1 Eigenvalue computation

In this section, we will see an algorithm to compute eigenvalues and eigenvectors. Specifically, we are given a matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$. We will find a vector $\boldsymbol{v}$ and value λ such that $\mathbf{A}\boldsymbol{v} = \lambda \boldsymbol{v}$. We will want to do this more simply/efficiently than setting up the equation $\det(\mathbf{A} - \lambda \mathbf{I}) = 0$, solving, then solving the resulting set of linear systems.

To see where this algorithm might come from, notice that when we are done, we should satisfy the eigenvector-eigenvalue condition:

$$\mathbf{A}\boldsymbol{v} = \lambda \boldsymbol{v}$$

This suggests a natural algorithm. We take our eigenvector guess $\boldsymbol{v}$, plug it into this fixed-point (i.e., compute $\mathbf{A}\boldsymbol{v}$), and keep repeating this.

This type of algorithm is called the *power iteration*. We will see that this algorithm does in fact return something useful.

Theorem 1. *Let $\mathbf{A} \in \mathbb{R}^{n \times n}$ be a symmetric positive semidefinite matrix with eigenvalue-eigenvector pairs $(\lambda_i, \boldsymbol{u}_i)$ where $\lambda_1 \geq \cdots \geq \lambda_n$. If the initialization $\boldsymbol{v}$ satisfies $|\langle \boldsymbol{v}, \boldsymbol{u}_i \rangle| \geq 0.01/\sqrt{n}$, then, the power iteration converges to the largest eigenvalue-eigenvector pair of $\mathbf{A}$.*

Remark 1. *The power iteration actually converges for a wider class of matrices than the ones stated in Theorem 1. However, we present it just for symmetric positive semidefinite matrices to keep things relatively simple.*

Proof of Theorem 1. To clean up notation, let $\boldsymbol{v}_1, \ldots, \boldsymbol{v}_t$ be the iterates of our algorithm.

Recall that for symmetric positive semidefinite matrices, we can decompose them as follows. Let $\lambda_1 \geq \cdots \geq \lambda_n$ be the eigenvalues of $\mathbf{A}$ and $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_n$ be the corresponding eigenvectors. Notice that

$$\mathbf{A} = \sum_{i=1}^{n} \lambda_i \boldsymbol{u}_i \boldsymbol{u}_i^T = \mathbf{U}\Lambda\mathbf{U}^T.$$

Here, the columns of $\mathbf{U}$ are the $\boldsymbol{u}_i$.

Since $\mathbf{U}$ is orthonormal, we have $\mathbf{U}^T\mathbf{U} = \mathbf{I}_n$ and therefore $\mathbf{U}^T = \mathbf{U}^{-1}$. This means we can write

$$\mathbf{A} = \mathbf{U}\Lambda\mathbf{U}^T = \mathbf{U}\Lambda\mathbf{U}^{-1}.$$

So, after powering $\mathbf{A}$ k times, we have

$$\mathbf{A}^k = \mathbf{U}\Lambda^k\mathbf{U}^{-1}.$$

We now try to understand $\mathbf{A}^k\boldsymbol{v}$. From the above, we see that

$$\mathbf{A}^k\boldsymbol{v} = \mathbf{U}\Lambda^k\mathbf{U}^{-1}\boldsymbol{v} = \mathbf{U}\Lambda^k\mathbf{U}^T\boldsymbol{v} = \sum_{i=1}^{n} \lambda_i^k \boldsymbol{u}_i \boldsymbol{u}_i^T \boldsymbol{v} = \sum_{i=1}^{n} \lambda_i^k \langle \boldsymbol{u}_i, \boldsymbol{v} \rangle \boldsymbol{u}_i.$$

Now, by scale invariance, and the interpretation of the $\boldsymbol{u}_i$ as a change of basis, it is enough to argue about the *direction* of the vector $\mathbf{A}^k\boldsymbol{v}$ in the following way:

$$\frac{\mathbf{A}^k\boldsymbol{v}}{\lambda_1^k} = \sum_{i=1}^{n} \left(\frac{\lambda_i}{\lambda_1}\right)^k \langle \boldsymbol{u}_i, \boldsymbol{v} \rangle \boldsymbol{u}_i$$

At this point, the idea is that if λ_2/λ_1 is small, then as we take k very large, terms $2, \ldots, n$ in the above summation disappear and we are just left with $\langle \boldsymbol{u}_1, \boldsymbol{v} \rangle \boldsymbol{u}_1$ – this is in the direction of the largest eigenvector, which is what we wanted.

To make this formal, we will normalize somewhat differently. Let $\boldsymbol{t}_i \coloneqq \lambda_i^k \langle \boldsymbol{u}_i, \boldsymbol{v} \rangle$ and write

$$\left\langle \boldsymbol{u}_1, \frac{\mathbf{A}^k \boldsymbol{v}}{\|\mathbf{A}^k \boldsymbol{v}\|_2} \right\rangle = \left\langle \boldsymbol{u}_1, \sum_{i=1}^{n} \frac{\boldsymbol{t}_i}{\|\boldsymbol{t}\|} \right\rangle$$

[nsm: This is not complete]

This completes the proof of Theorem 1. □

2 Solving linear systems

The notes in this subsection are loosely based on the monograph by Vishnoi [Vis13] and the paper by Needell, Srebro, and Ward [NSW15]. As far as I know, this type of method was first developed by Strohmer and Vershynin [SV07].

We will now see a greedy algorithmic framework for solving $\mathbf{A}\boldsymbol{x} = \boldsymbol{b}$. This approach will work if we assume only that $\mathbf{A}$ is invertible. Based on this, there must exist a solution $\boldsymbol{x}^\star$ for which $\mathbf{A}\boldsymbol{x}^\star = \boldsymbol{b}$. Thus, without loss of generality, let us assume that the rows of $\mathbf{A}$ are such that $\|\boldsymbol{a}_i\|_2 = 1$ for all $i \in [n]$. This can be ensured in n vector-vector products. Now, see Algorithm 1.

Algorithm 1 General recipe for randomized linear system solving

```
1: Input: $\mathbf{A} \in \mathbb{R}^{n \times n}, \boldsymbol{b} \in \mathbb{R}^n$
2: Output: Solution $\widehat{\boldsymbol{x}}$ such that $\mathbf{A}\widehat{\boldsymbol{x}} \approx \boldsymbol{b}$.
3: Initialize $\boldsymbol{x}_0 = 0$.
4: Construct some probability distribution $p_1, \ldots, p_n$ over the rows of $\mathbf{A}$.
5: for $t = 1, \ldots, T-1$ do
6:     Choose equation $i$ according to probability $p_i$.
7:     Update $\boldsymbol{x}_t$ to be the orthogonal projection of $\boldsymbol{x}_{t-1}$ onto the hyperplane:
           $H_i \coloneqq \{\boldsymbol{z} \in \mathbb{R}^n \ : \ \langle \boldsymbol{a}_i, \boldsymbol{z} \rangle = \boldsymbol{b}_i\}$
       Formally, $\boldsymbol{x}_t = \boldsymbol{x}_{t-1} - \gamma \boldsymbol{a}_i$ where
           $\gamma = \langle \boldsymbol{x}_{t-1}, \boldsymbol{a}_i \rangle - \boldsymbol{b}_i.$
8: end for
9: return $\boldsymbol{x}_T$
```

Essentially, Algorithm 1 chooses an equation according to some probability distribution and greedily fits that equation (note that the SGD update also does this, but somewhat less explicitly). We then keep doing this over and over again. A natural question is – should this work? And if so, what is the convergence rate that we get?

The main result that we will show is that the update rule as written here is pretty good in expectation:

Theorem 2. *Let $\boldsymbol{x}^\star$ be the solution to $\mathbf{A}\boldsymbol{x}^\star = \boldsymbol{b}$. Let $\mathbf{P}$ be the diagonal matrix whose entries are the distribution p_i. Then,*

$$\mathbb{E}\left[\|\boldsymbol{x}_t - \boldsymbol{x}^\star\|_2^2\right] \leq \left(1 - \sigma_{\min}\left(\mathbf{P}^{1/2}\mathbf{A}\right)^2\right) \|\boldsymbol{x}_{t-1} - \boldsymbol{x}^\star\|_2^2 .$$

In each iteration, we perform one vector-vector product.

To prove Theorem 2, we first prove a quick lemma about the form of the update:

Lemma 1. *Let $t \in \{1, \ldots, T-1\}$ denote some iteration in Algorithm 1. We have*

$$\boldsymbol{x}_t = \boldsymbol{x}_{t-1} - \boldsymbol{a}_i \langle \boldsymbol{a}_i, \boldsymbol{x}_{t-1} - \boldsymbol{x}^\star \rangle \tag{1}$$

and therefore

$$\|\boldsymbol{x}_{t-1} - \boldsymbol{x}^\star\|_2^2 - \|\boldsymbol{x}_t - \boldsymbol{x}^\star\|_2^2 = |\langle \boldsymbol{a}_i, \boldsymbol{x}_{t-1} - \boldsymbol{x}^\star \rangle|^2 \tag{2}$$

Proof of Lemma 1. We first prove

$$\|\boldsymbol{x}_{t-1} - \boldsymbol{x}^\star\|_2^2 - \|\boldsymbol{x}_t - \boldsymbol{x}^\star\|_2^2 = \|\boldsymbol{x}_t - \boldsymbol{x}_{t-1}\|_2^2 .$$

To do so, we use the Pythagorean Theorem. We know that $\boldsymbol{x}_t$ is the orthogonal projection of $\boldsymbol{x}_{t-1}$ onto the hyperplane H_i. Since $\boldsymbol{x}^\star$ also lies on H_i, the angle formed by the vectors $\boldsymbol{x}^\star - \boldsymbol{x}_t$ and $\boldsymbol{x}_{t-1} - \boldsymbol{x}_t$ is a right angle.

Next, we prove (1). It is easy to see that (2) follows from plugging in (1) into the Pythagorean Theorem statement we have above. First, observe that since $\boldsymbol{x}_t$ is the orthogonal projection of $\boldsymbol{x}_{t-1}$ onto H_i, the vector $\boldsymbol{x}_{t-1} - \boldsymbol{x}_t$ must be aligned with $\boldsymbol{a}_i$. In particular, for some coefficient γ, we have

$$\boldsymbol{x}_{t-1} - \boldsymbol{x}_t = \gamma \boldsymbol{a}_i .$$

Let us figure out what γ is. Since $\boldsymbol{x}_t$ is on H_i, we must have $\langle \boldsymbol{a}_i, \boldsymbol{x}_t \rangle = \boldsymbol{b}_i$. We also know that since the system is satisfiable, we have $\langle \boldsymbol{a}_i, \boldsymbol{x}^\star \rangle = \boldsymbol{b}_i$. Thus, $\langle \boldsymbol{a}_i, \boldsymbol{x}_t - \boldsymbol{x}^\star \rangle = 0$. Now, we write

$$\gamma = \langle \boldsymbol{a}_i, \boldsymbol{x}_{t-1} - \boldsymbol{x}_t \rangle = \langle \boldsymbol{a}_i, \boldsymbol{x}_{t-1} - \boldsymbol{x}^\star \rangle - \langle \boldsymbol{a}_i, \boldsymbol{x}_t - \boldsymbol{x}^\star \rangle = \langle \boldsymbol{a}_i, \boldsymbol{x}_{t-1} - \boldsymbol{x}^\star \rangle .$$

Plug everything back in and conclude the proof of Lemma 1. □

With this in hand, we are ready to complete the proof of Theorem 2.

Proof of Theorem 2. We simply take the expectation of both sides of (2). We get

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{x}_{t-1} - \boldsymbol{x}^\star\|_2^2 - \|\boldsymbol{x}_t - \boldsymbol{x}^\star\|_2^2\right] &= \sum_{i=1}^n p_i |\langle \boldsymbol{a}_i, \boldsymbol{x}_{t-1} - \boldsymbol{x}^\star \rangle|^2 \\ &= \left\|\mathbf{P}^{1/2}\mathbf{A}(\boldsymbol{x}_{t-1} - \boldsymbol{x}^\star)\right\|_2^2 \geq \sigma_{\min}\left(\mathbf{P}^{1/2}\mathbf{A}\right)^2 \|\boldsymbol{x}_{t-1} - \boldsymbol{x}^\star\|_2^2 .\end{aligned}$$

Rearranging completes the proof of Theorem 2. □

Theorem 2 suggests that we can think of the probability distribution $p_1, \ldots, p_n$ as a "preconditioner", as we can directly control this in order to optimize the convergence rate of the algorithm. A naïve choice would be to set $p_1 = \cdots = p_n = 1/n$, in which case we recover the RK algorithm of Strohmer and Vershynin [SV07]. This yields Corollary 1.

Corollary 1. *If $p_i = 1/n$ for all $i \in [n]$, then with probability $\geq 2/3$, after $T \asymp \frac{n}{\sigma_{\min}(\mathbf{A})^2} \cdot \log(1/\varepsilon)$ steps, we get*

$$\|\boldsymbol{x}_T - \boldsymbol{x}^\star\|_2^2 \leq \varepsilon^2 \|\boldsymbol{x}^\star\|_2^2 .$$

Proof of Corollary 1. Applying Theorem 2 repeatedly (very loose, since we didn't ensure that *every* step made enough progress – however, by Markov's inequality, most of these steps should have made nontrivial progress), we get

$$\mathbb{E}\left[\|\boldsymbol{x}_T - \boldsymbol{x}^\star\|_2^2\right] \leq \left(1 - \frac{\sigma_{\min}(\mathbf{A})^2}{n}\right)^T \|\boldsymbol{x}^\star\|_2^2 \leq \exp\left(-T \cdot \frac{\sigma_{\min}(\mathbf{A})^2}{n}\right) \|\boldsymbol{x}^\star\|_2^2 \lesssim \varepsilon^2 \|\boldsymbol{x}^\star\|_2^2 ,$$

completing the proof of Corollary 1. □

Remark 2. *The dimension-dependent term in Corollary 1 may not look that nice. However, recall that we first normalized our input linear system so that every equation had unit Euclidean norm. To undo this change and to solve the original system, we can set $p_i = \|\boldsymbol{a}_i\|_2^2 / \|\mathbf{A}\|_F^2$ and run the algorithm. Pushing this change through the analysis, we will be able to think of $n/\sigma_{\min}(\mathbf{A})^2$ as the square of an "average condition number" (see [Vis13, Chapter 14])*

References

[NSW15] Deanna Needell, Nathan Srebro, and Rachel Ward. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm, 2015. arXiv: 1310.5715 [math.NA] (cited on page 2).

[SV07] Thomas Strohmer and Roman Vershynin. A randomized kaczmarz algorithm with exponential convergence, 2007. arXiv: math/0702226 [math.NA] (cited on pages 2, 3).

[Vis13] N.K. Vishnoi. *Lx = B*. Foundations and Trends in Theoretical Computer Science. Now Publishers, 2013. ISBN: 9781601989468. URL: https://books.google.com/books?id=cGnJnAEACAAJ (cited on pages 2, 4).