Random Osborne Algorithm for Matrix Balancing

Optimal transport report

Alexi Canesse[1,2]

Optimal transport course
Part of the MVA program at ENS Paris-Saclay.

Computer Science and Mathematics
École Normale Supérieur Paris-Saclay
École Normale Supérieur de Lyon
Paris, France
15th January 2024

1 alexi.canesse@ens-lyon.fr, ENS de Lyon, France
2 alexi.canesse@ens-paris-saclay.fr, ENS Paris-Saclay, France

Table of content

Abstract (½ page): What problems is studied? Why is it relevant? What solutions is proposed? Which contributions (theory, numerics, etc)?

1 Introduction (3 pages)

[AP23]

1.1 Presentation of the problem

The matrix balancing problem arises from the recognition that matrices encountered in practical applications may have varying scales across rows and columns. These imbalances can lead to numerical instability and adversely impact the accuracy of numerical computations such as eigenvalues/eigenvectors decomposition [CD00; Che01] or solving linear systems [Che01], . Matrix balancing algorithms aim to address these issues by adjusting the scaling of rows and columns, thereby enhancing the overall numerical properties of the matrix.

Other problems can be reduced to matrix balancing, which gives direct applications. For instance, in optimal transport with problems where decision variables are probability distributions [Alt22]; the approximating Min-Mean-Cycle which can be solved in near-linear runtime in the number of edges [AP22] using results from matrix balancing and in particular, the main result of the studied article.

Notation 1.1 *Let c and r respectively be the column-wise sum and the row-wise sum of matrices ie.*

$$c : \begin{cases} \mathcal{M}_{n,m}(\mathbb{K}) & \to & \mathbb{K} \\ A & \mapsto & A^\top \mathbf{1} \end{cases} \quad \text{and} \quad r : \begin{cases} \mathcal{M}_{n,m}(\mathbb{K}) & \to & \mathbb{K} \\ A & \mapsto & A\mathbf{1}. \end{cases}$$

Definition 1.1 *Let $A \in \mathcal{M}_n(\mathbb{R}_+)$ be a non negative square matrix, $\varepsilon \geq 0$ and $k \in \mathbb{N}^\star$. The matrix A is $(\varepsilon, k$-balanced if*

$$\frac{||c(A) - r(A)||_k}{\sum_{i,j} a_{i,j}} \leq 0. \tag{1}$$

Furthermore, if A is $(0, k)$-balanced, we say that A is balanced.

Definition 1.2 *The ε, k-**approximate matrix balancing problem** is: given a square non-negative matrix $K \in \mathcal{M}_n(\mathbb{R}_+)$, $\varepsilon \geq 0$ and $k \in \mathbb{N}^\star$, find a vector $x \in \mathbb{R}^n$ such that $\mathcal{D}(e^x)K\mathcal{D}(e^x)$ is (ε, k)-balanced.*

1.2 Related work

Iterative algorithms The Osborn algorithm, introduced by Osborn in [Osb60] and later discussed in [PR71], is implemented in Scipy as a default matrix balancing method. Other iterative methods have been introduces. Notably, the Sinkhorn-Knopp algorithm [SK67], a special case of Bregman's balancing method [LS81], iteratively rescales

each row and column until convergence. However, it converges linearly, which is impractical for large and sparse matrices, as discussed by Soules et al. in [Sou91]. Parlett introduced an approach in [PR71] to approximately balance matrices by ensuring the diagonal consists only of powers of 2. The advantage of this method is the elimination of floating-point errors when computing the balanced matrix on base-two computers, making it "good enough" in practice. Some efforts are also made to compare balancing algorithms on real use cases [SZ90].

Issues in Matrix Balancing While matrix balancing is generally effective, there are corner cases where it can worsen the conditioning of matrices. Watkins et al. highlighted some of these cases in [Wat06]. Additionally, James et al. explained, in [JLL14], these issues and proposed modifications to LAPACK to mitigate them.

Theoretical bounds [KK96] establishes an upper bound on the norm of the scaling factors and presents a polynomial-time complexity bound for the computation of scaling factors with a prescribed accuracy.

Characterizations of Non-Negative Balancable Matrices [Osb60] and [Eav+85] offer characterizations of non-negative balancable matrices, contributing to the theoretical understanding of the properties of such matrices. Additionally, [KKS97] provides insights into the complexity aspects of this problem.

Tensor Matrix Balancing Sugiyama et al. extended matrix balancing to tensors in [SNT17], providing a perspective on balancing operations in multi-dimensional spaces. This work contributes to the broader understanding of matrix balancing techniques applied beyond traditional matrices.

1.3 Contributions of the paper

Their main contribution is theorem 2.2. It exhibits a variant of Osborn's algorithm with near-linear runtime in the input sparsity. It also shows that improving the runtime dependence in ε can be improve from ε^{-2} to ε^{-1} without an additional factor n.

Theorem 1.1 *Let $K \in \mathcal{M}_n(\mathbb{R}_+)$ be a balanceable non negative square matrix and $\varepsilon \geq 0$. Random Osborn solves $(\varepsilon, 1)$-approximate matrix balancing problem in T operations where there exists $c > 0, \delta > 0$ such that*

$$\mathbb{E}(T) = \mathcal{O}\left(\frac{m}{\varepsilon} \min\left\{\frac{1}{\varepsilon}; d\right\} \log \kappa\right) \quad \textit{and} \quad \mathbb{P}\left(T \leq c\frac{m}{\varepsilon} \min\left\{\frac{1}{\varepsilon}; d\right\} \log \kappa \log \frac{1}{\delta}\right) \geq 1 - \delta$$

where m is the number of nonzero entries in K, d is the diameter of the graph associated to K and $\kappa = \sum_{i,j} K_{i,j} / \min_{i,j} K_{i,j}$.

1.4 Our contributions

numerics? limits? optimization ?

2 Main body (10 pages)

2.1 Notations

2.2 Presentation of the method

```
osborn(K, ε):
    x = 0 ∈ ℝ^n
    while not(is_balanced(𝔻(e^x)K𝔻(e^{-x}), ε)):
        Choose k ∈ [n] # This is where variants differ
        x += (log(c_k(𝔻(e^x)K𝔻(e^{-x}))) - log(r_k(𝔻(e^x)K𝔻(e^{-x}))))/2
    return x
```

There are *many* way to choose the next coordinate to update and hence many variants of the algorithm. The article focuses on four of them:

- **Cyclic Osborn** Cycle through the coordinates. (*eg.* 1, 2, 3, 1, 2, 3, 1, ...).
- **Random-Reshuffle Cyclic Osborn** Cycle through the coordinates using a new random permutation for each cycle. (*eg.* 2, 1, 3, 1, 2, 3, 1, 3, 2, ...).
- **Greedy Osborn** Choose k where the imbalance is maximal *eg.*

$$k = \text{argmax}_k \left| \sqrt{r_k(\mathbb{D}(e^x)K\mathbb{D}(e^{-x}))} - \sqrt{c_k(\mathbb{D}(e^x)K\mathbb{D}(e^{-x}))} \right|.$$

- **Random Osborn** Uniformly sample k independently between each call.

2.2.1 Implementation

The implementations were carried out in Python, providing us with a transparent and customizable framework for thorough experimentation. The source code for our implementations is available on GitHub. The implementation uses sparse representation for matrices and aimed to provide an implementation using data structures as close as possible as those used to compute the theoretical results. However, we were not able to find a detailed presentation of those and had to do our best to match the article. We did not give any focus on bit complexity. The authors recommanded a log-domain implementation to enable a logarithmic bit-complexity for some variants. However, we decided not to follow through in order to increase speed because we did not worry about bit-complexity.

At each update of x, we update the balanced matrix by multiplying the corresponding row and column. With the sparse representation, this operation is proportional to m. It is even $\mathcal{O}(m/n)$ on average! To compute the optimization function, the sum of elements in DAD^{-1} is computed in $\mathcal{O}(m)$ thanks to the sparse representation and $||c(A) - r(A)||_1$ is also computed in $\mathcal{O}(m)$. Indeed, we use $||c(A) - r(A)||_1 = \sum_j \left|\sum_i (DAD^{-1} - (DAD^{-1})^\top)_{i,j}\right|$. **Those techniques are not presented in the article and we had to come up with on our own.**

2.3 Matrix balancing as a convex optimization problem

A key part of the proofs of convergence given on the paper and other works rely mainly on a seeing the matrix balancing problem as an optimization problem. Let $\Phi : x \mapsto \log \sum_{i,j} e^{x_i - x_j} K_{i,j}$. The gradient of this function is

$$\nabla \Phi(x) = \frac{r\left(\mathcal{D}(e^x) K \mathcal{D}(e^x)\right) - c\left(\mathcal{D}(e^x) K \mathcal{D}(e^x)\right)}{\sum_{i,j} \left(\mathcal{D}(e^x) K \mathcal{D}(e^x)\right)_{i,j}}.$$

We can clearly notice a connection with eq. (1) and see that x is a solution to the ε, k-matrix balancing problem if and only if $||\nabla \Phi(x)||_k \leq \varepsilon$. The function Φ is convex and approximately solving the convex optimization problem $\min_x \Phi(x)$ solves the matrix balancing problem.

Alexandre d'Aspremont says in his MVA course on convex optimization that writing a problem as a convex optimization problem is almost solving it, which shows how important this statement is.

2.4 Theoretical guarantees

Lemma 2.1 *A matrix $K \in \mathcal{M}_n(\mathbb{R}_+^\star)$ is balanceable if and only if its associated graph is strongly connected [Osb60].*

Theorem 2.1 *Let $K \in \mathcal{M}_n(\mathbb{R}_+)$ be a balanceable non negative square matrix and $\varepsilon \geq 0$. Greedy Osborn solves $(\varepsilon, 1)$-approximate matrix balancing problem in*

$$\mathcal{O}\left(\frac{n^2}{\varepsilon} \min\left\{\frac{1}{\varepsilon}; d\right\} \log \kappa\right)$$

where d is the diameter of the graph associated to K and $\kappa = \sum_{i,j} K_{i,j} / \min_{i,j} K_{i,j}$.

Theorem 2.2 *Let $K \in \mathcal{M}_n(\mathbb{R}_+)$ be a balanceable non negative square matrix and $\varepsilon \geq 0$. Random Osborn solves $(\varepsilon, 1)$-approximate matrix balancing problem in T operations where there exists $c > 0, \delta > 0$ such that*

$$\mathbb{E}(T) = \mathcal{O}\left(\frac{m}{\varepsilon} \min\left\{\frac{1}{\varepsilon}; d\right\} \log \kappa\right) \quad \text{and} \quad \mathbb{P}\left(T \leq c \frac{m}{\varepsilon} \min\left\{\frac{1}{\varepsilon}; d\right\} \log \kappa \log \frac{1}{\delta}\right) \geq 1 - \delta$$

where m is the number of nonzero entries in K, d is the diameter of the graph associated to K and $\kappa = \sum_{i,j} K_{i,j} / \min_{i,j} K_{i,j}$.

Theorem 2.3 *Let $K \in \mathcal{M}_n(\mathbb{R}_+)$ be a balanceable non negative square matrix and $\varepsilon \geq 0$. Random cyclic Osborn solves $(\varepsilon, 1)$-approximate matrix balancing problem in T operations where there exists $c > 0, \delta > 0$ such that*

$$\mathbb{E}(T) = \mathcal{O}\left(\frac{mn}{\varepsilon} \min\left\{\frac{1}{\varepsilon}; d\right\} \log \kappa\right) \quad \text{and} \quad \mathbb{P}\left(T \leq c \frac{mn}{\varepsilon} \min\left\{\frac{1}{\varepsilon}; d\right\} \log \kappa \log \frac{1}{\delta}\right) \geq 1 - \delta$$

where m is the number of nonzero entries in K, d is the diameter of the graph associated to K and $\kappa = \sum_{i,j} K_{i,j} / \min_{i,j} K_{i,j}$.

2.5 Numerics

it indeed converges even with high sparsity (cf theorem)

LOOK IN schneider1990comparative https://sci-hub.st/10.2307/171357 for inspiration on applications

per iteration runtime (1.3)

2.5.1 Data generation

In order to conduct numerical experiments, it is useful to be able to generate matrices with given parameters such as value of κ and sparsity. For that, we start by generating a (n, n) random matrix with m non-zero values using `scipy.sparse.random`. Then we assign values to those non-zero entries according to a EXPONENTIAL distribution of scale 1. We then want to modify those values to get the expected value for κ. To do so, we add a value x to all non-zeros entries. If the generated matrix was K, to find x, we solve

$$\kappa = \frac{\sum_{i,j} K_{i,j} + m \times x}{K_{\min} + x}.$$

We then find that the value to add is

$$x = \frac{\kappa \times K_{\min} - \sum_{i,j} K_{i,j}}{m - \kappa}.$$

Sometimes, x is negative and K end up not being non-negative and we just start again from the start. Finally, we want the matrix to be balanceable. Hence, we start again until its associated graph is strongly connected (lemma 2.1).

2.5.2 Complexity as a function of n

2.5.3 Complexity as a function of m

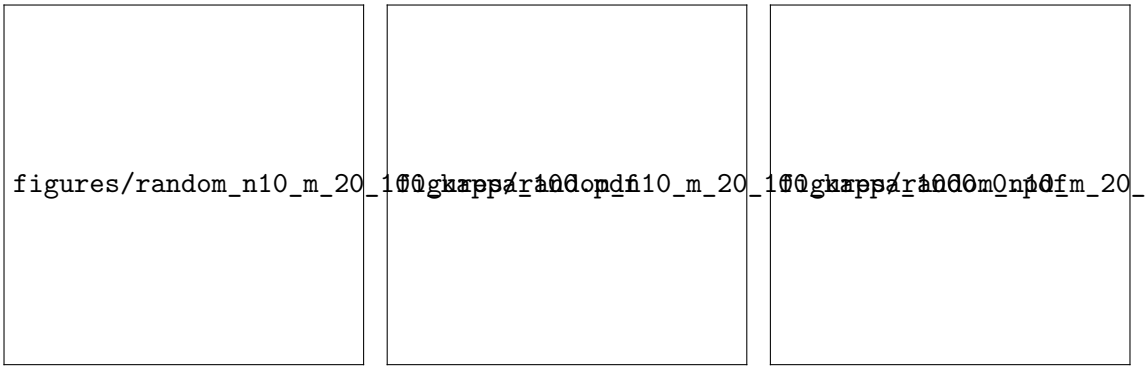
2.5.4 Complexity as a function of *kappa*

2.5.5 Sparsity

We conducted numerical experiments to investigate the behaviour of Osborn's algorithm in the presence of varying numbers of zero entries within randomly generated matrices. The goal is to experimentally see how tight the bounds provided in the article are.

(a) $\kappa = 100$ (b) $\kappa = 10^3$ (c) $\kappa = 10^6$

Figure 1: Plot of $\text{avg}(\text{time}/(n \min(d, 1/\varepsilon) \log(\kappa)/\varepsilon))$ as a function of m, using the random cyclic variation with $\varepsilon = 10^{-4}$. Matrices are generated according to the framework presented in section 2.5.1. For $n = 10$ and values of m in $[\![20, n]\!]$, we generate 100 matrices of size n and sparsity m for different fixed values of κ.



(a) $\kappa = 100$ (b) $\kappa = 10^3$ (c) $\kappa = 10^6$

Figure 2: Plot of avg(time/(min$(d, 1/\varepsilon)\log(\kappa)/\varepsilon)$) as a function of m, using the random variation with $\varepsilon = 10^{-4}$. Matrices are generated according to the framework presented in section 2.5.1. For $n = 10$ and values of m in $[\![20, n]\!]$, we generate 100 matrices of size n and sparsity m for different fixed values of κ.

Use obeservation 2.5 to compare with other convex optim algorithms.

Look at per iteration runtime (5.2)

3 Conclusion and perspective

Summary of the result obtained: pros and cons (limitation, problems, error in the articles, etc) Possible improvement/extension

4 Connexion with the course

MANDATORY SECTION:. What are the notions/results/algorithms presented in the course that are used or related to the one presented in this paper?

Todo list

References

[Alt22] Jason M Altschuler. "Transport and Beyond: Efficient Optimization over Probability Distributions". PhD thesis. Massachusetts Institute of Technology, 2022.

[AP22] Jason M Altschuler and Pablo A Parrilo. "Approximating Min-Mean-Cycle for low-diameter graphs in near-optimal time and memory". In: *SIAM Journal on Optimization* 32.3 (2022), pp. 1791–1816.

[AP23] Jason M Altschuler and Pablo A Parrilo. "Near-linear convergence of the Random Osborne algorithm for Matrix Balancing". In: *Mathematical Programming* 198.1 (2023), pp. 363–397.

[CD00] Tzu-Yi Chen and James W Demmel. "Balancing sparse matrices for computing eigenvalues". In: *Linear algebra and its applications* 309.1-3 (2000), pp. 261–287.

[Che01] Tzu-Yi Chen. *Preconditioning sparse matrices for computing eigenvalues and solving linear systems of equations*. University of California, Berkeley, 2001.

[Eav+85] B Curtis Eaves et al. "Line-sum-symmetric scalings of square nonnegative matrices". In: *Mathematical Programming Essays in Honor of George B. Dantzig Part II* (1985), pp. 124–141.

[JLL14] Rodney James, Julien Langou and Bradley R Lowery. "On matrix balancing and eigenvector computation". In: *arXiv preprint arXiv:1401.5766* (2014).

[KK96] Bahman Kalantari and Leonid Khachiyan. "On the complexity of nonnegative-matrix scaling". In: *Linear Algebra and its applications* 240 (1996), pp. 87–103.

[KKS97] Bahman Kalantari, Leonid Khachiyan and Ali Shokoufandeh. "On the complexity of matrix balancing". In: *SIAM Journal on Matrix Analysis and Applications* 18.2 (1997), pp. 450–463.

[LS81] B Lamond and Neil F Stewart. "Bregman's balancing method". In: *Transportation Research Part B: Methodological* 15.4 (1981), pp. 239–248.

[Osb60] EE Osborne. "On pre-conditioning of matrices". In: *Journal of the ACM (JACM)* 7.4 (1960), pp. 338–345.

[PR71] Beresford N Parlett and Christian Reinsch. "Balancing a matrix for calculation of eigenvalues and eigenvectors". In: *Handbook for Automatic Computation* 2 (1971), pp. 315–326.

[SK67] Richard Sinkhorn and Paul Knopp. "Concerning nonnegative matrices and doubly stochastic matrices". In: *Pacific Journal of Mathematics* 21.2 (1967), pp. 343–348.

[SNT17] Mahito Sugiyama, Hiroyuki Nakahara and Koji Tsuda. "Tensor balancing on statistical manifold". In: *International Conference on Machine Learning*. PMLR. 2017, pp. 3270–3279.

[Sou91] George W Soules. "The rate of convergence of Sinkhorn balancing". In: *Linear algebra and its applications* 150 (1991), pp. 3–40.

[SZ90] Michael H Schneider and Stavros A Zenios. "A comparative study of algorithms for matrix balancing". In: *Operations research* 38.3 (1990), pp. 439–455.

[Wat06] David S Watkins. “A case where balancing is harmful”. In: *Electron. Trans. Numer. Anal* 23 (2006), pp. 1–4.