A QCNN for Quantum State Preparation

Carnegie Vacation Scholarship

David Amorim

Week 5
(29/07/2024 - 02/08/2024)

Aims for the Week

The following aims were set at the last meeting (29/07/2024):

Improve Loss Function

Work on an improved version of WILL. Incorporate some phase extraction metrics (e.g. χ, ϵ) into the loss function.

Investigate Phase Extraction

Study the relationship between mismatch and the extracted phase, i.e. study the operator $\tilde{Q}^\dagger \hat{R} \tilde{Q}$.

Mitigate Barren Plateaus

Work on strategies to mitigate barren plateaus, e.g. implement layer-by-layer training.

Table of Contents

WILL Revisited

- As discussed at the meeting on 29/07, the definition of WILL (weighted $\mathsf{L_p}$ loss) was amended to:

$$\mathsf{WILL_{p,q}} = \left(\sum_k \left| x_k - y_k \right|^p + |x_k| \left| [k]_m - \Psi([k]_n) \right|^q \right)^{1/p}, \quad (1)$$

  where the changes to the previous definition are highlighted
- Testing this for different Ψ (with $L = 6$, $m = 3$ and 600 epochs) yielded the following optimal values for p, q:

$\Psi(f)$	p	q
$\sim f$	0.25	0.5
$\sim f^2$	1	1.5
Ψ_{H23}	0.75	2

Table 1: Optimal identified p, q values for WILL

Comparing SAM, WIM, and WILL

	SAM	WIM	WILL
μ	**3.4e-2**	6.0e-2	4.5e-1
σ	1.4e-1	**1.1e-1**	4.7e-1
ϵ	**1.9e-2**	9.2e-2	2.6e-1
χ	**3.2e-2**	5.1e-2	3.7e-1
Ω	**4.46**	3.19	0.76

Table 2: Comparing loss function metrics for $\Psi(f) \sim f$ ($L = 6$, $m = 3$, 600 epochs)

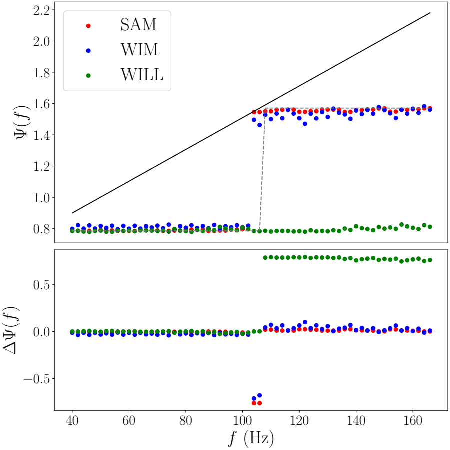

Figure 1: Comparing extracted phase functions for $\Psi(f) \sim f$ ($L = 6$, $m = 3$, 600 epochs)

Comparing SAM, WIM, and WILL

	SAM	WIM	WILL
μ	**1.9e-1**	2.3e-1	6.6e-1
σ	1.2e-1	**1.0e-1**	4.1e-1
ϵ	2.2e-1	4.2e-1	**2.8e-2**
χ	**1.9e-1**	2.0e-1	6.1e-1
Ω	**1.39**	1.05	0.57

Table 3: Comparing loss function metrics for $\Psi(f) \sim f^2$ ($L = 6$, $m = 3$, 600 epochs)

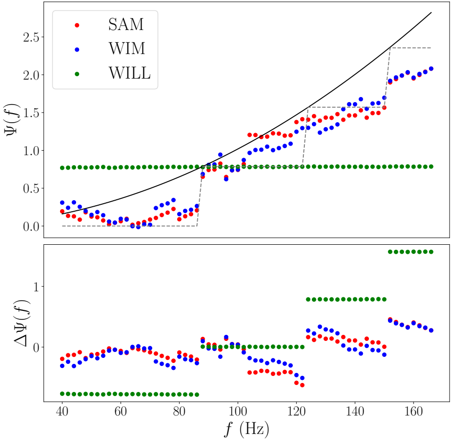

Figure 2: Comparing extracted phase functions for $\Psi(f) \sim f^2$ ($L = 6$, $m = 3$, 600 epochs)

Comparing SAM, WIM, and WILL

	SAM	WIM	WILL
μ	**6.8e-2**	8.4e-2	7.6e-2
σ	1.8e-1	**1.2e-1**	2.6e-1
ϵ	4.5e-2	1.8e-1	**7.3e-3**
χ	7.4e-2	1.0e-1	**6.2e-2**
Ω	**2.75**	2.07	2.48

Table 4: Comparing loss function metrics for Ψ_{H23} ($L = 6$, $m = 3$, 600 epochs)

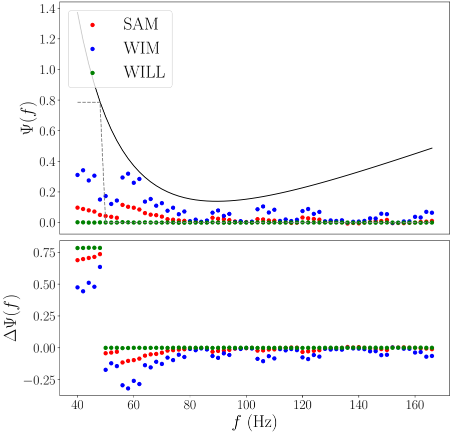

Figure 3: Comparing extracted phase functions for Ψ_{H23} ($L = 6$, $m = 3$, 600 epochs)

Other Approaches

- Attempts to define a loss function based directly on $\hat{Q}^\dagger \hat{R} \hat{Q}$, e.g. minimising χ, were unsuccessful
- This is due to the *qiskit machine learning* environment being build around sampler primitives which return quasi-probabilities instead of probability amplitudes
- Thus, phases cannot be directly taken into account for gradient calculation
- A possible work-around could be to switch to a QCNN based on an estimator primitive, which calculates the expectation value of an observable w.r.t to the state prepared by the network
- This would require the construction of an appropriate operator (note: qiskit supports non-Hermitian observables)

An Estimator-based PQC

- Let $|\tilde{\phi}\rangle$ be the n-qubit state produced by the PQC:

$$|\tilde{\phi}\rangle = \sum_k \tilde{A}(k) e^{i\tilde{\Psi}(k)} |k\rangle \tag{2}$$

- The desired output state is

$$|\phi\rangle = \sum_k A(k) e^{i\Psi(k)} |k\rangle \tag{3}$$

- An estimator-based optimiser calculates the loss and gradients for each epoch based on the expectation value

$$\mathbb{E}(\tilde{\phi}) \equiv \langle \tilde{\phi} | \hat{O} | \tilde{\phi} \rangle = \sum_{k,k'} \tilde{A}(k') \tilde{A}(k) \exp\left(i \left[\tilde{\Psi}(k) - \tilde{\Psi}(k') \right] \right) \langle k' | \hat{O} | k \rangle , \tag{4}$$

for some operator $\hat{O}$

An Estimator-based PQC

- Now construct $\hat{O}$ such that

$$\langle k'|\hat{O}|k\rangle \equiv \frac{1}{A(k')A(k)} \exp\left(-i\left[\Psi(k) - \Psi(k')\right]\right) \tag{5}$$

  for $A(k), A(k') \neq 0$
- Then

$$\mathbb{E}(\phi) = \sum_{k,k'} 1 = 2^{2n} \tag{6}$$

  so that we can train the network to generate $|\phi\rangle$ by minimising $|1 - \mathbb{E}(\tilde{\phi})/2^{2n}|$
- This is highly speculative and computationally very expensive even for simple PQCs due to the way custom operators are handled in qiskit
- Thus, estimator-based PQCs cannot feasibly replace the sampler-based QCNN

Loss Function: Conclusion

- The design of the *qiskit machine learning* library constrains the customisability of loss functions, in particular relating to phases
- Thus, loss functions based directly on the extracted phase factors are (apparently) impossible
- Within the limits of these constraints the best possible loss function seems to be SAM
- Beyond the unsuccessful attempts of WIM and WILL no further *ansätze* for loss functions come to mind
- For the time being, the search for an improved loss function will be put on hold

Table of Contents

Investigating Phase Extraction

- The operator $\hat{Q}$ is defined via

$$\hat{Q}\,|j\rangle\,|0\rangle = |j\rangle\,|\Psi(j)\rangle \tag{7}$$

- This leaves its action on more general input states $|j\rangle\,|k\rangle$ (with $|k\rangle \neq |0\rangle$) undetermined
- Thus, there is a family $\mathcal{Q}$ of valid implementations of $\hat{Q}$ with $|\mathcal{Q}| = (n+m)^2 - n$
- We can represent a flawed implementation, $\tilde{Q}$, of $\hat{Q}$ via $\tilde{Q} = \hat{Q} + \lambda\hat{P}$ so that

$$\tilde{Q}\hat{R}\tilde{Q} = \hat{Q}^\dagger\hat{R}\hat{Q} + \lambda\left[\hat{Q}^\dagger\hat{R}\hat{P} + \hat{P}^\dagger\hat{R}\hat{Q}\right] + \lambda^2\left[\hat{P}^\dagger\hat{R}\hat{P}\right] \tag{8}$$

- Beyond these very general observations no analytical insight into the problem was gained

Visualising Phase Extraction: Amplitudes

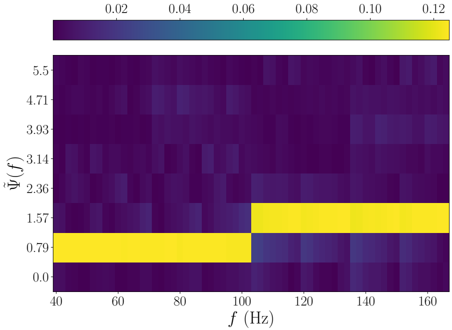

Figure 4: Amplitudes after applying $\tilde{Q}$ with $\Psi(f) \sim f$ and the input register in initial state $\hat{H}\,|0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

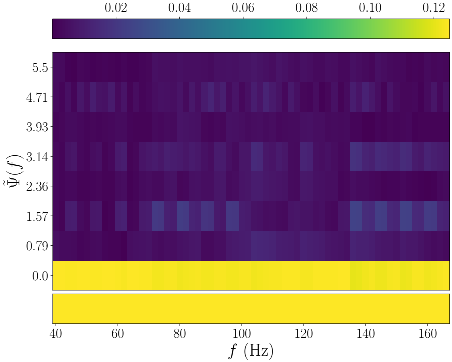

Figure 5: Amplitudes after applying $\tilde{Q}^\dagger \hat{R} \tilde{Q}$ with $\Psi(f) \sim f$ and the input register in initial state $\hat{H}\,|0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs). Target state added for reference.

Visualising Phase Extraction: Amplitudes

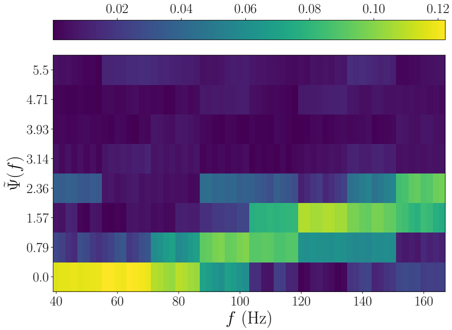

Figure 6: Amplitudes after applying $\tilde{Q}$ with $\Psi(f) \sim f^2$ and the input register in initial state $\hat{H}\,|0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

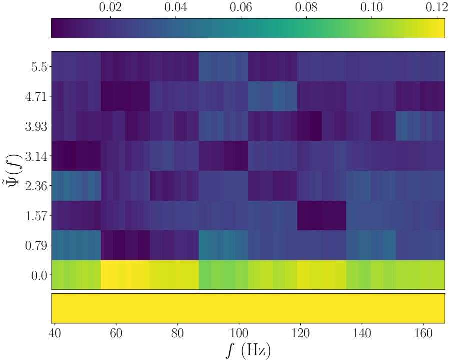

Figure 7: Amplitudes after applying $\tilde{Q}^\dagger \hat{R} \tilde{Q}$ with $\Psi(f) \sim f^2$ and the input register in initial state $\hat{H}\,|0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs). Target state added for reference.

The Effect of $\hat{U}_A$

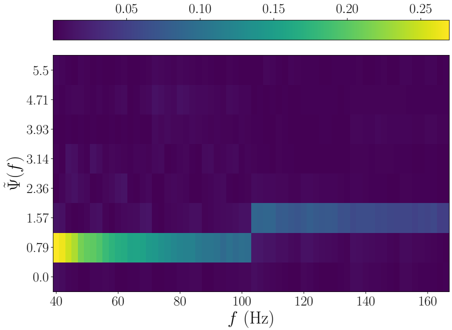

Figure 8: Amplitudes after applying $\tilde{Q}$ with $\Psi(f) \sim f$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

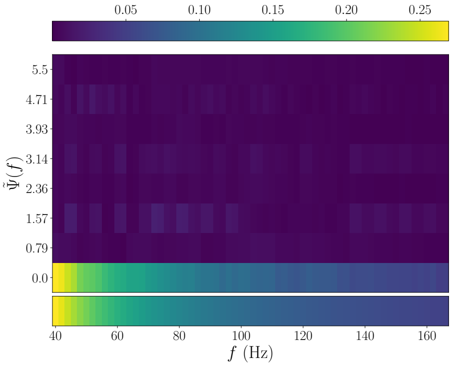

Figure 9: Amplitudes after applying $\tilde{Q}^\dagger \hat{R} \tilde{Q}$ with $\Psi(f) \sim f$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs). Target state added for reference.

The Effect of $\hat{U}_A$

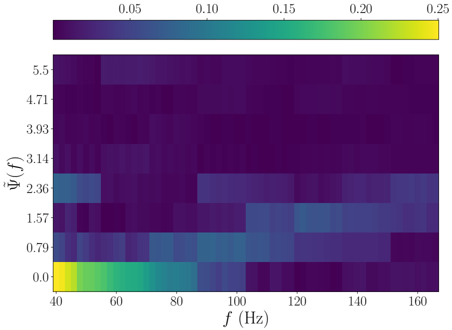

Figure 10: Amplitudes after applying $\tilde{Q}$ with $\Psi(f) \sim f^2$ and the input register in initial state $\hat{H}\,|0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

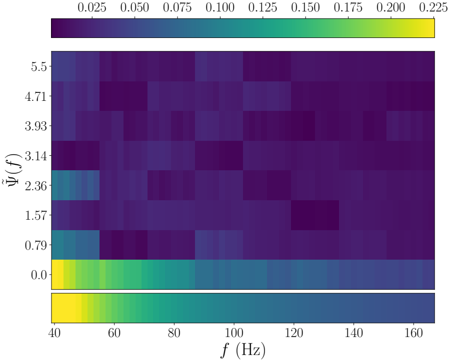

Figure 11: Amplitudes after applying $\tilde{Q}^\dagger \hat{R} \tilde{Q}$ with $\Psi(f) \sim f^2$ and the input register in initial state $\hat{U}_A\,|0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).Target state added for reference.

Visualising Phase Extraction: Phases

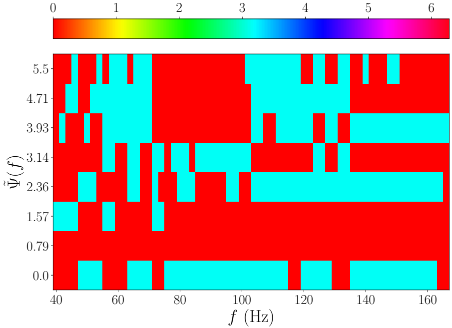

Figure 12: Phases after applying $\tilde{Q}$ with $\Psi(f) \sim f$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

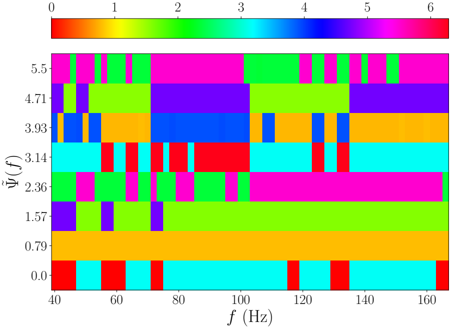

Figure 13: Phases after applying $\hat{R}\tilde{Q}$ with $\Psi(f) \sim f$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

Visualising Phase Extraction: Phases

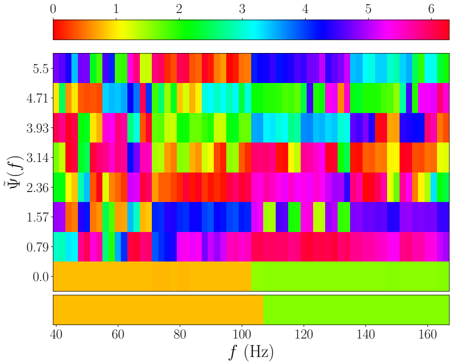

Figure 14: Phases after applying $\tilde{Q}^\dagger \hat{R} \tilde{Q}$ with $\Psi(f) \sim f$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs). Target state added for reference

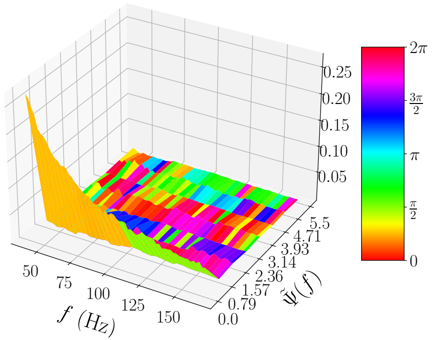

Figure 15: Phases (colour) and amplitudes (vertical axis) after applying $\tilde{Q}^\dagger \hat{R} \tilde{Q}$ with $\Psi(f) \sim f$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

Visualising Phase Extraction: Phases

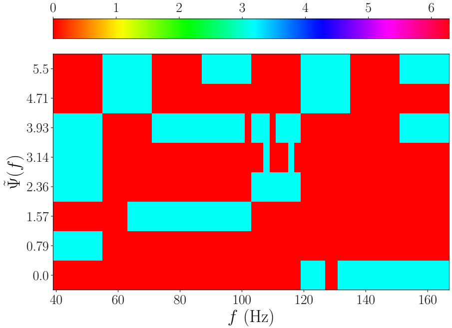

Figure 16: Phases after applying $\tilde{Q}$ with $\Psi(f) \sim f^2$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

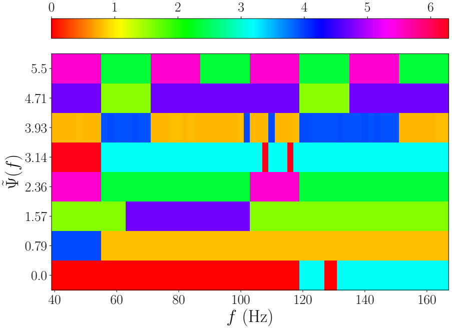

Figure 17: Phases after applying $\hat{R}\tilde{Q}$ with $\Psi(f) \sim f^2$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

Visualising Phase Extraction: Phases

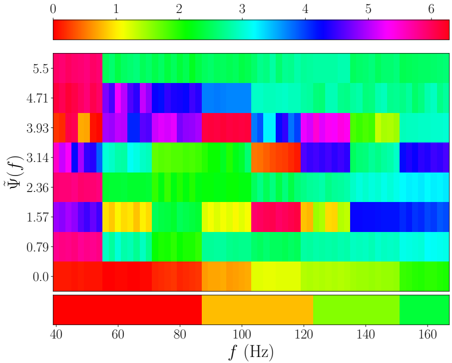

Figure 18: Phases after applying $\tilde{Q}^\dagger \hat{R} \tilde{Q}$ with $\Psi(f) \sim f^2$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs). Target state added for reference

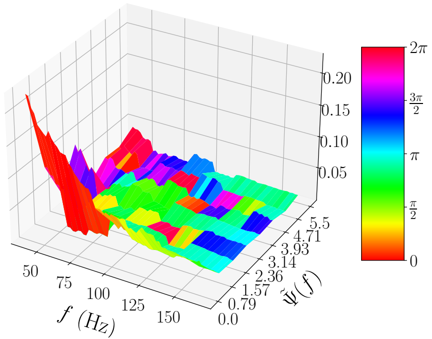

Figure 19: Phases (colour) and amplitudes (vertical axis) after applying $\tilde{Q}^\dagger \hat{R} \tilde{Q}$ with $\Psi(f) \sim f^2$ and the input register in initial state $\hat{U}_A |0\rangle$ ($L = 6$, $m = 3$, SAM, 600 epochs).

Phase Extraction: Conclusion

- As expected, the flawed implementation of $\hat{Q}$ results in $\hat{R}$ not producing an eigenstate and hence $\hat{Q}^\dagger$ in not clearing the target register
- Applying $\hat{U}_A$ to the input register, thus reducing the amplitudes of some input states, can exacerbate this issue due to a decreased 'signal-to-noise ratio'
- Interestingly, $\tilde{Q}$ induces π-phase shifts on some input states which interfere with the phases extracted by $\hat{R}$
- Unclear to which extent the extracted phase function is affected by these phase shifts

Table of Contents

Mitigating Barren Plateaus

- The most important strategy to mitigate barren plateaus seems to be a so-called warm start, also known as smart initialisation
- Common approaches include layerwise training[1], training via identity blocks[2] and training using a fast-and-slow approach[3]
- Implementing these strategies requires signifcant adaptation of the existing code base
- Thus, must first spend some time restructuring (and re-documenting) the code

[1] https://arxiv.org/pdf/2006.14904
[2] https://arxiv.org/pdf/1903.05076
[3] https://arxiv.org/pdf/2203.02464

Table of Contents

Next Steps

- Try to remove or account for the π phase shifts due to $\tilde{Q}$
- Keep re-writing code for easier implementation of barren plateau mitigation techniques
- Implement barren plateau mitigation
- ...?