

COLLÈGE
DE FRANCE
— 1530 —

Algorithmes quantiques

Simulation hamiltonienne

26-05-2021

Frédéric Magniez

Professeur invité sur la chaire Informatique et sciences numériques

En partenariat avec Inria

Année académique 2020-2021

frederic.magniez@college-de-france.fr

Partie 3 - Algorithmique avancée

- Apprentissage automatique
- Limites du calcul quantique
- Usage décentralisé de type Internet

26 mai 2021



Cours : Simulation hamiltonienne, résolution ultra-rapide de systèmes linéaires, et applications

Séminaire : Quantum Machine Learning, Iordanis KERENIDIS, *CNRS, Paris*

02 juin 2021

Cours : Limites du calcul quantique : liens entre complexité classique et quantique

Séminaire : Suprématie quantique : où en sommes-nous aujourd'hui ?

André CHAILLOUX, *Inria, Paris*

09 juin 2021

Cours : Conclusion et ouverture vers le calcul distribué quantique

Séminaire : Quantum Computing as a Service:

Secure and Verifiable Multi-Tenant Quantum Data Centre

Elham KASHEFI, *CNRS, Paris et University of Edinburgh*



Simulation hamiltonienne

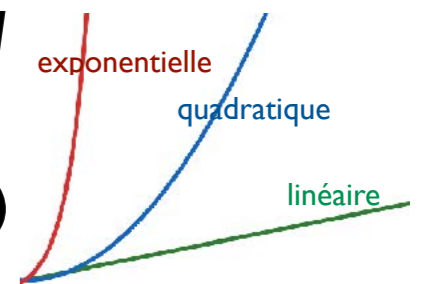
Question de Richard Feynman

- Exposé invité où il questionne :
***Can quantum systems
 be probabilistically simulated
 by a classical computer?***



Discussion

- [...] Quantum mechanics can't seem to be imitable by a local classical computer.
 → Explosion des ressources nécessaires (temps, mémoire, ...)



- *Can you do it with a new kind of computer - a quantum computer? [...] It's not a Turing machine, but a machine of a different kind. [...] I'm not sure that it's sufficient.*
 → Remise en cause de la version quantitative/algorithmique
 de la thèse de Church-Turing

Les progrès technologiques permettent d'augmenter "uniquement" vitesse et quantité de ressources (mémoire, processeurs, ...)

Problème

- Etant donné un système physique régi par un Hamiltonien H (ie, $H^*=H$)
Taille exponentielle (taille $2^n \times 2^n$ si n qubits)
Borné, Description compacte (qui permet de calculer H_{uv} rapidement)
- Simuler l'évolution du système après un temps t : $|\psi(t)\rangle = e^{-itH}|\psi(0)\rangle$
Question de Feynman en 1981
- Puis observer une de ses caractéristiques

Equation de Schrödinger

$$i\hbar \frac{d|\psi(t)\rangle}{dt} = H|\psi(t)\rangle \quad \text{avec la convention } \hbar = 1$$

admet pour solution $|\psi(t)\rangle = e^{-itH}|\psi(0)\rangle$ (e^{-itH} est unitaire)

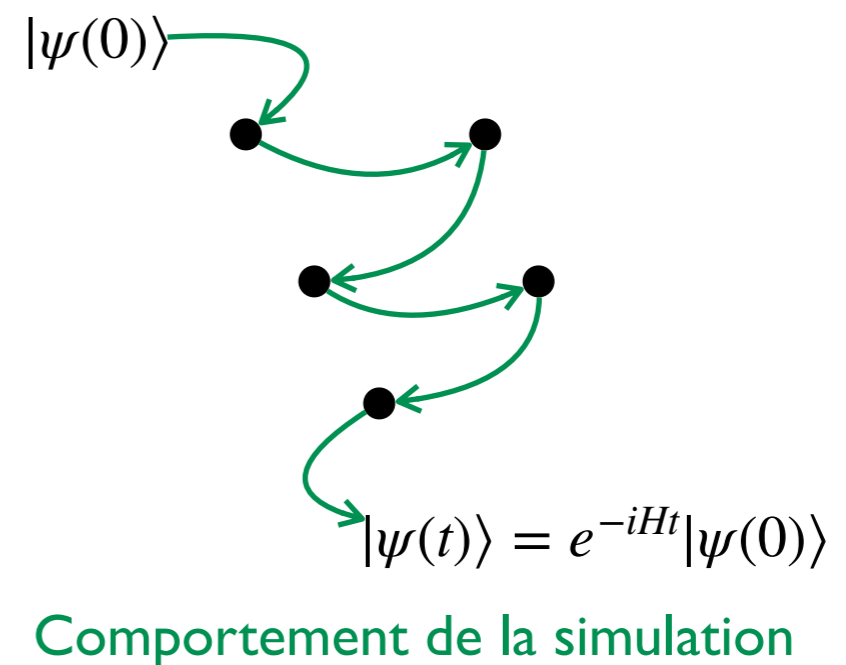
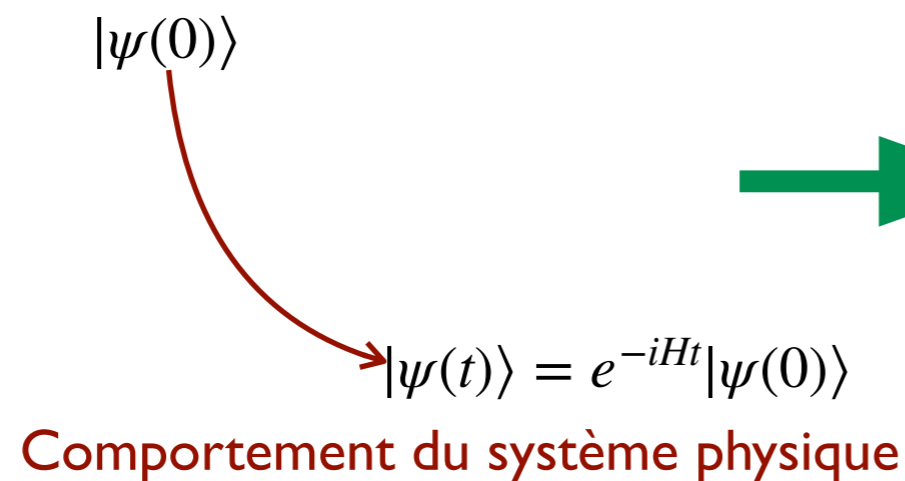
Enjeux

- Simuler la physique permet de mieux la comprendre, de faire des découvertes, d'économiser en ressources
- Applications : chimie, matériaux, physique des particules, ...
- Autres possibilités : préparer des états, mesurer l'énergie...

Ordinateur quantique

- Pourquoi ne pas programmer directement un ordinateur quantique qui suit l'hamiltonien à simuler ?
- Programmation : Comment construire ce circuit qui simule cet hamiltonien à l'aide de portes élémentaires ?

Simulation non directe



Variante quantique

- Input
 - t, ε et description de taille ℓ d'un hamiltonien H
(qui permet de calculer H_{uv} en temps $\text{poly}(\ell)$)
 - Etat quantique $|\psi(0)\rangle$
- Output : Etat quantique $|\psi(t)\rangle = e^{-itH}|\psi(0)\rangle$ à ε près

Variante classique

- Input : t, ε et description de taille ℓ d'un hamiltonien H
- Output : Description d'un circuit qui approche e^{-itH} à ε près

Réponses

- [Seth Lloyd 1996] : Complexité en t^2 et $\text{poly}(\ell/\varepsilon)$
- Depuis de nombreuses optimisations : dépendance en $t, \text{poly}(\ell)$ et $\log(1/\varepsilon)$
- Etudes pour des structures concrètes
- Extensions possibles à des hamiltoniens dépendent du temps (par exemple via le calcul adiabatique)

Difficulté

- Même si H possède une représentation compacte explicite qui permet de calculer H_{uv} facilement pour chaque (u,v)
- En général $U = e^{-iH}$ n'a plus de telle représentation

$$e^{-iH} = \text{Id} - iH + \frac{1}{2}H^2 + \dots + \frac{(-i)^j}{j!}H^j + \dots$$

Cas simple I

- Si H agit sur uniquement k qubits ($\ell = 4^k$), alors e^{-iH} aussi !

$$H = H_k \otimes \text{Id}_{n-k} \mapsto e^{-iH} = e^{-iH_k} \otimes \text{Id}_{n-k}$$

(la phase e^{-i} a été supprimée)

- Un circuit de taille $O(k4^k)$ (cours 3) peut réaliser e^{-iH_k}
- Pour réaliser e^{-itH}

Il suffit en suite de composer t fois le circuit de $e^{-iH_k} \otimes \text{Id}_{n-k}$

Ou de directement construire celui pour $e^{-itH_k} \otimes \text{Id}_{n-k}$

Localité

- H sur n qubits est k -local s'il se décompose en une somme

$$H = \sum_{j=1}^m H_j \text{ où chaque } H_j \text{ agit sur au plus } k \text{ qubits}$$

Dans le cas général, $m \leq \binom{n}{k} \leq n^k$

- Localité au sens géométrique : $O(n)$ termes
- Opérateurs bornés : $\|H_j\| \leq 1$

Extension

- $H = \sum_{j=1}^m H_j$ sur n qubits avec $\|H_j\| \leq 1$
- Implémentation : $e^{-i\delta H_j}$ admet un circuit de taille $\text{poly}(n)$
- Somme bornée : $m = \text{poly}(n)$

Question

- Comment simuler l'exponentiel d'une somme de matrices ?

Cas commutatif

- Si $H = \sum_{j=1}^m H_j$ avec $H_i H_j = H_j H_i$ alors $e^{-itH} = (e^{-iH_1} \times e^{-iH_2} \times \dots \times e^{-iH_m})^t$
- Donc circuit pour e^{-itH} de taille
 m fois la taille du circuit pour chaque e^{-itH_j}
ou encore mt fois la taille du circuit pour chaque e^{-iH_j}

Cas non commutatif - Plusieurs approches



- Formules produit de Lie-Suzuki-Trotter [Lloyd 1996]
- Marches quantiques [Berry, Childs 2012]
- Combinaison linéaire de matrices unitaires [Childs, Wiebe 2012]
- Échantillonnage probabiliste d'état quantique [Lloyd, Mohseni, Rebentrost 2014]
- Combinaison linéaire de marches quantiques [Berry, Childs, Kothari 2015]
- Quantum signal processing [Low, Chuang 2016]
- ...

Idée

- $e^{A+B} = \lim_{r \rightarrow \infty} (e^{A/r} e^{B/r})^r$
- $e^{A+B} = e^A e^B + O(\|A\| \times \|B\|)$

Utilisation

- $e^{-i(t/r)H} = e^{-i(t/r)H_1} \times \dots \times e^{-i(t/r)H_m} + O(m^2(t/r)^2)$
- $e^{-itH} = (e^{-i(t/r)H_1} \times \dots \times e^{-i(t/r)H_m})^r + O(m^2 t^2 / r)$
- Circuit de $O(mr) = O(m^3 t^2 / \varepsilon)$ blocs pour une approximation ε
Chaque bloc correspond au circuit de $e^{-i(\varepsilon/(m^2 t))H_j}$

Amélioration

- $e^{-itH} = (e^{-i(t/2r)H_1} \dots e^{-i(t/2r)H_m} e^{-i(t/2r)H_m} \dots e^{-i(t/2r)H_1})^r + O(m^3 t^3 / r^2)$
- Circuit de $O(mr) = O(m^{5/2} t^{3/2} / \varepsilon^{1/2})$ blocs pour une approximation ε
Chaque bloc correspond au circuit de $e^{-i(\varepsilon^{1/2}/(m^{3/2} t^{1/2}))H_j}$

Simulation (quasi)-optimale (autres méthodes)

- Circuit en $\tilde{O}(m^2 t \log(1/\varepsilon))$ blocs
- Extension aux matrices creuses : au plus s entrées non nulles par ligne
- Extension à des décompositions et encodages variés de H

Résultats optimaux

- Optimal Hamiltonian Simulation by Quantum Signal Processing [Low, Chuang 2016]

Structures explicites

- Quantum algorithm for simulating real time evolution of lattice Hamiltonians [Haah, Hastings, Kothari, Low 2018]

En pratique

- Toward the first quantum simulation with quantum speedup [Childs, Maslov, Nam, Ross, Su 2018]

La méthode la plus simple mais non optimale semble se comporter aussi bien “en pratique” que la méthode optimale...

Systemes linéaires

Systeme linéaire

- Input : Matrice A de taille $N \times N$ (à coefficients réels ou complexes)
Vecteur b de taille N
- Output : Vecteur x de taille N tel que $Ax = b$

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \dots + a_{1N}x_N = b_1 \\ a_{21}x_1 + a_{22}x_2 + \dots + a_{2N}x_N = b_2 \\ \vdots \\ a_{N1}x_1 + a_{N2}x_2 + \dots + a_{NN}x_N = b_N \end{cases}$$

- Hypothèse 1 : A inversible
- Hypothèse 2 : A hermitienne i.e. $A^* = A$

Sinon remplacer le système $Ax = b$


par le système
$$\begin{pmatrix} 0 & A \\ A^* & 0 \end{pmatrix} y = \begin{pmatrix} b \\ 0 \end{pmatrix}$$

dont la solution est de la forme $y = \begin{pmatrix} 0 \\ x \end{pmatrix}$ telle que $Ax = b$

Solutions exactes

- Pivot de Gauss : $O(N^3)$ opérations arithmétiques
- Multiplication rapide de matrice : $O(N^{2.373})$ opérations arithmétiques
- Matrice creuse : au plus s éléments non nuls par ligne $\rightarrow O(sN^2)$

Par décomposition spectrale (A hermitienne)

- Diagonalisation de A dans une base orthonormée v_j de v.p λ_j
- Alors la solution x a pour coordonnées dans cette base $\alpha_j = \frac{1}{\lambda_j} (v_j \cdot b)$

- Variante approchée et itérative

Fournit une solution \hat{x} telle que $\|x - \hat{x}\| \leq \varepsilon \|x\|$

en temps $O(Ns\kappa \log(1/\varepsilon))$

avec $\kappa = \|A\| \|A^{-1}\| = |\lambda_{\max}(A)| / |\lambda_{\min}(A)|$

et s le nb maximum d'éléments non nuls par ligne

Référence : An Introduction to the Conjugate Gradient Method Without the Agonizing Pain, [Jonathan R. Shewchuk 1994]

- Domaine très actif en mathématiques et informatique !

Encodage quantique

- Vecteur z de taille $N = 2^n \rightarrow |z\rangle = \frac{1}{\|z\|} \sum_i z_j |j\rangle$ de n qubits

$$\text{où } \|z\| = \sqrt{\sum_j |z_j|^2}$$

Système linéaire quantique

- Input
 - Une description compacte d'une matrice A de taille $2^n \times 2^n$
 - Un algorithme quantique C_b qui produit $|b\rangle = \frac{1}{\|b\|} \sum_j b_j |j\rangle$ sur n qubits
- Output : Un algorithme quantique C_x qui produit $|x\rangle = \frac{1}{\|x\|} \sum_j x_j |j\rangle$ sur n qubits tel que $Ax = b$

Hypothèses et paramètres

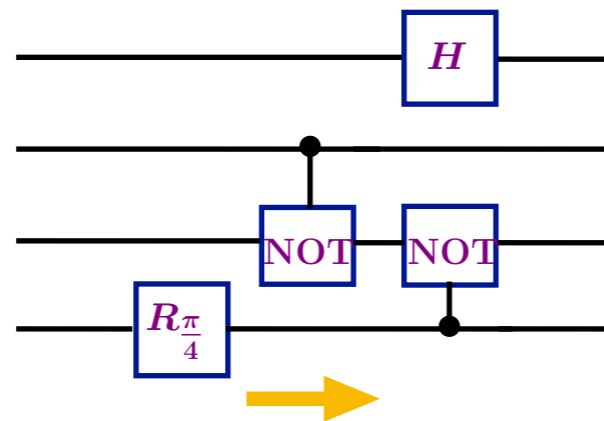
- Hypothèses : A inversible, A hermitienne i.e. $A^* = A$,
Valeurs propres λ de A satisfont $1/\kappa \leq |\lambda| \leq 1$
- Paramètres : Au plus s éléments non nuls par ligne de A
 $\kappa = \|A\| \|A^{-1}\| = |\lambda_{\max}(A)| / |\lambda_{\min}(A)|$

A unitaire

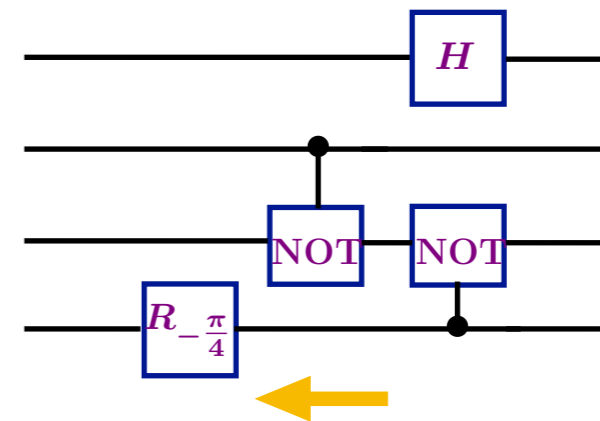
- Alors effectivement les solutions $Ax = b$ correspondent aux solutions $A|x\rangle = |b\rangle$ (pas de problème de renormalisation car $\|x\| = \|b\|$)

Solution directe

- Etant donné un circuit pour A construire un circuit pour A^{-1}
Il suffit de parcourir le circuit en sens inverse en remplaçant chaque porte par leur inverse



A : Sens de composition



A^{-1} : Sens de composition

$$A = \text{diag}(\lambda_1, \lambda_2, \dots, \lambda_N)$$

- Réaliser $|b\rangle = \frac{1}{\|b\|} \sum_j b_j |j\rangle \mapsto |x\rangle = |A^{-1}b\rangle = \frac{1}{\|x\|} \sum_j \frac{b_j}{\lambda_j} |j\rangle$
avec $\|x\|^2 = \sum_j |x_j|^2 = \sum_j |b_j/\lambda_j|^2 \neq \|b\|^2$
- Rappel : $1/\kappa \leq |\lambda| \leq 1$

Réaliser l'inversion

- On veut implémenter $|j\rangle = \frac{1}{\lambda_j} |j\rangle$ qui n'est pas unitaire...
mais $|j\rangle|0\rangle \mapsto |j\rangle \left(\frac{1}{\kappa\lambda_j} |0\rangle + \sqrt{1 - 1/(\kappa\lambda_j)^2} |1\rangle \right)$ si ! ($1/\kappa \leq |\lambda_i| \leq 1$)
Simple rotation calculée à partir de $|j\rangle$
- Il existe donc un circuit de taille $O(\text{poly}(n, \log(\kappa/\varepsilon)))$ qui réalise
$$|b\rangle|0\rangle = \frac{1}{\|b\|} \sum_j b_j |j\rangle \mapsto \frac{1}{\|b\|} \sum_j b_j |j\rangle \left(\frac{1}{\kappa\lambda_j} |0\rangle + \sqrt{1 - 1/(\kappa\lambda_j)^2} |1\rangle \right)$$
- La contribution sur $|\dots\rangle|0\rangle$ a pour norme $\sqrt{\sum_j b_j^2 / (\kappa\lambda_j)^2 / \sum_j b_j^2} \geq 1/\kappa$
L'observation du dernier bit renvoie **0** avec probabilité au moins $1/\kappa^2$
- Amplitude amplification (cours 5) : κ itérations suffisent !

A hermitien

- A est diagonalisable en base orthonormée $|\psi_j\rangle$

$$A = \sum_j \lambda_j |\psi_j\rangle \langle \psi_j|, \quad \text{avec } \lambda_j \text{ réels}$$

Réalisation des inversions

- Réaliser $|\psi_j\rangle |0\rangle \mapsto |\psi_j\rangle \left(\frac{1}{\kappa \lambda_j} |0\rangle + \sqrt{1 - 1/(\kappa \lambda_j)^2} |1\rangle \right)$

mais $|\psi_j\rangle$ est une superposition et de plus inconnue !

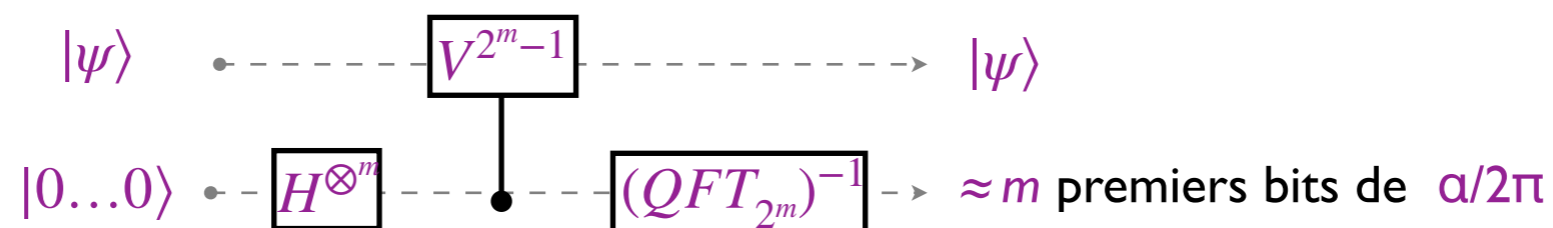
Méthodologie

- Remarque : Seule la connaissance de λ_j est nécessaire !
- Idée I : Utiliser l'estimation de phase (cours 4) pour calculer λ_j

Estimation de phase (forme compacte et optimisée)

- Entrée : Une superposition $|\psi\rangle$ telle que $V|\psi\rangle = e^{i\alpha}|\psi\rangle$
 et un accès à V via la transformation unitaire $c\text{-}V^T$
 $|t\rangle|\psi\rangle \mapsto |t\rangle(V^t|\psi\rangle)$ pour $0 \leq t \leq T$
 Sortie : α à précision ε ($\log(1/\varepsilon)$ bits de précision)
- Complexité : $O(\log(1/\varepsilon)/\varepsilon)$ portes à 2 qubits
 et 1 porte $c\text{-}V^T$ avec $T = 1/\varepsilon + O(1)$

On néglige ici la probabilité d'erreur.
 Avec un surcoût multiplicatif en $\log(1/\delta)$, elle est bornée par δ



A hermitien

- A est diagonalisable en base orthonormée $|\psi_j\rangle$

$$A = \sum_j \lambda_j |\psi_j\rangle \langle \psi_j|, \quad \text{avec } \lambda_j \text{ réels}$$

Réalisation des inversions

- Réaliser $|\psi_j\rangle |0\rangle \mapsto |\psi_j\rangle \left(\frac{1}{\kappa \lambda_j} |0\rangle + \sqrt{1 - 1/(\kappa \lambda_j)^2} |1\rangle \right)$
mais $|\psi_j\rangle$ est une superposition et de plus inconnue !

Méthodologie

- Remarque : Seule la connaissance de λ_j est nécessaire !
- Idée 1 : Utiliser l'estimation de phase (cours 4) pour calculer λ_j
Mais A n'est pas unitaire
- Idée 2 : Cependant e^{iA} l'est et est simulable à partir de A !
En particulier si A a au plus s éléments non nuls par ligne...
- Observation : e^{iA} a les mêmes vecteurs propres $|\psi_j\rangle$ que A
mais les valeurs propres sont devenues $e^{i\lambda_j}$

Inversion partielle sur un vecteur propre

- Etat initial : $|\psi_j\rangle|0\dots 0\rangle|0\rangle$
 - Estimation de phase sur l'opérateur e^{iA} : $|\psi_j\rangle|\lambda_j\rangle|0\rangle$
 - Inversion : $|\psi_j\rangle|\lambda_j\rangle\left(\frac{1}{\kappa\lambda_j}|0\rangle + \sqrt{1 - 1/(\kappa\lambda_j)^2}|1\rangle\right)$
 Rotation calculée à partir de $|\lambda_j\rangle$
 - Inversion de l'estimation de phase : $|\psi_j\rangle|0\dots 0\rangle\left(\frac{1}{\kappa\lambda_j}|0\rangle + \sqrt{1 - 1/(\kappa\lambda_j)^2}|1\rangle\right)$
- Précision $O(\varepsilon/\kappa)$ nécessaire dans l'estimation de λ_j pour obtenir une erreur relative ε ($1/\kappa \leq |\lambda| \leq 1$)

Inversion partielle - vue d'ensemble

- Etat initial : $|0\dots 0\rangle|0\dots 0\rangle|0\rangle$
- Créer le vecteur b : $|b\rangle|0\dots 0\rangle|0\rangle$
- Inversion partielle : $\alpha|x\rangle|0\dots 0\rangle|0\rangle + \sqrt{1 - \alpha^2}|\dots\rangle|0\dots 0\rangle|1\rangle$ avec $|\alpha| \geq 1/\kappa$

Amplification et analyse

- $O(\kappa)$ inversions partielles (et vérifications que le dernier bit est 0)
- Estimation de phase avec précision $O(\varepsilon/\kappa)$: Simulation de e^{itA} avec $t = O(\kappa/\varepsilon)$
- Complexité totale : $O(\kappa^2/\varepsilon)$ simulations de e^{iA} , $O(\kappa)$ créations de $|b\rangle$ et $O(\kappa^2/\varepsilon \times \text{polylog}(\kappa/\varepsilon))$ portes 2-qubits
 → Meilleure complexité finale connue en $O(s\kappa \times \text{polylog}(s\kappa N/\varepsilon))$

Applications

Applications

- Equations différentielles linéaires, aux dérivées partielles
- Machine learning, optimisation : cf le séminaire associé au cours !

Difficultés

- La solution classique du système linéaire est encodée en superposition
- Contrairement à la simulation hamiltonienne, cette superposition est artificielle, et donc a priori inutilisable
- Il faut donc revenir à la solution du problème

Tomographie

- Apprendre x de dimension $N = 2^n$ à partir $|x\rangle$ sur n qubits
- Coût : $O(N \log N / \epsilon^2)$ [Kerenidis, Prakash 2020]

Challenge

- Extraire des propriétés globale de la solution encodée quantiquement sans avoir à faire de tomographie
- Succès du machine learning classique en partie empirique
Gain effectif à expérimenter !

Contexte

- Résoudre $\frac{d}{dt}x = Ax + b$ à N variables
au temps $T=t$ avec condition initiale $x(t=0)$

Approche [Berry 2014]

- **Discrétisation** : système linéaire
 $x(t+h) = x(t) + (Ax(t) + b)h$ où h est le **pas de discrétisation**
- $x(h), x(2h), \dots, x(T)$ est solution du système linéaire $(NT/h) \times (NT/h)$:

$$\begin{pmatrix} \text{Id} & 0 & 0 & 0 & \dots \\ -(\text{Id} + Ah) & \text{Id} & 0 & 0 & \dots \\ 0 & -(\text{Id} + Ah) & \text{Id} & 0 & \dots \\ \vdots & \ddots & \ddots & \ddots & \dots \end{pmatrix} \begin{pmatrix} x(h) \\ x(2h) \\ x(3h) \\ \vdots \end{pmatrix} = \begin{pmatrix} x(0) \\ bh \\ bh \\ \vdots \end{pmatrix}$$
- Résoudre numériquement. Meilleure complexité actuelle en $O((\kappa s T/h) \times \text{polylog}(\kappa s NT/(h\varepsilon)))$

Généralisations

- Equations différentielles partielles
- Coefficients non-constants
- Equations non-linéaires

Améliorations Extensions

Résolution du système linéaire

- Quantum Algorithm for Systems of Linear Equations with Exponentially Improved Dependence on Precision [[Childs, Kothari, Somma 2017](#)]
- A quantum linear system algorithm for dense matrices [[Wossnig, Zhao, Prakash 2018](#)]
- Concrete Resource Analysis of the Quantum Linear System Algorithm used to Compute the Electromagnetic Scattering Cross Section of a 2D Target [[Scherer, Valiron, Mau, Alexander, Berg, Chapuran 2017](#)]

Algèbre linéaire

- Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics [[Gilyén, Su, Low, Wiebe 2019](#)]
- The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation [[Chakraborty, Gilyén, Jeffery 2019](#)]

Autres approches

- Quantum Speedup for Graph Sparsification, Cut Approximation and Laplacian Solving [[Apers, Wolf 2020](#)]

Equations différentielles

- High-order quantum algorithm for solving linear differential equations [Berry 2014]
- Quantum algorithms and the finite element method [Montanaro, Pallister 2016]
- Quantum Algorithm for Linear Differential Equations with Exponentially Improved Dependence on Precision [Berry, Childs, Ostrander, Wang 2017]
- Quantum algorithm for simulating the wave equation [Costa, Jordan, Ostrander 2019]
- Efficient quantum algorithm for dissipative nonlinear differential equations [Liu, Kolden, Krovi, Loureiro, Trivisa, Childs 2020]

Machine learning, optimisation

- Quantum Recommendation System [Kerenidis, Prakash 2017]
- A co-design framework of neural networks and quantum circuits towards quantum advantage [Jiang, Xiong, Shi 2021]
- Quantum algorithms for second-order cone programming and support vector machines [Kerenidis, Prakash, Szilágyi 2021]

Déquantization

- A quantum-inspired classical algorithm for recommendation systems [Tang 2019]
- Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning [Chia, Gilyén, Li, Lin, Tang, Wang 2020]
- Quantum-inspired algorithms in practice [Arrazola, Delgado, Bardhan, Lloyd 2019]

Algèbre linéaire revisitée (quantum inspired/motivated)

- On Solving Linear Systems in Sublinear Time [Andoni, Krauthgamer, Pogrow 2019]
- Faster quantum-inspired algorithms for solving linear systems [Changpeng Shao, Montanaro 2021]
- Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension [Gilyén, Lloyd, Tang 2020]

Par où commencer ?

Articles fondateurs

- Quantum algorithm for linear systems of equations [[Harrow, Hassidim, Lloyd 2009](#)]
- High-order quantum algorithm for solving linear differential equations [[Berry 2014](#)]
- Quantum Recommendation System [[Kerenidis, Prakash 2017](#)]

Thèses

- Quantum singular value transformation and its algorithmic applications [[Gilyén 2019](#)]
- Quantum Algorithms for Differential Equations [[Ostrander 2019](#)]
- Quantum algorithms for machine learning [[Luongo 2021](#)]

Séminaire du cours !

- Quantum Machine Learning
avec Jordanis Kerenidis, CNRS, Paris