

Deck 10: Randomized least squares

Math 7870: Topics in Randomized Numerical Linear Algebra

Spring 2026

Akil Narayan

(Linear) Embeddings

Let $V \subset \mathbb{R}^n$ be an arbitrary set. A linear map (matrix) $\mathbf{S} : \mathbb{R}^n \rightarrow \mathbb{R}^d$ is called an ℓ^2 **embedding of distortion** $\epsilon \in (0, 1)$ on V if,

$$(1 - \epsilon)\|\mathbf{x}\|_2 \leq \|\mathbf{S}\mathbf{x}\|_2 \leq (1 + \epsilon)\|\mathbf{x}\|_2, \quad \mathbf{x} \in V.$$

We know how to generate \mathbf{S} randomly to achieve this. In particular:

$$d \gtrsim \frac{w(V)^2}{\epsilon^2}$$

is a sufficiently large embedding dimension for \mathbf{S} generated via:

- iid Gaussian entries
- a sparse sign model
- a subsampled randomized Fourier transform model

Now: what can we do with this?

Application 1: Randomized least squares

Here's the generic setup: Let $\mathbf{A} \in \mathbb{C}^{m \times n}$, $\mathbf{b} \in \mathbb{C}^m$ be given.

We assume $m \gg n$, and that \mathbf{A} has full (column) rank (for simplicity).

We seek the solution to,

$$\mathbf{x} = \arg \min_{\mathbf{y} \in \mathbb{C}^n} \|\mathbf{A}\mathbf{y} - \mathbf{b}\|_2^2.$$

Application 1: Randomized least squares

Here's the generic setup: Let $\mathbf{A} \in \mathbb{C}^{m \times n}$, $\mathbf{b} \in \mathbb{C}^m$ be given.

We assume $m \gg n$, and that \mathbf{A} has full (column) rank (for simplicity).

We seek the solution to,

$$\mathbf{x} = \arg \min_{\mathbf{y} \in \mathbb{C}^n} \|\mathbf{A}\mathbf{y} - \mathbf{b}\|_2^2.$$

The “standard” approach to solve this problem when \mathbf{A} can be stored in memory:

- (Thin QR decomposition) $\mathbf{A} = \mathbf{Q}\mathbf{R}$.
($\mathbf{Q} \in \mathbb{C}^{m \times n}$, $\mathbf{R} \in \mathbb{C}^{n \times n}$, $\mathbf{Q}^*\mathbf{Q} = \mathbf{I}_n$, \mathbf{R} upper triangular)
- Compute $\mathbf{x} = \mathbf{R}^{-1}\mathbf{Q}^*\mathbf{b}$.

This algorithm requires $\mathcal{O}(mn^2)$ complexity.

Strategy 1: sketch-and-solve, I

$$\|\mathbf{Ax} - \mathbf{b}\|_2 \leq \|\mathbf{Ay} - \mathbf{b}\|_2, \quad \mathbf{y} \in \mathbb{C}^n.$$

Here's the core idea for randomized least squares: Let \mathbf{S} be an ϵ -distortion embedding for V defined as,

$$V := \text{span} \{ \text{range}(\mathbf{A}), \mathbf{b} \}.$$

Terminology: the embedding matrix \mathbf{S} is also called a (linear) *sketch*.

Strategy 1: sketch-and-solve, I

$$\|\mathbf{Ax} - \mathbf{b}\|_2 \leq \|\mathbf{Ay} - \mathbf{b}\|_2, \quad \mathbf{y} \in \mathbb{C}^n.$$

Here's the core idea for randomized least squares: Let \mathbf{S} be an ϵ -distortion embedding for V defined as,

$$V := \text{span} \{ \text{range}(\mathbf{A}), \mathbf{b} \}.$$

Terminology: the embedding matrix \mathbf{S} is also called a (linear) *sketch*.

Now consider the *sketched* least squares problem:

$$\mathbf{z} = \arg \min_{\mathbf{y} \in \mathbb{C}^n} \|\mathbf{SAy} - \mathbf{Sb}\|_2^2.$$

The solution \mathbf{z} is a near-optimal solution, essentially by definition:

$$\left. \begin{aligned} (1 - \epsilon) \|\mathbf{Az} - \mathbf{b}\|_2^2 &\leq \|\mathbf{SAz} - \mathbf{Sb}\|_2^2 \\ &\leq \|\mathbf{SAx} - \mathbf{Sb}\|_2^2 \\ &\leq (1 + \epsilon) \|\mathbf{Ax} - \mathbf{b}\|_2^2 \end{aligned} \right\} \Rightarrow \|\mathbf{Az} - \mathbf{b}\|_2^2 \leq \frac{1+\epsilon}{1-\epsilon} \|\mathbf{Ax} - \mathbf{b}\|_2^2$$

Strategy 1: sketch-and-solve, II

$$\mathbf{z} = \arg \min_{\mathbf{y} \in \mathbb{C}^n} \|\mathbf{SAy} - \mathbf{Sb}\|_2^2, \quad \|\mathbf{Az} - \mathbf{b}\|_2 \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}} \|\mathbf{Az} - \mathbf{b}\|_2.$$

Note that computing \mathbf{z} directly by a QR decomposition requires only $\mathcal{O}(dn^2)$ effort, with $d \ll m$.

However, direct computation of the sketched system requires $\mathcal{O}(mdn)$ complexity. Since $d \gtrsim \dim V \sim n$, we haven't gained anything computationally.

Strategy 1: sketch-and-solve, II

$$\mathbf{z} = \arg \min_{\mathbf{y} \in \mathbb{C}^n} \|\mathbf{SAy} - \mathbf{Sb}\|_2^2, \quad \|\mathbf{Az} - \mathbf{b}\|_2 \leq \sqrt{\frac{1+\epsilon}{1-\epsilon}} \|\mathbf{Az} - \mathbf{b}\|_2.$$

Note that computing \mathbf{z} directly by a QR decomposition requires only $\mathcal{O}(dn^2)$ effort, with $d \ll m$.

However, direct computation of the sketched system requires $\mathcal{O}(mdn)$ complexity. Since $d \gtrsim \dim V \sim n$, we haven't gained anything computationally.

The saving grace is that we can apply FFT-based sketches much faster:

If \mathbf{S} is a subsampled random Fourier transform-based sketch, we require $d \gtrsim (n + \log m) \log n$, which achieves an ϵ -embedding for fixed $\epsilon > 0$.

This FFT-based sketch can be applied in $\mathcal{O}(mn \log d)$ complexity.

Therefore, the full cost of a subsampled randomized Fourier transform-based sketch-and-solve procedure is:

- $\mathcal{O}(mn \log d)$ complexity to generate and apply the sketch \mathbf{S} to \mathbf{A} and \mathbf{b} .
- $\mathcal{O}(dn^2)$ complexity to directly solve the sketched system.

Total $\mathcal{O}(mn \log d + dn^2)$ complexity.

Cf.: Direct least squares is $\mathcal{O}(mn^2)$. Significant savings when $d \sim n \ll m$.

Strategy 2: iterative sketching, I

Before we use embeddings to sketch, here's the deterministic idea: Notice first that the minimizer of $\|\mathbf{A}\mathbf{y} - \mathbf{b}\|_2^2$ is the same as the minimizer of,

$$f(\mathbf{y}) = \frac{1}{2}\mathbf{y}^* \mathbf{A}^* \mathbf{A} \mathbf{y} - (\mathbf{b}^* \mathbf{A})\mathbf{y}.$$
$$\mathbf{x} = \arg \min_{\mathbf{y}} f(\mathbf{y})$$

(In addition, the stationary point of f is the solution to the normal equations.)

Strategy 2: iterative sketching, I

Before we use embeddings to sketch, here's the deterministic idea: Notice first that the minimizer of $\|\mathbf{A}\mathbf{y} - \mathbf{b}\|_2^2$ is the same as the minimizer of,

$$f(\mathbf{y}) = \frac{1}{2}\mathbf{y}^* \mathbf{A}^* \mathbf{A} \mathbf{y} - (\mathbf{b}^* \mathbf{A})\mathbf{y}.$$
$$\mathbf{x} = \arg \min_{\mathbf{y}} f(\mathbf{y})$$

(In addition, the stationary point of f is the solution to the normal equations.)

One key realization from this is that we can make this process iterative simply by changing \mathbf{b} above.

Suppose we have an approximation \mathbf{x} : $\mathbf{x} \approx \mathbf{y}$. Writing $\mathbf{x} = \mathbf{w} + \mathbf{y}$, the exact solution is given by,

$$\min_{\mathbf{v}} \|\mathbf{A}(\mathbf{v} + \mathbf{y}) - \mathbf{b}\|_2^2 = \min_{\mathbf{v}} \|\mathbf{A}\mathbf{v} - (\mathbf{b} - \mathbf{A}\mathbf{y})\|_2^2,$$

i.e.,:

$$\mathbf{w} = \arg \min_{\mathbf{v}} \frac{1}{2}\mathbf{v}^* \mathbf{A}^* \mathbf{A} \mathbf{v} - ((\mathbf{b} - \mathbf{A}\mathbf{y})^* \mathbf{A}) \mathbf{v}.$$

Strategy 2: iterative sketching, II

To make this iterative, we presume that each minimization is performed approximately. For $k \geq 1$:

$$\mathbf{x}_0 = \mathbf{0}, \mathbf{w}_k \approx \arg \min_{\mathbf{v}} \frac{1}{2} \mathbf{v}^* \mathbf{A}^* \mathbf{A} \mathbf{v} - ((\mathbf{b} - \mathbf{A} \mathbf{x}_{k-1})^* \mathbf{A}) \mathbf{v}$$
$$\mathbf{x}_k \mathbf{x}_{k-1} + \mathbf{w}_k.$$

Why should we solve the minimization step approximately? This is the expensive part, requiring $\mathcal{O}(mn^2)$ complexity.

Strategy 2: iterative sketching, II

To make this iterative, we presume that each minimization is performed approximately. For $k \geq 1$:

$$\begin{aligned} \mathbf{x}_0 = \mathbf{0}, \mathbf{w}_k & \approx \arg \min_{\mathbf{v}} \frac{1}{2} \mathbf{v}^* \mathbf{A}^* \mathbf{A} \mathbf{v} - ((\mathbf{b} - \mathbf{A} \mathbf{x}_{k-1})^* \mathbf{A}) \mathbf{v} \\ \mathbf{x}_k & \mathbf{x}_{k-1} + \mathbf{w}_k. \end{aligned}$$

Why should we solve the minimization step approximately? This is the expensive part, requiring $\mathcal{O}(mn^2)$ complexity.

The core improvement: we can *approximately* solve this equation via sketching. We need only sketch the quadratic term:

$$\mathbf{w}_k = \arg \min_{\mathbf{v}} \frac{1}{2} \mathbf{v}^* \mathbf{A}^* \mathbf{S}^* \mathbf{S} \mathbf{A} \mathbf{v} - ((\mathbf{b} - \mathbf{A} \mathbf{x}_{k-1})^* \mathbf{A}) \mathbf{v}$$

Since we're only sketching \mathbf{A} , the embedding \mathbf{S} needs to be an embedding only for $\text{range}(\mathbf{A})$.

Strategy 2: iterative sketching, III

As before, we need to sketch with a subsampled randomized Fourier transform to yield asymptotic computational gains.

Without presenting the details of the analysis:

- To achieve ϵ -proximity to the solution, strategy 1 (sketch-and-solve) requires $d \sim (n \log n)\epsilon^{-2}$. This strategy, iterative sketching, can circumvent this.

Strategy 2: iterative sketching, III

As before, we need to sketch with a subsampled randomized Fourier transform to yield asymptotic computational gains.

Without presenting the details of the analysis:

- To achieve ϵ -proximity to the solution, strategy 1 (sketch-and-solve) requires $d \sim (n \log n)\epsilon^{-2}$. This strategy, iterative sketching, can circumvent this.
- We need only take $d \sim Cn \log n$ for a universal C here.
- Through an inductive argument: the iterative scheme is linearly convergent, so the error decays like c^k after k iterations, where $c < 1$.
- To achieve ϵ -proximity to the solution, the number of *iterations* must be $\mathcal{O}(D/\epsilon)$, where D is the “dynamic range”, i.e., the relative amount of \mathbf{b} that lies outside $\text{range}(\mathbf{A})$:

$$D = \frac{\|\mathbf{P}\mathbf{b}\|_2}{\|\mathbf{b} - \mathbf{P}\mathbf{b}\|_2}, \quad \mathbf{P} = \mathbf{Q}\mathbf{Q}^*, \quad \mathbf{A} = \mathbf{Q}\mathbf{R}.$$

- Overall, we need $\mathcal{O}(mn \log n)$ complexity to apply a sketch, $\mathcal{O}(dn^2)$ to directly solve a sketched least squares system, and $\mathcal{O}(mn)$ complexity for $\log(D/\epsilon)$ iterations to update the “ \mathbf{b} ” vector in f .