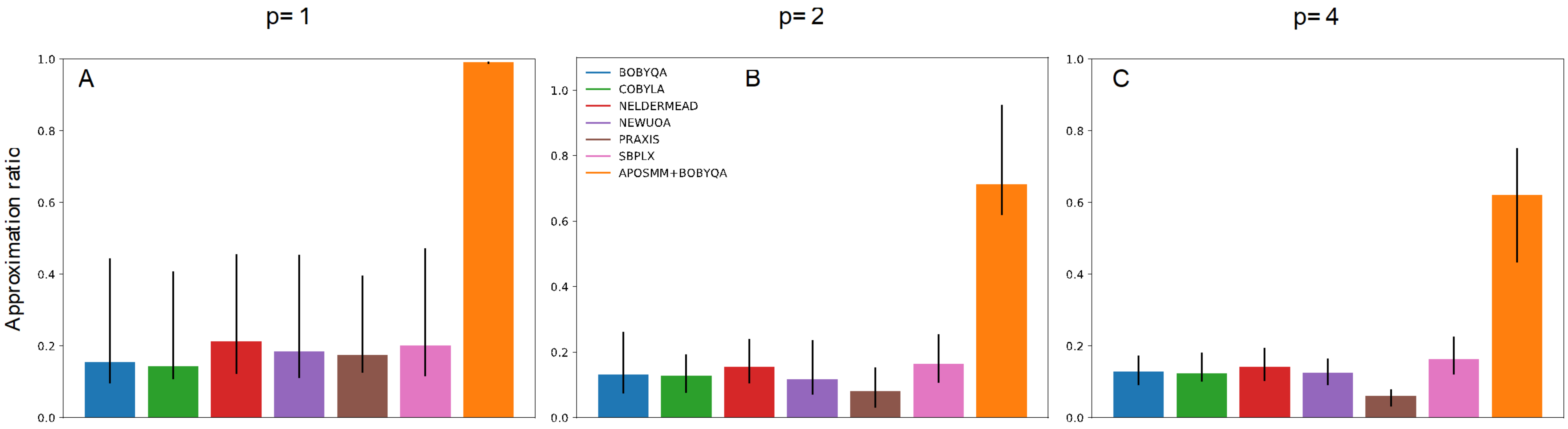
APOSMM vs no-restart local methods

- Set the tolerances of the local solvers to zero and allow them to run until convergence



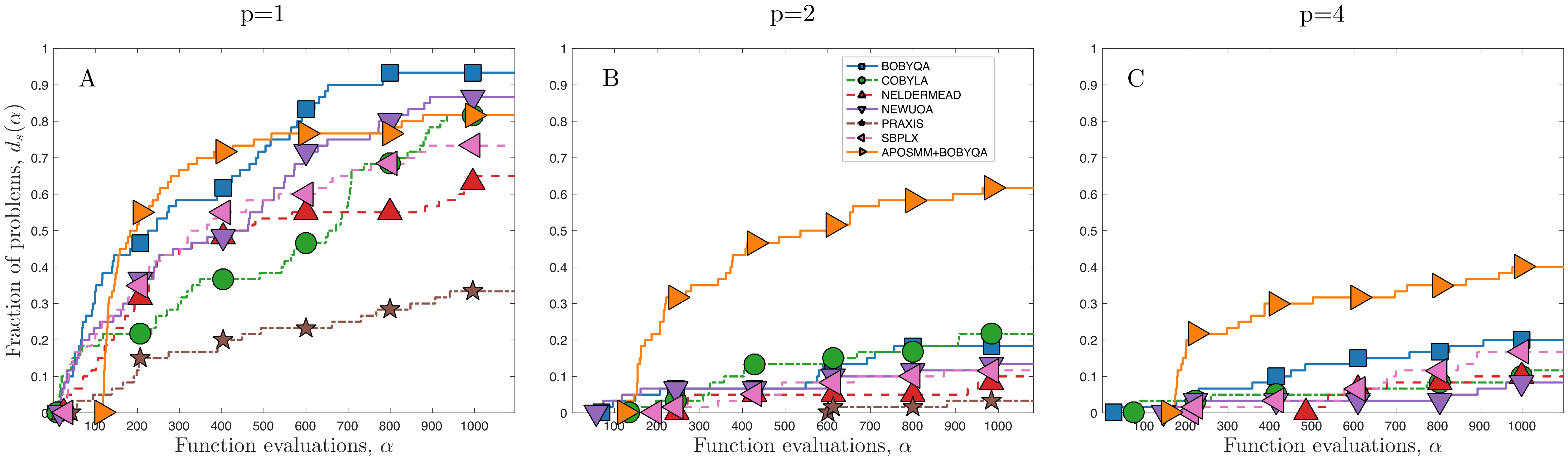
APOSMM vs no-restart local methods

- Set the tolerances of the local solvers to zero and allow them to run until convergence
- However, here APOSMM may start another local optimization run after one has converged and local methods are not restarted



APOSMM vs naïve restart local methods

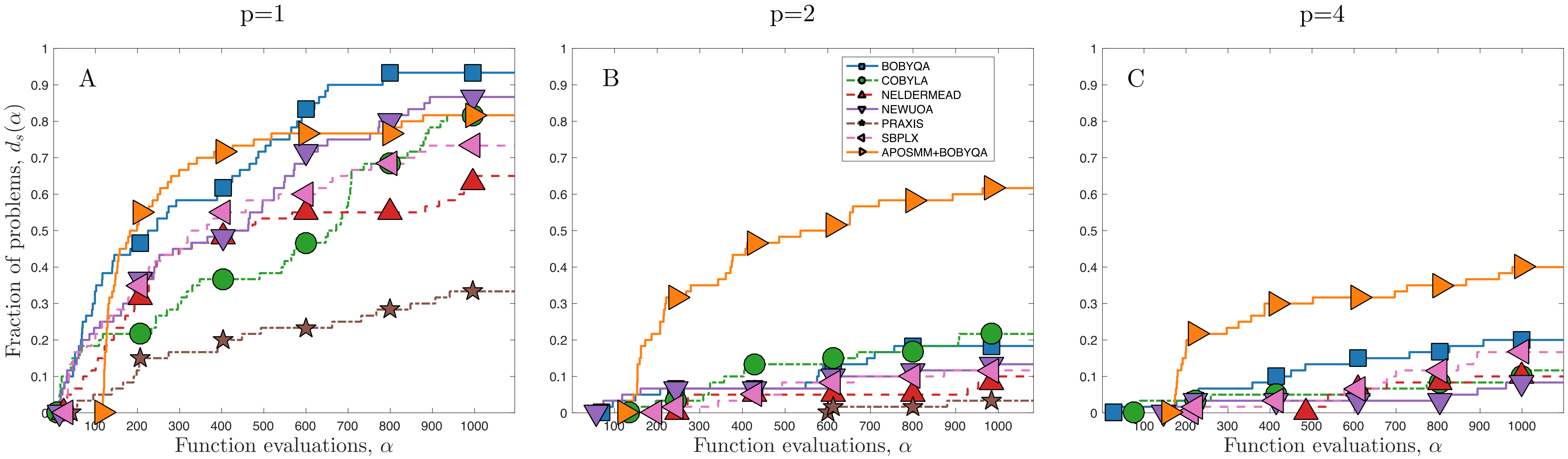
- Set the tolerances of local solvers to be the same across all seven methods
- If a local method converges before exhausting its budget of 1,000 function evaluations, it is restarted at a different random point



CLEMSON UNIVERSITY

APOSMM vs naïve restart local methods

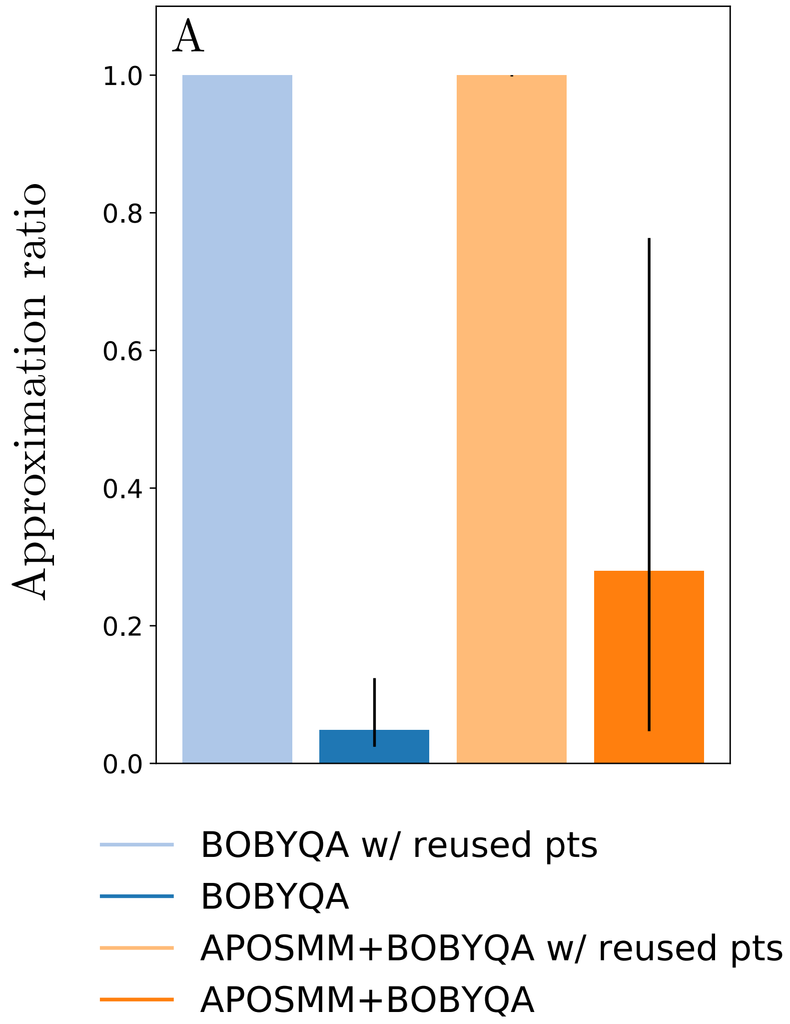
- For p=2, 4 the best-performing method (APOSMM+BOBYQA) solves only 60% and 40% of the problems, respectively
- These results indicate that even for a small number of QAOA steps, **finding good variational parameters is hard** under realistic time constraints



Is there hope?

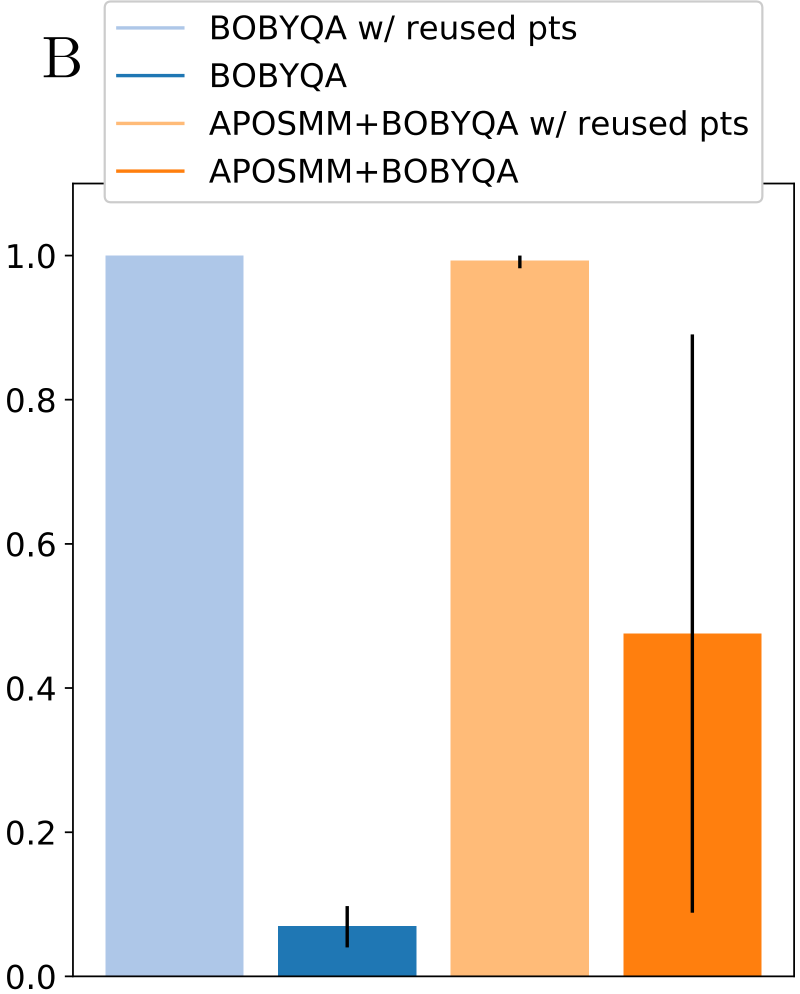
Reusing Optimal QAOA parameters

- We estimate true optimal parameters restarting BOBYQA from random points until 100,000 function evaluations have been used
- This approach identifies multiple high-quality local optima
- We then use these high-quality QAOA parameters as initial guesses for local methods and APOSMM+BOBYQA for a graph where one edge is removed, simulating a realistic "dynamic network" scenario



Reusing Optimal QAOA parameters

- We estimate true optimal parameters restarting BOBYQA from random points until 100,000 function evaluations have been used
- This approach identifies multiple high-quality local optima
- We then use these high-quality QAOA parameters as initial guesses for local methods and APOSMM+BOBYQA for a graph where one edge is removed, simulating a realistic "dynamic network" scenario
- Additionally, we simulate "worst-case" scenario by removing an edge that results in the maximum change in graph spectrum



Reusing Optimal QAOA Parameters

- Similar concentration results have been shown for MAXCUT on regular graphs (Brandao et al. 2018)
- Our results extend previous work in the following ways:
  - we show QAOA benefits from such reusing on a problem with different properties (modularity community detection), where the number of clauses in which a variable participates is not bounded
  - we consider a "worst-case" scenario
- Amortizing the cost of parameter optimization can drastically reduce the cost of running QAOA:

$$\begin{aligned} (\text{1ms per measurement}) &\quad \times \\ (\text{1,000 measurements per evaluation}) &\quad \times \\ (\text{10 evaluations to locally refine the solution}) &\quad \approx 10 \text{ sec} \end{aligned}$$

Conclusions

- Directly optimizing QAOA parameters is hard
- Multistart APOSMM approach is capable of identifying better local minima within the same budget of function evaluations than naïve local methods
- In this work, we focused on derivative-free methods, but our approach is trivially extendable to gradient-based methods by using a gradient-based local method within APOSMM
- Amortizing the cost of finding optimal QAOA parameters can make the projected running time competitive with classical state-of-the-art solvers
- Machine learning methods can be helpful – stay tuned for more results coming soon!

Conclusions

- Directly optimizing QAOA parameters is hard
- Multistart APOSMM approach is capable of identifying better local minima within the same budget of function evaluations than naïve local methods
- In this work, we focused on derivative-free methods, but our approach is trivially extendable to gradient-based methods by using a gradient-based local method within APOSMM
- Amortizing the cost of finding optimal QAOA parameters can make the projected running time competitive with classical state-of-the-art solvers
- Machine learning methods can be helpful – stay tuned for more results coming soon!

Questions? Comments? Find me after the talk or online!
I plan to graduate in May 2020, so I'm looking for postdoc / research scientist opportunities Email:
rshaydu@clemson.edu
Web: shaydul.in
LinkedIn: https://www.linkedin.com/in/rshaydu/