For any $X \in \mathbb{R}^{n\times n}$ and $\epsilon \in (0,1]$. Then, with $r = \lceil 72\log(2n+1)/\epsilon^2 \rceil$, we have

$$\inf_{\operatorname{rank}(Y)\leq r} \|X - Y\|_{\max} \leq \epsilon \|X\|_2$$

$$\begin{bmatrix} 1 & 2 & 3 & 4 & 5 \\ 5 & 1 & 2 & 3 & 4 \\ 4 & 5 & 1 & 2 & 3 \\ 3 & 4 & 5 & 1 & 2 \\ 2 & 3 & 4 & 5 & 1 \end{bmatrix}$$

N charges: $\{q_i, r_i\}_{i=1}^N$

Potential: $\phi_i = \sum_{j \neq i} \frac{q_j}{\|r_i - r_j\|}$ for $i \in \{1, 2, \ldots, N\}$

Note that we have $\phi = Aq$, ϕ is the set of potentials, q is the set of charges and $A_{ij} = \frac{1}{\|r_i - r_j\|}$ for $i \neq j$

Matrix-vector product: Given q, compute ϕ. Cost is $\mathcal{O}(N^2)$

Solve linear system: Given ϕ, compute q. Cost is $\mathcal{O}(N^3)$

Can we reduce the above computational complexity to say $\mathcal{O}(N)$?

Let's look at a slightly easier problem

N charges: $\{q_i, r_i\}_{i=1}^N$

Potential: $\phi_i = \sum_{j=1}^{N} \sin\left(k\left(r_i - r_j\right)\right) q_j$ for $i \in \{1, 2, \ldots, N\}$

Matrix-vector product: Aq, where $A_{ij} = \sin\left(k\left(r_i - r_j\right)\right)$ for $i \neq j$

Computational cost is $\mathcal{O}(N^2)$

Can we reduce the computational complexity?

Note that $\sin\left(k\left(r_i - r_j\right)\right) = \sin\left(kr_i\right)\cos\left(kr_j\right) - \cos\left(kr_i\right)\sin\left(kr_j\right)$

$$\phi_i = \sum_{j=1}^{N} \sin\left(kr_i\right)\cos\left(kr_j\right) q_j - \sum_{j=1}^{N} \cos\left(kr_i\right)\sin\left(kr_j\right) q_j$$

for $i \in \{1, 2, \ldots, N\}$

$$\phi_i = \left(\sum_{j=1}^{N} \cos\left(kr_j\right) q_j\right) \sin\left(kr_i\right) - \left(\sum_{j=1}^{N} \sin\left(kr_j\right) q_j\right) \cos\left(kr_i\right)$$

for $i \in \{1, 2, \ldots, N\}$

Compute $a = \sum_{j=1}^{N} \cos\left(kr_j\right) q_j$; Cost is $\mathcal{O}(N)$

Compute $b = \sum_{j=1}^{N} \sin\left(kr_j\right) q_j$; Cost is $\mathcal{O}(N)$

$\phi_i = a \sin\left(kr_i\right) - b \cos\left(kr_i\right)$ for $i \in \{1, 2, \ldots, N\}$; Cost is $\mathcal{O}(N)$

Total cost is $\mathcal{O}(N)$

Wait... What happened?

How did we go from $\mathcal{O}(N^2)$ to $\mathcal{O}(N)$?

In matrix form, we rewrote the matrix A as

$$A = u_1 v_1^T - v_1 u_1^T$$

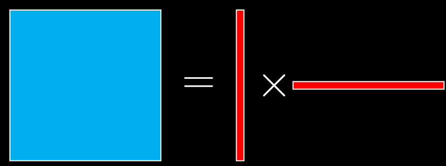
$$A = \begin{bmatrix} u_1 & -v_1 \end{bmatrix}_{N\times 2} \begin{bmatrix} v_1^T \\ u_1^T \end{bmatrix}_{2\times N}$$

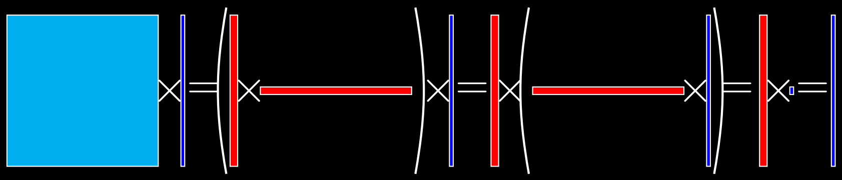
$$u_1 = \begin{bmatrix} \sin(kr_1) \\ \sin(kr_2) \\ \vdots \\ \sin(kr_N) \end{bmatrix}$$

$$v_1 = \begin{bmatrix} \cos(kr_1) \\ \cos(kr_2) \\ \vdots \\ \cos(kr_N) \end{bmatrix}$$

Hence,

$$Aq = u_1 \left(v_1^T q\right) - v_1 \left(u_1^T q\right)$$





Does the same idea work for $1/\left\|r_i - r_j\right\|$ instead of $\sin\left(k\left(r_i - r_j\right)\right)$?

Unfortunately, not.

The "kernel" (Green's function) $1/r$ has a singularity at the origin.

Let's look at other kernel functions; For instance, $\log(r)$

But what about away from the singularity?

Away from the singularity the matrix looks like low-rank (?)

Can this be proved?

Yes! The matrix is indeed low-rank in finite precision!

Away from the singularity, the kernel is smooth and a separable expansion of the kernel can be obtained

$$K\left(x, y\right) = \sum_{i,j=1}^{r} c_{ij}\phi_i(x)\psi_j(y) + \mathcal{O}\left(\epsilon\right)$$

where $r = \mathcal{O}\left(\log\left(1/\epsilon\right)\right)$ uniformly across the domain of x and y.

An appropriate series (Taylor or Eigen-function) expansion of the kernel into separable expansions

We could also rely on polynomial interpolation to construct separable expansions

The rank r is *independent* of N

Fast Multipole Method - souped version of the previous mentioned

- Relies on hierarchically sub-dividing your domain
- Identifying blocks to leverage low-rank matrix vector products

Total computational complexity is $\mathcal{O}(rN)$

Fast Multipole Method in Computational Physics

Matrices arising out of N-body problems have an even nicer structure