Quantum gradient estimation and its application to quantum reinforcement learning

A. J. Cornelissen^{1,2}

¹Applied Mathematics Delft University of Technology

²QuSoft Centrum Wiskunde & Informatica

June 5th, 2019



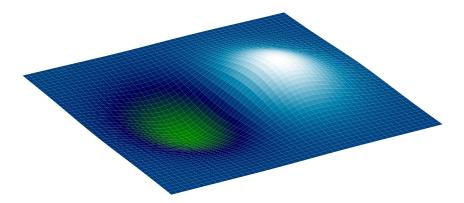


QGE and its application to QRL

A.J. Cornelissen (CWI / TU Delft)

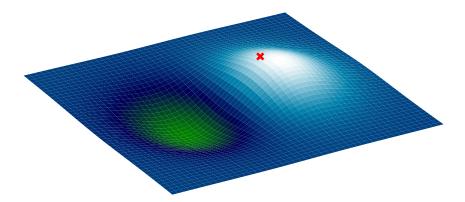
Problem: find the minimum of $f : \mathbb{R}^d \to \mathbb{R}$.

Problem: find the minimum of $f : \mathbb{R}^d \to \mathbb{R}$.



QGE and its application to QRL

Problem: find the minimum of $f : \mathbb{R}^d \to \mathbb{R}$.

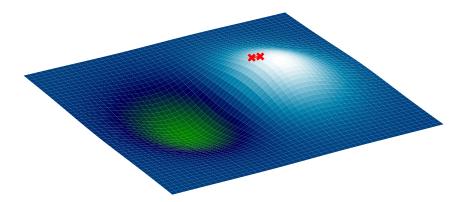


A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 2 / 21

Problem: find the minimum of $f : \mathbb{R}^d \to \mathbb{R}$.

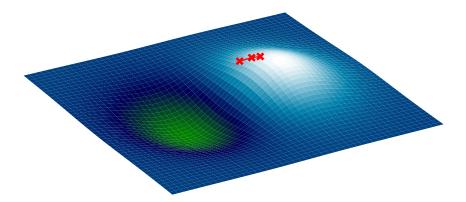


A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 2 / 21

Problem: find the minimum of $f : \mathbb{R}^d \to \mathbb{R}$.

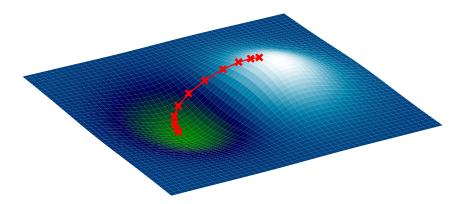


A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 2 / 21

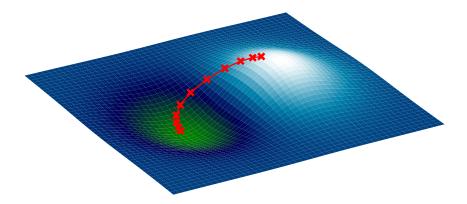
Problem: find the minimum of $f : \mathbb{R}^d \to \mathbb{R}$.



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

Problem: find the minimum of $f : \mathbb{R}^d \to \mathbb{R}$.



Can we speed up the gradient calculation step when d is large?

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

≣ ▶ < ≣ ▶ ≣ •ી ૧. June 5th, 2019 2/21

A D N A B N A B N A B N

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 3 / 21

æ

(日) (四) (日) (日) (日)

• Easiest case: let $f : \mathbb{R}^d \to \mathbb{R}$ be linear.

$$f(\mathbf{x}) = \mathbf{a} + g_1 x_1 + \dots + g_d x_d, \qquad \nabla f = \begin{bmatrix} g_1 \\ \vdots \\ g_d \end{bmatrix}$$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

≣ ► < ≣ ► Ξ ∽ < ⊂ June 5th, 2019 3/21

Image: A mathematical states and a mathem

• Easiest case: let $f : \mathbb{R}^d \to \mathbb{R}$ be linear.

$$f(\mathbf{x}) = \mathbf{a} + g_1 x_1 + \dots + g_d x_d, \qquad \nabla f = \begin{bmatrix} g_1 \\ \vdots \\ g_d \end{bmatrix}$$

• Every function evaluation yields a linear constraint on the unknowns.

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 3 / 21

• Easiest case: let $f : \mathbb{R}^d \to \mathbb{R}$ be linear.

$$f(\mathbf{x}) = \mathbf{a} + g_1 x_1 + \dots + g_d x_d, \qquad \nabla f = \begin{bmatrix} g_1 \\ \vdots \\ g_d \end{bmatrix}$$

• Every function evaluation yields a linear constraint on the unknowns.

$$\begin{bmatrix} f(\mathbf{x}^{(1)}) \\ f(\mathbf{x}^{(2)}) \\ \vdots \\ f(\mathbf{x}^{(N)}) \end{bmatrix} = \begin{bmatrix} 1 & x_1^{(1)} & \cdots & x_d^{(1)} \\ 1 & x_1^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & \cdots & x_d^{(N)} \end{bmatrix} \begin{bmatrix} a \\ g_1 \\ \vdots \\ g_d \end{bmatrix}$$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 3 / 21

• Easiest case: let $f : \mathbb{R}^d \to \mathbb{R}$ be linear.

$$f(\mathbf{x}) = \mathbf{a} + g_1 x_1 + \dots + g_d x_d, \qquad \nabla f = \begin{bmatrix} g_1 \\ \vdots \\ g_d \end{bmatrix}$$

Every function evaluation yields a linear constraint on the unknowns.

$$\begin{bmatrix} f(\mathbf{x}^{(1)}) \\ f(\mathbf{x}^{(2)}) \\ \vdots \\ f(\mathbf{x}^{(N)}) \end{bmatrix} = \begin{bmatrix} 1 & x_1^{(1)} & \cdots & x_d^{(1)} \\ 1 & x_1^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & \cdots & x_d^{(N)} \end{bmatrix} \begin{bmatrix} a \\ g_1 \\ \vdots \\ g_d \end{bmatrix}$$

• So, at least d + 1 function evaluations required classically.

• Easiest case: let $f : \mathbb{R}^d \to \mathbb{R}$ be linear.

$$f(\mathbf{x}) = \mathbf{a} + g_1 x_1 + \dots + g_d x_d, \qquad \nabla f = \begin{bmatrix} g_1 \\ \vdots \\ g_d \end{bmatrix}$$

Every function evaluation yields a linear constraint on the unknowns.

$$\begin{bmatrix} f(\mathbf{x}^{(1)}) \\ f(\mathbf{x}^{(2)}) \\ \vdots \\ f(\mathbf{x}^{(N)}) \end{bmatrix} = \begin{bmatrix} 1 & x_1^{(1)} & \cdots & x_d^{(1)} \\ 1 & x_1^{(2)} & \cdots & x_d^{(2)} \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_1^{(N)} & \cdots & x_d^{(N)} \end{bmatrix} \begin{bmatrix} a \\ g_1 \\ \vdots \\ g_d \end{bmatrix}$$

• So, at least d + 1 function evaluations required classically.

• Can we do better with a quantum computer?

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL



A D N A B N A B N A B N

Quantum gradient estimation

Visualization of quantum states

(日) (四) (日) (日) (日)

Quantum gradient estimation

- Visualization of quantum states
- **2** Visualizaiton of the quantum Fourier transform

Quantum gradient estimation

- Visualization of quantum states
- Visualization of the quantum Fourier transform
- Quantum gradient estimation algorithm

< □ > < □ > < □ > < □ >

Quantum gradient estimation

- Visualization of quantum states
- Visualization of the quantum Fourier transform
- Quantum gradient estimation algorithm
- Quantum reinforcement learning

< (17) > < (27 >)

Quantum gradient estimation

- Visualization of quantum states
- Ø Visualization of the quantum Fourier transform
- Quantum gradient estimation algorithm
- Quantum reinforcement learning
 - Quantum value estimation

Quantum gradient estimation

- Visualization of quantum states
- Ø Visualization of the quantum Fourier transform
- Quantum gradient estimation algorithm
- Quantum reinforcement learning
 - Quantum value estimation
 - Quantum policy optimization

Quantum gradient estimation

- Visualization of quantum states
- Visualization of the quantum Fourier transform
- Quantum gradient estimation algorithm
- Quantum reinforcement learning
 - Quantum value estimation
 - Quantum policy optimization
- Summary & outlook

A.J. Cornelissen (CWI / TU Delft)

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 5 / 21

æ

(日) (四) (日) (日) (日)

 An *n*-qubit state |ψ⟩ is a unit vector in C^{2ⁿ}:

$$|\psi\rangle = \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{2^n-1} \end{bmatrix} = \sum_{j=0}^{2^n-1} \alpha_j |j\rangle$$

For all j: $|\alpha_j| \leq 1$.

(日) (四) (日) (日) (日)

• An *n*-qubit state $|\psi\rangle$ is a unit vector in \mathbb{C}^{2^n} :

$$|\psi\rangle = \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{2^n-1} \end{bmatrix} = \sum_{j=0}^{2^n-1} \alpha_j |j\rangle$$

0> 1> 2> 3> 4> 5> 6> 7>

Image: A mathematical states and a mathem

For all $j: |\alpha_j| \leq 1$.

-

• An *n*-qubit state $|\psi\rangle$ is a unit vector in \mathbb{C}^{2^n} :

$$|\psi\rangle = \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{2^n-1} \end{bmatrix} = \sum_{j=0}^{2^n-1} \alpha_j |j\rangle$$

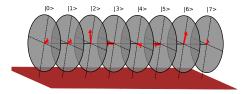
0> 1> 2> 3> 4> 5> 6> 7>

Image: A mathematical states and a mathem

For all $j: |\alpha_j| \leq 1$.

• An *n*-qubit state $|\psi\rangle$ is a unit vector in \mathbb{C}^{2^n} :

$$|\psi\rangle = \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{2^n-1} \end{bmatrix} = \sum_{j=0}^{2^n-1} \alpha_j |j\rangle$$



A B A B A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 A
 A
 A
 A

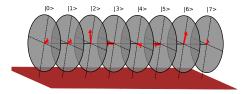
For all $j: |\alpha_i| \leq 1$.

Quantum gates move the arrows around. ۰

4 E b

 An *n*-qubit state |ψ⟩ is a unit vector in C^{2ⁿ}:

$$|\psi\rangle = \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{2^n-1} \end{bmatrix} = \sum_{j=0}^{2^n-1} \alpha_j |j\rangle$$



For all j: $|\alpha_j| \leq 1$.

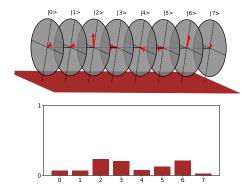
- Quantum gates move the arrows around.
- The probability of getting outcome j is the length of the arrow in circle |j>.

 An *n*-qubit state |ψ⟩ is a unit vector in C^{2ⁿ}:

$$|\psi\rangle = \begin{bmatrix} \alpha_0 \\ \vdots \\ \alpha_{2^n-1} \end{bmatrix} = \sum_{j=0}^{2^n-1} \alpha_j |j\rangle$$

For all j: $|\alpha_j| \leq 1$.

- Quantum gates move the arrows around.
- The probability of getting outcome j is the length of the arrow in circle |j>.



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 6 / 21

3

イロト イヨト イヨト イヨト

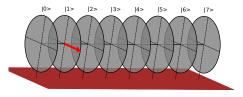
• The *n*-qubit quantum Fourier transform is defined as:

$$\mathsf{QFT}_{2^n}: |j
angle \mapsto rac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{rac{2\pi ijk}{2^n}} |k
angle$$

A D N A B N A B N A B N

• The *n*-qubit quantum Fourier transform is defined as:

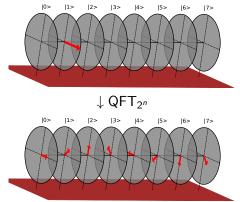
$$\mathsf{QFT}_{2^n}: |j\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \mathsf{e}^{\frac{2\pi j jk}{2^n}} |k\rangle$$



A D N A B N A B N A B N

• The *n*-qubit quantum Fourier transform is defined as:

$$\mathsf{QFT}_{2^n}: |j
angle \mapsto rac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{rac{2\pi ijk}{2^n}} |k
angle$$



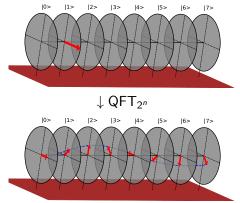
A D N A B N A B N A B N

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

• The *n*-qubit quantum Fourier transform is defined as:

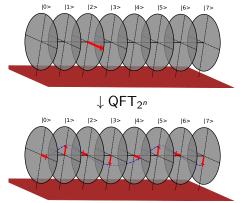
$$\mathsf{QFT}_{2^n}: |j
angle \mapsto rac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{rac{2\pi ijk}{2^n}} |k
angle$$



A D N A B N A B N A B N

• The *n*-qubit quantum Fourier transform is defined as:

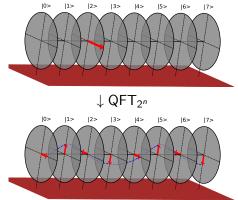
$$\mathsf{QFT}_{2^n}: |j
angle \mapsto rac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{rac{2\pi ijk}{2^n}} |k
angle$$



• The *n*-qubit quantum Fourier transform is defined as:

$$\mathsf{QFT}_{2^n}: \ket{j} \mapsto rac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{rac{2\pi i j k}{2^n}} \ket{k}$$

 The state QFT_{2ⁿ} |j> can be visualized as a helix making j revolutions.

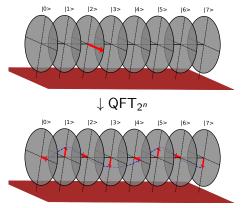


• The *n*-qubit quantum Fourier transform is defined as:

$$\mathsf{QFT}_{2^n}: |j\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \mathsf{e}^{\frac{2\pi j j k}{2^n}} |k\rangle$$

- The state QFT_{2n} |j> can be visualized as a helix making j revolutions.
- The inverse QFT counts the number of revolutions:

$$\mathsf{QFT}_{2^n}^\dagger: \ket{j}\mapsto rac{1}{\sqrt{2^n}}\sum_{k=0}^{2^n-1} e^{-rac{2\pi i jk}{2^n}}\ket{k}$$

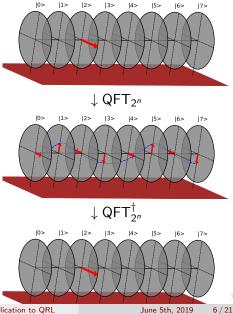


• The *n*-qubit quantum Fourier transform is defined as:

$$\mathsf{QFT}_{2^n}: |j\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \mathsf{e}^{\frac{2\pi j jk}{2^n}} |k\rangle$$

- The state QFT_{2n} |j> can be visualized as a helix making j revolutions.
- The inverse QFT counts the number of revolutions:

$$\mathsf{QFT}_{2^n}^\dagger: |j
angle \mapsto rac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{-rac{2\pi i jk}{2^n}} |k
angle$$



QGE and its application to QRL

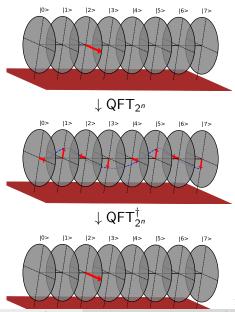
• The *n*-qubit quantum Fourier transform is defined as:

$$\mathsf{QFT}_{2^n}: |j\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{\frac{2\pi i jk}{2^n}} |k\rangle$$

- The state QFT_{2n} |j> can be visualized as a helix making j revolutions.
- The inverse QFT counts the number of revolutions:

$$\mathsf{QFT}_{2^n}^\dagger: |j\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} e^{-\frac{2\pi i jk}{2^n}} |k\rangle$$

• Efficient implementations available.



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 6 / 21

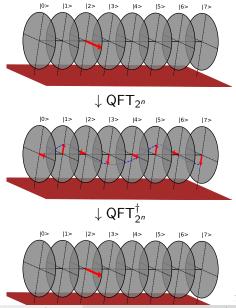
The n-qubit quantum Fourier transform is defined as:

$$\mathsf{QFT}_{2^n}: |j\rangle \mapsto \frac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \mathsf{e}^{\frac{2\pi i jk}{2^n}} |k\rangle$$

- The state QFT_{2n} |j> can be visualized as a helix making j revolutions.
- The inverse QFT counts the number of revolutions:

$$\mathsf{QFT}_{2^n}^\dagger: \ket{j} \mapsto rac{1}{\sqrt{2^n}} \sum_{k=0}^{2^n-1} \mathrm{e}^{-rac{2\pi i j k}{2^n}} \ket{k}$$

- Efficient implementations available.
- Also works for non-integer revolutions.



QGE and its application to QRL

June 5th, 2019 6 / 21

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 7 / 21

æ

(日) (四) (日) (日) (日)

• Let $f : \mathbb{R} \to \mathbb{R}$.

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

≧ ► ◀ **≧** ► **≧** ∽ ∝ ⊂ June 5th, 2019 7/21

(日) (四) (日) (日) (日)

• Let $f : \mathbb{R} \to \mathbb{R}$.

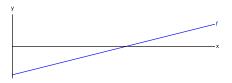
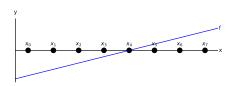
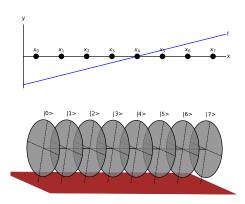


Image: A mathematical states and a mathem

- Let $f : \mathbb{R} \to \mathbb{R}$.
- Let $G = \{x_0, \ldots, x_{2^n-1}\} \subseteq \mathbb{R}$.

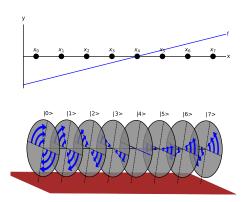


- Let $f : \mathbb{R} \to \mathbb{R}$.
- Let $G = \{x_0, \ldots, x_{2^n-1}\} \subseteq \mathbb{R}$.
- We associate every state |j> to the point x_j in the domain of f.



- Let $f : \mathbb{R} \to \mathbb{R}$.
- Let $G = \{x_0, \ldots, x_{2^n-1}\} \subseteq \mathbb{R}$.
- We associate every state |j> to the point x_j in the domain of f.
- We can evaluate *f* as follows:

 $O_f:\ket{j}\mapsto e^{if(x_j)}\ket{j}$

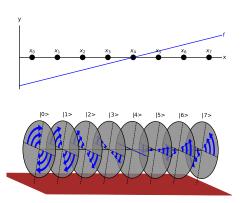


▶ ∢ ⊒

- Let $f : \mathbb{R} \to \mathbb{R}$.
- Let $G = \{x_0, \ldots, x_{2^n-1}\} \subseteq \mathbb{R}$.
- We associate every state |j> to the point x_j in the domain of f.
- We can evaluate *f* as follows:

 $O_f:\ket{j}\mapsto e^{if(x_j)}\ket{j}$

• This is called the phase oracle of *f* on *G*.

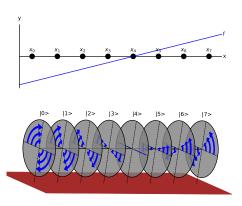


▶ < ∃ >

- Let $f : \mathbb{R} \to \mathbb{R}$.
- Let $G = \{x_0, \ldots, x_{2^n-1}\} \subseteq \mathbb{R}$.
- We associate every state |j> to the point x_j in the domain of f.
- We can evaluate *f* as follows:

 $O_f:\ket{j}\mapsto e^{if(x_j)}\ket{j}$

- This is called the phase oracle of *f* on *G*.
- One application of this phase oracle is one *quantum function evaluation*.



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

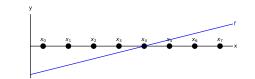
∃ ► < ∃ ► ∃

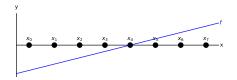
 O <

 O

< □ > < 同 > < 回 > < 回 > < 回 >

Let $f : \mathbb{R} \to \mathbb{R}$ linear with $|f'| \leq C$.



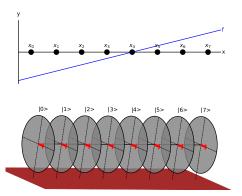


Let $f : \mathbb{R} \to \mathbb{R}$ linear with $|f'| \leq C$.

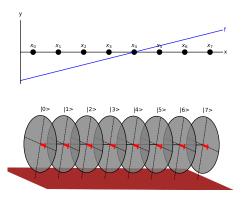
Create a uniform superposition over the grid.

Let $f : \mathbb{R} \to \mathbb{R}$ linear with $|f'| \leq C$.

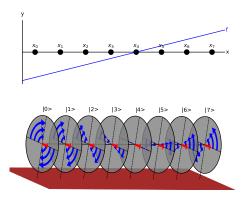
Create a uniform superposition over the grid.



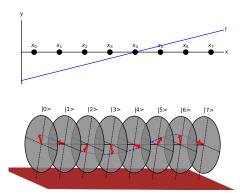
- Create a uniform superposition over the grid.
- 2 Apply the phase oracle O_f .



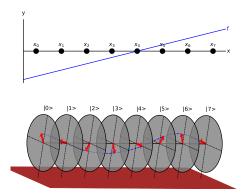
- Create a uniform superposition over the grid.
- 2 Apply the phase oracle O_f .



- Create a uniform superposition over the grid.
- Apply the phase oracle O_f.



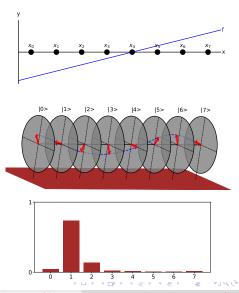
- Create a uniform superposition over the grid.
- 2 Apply the phase oracle O_f .
- Apply the inverse QFT.
- Measure.



Let $f : \mathbb{R} \to \mathbb{R}$ linear with $|f'| \leq C$.

- Create a uniform superposition over the grid.
- 2 Apply the phase oracle O_f .
- Apply the inverse QFT.

Measure.

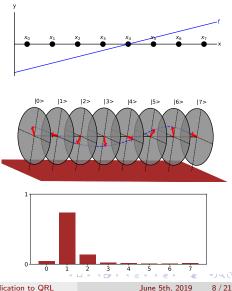


QGE and its application to QRL

Let $f : \mathbb{R} \to \mathbb{R}$ linear with $|f'| \leq C$.

- Create a uniform superposition over the grid.
- Apply the phase oracle O_f.
- Apply the inverse QFT.
- Measure.

Generalizes to $f : \mathbb{R}^d \to \mathbb{R}$. (Jordan, 2004)



QGE and its application to QRL

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 9 / 21

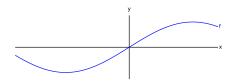
3

< □ > < 同 > < 回 > < 回 > < 回 >

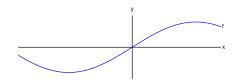
• Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).

< □ > < 同 > < 回 > < 回 > < 回 >

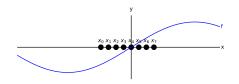
• Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).



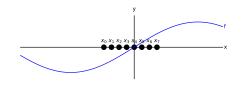
- Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).
- Naive approach: {x₀,..., x_{2ⁿ-1}} tight around the origin.

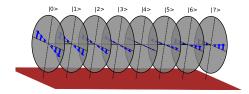


- Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).
- Naive approach: {x₀,..., x_{2ⁿ-1}} tight around the origin.



- Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).
- Naive approach: {x₀,..., x_{2ⁿ-1}} tight around the origin.





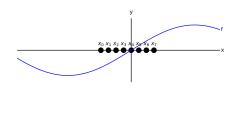
→ ∢ ∃

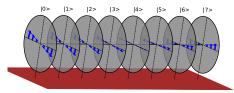
A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 9 / 21

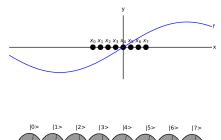
- Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).
- Naive approach: {x₀,..., x_{2ⁿ-1}} tight around the origin.
- Problems:

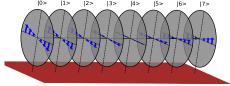




→ ∢ ∃

- Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).
- Naive approach: {x₀,..., x_{2ⁿ-1}} tight around the origin.
- Problems:
 - Rotations become very small.



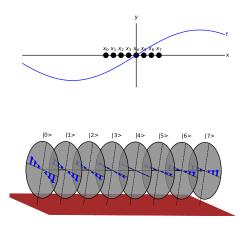


A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 9 / 21

- Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).
- Naive approach: {x₀,..., x_{2ⁿ-1}} tight around the origin.
- Problems:
 - Rotations become very small.
 - Function evaluations must be very precise.

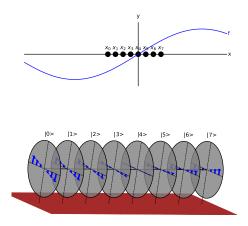


A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 9 / 21

- Let $f : \mathbb{R} \to \mathbb{R}$, want to find f'(0).
- Naive approach: {x₀,..., x_{2ⁿ-1}} tight around the origin.
- Problems:
 - Rotations become very small.
 - Function evaluations must be very precise.
- Key idea: central difference method to extend region of approximate linearity.



Central difference method

A.J. Cornelissen (CWI / TU Delft) QGE a

QGE and its application to QRL

June 5th, 2019 10 / 21

2

Let f : ℝ^d → ℝ and m > 0. We define:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \sum_{\ell=-m}^{m} a_{\ell}^{(2m)} f(\ell \mathbf{x})$$

• such that:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \nabla f(\mathbf{0}) \cdot \mathbf{x} + \mathcal{O}\left(\|\mathbf{x}\|^{2m+1} \right)$$

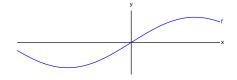
A.J. Cornelissen (CWI / TU Delft)

- 20

A D N A B N A B N A B N

Let f : ℝ^d → ℝ and m > 0. We define:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \sum_{\ell=-m}^{m} a_{\ell}^{(2m)} f(\ell \mathbf{x})$$



• such that:

$$\widetilde{f}_{(2m)}(\mathbf{x}) =
abla f(\mathbf{0}) \cdot \mathbf{x} + \mathcal{O}\left(\|\mathbf{x}\|^{2m+1}
ight)$$

A.J. Cornelissen (CWI / TU Delft)

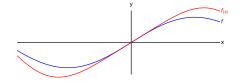
QGE and its application to QRL

June 5th, 2019 10 / 21

э

Let f : ℝ^d → ℝ and m > 0. We define:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \sum_{\ell=-m}^{m} a_{\ell}^{(2m)} f(\ell \mathbf{x})$$



• such that:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \nabla f(\mathbf{0}) \cdot \mathbf{x} + \mathcal{O}\left(\|\mathbf{x}\|^{2m+1} \right)$$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 10 / 21

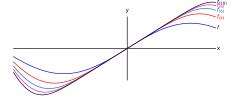
э

• Let $f : \mathbb{R}^d \to \mathbb{R}$ and m > 0. We define:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \sum_{\ell=-m}^{m} a_{\ell}^{(2m)} f(\ell \mathbf{x})$$

• such that:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \nabla f(\mathbf{0}) \cdot \mathbf{x} + \mathcal{O}\left(\|\mathbf{x}\|^{2m+1} \right)$$



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

-June 5th, 2019 10/21

э

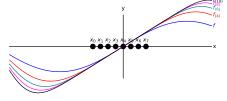
A (1) > A (2) > A

Let f : ℝ^d → ℝ and m > 0. We define:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \sum_{\ell=-m}^{m} a_{\ell}^{(2m)} f(\ell \mathbf{x})$$

• such that:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \nabla f(\mathbf{0}) \cdot \mathbf{x} + \mathcal{O}\left(\|\mathbf{x}\|^{2m+1} \right)$$



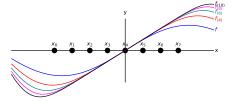
э

• Let $f : \mathbb{R}^d \to \mathbb{R}$ and m > 0. We define:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \sum_{\ell=-m}^{m} a_{\ell}^{(2m)} f(\ell \mathbf{x})$$

• such that:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \nabla f(\mathbf{0}) \cdot \mathbf{x} + \mathcal{O}\left(\|\mathbf{x}\|^{2m+1} \right)$$



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 10 / 21

э

• Let $f : \mathbb{R}^d \to \mathbb{R}$ and m > 0. We define:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \sum_{\ell=-m}^{m} a_{\ell}^{(2m)} f(\ell \mathbf{x})$$

• such that:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \nabla f(\mathbf{0}) \cdot \mathbf{x} + \mathcal{O}\left(\|\mathbf{x}\|^{2m+1} \right)$$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 10 / 21

э

Let f : ℝ^d → ℝ and m > 0. We define:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \sum_{\ell=-m}^{m} a_{\ell}^{(2m)} f(\ell \mathbf{x})$$
such that:

$$\widetilde{f}_{(2m)}(\mathbf{x}) = \nabla f(\mathbf{0}) \cdot \mathbf{x} + \mathcal{O}\left(||\mathbf{x}||^{2m+1} \right)$$
One can implement $O_{\widetilde{f}_{(2m)}}$ using $\widetilde{\mathcal{O}}(m)$ queries to O_{f} .
(Gilyén, Arunachalam, Wiebe, 2018)

A.J. Cornelissen (CWI / TU Delft)

۲

QGE and its application to QRL

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 11 / 21

(日) (四) (日) (日) (日)

Case 1: Polynomial

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 11 / 21

3

Case 1: Polynomial

 Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.

3

(日) (四) (日) (日) (日)

Case 1: Polynomial

- Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.
- Then $\tilde{f}_{(k)}$ is linear.

3

- 4 回 ト - 4 三 ト

Case 1: Polynomial

- Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.
- Then $\tilde{f}_{(k)}$ is linear.
- So gradient estimation takes \$\tilde{O}(k)\$ queries.

3

Case 1: Polynomial

- Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.
- Then $\tilde{f}_{(k)}$ is linear.
- So gradient estimation takes \$\tilde{O}(k)\$ queries.

Smoothness	Polynomial		
condition	degree <i>k</i>		
Best known algorithm	$\widetilde{\mathcal{O}}(k)$		
Best known lower bound	$\Omega(1)$		

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 11 / 21

3

Case 2: Gevrey

Case 1: Polynomial

- Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.
- Then $\tilde{f}_{(k)}$ is linear.
- So gradient estimation takes \$\tilde{O}(k)\$ queries.

Smoothness	Polynomial		
condition	degree <i>k</i>		
Best known algorithm	$\widetilde{\mathcal{O}}(k)$		
Best known lower bound	$\Omega(1)$		

Case 2: Gevrey

Let f : ℝ^d → ℝ have a convergent Taylor series:

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\alpha \in [d]^k} \frac{\partial_{\alpha} f(\mathbf{0})}{k!} \mathbf{x}^{\alpha}$$

Case 1: Polynomial

- Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.
- Then $\tilde{f}_{(k)}$ is linear.

Smoothness	Polynomial		
condition	degree k		
Best known algorithm	$\widetilde{\mathcal{O}}(k)$		
Best known lower bound	$\Omega(1)$		

A.J. Cornelissen (CWI / TU Delft)

Case 2: Gevrey

Let f : ℝ^d → ℝ have a convergent Taylor series:

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\alpha \in [d]^k} \frac{\partial_{\alpha} f(\mathbf{0})}{k!} \mathbf{x}^{\alpha}$$

• Let $\sigma \in [\frac{1}{2}, 1]$:

 $|\partial_{\alpha}f(\mathbf{0})| \leq (k!)^{\sigma}$

・ 何 ト ・ ヨ ト ・ ヨ ト

Smoothness	Polynomial		
condition	degree k		
Best known algorithm	$\widetilde{\mathcal{O}}(k)$		
Best known lower bound	$\Omega(1)$		

Case 1: Polynomial

queries.

• Then $\tilde{f}_{(k)}$ is linear.

 Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.

• So gradient estimation takes $\widetilde{\mathcal{O}}(k)$

Case 2: Gevrey

Let f : ℝ^d → ℝ have a convergent Taylor series:

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\alpha \in [d]^k} \frac{\partial_{\alpha} f(\mathbf{0})}{k!} \mathbf{x}^{\alpha}$$

• Let $\sigma \in [\frac{1}{2}, 1]$:

 $|\partial_{lpha} f(\mathbf{0})| \leq (k!)^{\sigma}$

・ 何 ト ・ ヨ ト ・ ヨ ト

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$			
Best known lower bound	$\Omega(1)$			

Case 1: Polynomial

queries.

• Then $\tilde{f}_{(k)}$ is linear.

 Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.

• So gradient estimation takes $\widetilde{\mathcal{O}}(k)$

Case 2: Gevrey

Let f : ℝ^d → ℝ have a convergent Taylor series:

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\alpha \in [d]^k} \frac{\partial_{\alpha} f(\mathbf{0})}{k!} \mathbf{x}^{\alpha}$$

• Let $\sigma \in [\frac{1}{2}, 1]$:

 $|\partial_{\alpha}f(\mathbf{0})| \leq (k!)^{\sigma}$

・ 何 ト ・ ヨ ト ・ ヨ ト

Smoothness	Polynomial		Gevrey	
condition	degree <i>k</i>	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	
Best known lower bound	$\Omega(1)$			

Case 1: Polynomial

queries.

• Then $\tilde{f}_{(k)}$ is linear.

 Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.

• So gradient estimation takes $\widetilde{\mathcal{O}}(k)$

Case 2: Gevrey

Let f : ℝ^d → ℝ have a convergent Taylor series:

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\alpha \in [d]^k} \frac{\partial_{\alpha} f(\mathbf{0})}{k!} \mathbf{x}^{\alpha}$$

• Let $\sigma \in [\frac{1}{2}, 1]$:

 $|\partial_{lpha} f(\mathbf{0})| \leq (k!)^{\sigma}$

・ 何 ト ・ ヨ ト ・ ヨ ト

Smoothness	Polynomial		Gevrey	
condition	degree <i>k</i>	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$			

Case 1: Polynomial

queries.

• Then $\tilde{f}_{(k)}$ is linear.

 Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.

• So gradient estimation takes $\widetilde{\mathcal{O}}(k)$

June 5th, 2019 11 / 21

Case 2: Gevrey

Let f : ℝ^d → ℝ have a convergent Taylor series:

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\alpha \in [d]^k} \frac{\partial_{\alpha} f(\mathbf{0})}{k!} \mathbf{x}^{\alpha}$$

• Let $\sigma \in [\frac{1}{2}, 1]$:

 $|\partial_{\alpha}f(\mathbf{0})| \leq (k!)^{\sigma}$

・ 何 ト ・ ヨ ト ・ ヨ ト

Smoothness	Polynomial	Gevrey		
condition	degree <i>k</i>	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$		$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

Case 1: Polynomial

- Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.
- Then $\tilde{f}_{(k)}$ is linear.
- So gradient estimation takes \$\tilde{O}(k)\$ queries.

Case 2: Gevrey

Let f : ℝ^d → ℝ have a convergent Taylor series:

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\alpha \in [d]^k} \frac{\partial_{\alpha} f(\mathbf{0})}{k!} \mathbf{x}^{\alpha}$$

• Let $\sigma \in [\frac{1}{2}, 1]$:

 $|\partial_{lpha} f(\mathbf{0})| \leq (k!)^{\sigma}$

・ 何 ト ・ ヨ ト ・ ヨ ト

Smoothness	Polynomial	Gevrey		
condition	degree <i>k</i>	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

Case 1: Polynomial

queries.

• Then $\tilde{f}_{(k)}$ is linear.

 Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.

• So gradient estimation takes $\widetilde{\mathcal{O}}(k)$

Case 2: Gevrey

Let f : ℝ^d → ℝ have a convergent Taylor series:

$$f(\mathbf{x}) = \sum_{k=0}^{\infty} \sum_{\alpha \in [d]^k} \frac{\partial_{\alpha} f(\mathbf{0})}{k!} \mathbf{x}^{\alpha}$$

• Let $\sigma \in [\frac{1}{2}, 1]$:

 $|\partial_{\alpha}f(\mathbf{0})| \leq (k!)^{\sigma}$

・ 何 ト ・ ヨ ト ・ ヨ ト

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$
From ℓ^∞ to ℓ^p approximat	ions: multiply	upper and lov	wer bound	s by $\Theta(d^{\frac{1}{p}})$.

Case 1: Polynomial

- Let f : ℝ^d → ℝ be a multivariate polynomial of total degree k.
- Then $\tilde{f}_{(k)}$ is linear.
- So gradient estimation takes \$\tilde{O}(k)\$ queries.

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 12 / 21

イロト イポト イヨト イヨト

- 2

• How to construct a phase oracle?

A.J. Cornelissen (CWI / TU Delft) QGE

QGE and its application to QRL

June 5th, 2019 12 / 21

3

Image: A match a ma

• How to construct a phase oracle?

 $\begin{array}{|c|c|} \hline \textbf{Binary oracle} \\ B_{f}: \ket{\textbf{x}}\ket{0} \mapsto \ket{\textbf{x}}\ket{f(\textbf{x})} \end{array} \end{array}$

• How to construct a phase oracle?

	Binary oracle
B_f :	$\begin{array}{l} \textbf{Binary oracle} \\ \left \textbf{x} \right\rangle \left 0 \right\rangle \mapsto \left \textbf{x} \right\rangle \left f(\textbf{x}) \right\rangle \end{array}$

$$\begin{array}{|c|} \hline \textbf{Phase oracle} \\ O_f: \ket{\textbf{x}} \mapsto e^{if(\textbf{x})} \ket{\textbf{x}} \end{array}$$

A.J. Cornelissen (CWI / TU Delft)

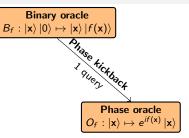
QGE and its application to QRL

June 5th, 2019 12 / 21

э

< A > <

• How to construct a phase oracle?

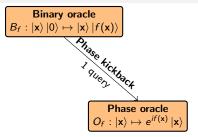


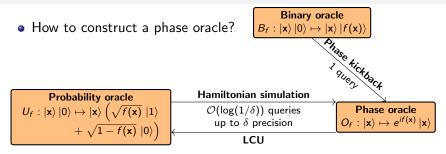
A.J. Cornelissen (CWI / TU Delft)

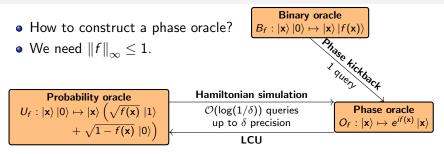
QGE and its application to QRL

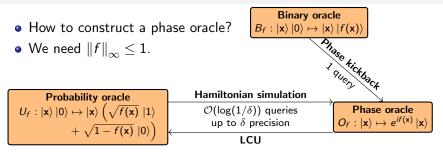
• How to construct a phase oracle?

$$\begin{array}{l} \textbf{Probability oracle} \\ U_f: \ket{\mathtt{x}}\ket{\mathtt{0}} \mapsto \ket{\mathtt{x}} \left(\sqrt{f(\mathtt{x})} \ \ket{\mathtt{1}} \\ + \sqrt{1 - f(\mathtt{x})} \ \ket{\mathtt{0}} \right) \end{array}$$





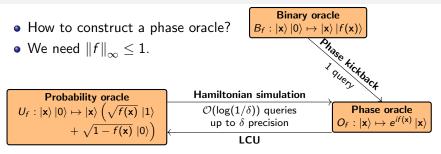




Analog arithmetical operations:

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

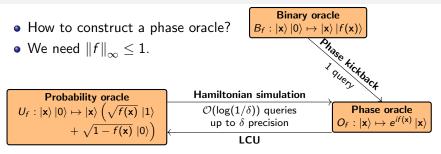


Analog arithmetical operations:

• Addition: consecutive applications of phase oracles.

$$O_f O_g : \ket{\mathbf{x}} \mapsto e^{i(f(\mathbf{x}) + g(\mathbf{x}))} \ket{\mathbf{x}}$$

A.J. Cornelissen (CWI / TU Delft)



Analog arithmetical operations:

• Addition: consecutive applications of phase oracles.

$$O_f O_g : \ket{\mathbf{x}} \mapsto e^{i(f(\mathbf{x}) + g(\mathbf{x}))} \ket{\mathbf{x}}$$

• Multiplication: consecutive applications of probability oracles.

$$(U_f)_1(U_g)_2: \ket{\mathbf{x}}\ket{00}\mapsto \sqrt{f(\mathbf{x})g(\mathbf{x})}\ket{\mathbf{x}}\ket{11}+\ket{\perp}$$

▲ □ ▶ ▲ □ ▶ ▲ □

Markov reward processes

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 13 / 21

- 2

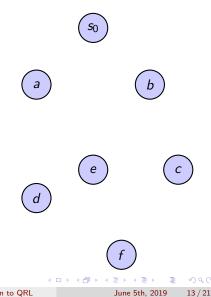
Markov reward processes

Let S be a state space and s₀ ∈ S some initial state.

- 20

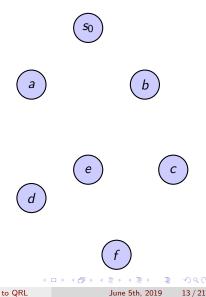
< □ > < 同 > < 回 > < 回 > < 回 >

 Let S be a state space and s₀ ∈ S some initial state.



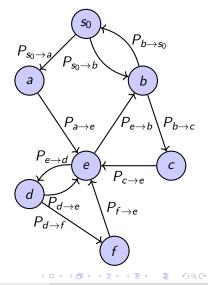
A.J. Cornelissen (CWI / TU Delft)

- Let S be a state space and s₀ ∈ S some initial state.
- Let P_{s→s'} be the transition probability function.



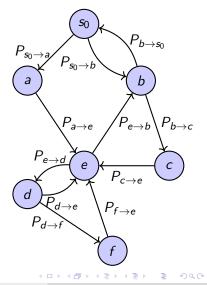
A.J. Cornelissen (CWI / TU Delft)

- Let S be a state space and s₀ ∈ S some initial state.
- Let P_{s→s'} be the transition probability function.



- Let S be a state space and s₀ ∈ S some initial state.
- Let P_{s→s'} be the transition probability function.

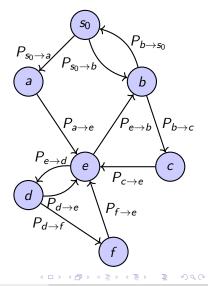
$$\mathcal{P}: \left| s
ight
angle \left| 0
ight
angle \mapsto \left| s
ight
angle \sum_{s' \in S} \sqrt{P_{s
ightarrow s'}} \left| s'
ight
angle$$



- Let S be a state space and s₀ ∈ S some initial state.
- Let P_{s→s'} be the transition probability function.

$$\mathcal{P}: \ket{s}\ket{0}\mapsto \ket{s}\sum_{s'\in S}\sqrt{P_{s
ightarrow s'}}\ket{s'}$$

• Let R(s) be the reward that you obtain at state s.



A.J. Cornelissen (CWI / TU Delft)

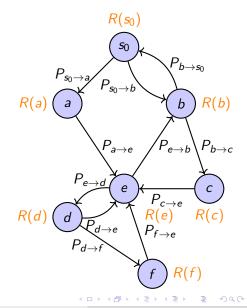
QGE and its application to QRL

June 5th, 2019 13 / 21

- Let S be a state space and s₀ ∈ S some initial state.
- Let P_{s→s'} be the transition probability function.

$$\mathcal{P}: \ket{s}\ket{0}\mapsto \ket{s}\sum_{s'\in S}\sqrt{P_{s
ightarrow s'}}\ket{s'}$$

• Let R(s) be the reward that you obtain at state *s*.

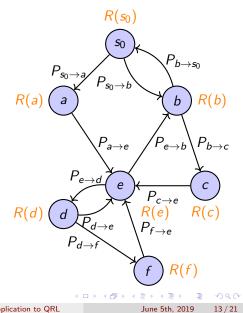


- Let S be a state space and $s_0 \in S$ some initial state.
- Let $P_{s \to s'}$ be the transition probability function.

$$\mathcal{P}: \ket{s}\ket{0}\mapsto \ket{s}\sum_{s'\in S}\sqrt{P_{s
ightarrow s'}}\ket{s'}$$

• Let R(s) be the reward that you obtain at state s.

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$



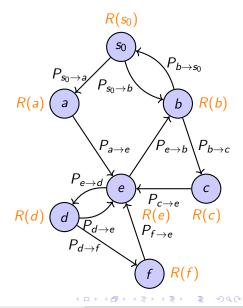
- Let S be a state space and s₀ ∈ S some initial state.
- Let P_{s→s'} be the transition probability function.

$$\mathcal{P}: \ket{s}\ket{0}\mapsto \ket{s}\sum_{s'\in S}\sqrt{P_{s
ightarrow s'}}\ket{s'}$$

• Let R(s) be the reward that you obtain at state s.

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

• Let $0 < \gamma < 1$.



- Let S be a state space and s₀ ∈ S some initial state.
- Let P_{s→s'} be the transition probability function.

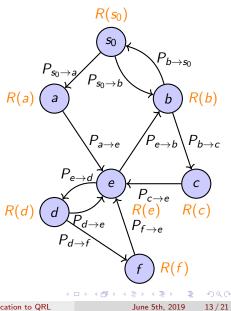
$$\mathcal{P}: \ket{s}\ket{0}\mapsto \ket{s}\sum_{s'\in S}\sqrt{P_{s
ightarrow s'}}\ket{s'}$$

• Let R(s) be the reward that you obtain at state s.

$$\mathcal{R}: \ket{s} \mapsto e^{i R(s)} \ket{s}$$

- Let $0 < \gamma < 1$.
- Problem: evaluate the value function:

$$V(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$



- Let S be a state space and s₀ ∈ S some initial state.
- Let P_{s→s'} be the transition probability function.

$$\mathcal{P}: \ket{s}\ket{0}\mapsto \ket{s}\sum_{s'\in S}\sqrt{P_{s
ightarrow s'}}\ket{s'}$$

• Let R(s) be the reward that you obtain at state s.

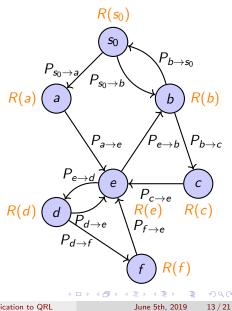
$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

- Let $0 < \gamma < 1$.
- Problem: evaluate the value function:

$$V(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right]$$

• Quantum value estimation

A.J. Cornelissen (CWI / TU Delft)



A.J. Cornelissen (CWI / TU Delft) QGE a

QGE and its application to QRL

June 5th, 2019 14 / 21

(日) (四) (日) (日) (日)

• Let's consider the tree of possible paths.

э

・ 同 ト ・ ヨ ト ・ ヨ

• Let's consider the tree of possible paths.



A.J. Cornelissen (CWI / TU Delft)

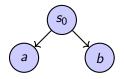
QGE and its application to QRL

June 5th, 2019 14 / 21

э

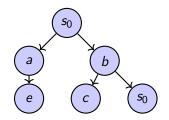
(日) (四) (日) (日) (日)

• Let's consider the tree of possible paths.



A.J. Cornelissen (CWI / TU Delft)

• Let's consider the tree of possible paths.

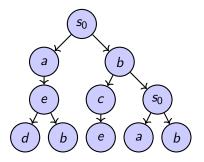


A.J. Cornelissen (CWI / TU Delft)

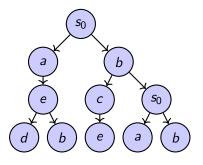
QGE and its application to QRL

June 5th, 2019 14 / 21

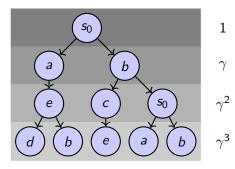
• Let's consider the tree of possible paths.



• Let's consider the tree of possible paths.



• Let's consider the tree of possible paths.



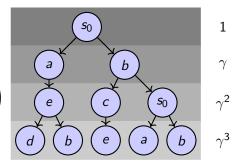
A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 14 / 21

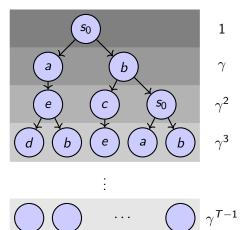
- Let's consider the tree of possible paths.
- Cutoff at:

$$\mathcal{T} = \Theta\left(rac{1}{1-\gamma}\log\left(rac{|\mathcal{R}|_{\max}}{arepsilon(1-\gamma)}
ight)
ight)$$



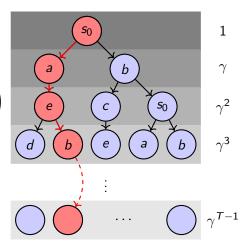
- Let's consider the tree of possible paths.
- Cutoff at:

$$T = \Theta\left(\frac{1}{1-\gamma}\log\left(\frac{|R|_{\max}}{\varepsilon(1-\gamma)}\right)\right)$$



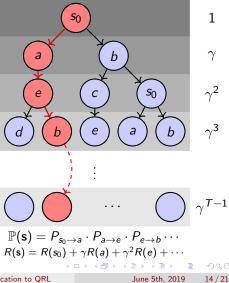
- Let's consider the tree of possible paths.
- Cutoff at:

$$T = \Theta\left(rac{1}{1-\gamma}\log\left(rac{|R|_{\max}}{arepsilon(1-\gamma)}
ight)
ight)$$



- Let's consider the tree of possible paths.
- Cutoff at:

$$\mathcal{T} = \Theta\left(rac{1}{1-\gamma}\log\left(rac{|R|_{\max}}{arepsilon(1-\gamma)}
ight)
ight)$$



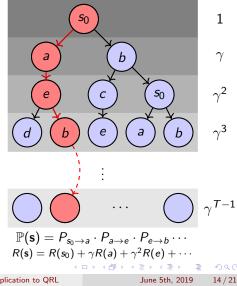
A.J. Cornelissen (CWI / TU Delft)

- Let's consider the tree of possible paths.
- Cutoff at:

$$\mathcal{T} = \Theta\left(rac{1}{1-\gamma}\log\left(rac{|R|_{\mathsf{max}}}{arepsilon(1-\gamma)}
ight)
ight)$$

 Value function approximately equal to:

$$V(s_0) = \sum_{\mathbf{s} \in S^{T-1}} \mathbb{P}(\mathbf{s}) R(\mathbf{s}) + \mathcal{O}(\varepsilon)$$



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 15 / 21

- 2

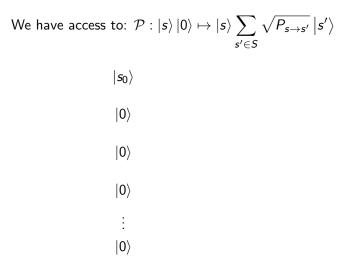
We have access to:
$$\mathcal{P}: \ket{s}\ket{0} \mapsto \ket{s} \sum_{s' \in S} \sqrt{P_{s \to s'}} \ket{s'}$$

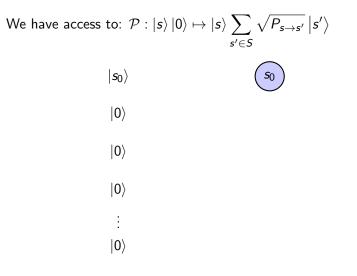
A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 15 / 21

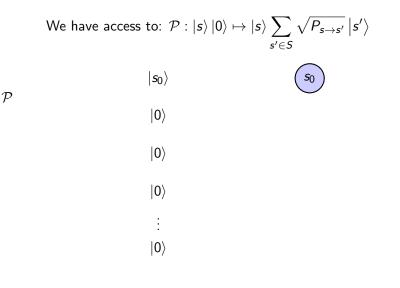
- 2





A.J. Cornelissen (CWI / TU Delft)

June 5th, 2019 15 / 21



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 15 / 21

- 31

< ロ > < 同 > < 回 > < 回 > < 回 > < 回 > < 回 > < 回 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 > < 0 >

We have access to:
$$\mathcal{P} : |s\rangle |0\rangle \mapsto |s\rangle \sum_{s' \in S} \sqrt{P_{s \to s'}} |s'\rangle$$

$$\sum_{s_1 \in S} \sqrt{P_{s_0 \to s_1}} |s_0\rangle |s_1\rangle$$

$$|0\rangle$$

$$|0\rangle$$

$$\vdots$$

$$|0\rangle$$

A.J. Cornelissen (CWI / TU Delft)

 \mathcal{P}

QGE and its application to QRL

June 5th, 2019 15 / 21

- 2

We have access to:
$$\mathcal{P} : |s\rangle |0\rangle \mapsto |s\rangle \sum_{s' \in S} \sqrt{P_{s \to s'}} |s'\rangle$$

$$\sum_{s_1 \in S} \sqrt{P_{s_0 \to s_1}} |s_0\rangle |s_1\rangle$$

$$|0\rangle$$

$$|0\rangle$$

$$|0\rangle$$

$$\vdots$$

$$|0\rangle$$

A.J. Cornelissen (CWI / TU Delft)

 \mathcal{P}

QGE and its application to QRL

June 5th, 2019 15 / 21

- 2

We have access to:
$$\mathcal{P} : |s\rangle |0\rangle \mapsto |s\rangle \sum_{s' \in S} \sqrt{P_{s \to s'}} |s'\rangle$$

 $\mathcal{P} \qquad \sum_{s_1 \in S} \sqrt{P_{s_0 \to s_1}} |s_0\rangle |s_1\rangle$
 $\mathcal{P} \qquad |0\rangle$
 $|0\rangle$
 \vdots
 $|0\rangle$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 15 / 21

3

A D N A B N A B N A B N

$$\begin{array}{c} \text{We have access to: } \mathcal{P} : \left| s \right\rangle \left| 0 \right\rangle \mapsto \left| s \right\rangle \sum_{s' \in S} \sqrt{P_{s \to s'}} \left| s' \right\rangle \\ \mathcal{P} & \sum_{s_1, s_2 \in S} \sqrt{P_{s_0 \to s_1} P_{s_1 \to s_2}} \left| s_0 \right\rangle \left| s_1 \right\rangle \left| s_2 \right\rangle \begin{array}{c} \text{s}_0 \\ \text{a} \end{array} \begin{array}{c} \text{s}_0 \\ \text{b} \end{array}$$

|0⟩ ∶

|0
angle

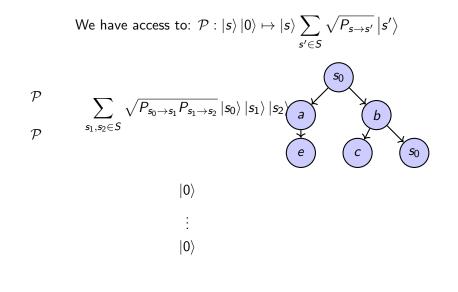
A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 15 / 21

3

A D N A B N A B N A B N



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

- ∢ ⊒ → June 5th, 2019 15 / 21

- 34

▲ 同 ▶ → 三 ▶

We have access to:
$$\mathcal{P} : |s\rangle |0\rangle \mapsto |s\rangle \sum_{s' \in S} \sqrt{P_{s \to s'}} |s'\rangle$$

 \mathcal{P}
 \mathcal{P}
 \mathcal{P}
 $\sum_{s \in S^2} \sqrt{\mathbb{P}(s)} |s\rangle$
 $|0\rangle$
 \vdots
 $|0\rangle$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 15 / 21

- 2

We have access to:
$$\mathcal{P} : |s\rangle |0\rangle \mapsto |s\rangle \sum_{s' \in S} \sqrt{P_{s \to s'}} |s'\rangle$$

 \mathcal{P}
 \mathcal{P}
 \mathcal{P}
 \mathcal{P}
 \mathcal{P}
 $|0\rangle$
 \vdots
 $|0\rangle$

QGE and its application to QRL

June 5th, 2019 15 / 21

- 2

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 15 / 21

- 2

QVE step 1: Setting up the tree

We have access to:
$$\mathcal{P} : |s\rangle |0\rangle \mapsto |s\rangle \sum_{s' \in S} \sqrt{P_{s \to s'}} |s'\rangle$$

 \mathcal{P}
 $\mathcal{P$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

→ ∃ → June 5th, 2019 15 / 21

Image: A mathematical states and a mathem

QVE step 1: Setting up the tree

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 15 / 21

- 2

イロト イポト イヨト イヨト

QVE step 1: Setting up the tree

We have access to:
$$\mathcal{P} : |s\rangle |0\rangle \mapsto |s\rangle \sum_{s' \in S} \sqrt{\mathcal{P}_{s \to s'}} |s'\rangle$$

 \mathcal{P}
 \mathcal{P}

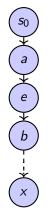
A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 16 / 21

- 20

< □ > < 同 > < 回 > < 回 > < 回 >



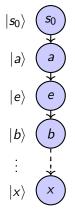
A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 16 / 21

3

< /⊒> <



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

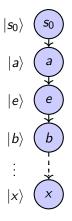
June 5th, 2019

э

16/21

We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$



A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

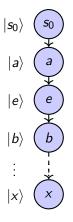
June 5th, 2019 16 / 21

We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$



We have access to:

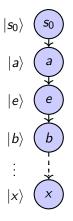
$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$



We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

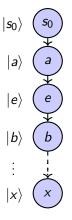
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

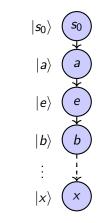
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



 \mathcal{R}

We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

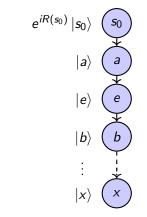
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



 \mathcal{R}

We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

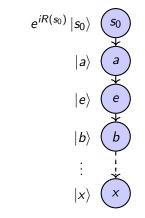
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



 \mathcal{R}

 \mathcal{R}^{γ}

We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

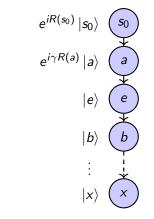
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



A.J. Cornelissen (CWI / TU Delft)

 \mathcal{R}

 \mathcal{R}^{γ}

We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

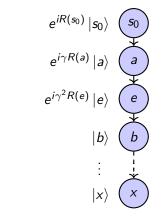
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



 \mathcal{R}

 \mathcal{R}^{γ}

 \mathcal{R}^{γ^2}

We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

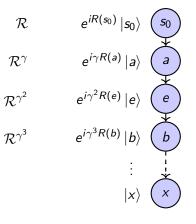
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



 \mathcal{R}^{γ}

 \mathcal{R}^{γ^3}

We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

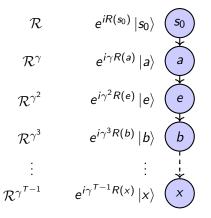
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



We have access to:

$$\mathcal{R}:\ket{s}\mapsto e^{iR(s)}\ket{s}$$

Convert:

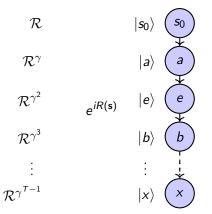
$$\ket{s}\ket{0}\mapsto \sqrt{R(s)}\ket{s}\ket{1}+\ket{\perp}$$

Now multiply by c:

$$\ket{s}\ket{00}\mapsto \sqrt{cR(s)}\ket{s}\ket{11}+\ket{\perp}$$

Convert back:

$$\mathcal{R}^{c}:\ket{s}\mapsto e^{icR(s)}\ket{s}$$



We have access to:

$$\begin{array}{c|c} \mathcal{R} : |s\rangle \mapsto e^{i\mathcal{R}(s)} |s\rangle \\ \hline \mathcal{R} & |s_0\rangle & \overbrace{s_0} \\ \hline \text{Convert:} & |s\rangle |0\rangle \mapsto \sqrt{\mathcal{R}(s)} |s\rangle |1\rangle + |\bot\rangle & \mathcal{R}^{\gamma} & |a\rangle & a \\ \hline \text{Now multiply by } c: & \mathcal{R}^{\gamma^2} & e^{i\mathcal{R}(s)} & |e\rangle & e \\ \hline |s\rangle |00\rangle \mapsto \sqrt{c\mathcal{R}(s)} |s\rangle |11\rangle + |\bot\rangle & \mathcal{R}^{\gamma^3} & |b\rangle & b \\ \hline \text{Convert back:} & \vdots & \vdots & \vdots \\ \mathcal{R}^c : |s\rangle \mapsto e^{ic\mathcal{R}(s)} |s\rangle & \mathcal{R}^{\gamma^{T-1}} & |x\rangle & \swarrow \\ \hline |\mathbf{s}\rangle \mapsto e^{i\mathcal{R}(s)} |\mathbf{s}\rangle & \text{with } \widetilde{\mathcal{O}}(T) \text{ queries to } \mathcal{R} \end{array}$$

э

Image: A matched by the second sec

We have access to:

$$\begin{array}{c|c} \mathcal{R}: |s\rangle \mapsto e^{i\mathcal{R}(s)} |s\rangle \\ \hline \mathcal{R}: |s\rangle \mapsto e^{i\mathcal{R}(s)} |s\rangle \\ \hline \text{Convert:} \\ |s\rangle |0\rangle \mapsto \sqrt{\mathcal{R}(s)} |s\rangle |1\rangle + |\bot\rangle \\ \hline \mathcal{R}^{\gamma} \\ |s\rangle |0\rangle \mapsto \sqrt{\mathcal{R}(s)} |s\rangle |1\rangle + |\bot\rangle \\ \hline \mathcal{R}^{\gamma^2} \\ e^{i\mathcal{R}(s)} \\ \hline e^{i\mathcal{R}(s)} \\ \hline e^{i\mathcal{R}(s)} \\ |s\rangle |0\rangle \mapsto \sqrt{\mathcal{C}\mathcal{R}(s)} |s\rangle |11\rangle + |\bot\rangle \\ \hline \mathcal{R}^{\gamma^3} \\ \hline e^{i\mathcal{R}(s)} \\ \hline e^{i\mathcal{R}(s)}$$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 16 / 21

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 17 / 21

< □ > < 同 > < 回 > < 回 > < 回 >

• We have constructed the following operations with $\widetilde{\mathcal{O}}(\mathcal{T})$ queries to \mathcal{P} and \mathcal{R} :

$$\overline{\mathcal{P}}: |s_0\rangle |0\rangle^{\otimes (\mathcal{T}-1)} \mapsto \sum_{\mathbf{s} \in S^{\mathcal{T}-1}} \sqrt{\mathbb{P}(\mathbf{s})} |\mathbf{s}\rangle \qquad \text{and} \qquad \overline{\mathcal{R}}: |\mathbf{s}\rangle |0\rangle \mapsto \sqrt{R(\mathbf{s})} |\mathbf{s}\rangle |1\rangle + |\bot\rangle$$

3

A B A B A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 B
 A
 A
 A
 A
 A

• We have constructed the following operations with $\widetilde{\mathcal{O}}(\mathcal{T})$ queries to \mathcal{P} and \mathcal{R} :

$$\overline{\mathcal{P}}: |\mathbf{s}_{0}\rangle |\mathbf{0}\rangle^{\otimes (\mathcal{T}-1)} \mapsto \sum_{\mathbf{s} \in \mathcal{S}^{\mathcal{T}-1}} \sqrt{\mathbb{P}(\mathbf{s})} |\mathbf{s}\rangle \qquad \text{and} \qquad \overline{\mathcal{R}}: |\mathbf{s}\rangle |\mathbf{0}\rangle \mapsto \sqrt{R(\mathbf{s})} |\mathbf{s}\rangle |\mathbf{1}\rangle + |\bot\rangle$$

• We want to calculate:

$$V(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right] \approx \sum_{\mathbf{s} \in S^{T-1}} \mathbb{P}(\mathbf{s}) R(\mathbf{s})$$

3

▲ 同 ▶ → 三 ▶

• We have constructed the following operations with $\widetilde{\mathcal{O}}(T)$ queries to \mathcal{P} and \mathcal{R} :

 $\overline{\mathcal{P}}: |\mathbf{s}_0\rangle \, |\mathbf{0}\rangle^{\otimes (\mathcal{T}-1)} \mapsto \sum_{\mathbf{s} \in \mathcal{S}^{\mathcal{T}-1}} \sqrt{\mathbb{P}(\mathbf{s})} \, |\mathbf{s}\rangle \qquad \text{and} \qquad \overline{\mathcal{R}}: |\mathbf{s}\rangle \, |\mathbf{0}\rangle \mapsto \sqrt{R(\mathbf{s})} \, |\mathbf{s}\rangle \, |\mathbf{1}\rangle + |\bot\rangle$

We want to calculate:

$$V(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right] \approx \sum_{\mathbf{s} \in S^{T-1}} \mathbb{P}(\mathbf{s}) R(\mathbf{s})$$

• Composing
$$\overline{\mathcal{P}}$$
 and $\overline{\mathcal{R}}$ yields:

 $\ket{s_0}\ket{0}^{\otimes (T-1)}\ket{0}$

▲□▶ ▲圖▶ ▲ 臣▶ ▲ 臣▶ 三臣 - のへの

• We have constructed the following operations with $\widetilde{\mathcal{O}}(T)$ queries to \mathcal{P} and \mathcal{R} :

$$\overline{\mathcal{P}}: \left| \boldsymbol{s}_{0} \right\rangle \left| \boldsymbol{0} \right\rangle^{\otimes (\mathcal{T}-1)} \mapsto \sum_{\boldsymbol{s} \in \mathcal{S}^{\mathcal{T}-1}} \sqrt{\mathbb{P}(\boldsymbol{s})} \left| \boldsymbol{s} \right\rangle \qquad \text{and} \qquad \overline{\mathcal{R}}: \left| \boldsymbol{s} \right\rangle \left| \boldsymbol{0} \right\rangle \mapsto \sqrt{\mathcal{R}(\boldsymbol{s})} \left| \boldsymbol{s} \right\rangle \left| \boldsymbol{1} \right\rangle + \left| \bot \right\rangle$$

• We want to calculate:

$$V(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right] \approx \sum_{\mathbf{s} \in S^{T-1}} \mathbb{P}(\mathbf{s}) R(\mathbf{s})$$

• Composing $\overline{\mathcal{P}}$ and $\overline{\mathcal{R}}$ yields:

$$\ket{s_{0}}\ket{0}^{\otimes(\mathcal{T}-1)}\ket{0} \stackrel{\overline{\mathcal{P}}}{\mapsto} \sum_{\mathbf{s}\in\mathcal{S}^{\mathcal{T}-1}} \sqrt{\mathbb{P}(\mathbf{s})}\ket{\mathbf{s}}\ket{0}$$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 17 / 21

• We have constructed the following operations with $\widetilde{\mathcal{O}}(\mathcal{T})$ queries to \mathcal{P} and \mathcal{R} :

$$\overline{\mathcal{P}}: |\mathbf{s}_{0}\rangle |\mathbf{0}\rangle^{\otimes (\mathcal{T}-1)} \mapsto \sum_{\mathbf{s} \in S^{\mathcal{T}-1}} \sqrt{\mathbb{P}(\mathbf{s})} |\mathbf{s}\rangle \qquad \text{and} \qquad \overline{\mathcal{R}}: |\mathbf{s}\rangle |\mathbf{0}\rangle \mapsto \sqrt{R(\mathbf{s})} |\mathbf{s}\rangle |\mathbf{1}\rangle + |\bot\rangle$$

• We want to calculate:

$$V(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right] \approx \sum_{\mathbf{s} \in S^{T-1}} \mathbb{P}(\mathbf{s}) R(\mathbf{s})$$

• Composing
$$\overline{\mathcal{P}}$$
 and $\overline{\mathcal{R}}$ yields:

$$|s_{0}\rangle|0\rangle^{\otimes(\tau-1)}|0\rangle \stackrel{\overline{\mathcal{P}}}{\mapsto} \sum_{\mathbf{s}\in\mathcal{S}^{\tau-1}} \sqrt{\mathbb{P}(\mathbf{s})} |\mathbf{s}\rangle|0\rangle \stackrel{\overline{\mathcal{R}}}{\mapsto} \sum_{\mathbf{s}\in\mathcal{S}^{\tau-1}} \sqrt{\mathbb{P}(\mathbf{s})R(\mathbf{s})} |\mathbf{s}\rangle|1\rangle + |\bot\rangle$$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

3

▲ 同 ▶ → 三 ▶

• We have constructed the following operations with $\widetilde{\mathcal{O}}(\mathcal{T})$ queries to \mathcal{P} and \mathcal{R} :

$$\overline{\mathcal{P}}: |\mathbf{s}_{0}\rangle |\mathbf{0}\rangle^{\otimes (\mathcal{T}-1)} \mapsto \sum_{\mathbf{s} \in S^{\mathcal{T}-1}} \sqrt{\mathbb{P}(\mathbf{s})} |\mathbf{s}\rangle \qquad \text{and} \qquad \overline{\mathcal{R}}: |\mathbf{s}\rangle |\mathbf{0}\rangle \mapsto \sqrt{R(\mathbf{s})} |\mathbf{s}\rangle |\mathbf{1}\rangle + |\bot\rangle$$

We want to calculate:

$$V(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right] \approx \sum_{\mathbf{s} \in S^{T-1}} \mathbb{P}(\mathbf{s}) R(\mathbf{s})$$

• Composing
$$\overline{\mathcal{P}}$$
 and $\overline{\mathcal{R}}$ yields:

$$|s_{0}\rangle |0\rangle^{\otimes (T-1)} |0\rangle \stackrel{\overline{\mathcal{P}}}{\mapsto} \sum_{\mathbf{s} \in S^{T-1}} \sqrt{\mathbb{P}(\mathbf{s})} |\mathbf{s}\rangle |0\rangle \stackrel{\overline{\mathcal{R}}}{\mapsto} \sum_{\mathbf{s} \in S^{T-1}} \sqrt{\mathbb{P}(\mathbf{s})R(\mathbf{s})} |\mathbf{s}\rangle |1\rangle + |\bot\rangle$$

• One can obtain the value function with amplitude estimation up to precision ε with

$$\widetilde{\mathcal{O}}\left(\frac{T|R|_{\max}}{\varepsilon(1-\gamma)}\right) = \widetilde{\mathcal{O}}\left(\frac{|R|_{\max}}{\varepsilon(1-\gamma)^2}\right)$$

queries to ${\mathcal P}$ and ${\mathcal R},$ quadratically faster than classical algorithms.

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

イロト イポト イヨト イヨ

• We have constructed the following operations with $\widetilde{\mathcal{O}}(T)$ queries to \mathcal{P} and \mathcal{R} :

$$\overline{\mathcal{P}}: |\mathbf{s}_{0}\rangle |\mathbf{0}\rangle^{\otimes (\mathcal{T}-1)} \mapsto \sum_{\mathbf{s} \in S^{\mathcal{T}-1}} \sqrt{\mathbb{P}(\mathbf{s})} |\mathbf{s}\rangle \qquad \text{and} \qquad \overline{\mathcal{R}}: |\mathbf{s}\rangle |\mathbf{0}\rangle \mapsto \sqrt{R(\mathbf{s})} |\mathbf{s}\rangle |\mathbf{1}\rangle + |\bot\rangle$$

We want to calculate:

$$V(s_0) = \mathbb{E}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t)\right] \approx \sum_{\mathbf{s} \in S^{T-1}} \mathbb{P}(\mathbf{s}) R(\mathbf{s})$$

• Composing
$$\overline{\mathcal{P}}$$
 and $\overline{\mathcal{R}}$ yields:

$$|s_{0}\rangle |0\rangle^{\otimes (T-1)} |0\rangle \stackrel{\overline{\mathcal{P}}}{\mapsto} \sum_{\mathbf{s} \in S^{T-1}} \sqrt{\mathbb{P}(\mathbf{s})} |\mathbf{s}\rangle |0\rangle \stackrel{\overline{\mathcal{R}}}{\mapsto} \sum_{\mathbf{s} \in S^{T-1}} \sqrt{\mathbb{P}(\mathbf{s})R(\mathbf{s})} |\mathbf{s}\rangle |1\rangle + |\bot\rangle$$

• One can obtain the value function with amplitude estimation up to precision ε with

$$\widetilde{\mathcal{O}}\left(\frac{T|R|_{\max}}{\varepsilon(1-\gamma)}\right) = \widetilde{\mathcal{O}}\left(\frac{|R|_{\max}}{\varepsilon(1-\gamma)^2}\right)$$

queries to \mathcal{P} and \mathcal{R} , quadratically faster than classical algorithms.

• This is essentially optimal for $\varepsilon \downarrow 0$, $|R|_{\max} \rightarrow \infty$, $\gamma \uparrow 1$.

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 18 / 21

- 2

• Let S be a set of states, A be a set of actions, and s₀ the initial state.

- 20

< □ > < 同 > < 回 > < 回 > < 回 >

• Let S be a set of states, A be a set of actions, and s₀ the initial state.









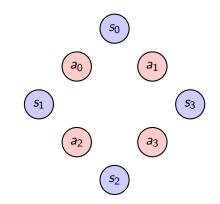
A D N A B N A B N A B N

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 18 / 21

• Let S be a set of states, A be a set of actions, and s₀ the initial state.



A D N A B N A B N A B N

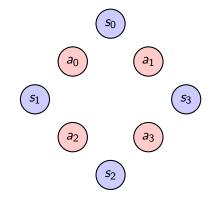
A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 18 / 21

- Let S be a set of states, A be a set of actions, and s₀ the initial state.
- Let $\pi_{s \to a}$ be a policy.

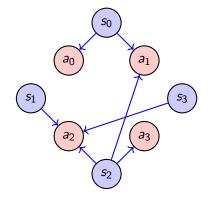
$$egin{aligned} \mathsf{\Pi} : \ket{s}\ket{\mathsf{0}} &\mapsto \ket{s}\sum_{a\in A}\sqrt{\pi_{s o a}}\ket{a} \end{aligned}$$



< □ > < 同 > < 回 > < 回 > < 回 >

- Let S be a set of states, A be a set of actions, and s₀ the initial state.
- Let $\pi_{s \to a}$ be a policy.

$$egin{aligned} \mathsf{\Pi} : \ket{s}\ket{\mathsf{0}} &\mapsto \ket{s}\sum_{m{a}\in A}\sqrt{\pi_{m{s}
ightarrow m{a}}}\ket{m{a}} \end{aligned}$$



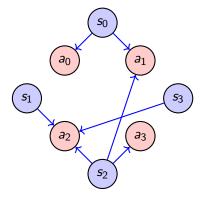
< □ > < 同 > < 回 > < 回 > < 回 >

- Let S be a set of states, A be a set of actions, and s₀ the initial state.
- Let $\pi_{s \to a}$ be a policy.

$$\Pi:\ket{s}\ket{0}\mapsto \ket{s}\sum_{a\in A}\sqrt{\pi_{s
ightarrow a}}\ket{a}$$

• Let $P_{s,a \rightarrow s'}$ be the transition function.

$$\mathcal{P}: \ket{s}\ket{a}\ket{0}\mapsto \ket{s}\ket{a}\sum_{s'\in S}\sqrt{P_{s,a
ightarrow s'}}\ket{s'}$$



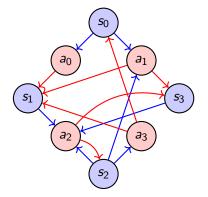
< □ > < 同 > < 回 > < 回 > < 回 >

- Let S be a set of states, A be a set of actions, and s₀ the initial state.
- Let $\pi_{s \to a}$ be a policy.

$$\Pi:\ket{s}\ket{0}\mapsto \ket{s}\sum_{a\in A}\sqrt{\pi_{s
ightarrow a}}\ket{a}$$

• Let $P_{s,a \rightarrow s'}$ be the transition function.

$$\mathcal{P}: \ket{s}\ket{a}\ket{0}\mapsto \ket{s}\ket{a}\sum_{s'\in S}\sqrt{P_{s,a
ightarrow s'}}\ket{s'}$$



A D N A B N A B N A B N

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 18 / 21

- Let S be a set of states, A be a set of actions, and s₀ the initial state.
- Let $\pi_{s \to a}$ be a policy.

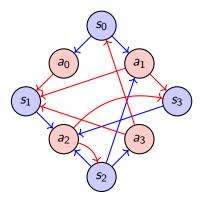
$$\Pi:\ket{s}\ket{0}\mapsto \ket{s}\sum_{a\in A}\sqrt{\pi_{s
ightarrow a}}\ket{a}$$

• Let $P_{s,a \rightarrow s'}$ be the transition function.

$$\mathcal{P}: \ket{s}\ket{a}\ket{0}\mapsto \ket{s}\ket{a}\sum_{s'\in \mathsf{S}}\sqrt{P_{s,a
ightarrow s'}}\ket{s'}$$

• Let R(s, a) be the reward function.

 $\mathcal{R}: \ket{s}\ket{a}\mapsto e^{iR(s,a)}\ket{s}\ket{a}$



A D N A B N A B N A B N

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 18 / 21

- 20

- Let S be a set of states, A be a set of actions, and s₀ the initial state.
- Let $\pi_{s \to a}$ be a policy.

$$\Pi:\ket{s}\ket{0}\mapsto \ket{s}\sum_{a\in A}\sqrt{\pi_{s
ightarrow a}}\ket{a}$$

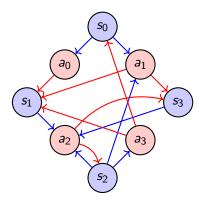
• Let $P_{s,a \rightarrow s'}$ be the transition function.

$$\mathcal{P}: \ket{s}\ket{a}\ket{0}\mapsto \ket{s}\ket{a}\sum_{s'\in S}\sqrt{P_{s,a
ightarrow s'}}\ket{s'}$$

• Let R(s, a) be the reward function.

$$\mathcal{R}:\ket{s}\ket{a}\mapsto e^{iR(s,a)}\ket{s}\ket{a}$$

• Let $0 < \gamma < 1$.



A D N A B N A B N A B N

- Let S be a set of states, A be a set of actions, and s₀ the initial state.
- Let $\pi_{s \to a}$ be a policy.

$$\Pi:\ket{s}\ket{0}\mapsto \ket{s}\sum_{a\in A}\sqrt{\pi_{s
ightarrow a}}\ket{a}$$

• Let $P_{s,a \rightarrow s'}$ be the transition function.

$$\mathcal{P}: \ket{s}\ket{a}\ket{0}\mapsto \ket{s}\ket{a}\sum_{s'\in \mathsf{S}}\sqrt{P_{s,a
ightarrow s'}}\ket{s'}$$

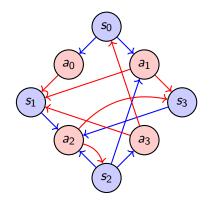
• Let R(s, a) be the reward function.

$$\mathcal{R}:\ket{s}\ket{a}\mapsto \mathrm{e}^{\mathrm{i}R(s,a)}\ket{s}\ket{a}$$

- Let $0 < \gamma < 1$.
- Goal: find the policy π such that:

$$V(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t R(S_t, A_t)\right]$$

is maximized.



A D N A B N A B N A B N

- 34

- Let S be a set of states, A be a set of actions, and s₀ the initial state.
- Let $\pi_{s \to a}$ be a policy.

$$\Pi:\ket{s}\ket{0}\mapsto \ket{s}\sum_{a\in A}\sqrt{\pi_{s
ightarrow a}}\ket{a}$$

• Let $P_{s,a \rightarrow s'}$ be the transition function.

$$\mathcal{P}: \ket{s}\ket{a}\ket{0}\mapsto \ket{s}\ket{a}\sum_{s'\in S}\sqrt{P_{s,a
ightarrow s'}}\ket{s'}$$

• Let R(s, a) be the reward function.

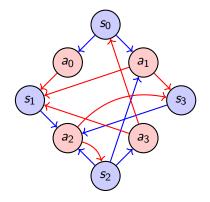
$$\mathcal{R}:\ket{s}\ket{a}\mapsto \mathrm{e}^{\mathrm{i}R(s,a)}\ket{s}\ket{a}$$

- Let $0 < \gamma < 1$.
- Goal: find the policy π such that:

$$V(\pi) = \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^{t} R(S_{t}, A_{t})\right]$$

is maximized.

Quantum policy optimization



A D N A B N A B N A B N

- 31

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 19 / 21

- 2

・ロト ・四ト ・ヨト ・ヨト

• Classical idea: define some parameter space \mathbb{R}^d , and embed it into the space of policies.

$$\kappa: \theta \mapsto \pi^{(\theta)}$$

• *Classical* idea: define some parameter space \mathbb{R}^d , and embed it into the space of policies.

$$\kappa: \theta \mapsto \pi^{(\theta)}$$

• Now, we define the following function $f : \mathbb{R}^d \to \mathbb{R}$:

$$f: \theta \mapsto V(\kappa(\theta))$$

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 19 / 21

• *Classical* idea: define some parameter space \mathbb{R}^d , and embed it into the space of policies.

$$\kappa: \theta \mapsto \pi^{(\theta)}$$

• Now, we define the following function $f : \mathbb{R}^d \to \mathbb{R}$:

$$f: \theta \mapsto V(\kappa(\theta))$$

• We wish to find the maximum of this function \Rightarrow gradient ascent!

• *Classical* idea: define some parameter space \mathbb{R}^d , and embed it into the space of policies.

$$\kappa: \theta \mapsto \pi^{(\theta)}$$

• Now, we define the following function $f : \mathbb{R}^d \to \mathbb{R}$:

$$f: \theta \mapsto V(\kappa(\theta))$$

- We wish to find the maximum of this function \Rightarrow gradient ascent!
- Construction of a phase oracle of *f* can be done by modifying the quantum value estimation algorithm.

• *Classical* idea: define some parameter space \mathbb{R}^d , and embed it into the space of policies.

$$\kappa: \theta \mapsto \pi^{(\theta)}$$

• Now, we define the following function $f : \mathbb{R}^d \to \mathbb{R}$:

$$f: \theta \mapsto V(\kappa(\theta))$$

- We wish to find the maximum of this function \Rightarrow gradient ascent!
- Construction of a phase oracle of *f* can be done by modifying the quantum value estimation algorithm.
- How well does it work?

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 20 / 21

< □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > < □ > <

• Performance depends on the smoothness of the objective function.

$$f(\theta) = V(\kappa(\theta))$$

3

• Performance depends on the smoothness of the objective function.

$$f(\theta) = V(\kappa(\theta))$$

• Smoothness of κ can be freely chosen.

$$f(\theta) = V(\kappa(\theta))$$

- Smoothness of κ can be freely chosen.
- Smoothness of V is in general of Gevrey type with $\sigma = 1$.

$$f(\theta) = V(\kappa(\theta))$$

- Smoothness of κ can be freely chosen.
- Smoothness of V is in general of Gevrey type with $\sigma = 1$.
- Smoothness of compositions of two Gevrey functions is in general Gevrey with $\sigma=1.$

$$f(\theta) = V(\kappa(\theta))$$

- Smoothness of κ can be freely chosen.
- Smoothness of V is in general of Gevrey type with $\sigma = 1$.
- Smoothness of compositions of two Gevrey functions is in general Gevrey with $\sigma=1.$
- What happens when we use polynomial approximations to V?

$$f(\theta) = V(\kappa(\theta))$$

- Smoothness of κ can be freely chosen.
- Smoothness of V is in general of Gevrey type with $\sigma = 1$.
- Smoothness of compositions of two Gevrey functions is in general Gevrey with $\sigma=1.$
- What happens when we use polynomial approximations to V?
- What happens if we restrict to policies that are sufficiently non-deterministic?

$$f(\theta) = V(\kappa(\theta))$$

- Smoothness of κ can be freely chosen.
- Smoothness of V is in general of Gevrey type with $\sigma = 1$.
- Smoothness of compositions of two Gevrey functions is in general Gevrey with $\sigma=1.$
- What happens when we use polynomial approximations to V?
- What happens if we restrict to policies that are sufficiently non-deterministic?
- Even classical gradient ascent with quantum value evaluation as subroutine provides speed-up!

A.J. Cornelissen (CWI / TU Delft) QGE and its a

QGE and its application to QRL

June 5th, 2019 21 / 21

イロト イ部ト イヨト イヨト 二日

Quantum gradient estimation:

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 21 / 21

< □ > < 同 > < 回 > < 回 > < 回 >

э

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

イロト イポト イヨト イヨト

- 34

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

• Speed-up for polynomials and $\sigma < 1$.

< □ > < 同 > < 回 > < 回 > < 回 >

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

- Speed-up for polynomials and $\sigma < 1$.
- Open problem: Close the gap for $\sigma = 1$.

- 31

< □ > < 同 > < 回 > < 回 > < 回 >

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

- Speed-up for polynomials and $\sigma < 1$.
- Open problem: Close the gap for $\sigma = 1$.
- Quantum value estimation:

< □ > < 同 > < 三 > < 三 >

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

- Speed-up for polynomials and $\sigma < 1$.
- Open problem: Close the gap for $\sigma = 1$.
- Quantum value estimation:
 - Optimal algorithm with analog computation.

イロト 不得下 イヨト イヨト 二日

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

- Speed-up for polynomials and $\sigma < 1$.
- Open problem: Close the gap for $\sigma=1.$
- Quantum value estimation:
 - Optimal algorithm with analog computation.
- Quantum policy optimization:

イロト 不得下 イヨト イヨト 二日

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

- Speed-up for polynomials and $\sigma < 1$.
- Open problem: Close the gap for $\sigma = 1$.
- Quantum value estimation:
 - Optimal algorithm with analog computation.
- Quantum policy optimization:
 - Speed-up using QVE.

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

- Speed-up for polynomials and $\sigma < 1$.
- Open problem: Close the gap for $\sigma = 1$.
- Quantum value estimation:
 - Optimal algorithm with analog computation.
- Quantum policy optimization:
 - Speed-up using QVE.
 - Open problem: approximating V for more smoothness?

▲□▶ ▲□▶ ▲□▶ ▲□▶ □ ののの

Quantum gradient estimation:

Smoothness	Polynomial		Gevrey	
condition	degree k	$\sigma \in [0, \frac{1}{2})$	$\sigma = \frac{1}{2}$	$\sigma \in (\frac{1}{2}, 1]$
Best known algorithm	$\widetilde{\mathcal{O}}(k)$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\frac{1}{2}})$	$\widetilde{\mathcal{O}}(d^{\sigma})$
Best known lower bound	$\Omega(1)$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$	$\Omega(d^{\frac{1}{2}})$

From ℓ^{∞} to ℓ^{p} approximations: multiply upper and lower bounds by $\Theta(d^{\frac{1}{p}})$.

- Speed-up for polynomials and $\sigma < 1$.
- Open problem: Close the gap for $\sigma = 1$.
- Quantum value estimation:
 - Optimal algorithm with analog computation.
- Quantum policy optimization:
 - Speed-up using QVE.
 - Open problem: approximating V for more smoothness?

Thanks for your attention! arjan@cwi.nl

A.J. Cornelissen (CWI / TU Delft)

QGE and its application to QRL

June 5th, 2019 21 / 21