QUANTUM ALGORITHMS FOR "BIG DATA MACHINE LEARNING"

- QUANTUM LINEAR-ALGEBRAIC SUBROUTINES

- THE Q-DATABASE (QRAM) ASSUMPTION

- APPLICATIONS: Q. SVM (2013), RECOMMENDER SYSTEMS
  - Q. DATA FITTING (2012)

FROM PREVIOUS 2 LESSONS

① Hamiltonian simulation (Trotter): Given sparse oracle to $M \in \mathcal{L}(\mathcal{H})$ ($M = M^\dagger$)

can implement $U_M = e^{iMst}$; Let $M|\vec{\lambda}\rangle = \lambda|\vec{\lambda}\rangle$

+ Q. PHASE ESTIMATION + $|\vec{\lambda}\rangle|''\lambda''\rangle \mapsto |\vec{\lambda}\rangle|''\lambda''\rangle\left(f(\lambda)|0\rangle + \sqrt{1-f(\lambda)^2}\,|1\rangle\right)$

$\xrightarrow{\text{measure } ''0''} f(\lambda)|\vec{\lambda}\rangle|''\lambda''\rangle \xrightarrow{QPE^{-1}} f(\lambda)|\vec{\lambda}\rangle$ ..

= AMPLITUDE IMPRINTING

=> can implement $|\psi\rangle \mapsto f(M)|\psi\rangle$, e.g. $f(x) = x^{-1}$

(2.) Other methods (LIN. COMB. UNITARIES, QUBITIZATION, Q. SINGULAR VALUE TRANSFORM) allow directly to implement $|\psi\rangle \longmapsto f(M)|\psi\rangle$

(INTUITION: "BLOCK ENCODING": put M in A BLOCK OF A (LARGER) U)

$$U_M = \left(\begin{array}{c|c} M/\alpha & * \\ \hline * & * \end{array}\right) \longmapsto U_{f(M)} = \left(\begin{array}{c|c} f(M) & * \\ \hline * & * \end{array}\right)$$

Theorem (sketch)

Given U_M & a polynomial f can implement $U_{f(M)}$ using $O(\deg(f))$ calls...

(2.) Other methods (LIN. COMB. UNITARIES, QUBITIZATION, Q. SINGULAR VALUE TRANSFORM) allow directly to implement $|\psi\rangle \longmapsto f(M)|\psi\rangle$

(INTUITION: "BLOCK ENCODING": put M in A BLOCK OF A (LARGER) U)

WHY is THIS WHAT I WANT?

$$\left(\begin{array}{c|c} f(M) & * \\ \hline * & * \end{array}\right) \cdot |0\rangle|x\rangle = \left(\begin{array}{c|c} f(M) & \cdot \\ \hline \cdot & \cdot \end{array}\right)\begin{pmatrix} \vec{x} \\ 0 \end{pmatrix} = \begin{pmatrix} f(M)\vec{x} \\ \hline * \end{pmatrix}$$

$$(\langle 0| \otimes \mathbb{I}) \begin{pmatrix} f(M)\vec{x} \\ \hline * \end{pmatrix} = (\mathbb{I}, 0)\begin{pmatrix} f(M)\vec{x} \\ \hline * \end{pmatrix} = f(M)\vec{x}$$

? Topological data analysis ? ----> estimating Betti numbers

---> estimating kernel dimension of (large) MATRIX M

with SPARSE ACCESS (OR VIA DATA GRAPH)

NULLITY ? = (Kernel dim. estimation)

How ?

DO QPE ON e^{iMat}) ON a RANDOM EIGENVECTOR OF M

=> COUNT ZERO PHASES v.s. total count : $P(0) = \frac{\dim(\ker(M))}{\dim(M)}$

? Topological data analysis ? ----> estimating Betti numbers

---> estimating kernel dimension of (large) MATRIX M

with SPARSE ACCESS (OR VIA DATA GRAPH)

NULLITY ? = (Kernel dim. estimation)

How ?

BUT HOW DO I GET A RANDOM EIGENVECTOR OF M ?

DON'T NEED TO:

$$\mathbb{1}/2^n = \sum \frac{1}{2^n} |\psi_i\rangle\langle\psi_i| =$$

$$\mathbb{1}/2^n = U\,\mathbb{1}/2^n\,U^\dagger = \sum \frac{1}{2^n} |\varphi_i\rangle\langle\varphi_i| \leftarrow$$

For any orthon. basis

POWER OF SAMPLING FROM QUANTUM COMPUTATIONS

NB : CAN "PURIFY" ALL ...

SOME LINEAR ALGEBRA QUANTUM SUBROUTINES

Let $\vec{x} = (x_i) \in \mathbb{R}^N$, $\vec{y} = (y_j) \in \mathbb{R}^N$ (or $\mathbb{C}^N$)

→ Access entrywise, not sparse

→ what is the cost of approximating $\vec{x}^\dagger \cdot \vec{y} = \langle \vec{x} | \vec{y} \rangle = \sum_j x_i^* x_j$?

non-normalized states

SOME LINEAR ALGEBRA QUANTUM SUBROUTINES

Let $\vec{x}=(x_i)\in\mathbb{R}^N$, $\vec{y}=(y_j)\in\mathbb{R}^N$ (or $\mathbb{C}^N$)

Assume $\|\vec{x}\|=a$, $\|\vec{y}\|=b$ known, access to $U, V, U^\dagger, V^\dagger$,

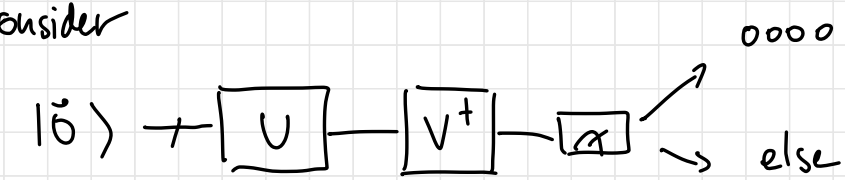
$U|0\rangle=|\vec{x}\rangle$ $\left(|\vec{x}\rangle:=\frac{1}{a}\sum x_i|i\rangle\right)$ ["AMPLITUDE ENCODING OF $\vec{x}$"]

$V|0\rangle=|\vec{y}\rangle$ $\left(|\vec{y}\rangle:=\frac{1}{b}\sum y_i|i\rangle\right)$

NB. $|\vec{x}\rangle$ is a $\lceil\log_2(N)\rceil$ - qubit state

Without loss of generality $N=2^n$.

Consider



$$P(\vec{0}) = |\langle \vec{0}| V^\dagger U |\vec{0}\rangle|^2 = |\langle \vec{x}|\vec{y}\rangle|^2 = \alpha$$

REPEAT → ESTIMATE $\tilde{\alpha}$ → Compute $\sqrt{\tilde{\alpha}} \cdot a \cdot b$ =>

in $\frac{1}{\varepsilon}$ reps., this is $O(\varepsilon)$-error estimate of $\vec{x}^\dagger \cdot \vec{y}$.

=> SEE: "STANDARD ERROR"
"ERROR UNCERTAINTY PROPAGATION"
"CHERNOFF BOUNDS"
"HOEFDING'S INEQUALITIES"

Cost of ε-approximation? $O(1/\varepsilon)$ × cost of generating U & V.

If U & V are polylogarithmic circuits (in N, poly in n)

$\Rightarrow$ $O(\text{polylog}(N)/\varepsilon)$ [vs $O(N)$ classically, naively]

CAVEAT

Cost of ε-approximation? $O(1/\varepsilon)$ × cost of generating U & V.

If U & V are polylogarithmic circuits (in N, poly in n)

$\Rightarrow$ $O(\text{polylog}(N)/\varepsilon)$ [vs $O(N)$ classically, naively]

CAVEAT

$$|\langle \vec{x}|\vec{y}\rangle| = |\langle \vec{x}|\vec{y}\rangle \cdot e^{i\theta}| \quad \forall\, \theta \ldots$$

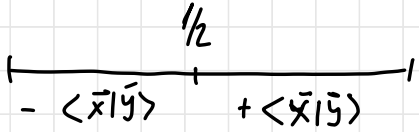
Esp. $\langle \vec{x}|\vec{y}\rangle \neq -\langle \vec{x}|\vec{y}\rangle$ may matter A LOT.

SOLUTION FOR $\in \mathbb{R}$ $\quad \vec{x} \in \mathbb{C}^N \to \vec{x}' \in \mathbb{C}^{N+1}$

$$\begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_N \end{bmatrix} \to \begin{bmatrix} 1/\sqrt{2} \\ x_1/\sqrt{2} \\ x_2/\sqrt{2} \\ \vdots \\ x_N/\sqrt{2} \end{bmatrix}$$

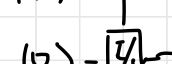
$$|\vec{x}\rangle = \sum_{i=1}^{N} \tilde{x}_i |i\rangle \quad \rightarrow |\vec{x}'\rangle = \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|\vec{x}\rangle$$

$$|\langle \vec{x}'|\vec{y}'\rangle| = |\frac{1}{2} + \frac{1}{2}\underbrace{\langle \vec{x}|\vec{y}\rangle}_{\text{Norm} - 1}|$$



⇒ Do this by modifying $U_1 U_f$ or using ctr-V

Eg.



$$= \frac{1}{\sqrt{2}}|0\rangle|0\rangle + |1\rangle|\vec{x}\rangle \rightarrow \frac{1}{\sqrt{2}}|0\rangle + \frac{1}{\sqrt{2}}|\vec{x}\rangle$$ with const. prob.

→ Fixed point amplitude amplification will do the job...

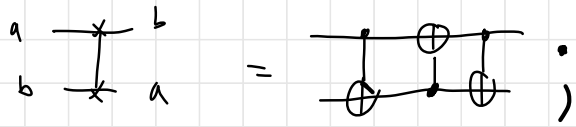
SUPPOSE YOU DO NOT HAVE U^+, V^+ ...

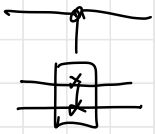
OR JUST HAVE $|\vec{x}\rangle^{\otimes k}$, $|\vec{y}\rangle^{\otimes k}$ ...

WHAT THEN?

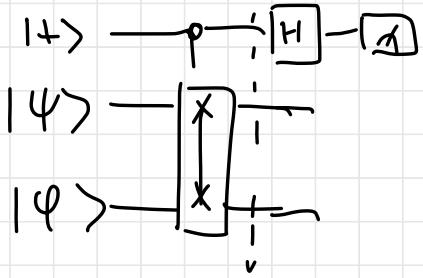
"SWAP TEST"

Buhrman, Cleve, Watrous, de Wolf (2001). "QUANTUM FINGERPRINTING"

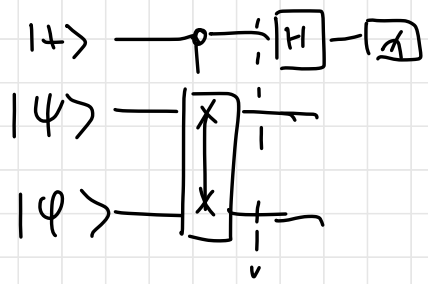
SWAP = 

; Ctrl-Swap = 

"SWAP TEST"



"SWAP TEST"



$$\frac{1}{\sqrt{2}}|0\rangle|\psi\rangle|\varphi\rangle + |1\rangle|\varphi\rangle|\psi\rangle \rightarrow$$

outcome "0" $= (\langle 0|H\otimes\mathbb{1})(-\!|\!|-) = ((\langle +|\otimes\mathbb{1})(-\!|\!|-)$

$$= \left\| \frac{1}{2}|\psi\rangle|\varphi\rangle + \frac{1}{2}|\varphi\rangle|\psi\rangle \right\|^2 = \left\| \frac{1}{2}\left(|\psi\rangle|\varphi\rangle + |\varphi\rangle|\psi\rangle\right) \right\|^2 = \frac{1}{2}\left(1+|\langle\varphi|\psi\rangle|^2\right)$$

"1" $= \left\| \frac{1}{2}\left(|\psi\rangle|\varphi\rangle - |\varphi\rangle|\psi\rangle\right) \right\|^2 = \frac{1}{2}\left(1-|\langle\varphi|\psi\rangle|^2\right)$

From $\frac{1}{2}(1+|\langle\psi|\psi\rangle|^2)$ to $|\langle\psi|\psi\rangle|$...

$$\frac{1}{2}(x^2+1) = p_0 \Rightarrow \frac{1}{2}x^2 = p_0 - \frac{1}{2} \Rightarrow x^2 = 2p_0 - 1 \Rightarrow x = \sqrt{2p_0-1} \ldots$$

SWAP TEST & QUANTUM PROGRAMS

... Almost like simulating projective measurement onto $|\psi\rangle$ given A REGISTER $|\psi\rangle^{\otimes k}$.

Quantum Density Matrix Exponentiation.

Density Matrix Formalism

$|\psi\rangle \to |\psi\rangle\langle\psi| \in \mathcal{L}(\mathcal{H})$

↑ Positive-semidefinite, Hermitian Matrix

$\rho = \sum_j p_j |\psi_j\rangle\langle\psi_j|$

↑ $\mathrm{Tr}(\rho) = 1$ & ρ is PSD

$\Leftrightarrow$ ρ is a STATE

- ignorance
- randomness
- subsystem ...

Hamiltonian Simulation

Given $H \in \mathcal{L}(\mathcal{H})$

↑ matrix / operator ↑ vector (Hilbert) space

if H is Hermitian ($H = H^\dagger$), it is a Hamiltonian.

$\to$ $|\psi_{\Delta t}\rangle = e^{iH\Delta t}|\psi_0\rangle$

Quantum Density Matrix Exponentiation.

Density Matrix Formalism

$|\psi\rangle \rightarrow |\psi\rangle\langle\psi| \in \mathcal{L}(\mathcal{H})$

↑ Positive-semidefinite, Hermitian Matrix

$\rho = \sum_j p_j |\psi_j\rangle\langle\psi_j|$

↑ $\mathrm{Tr}(\rho) = 1$ & ρ is PSD

$\Leftrightarrow$ ρ is a state

Hamiltonian Simulation

Given $H \in \mathcal{L}(\mathcal{H})$

↑ matrix / operator; ↑ vector (Hilbert) space

if H is Hermitian ($H = H^\dagger$),

it is a Hamiltonian.

$\rightarrow |\psi_{\Delta t}\rangle = e^{iH\Delta t} |\psi_0\rangle$

A matrix representation of a quantum state... is a valid representation of an evolution generator / dynamics

For Ham simulation, usually

H is given as a sparse (access) ORACLE...

- Let H be a PSD Hamiltonian
- Let $\rho = \frac{H}{\mathrm{Tr}(H)}$...

DENSITY MATRIX EXPONENTIATION:

I give you a register in state $\rho^{\otimes k}$... and input state $|\psi\rangle$...

Can you return $\exp(i\rho\Delta t)|\psi\rangle$ back?

$\Rightarrow$ Trivial: $k \to \infty$, tomography, reconstruct $[\rho]$ (matrix)

→ make circuit... exponential effort in $\log_2(\dim H)$

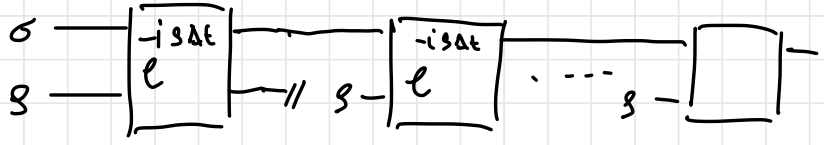
Lloyd, Mosheni, Rebentrost (2013) "QUANTUM PRINCIPAL COMPONENT ANALYSIS".

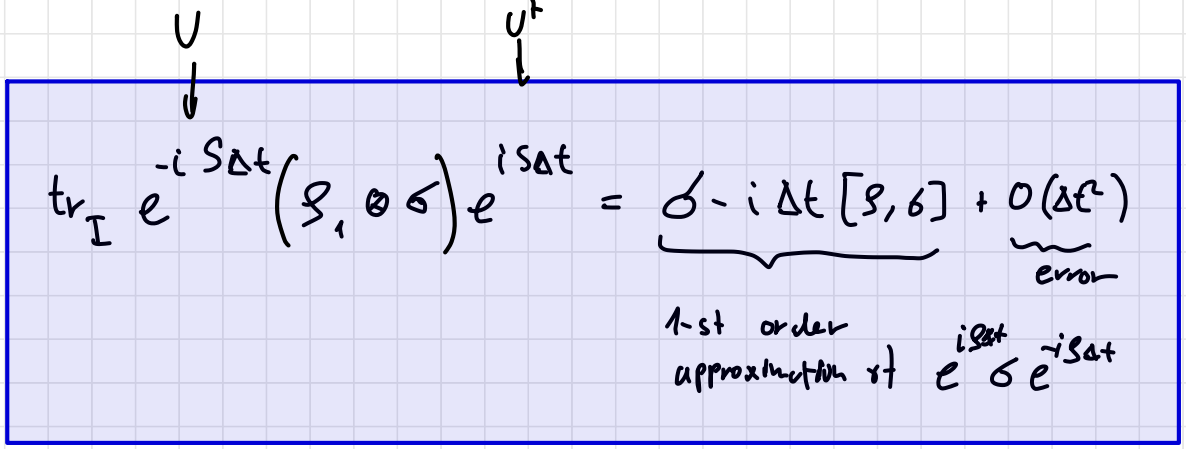
"Partial SWAP". $S = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix} \ldots$ → UNITARY, → HERMITIAN!

$e^{-iS\Delta t}$ ... A PARTIAL SWAP ... (@ $\Delta t = \pi/2$ ... iSWAP ...)

→ "program" $S^{\otimes k}$

→ input σ (think $\sigma = |\psi\rangle\langle\psi|$)



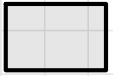


--- repetition .. using smaller Δt (recall trotterization trick)

to realize $e^{iSt}|\psi\rangle$ within ε-error,

$O(t^2/\varepsilon)$ copies /steps

QRAM : Last piece of puzzle...

RAM address →  → value @ address

QRAM : ?

" Linear, reversible extension of RAM "

QRAM: $|add.\rangle|0\rangle \rightarrow$ [QRAM] $\rightarrow |add.\rangle|Value\ @\ address\rangle$

Eg. $\vec{x} = (x_i) \quad i = 0 \ldots 2^n - 1 = N-1$

$|\vec{i}\rangle|0\rangle \xrightarrow{U_{RAM}} |\vec{i}\rangle|x_i\rangle$ But also

$\sum \alpha_i |i\rangle|0\rangle \xrightarrow{U_{RAM}} \sum \alpha_i |i\rangle|x_i\rangle$

CAN BE REALIZED USING $O(\text{polylog}(N))$ INTERACTIONS

(1.) AMPLITUDE ENCODING

$$\sum \frac{1}{\sqrt{N}} |i\rangle|0\rangle \longrightarrow \sum \frac{1}{\sqrt{N}} |i\rangle|x_i\rangle \xrightarrow{\text{PROP. TO}} \sum \frac{1}{\sqrt{N}} |i\rangle|x_i\rangle \left(x_i|0\rangle + \sqrt{1-x_i^2}|1\rangle\right)$$

$$\xrightarrow{\langle 0|} \sum \frac{1}{\sqrt{N}} x_i |i\rangle|x_i\rangle \xrightarrow{U_{RAM}^{-1}} \sum \frac{1}{\sqrt{N}} x_i |i\rangle$$

=> CAN TAKE $O(\sqrt{N})$! By GROVER OPTIMALITY

↳ "QUANTUM RECOMMENDER SYSTEMS"

↳ Zhao, Fitzsimons, Rebentrost, VD, Fitzsimons .. "smooth preparation"

Bottom line : DATABASE

$\vec{x}^j \qquad \vec{x}^j = (\vec{x}^j_i)$

$$|j\rangle|0\rangle \rightarrow |j\rangle|\vec{x}^j\rangle$$

$$\uparrow$$

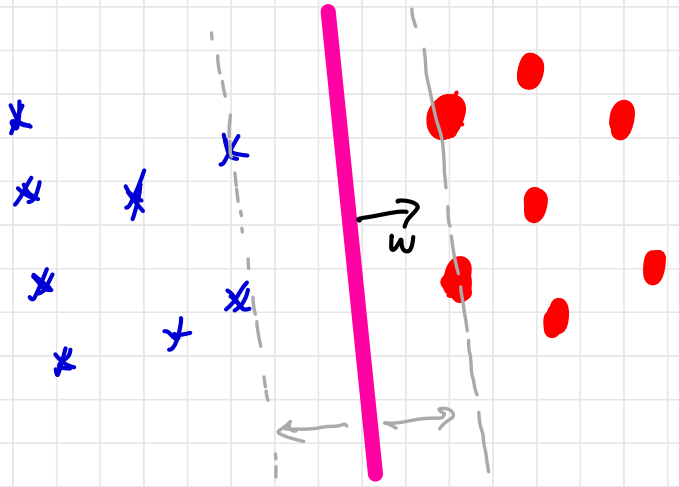
$$\propto \sum_i \vec{x}^j_i |i\rangle$$

In polylog(N) !!! (IF DATABASE ALREADY LOADED)

APPLICATIONS IN BIG DATA.

QUANTUM SUPPORT VECTOR MACHINE

INPUT: M x N-dimensional (real) vector, (in QRAM)



→ "MAXIMUM MARGIN SEPARATING HYPERPLANE"
defined by N - coordinates

OR... CLASS OF NEW DATAPOINT

CLASSICALLY: $(\vec{x}_i, y_i)$

$\vec{x}_i$: Point; y_i: label $\in \{-1, 1\}$

$$\min \|w\|$$

Subject to

$$y_i (\vec{w} \cdot \vec{x}_i - b) \geq 1 \quad \forall i$$

DUAL FORMULATION: maximize $\vec{\alpha} = (\alpha_1 \dots \alpha_m)$

$$L(\vec{\alpha}) = \sum_{j=1}^{M} y_j \alpha_j - \frac{1}{2} \sum_{j,k=1}^{M} \alpha_j K_{jk} \alpha_k \; ; \quad K_{jk} = k(\vec{x}_j, \vec{x}_k) \overset{?}{=} (\vec{x}_j, \vec{x}_k)$$

Subject to $\sum \alpha_j = 0$, $y_i \alpha_i \geq 0$

Classical cost -- quadratic program less than $O(n^3)$

$O\left(M^2(N+M)\log(1/\varepsilon)\right)$ -- note K has $\frac{M(M-1)}{2}$ inner products, each costs $O(N)$

INNER PRODUCT TRICK:

Given QRAM (containing $\|\vec{x}_j\|$ & $|\vec{x}_j\rangle$'s)

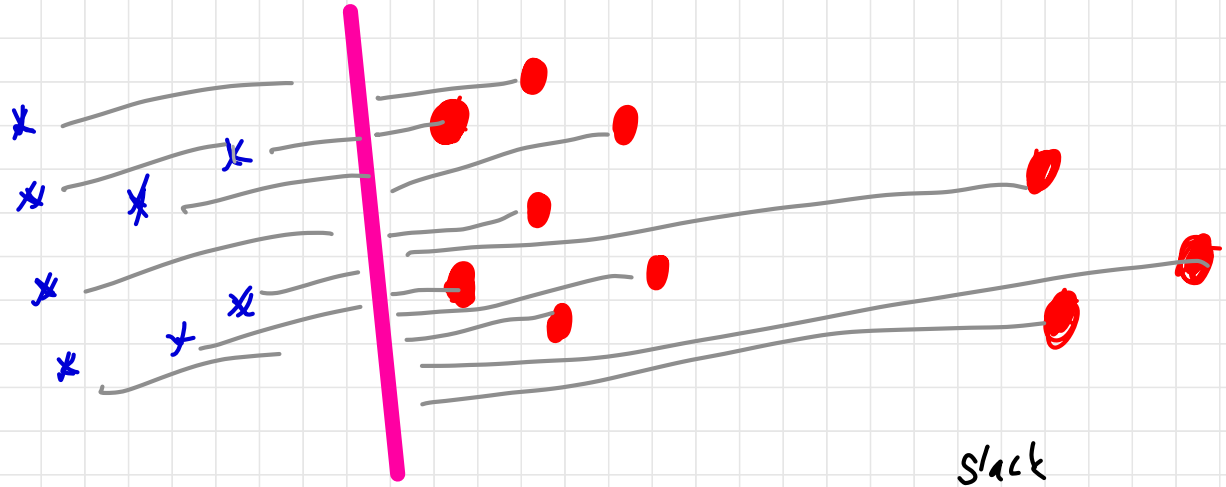
Estimate all inner products in time $O(\text{Log}(N) \times M^2)$ + Program

$$= O\left(\log(1/\varepsilon) M^3 + M^2 \log(N)/\varepsilon\right)$$

EXPONENTIAL IMPROVEMENT IN N

Exponentially worse in ε -- IT'S ML, WHO CARES, SET $\varepsilon = 10^{-2}$!

MUCH BETTER IMPROVEMENTS POSSIBLE... FOR LEAST-SQUARES SUM ?



slack

$y_j(\vec{w}.\vec{x}_j+b) \geq 1 \longrightarrow y_i(\vec{w}.\vec{x}_j+b) \geq 1-e_i \longrightarrow \underbrace{y_i(\vec{w}.\vec{x}_j+b)=1-e_j}_{\text{LS-SUM}}$

Penalty $\gamma/2 \sum e_j^2$ ← SQUARED DISTANCE

=> ESSENTIALLY LEAST SQUARES FIT

$$f(\vec{x}^i, \vec{\beta}) = \sum_{j=1}^{m} \beta_j x_j^i$$

Error = $r_i = y_i - f(x^i, \vec{\beta})$

minimize $\sum r_i^2$ $\quad$ $\mathbf{X}_{ij}$ = matrix of datapoints

=> minimize $|\mathbf{X}\vec{\beta} - \vec{y}|$

Solution: $\vec{\beta} = \mathbf{X}^+ \vec{y}$; MOORE-PENROSE PSEUDOINVERSE

=> More powerful than may seem.

$$f(\vec{x}, \vec{\beta}) = \sum_{j=1}^{m} \beta_j \phi_j(x)$$

ϕ_j was projection onto j^{th} component in prev. slide..

Error = $r_i = y_i - f(x_i, \vec{\beta})$

Minimize $\sum r_i^2$

$\mathbf{X}_{ij} = \phi_j(x_i)$

=> minimize $|\mathbf{X}\vec{\beta} - \vec{y}|$

Solution: $\vec{\beta} = \mathbf{X}^{+}\vec{y}$; MOORE-PENROSE PSEUDOINVERSE

=> More powerful than may seem. Non-linear

$$f(\vec{x}_i, \vec{\beta}) = \sum_{j=1}^{m} \beta_j \phi_j(x_i)$$

$$\phi_j(\vec{x}) = x^j$$

$$X = \begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^m \\ 1 & x_2 & \vdots & & \vdots \\ 1 & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & & x_n^m \end{bmatrix}$$

$$y_i = \sum \beta_j x_i^j \quad \Rightarrow \text{ polynomial fit ...}$$

$$F^+ = (F^\dagger F)^{-1} F^\dagger \qquad \text{(when } F^\dagger F \text{ is invertible)}$$

otherwise $F = U D V^\dagger \rightarrow V D^{-1} U^\dagger$

D^{-1}: q with $\frac{1}{q} := 0$

IDEA: GIVEN ACCESS TO F AS A QUANTUM DATABASE... IT IS JUST APPLYING MATRICES + ONE MATRIX INVERSION..

LS-SVM : $\min \frac{1}{2}\|w\| + \gamma \sum e^2$

subject to $y_i(w.\vec{x}_i + b) = 1 - e_i$

=> Lagrangian form leads to just a Linear system

TURNS OUT .. LS-SVM SOLUTION REDUCES

TO FOLLOWING SYSTEM

$$F\begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} \equiv \begin{pmatrix} 0 & \vec{1}^T \\ \vec{1} & K+\gamma^{-1}I \end{pmatrix}\begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ y \end{pmatrix}$$

$$b\vec{1} + (K+\gamma^{-1}I)\vec{\alpha} = \vec{y}$$

$w = \sum_j \alpha_j \vec{x}_j$

almost Lazy learner...

$w.\vec{x}' = \sum \alpha_i (x_i | x') \ldots$

SUPPOSE HAVE SOLUTION VIA QLS

$$\Rightarrow \underline{|b, \vec{\alpha}\rangle} = \frac{1}{\sqrt{C}}\left(b|0\rangle + \sum_{i=1}^{M} \alpha_i |i\rangle \right)$$

To CLASSIFY $|\vec{x}\rangle$,

First construct $|u\rangle = \frac{1}{\sqrt{N_v}}\left(b|0\rangle|0\rangle + \sum_{k=1}^{n} \alpha_k |\vec{x_k}| \, |k\rangle|\vec{x_k}\rangle \right)$

Possible if using $|b, \vec{\alpha}\rangle$ in Query!

$$|u\rangle = \frac{1}{\sqrt{N_v}} \left(b|0\rangle|0\rangle + \sum_{k=1}^{M} \alpha_k |\vec{x}_k| \, |k\rangle |\vec{x}_k\rangle \right)$$

Next construct $$|\vec{x}\rangle = \frac{1}{\sqrt{N_x}} \left(|0\rangle|0\rangle + \sum_{k=1}^{M} |\vec{x}| \, |k\rangle |\vec{x}\rangle \right)$$

Estimate $|\langle \vec{x}|u\rangle|^2$ via SWAP TEST

$$= \frac{1}{\sqrt{N_v N_x}} \left(b + \sum_{i=1}^{M} \alpha_i |\vec{x}_k| \, |\vec{x}| \, \langle \vec{x}_k | \vec{x} \rangle \right)$$

(efficient when $\vec{\alpha}$ not sparse)

$\alpha_i |\vec{x}_k|$ = expresses hyperplane normal vector

$= \underbrace{b + (w, \vec{x})}$ → estimate sign. done.

REMAINING.

unknown

$$F\begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} \equiv \begin{pmatrix} 0 & \mathbb{1}^T \\ \vec{1} & K + \gamma^{-1}\mathbb{1} \end{pmatrix} \begin{pmatrix} b \\ \vec{\alpha} \end{pmatrix} = \begin{pmatrix} 0 \\ y \end{pmatrix}$$

Sparse access? NOT!!

data vector in database ✓

$K_{ij} = x_i^T \cdot x_j$ ...

IDEA : construct Quantum state

$\rho \propto K$ (note K is GRAMMIAN, so PSD)

And do $K \to e^{iK\Delta t}$ → suffices for Q.L.S.

How? Use <u>QRAM</u>:

$$|\chi\rangle \propto \sum_{i=1}^{M} |\vec{x}_i| |i\rangle_1 |\vec{x}_i\rangle_2$$

↑ norm ↑ normalized

Cost $O(\log(MN))$

→ TRACE OUT 1!

$$= \frac{1}{N_x} \sum_{i,j}^{M} \underbrace{\langle \vec{x}_i | \vec{x}_j \rangle}_{\text{Value}} |x_i||x_j| \, |i\rangle\langle j|$$

↗ entry of matrix $= \frac{K}{\mathrm{tr}(K)}$

Complexity:

$$O\left(k_{eff}^3 \log(M \cdot N)/\varepsilon^3\right) \overset{?}{=} 0$$

$\Rightarrow$ Non-linearities: $K = \vec{\phi}(\vec{x_j}) \cdot \vec{\phi}(\vec{x_k})$

$$|\phi(\vec{x_j})\rangle = \underbrace{|\vec{x_j}\rangle |\vec{x_j}\rangle \cdots |\vec{x_j}\rangle}_{e}$$

$\Rightarrow$ $\langle \phi(x_j) | \phi(x_k) \rangle = (x_j^T \cdot x_k)^e$ $\Rightarrow$ can be used to construct polynomial kernels.

→ REGRESSION => QUANTUM L.S. FIT APPROXIMATION

→ QUANTUM RECCOMENDER SYSTEMS:
Polylog approximation of low-rank matrix of data.

→ QUANTUM K-means

→ QUANTUM KRIGING (GAUSSIAN PROCESS REGRESSION)

→ QUANTUM PCA

ALL IN POLYLOG(N) ...

... (2018) .. EWIN TANG .. et al.

ALL THESE ALGORITMS IN

CLASSICAL POLYLOG (N) ...

But $O(n^{16})$ vs $O(n^3)$ .... for now.

STILL AMAZING .. BUT ..

HAVE YOU SEEN ANY Q-DATABASES AROUND ?

TDA∇_0 ...

QUANTUM LINEAR ALGEBRA WITH APPLICATIONS IN BIG DATA MACHINE LEARNING

BIG PICTURE IDEA : $\vec{x} \in \mathbb{R}^N$, $(x_i)_i = \vec{x}$

- An (very-high) N-dimensional vector (data-point) can be "stored" in just $n = \lceil \log_2(N) \rceil$ qubit states, up to norm. $\vec{x}$ "$=$" $\|\vec{x}\|_2 \times \left(|\vec{x}\rangle = \sum_{i=1}^{N} \frac{x_i}{\|\vec{x}\|} |i\rangle \right)$ $\Rightarrow$ "amplitude encoding"
- THEY CAN BE MANIPULATED IN polylog time.. which is actually POLYNOMIAL time in n = size of Quantum register.
- GENERATING $\vec{x} \mapsto |\vec{x}\rangle$ ofc. costs $O(N)$ but with a "Quantum database" framework once the database is filled (= this counts as a "one-off" overhead)

  Preparing $|\vec{x}\rangle$ from the "Quantum database" can be done in $O(\text{polylog}(N))$ time

$\Rightarrow$ Superpositions over exponentially many exponentially long AMPLITUDE ENCODED DATA-VECTORS CAN BE PREPARED efficiently... $\Rightarrow$ $|0\rangle \Rightarrow \sum_{i=0}^{M-1} |i\rangle |\vec{x}^{(i)}\rangle$, $M = \exp(N)$, $\vec{x}^i$ is in $\mathbb{C}^N$... in $\text{polylog}(MN)$

MANIPULATING AMPLITUDE ENCODED DATA

=> inner products in $O(\text{polylog}(N)/\varepsilon)$ via swap test, or directly

(ε ↑ error)

$U|0\rangle = |\vec{x}\rangle$, $V|0\rangle = |\vec{y}\rangle$ $\Big\}$ $|\langle 0|V^\dagger U|0\rangle|^2 = |\langle \vec{x}|\vec{y}\rangle|^2$ ...

=> data can be "loaded in operations": can do unitaries defined by data... Let $\mathcal{S} \in \mathcal{L}(\mathcal{H})$ be a (P.S.D.) quantum (mixed) state

=> can generate $U = \exp(i\mathcal{S}\Delta t)$ via DENSITY MATRIX EXPONENTIATION

Also: Block encoding => $U' = \begin{bmatrix} \mathcal{S} & \cdot \\ \cdot & \cdot \end{bmatrix}$

=> Gram matrix of data (covariance matrix essentially same thing) in a state: $|0\rangle_1|0\rangle_2 \to \sum_i |i\rangle_1|\vec{x}^i\rangle_2 \xrightarrow{\text{Tr}_1} \sum_{ij} \langle \vec{x}^i|\vec{x}^j\rangle |i\rangle\langle j| = \mathcal{S} \in$ Gramm!

WHY MATTERS? E.G. Training LS-SVM (least-square support vector machine)

Involves computing inverses like $S^{-1}|\vec{5}\rangle$ (actually pseudoinverses...)

FOR THIS USE:

1) Q-database to load superpositions of all amplitude encoded states + trace-out encodes GRAM Matrix in Quantum density matrix

2) Density matrix exponentiation to "load" GRAM-matrix-state into unitary U_G

3) Quantum Linear algebra on U_G allows A LOT of stuff in polylog...

WARNING: A LOT OF FINE-PRINT COST.