Random Matrix Theory for Machine Learning

Part 3: Analysis of numerical algorithms

Fabian Pedregosa[1], Courtney Paquette[1,2], Tom Trogdon[3], Jeffrey Pennington[1]

[1] Google Research, [2] McGill University, [3] University of Washington

https://random-matrix-learning.github.io

Notation

In this section:

- A is typically a rectangular matrix with more rows than columns
- W is a symmetric (square) matrix
- Often $W \propto A^T A$

Motivation: Average-case versus worst-case in high dimensions

Issues with worst-case bounds in high dimensions

In some very specific cases, the high-dimensionality of a given problem provides it with enough degrees of freedom to “conspire against” a given algorithm.

Issues with worst-case bounds in high dimensions

In some very specific cases, the high-dimensionality of a given problem provides it with enough degrees of freedom to "conspire against" a given algorithm.

For, example, consider solving a $n \times n$ linear system $\boldsymbol{W}\boldsymbol{x} = \boldsymbol{b}$ using the conjugate gradient (CG) algorithm where

$$W = \begin{bmatrix} \mathfrak{r} & \sqrt{\mathfrak{r}} & & & & & \\ \sqrt{\mathfrak{r}} & 1+\mathfrak{r} & \sqrt{\mathfrak{r}} & & & & \\ & \sqrt{\mathfrak{r}} & 1+\mathfrak{r} & \sqrt{\mathfrak{r}} & & & \\ & & \sqrt{\mathfrak{r}} & \ddots & \ddots & & \\ & & & \ddots & & & \\ & & & & & \sqrt{\mathfrak{r}} & \\ & & & & \sqrt{\mathfrak{r}} & 1+\mathfrak{r} & \end{bmatrix}, \quad b = \begin{bmatrix} 1 \\ 0 \\ \vdots \\ 0 \end{bmatrix}, \quad 0 < \mathfrak{r} < 1.$$

Issues with worst-case bounds in high dimensions

The CG algorithm is iterative and produces approximations x_k that satisfy:

$$x_k = \underset{y \in \mathbb{R}^n}{\arg\min} \left\{ (x-y)^T W (x-y)^T : y \in \operatorname{span}\{b, Wb, \ldots, W^{k-1}b\} \right\}.$$

Issues with worst-case bounds in high dimensions

The CG algorithm is iterative and produces approximations x_k that satisfy:

$$x_k = \underset{y \in \mathbb{R}^n}{\arg\min} \left\{ (x-y)^T W (x-y)^T : y \in \operatorname{span}\{b, Wb, \ldots, W^{k-1}b\} \right\}.$$

It can be shown that for the above choice of W, b, and $1 \leq k < n$

$$\|b - Wx_k\|^2 = \left(\frac{1}{\mathfrak{r}}\right)^k \quad \text{but} \quad \|b - Wx_n\| = 0.$$

Issues with worst-case bounds in high dimensions

The CG algorithm is iterative and produces approximations x_k that satisfy:

$$x_k = \underset{y \in \mathbb{R}^n}{\arg\min} \left\{ (x-y)^T W (x-y)^T : y \in \operatorname{span}\{b, Wb, \ldots, W^{k-1}b\} \right\}.$$

It can be shown that for the above choice of W, b, and $1 \leq k < n$

$$\|b - Wx_k\|^2 = \left(\frac{1}{\mathfrak{r}}\right)^k \quad \text{but} \quad \|b - Wx_n\| = 0.$$

The residuals (or norms of the gradient) appear to diverge exponentially before the iteration finally converges!

Issues with worst-case bounds in high dimensions

The CG algorithm is iterative and produces approximations x_k that satisfy:

$$x_k = \underset{y \in \mathbb{R}^n}{\arg\min} \left\{ (x-y)^T W (x-y)^T : y \in \operatorname{span}\{b, Wb, \ldots, W^{k-1}b\} \right\}.$$

It can be shown that for the above choice of W, b, and $1 \leq k < n$

$$\|b - Wx_k\|^2 = \left(\frac{1}{\mathfrak{r}}\right)^k \quad \text{but} \quad \|b - Wx_n\| = 0.$$

The residuals (or norms of the gradient) appear to diverge exponentially before the iteration finally converges!

And as n increases, this becomes worse. And a worst-case bound needs to account for this pathological example.

Instead, we may want to choose W and b to be random and consider

$$\mathbb{E}\|b - Wx_k\|^2.$$

If one chooses W to be distributed according to the Wishart distribution, as $n \to \infty$,

$$\mathbb{E}\|b - Wx_k\|^2 = \mathfrak{r}^k + o(1), \quad \mathfrak{r} = r^{-1} = \lim_{n\to\infty} \frac{n}{d}.$$

Instead, we may want to choose W and b to be random and consider

$$\mathbb{E}\|b - Wx_k\|^2.$$

If one chooses W to be distributed according to the Wishart distribution, as $n \to \infty$,

$$\mathbb{E}\|b - Wx_k\|^2 = \mathfrak{r}^k + o(1), \quad \mathfrak{r} = r^{-1} = \lim_{n\to\infty} \frac{n}{d}.$$

But a valid important open problem is: To model optimization in a ML context, what distribution is relevant for W?

This is an open problem. See Liao and Mahoney [2021] for work in this direction.

Main RMT tool: Matrix moments

Main linear algebra tool

Recall Cauchy's integral formula: If f is analytic in a sufficiently large region and $\mathcal{C}$ is smooth, simple, closed curve then

$$f(z) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} \frac{f(z')}{z' - z} \, \mathrm{d}z'.$$

Main linear algebra tool

Recall Cauchy's integral formula: If f is analytic in a sufficiently large region and $\mathcal{C}$ is smooth, simple, closed curve then

$$f(z) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} \frac{f(z')}{z' - z} \, \mathrm{d}z'.$$

Suppose the eigenvalues of an $n \times n$ matrix W are enclosed by $\mathcal{C}$, then

$$f(W) := U f(\Lambda) U^{-1} = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} f(z) (z I_n - W)^{-1} \, \mathrm{d}z.$$

Recall Cauchy's integral formula: If f is analytic in a sufficiently large region and $\mathcal{C}$ is smooth, simple, closed curve then

$$f(z) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} \frac{f(z')}{z' - z} \, \mathrm{d}z'.$$

Suppose the eigenvalues of an $n \times n$ matrix W are enclosed by $\mathcal{C}$, then

$$f(W) := U f(\Lambda) U^{-1} = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} f(z) (z I_n - W)^{-1} \, \mathrm{d}z.$$

In particular,

$$W^k = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z I_n - W)^{-1} \, \mathrm{d}z.$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{2\pi n \mathrm{i}} \int_{\mathcal{C}} z^k \operatorname{tr}\left(z I_n - W\right)^{-1} \mathrm{d}z$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_C z^k (z-\lambda_j)^{-1}\,\mathrm{d}z$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z-\lambda_j)^{-1}\,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ n^{-1}\sum_j \delta_{\lambda_j}}} \mathrm{d}z$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z-\lambda_j)^{-1}\,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ n^{-1}\sum_j \delta_{\lambda_j}}} \mathrm{d}z$$

$$u^T W^k u =$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z - \lambda_j)^{-1} \,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ n^{-1}\sum_j \delta_{\lambda_j}}} \,\mathrm{d}z$$

$$u^T W^k u = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k u^T (z I_n - W)^{-1} u \,\mathrm{d}z$$

Two consequences

$$\frac{1}{n}\operatorname{tr} W^k = \frac{1}{n}\sum_{j=1}^{n} \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k (z-\lambda_j)^{-1}\,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ n^{-1}\sum_j \delta_{\lambda_j}}} \mathrm{d}z$$

$$u^T W^k u = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k u^T (zI_n - W)^{-1} u\,\mathrm{d}z = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k \underbrace{m_{\mathrm{ESD}}(z)}_{\substack{\text{Stieltjes transform of} \\ \sum_j w_j \delta_{\lambda_j}}} \mathrm{d}z$$

$$\sum_j w_j \delta_{\lambda_j}, \quad w_j = (v_j^T u)^2$$

Analysis of matrix moments

A main RMT takeaway:

Matrix moments $\Leftrightarrow$ Classical moments of ESD $\Leftrightarrow$ Contour integrals of Stieltjes transform

A main RMT takeaway:

$$\tfrac{1}{n}\operatorname{tr} W^k \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

A main RMT takeaway:

$$\tfrac{1}{n}\operatorname{tr} W^k = \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

If $\mu_{\mathrm{ESD}} = \frac{1}{n}\sum_{j=1}^{n} \delta_{\lambda_j(W)}$ then the first $\approx$ becomes $=$.

A main RMT takeaway:

$$\tfrac{1}{n}\operatorname{tr} W^k \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

If μ_{ESD} is the limiting ESD then the second $\approx$ becomes $=$.

Analysis of matrix moments

A main RMT takeaway:

$$\tfrac{1}{n}\operatorname{tr} W^k \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

If μ_{ESD} is the limiting ESD then errors are typically on the order of $1/n$.

Analysis of matrix moments

A main RMT takeaway:

$$u^T W^k u \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\, \mathrm{d}z$$

A main RMT takeaway:

$$u^T W^k u = \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z$$

If $\mu_{\mathrm{ESD}} = \sum_{j=1}^{n} w_j \delta_{\lambda_j(W)}$, $w_j = (v_j^T u)^2$ for eigenvectors v_j then the first $\approx$ becomes $=$.

A main RMT takeaway:

$$u^T W^k u \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) = \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\, \mathrm{d}z$$

If μ_{ESD} is the limiting ESD then errors are typically on the order of $1/\sqrt{n}$.

A main RMT takeaway:

$$u^T W^k u \approx \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\, \mathrm{d}z$$

If a statistic, (which might be the error encountered in, or the runtime of, an algorithm) depends strongly on these generalized moments, it may be analyzable directly using RMT.

A main RMT takeaway:

$$u^T W^k u = \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z \quad \mu_{\mathrm{ESD}} = \sum_{j=1}^{d} w_j \delta_{\lambda_j(W)}$$

Theorem (Knowles and Yin [2017])

For a large class of sample covariance matrices W there exists a deterministic measure μ_{SCM} with Stieltjes transform m_{SCM} such that

$$\Pr\left(\left|a^T(W - zI_n)^{-1}b - (a^T b)m_{\mathrm{SCM}}(z)\right| \geq \|a\|\|b\|t\right) = O(n^{-D})$$

for any $D > 0$, uniformly in a large subset of the complex plane.

A main RMT takeaway:

$$u^T W^k u = \int_{\mathbb{R}} x^k \mu_{\mathrm{ESD}}(\mathrm{d}x) \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z \quad \mu_{\mathrm{ESD}} = \sum_{j=1}^{d} w_j \delta_{\lambda_j(W)}$$

Theorem (Knowles and Yin [2017])

For a large class of sample covariance matrices W there exists a deterministic measure μ_{SCM} with Stieltjes transform m_{SCM} such that

$$\Pr\left(|m_{\mathrm{ESD}}(z) - m_{\mathrm{SCM}}(z)| \geq t\right) = O(n^{-D})$$

for any $D > 0$, uniformly in a large subset of the complex plane.

Resolvent estimates lead to moment estimates

$$u^T W^k u \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{SCM}}(z)\,\mathrm{d}z = \int_{\mathbb{R}} x^k \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

Resolvent estimates lead to moment estimates

$$u^T W^k u \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{ESD}}(z)\,\mathrm{d}z \approx \frac{1}{2\pi \mathrm{i}} \int_{\mathcal{C}} z^k m_{\mathrm{SCM}}(z)\,\mathrm{d}z = \int_{\mathbb{R}} x^k \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

$$u^T P(W) u \approx \int_{\mathbb{R}} P(x) \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

Algorithm halting times (runtimes)

Statistics of algorithm runtimes

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

Statistics of algorithm runtimes

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

- Let $\mathcal{A}$ be an iterative algorithm to solve a problem P_n (e.g., an $n \times n$ linear system with gradient descent).

Statistics of algorithm runtimes

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

- Let $\mathcal{A}$ be an iterative algorithm to solve a problem P_n (e.g., an $n \times n$ linear system with gradient descent).
- Included with the algorithm, we assume there is a measure of error $E_k(P_n; \mathcal{A})$ at iteration k (e.g., norm of the gradient).

Statistics of algorithm runtimes

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

- Let $\mathcal{A}$ be an iterative algorithm to solve a problem P_n (e.g., an $n \times n$ linear system with gradient descent).
- Included with the algorithm, we assume there is a measure of error $E_k(P_n; \mathcal{A})$ at iteration k (e.g., norm of the gradient).
- Let $\mathcal{E}$ represent a distribution from which problems P_n are drawn (e.g., a random matrix and vector for a linear system).

Our abstract setup to analyze algorithms is as follows. Suppose first that there is a intrinsic notion of dimension n.

- Let $\mathcal{A}$ be an iterative algorithm to solve a problem P_n (e.g., an $n \times n$ linear system with gradient descent).
- Included with the algorithm, we assume there is a measure of error $E_k(P_n; \mathcal{A})$ at iteration k (e.g., norm of the gradient).
- Let $\mathcal{E}$ represent a distribution from which problems P_n are drawn (e.g., a random matrix and vector for a linear system).
- The halting time is then defined as

$$T_{\mathcal{A}}(P_n, \varepsilon) = \min\{k : E_k(P_n; \mathcal{A}) < \varepsilon\}.$$

Some history

Probably the most famous, and maybe the most influential, instance of the probabilistic analysis of an algorithm, was the analysis of the simplex algorithm developed by Dantzig [1951] for linear programming (see also Dantzig [1990]).

Some history

Probably the most famous, and maybe the most influential, instance of the probabilistic analysis of an algorithm, was the analysis of the simplex algorithm developed by Dantzig [1951] for linear programming (see also Dantzig [1990]).

For many years after its inception the simplex method had no provable complexity guarantees. Indeed, with a fixed pivot rule, there typically exists a problem on which the simplex method takes an exponentially large number of steps.

Some history

Probably the most famous, and maybe the most influential, instance of the probabilistic analysis of an algorithm, was the analysis of the simplex algorithm developed by Dantzig [1951] for linear programming (see also Dantzig [1990]).

For many years after its inception the simplex method had no provable complexity guarantees. Indeed, with a fixed pivot rule, there typically exists a problem on which the simplex method takes an exponentially large number of steps.

Despite the existence of other algorithms for linear programming with provable polynomial runtime guarantees, the simplex method persisted as the most widely used algorithm.

Some history

Borgwardt [1987] and, independently, Smale [1983] proved that under certain probabilistic assumptions and under certain pivot rules, the expected runtime of the simplex algorithm is polynomial:

$$\mathbb{E}T_{\text{Simplex}}(P_n; \varepsilon) \leq \text{polynomial in } n.$$

Some history

Borgwardt [1987] and, independently, Smale [1983] proved that under certain probabilistic assumptions and under certain pivot rules, the expected runtime of the simplex algorithm is polynomial:

$$\mathbb{E}T_{\text{Simplex}}(P_n; \varepsilon) \leq \text{polynomial in } n.$$

Limited only by their statistical assumptions, these analyses demonstrated, at least conceptually, why the simplex algorithm typically behaves well and is efficient.

Some history

Borgwardt [1987] and, independently, Smale [1983] proved that under certain probabilistic assumptions and under certain pivot rules, the expected runtime of the simplex algorithm is polynomial:

$$\mathbb{E}T_{\text{Simplex}}(P_n; \varepsilon) \leq \text{polynomial in } n.$$

Limited only by their statistical assumptions, these analyses demonstrated, at least conceptually, why the simplex algorithm typically behaves well and is efficient.

The subsequent analysis by Spielman and Teng [2004] improved these analyses by providing estimates for randomly perturbed linear programs. This analysis has since been improved, see [Dadush and Huiberts [2020], Vershynin [2009], Deshpande and Spielman [2005]], for example.

This highlights something we will face here: While we will go through the precise average case analysis of some optimization algorithms, one can always take issue with the underlying statistical assumptions we make.

This highlights something we will face here: While we will go through the precise average case analysis of some optimization algorithms, one can always take issue with the underlying statistical assumptions we make.

For any average-case analysis one hopes to continue to:

- Expand the class of distributions that can be considered.
- Increase the precision of the resulting estimates.
- Collect additional algorithms that can be analyzed with the same or similar techniques.

Some history

We also highlight two other success stories in average-case analysis. These are of a different flavor because randomization is introduced to algorithms to improve their performance. And subsequently, one has a natural distribution over which to compute averages, but the problem being solved is deterministic.

We also highlight two other success stories in average-case analysis. These are of a different flavor because randomization is introduced to algorithms to improve their performance. And subsequently, one has a natural distribution over which to compute averages, but the problem being solved is deterministic.

The first algorithm is the power method with randomized starting. The power method is an algorithm to compute the dominant eigenvalue (provided it exists) of a matrix. It also also approximates the dominant eigenvector.

The power method

1. x_0 is the initial vector, $\|x_0\| = 1$ and W is given.
2. For $k = 1, 2, \ldots$
   - 2.1 Compute $v_k = Wx_{k-1}$
   - 2.2 Compute $\mu_k = v_k^T x_{k-1}$
   - 2.3 Compute $x_k = v_k / \|v_k\|$

The iterates μ_k, under mild assumptions, will converge to the dominant eigenvalue of W. It is well-known that the power method will converge at a exponential rate depending on the ratio of the largest-to-next-largest eigenvalue (a relative spectral gap).

If W is positive semi-definite and x_0 is chosen randomly ($x_0 = \texttt{np.random.randn(n)}$, $x_0 \leftarrow x_0/\|x_0\|$), then it was shown in Kuczyński and Woźniakowski [1992] that a spectral gap is not need to get average-case error bounds of the form:

$$\mathbb{E} \underbrace{\frac{|\mu_k - \lambda_{\max}|}{|\lambda_{\max} - \lambda_{\min}|}}_{E_k(P_n;\mathrm{Power})} \leq 0.871 \frac{\log n}{k-1}.$$

The power method can also be analyzed on random matrices, see Kostlan [1988], Deift and Trogdon [2017].

Lastly, a discussion that is closer to the heart of the matter is the work of Strohmer and Vershynin [2009] on the randomized version of the original Kaczmarz algorithm [Kaczmarz [1937]] for the solution of overdetermined linear systems.

The Kaczmarz Algorithm

1. x_0 is the initial vector and A is given.
2. For $k = 1, 2, \ldots$
   2.1 Select a row a_j of A (add randomness here!)
   2.2 Compute $x_k = x_{k-1} - \frac{b_j - a_j^T x_{k-1}}{\|a_j\|^2} a_j$

For a consistent overdetermined system $Ax = b$ it was shown that the method satisfies

$$\mathbb{E} \underbrace{\|x_k - x\|^2}_{E_k(P_n;\mathrm{Kaczmarz})} \leq \left(1 - \frac{1}{\kappa(A^TA)}\right)^k \|x_0 - x\|^2$$

where $\kappa(A^TA)$ is the condition number of A^TA (to be discussed more later).

The power of random matrix theory (RMT) is that one can ask and answer more involved questions:

- If P_n is drawn randomly, then

$$T_{\mathcal{A}}(P_n, \varepsilon)$$

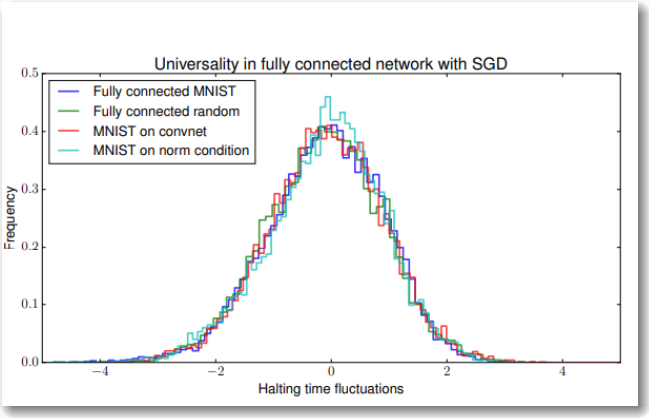
  is an integer-valued random variable. While it is important bound its expectation or moments, what about its distribution as $n \to \infty$?

- With the same considerations

$$E_k(P_n; \mathcal{A})$$

  is a random variable. Can we understand its distribution?

Universality of the halting time



Sagun et al. [2017] present experiments to demonstrate that the halting time $T_{\mathrm{SGD}}(P_n; \varepsilon)$ for a number of neural network architectures exhibits universality. That is, after proper centering and rescaling, the resulting statistics do not depend (in the limit) on the distribution on P_n.

A wide variety of numerical algorithms have been demonstrated (both empirically and rigorously) to have universal halting times (i.e., runtimes, iteration counts, etc.). The study of universality in halting time was initiated by Pfrang et al. [2014] and broaded in Deift et al. [2014].

Universality in halting time is the statement that for a given $\mathcal{A}$, and a wide class of ensembles $\mathcal{E}$, there are constants $\mu = \mu(\mathcal{E}, \varepsilon, n)$ and $\sigma = \sigma(\mathcal{E}, \varepsilon, n)$ and $\varepsilon = \varepsilon(\mathcal{E}, n)$ such that

$$\lim_{n\to\infty} \mathbb{P}_{\mathcal{E}} \left(\frac{T_{\mathcal{A}}(P_n, \varepsilon) - \mu}{\sigma} \leq t \right) = F_{\mathcal{A}}(t).$$

A wide variety of numerical algorithms have been demonstrated (both empirically and rigorously) to have universal halting times (i.e., runtimes, iteration counts, etc.). The study of universality in halting time was initiated by Pfrang et al. [2014] and broaded in Deift et al. [2014].

Universality in halting time is the statement that for a given $\mathcal{A}$, and a wide class of ensembles $\mathcal{E}$, there are constants $\mu = \mu(\mathcal{E}, \varepsilon, n)$ and $\sigma = \sigma(\mathcal{E}, \varepsilon, n)$ and $\varepsilon = \varepsilon(\mathcal{E}, n)$ such that

$$\lim_{n\to\infty} \mathbb{P}_{\mathcal{E}} \left(\frac{T_{\mathcal{A}}(P_n, \varepsilon) - \mu}{\sigma} \leq t \right) = F_{\mathcal{A}}(t).$$

The limiting distribution is independent of the choice for $\mathcal{E}$.

A case study: Regression

A natural first step

A natural first place to combine RMT and optimization/ML with a view toward universality is in the study of linear regression:

$$\underset{x\in\mathbb{R}^n}{\arg\min}\left\{\mathcal{L}(x) := \frac{1}{2d}\|Ax-b\|^2, \quad b = Aa+\eta\right\},$$

where a is the signal, η is additive noise, and

$$\boxed{A \text{ is a } d\times n \text{ matrix}}.$$

A natural first step

A natural first place to combine RMT and optimization/ML with a view toward universality is in the study of linear regression:

$$\arg\min_{x\in\mathbb{R}^n} \left\{ \mathcal{L}(x) := \frac{1}{2d}\|Ax-b\|^2, \quad b = Aa+\eta \right\},$$

where a is the signal, η is additive noise, and

$$\boxed{A \text{ is a } d \times n \text{ matrix}}.$$

There are, of course, many iterative algorithms to solve this problem and we focus on two:

1. the conjugate gradient algorithm (CG) [Hestenes and Steifel [1952]], and
2. the gradient descent algorithm (GD).

The Conjugate Gradient Algorithm

1. x_0 is the initial guess.
2. Set $r_0 = A^T b - A^T A x_0$, $p_0 = r_0$.
3. For $k = 1, 2, \ldots, n$
   - 3.1 Compute $a_{k-1} = \dfrac{r_{k-1}^* r_{k-1}}{r_{k-1}^* A^T A p_{k-1}}$.
   - 3.2 Set $x_k = x_{k-1} + a_{k-1} p_{k-1}$.
   - 3.3 Set $r_k = r_{k-1} - a_{k-1} A^T A p_{k-1}$.
   - 3.4 Compute $b_{k-1} = -\dfrac{r_k^* r_k}{r_{k-1}^* r_{k-1}}$.
   - 3.5 Set $p_k = r_k - b_{k-1} p_{k-1}$.

A natural first step

Why consider CG?

A natural first step

Why consider CG?

CG is a highly-structured algorithm with connections to the Lanczos iteration and the theory of orthogonal polynomials. While we do not discuss many of these details, they play an important role in the analysis. CG is also a method-of-choice in the broader computational mathematics community.

A natural first step

Why consider CG?

CG is a highly-structured algorithm with connections to the Lanczos iteration and the theory of orthogonal polynomials. While we do not discuss many of these details, they play an important role in the analysis. CG is also a method-of-choice in the broader computational mathematics community.

A simplification.

A natural first step

Why consider CG?

CG is a highly-structured algorithm with connections to the Lanczos iteration and the theory of orthogonal polynomials. While we do not discuss many of these details, they play an important role in the analysis. CG is also a method-of-choice in the broader computational mathematics community.

A simplification.

Instead of considering CG on the normal equations, $A^T A x = A^T b$, we first consider a slightly simpler problem:

$$\text{CG applied to } A^T A x = c.$$

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$
- $d = n$
- $d = \lfloor n + cn^{\alpha} \rfloor$ for $0 < \alpha < 1$

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$
- $d = n$
- $d = \lfloor n + cn^{\alpha} \rfloor$ for $0 < \alpha < 1$

Recall that condition number of a matrix W is defined to be

$$\kappa(W) = \frac{\sigma_1(W)}{\sigma_n(W)},$$

i.e., the ratio of the largest to the smallest singular value of W.

Matrices with small condition numbers are said to be well conditioned while those with larger condition numbers are said to be ill conditioned.

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$ (well conditioned)
- $d = n$
- $d = \lfloor n + cn^\alpha \rfloor$ for $0 < \alpha < 1$

Recall that condition number of a matrix W is defined to be

$$\kappa(W) = \frac{\sigma_1(W)}{\sigma_n(W)},$$

i.e., the ratio of the largest to the smallest singular value of W.

Matrices with small condition numbers are said to be well conditioned while those with larger condition numbers are said to be ill conditioned.

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$ (well conditioned)
- $d = n$ (ill conditioned, but still invertible)
- $d = \lfloor n + cn^\alpha \rfloor$ for $0 < \alpha < 1$

Recall that condition number of a matrix W is defined to be

$$\kappa(W) = \frac{\sigma_1(W)}{\sigma_n(W)},$$

i.e., the ratio of the largest to the smallest singular value of W.

Matrices with small condition numbers are said to be well conditioned while those with larger condition numbers are said to be ill conditioned.

Scaling regions

Scaling regions show up here in the relationship between n and d in a sample covariance matrix $W = \frac{A^T A}{d}$ (A is $d \times n$).

Scalings of sample covariance matrices

- $d = \lfloor nr \rfloor$ for $r > 1$ (well conditioned)
- $d = n$ (ill conditioned, but still invertible)
- $d = \lfloor n + cn^\alpha \rfloor$ for $0 < \alpha < 1$ (somewhere in between)

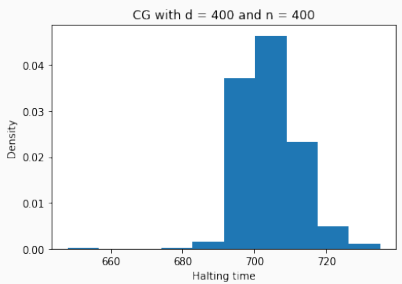
Recall that condition number of a matrix W is defined to be

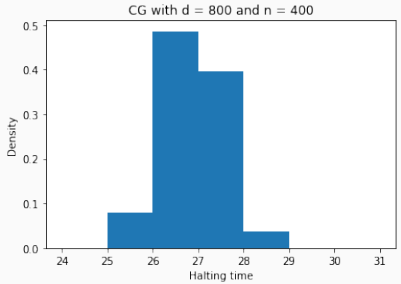
$$\kappa(W) = \frac{\sigma_1(W)}{\sigma_n(W)},$$

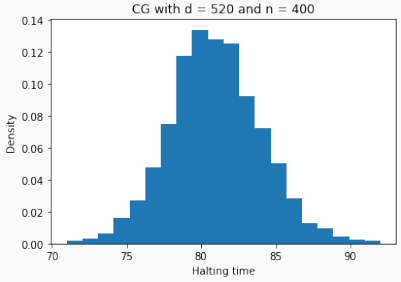
i.e., the ratio of the largest to the smallest singular value of W.

Matrices with small condition numbers are said to be well conditioned while those with larger condition numbers are said to be ill conditioned.

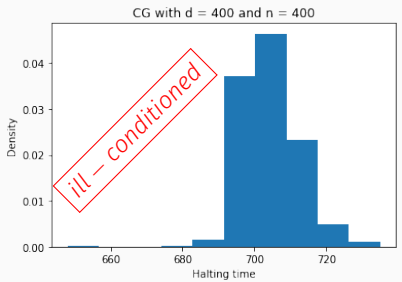
Three distinct behaviors of CG depending on scaling region

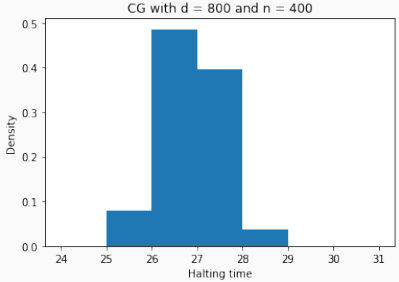


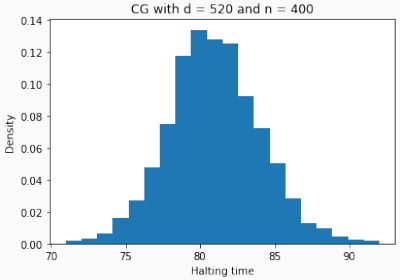




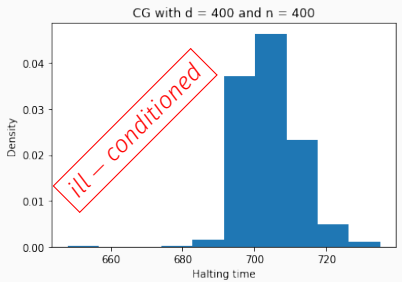
Three distinct behaviors of CG depending on scaling region

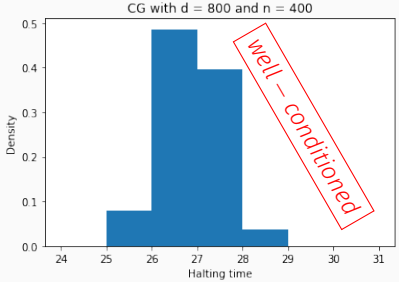


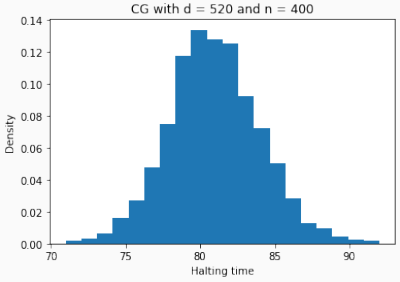




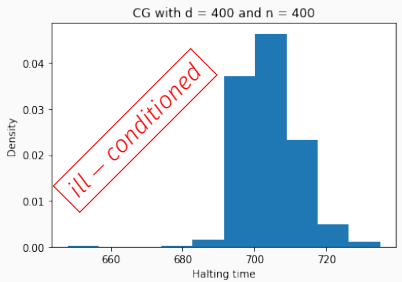
Three distinct behaviors of CG depending on scaling region

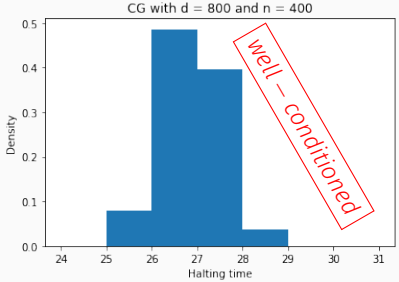


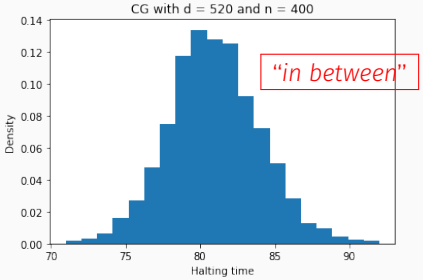




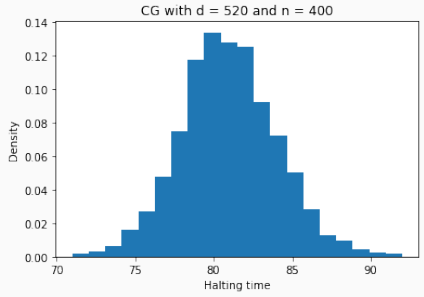
Three distinct behaviors of CG depending on scaling region

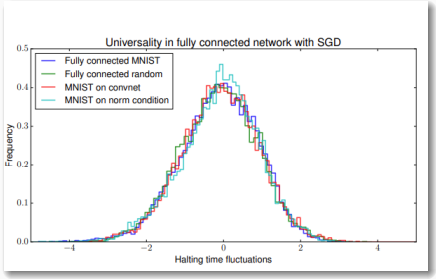






Qualitative comparison with SGD





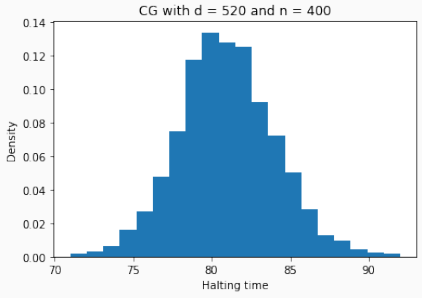
While the mechanisms behind these behaviors are surely different, we see a non-trivial histogram in each setting.

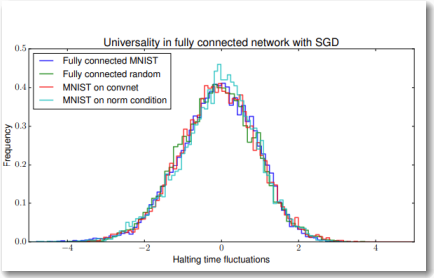
For CG on Wishart matrices, it can be shown that

$$\|r_k\| = \|c - A^T A x_k\| \overset{(\text{dist})}{=} \prod_{j=0}^{k-1} \frac{\chi_{n-j-1}}{\chi_{d-j}},$$

for independent chi-distributed random variables.

Qualitative comparison with SGD





So, if we set

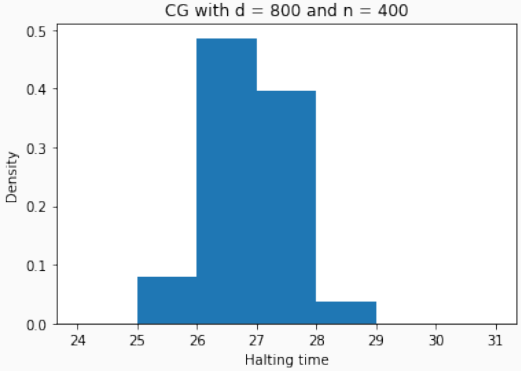
$$E_k(\text{Wishart}; \text{CG}) = \|r_k\|$$

we can analyze the halting time to see that

$$T_{\text{CG}}(\text{Wishart}, \varepsilon) \approx \frac{2}{c} n^{1-\alpha} \log \varepsilon^{-1} + O(n^{3/2-2\alpha}) \mathcal{N}(0,1),$$

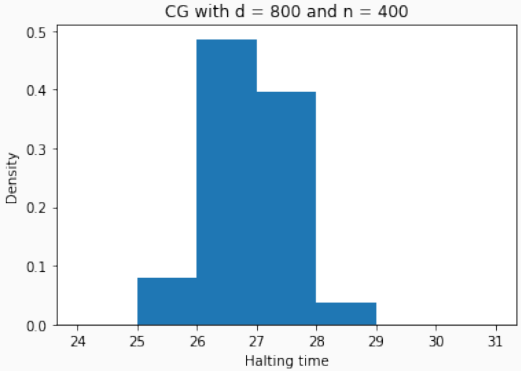
for $1/2 < \alpha < 1$.

To the well-conditioned regime!



It turns out that the errors $E_k(P_N; \mathcal{A})$ for iterative methods for a linear system involving A^TA are often analyzable in the well-conditioned, ill-conditioned and "in between" regimes. But the analysis of the halting time can be much more involved because the halting time $T_{\mathcal{A}}(P_n, \varepsilon)$ can tend to infinity with n!

To the well-conditioned regime!



So, we, for the time being, let $d = \lfloor nr \rfloor$ for $r > 1$.

The Gradient Descent Algorithm

1. x_0 is the initial vector.
2. For $k = 1, 2, \ldots$
   2.1 Select step size γ_k
   2.2 Compute $x_k = x_{k-1} - \gamma_k \nabla \mathcal{L}(x_{k-1})$

The Gradient Descent Algorithm

1. x_0 is the initial vector.
2. For $k = 1, 2, \ldots$
   2.1 Select step size γ_k
   2.2 Compute $x_k = x_{k-1} - \gamma_k \nabla \mathcal{L}(x_{k-1})$

Recall that the gradient of the regression functional is

$$\nabla \mathcal{L}(x) = Wx - c, \quad W = \frac{A^T A}{d}.$$

The Gradient Descent Algorithm

1. x_0 is the initial vector.
2. For $k = 1, 2, \ldots$
   2.1 Select step size γ_k
   2.2 Compute $x_k = x_{k-1} - \gamma_k \nabla \mathcal{L}(x_{k-1})$

Recall that the gradient of the regression functional is

$$\nabla \mathcal{L}(x) = Wx - c, \quad W = \frac{A^T A}{d}.$$

A direction calculation reveals that

$$x_k = Q_k(W)c,$$

for a polynomial Q_k of degree $k-1$ with coefficients that depend on γ_j, $j = 1, 2, \ldots, k$.

For simplicity, suppose that W is full rank. Then if x is the true minimizer, a crucial calculation is that

$$x - x_k = W^{-1}c - Q_k(W)c = W^{-1}\underbrace{(I_n - WQ_k(W))}_{R_k(W)}c.$$

Note that R_k is a polynomial of degree k satisfying $R_k(0) = 1$.

Then

$$\nabla \mathcal{L}(x_k) = Wx_k - Wx = R_k(W)c,$$
$$\|R_k(W)c\|^2 = c^T R_k(W)^2 c.$$

Example: GD

For GD follows that the difference $\boldsymbol{x}_k - \boldsymbol{x}$ satisfies

$$\boldsymbol{x}_k - \boldsymbol{x} = \boldsymbol{x}_{k-1} - \boldsymbol{x} - \gamma_k(W\boldsymbol{x}_{k-1} - W\boldsymbol{x}) = (\boldsymbol{I}_n - \gamma_k W)(\boldsymbol{x}_{k-1} - \boldsymbol{x}).$$

And so,

$$R_k(x) = \prod_{j=1}^{k}(1 - \gamma_j x).$$

Example: GD

For GD follows that the difference $\boldsymbol{x}_k - \boldsymbol{x}$ satisfies

$$\boldsymbol{x}_k - \boldsymbol{x} = \boldsymbol{x}_{k-1} - \boldsymbol{x} - \gamma_k(W\boldsymbol{x}_{k-1} - W\boldsymbol{x}) = (\boldsymbol{I}_n - \gamma_k W)(\boldsymbol{x}_{k-1} - \boldsymbol{x}).$$

And so,

$$R_k(x) = \prod_{j=1}^{k}(1 - \gamma_j x).$$

For CG the polynomial R_k is best characterized using the theory of orthogonal polynomials.

Enter RMT

$$\|R_k(W)\boldsymbol{c}\|^2 = \boldsymbol{c}^T R_k(W)^2 \boldsymbol{c}$$

The error analysis of GD (and, as it turns out, CG) is reduced to:

1. The determination/characterization of the polynomial R_k.
2. The estimation of $\boldsymbol{c}^T R_k(W)^2 \boldsymbol{c}$.

For many methods of interest (CG and GD included), the coefficients of R_k depend continuously on the eigenvalues and eigenvectors of W in a sufficiently strong sense that

$$R_k(x) \xrightarrow[n\to\infty]{\Pr} \mathcal{R}_k(x) \longleftarrow \text{deterministic}.$$

Then, one can conclude

$$c^T R_k(W)^2 c \xrightarrow[n\to\infty]{\Pr} \int_{\mathbb{R}} \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x).$$

This provides a deterministic limit for the (random) errors that are encountered throughout the algorithm.

Note: This is true only if c is independent of W and in the regression problem it is not.

Building back to true regression

For the regression problem, we have

$$c = \frac{1}{n}\left[A^T A a + A^T \eta\right].$$

Then

$$\|\mathcal{L}(x_k)\|^2 = a^T W^2 R_k(W)^2 a^T + \frac{1}{n^2}\eta^T A R_k(W)^2 A^T \eta + \underbrace{\frac{2}{n} a^T W R_k(W)^2 A^T \eta}$$

Building back to true regression

For the regression problem, we have

$$c = \frac{1}{n}\left[A^T A a + A^T \eta\right].$$

Then

$$\|\mathcal{L}(x_k)\|^2 = a^T W^2 R_k(W)^2 a^T + \frac{1}{n^2}\eta^T A R_k(W)^2 A^T \eta + \underbrace{\frac{2}{n} a^T W R_k(W)^2 A^T \eta}_{\approx 0 \quad \text{if} \quad a, \eta \text{ indep.}}$$

Building back to true regression

For the regression problem, we have

$$c = \frac{1}{n}\left[A^T A a + A^T \eta\right].$$

Then

$$\|\mathcal{L}(x_k)\|^2 = a^T W^2 R_k(W)^2 a^T + \frac{1}{n^2}\eta^T A R_k(W)^2 A^T \eta + \underbrace{\frac{2}{n} a^T W R_k(W)^2 A^T \eta}_{\approx 0 \quad \text{if} \quad a,\eta \text{ indep.}}$$

$$\xrightarrow[n\to\infty]{\Pr} \underbrace{R\int_{\mathbb{R}} x^2 \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x) + \tilde{R}\int_{\mathbb{R}} x\mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x)}_{\epsilon_k^2}.$$

Building back to true regression

For the regression problem, we have

$$c = \frac{1}{n}\left[A^T A a + A^T \eta\right].$$

Then

$$\|\mathcal{L}(x_k)\|^2 = a^T W^2 R_k(W)^2 a^T + \frac{1}{n^2}\eta^T A R_k(W)^2 A^T \eta + \underbrace{\frac{2}{n} a^T W R_k(W)^2 A^T \eta}_{\approx 0 \text{ if } a,\eta \text{ indep.}}$$

$$\xrightarrow[n\to\infty]{\Pr} \underbrace{R\int_{\mathbb{R}} x^2 \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x) + \tilde{R}\int_{\mathbb{R}} x\mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x)}_{\epsilon_k^2}.$$

Important features:

- This demonstrates that the entire spectrum of W contributes via μ_{SCM}
- Nearly all probabilistic analyses of algorithms give inequalities whereas this gives true leading-order behavior.

$$\|\mathcal{L}(x_k)\|^2 \xrightarrow[n\to\infty]{\Pr} R \int_{\mathbb{R}} x^2 \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x) + \tilde{R} \int_{\mathbb{R}} x \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

If one has a good guess as to what the limiting distribution μ_{SCM} is then the γ_k's in GD can be chosen based on this limit — to minimize this expression, see Pedregosa and Scieur [2020].

$$\|\mathcal{L}(x_k)\|^2 \xrightarrow[n\to\infty]{\Pr} R \int_{\mathbb{R}} x^2 \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x) + \tilde{R} \int_{\mathbb{R}} x \mathcal{R}_k(x)^2 \mu_{\mathrm{SCM}}(\mathrm{d}x)$$

If one has a good guess as to what the limiting distribution μ_{SCM} is then the γ_k's in GD can be chosen based on this limit — to minimize this expression, see Pedregosa and Scieur [2020].

Furthermore, by preconditioning one can make such a guess valid, see Lacotte and Pilanci [2020].

Provided that $\mathfrak{e}_k \xrightarrow{k\to\infty} 0$, one finds that

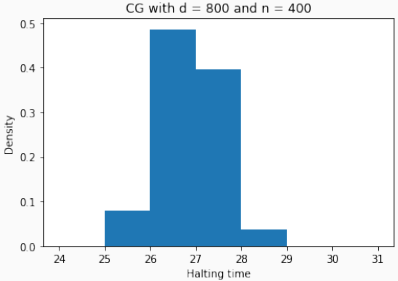
$$\lim_{n\to\infty} \mathbb{P}\left(T_{\mathcal{A}}(P_n;\varepsilon) = \min\{k : \mathfrak{e}_k < \varepsilon\}\right) = 1,$$

for most choices of ε.

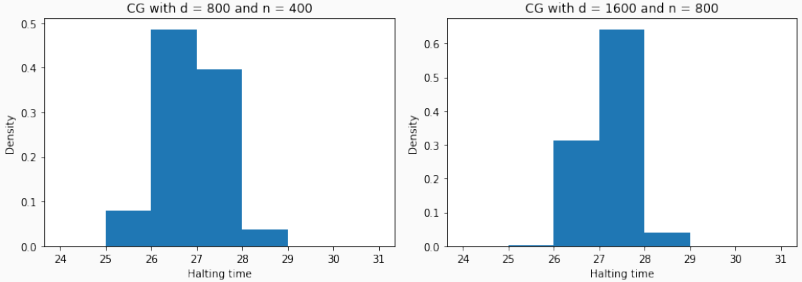
This turns out to be true for all $d \geq n$, $n \to \infty$, for the regression problem with CG or GD.

Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$

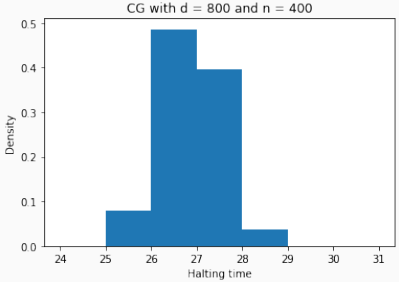
Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$

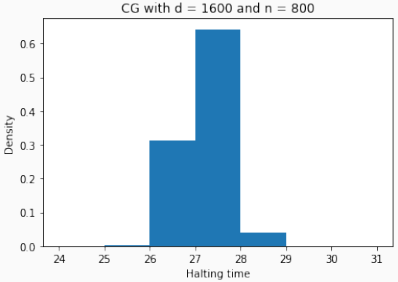


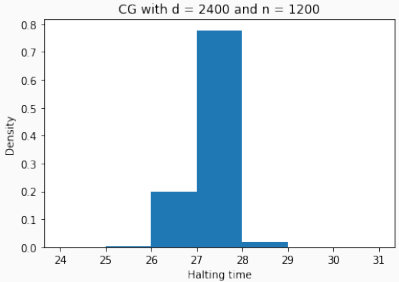
Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$



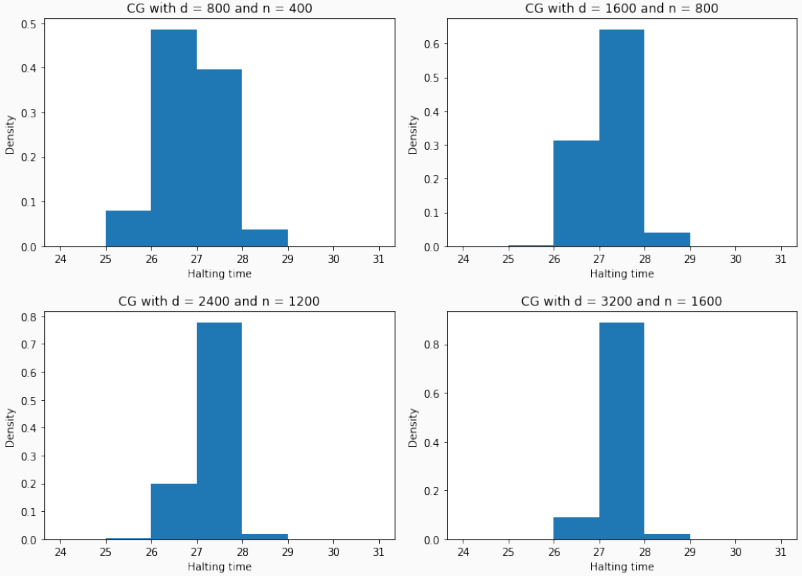
Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$







Deterministic halting for CG with $r = 2$, $\varepsilon = 10^{-4}$



Outlook

RMT provides non-trivial tractable models to analyze the statistics of optimization algorithms.

Other algorithms are analyzable:

- MINRES algorithm
- Polyak algorithm
- Nesterov accelerated algorithm
- SGD for regression
- ...

See the preprints: Paquette and Trogdon [2020], Paquette et al. [2021], Ding and Trogdon [2021], Paquette et al. [2020]

Outlook

Other ensembles are analyzable using the following results from RMT:

- Spiked random matrices (see Baik et al. [2005], Bloemendal and Virág [2013], Ding and Yang [2019], and many more)
- Nonlinear models (see Part 4)
- Random graphs (see Erdős et al. [2013], for example)
- Invariant ensembles (see Bourgade et al. [2014], Deift [2000] and many more)

Open questions

Many open questions remain:

Open questions

Many open questions remain:

- To what extent can one move these ideas beyond regression? To a two-layer network? Rank-one matrix completion problem?

Open questions

Many open questions remain:

- To what extent can one move these ideas beyond regression? To a two-layer network? Rank-one matrix completion problem?
- What is a good probability distribution to study? Wishart is clearly the place to start but what is relevant in practice?

A CG demo

See Colab for a CG demo

https://colab.research.google.com/drive/1UZRSK665b8sqq0NQFwMCwrVabPlB-7nK?usp=sharing

References

J Baik, G Ben Arous, and S Péché. Phase transition of the largest eigenvalue for nonnull complex sample covariance matrices. *The Annals of Probability*, 33(5), sep 2005. ISSN 0091-1798. doi: 10.1214/009117905000000233. URL *https://projecteuclid.org/journals/annals-of-probability/volume-33/issue-5/Phase-transition-of-the-largest-eigenvalue-for-nonnull-complex-sample/10.1214/009117905000000233.full*.

A Bloemendal and B Virág. Limits of spiked random matrices I. *Probability Theory and Related Fields*, 156(3-4):795–825, aug 2013. ISSN 0178-8051. doi: 10.1007/s00440-012-0443-2. URL *http://link.springer.com/10.1007/s00440-012-0443-2*.

K H Borgwardt. *The simplex method: A probabilistic analysis*. Springer–Verlag, Berlin, Heidelberg, 1987. URL *https://www.springer.com/gp/book/9783540170969*.

P Bourgade, L Erdős, and H-T Yau. Edge Universality of Beta Ensembles. *Communications in Mathematical Physics*, 332(1):261–353, nov 2014. ISSN 0010-3616. doi: 10.1007/s00220-014-2120-z. URL *http://link.springer.com/10.1007/s00220-014-2120-z*.

D Dadush and S Huiberts. A Friendly Smoothed Analysis of the Simplex Method. *SIAM Journal on Computing*, 49(5):STOC18–449–STOC18–499, jan 2020. ISSN 0097-5397. doi: 10.1137/18M1197205. URL *https://epubs.siam.org/doi/10.1137/18M1197205*.

G B Dantzig. Maximization of a linear function of variables subject to linear inequalities. In *Activity Analysis of Production and Allocation*, pages 339–347. 1951. ISBN 0804748349. URL *http://cowles.econ.yale.edu/P/cm/m13/m13-21.pdf*.

G B Dantzig. Origins of the simplex method. In *A history of scientific computing*, pages 141–151. ACM, New York, NY, USA, jun 1990. doi: 10.1145/87252.88081. URL *http://dl.acm.org/doi/10.1145/87252.88081*.

P Deift. *Orthogonal Polynomials and Random Matrices: a Riemann-Hilbert Approach*. Amer. Math. Soc., Providence, RI, 2000.

References ii

P Deift and T Trogdon. Universality for Eigenvalue Algorithms on Sample Covariance Matrices. *SIAM Journal on Numerical Analysis*, 2017. URL *http://arxiv.org/abs/1701.01896*.

Percy A Deift, Govind Menon, Sheehan Olver, and Thomas Trogdon. Universality in numerical computations with random data. *Proceedings of the National Academy of Sciences*, 2014. URL *https://doi.org/10.1073/pnas.1413446111*.

A Deshpande and D A Spielman. Improved Smoothed Analysis of the Shadow Vertex Simplex Method. In *46th Annual IEEE Symposium on Foundations of Computer Science (FOCS'05)*, pages 349–356. IEEE, 2005. ISBN 0-7695-2468-0. doi: 10.1109/SFCS.2005.44. URL *http://ieeexplore.ieee.org/document/1530727/*.

X Ding and T Trogdon. The conjugate gradient algorithm on a general class of spiked covariance matrices. *arXiv preprint arXiv:2106.13902*, jun 2021. URL *http://arxiv.org/abs/2106.13902*.

X Ding and F Yang. Spiked separable covariance matrices and principal components. *arXiv preprint arXiv:1905.13060*, may 2019. URL *http://arxiv.org/abs/1905.13060*.

L Erdős, A Knowles, H-T Yau, and J Yin. Spectral statistics of Erdős–Rényi graphs I: Local semicircle law. *The Annals of Probability*, 41(3B), may 2013. ISSN 0091-1798. doi: 10.1214/11-AOP734. URL *https://projecteuclid.org/journals/annals-of-probability/volume-41/issue-3B/Spectral-statistics-of-ErdősR{é}nyi-graphs-I-Local-semicircle-law/10.1214/11-AOP734.full*.

M Hestenes and E Steifel. Method of Conjugate Gradients for Solving Linear Systems. *J. Research Nat. Bur. Standards*, 20:409–436, 1952.

S Kaczmarz. Angenäherte auflösung von systemen linearer gleichungen (english translation by jason stockmann): Bulletin international de l'académie polonaise des sciences et des lettres. 1937.

A Knowles and J Yin. Anisotropic local laws for random matrices. *Probability Theory and Related Fields*, 169(1-2):257–352, oct 2017. ISSN 0178-8051. doi: 10.1007/s00440-016-0730-4. URL *http://link.springer.com/10.1007/s00440-016-0730-4*.

E Kostlan. Complexity theory of numerical linear algebra. *Journal of Computational and Applied Mathematics*, 22(2-3):219–230, jun 1988. ISSN 03770427. doi: 10.1016/0377-0427(88)90402-5. URL *http://www.sciencedirect.com/science/article/pii/0377042788904025*.

References iii

J Kuczyński and H Woźniakowski. Estimating the Largest Eigenvalue by the Power and Lanczos Algorithms with a Random Start. *SIAM Journal on Matrix Analysis and Applications*, 13(4):1094–1122, oct 1992. ISSN 0895-4798. doi: 10.1137/0613066. URL *http://epubs.siam.org/doi/10.1137/0613066*.

Jonathan Lacotte and Mert Pilanci. Optimal randomized first-order methods for least-squares problems. In *International Conference on Machine Learning*. PMLR, 2020. URL *https://arxiv.org/pdf/2002.09488.pdf*.

Z Liao and M W Mahoney. Hessian Eigenspectra of More Realistic Nonlinear Models. mar 2021. URL *http://arxiv.org/abs/2103.01519*.

C Paquette, B van Merriënboer, and F Pedregosa. Halting Time is Predictable for Large Models: A Universality Property and Average-case Analysis. *arXiv preprint arXiv:2006.04299*, jun 2020. URL *http://arxiv.org/abs/2006.04299*.

C Paquette, K Lee, F Pedregosa, and E Paquette. SGD in the Large: Average-case Analysis, Asymptotics, and Stepsize Criticality. *arXiv preprint 2102.04396*, feb 2021. URL *http://arxiv.org/abs/2102.04396*.

Elliot Paquette and Thomas Trogdon. Universality for the conjugate gradient and minres algorithms on sample covariance matrices. *arXiv preprint arXiv:2007.00640*, 2020. URL *https://arxiv.org/pdf/2007.00640.pdf*.

Fabian Pedregosa and Damien Scieur. Acceleration through spectral density estimation. In *International Conference on Machine Learning*. PMLR, 2020. URL *https://arxiv.org/pdf/2002.04756.pdf*.

C W Pfrang, P Deift, and G Menon. How long does it take to compute the eigenvalues of a random symmetric matrix? *Random matrix theory, interacting particle systems, and integrable systems, MSRI Publications*, 65:411–442, 2014. URL *http://arxiv.org/abs/1203.4635*.

Levent Sagun, Thomas Trogdon, and Yann LeCun. Universal halting times in optimization and machine learning. *Quarterly of Applied Mathematics*, 2017. URL *https://doi.org/10.1090/qam/1483*.

S Smale. On the average number of steps of the simplex method of linear programming. *Mathematical Programming*, 27(3):241–262, oct 1983. ISSN 0025-5610. doi: 10.1007/BF02591902. URL *http://link.springer.com/10.1007/BF02591902*.

References iv

D A Spielman and S-H Teng. Smoothed analysis of algorithms. *Journal of the ACM*, 51(3):385–463, may 2004. ISSN 00045411. doi: 10.1145/990308.990310. URL http://dl.acm.org/citation.cfm?id=990308.990310.

T Strohmer and R Vershynin. A Randomized Kaczmarz Algorithm with Exponential Convergence. *Journal of Fourier Analysis and Applications*, 15(2): 262–278, apr 2009. ISSN 1069-5869. doi: 10.1007/s00041-008-9030-4. URL http://link.springer.com/10.1007/s00041-008-9030-4.

R Vershynin. Beyond Hirsch Conjecture: Walks on Random Polytopes and Smoothed Complexity of the Simplex Method. *SIAM Journal on Computing*, 39, 2009. URL http://epubs.siam.org/doi/10.1137/070683386.