

# QAMP 2021 : Operationalizing Quantum Kernels

---

Mentees : Cheryl Fillekes & Michaël Rollin

Mentor : Travis Scholten

Voice : Neural network cloned voice of Michaël Rollin



# The project

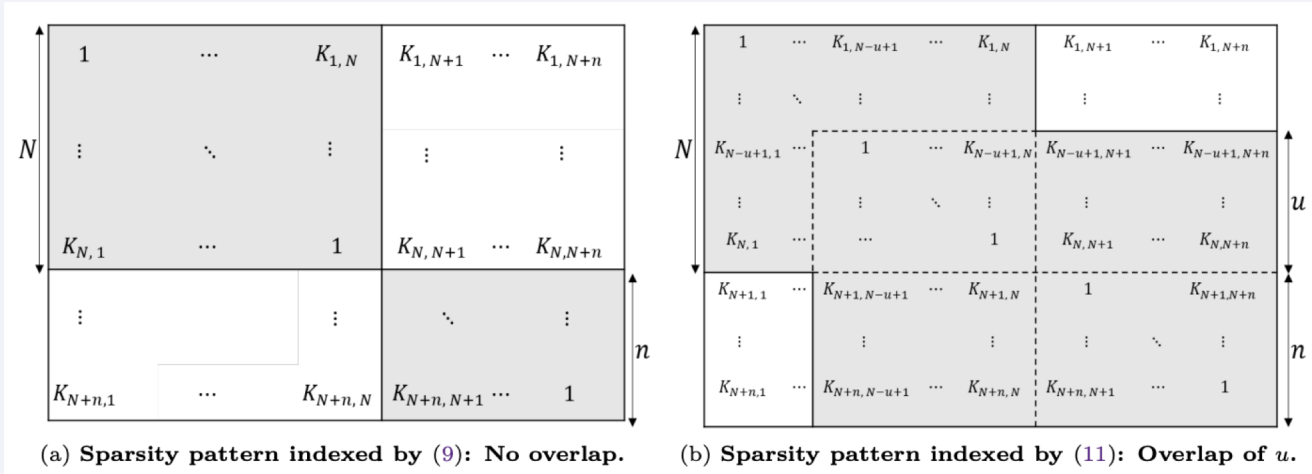
- Based of the paper : *Kernel Matrix Completion for Offline Quantum-Enhanced Machine Learning* ([2112.08449](#))
- Goal : Operationalize the workflow
- Problem : Extending quantum kernel matrices in the context of a batched, streaming data workload.



# How does it works ?



- Create matrix  $N$  and  $n$  from Quantum kernel
- Generate  $U$  from part of  $N$
- Compute the leftover with classical matrix completion



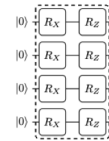
# Kernel circuit



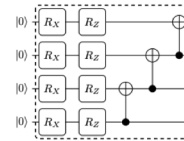
- Data  $x$  and  $y$ , create from random seed
- Template circuit from the paper : *Expressibility and entangling capability of parameterized quantum circuits for hybrid quantum-classical algorithms* ([1905.10876](https://arxiv.org/abs/1905.10876))

- Kernel circuit formula :

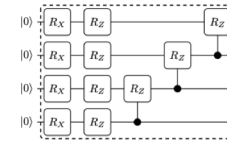
$$|\langle 0^{\otimes w} | U^\dagger(\mathbf{x}_l) U(\mathbf{x}_m) | 0^{\otimes w} \rangle|^2$$



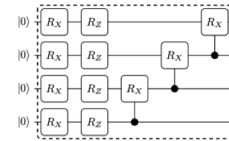
Circuit 1



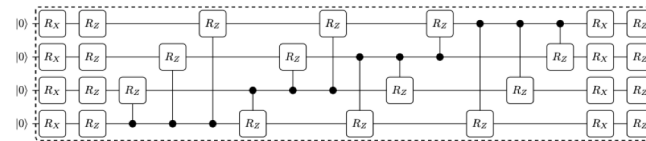
Circuit 2



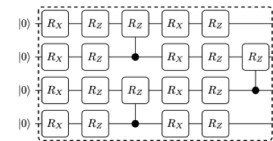
Circuit 3



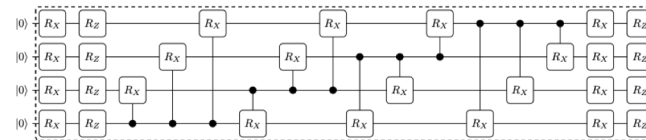
Circuit 4



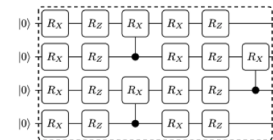
Circuit 5



Circuit 7

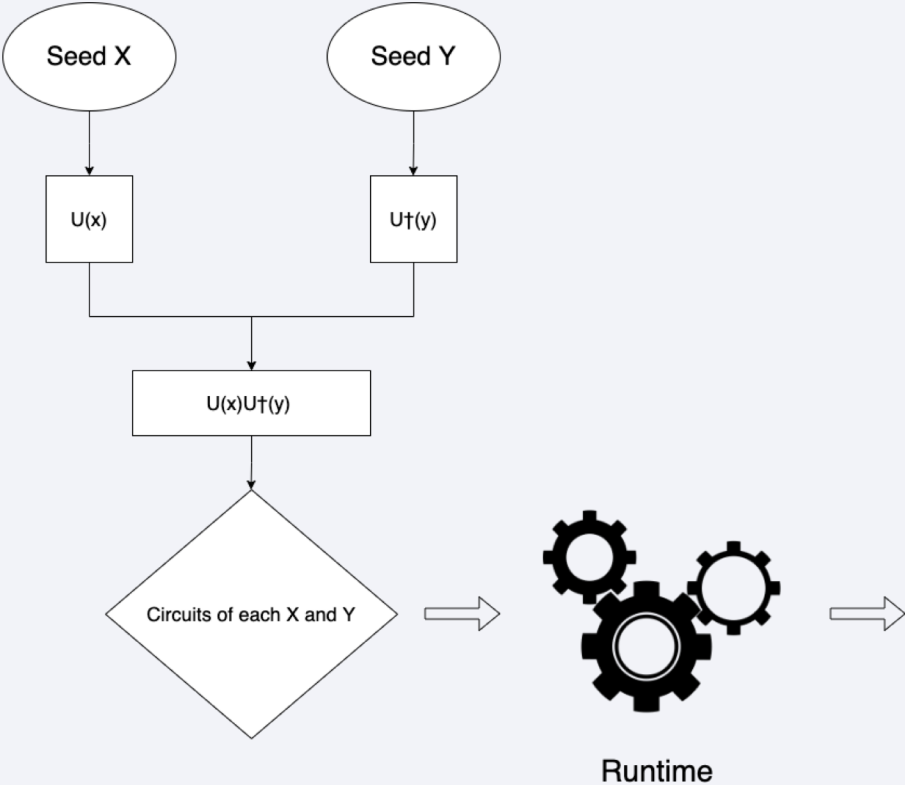


Circuit 6



Circuit 8

# Quantum kernel schematic



	width	layers	shots	seed_x	seed_y	fidelity
0	4	1	1024	0	0	1.000000
1	4	1	1024	0	1	0.613281
2	4	1	1024	0	2	0.443359
3	4	1	1024	0	3	0.453125
4	4	1	1024	0	4	0.531250
...	...	...	...	...	...	...
10196	4	1	1024	100	96	0.583008
10197	4	1	1024	100	97	0.575195
10198	4	1	1024	100	98	0.790039
10199	4	1	1024	100	99	0.725586
10200	4	1	1024	100	100	1.000000

# Matrix completion schematic



	width	layers	shots	seed_x	seed_y	fidelity
0	4	1	1024	0	0	1.000000
1	4	1	1024	0	1	0.613281
2	4	1	1024	0	2	0.443359
3	4	1	1024	0	3	0.453125
4	4	1	1024	0	4	0.531250
...	...	...	...	...	...	...
10196	4	1	1024	100	96	0.583008
10197	4	1	1024	100	97	0.575195
10198	4	1	1024	100	98	0.790039
10199	4	1	1024	100	99	0.725586
10200	4	1	1024	100	100	1.000000

[N, N] matrix

[n, n] matrix

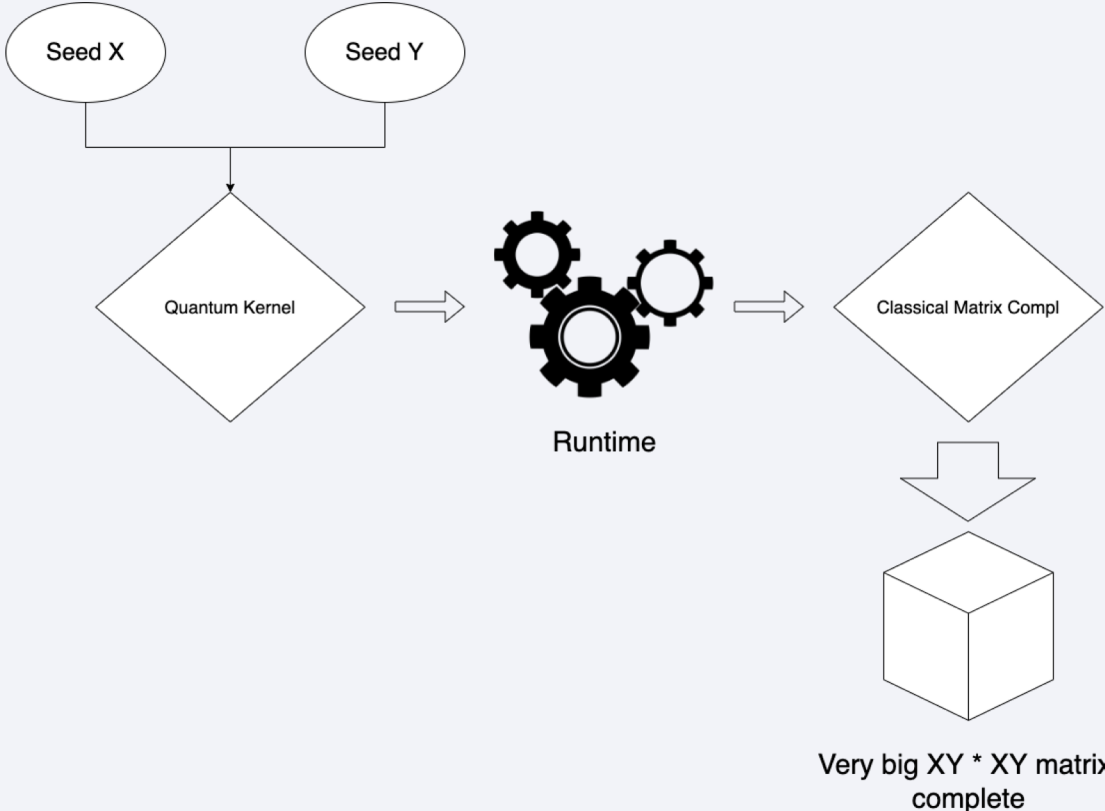
[u, u] matrix

[N+n, N+n]  
& overlap of N with u




Chordal graph

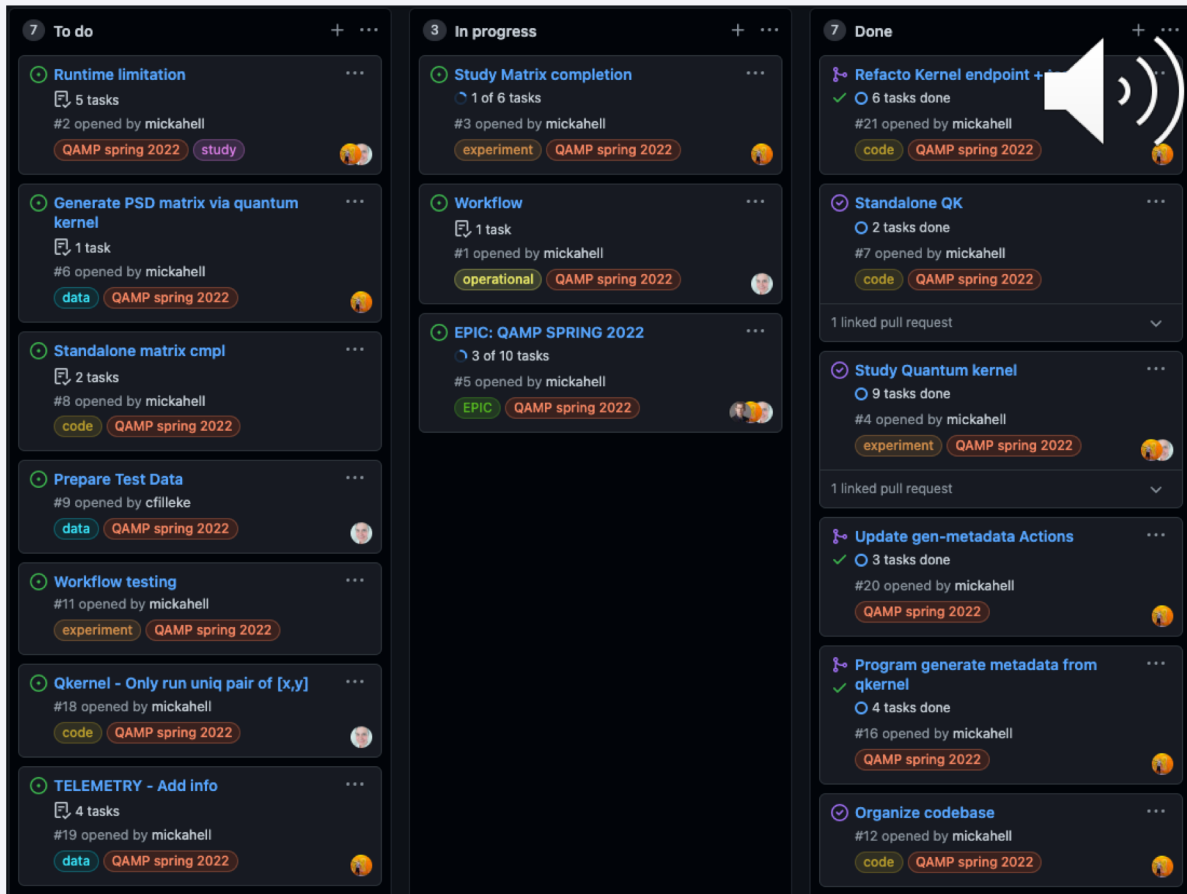
Very big XY \* XY matrix  
complete

# Functional structure



# Current state

- ❑ Python lib
  - Quantum kernel program
  - Automation
  - Generating metadata 
  - Matrix completion ✕
- ❑ Study & No-functional US
  - Documentation 
  - Runtime limitation 



The screenshot shows a Jira board with three columns: 'To do' (7 items), 'In progress' (3 items), and 'Done' (7 items). Each item includes a title, task count, assignee, and labels. A speaker icon is visible in the top right of the 'Done' column.

Column	Item Title	Task Count	Assignee	Labels
To do	Runtime limitation	5 tasks	mickahell	QAMP spring 2022, study
	Generate PSD matrix via quantum kernel	1 task	mickahell	data, QAMP spring 2022
	Standalone matrix cmpl	2 tasks	mickahell	code, QAMP spring 2022
	Prepare Test Data	9 tasks	cfilleke	data, QAMP spring 2022
	Workflow testing	11 tasks	mickahell	experiment, QAMP spring 2022
	Qkernel - Only run uniq pair of [x,y]	18 tasks	mickahell	code, QAMP spring 2022
	TELEMETRY - Add info	4 tasks	mickahell	data, QAMP spring 2022
In progress	Study Matrix completion	1 of 6 tasks	mickahell	experiment, QAMP spring 2022
	Workflow	1 task	mickahell	operational, QAMP spring 2022
	EPIC: QAMP SPRING 2022	3 of 10 tasks	mickahell	EPIC, QAMP spring 2022
Done	Refacto Kernel endpoint	6 tasks done	mickahell	code, QAMP spring 2022
	Standalone QK	2 tasks done	mickahell	code, QAMP spring 2022
	Study Quantum kernel	9 tasks done	mickahell	experiment, QAMP spring 2022
	Update gen-metadata Actions	3 tasks done	mickahell	QAMP spring 2022
	Program generate metadata from qkernel	4 tasks done	mickahell	QAMP spring 2022
	Organize codebase	12 tasks done	mickahell	code, QAMP spring 2022
	(Unlabeled)	7 tasks done	mickahell	code, QAMP spring 2022



# Roadmap



- Actual goal :
  - Quantum kernel program
  - Matrix completion program
  - Unittests
  - Full usable workflow
  - Telemetry of workflow usage
- Future goal (or if we have time) :
  - Add the project to Qiskit Ecosystem
  - Writing a Qiskit blog post

```
from qiskit import QuantumCircuit, execute
from qiskit import Aer, IBMQ
from qiskit.providers.aer.noise import NoiseModel

# Choose a real device to simulate from IBMQ provider
provider = IBMQ.load_account()
backend = provider.get_backend('ibmq_igo')
coupling_map = backend.configuration().coupling_map

# Generate an Aer noise model for device
noise_model = NoiseModel.from_backend(backend)
basis_gates = noise_model.basis_gates

# Generate 3-qubit G2C state
num_qubits = 3
circuit = QuantumCircuit(3, 3)
circuit.h(0)
circuit.cx(0, 1)
circuit.cx(1, 2)
circuit.measure([0, 1, 2], [0, 1, 2])

# Perform noisy simulation
backend = Aer.get_backend('aer_simulator')
job = execute(circuit, backend,
              coupling_map=coupling_map,
              noise_model=noise_model,
              basis_gates=basis_gates)
result = job.result()

print(result.get_counts(0))
```

# Thanks for your attention !

Cheryl, Michaël & Travis

