BRIEF UPDATE ON CQN SIMULATION STACK

Stefan Krastanov
UMass Amherst
May 2023

THE QUANTUM TECHNOLOGY STACK

THE QUANTUM TECHNOLOGY STACK

MATERIALS

THE QUANTUM TECHNOLOGY STACK

MATERIALS

ANALOG CONTROL

THE QUANTUM TECHNOLOGY STACK

MATERIALS

ANALOG CONTROL

NOISY DIGITAL CIRCUITS

THE QUANTUM TECHNOLOGY STACK

MATERIALS

ANALOG CONTROL

NOISY DIGITAL CIRCUITS

ERROR CORRECTION

THE QUANTUM TECHNOLOGY STACK

MATERIALS

ANALOG CONTROL

NOISY DIGITAL CIRCUITS

ERROR CORRECTION

QUANTUM ALGORITHMS

THE QUANTUM TECHNOLOGY STACK

ANALOG CONTROL

NOISY DIGITAL CIRCUITS

ERROR CORRECTION

THE QUANTUM TECHNOLOGY STACK

ANALOG CONTROL

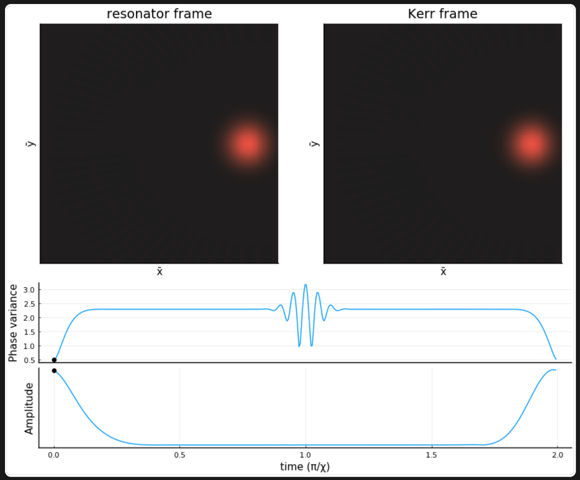
NOISY DIGITAL CIRCUITS

ERROR CORRECTION

FULL-STACK DESIGN AND OPTIMIZATION TOOLKIT

TYPES OF DYNAMICS

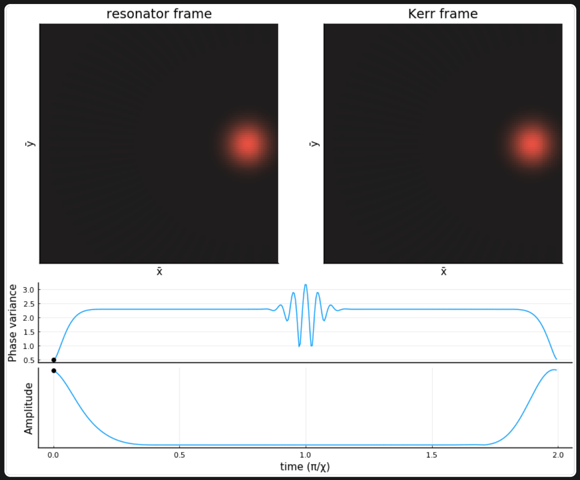
TYPES OF DYNAMICS



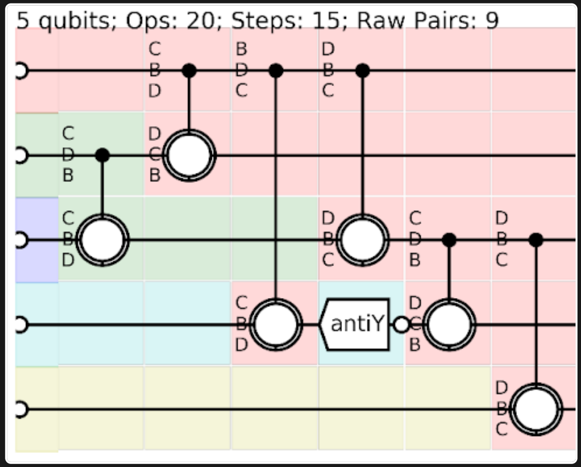
Continuous:

Hamiltonians, Master Equations

TYPES OF DYNAMICS

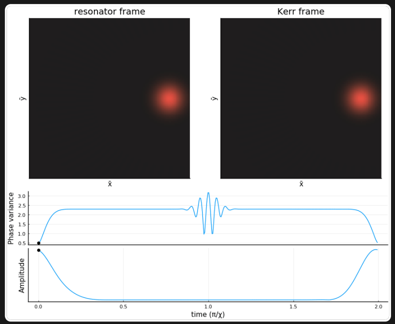


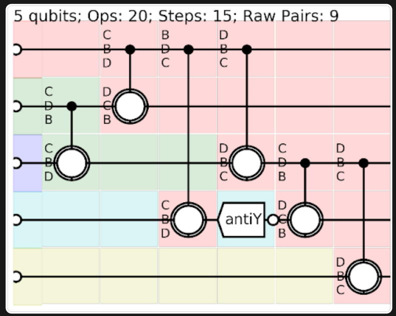
Continuous:
Hamiltonians, Master Equations

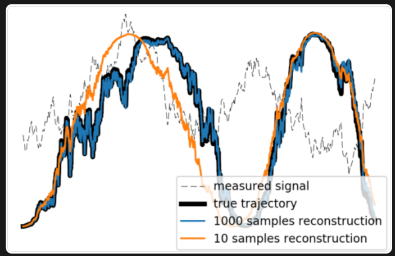


Discrete:
Gates, Circuits

TYPES OF DYNAMICS





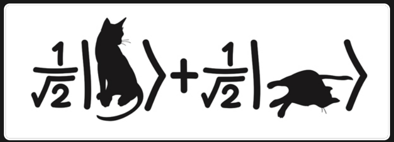


Continuous:
Hamiltonians,
Master Equations

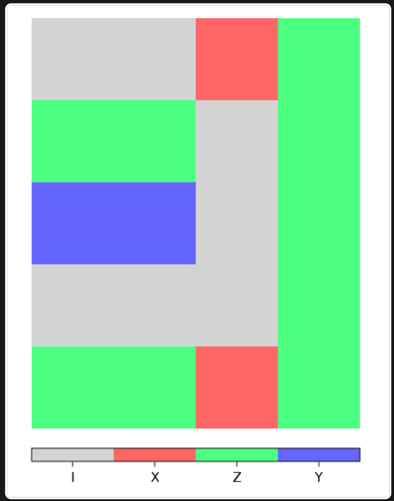
Discrete:
Gates, Circuits

Stochastic:
Weak
Measurements,
Feedback

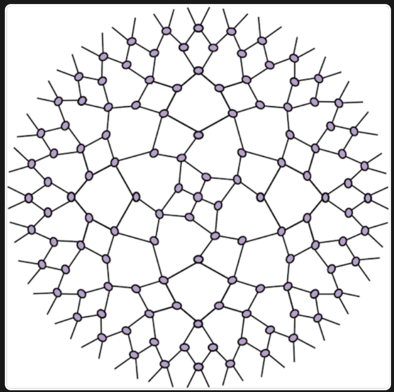
STATE REPRESENTATION



Kets and density matrices



Tableaux and graphs



Matrix product states and tensor network states

WHY SO MANY DIFFERENT REPRESENTATIONS?

WHY SO MANY DIFFERENT REPRESENTATIONS?

Classically we get to just do stacked Monte Carlo simulations...

WHY SO MANY DIFFERENT REPRESENTATIONS?

Classically we get to just do stacked Monte Carlo simulations...

... Quantum effects are interesting mostly when Monte Carlo fails!

WHY SO MANY DIFFERENT REPRESENTATIONS?

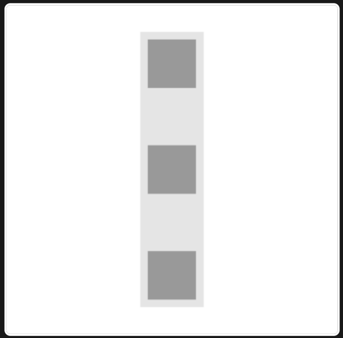
Classically we get to just do stacked Monte Carlo simulations...

... Quantum effects are interesting mostly when Monte Carlo fails!

... or because Monte Carlo fails!

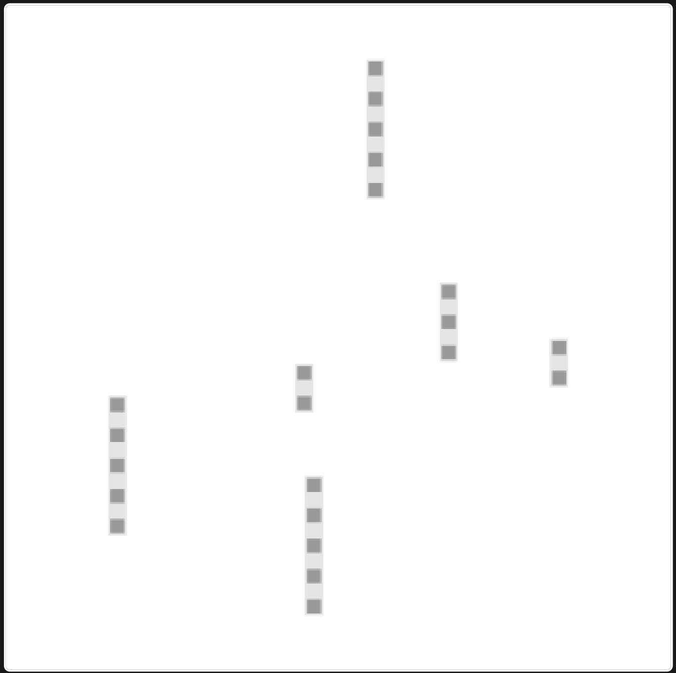
WHE NEED TO MARSHAL DIVERSE SIMULATORS TOGETHER AND CONVERT BETWEEN REPRESENTATIONS.

```
traits = [Qubit(), Qubit(), Qumode()]
reg = Register(traits)
```



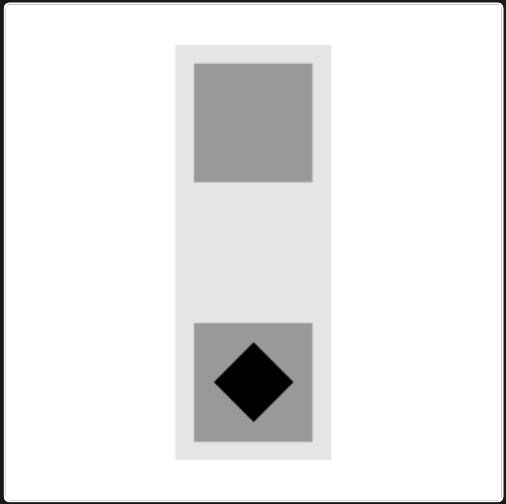
A register "stores" the states being simulated.

```
graph = grid([2,3])
registers = [...]
net = RegisterNet(graph, registers)
```



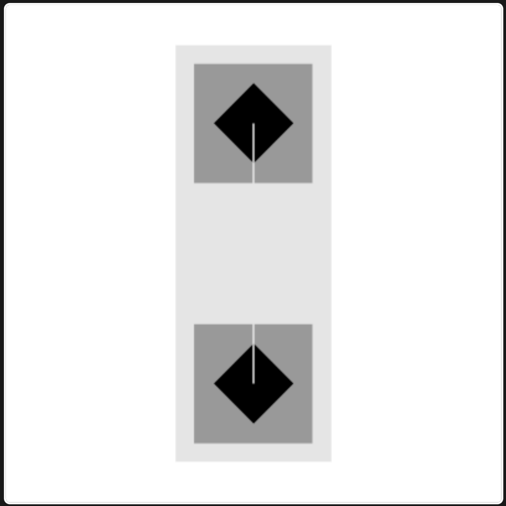
A "graph" of registers can represent a network.

```
initialize!(reg[1], X₁)
```



A register's slot can be initialized to an arbitrary state, e.g. $|x_1\rangle$ an eigenstate of $\hat{\sigma}_x$.

```
initialize!(reg[1], X₁)
initialize!(reg[2], Z₁)
apply!((reg[1], reg[2]), CNOT)
```



Arbitrary quantum gates or channels can be applied.

```
project_traceout!(reg[1], σˣ) # Projective measurement

observable((reg[1],reg[2]), σᶻ⊗σˣ) # Calculate an expectation
```

Measurements and expectation values...

```
project_traceout!(reg[1], σˣ) # Projective measurement

observable((reg[1],reg[2]), σᶻ⊗σˣ) # Calculate an expectation
```

Measurements and expectation values...

FULL SYMBOLIC COMPUTER ALGEBRA SYSTEM

FULL SYMBOLIC COMPUTER ALGEBRA SYSTEM

```
julia> Z₁
|Z₁⟩
```

FULL SYMBOLIC COMPUTER ALGEBRA SYSTEM

```
julia> Z₁
|Z₁⟩
```

```
julia> ( Z₁⊗X₂+Y₁⊗Y₁ ) / √2
0.707 (|Y₁⟩|Y₁⟩+|Z₁⟩|X₂⟩)
```

SYMBOLIC TO NUMERIC CONVERSION

SYMBOLIC TO NUMERIC CONVERSION

```
julia> express( ( Z₁⊗X₂+Y₁⊗Y₁ ) / √2 )
Ket(dim=4)
  basis: [Spin(1/2) ⊗ Spin(1/2)]
   0.8535533905932736 + 0.0im
                  0.0 + 0.3535533905932737im
 -0.49999999999999994 + 0.3535533905932737im
  -0.3535533905932737 + 0.0im
```

SYMBOLIC TO NUMERIC CONVERSION

```
julia> express( ( Z₁⊗X₂+Y₁⊗Y₁ ) / √2 )
Ket(dim=4)
  basis: [Spin(1/2) ⊗ Spin(1/2)]
   0.8535533905932736 + 0.0im
                  0.0 + 0.3535533905932737im
 -0.4999999999999994 + 0.3535533905932737im
  -0.3535533905932737 + 0.0im
```

```
julia> express( Y₁⊗Y₂, CliffordRepr() )
Rank 2 stabilizer
+ Z_
+ _Z
════
+ Y_
- _Y
════
```

Simulations of the generation of 3×2 cluster states in Tin-vacancy color centers

The top-left plot shows the state of the network of registers. Each register has two slots, one for an electron spin where the entanglement gets established through a Barrett-Kok protocol, and one for a nuclear spin for long term storage. The colored-line overlay on top of the registers gives the fidelity of the various operators stabilizing the cluster state.

The plot at the bottom left gives the overall fidelity of the state, together with the fidelity of the best and worst components of the state, over time.

To the right the various locks and resource queues being tracked by the simulation are plotted in real time. For instance, whether the electron spin is currently being reserved by an entangler process is shown in the top plot.

Press "Run" to start the simulation.

The following parameters are used in this simulation:

Parameters:

ξ^{OB} = 0.8
F^{ent} = 1.0
g^{hf} = 42600.0
η^{opt} = 0.1
ξ^{DW} = 0.57
T_2^{e} = 0.01
F^{purc} = 10.0
ξ^{E} = 0.8
F^{meas} = 0.99
T_1^{e} = 1.0
T_2^{n} = 100.0
T_1^{n} = 100000.0
τ^{ent} = 0.015

Simulations of the generation of 3×2 cluster states in Tin-vacancy color centers

The top-left plot shows the state of the network of registers. Each register has two slots, one for an electron spin where the entanglement gets established through a Barrett-Kok protocol, and one for a nuclear spin for long term storage. The colored-line overlay on top of the registers gives the fi

Play with it at areweentangledyet.com

T

To

currently being reserved by an entangler process is shown in the top plot.

Press "Run" to start the simulation.

The following parameters are used in this simulation:

Parameters:

ξ^{OB} = 0.8
F^{ent} = 1.0
g^{hf} = 42600.0
η^{opt} = 0.1
ξ^{DW} = 0.57
T_2^{e} = 0.01
F^{purc} = 10.0
ξ^{E} = 0.8
F^{meas} = 0.99
T_1^{e} = 1.0
T_2^{n} = 100.0
T_1^{n} = 100000.0
τ^{ent} = 0.015

OTHER FEATURES...

OTHER FEATURES...

Declarative specification of "imperfections"

OTHER FEATURES...

Declarative specification of "imperfections"

Discrete event scheduling

OTHER FEATURES...

Declarative specification of "imperfections"

Discrete event scheduling

Traveling wavepackets modeling

OTHER FEATURES...

Declarative specification of "imperfections"

Discrete event scheduling

Traveling wavepackets modeling

More formalisms

OTHER FEATURES...

Declarative specification of "imperfections"

Discrete event scheduling

Traveling wavepackets modeling

More formalisms

More symbolic algebra

OTHER FEATURES...

Declarative specification of "imperfections"

Discrete event scheduling

Traveling wavepackets modeling

More formalisms

More symbolic algebra

Digital twin / surrogate modeling

QUANTUMSAVORY.JL

github.com/QuantumSavory/QuantumSavory.jl

A FEW STATE-OF-THE-ART SIMULATORS

MOST SOPHISTICATED CLIFFORD ALGEBRA SIMULATOR

github.com/QuantumSavory/QuantumClifford.jl

Multiplying two 1 gigaqubit Paulis in 32 ms.

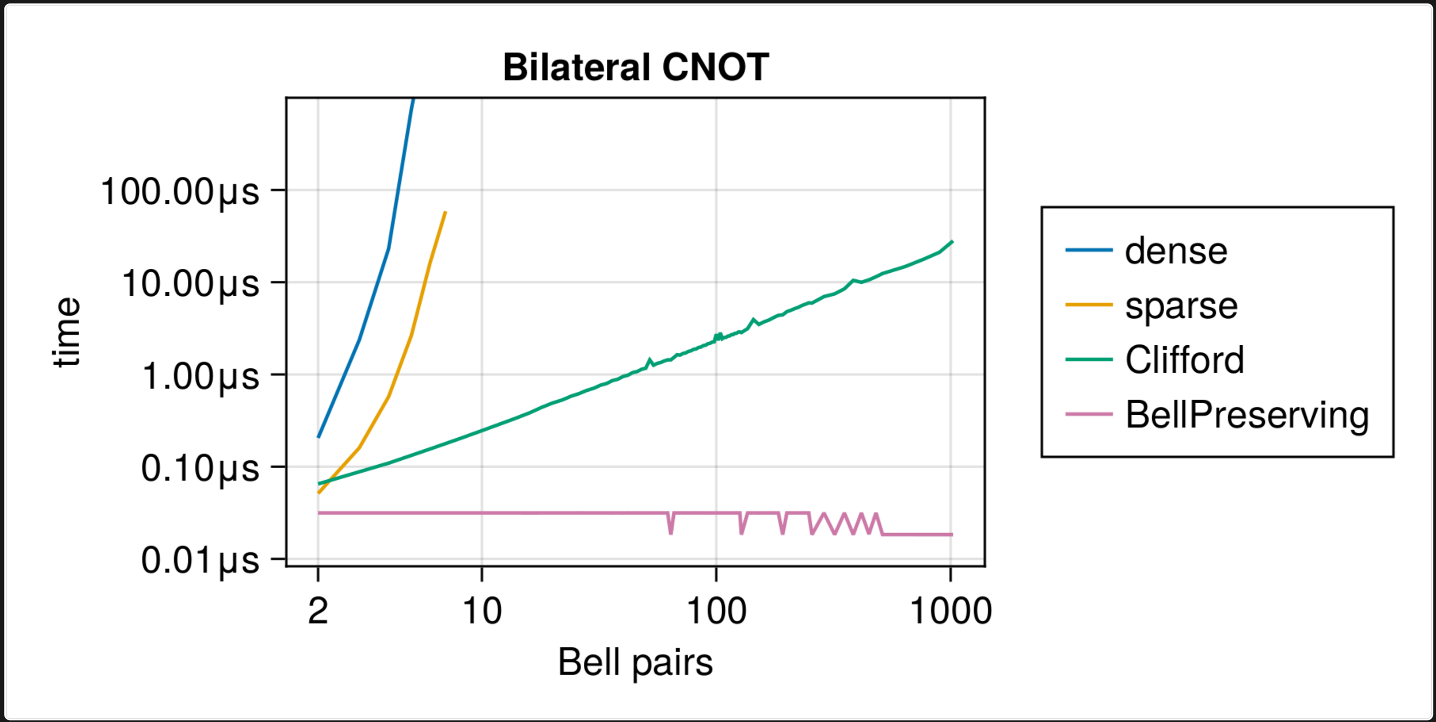
With upcoming "Google Summer of Code" contributors working on GPU acceleration and ECC zoo.

MIT and UMass students working on code generators.

Incoming master student working on code decoders.

FASTER-THAN-CLIFFORD BELL PAIR CIRCUITS

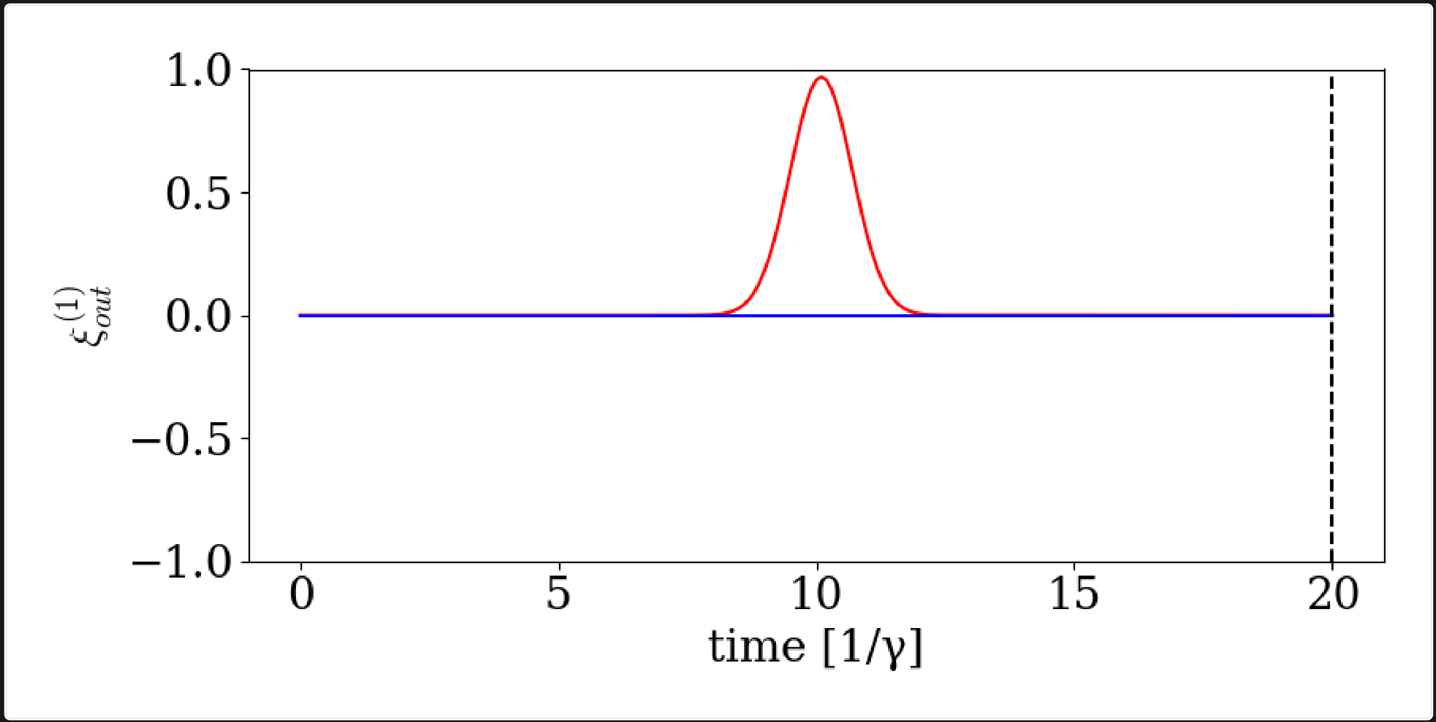
github.com/QuantumSavory/BPGates.jl



Time to perform a pair of CNOT gates, depending on formalism

WAVEGUIDE QUANTUM ELECTRODYNAMICS

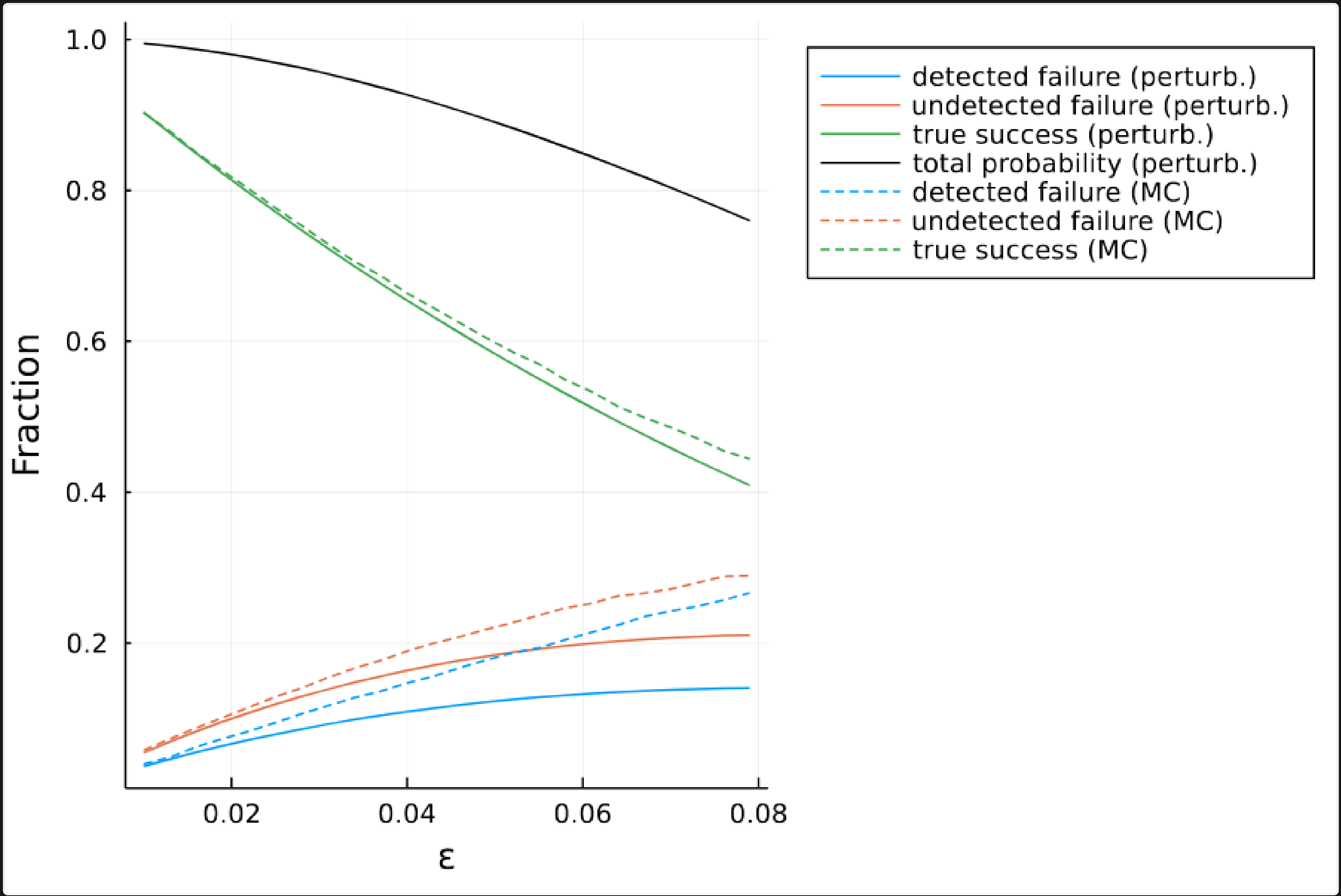
github.com/qojulia/WaveguideQED.jl



Quantum wavepacket reflected from a cavity

TAKING OPTIMIZATION SERIOUSLY

Even your Monte Carlo simulations should be "differentiable"![1]



Monte Carlo vs Perturbative Expansion results.

QUANTUMSAVORY.JL

github.com/QuantumSavory/QuantumSavory.jl

message me at
stefan@krastanov.org
skrastanov@umass.edu
stefankr@mit.edu