Quantum Computing: Programs and Systems

Wednesday, September 1, 2021

Rutgers University

Yipeng Huang

How does a computer work?

What are the parts of a computer?

What are our fundamental assumptions about how computers work / what computers are made of?

Some show and tell

- Non-digital, non-discrete time computation

- Analog computers

- Biological computers

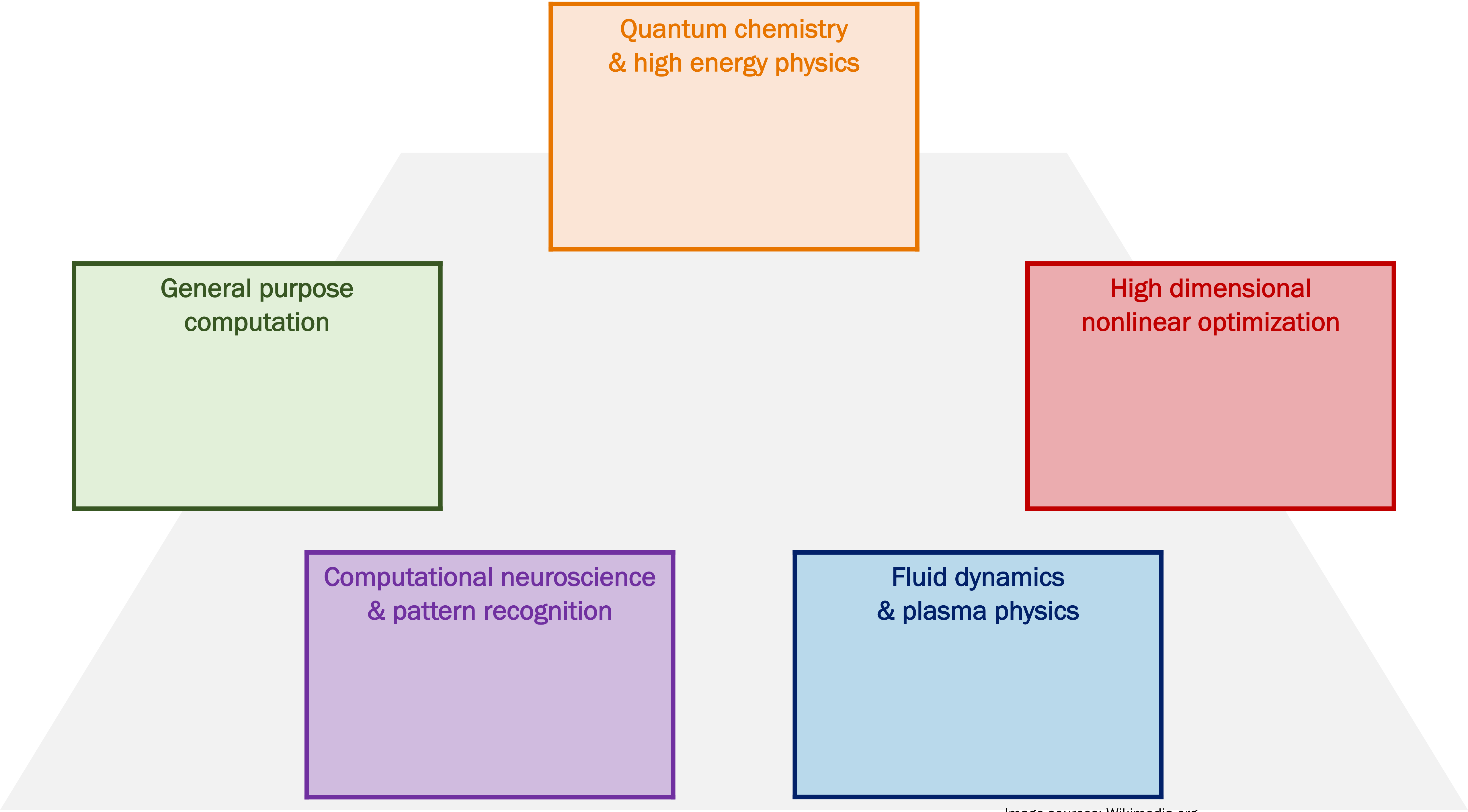


Image sources: Wikimedia.org

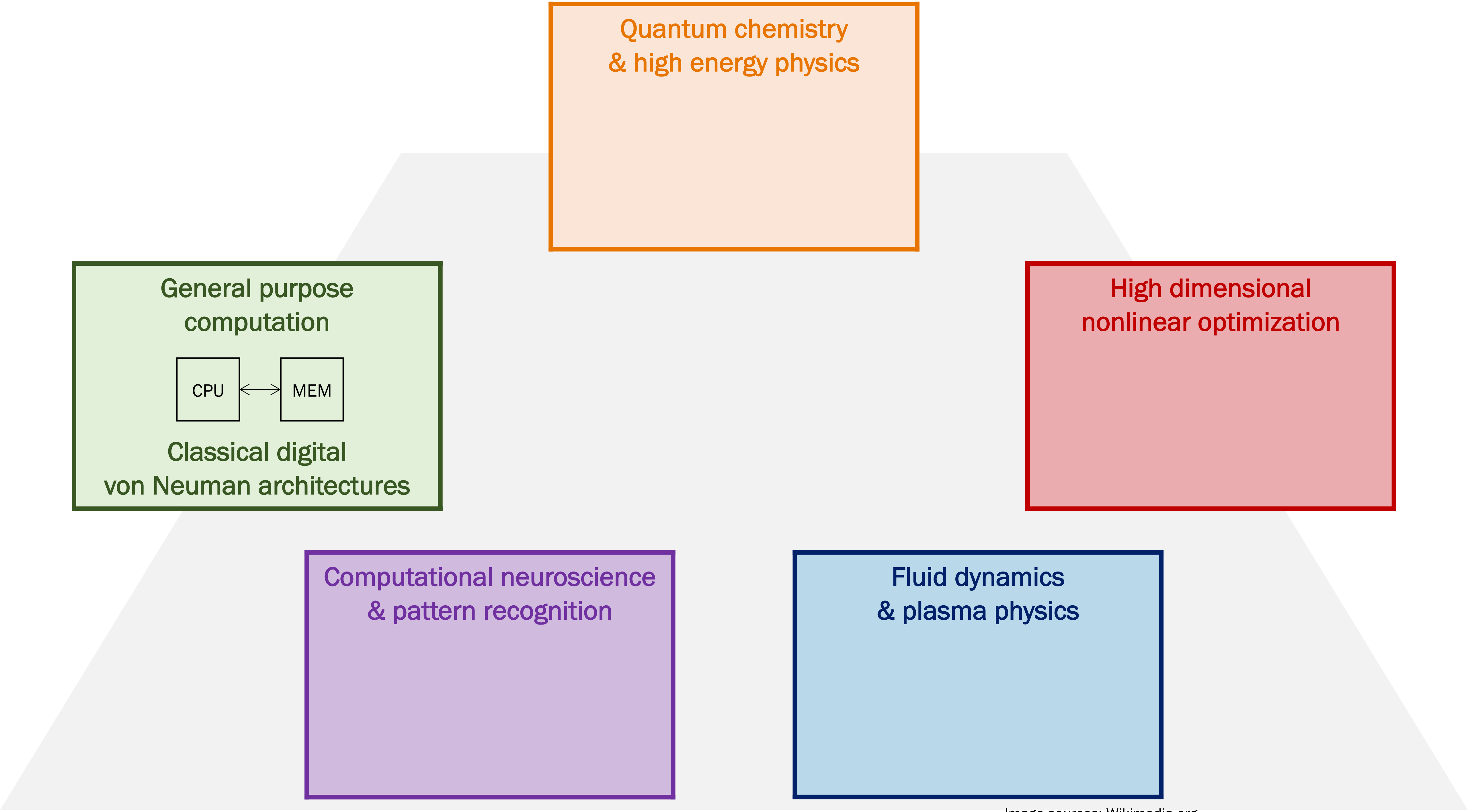
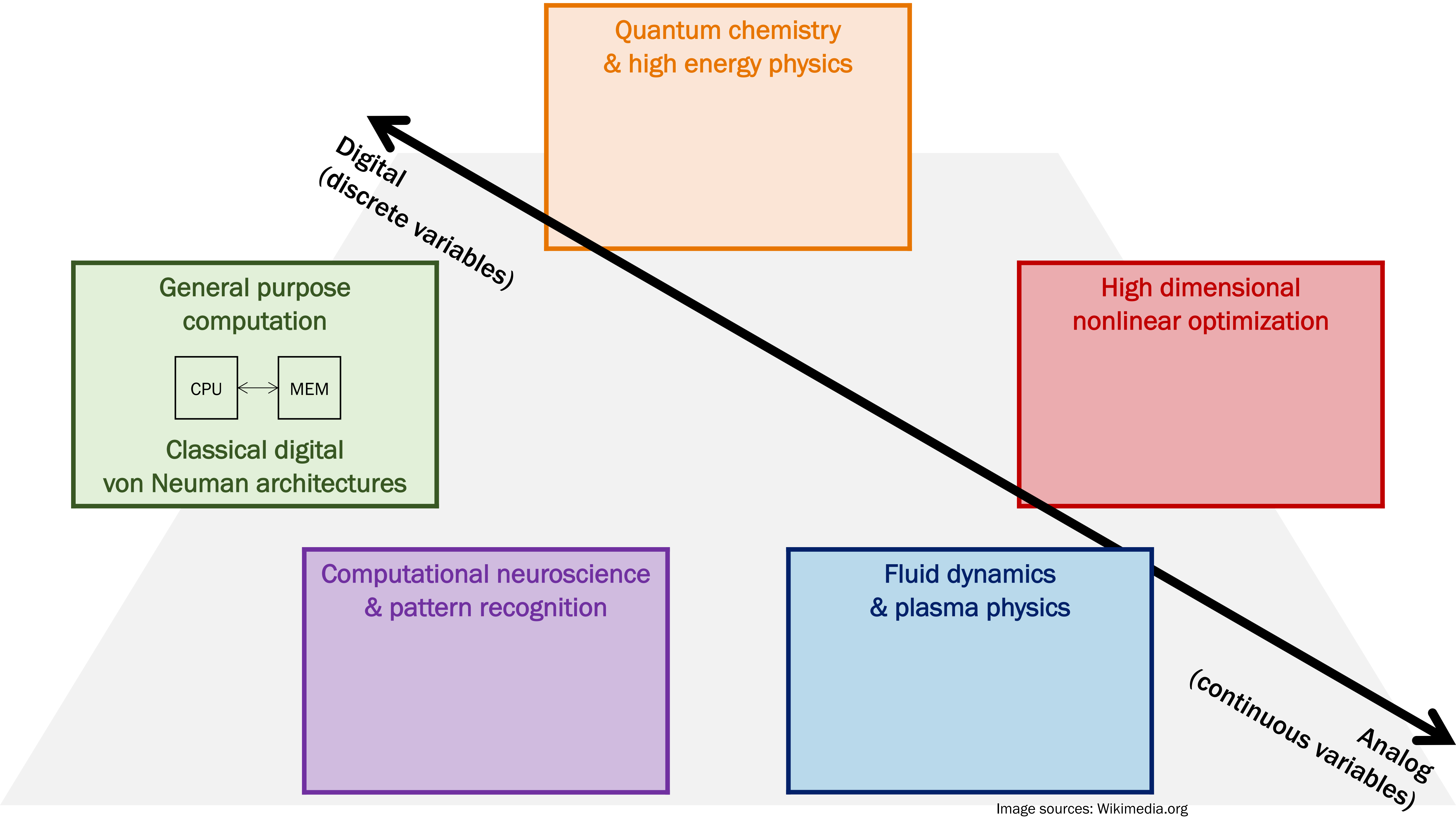
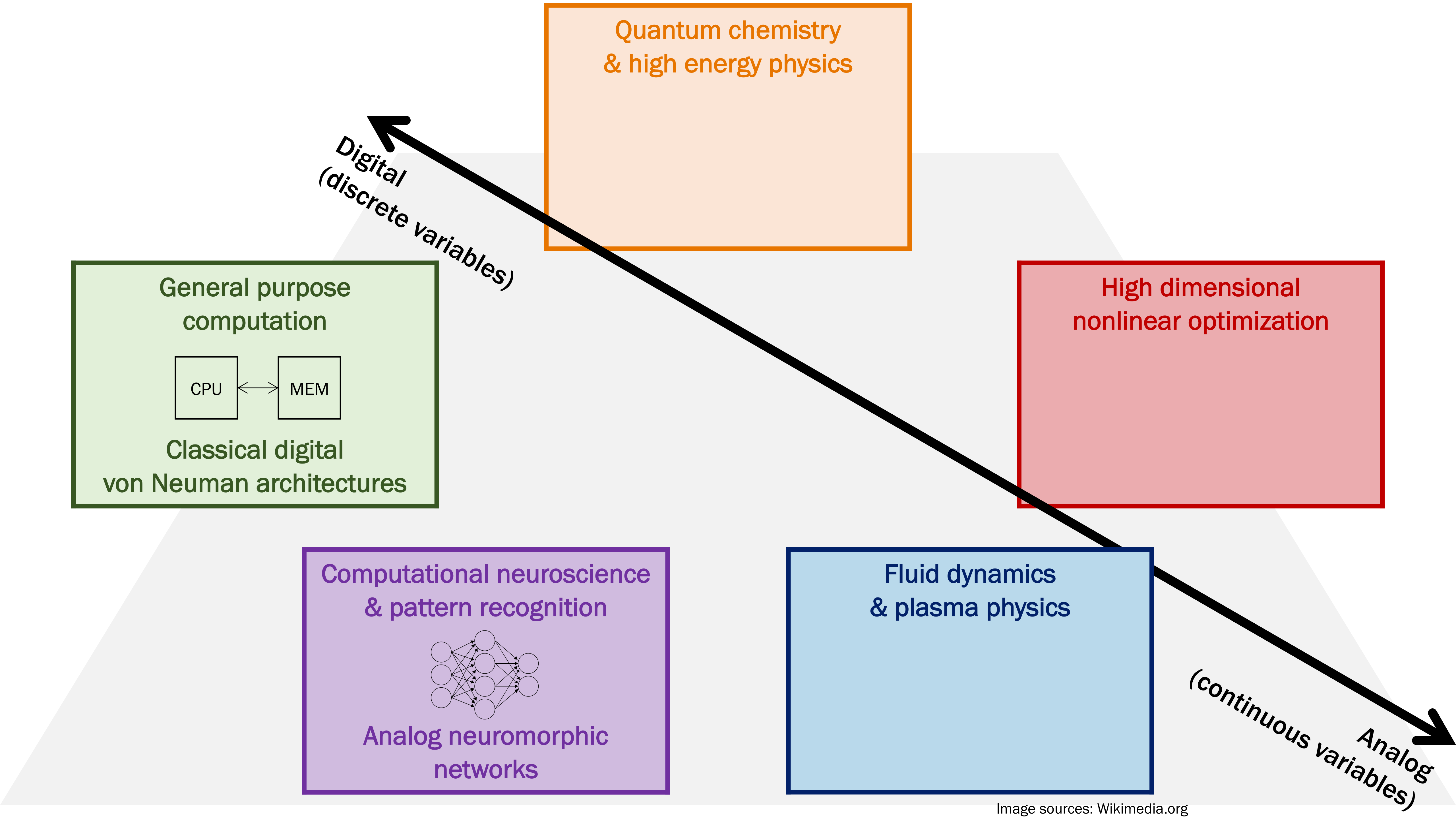


Image sources: Wikimedia.org


Quantum chemistry
& high energy physics
Digital
(discrete variables)
General purpose
computation
CPU
MEM
Classical digital
von Neuman architectures
High dimensional
nonlinear optimization
Computational neuroscience
& pattern recognition
Fluid dynamics
& plasma physics
Analog
(continuous variables)
Image sources: Wikimedia.org



Quantum chemistry
& high energy physics
Digital
(discrete variables)
General purpose
computation
CPU
MEM
Classical digital
von Neuman architectures
High dimensional
nonlinear optimization
Computational neuroscience
& pattern recognition
Analog neuromorphic
networks
Fluid dynamics
& plasma physics
Analog
(continuous variables)
Image sources: Wikimedia.org

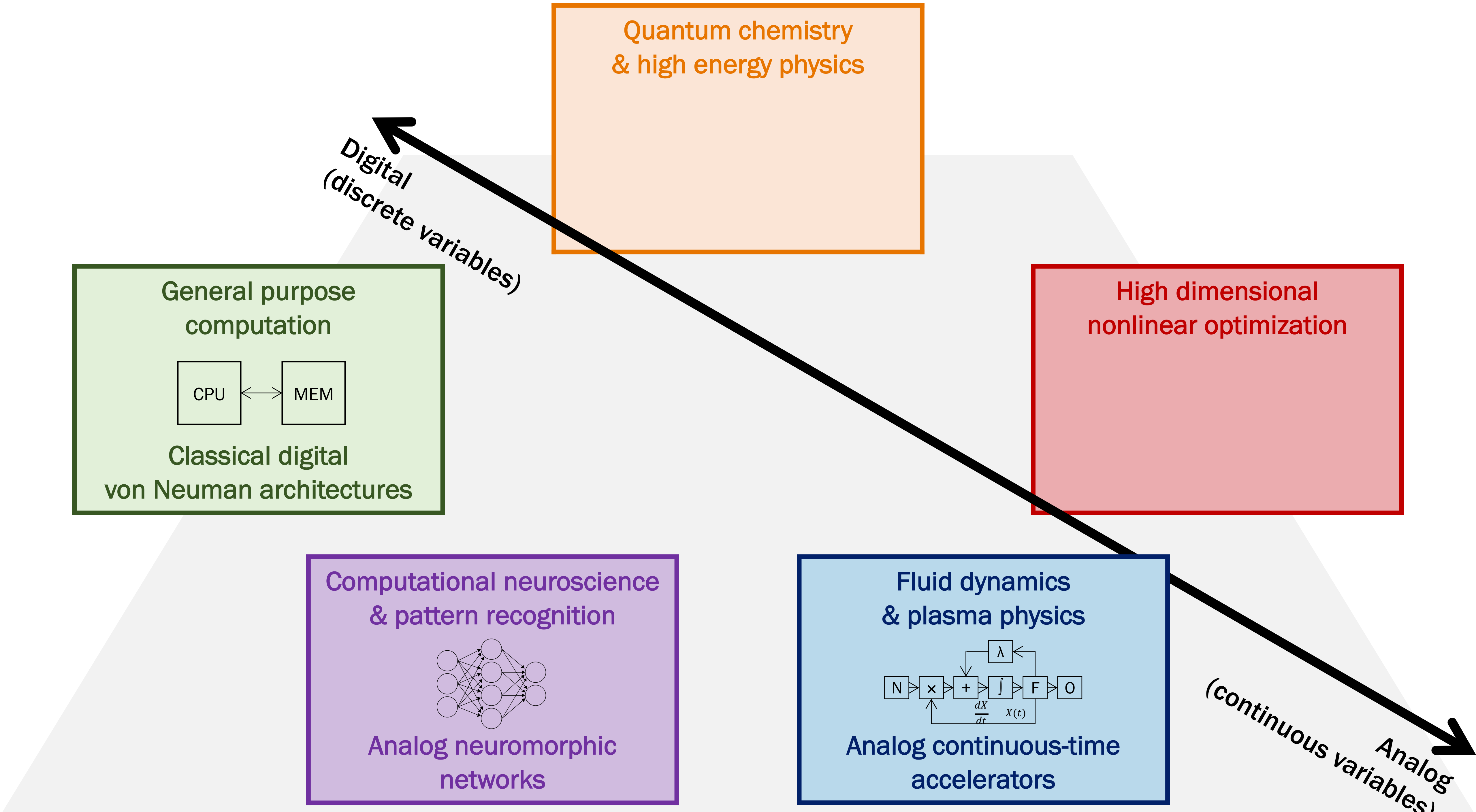
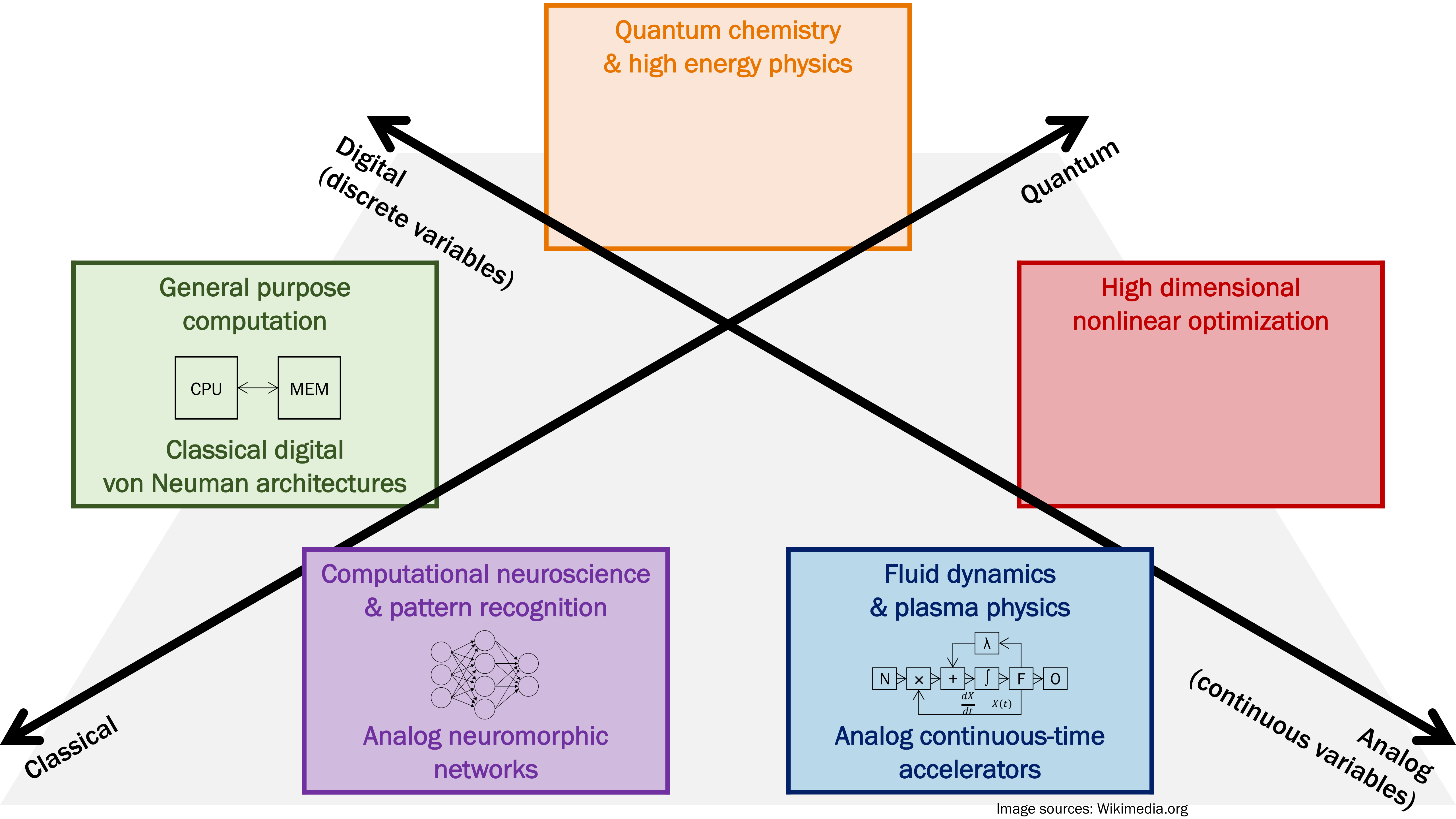
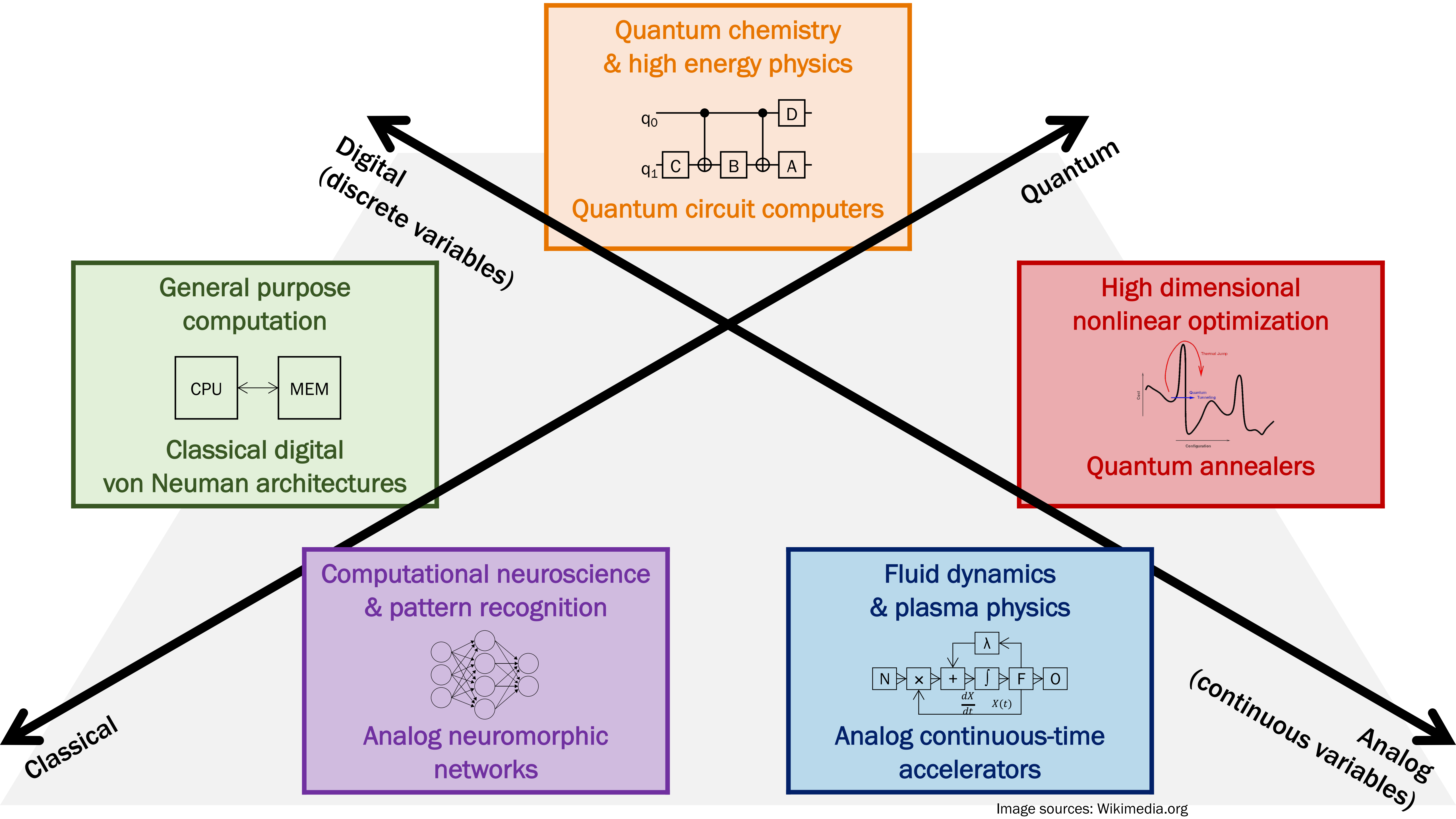
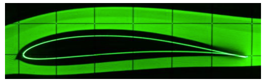
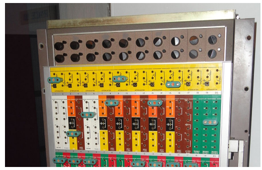


Image sources: Wikimedia.org



Quantum chemistry
& high energy physics
Digital
(discrete variables)
Quantum
General purpose
computation
CPU
MEM
Classical digital
von Neuman architectures
High dimensional
nonlinear optimization
Computational neuroscience
& pattern recognition
Analog neuromorphic
networks
Fluid dynamics
& plasma physics
λ
N
×
+
∫
F
O
X(t)
Analog continuous-time
accelerators
Classical
(continuous variables)
Analog
Image sources: Wikimedia.org


Quantum chemistry
& high energy physics
q0
q1
C
B
D
A
Quantum circuit computers
Digital
(discrete variables)
Quantum
General purpose
computation
CPU
MEM
Classical digital
von Neuman architectures
High dimensional
nonlinear optimization
Quantum annealers
Computational neuroscience
& pattern recognition
Analog neuromorphic
networks
Fluid dynamics
& plasma physics
λ
N
×
+
∫
F
O
Analog continuous-time
accelerators
Classical
(continuous variables)
Analog
Image sources: Wikimedia.org

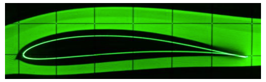
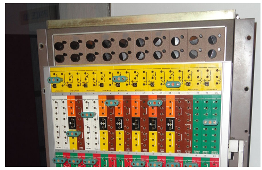
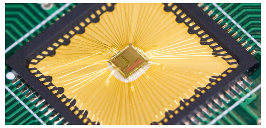
	1940s	1950s	1960s	1970s	1980s	1990s	2000s	2010s
Analog continuous-time computing	Analog computers for rocket and artillery controllers.	Analog computers for field problems. 	1st transistorized analog computer. 	Analog-digital hybrid computers.	…			
Digital discrete-time computing	Turing's *Bomba*. Stored program computer.	1st transistorized digital computer. Microprogramming.	Moore's law projection for transistor scaling. Instruction set architecture.	Dennard's scaling for transistor power density. Reduced instruction set computers.	VLSI democratized.	FPGAs introduced. GPUs introduced.	End of Dennard's scaling. CPUs go multicore. Nvidia introduces CUDA.	Cloud FPGAs: Microsoft Catapult. Amazon F1. ASICs: Google TPUs. DE Shaw Research Anton.

Heterogenous architectures (1990s–2010s)

Transistor scaling and architectural abstractions drive digital revolution, make analog alternatives irrelevant

Scaling challenges drive heterogenous architectures

Image sources: AnalogMuseum.org, ComputerHistory.org

	1940s	1950s	1960s	1970s	1980s	1990s	2000s	2010s
Analog continuous-time computing	Analog computers for rocket and artillery controllers.	Analog computers for field problems. 	1st transistorized analog computer. 	Analog-digital hybrid computers.	…	Analog neural networks proposed.	VLSI analog computers proposed.	Columbia University prototype analog accelerators. 
Digital discrete-time computing	Turing's *Bomba*. Stored program computer.	1st transistorized digital computer. Microprogramming.	Moore's law projection for transistor scaling. Instruction set architecture.	Dennard's scaling for transistor power density. Reduced instruction set computers.			End of CPUs go multicore.	Amazon F Google TP
Quantum computing					Feynman. "Simulating Physics with Computers."	Shor's algorithm. Demo of ion trap quantum computation.	Demo of super-conductor quantum computation.	IBM quantum cloud. Google quantum supremacy.

Challenges in digital scaling motivate revisiting analog alternative

Fundamental limits in digital model of computing motivate exploring quantum possibilities

Image sources: AnalogMuseum.org, ComputerHistory.org

What this class is about

Graduate seminar on latest developments in quantum computer engineering

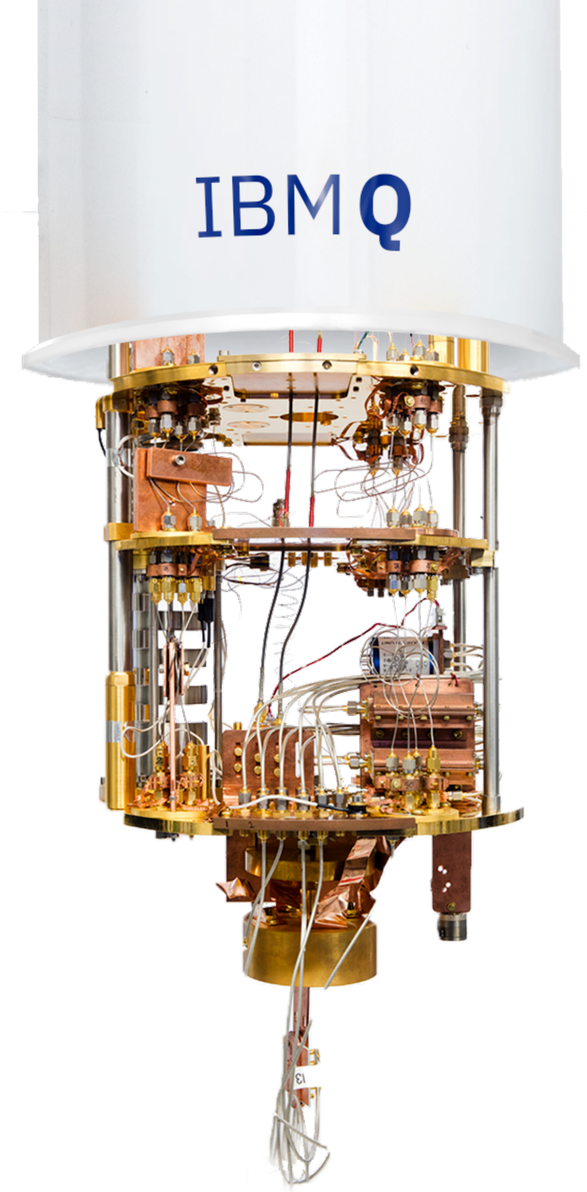
What is quantum computer engineering??

- realizing quantum algorithms
- on prototype quantum computers

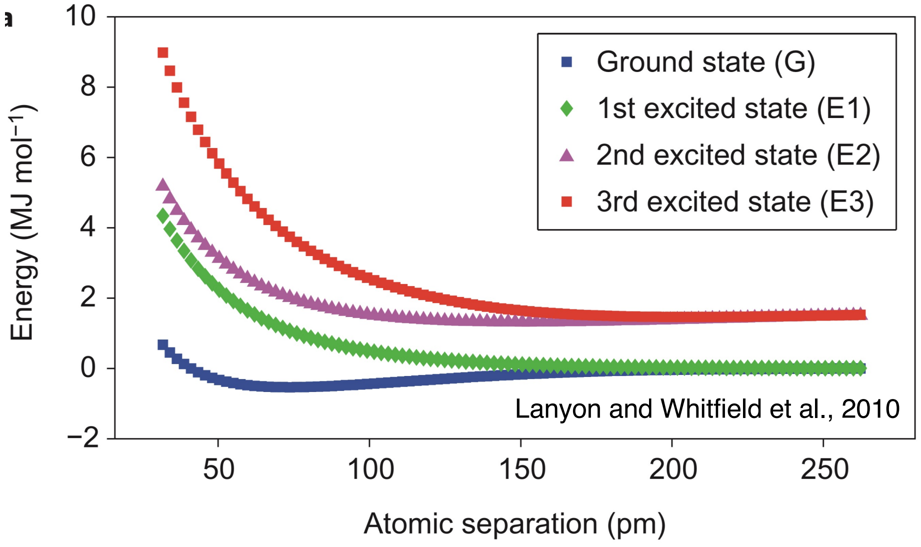
—a rapidly growing field!!

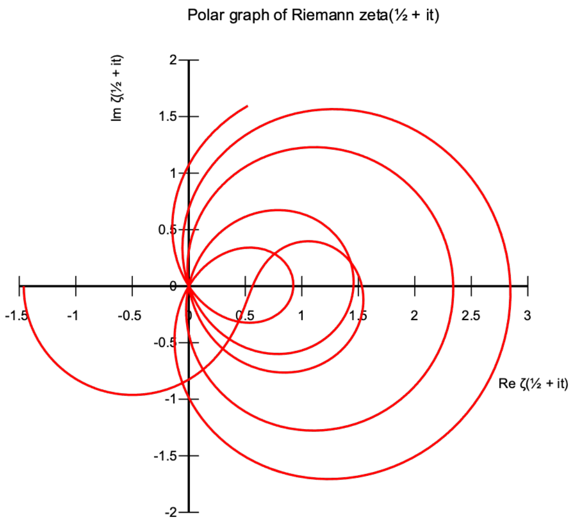
Goals of the course:

- explore open-source tools for using quantum computers
- read and discuss recent developments
- build foundation for you to pursue research or to be experts in industry



Motivation: Race to practical quantum computation





Quantum algorithms for chemical simulations

- Calculate properties of molecules directly from governing equations
- Use quantum mechanical computer to simulate quantum mechanics!

Shor's quantum algorithm for factoring integers

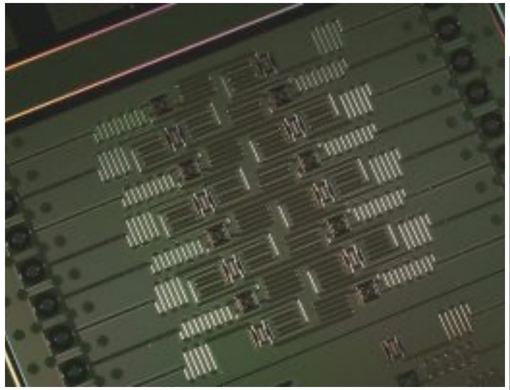
- Factor large integers to primes in polynomial time complexity
- Surpasses any known classical algorithm taking exponential time complexity

Hundreds of algorithms @ QuantumAlgorithmZoo.org

Motivation: Race to practical quantum computation

Superconducting qubits

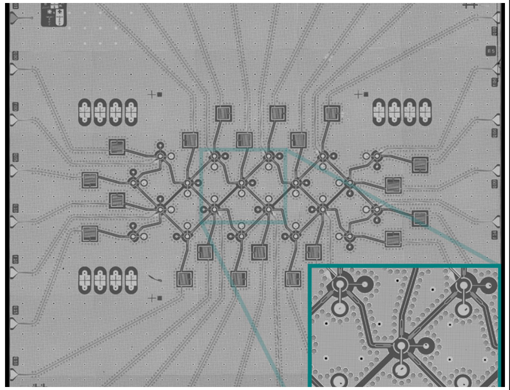
Trapped ion qubits



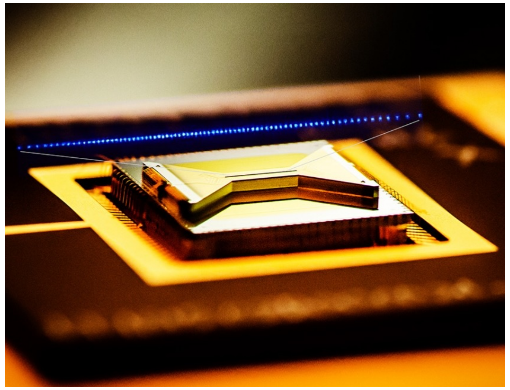
IBM

Google

Intel



Rigetti



University of Maryland / IonQ

Many research teams now competing towards more reliable and more numerous qubits.

Broad view of open challenges in quantum computer engineering

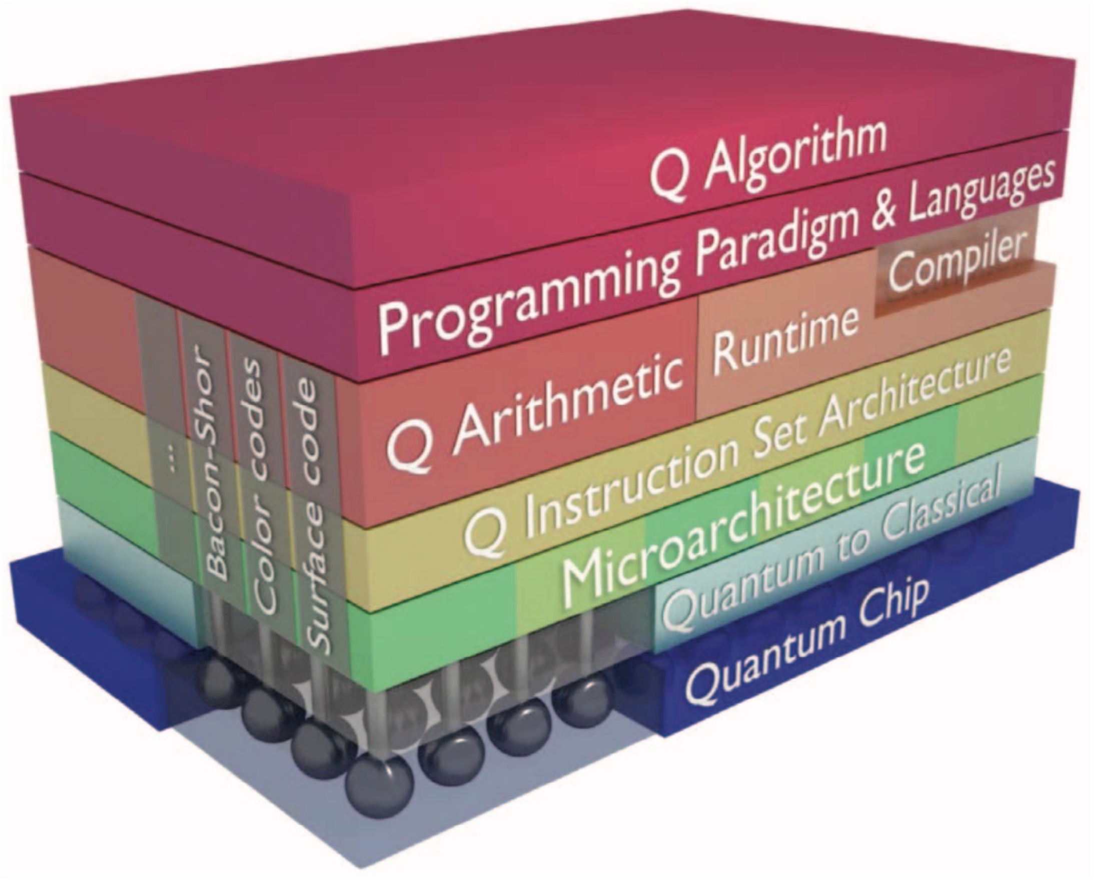


Figure 1. Overview of the quantum computer system stack.

A Microarchitecture for a Superconducting Quantum Processor. Fu et al.

- A complete view of full-stack quantum computing.
- In short, challenges are in finding and building abstractions.
- In each layer, why we don't or can't have good abstractions right now.
- Recent and rapidly developing field of research.

Outline

- ***Curiosity:** digital, discrete time abstractions; unconventional computing*
- **Community:** welcome to class; prerequisites; introductions
- **Learning:** preview of the syllabus; quantum computer systems stack
- **Expectations:** reading; programming; presentations

(Taking advice from https://www.chronicle.com/article/how-to-teach-a-good-first-day-of-class/)

Outline

- **Curiosity:** digital, discrete time abstractions; unconventional computing
- ***Community:** welcome to class; prerequisites; introductions*
- **Learning:** preview of the syllabus; quantum computer systems stack
- **Expectations:** reading; programming; presentations

(Taking advice from https://www.chronicle.com/article/how-to-teach-a-good-first-day-of-class/)

Prerequisites

Algorithms: time and space complexity of algorithms

Complex numbers

Linear algebra: vector, matrix notation and multiplication. Matrix properties.

Probability and statistics

Python programming: working with Git, extending open source projects, Jupyter notebooks

Access to iLab CS computing resources: https://resources.cs.rutgers.edu/

Useful, but not strictly required

Quantum information science course

- Bra-ket, gates, circuits, measurement, superposition, entanglement
- 2021 Fall: ECE 493. Soljanin. Quantum Computing Algorithms.
- 2022 Spring: Physics 421. Schnetzer. An Introduction to Quantum Comp

Quantum mechanics

- Problems and methods for quantum chemistry

Personal introductions

Personal introductions

My name is:

How far along I am in my studies:

Why I am interested in quantum computing:

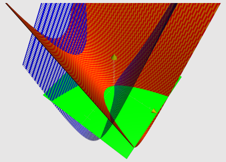
Something I am interested in computer science / engineering broadly:

Personal introductions

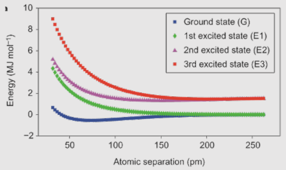
Yipeng

- Assistant professor, Rutgers, 2020 -
- Postdoc, Princeton, 2018 - 2020
- PhD, Columbia, - 2018

Nonlinear scientific computation



Quantum simulation & optimization



New and extreme workload challenges

Multicore CPUs, GPUs, FPGAs, ASICs, analog, quantum, etc.

Limitations in transistor scaling

Dennard's scaling already ended

Moore's law increasingly costly to sustain

Open challenges in emerging architectures:

Problem abstractions

- How do you accurately solve big problems?

Programming abstractions

- Can you borrow ideas from conventional computing?

Architecture abstractions

- How to interface with the unconventional hardware?

Image sources: Lanyon and Whitfield et al., 2010

My work in problem and programming abstractions for emerging architectures

Continuous-time analog scientific computation

Accelerator chip prototype	Support for solving differential equations	Support for solving linear algebra	Support for solving nonlinear equations	Fluid dynamics application feasibility study
Successful hand-off to MIT, Ulm University, and two companies for further research.	JSSC 2016 (co-authored).	ISCA 2016. One of twelve Micro Top Picks best architecture papers of 2016.	MICRO 2017. Micro Top Picks honorable mention.	PI for DARPA STTR phase 1 grant. Thesis nominated for ACM dissertation award.

Quantum algorithm debugging & simulation

	Assertions for quantum program patterns and bugs	Graphical model inference for quantum program simulation and analysis	Analog computing support for quantum control & measurement	
	ISCA 2019. mentees placed at MICRO SRC. IBM Qiskit open-source contribution.			

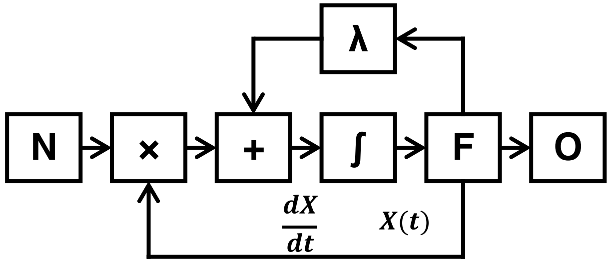
Requirements for supporting workloads

- **How to do problems?**
- **How to get high accuracy solutions?**
- **How to handle large problem sizes?**

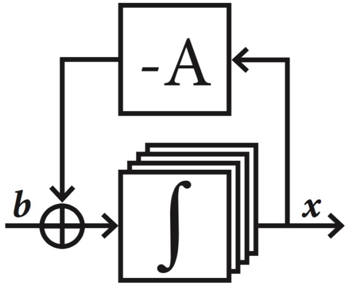
$$\frac{\partial \vec{u}}{\partial t} + (\vec{u} \cdot \nabla)\vec{u} - \frac{1}{\text{Re}} \nabla^2 \vec{u} = RHS$$

Numerical primitives as architectural abstractions

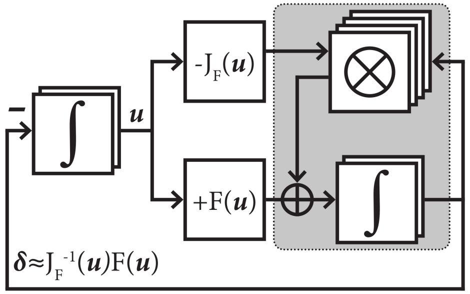
Analog-digital support for **differential equations**



Analog-digital support for **linear algebra**

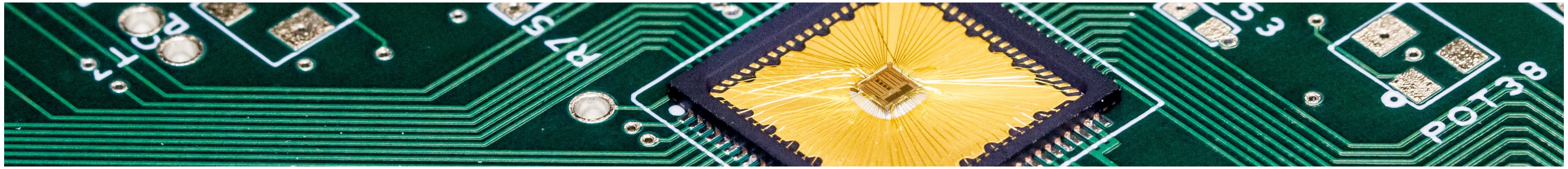


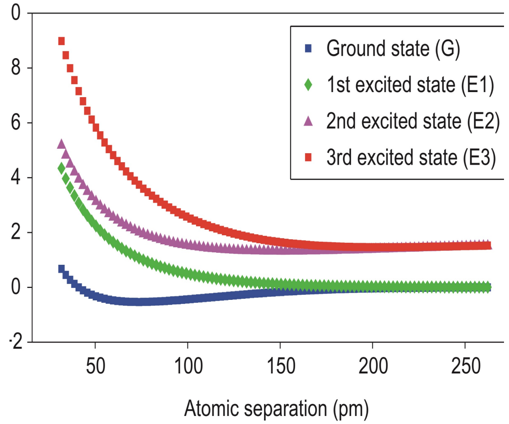
Analog-digital support for **nonlinear equations**



Unconventional architecture hardware prototyping

Prototype continuous-time analog accelerator





Awe-inspiring quantum algorithms

Chemistry simulations from governing equations	Shor's algorithm for factoring integers	Hundreds more near-term and far-future algorithms
Quantum computers as quantum mechanics simulator	Surpasses any known classical algorithm	QuantumAlgorithmZoo.org

My work in bridging quantum software-hardware gap

Assertions for quantum program patterns and bugs	Graphical model inference for quantum program simulation and analysis	Analog computing support for quantum control & measurement
ISCA 2019. IBM Qiskit open-source contribution.	ASPLOS 2021. Google Cirq simulation backend publicly available.	

Now-viable quantum prototypes

Superconducting qubits	Trapped ion qubits	Dozens of candidate qubit technologies
IBM, Google, Rigetti, …	IonQ, UMD, …	May yet surpass current leaders in capacity and reliability

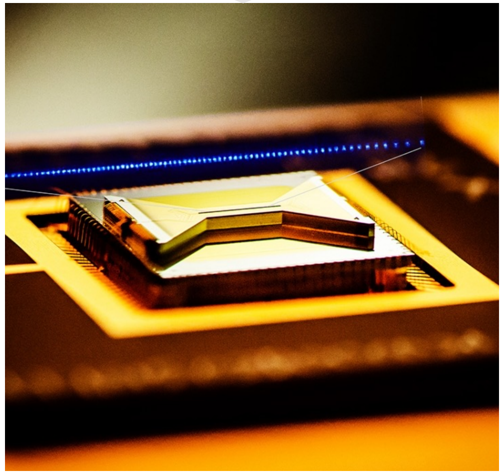


Image sources: Lanyon and Whitfield et al., 2010, IonQ

Personal introductions

Why I am excited about quantum computing:

- Broad field, rapidly changing, many new topics

Something I am interested in computer science / engineering broadly:

- New paradigms for computing

Welcome all to class

We welcome in this class diverse backgrounds and viewpoints spanning various dimensions:

- race, national origin, gender, sexuality, disability status, class, religious beliefs

We will treat each other with respect and strive to create a safe environment to exchange questions and ideas.

Outline

- **Curiosity:** digital, discrete time abstractions; unconventional computing
- **Community:** welcome to class; prerequisites; introductions
- ***Learning:** preview of the syllabus; quantum computer systems stack*
- **Expectations:** reading; programming; presentations

(Taking advice from https://www.chronicle.com/article/how-to-teach-a-good-first-day-of-class/)

Preview of the syllabus

- A systems view of quantum computer engineering
- Near-term intermediate-scale quantum algorithms
- Programming frameworks
- Emerging languages and representations
- Claims and counter claims for quantum advantage
- Extracting success
- Prototypes

Preview of the syllabus

- ***A systems view of quantum computer engineering***
- Near-term intermediate-scale quantum algorithms
- Programming frameworks
- Emerging languages and representations
- Claims and counter claims for quantum advantage
- Extracting success
- Prototypes

Semantic gap

- Need languages, abstractions…

Tools gap

- Need optimizing compilers, simulators, debuggers…

Infrastructure gap

- Need more abundant, more reliable qubits…

Educational gap

- Need researchers, students…

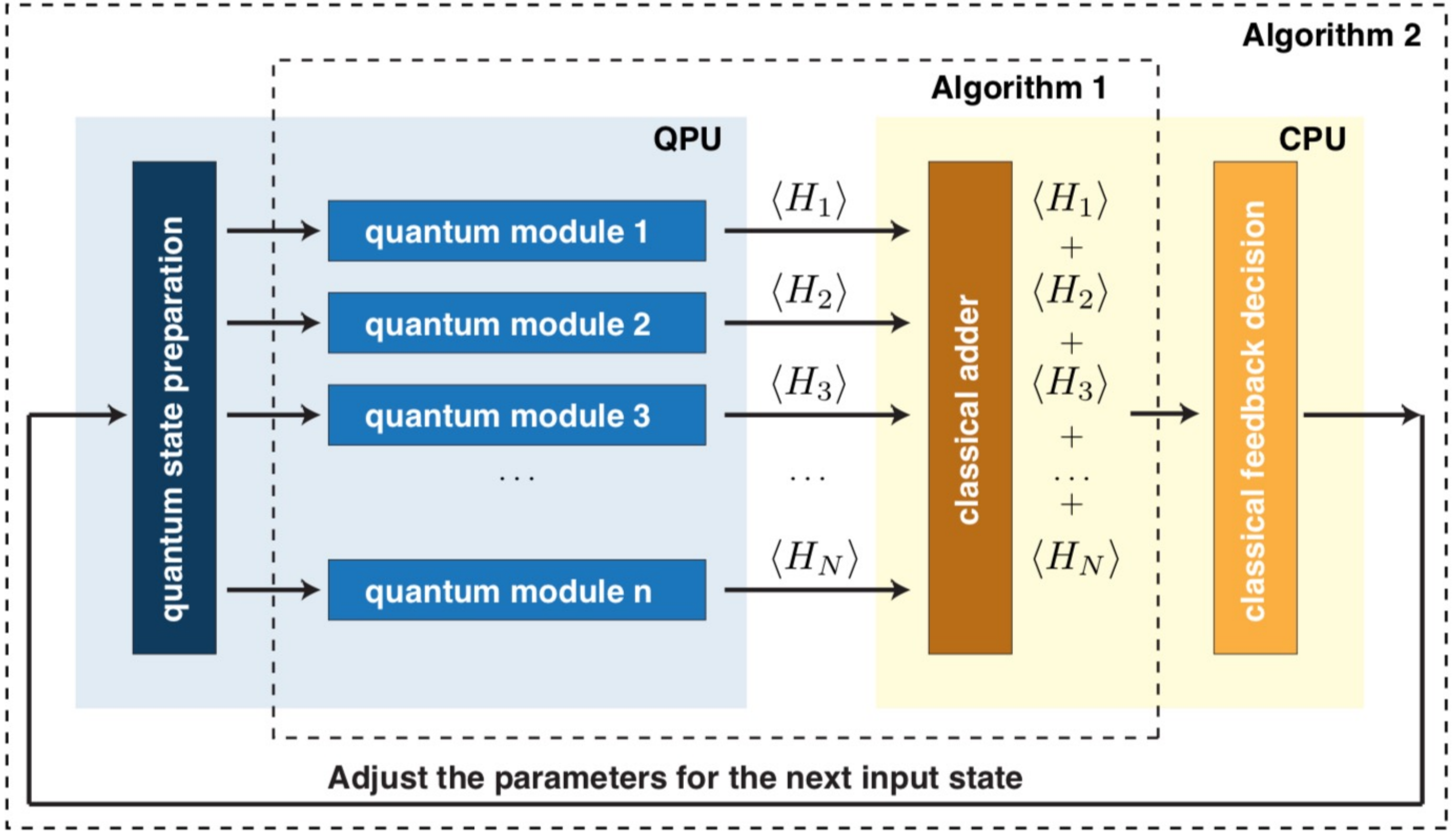
Preview of the syllabus

- A systems view of quantum computer engineering
- ***<u>Near-term intermediate-scale quantum algorithms</u>***
- Programming frameworks
- Emerging languages and representations
- Claims and counter claims for quantum advantage
- Extracting success
- Prototypes

What are variational algorithms and why are they important?

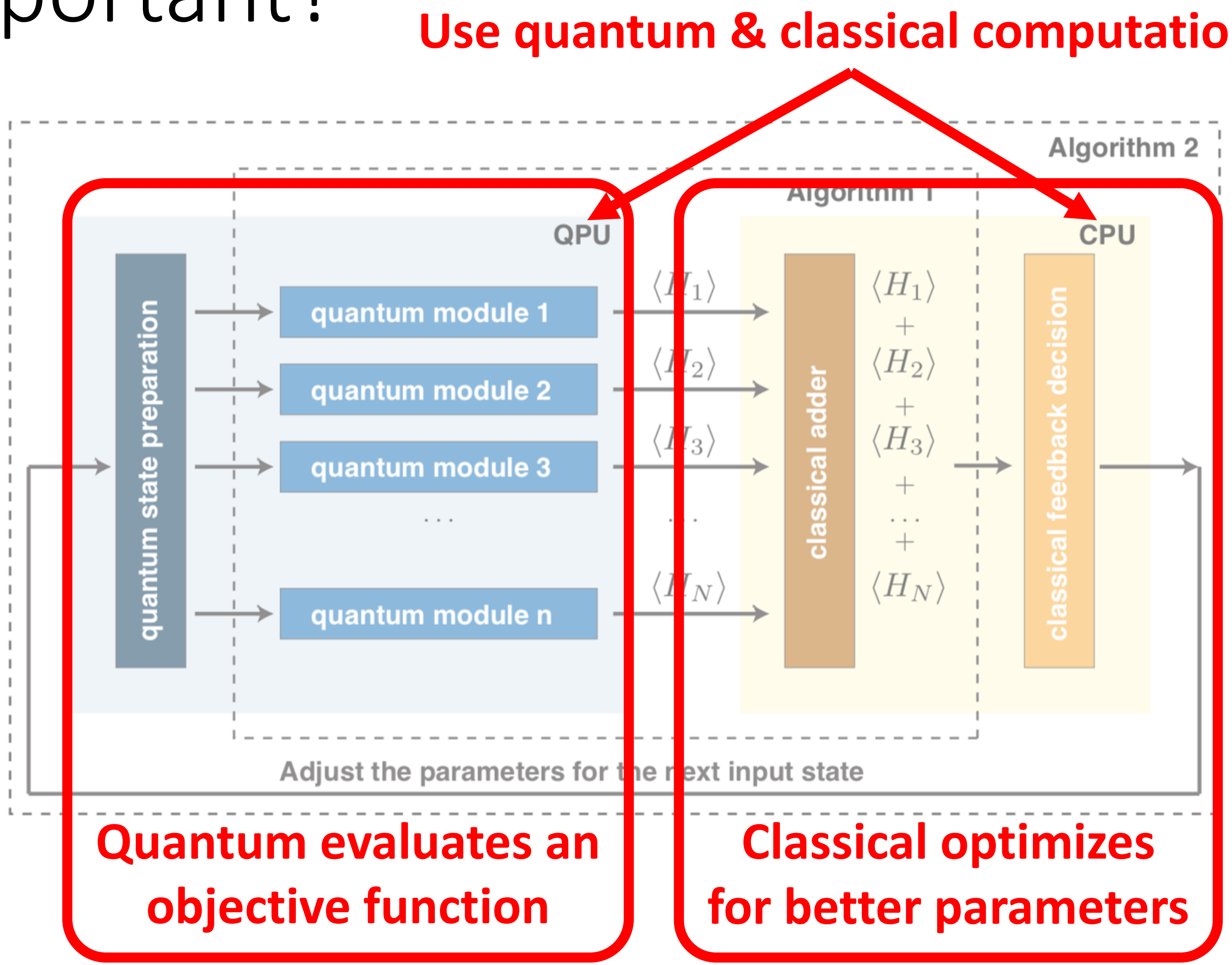
- It’s like using a classical computer to train a quantum neural network.

What are variational algorithms and why are they important?



Source: Peruzzo et al., 2013

What are variational algorithms and why are they important?



Source: Peruzzo et al., 2013

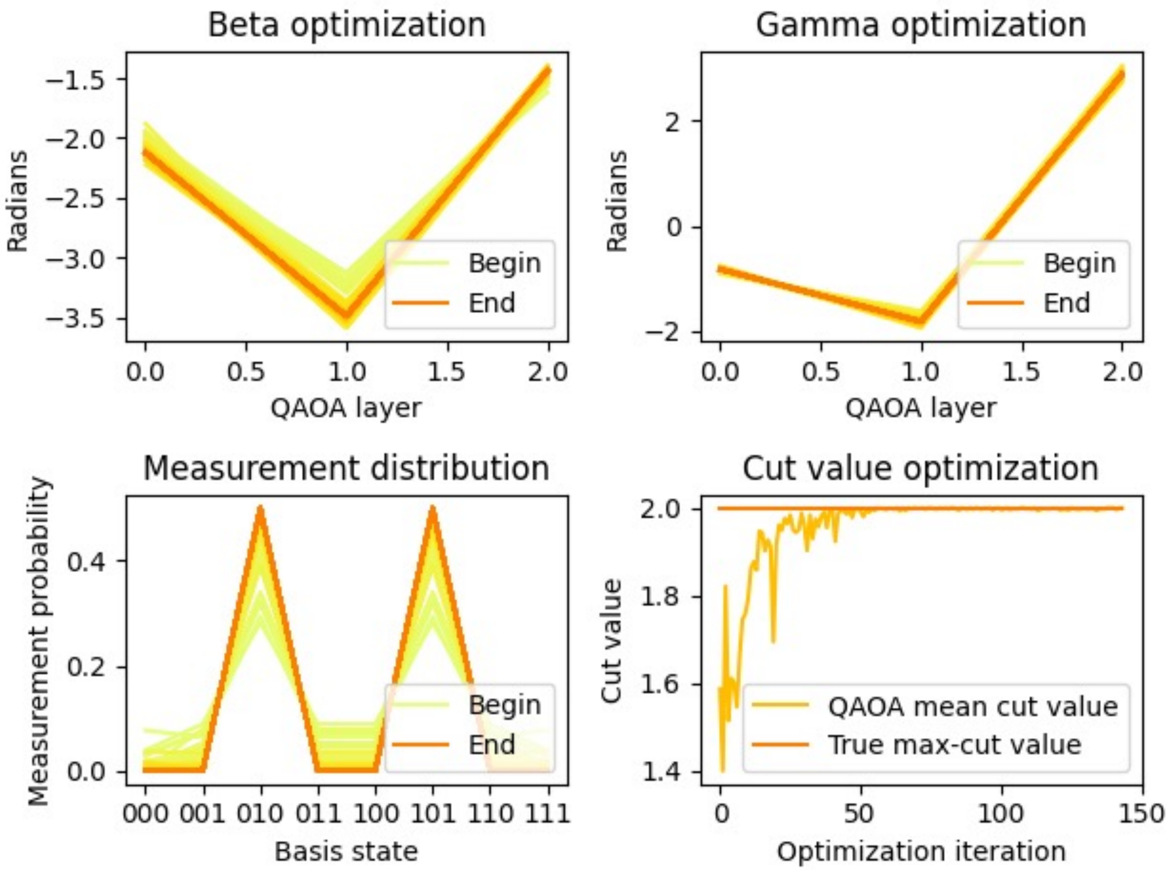
What are variational algorithms and why are they important?

- It’s like using a classical computer to train a quantum neural network.
- Quantum computer can be unreliable, needs 50-100 qubits, realizable in the near future.
- Most likely candidates for first demonstrations of useful quantum.
- Major examples include quantum approximate optimization algorithm (QAOA), variational quantum eigensolver (VQE).

Preview of the syllabus

- A systems view of quantum computer engineering
- Near-term intermediate-scale quantum algorithms
- ***Programming frameworks***
- ***Emerging languages and representations***
- Claims and counter claims for quantum advantage
- Extracting success
- Prototypes

Programming assignments



Preview of the syllabus

- A systems view of quantum computer engineering
- Near-term intermediate-scale quantum algorithms
- Programming frameworks
- Emerging languages and representations
- ***Claims and counter claims for quantum advantage***
- ***Prototypes***

Debates & presentations

Contentious topics in quantum computer engineering

Quantum programming: Verification vs. Debugging

Quantum/classical boundary: Prototypes vs. Simulation (Google vs. IBM)

Quantum device candidates: Superconductors vs. Ion traps

In this class, you will present the competing viewpoints in debate format

Preview of the syllabus

- A systems view of quantum computer engineering
- Near-term intermediate-scale quantum algorithms
- Programming frameworks
- Emerging languages and representations
- Claims and counter claims for quantum advantage
- Extracting success
- Prototypes

Research here at Rutgers

- Prof. Mario Szegedy, quantum algorithms, complexity theory
- Prof. Yipeng Huang, quantum program simulation and analysis
- Prof. Zheng Zhang, quantum circuit compilation
- Prof. Emina Soljanin, quantum communications
- Prof. Steve Schnetzer, high energy physics and quantum computing

Outline

- **Curiosity:** digital, discrete time abstractions; unconventional computing
- **Community:** welcome to class; prerequisites; introductions
- **Learning:** preview of the syllabus; quantum computer systems stack
- ***Expectations:** reading; programming; presentations*

(Taking advice from https://www.chronicle.com/article/how-to-teach-a-good-first-day-of-class/)

Logistics

https://yipenghuang.com/teaching/2021-fall/

https://rutgers.instructure.com/courses/140409

Recommend that you attend live, in-person

Join via Zoom if you need to be outside of the classroom for any reason

- Videos are recorded and posted

Office hours are Wednesdays 3pm, via Zoom

One of the few uppermost division classes you might take

- Very different expectations from any other class

Components

- Programming assignments (50% of course grade)
- Reading, debates & presentations (50% of course grade)

Programming assignments

- Quantum approximate optimization algorithm for solving MAX-CUT
- Quantum variational eigensolver for calculating chemical properties
- Google Cirq open source quantum framework
- IBM Qiskit open source framework and IBM Q public prototypes

Reading, debates & presentations

How to read and summarize papers efficiently:
- Who cares, so what?
- Why prior work is insufficient.
- Key insight.
- Methodology.
- Findings.

Deeper evaluation of the reading:
- What are limitations of the science in the paper?
- Is the paper effectively communicated?
- What background knowledge is preventing me from fully understanding this paper?

What you can expect from this year's offering of this class

- CS 672 (CS seminar) → CS 583 (systems topic) → 400-level new course

- More: gentle review of quantum computing fundamentals
- More: programming examples and exercises
- More: up-to-date selection of articles

- Less: lecturing