The Power Method and Convergence Rates

Overview

Part 1: Vanilla Power Method
Part 2: Accelerated Stochastic Power Iteration
Part 3: Power Method for Tensors

Power Method

Goal: The find the dominant eigenvalue/eigenvector of a matrix **A**.

Definition

Let $\lambda_1, \lambda_2, \ldots, \lambda_n$ be the eigenvalues of a matrix A. λ_1 is called the **dominant** eigenvalue of A if $|\lambda_1| > |\lambda_i|$, for $1 < i \leq n$.

Note: If we can find the dominant eigenvector **x**, we can recover the dominant eigenvalue.

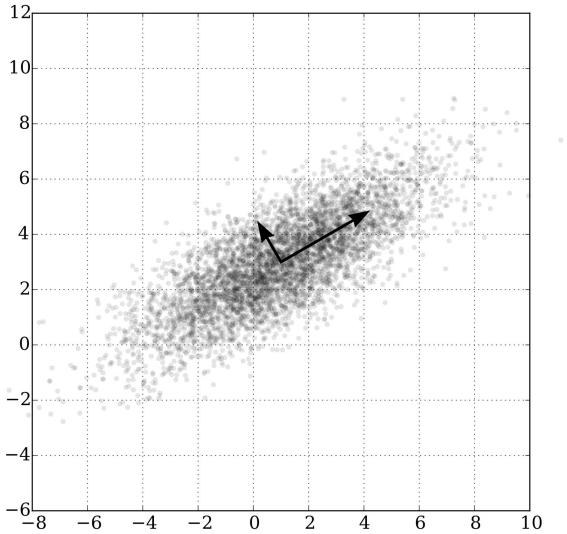
If $\mathbf{x}$ is an eigenvector of a matrix A, then its corresponding eigenvalue is given by

$$\lambda = \frac{A\mathbf{x} \cdot \mathbf{x}}{\mathbf{x} \cdot \mathbf{x}}.$$

This quotient is called the **Rayleigh quotient.**

Power Method

Applications: The most popular application is PCA. *k*-PCA refers to principal component analysis in which we wish to recover the top-*k* eigenvectors.



Power Method: Algorithm

Algorithm 1 The Power Method

Choose a random vector $q^{(0)} \in R^n$

for $k = 1, 2, \ldots$ (while $\|q^{(k-1)} - q^{(k-2)}\| > \epsilon$)

$$z^{(k)} = Aq^{(k-1)}$$
$$q^{(k)} = z^{(k)} / \|z^{(k)}\|$$
$$\lambda^{(k)} = [q^{(k)}]^T A q^{(k)}$$

end

Power Method: Algorithm

Algorithm 1 The Power Method

Choose a random vector $q^{(0)} \in R^n$

for $k = 1, 2, \ldots$ (while $\|q^{(k-1)} - q^{(k-2)}\| > \epsilon$)

$z^{(k)} = Aq^{(k-1)}$

$q^{(k)} = z^{(k)} / \|z^{(k)}\|$ ← Normalized eigenvector

$\lambda^{(k)} = [q^{(k)}]^T A q^{(k)}$ ← Rayleigh quotient to recover eigenvalue

end

Power Method: Proof

Assume A is diagonalizable. Then it has n linearly independent eigenvectors $v_1, v_2, ..., v_n$, with n eigenvalues $\lambda_1, \lambda_2, ..., \lambda_n$.

Power Method: Proof

Assume A is diagonalizable. Then it has n linearly independent eigenvectors $v_1, v_2, ..., v_n$, with n eigenvalues $\lambda_1, \lambda_2, ..., \lambda_n$.

Let v_1 be leading. Choose an initial guess x_0. Since the $\{v_i\}_{i=1}^n$ form a basis, we have $x_0 = x_1 v_1 + \cdots + c_n v_n$ for some constants c_i.

Power Method: Proof

Assume A is diagonalizable. Then it has n linearly independent eigenvectors $v_1, v_2, ..., v_n$, with n eigenvalues $\lambda_1, \lambda_2, ..., \lambda_n$.

Let v_1 be leading. Choose an initial guess x_0. Since the $\{v_i\}_{i=1}^n$ form a basis, we have $x_0 = x_1 v_1 + \cdots + c_n v_n$ for some constants c_i.

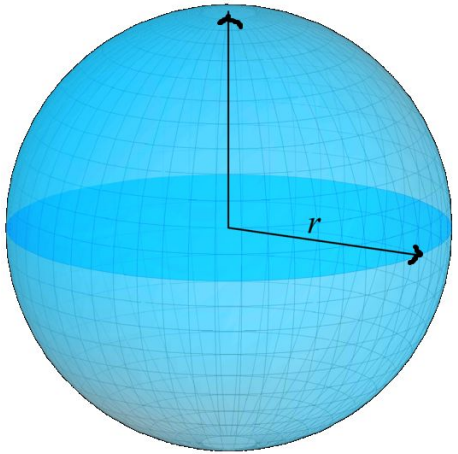
Power Method: Proof

- let's multiply both sides by A:
  - $Ax_0 = c_1Av_1 + \ldots + c_nAv_n = c_1\lambda_1 v_1 + \ldots + c_n\lambda_n v_n$
- repeat that k times:
  - $A^k x_0 = c_1\lambda_1^k v_1 + \ldots + c_n\lambda_n^k v_n$
  - or $A^k x_0 = \lambda_1^k \left[c_1 v_1 + c_2 \left(\frac{\lambda_2}{\lambda_1} \right)^k v_2 \ldots + c_n \left(\frac{\lambda_n}{\lambda_1} \right)^k v_n \right]$
- since λ_1 is dominating, the ratios $\left(\frac{\lambda_i}{\lambda_1} \right)^k \to 0$ as $k \to \infty$ for all i
- so $A^k x_0 = \lambda_1^k c_1 v_1$ and it gets better as k grows

□

Power Method: Proof

A note: if our initial guess $\mathbf{x}_0$ is orthogonal to $\mathbf{v}_1$ then the power method will not converge. However, the probability of this happening randomly is effectively 0 (think of the unit sphere).



Power Method: Convergence analysis

Theorem:

Let $M \in \mathbb{R}^{n \times n}$ a diagonalisable matrix such that its eigenvalues satisfy $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$. Let $v_1, \ldots, v_n$ be respective eigenvectors. Let $x_0 = \sum_{i=1}^{n} c_i v_i$ for some scalars c_i, $c_1 \neq 0$, $x_k = M^k x_0$, for all positive $k \in \mathbb{Z}$, and $w \in \mathbb{R}^n \setminus v_1^{\perp}$. Then

$$\frac{w^T x_k}{w^T x_{k-1}} = \lambda_1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$$

Power Method: Convergence analysis

Theorem:

Let $M \in \mathbb{R}^{n \times n}$ a diagonalisable matrix such that its eigenvalues satisfy $|\lambda_1| > |\lambda_2| \geq \cdots \geq |\lambda_n|$. Let $v_1, \ldots, v_n$ be respective eigenvectors. Let $x_0 = \sum_{i=1}^{n} c_i v_i$ for some scalars c_i, $c_1 \neq 0$, $x_k = M^k x_0$, for all positive $k \in \mathbb{Z}$, and $w \in \mathbb{R}^n \setminus v_1^{\perp}$. Then

$$\frac{w^T x_k}{w^T x_{k-1}} = \lambda_1 + O\left(\left|\frac{\lambda_2}{\lambda_1}\right|^k\right)$$

Here, we're looking at convergence of the Rayleigh quotient. If the ratio $|\lambda_2/\lambda_1|$ is large, we will require many steps.

Power method: A Variation

(2018)

Accelerated Stochastic Power Iteration

CHRISTOPHER DE SA† BRYAN HE† IOANNIS MITLIAGKAS† CHRISTOPHER RÉ† PENG XU*

Accelerated Stochastic Power Iteration (ASPI)

Setup:

$x_1, \ldots, x_n \in \mathbb{R}^d$ data points. The goal of PCA here is to find the top eigenvector of

$$A = \frac{1}{n} \sum_{i=1}^{n} x_i x_i^T \in \mathbb{R}^{d \times d}$$

A is the **covariance** matrix.

Accelerated Stochastic Power Iteration (ASPI)

$x_1, \ldots, x_n \in \mathbb{R}^d$ data points

Stochastic: ingest a random subset of the available data at every iteration

Offline: random access to a finite set of samples, potentially full-pass

Online: randomly drawn, full passes not possible

Accelerated Stochastic Power Iteration (ASPI)

$x_1, \ldots, x_n \in \mathbb{R}^d$ data points

Sample complexity: How many samples do we need to get the top eigenvector within a fixed error bound

Iteration complexity: Number of outer loop iterations required when the inner loops are “embarrassingly” parallel. An important measure for GPUs

Accelerated Stochastic Power Iteration (ASPI)

At a glance:

Let $\Delta := |\lambda_1 - \lambda_2|$, i.e. the eigen-gap.

- The standard power iteration requires $\mathcal{O}(1/\Delta)$ full passes
- Lanczos, a more complex method, requires $\mathcal{O}(1/\sqrt{\Delta})$ full passes
- In the online stochastic setting, algorithms such as Oja's method, achieves the *optimal* sample complexity $\mathcal{O}(\sigma^2/\Delta^2)$

Accelerated Stochastic Power Iteration (ASPI)

At a glance:

Let $\Delta := |\lambda_1 - \lambda_2|$, i.e. the eigen-gap.

- Lanczos cannot operate in a stochastic manner.
- Oja's algorithm has an iteration complexity of $\mathcal{O}(\sigma^2/\Delta^2)$
- We want a stochastic algorithm with an iteration complexity of $\mathcal{O}(1/\sqrt{\Delta})$

ASPI achieves both the optimal sample complexity $\mathcal{O}(\sigma^2/\Delta^2)$ and iteration complexity $\mathcal{O}(1/\sqrt{\Delta})$

Accelerated Stochastic Power Iteration (ASPI)

Preliminaries:

Assume all our eigenvalues are nonnegative and <= 1.

We consider error ε between our iterates and the top eigenvector not as L2 error, but instead the squared sine of the angle between them:

$$\sin^2 \angle(\mathbf{u}_1, \mathbf{w}_t) \triangleq 1 - (\mathbf{u}_1^T \mathbf{w}_t)^2 / \|\mathbf{w}_t\|^2$$

Accelerated Stochastic Power Iteration (ASPI)

Vanilla power method: $\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t$

After $\mathcal{O}\left(\frac{1}{\Delta}\log\frac{1}{\epsilon}\right)$ steps, the normalized iterate $w_t/\|w_t\|$ is an ε-accurate estimate of the top component.

Accelerated Stochastic Power Iteration (ASPI)

Vanilla power method: $\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t$

After $\mathcal{O}\left(\frac{1}{\Delta}\log\frac{1}{\epsilon}\right)$ steps, the normalized iterate $w_t/\|w_t\|$ is an ε-accurate estimate of the top component.

As stated before, if the eigengap is small, we'll converge slowly.

Accelerated Stochastic Power Iteration (ASPI)

Vanilla power method: $\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t$

After $\mathcal{O}\left(\frac{1}{\Delta}\log\frac{1}{\epsilon}\right)$ steps, the normalized iterate $w_t/\|w_t\|$ is an ε-accurate estimate of the top component.

As stated before, if the eigengap is small, we'll converge slowly.

Let's speed it up with accelerated power momentum. An alternative update:

$$\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t - \beta\mathbf{w}_{t-1}.$$

Accelerated Stochastic Power Iteration (ASPI)

Vanilla power method: $\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t$

After $\mathcal{O}\left(\frac{1}{\Delta}\log\frac{1}{\epsilon}\right)$ steps, the normalized iterate $w_t/\|w_t\|$ is an ε-accurate estimate of the top component.

As stated before, if the eigengap is small, we'll converge slowly.

Let's speed it up with accelerated power momentum. An alternative update:

$$\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t - \beta\mathbf{w}_{t-1}.$$

Accelerated Stochastic Power Iteration (ASPI)

Vanilla power method: $\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t$

After $\mathcal{O}\left(\frac{1}{\Delta}\log\frac{1}{\epsilon}\right)$ steps, the normalized iterate $w_t/\|w_t\|$ is an ε-accurate estimate of the top component.

As stated before, if the eigengap is small, we'll converge slowly.

Let's speed it up with accelerated power momentum. An alternative update:

$$\mathbf{w}_{t+1} = \mathbf{A}\mathbf{w}_t - \beta\mathbf{w}_{t-1}$$

⟸ "Momentum term"

Accelerated Power Method

The effect of momentum on iterations (due to Polyak in 1964):

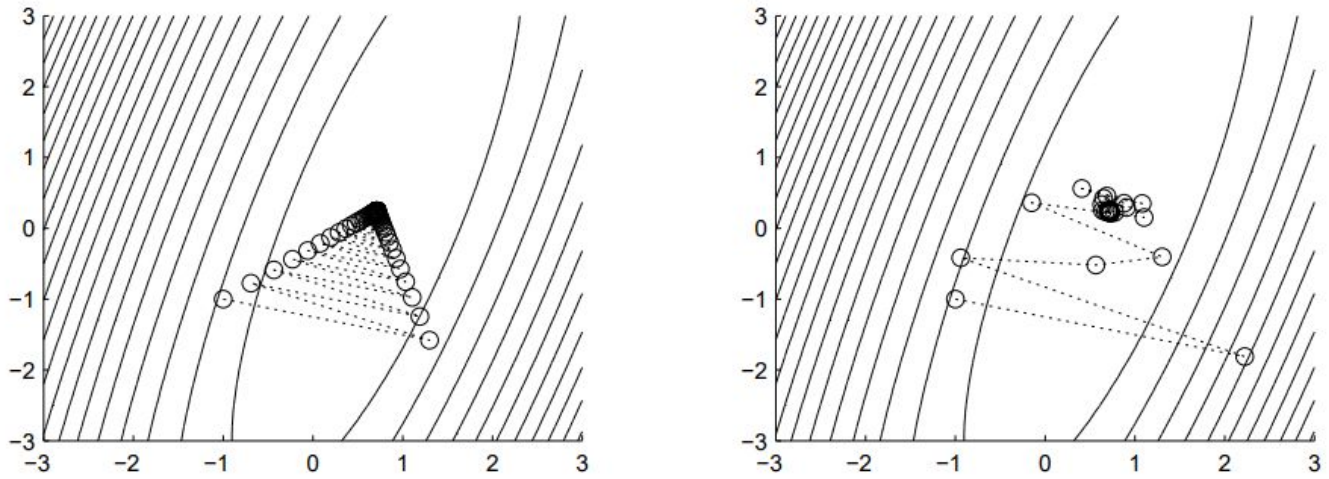


FIG 1. *The iterates of gradient descent (left panel) and the heavy ball method (right panel) starting at* $(-1, -1)$.

Accelerated Power Method

See my notes for a proof.

Theorem 1. *Given a PSD matrix* $\mathbf{A} \in \mathbb{R}^{n \times n}$ *with eigenvalues* $1 \geq \lambda_1 > \lambda_2 \geq \cdots \geq \lambda_n \geq 0$, *running update* (**A**) *with* $\lambda_2 \leq 2\sqrt{\beta} < \lambda_1$ *results in estimates with worst-case error*

$$\sin^2 \angle(\mathbf{u}_1, \mathbf{w}_t) \triangleq 1 - \frac{(\mathbf{u}_1^T \mathbf{w}_t)^2}{\|\mathbf{w}_t\|^2} \leq \frac{4}{\left|\mathbf{w}_0^T \mathbf{u}_1\right|^2} \cdot \left(\frac{2\sqrt{\beta}}{\lambda_1 + \sqrt{\lambda_1^2 - 4\beta}} \right)^{2t}.$$

Accelerated Power Method

Now for iteration complexity:

Corollary 2. *In the same setting as Theorem 1, update* (**A**) *with* $\mathbf{w}_0 \in \mathbb{R}^d$ *such that* $\mathbf{u}_1^T \mathbf{w}_0 \neq 0$, *for any* $\epsilon \in (0,1)$, *after* $T = \mathcal{O}\left(\frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}} \cdot \log \frac{1}{\epsilon}\right)$ *iterations achieves* $1 - \frac{(\mathbf{u}_1^T \mathbf{w}_T)^2}{\|\mathbf{w}_T\|^2} \leq \epsilon$.

Remark. Minimizing $\frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}}$ over $[\lambda_2^2/4, \lambda_1^2/4)$ tells us that $\beta = \lambda_2^2/4$ is the optimal setting.

*If we minimize □, we have $T = \mathcal{O}\left(\frac{1}{\Delta} \log \frac{1}{\epsilon}\right)$ *

Accelerated Power Method

A summary of these iteration complexities:

Setting	Algorithm	Number of Iterations	Batch Size
Deterministic	Power	$\mathcal{O}\left(\frac{1}{\Delta} \cdot \log\left(\frac{1}{\epsilon}\right)\right)$	n
	Lanczos	$\mathcal{O}\left(\frac{1}{\sqrt{\Delta}} \cdot \log\left(\frac{1}{\epsilon}\right)\right)$	n
	Power+M	$\mathcal{O}\left(\frac{1}{\sqrt{\Delta}} \cdot \log\left(\frac{1}{\epsilon}\right)\right)$	n

So, we've matched the Lanczos iteration complexity. But we still need to achieve this in a stochastic setting...

Stochastic PCA

We now consider a streaming setting, where we're given a series of iid samples, $\tilde{A}_t$, such that

$$\mathbb{E}[\tilde{\mathbf{A}}_t] = \mathbf{A}, \quad \max_t \|\tilde{\mathbf{A}}_t\| \leq r, \quad \mathbb{E}[\|\tilde{\mathbf{A}}_t - \mathbf{A}\|^2] = \sigma^2.$$

Stochastic PCA

We now consider a streaming setting, where we're given a series of iid samples, $\tilde{A}_t$, such that

$$\mathbb{E}[\tilde{\mathbf{A}}_t] = \mathbf{A}, \quad \max_t \|\tilde{\mathbf{A}}_t\| \leq r, \quad \mathbb{E}[\|\tilde{\mathbf{A}}_t - \mathbf{A}\|^2] = \sigma^2.$$

In the sample covariance setting as previously discussed, $\tilde{A}_t$ can be obtained by selecting $x_i x_i^T$, where x_i is uniformly sampled.

Stochastic PCA

$$\mathbb{E}[\tilde{\mathbf{A}}_t] = \mathbf{A}, \quad \max_t \|\tilde{\mathbf{A}}_t\| \leq r, \quad \mathbb{E}[\|\tilde{\mathbf{A}}_t - \mathbf{A}\|^2] = \sigma^2.$$

Oja's algorithm repeatedly runs the following update:

$$\mathbf{w}_{t+1} = (I + \eta \tilde{\mathbf{A}}_t)\mathbf{w}_t$$

Stochastic PCA: Oja's Algorithm

$$\mathbb{E}[\tilde{\mathbf{A}}_t] = \mathbf{A}, \quad \max_t \|\tilde{\mathbf{A}}_t\| \leq r, \quad \mathbb{E}[\|\tilde{\mathbf{A}}_t - \mathbf{A}\|^2] = \sigma^2.$$

Oja's algorithm repeatedly runs the following update:

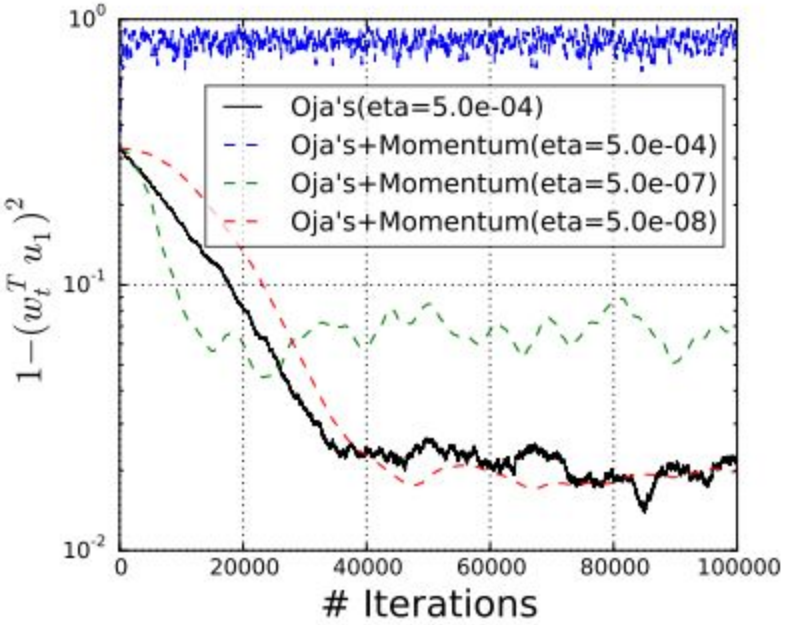
$$\mathbf{w}_{t+1} = (I + \eta \tilde{\mathbf{A}}_t)\mathbf{w}_t$$

We can try to naturally accelerate it via momentum:

$$\mathbf{w}_{t+1} = (I + \eta \tilde{\mathbf{A}}_t)\mathbf{w}_t - \beta \mathbf{w}_{t-1}.$$

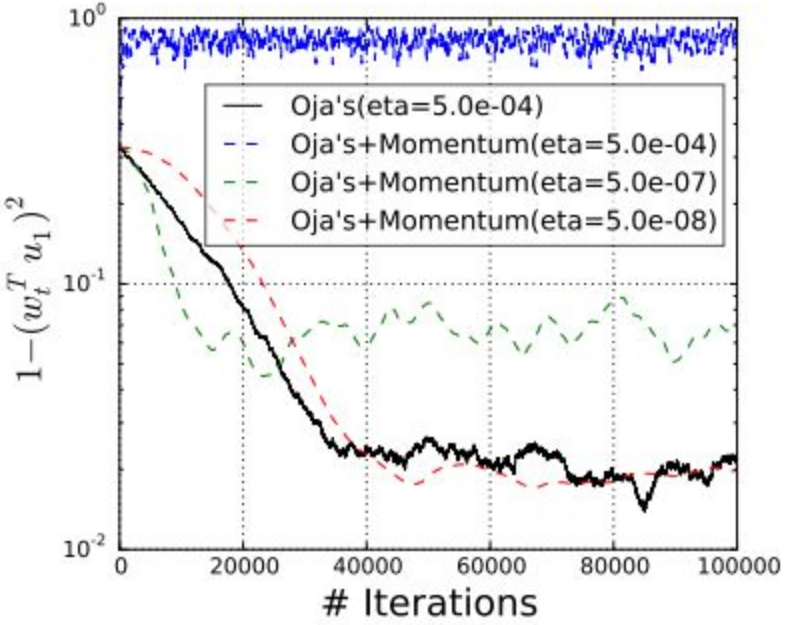
Stochastic PCA: Oja's Algorithm

Oja's algorithm + **M:** $$\mathbf{w}_{t+1} = (I + \eta\tilde{\mathbf{A}}_t)\mathbf{w}_t - \beta\mathbf{w}_{t-1}.$$



Stochastic PCA: Oja's algorithm

Oja's algorithm + **M:** $$\mathbf{w}_{t+1} = (I + \eta \tilde{\mathbf{A}}_t)\mathbf{w}_t - \beta \mathbf{w}_{t-1}.$$



As is typical with descent-like momentum algorithms, the variance converges to a so-called "noise ball" or upper-bound.

If increase momentum, we converge to the noise ball faster, but the noise ball gets larger! If we decrease the step size η, we cancel out acceleration.

Stochastic PCA: Mini-batch + M

So, we need to use variance-reduction techniques. $\mathbb{E}[\|\tilde{\mathbf{A}}_t - \mathbf{A}\|^2] = \sigma^2$

But we'll do this in just a little bit.

As an alternative to Oja's, let's turn the deterministic power algorithm + M into a stochastic algorithm:

$$\mathbf{w}_{t+1} = \mathbf{A}_t \mathbf{w}_t - \beta \mathbf{w}_{t-1}$$

I.i.d. unbiased random estimate of A

Stochastic PCA: Mini-batch + M

Algorithm 1 Mini-batch Power Method with Momentum (**Mini-batch Power+M**)

Require: Initial point $\mathbf{w}_0$, Number of Iterations T, Batch size s, Momentum parameter β

$\mathbf{w}_{-1} \leftarrow \mathbf{0}$,

for t = 0 **to** T - 1 **do**

Generate a mini-batch of i.i.d. samples $B = \{\tilde{\mathbf{A}}_{t_1}, \cdots, \tilde{\mathbf{A}}_{t_s}\}$

Update: $\mathbf{w}_{t+1} \leftarrow (\frac{1}{s}\sum_{i=1}^{s} \tilde{\mathbf{A}}_{t_i})\mathbf{w}_t - \beta \mathbf{w}_{t-1}$

Normalization: $\mathbf{w}_t \leftarrow \mathbf{w}_t / \|\mathbf{w}_{t+1}\|, \mathbf{w}_{t+1} \leftarrow \mathbf{w}_{t+1} / \|\mathbf{w}_{t+1}\|$.

end for

return $\mathbf{w}_T$

Note: If the variance is 0, the dynamics here are the same as the deterministic update.

Stochastic PCA: Mini-batch + M

$$\text{Update: } \mathbf{w}_{t+1} \leftarrow \left(\frac{1}{s}\sum_{i=1}^{s} \tilde{\mathbf{A}}_{t_i}\right)\mathbf{w}_t - \beta \mathbf{w}_{t-1}$$

If variance is nonzero, but sufficiently small we can still say something about the convergence rate. We have a PAC theorem:

Theorem 3. *Suppose we run Algorithm 1 with* $2\sqrt{\beta} \in [\lambda_2, \lambda_1)$. *Let* $\Sigma = \mathbb{E}\left[(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\right]^3$. *Suppose that* $\|\mathbf{w}_0\| = 1$ *and* $\left|\mathbf{u}_1^T \mathbf{w}_0\right| \geq 1/2$. *For any* $\delta \in (0,1)$ *and* $\epsilon \in (0,1)$, *if*

$$T = \frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}} \log\left(\frac{32}{\delta\epsilon}\right), \quad \|\Sigma\| \leq \frac{(\lambda_1^2 - 4\beta)\delta\epsilon}{256\sqrt{dT}} = \frac{(\lambda_1^2 - 4\beta)^{3/2}\delta\epsilon}{256\sqrt{d}\sqrt{\beta}} \log^{-1}\left(\frac{32}{\delta\epsilon}\right), \tag{4}$$

then with probability at least $1 - 2\delta$, *we have* $1 - (\mathbf{u}_1^T \mathbf{w}_T)^2 \leq \epsilon$.

Stochastic PCA: Mini-batch + M

A comparison:

Corollary 2. *In the same setting as Theorem 1, update (A) with* $\mathbf{w}_0 \in \mathbb{R}^d$ *such that* $\mathbf{u}_1^T \mathbf{w}_0 \neq 0$*, for any* $\epsilon \in (0,1)$*, after* $T = \mathcal{O}\left(\frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}} \cdot \log \frac{1}{\epsilon}\right)$ *iterations achieves* $1 - \frac{(\mathbf{u}_1^T \mathbf{w}_T)^2}{\|\mathbf{w}_T\|^2} \leq \epsilon$.

Theorem 3. *Suppose we run Algorithm 1 with* $2\sqrt{\beta} \in [\lambda_2, \lambda_1)$*. Let* $\Sigma = \mathbb{E}\left[(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\right]^3$*. Suppose that* $\|\mathbf{w}_0\| = 1$ *and* $\left|\mathbf{u}_1^T \mathbf{w}_0\right| \geq 1/2$*. For any* $\delta \in (0,1)$ *and* $\epsilon \in (0,1)$*, if*

$$T = \frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}} \log\left(\frac{32}{\delta\epsilon}\right), \quad \|\Sigma\| \leq \frac{(\lambda_1^2 - 4\beta)\delta\epsilon}{256\sqrt{d}T} = \frac{(\lambda_1^2 - 4\beta)^{3/2}\delta\epsilon}{256\sqrt{d}\sqrt{\beta}} \log^{-1}\left(\frac{32}{\delta\epsilon}\right), \tag{4}$$

then with probability at least $1 - 2\delta$*, we have* $1 - (\mathbf{u}_1^T \mathbf{w}_T)^2 \leq \epsilon$.

***If variance ||Σ|| is sufficiently small, then the number of iterations in the online setting is the same as the deterministic setting, up to a constant depending on δ. ***

Stochastic PCA: Mini-batch + M

If variance $\|\Sigma\|$ is sufficiently small, then the number of iterations in the online setting is the same as the deterministic setting, up to a constant depending on δ.
*

So, now we will move into variance reduction techniques. First, an inequality:

Stochastic PCA: Mini-batch + M

***If variance ||Σ|| is sufficiently small, then the number of iterations in the online setting is the same as the deterministic setting, up to a constant depending on δ. ***

So, now we will move into variance reduction techniques. First, an inequality:

Unbiased random estimate ⟹ $A_t = \frac{1}{s}\sum_{i=1}^{s} \tilde{A}_{t_i}$

$$\|\Sigma\| = \|\mathbb{E}\left[(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\right]\| \leq \mathbb{E}\left[\|(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\|\right] = \mathbb{E}\left[\|\mathbf{A}_t - \mathbf{A}\|^2\right] = \frac{\sigma^2}{s}$$

Stochastic PCA: Mini-batch + M

***If variance ||Σ|| is sufficiently small, then the number of iterations in the online setting is the same as the deterministic setting, up to a constant depending on δ. ***

So, now we will move into variance reduction techniques. First, an inequality:

Unbiased random estimate ⟹ $A_t = \frac{1}{s}\sum_{i=1}^{s} \tilde{A}_{t_i}$

$$\|\Sigma\| = \|\mathbb{E}\left[(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\right]\| \leq \mathbb{E}\left[\|(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\|\right] = \mathbb{E}\left[\|\mathbf{A}_t - \mathbf{A}\|^2\right] = \frac{\sigma^2}{s}$$

$\|\Sigma\| \leq \sigma^2/s$, where s is the size of the mini-batch and

Stochastic PCA: Mini-batch + M

So, now we will move into variance control techniques. First, an inequality:

$$\|\Sigma\| = \|\mathbb{E}\left[(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\right]\| \leq \mathbb{E}\left[\|(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\|\right] = \mathbb{E}\left[\|\mathbf{A}_t - \mathbf{A}\|^2\right] = \frac{\sigma^2}{s}$$

Corollary 4. *Suppose we run Algorithm 1 with* $2\sqrt{\beta} \in [\lambda_2, \lambda_1)$. *Assume that* $\|\mathbf{w}_0\| = 1$ *and* $\left|\mathbf{u}_1^T \mathbf{w}_0\right| \geq 1/2$. *For any* $\delta \in (0,1)$ *and* $\epsilon \in (0,1)$, *if*

$$T = \frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}} \log\left(\frac{32}{\delta\epsilon}\right), \quad s \geq \frac{256\sqrt{d}\sigma^2 T}{(\lambda_1^2 - 4\beta)\delta\epsilon} = \frac{256\sqrt{d}\sqrt{\beta}\sigma^2}{(\lambda_1^2 - 4\beta)^{3/2}\delta\epsilon} \log\left(\frac{32}{\delta\epsilon}\right),$$

then with probability at least $1 - 2\delta$, $1 - (\mathbf{u}_1^T \mathbf{w}_T)^2 \leq \epsilon$.

Stochastic PCA: Mini-batch + M

So, now we will move into variance control techniques. First, an inequality:

$$\|\Sigma\| = \|\mathbb{E}\left[(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\right]\| \leq \mathbb{E}\left[\|(\mathbf{A}_t - \mathbf{A}) \otimes (\mathbf{A}_t - \mathbf{A})\|\right] = \mathbb{E}\left[\|\mathbf{A}_t - \mathbf{A}\|^2\right] = \frac{\sigma^2}{s}$$

Corollary 4. *Suppose we run Algorithm 1 with* $2\sqrt{\beta} \in [\lambda_2, \lambda_1)$. *Assume that* $\|\mathbf{w}_0\| = 1$ *and* $\left|\mathbf{u}_1^T \mathbf{w}_0\right| \geq 1/2$. *For any* $\delta \in (0,1)$ *and* $\epsilon \in (0,1)$, *if*

$$T = \frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}} \log\left(\frac{32}{\delta\epsilon}\right), \quad s \geq \frac{256\sqrt{d}\sigma^2 T}{(\lambda_1^2 - 4\beta)\delta\epsilon} = \frac{256\sqrt{d}\sqrt{\beta}\sigma^2}{(\lambda_1^2 - 4\beta)^{3/2}\delta\epsilon} \log\left(\frac{32}{\delta\epsilon}\right),$$

then with probability at least $1 - 2\delta$, $1 - (\mathbf{u}_1^T \mathbf{w}_T)^2 \leq \epsilon$.

This means no matter what the variance is, we can converge at the same rate as the deterministic setting as long as we can compute mini-batches quickly (think many parallel workers)

Stochastic PCA: Mini-batch + M

A drawback: The required variance decreases as a function of ε….so if error decreases we need to increase mini-batch size *s*.

Stochastic PCA: Mini-batch + M

A drawback: The required variance decreases as a function of ε....so if error decreases we need to increase mini-batch size *s*.

We assume we have $\mathcal{O}(s)$ workers computing in $\mathcal{O}(1)$ time.

We could exhaust the resources of a parallel cluster.

So we need to reduce variance.

Stochastic PCA: VR Power + M

Shamir's (2015) variance reduction update:

$$\mathbf{A}\mathbf{w}_t + (\mathbf{A}_t - \mathbf{A})(\mathbf{w}_t - \tilde{\mathbf{w}})$$

$\tilde{w}$ is a normalized iterate for which we know $A\tilde{w}$

Note: This implies we have the target matrix *A*. In the offline setting we occasionally have access to a full pass.

Stochastic PCA: VR Power + M

A slight variation:

$$\left[\mathbf{A} + (\mathbf{A}_t - \mathbf{A})(I - \tilde{\mathbf{w}}\tilde{\mathbf{w}}^T)\right] \mathbf{w}_t = \mathbf{A}\mathbf{w}_t + (\mathbf{A}_t - \mathbf{A})(I - \tilde{\mathbf{w}}\tilde{\mathbf{w}}^T)\mathbf{w}_t.$$

Better when measuring progress in angle rather than L2 error

Stochastic PCA: VR Power + M

A slight variation:

$$\left[\mathbf{A} + (\mathbf{A}_t - \mathbf{A})(I - \tilde{\mathbf{w}}\tilde{\mathbf{w}}^T)\right]\mathbf{w}_t = \mathbf{A}\mathbf{w}_t + (\mathbf{A}_t - \mathbf{A})(I - \tilde{\mathbf{w}}\tilde{\mathbf{w}}^T)\mathbf{w}_t.$$

Better when measuring progress in angle rather than L2 error

Also generally lower variance because for all unit vectors we have

$$\|\mathbf{w}_t - \tilde{\mathbf{w}}\| \geq \|(I - \tilde{\mathbf{w}}\tilde{\mathbf{w}}^T)\mathbf{w}_t\|.$$

Stochastic PCA: VR Power + M

Algorithm 2 VR Power Method with Momentum (**VR Power+M**)

Require: Initial point $\mathbf{w}_0$, Number of Iterations T, Batch size s, Momentum parameter β

$\mathbf{w}_{-1} \leftarrow \mathbf{0}$

for k = 1 **to** K **do**

$\tilde{\mathbf{v}} \leftarrow \mathbf{A}\tilde{\mathbf{w}}_k$ (Usually there is no need to materialize $\mathbf{A}$ in practice).

for t = 1 **to** T **do**

Generate a mini-batch of i.i.d. samples $B = \{\tilde{\mathbf{A}}_{t_1}, \cdots, \tilde{\mathbf{A}}_{t_s}\}$

Update: $\alpha \leftarrow \mathbf{w}_t^T \tilde{\mathbf{w}}_k$, $\mathbf{w}_{t+1} \leftarrow \frac{1}{s}\sum_{i=1}^{s} \tilde{\mathbf{A}}_{t_i}(\mathbf{w}_t - \alpha\tilde{\mathbf{w}}_k) + \alpha\tilde{\mathbf{v}} - \beta\mathbf{w}_{t-1}$

Normalization: $\mathbf{w}_t \leftarrow \mathbf{w}_t/\|\mathbf{w}_{t+1}\|, \mathbf{w}_{t+1} \leftarrow \mathbf{w}_{t+1}/\|\mathbf{w}_{t+1}\|$.

end for

$\tilde{\mathbf{w}}_{k+1} \leftarrow \mathbf{w}_T$.

end for

return $\mathbf{w}_K$

Stochastic: VR Power + M

Theorem 5. *Suppose we run Algorithm 2 with* $2\sqrt{\beta} \in [\lambda_2, \lambda_1)$ *and a initial unit vector* $\mathbf{w}_0$ *such that* $1 - (\mathbf{u}_1^T \mathbf{w}_0)^2 \leq \frac{1}{2}$. *For any* $\delta, \epsilon \in (0,1)$, *if*

$$T = \frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}} \log\left(\frac{1}{c\delta}\right), \quad s \geq \frac{32\sqrt{d}\sqrt{\beta}\sigma^2}{c(\lambda_1^2 - 4\beta)\delta} \log\left(\frac{1}{c\delta}\right), \tag{7}$$

then after $K = \mathcal{O}(\log(1/\epsilon))$ *epochs, with probability at least* $1 - \log\left(\frac{1}{\epsilon}\right)\delta$, *we have* $1 - (\mathbf{u}_1^T \tilde{\mathbf{w}}_K)^2 \leq \epsilon$, *where* $c \in (0, 1/16)$ *is a numerical constant.*

Stochastic: VR Power + M

Theorem 5. *Suppose we run Algorithm 2 with* $2\sqrt{\beta} \in [\lambda_2, \lambda_1)$ *and a initial unit vector* $\mathbf{w}_0$ *such that* $1 - (\mathbf{u}_1^T \mathbf{w}_0)^2 \leq \frac{1}{2}$. *For any* $\delta, \epsilon \in (0,1)$, *if*

$$T = \frac{\sqrt{\beta}}{\sqrt{\lambda_1^2 - 4\beta}} \log\left(\frac{1}{c\delta}\right), \quad s \geq \frac{32\sqrt{d}\sqrt{\beta}\sigma^2}{c(\lambda_1^2 - 4\beta)\delta} \log\left(\frac{1}{c\delta}\right), \tag{7}$$

then after $K = \mathcal{O}(\log(1/\epsilon))$ *epochs, with probability at least* $1 - \log\left(\frac{1}{\epsilon}\right)\delta$, *we have* $1 - (\mathbf{u}_1^T \tilde{\mathbf{w}}_K)^2 \leq \epsilon$, *where* $c \in (0, 1/16)$ *is a numerical constant.*

Same convergence rate as deterministic algorithm.

Batch size doesn't depend on error. So we can employ a parallel cluster of fixed size.

Stochastic: VR Power + M

Using the VR Power + Momentum method, together with a parallel mini-batch computing cluster, it’s faster than any non-momentum method

Stochastic: VR Power + M

Using the VR Power + Momentum method, together with a parallel mini-batch computing clusterm, it’s faster than any non-momentum method

In terms of iterations, the momentum methods achieve accelerated linear convergence

A comparison of convergence rates

Setting	Algorithm	Number of Iterations	Batch Size	Reference
Deterministic	Power	$\mathcal{O}\left(\frac{1}{\Delta}\cdot\log\left(\frac{1}{\epsilon}\right)\right)$	n	[GVL12]
	Lanczos	$\mathcal{O}\left(\frac{1}{\sqrt{\Delta}}\cdot\log\left(\frac{1}{\epsilon}\right)\right)$	n	[GVL12]
	Power+M	$\mathcal{O}\left(\frac{1}{\sqrt{\Delta}}\cdot\log\left(\frac{1}{\epsilon}\right)\right)$	n	**This paper**
Online	Oja's	$\mathcal{O}\left(\frac{\sigma^2}{\Delta^2}\cdot\frac{1}{\epsilon}+\frac{1}{\sqrt{\epsilon}}\right)$	$\mathcal{O}(1)$	[Jai+16]
	Mini-batch Power+M	$\mathcal{O}\left(\frac{1}{\sqrt{\Delta}}\cdot\log\left(\frac{1}{\epsilon}\right)\right)$	$\mathcal{O}\left(\frac{\sqrt{d}\sigma^2}{\Delta^{3/2}}\cdot\frac{1}{\epsilon}\log\left(\frac{1}{\epsilon}\right)\right)$	**This paper**
Offline	VR-PCA	$\mathcal{O}\left(\frac{r^2}{\Delta^2}\cdot\log\left(\frac{1}{\epsilon}\right)\right)$	$\mathcal{O}(1)$	[Sha15]
	VR Power+M	$\mathcal{O}\left(\frac{1}{\sqrt{\Delta}}\cdot\log\left(\frac{1}{\epsilon}\right)\right)$	$\mathcal{O}\left(\frac{\sqrt{d}\sigma^2}{\Delta^{3/2}}\right)$	**This paper**