A Water-filling Algorithm Maximizing the Volume of Submatrices Above the Rank
Claude Petit, Aline Roumy, Thomas Maugey

To cite this version:
Claude Petit, Aline Roumy, Thomas Maugey. A Water-filling Algorithm Maximizing the Volume of Submatrices Above the Rank. EUSIPCO 2023 - 31st European Signal Processing Conference, Sep 2023, Helsinki, Finland. pp.1-5. hal-04132343

HAL Id: hal-04132343
https://inria.hal.science/hal-04132343
Submitted on 19 Jun 2023

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Distributed under a Creative Commons Attribution 4.0 International License
A Water-filling Algorithm Maximizing the Volume of Submatrices Above the Rank

Claude Petit  
Inria  
Rennes, France  
claude.petit@inria.fr

Aline Roumy  
Inria  
Rennes, France  
aliou.roumy@inria.fr

Thomas Maugey  
Inria  
Rennes, France  
thomas.maugey@inria.fr

Abstract—In this paper, we propose an algorithm to extract, from a given rectangular matrix, a submatrix with maximum volume, whose number of extracted columns is greater than the initial matrix rank. This problem arises in compression and summarization of databases, recommender systems, learning, numerical analysis or applied linear algebra. We use a continuous relaxation of the maximum volume matrix extraction problem, which admits a simple and closed form solution: the nonzero singular values of the extracted matrix must be equal. The proposed algorithm extracts matrices with singular values, which are close to be equal. It is inspired by a water-filling technique, traditionally dedicated to equalization strategies in communication channels. Simulations show that the proposed algorithm performs better than sampling methods based on determinantal point processes (DPPs) and achieves similar performance as the best known algorithm, but with a lower complexity.

Index Terms—Matrix volume, column subset selection

I. INTRODUCTION

In this paper, we address the problem of column subset selection (CSSP) [1], [13], [17] with the following framework: the elements of a dataset of cardinal $n$ are characterized by a real vector of $d$ coordinates, so that the dataset can be modeled by a matrix $A \in \mathbb{R}^{d \times n}$. Column vectors represent the elements, called items, rows represent features describing the items via a latent space isomorphic to $\mathbb{R}^d$. To allow the features to be comparable, we suppose that each column of $A$ is normalized. The problem is to select $m$ columns ($d \leq m \leq n$) of the initial data matrix $A$ (assumed to be of rank $d$, much lower than $n$ and $m$), thus forming a rectangular submatrix $B \in \mathbb{R}^{d \times m}$. Moreover, the criterion used to select the submatrix is the maximization of the volume of $B$. This framework can be seen as a compression method which samples the population of items, while keeping all the features characteristics.

Maximizing the volume occurs in many applications: some search engines data summarization algorithms rely on extracting a small representative subset from a given dataset [3]. Sampling data requires to select items by maximizing a cost function related to a quantity of information [6]. Compression of databases can be done by selecting a subset of the initial base, keeping as much information as possible [1], [13]. In each of these applications, the choice of the subset is performed by maximizing the volume of an extracted submatrix $B$; the extraction can be performed on rows or columns, reflecting either a feature selection, either a sampling process, and the submatrices can be transformed into a square Gram matrix $B' B$ or a covariance matrix $B B'$, in which case the volume is given by the determinant of the underlying symmetric positive semidefinite (SPSD) matrices. In numerical linear algebra, to optimize the condition number or to build a good low rank approximation of a square matrix, a popular method is to extract square submatrices by selecting same rows and columns, leading to pseudo-skeleton, cross or CUR approximations [4], [8], [9], [12]; there, the volume maximization relies on QR, LU or singular value decomposition. All methods based on determinantal point processes (DPPs) form a Gram matrix $A' A$, from which a square submatrix of maximum determinant is extracted [10]. All methods introduced above assume that the number of samples is below the rank. The problem and intuitions are different when the number of samples is above the rank. A method exists [14] but is greedy and does not really draw good intuitions on what is the best way to sample the matrix.

In this work, we propose a new algorithm relying on a water-filling technique [5], to extract maximum volume rectangular submatrices, under the assumption that the number of extracted columns is greater than the rank. Our method is one-shot, and is inspired by a continuous relaxation of the volume maximization problem, whose solution is a matrix with equal singular values. Therefore, we propose a water-filling approach [5], that tends to equalize the singular values of the submatrix. Interestingly, the achieved complexity is lower than the existing greedy algorithm. The overall complexity of the DPP algorithm is in $O(n^3)$ operations [16], that of the greedy algorithm in $O(n m^2)$ [14], while the proposed algorithm is in $O(n d^2)$ operations, which is significantly lower, as $d$ is assumed to be lower than $n$ and $m$.

The paper is organized as follows. In section II, we formalize the several optimization problems. We present some mathematical tools to precise the links between them and deduce an objective function related to the spread of the covariance matrix spectrum. Section III presents how to solve the new problem, describes the algorithm implementation and its complexity, while section IV is dedicated to simulations and comparisons with other methods.

II. PROBLEM FORMULATION AND ANALYSIS

In the following, we fix $n$, $d$, $m$ ($d \leq m \leq n$). We denote $b_i$ the columns of $A \in \mathbb{R}^{d \times n}$ and $B \in \mathbb{R}^{d \times m}$, $B_k = (b_1, ..., b_k)$ for $d \leq k \leq m$. $\Sigma = B B'$ is the covariance matrix and
the Gram matrix of $B$. The notation $B \subset A$ means that
the columns of $B$ form a subset of the columns of $A$. 
$\text{Tr}$ is the trace operator and $\text{Sp}$ denotes the spectrum. $||\cdot||_p$
represents the $p$-norm for $p = 1, 2$.

In the rest of the paper, we further assume that the matrix $B$
is of rank $d$. For such a rectangular matrix $B$ of rank $d$, we
define the volume, as Ben-Isra"el in [2], [17]:
\[
\text{vol}(B) = \prod_{i=1}^{d} \sigma_i = \sqrt{\prod_{i=1}^{d} \lambda_i}
\tag{1}
\]
where $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_d > 0$ are the $d$
strictly positive singular values of $B$, $\lambda_i = \sigma_i^2$ are the $d$
strictly positive eigenvalues of $\Sigma = BB'$ or $G = B'B$. In particular,
as $\Sigma = BB'$ is a square matrix of rank $d$, we have
\[
\text{vol}(B) = \sqrt{\det(BB')}.
\tag{2}
\]

We can now formulate our problem, where we look for a
submatrix of $A$ that maximizes the volume (see Pb. 1).

**Problem 1 (Discrete maximization of the volume).**
\[
\max_{B \subset A, \forall i \ |b_i| = 1} \text{vol}(B)^2 = \max_{B \subset A, \forall i \ |b_i| = 1} \det(BB')
\tag{3}
\]
where the equality follows from (2).

Since Pb. 1 is NP-hard [17], we consider a continuous relaxation
of this problem, where $B$ is an arbitrary matrix with
normalized columns, and is not necessarily a submatrix of $A$.

**Problem 2 (Continuous relaxation of Pb. 1).**
\[
\max_{B : \forall i \ |b_i| = 1} \det(BB') = \max_{B : \text{Tr}(BB') = m} \det(BB')
\tag{4}
\]

**Proof.** To prove this equality, we first solve the problem
defined in the RHS of (4). The Lagrangian is
\[
\mathcal{L}(\lambda, \alpha) = \prod_{i=1}^{d} \lambda_i + \alpha \left( \sum_{i=1}^{d} \lambda_i - m \right).
\tag{5}
\]
The differentiation of $\mathcal{L}$ with respect to each $\lambda_i$
gives
\[
\forall i = 1, \ldots, d, \ \alpha = \prod_{j \neq i} \lambda_j.
\tag{6}
\]
Therefore, all $\lambda_i$ are equal, and the common value is
\[
\forall i, \lambda_i = \frac{1}{d} \sum_{i=1}^{d} \lambda_i = \frac{m}{d}.
\tag{7}
\]
Finally, the maximum square volume is $\text{vol}(B)^2 = \left( \frac{m}{d} \right)^d$, and
is reached for the scalar matrix $BB' = \frac{m}{d} I_d$.

We now show the equality in (4):

- We first show $\max_{B : \forall i \ |b_i| = 1} \det(BB') \leq \max_{B : \text{Tr}(BB') = m} \det(BB')$.
This follows from the fact that,
\[
\forall i, \ |b_i| = 1 \Rightarrow \text{Tr}(BB') = \text{Tr}(B'B) = \sum_i ||b_i||^2 = m.
\tag{8}
\]
- Note that the converse is not true: $\text{Tr}(BB') = m \implies \forall i, \ |b_i| = 1$. Therefore, to show that both maximum are
equal, we show that the maximum value achieved in the RHS
of (4) can be achieved by a matrix $B$ with normalized columns.

More precisely, we show that the optimal scalar matrix $\frac{m}{d} I_d$
can be decomposed into the product of $BB'$, where $B$ has
normalized columns. This is illustrated in Fig. 1.

To show this, we introduce the $2m$ reals: $d_i = 1, 1 \leq i \leq m$
and $e_1 = \ldots = e_d = \frac{m}{d}$, $e_{d+1} = \ldots = e_m = 0$. Since
\[
\sum_{i=1}^{m} d_i = m = \sum_{i=1}^{m} e_i, \text{ and}
\tag{9a}
\forall k < m, \sum_{i=1}^{k} d_i = k \leq \sum_{i=1}^{k} e_i = \min(k, d) \frac{m}{d}
\tag{9b}
\]
((9b) follows from $m \geq d$), from the Horn theorem [11, Th.B.2, p. 302],
there exists a symmetric $m \times m$ real matrix $G$
with eigenvalues $e_i$ and diagonal entries $d_i$. $G$ being
symmetric, it can be decomposed into $G = B'B$, where $B$
satisfies $\forall i, \ |b_i|^2 = d_i = 1$ i.e. $B$ has normalized columns.
Moreover, the $d \times d$ matrix $BB'$ has the same non zero
eigenvalues as $G$ i.e. the eigenvalues are all equal to $\frac{m}{d}$
and is symmetric. So, $BB' = \frac{m}{d} I_d$ (see (10)). We have constructed
a $B$ matrix that maximizes the volume $(BB' = \frac{m}{d} I_d)$
and whose columns are normalized.

**Interpretation:** the continuous relaxation Pb. 2 gives
insights on how to solve the original discrete Pb. 1. More
precisely, we look for a submatrix $B$ of $A$ such that the
covariance matrix $BB'$ is close to a scalar matrix. The latter
constraint is however difficult to implement. Instead, we rely
on the following equivalent statement
\[
BB' = \lambda I_d \iff \text{Sp}(BB') = \left\{ \lambda \right\},
\tag{10}
\]
which states that all eigenvalues should be equal. This leads
to the minimization Pb. 3 proposed in the next section, that
leads to equalize the eigenvalues of the spectrum.

**III. PROPOSED ALGORITHM**

The solution of Pb. 2 is a diagonal matrix with a unique
eigenvalue equal to $\lambda = m/d$. This result gives the key
principle of the proposed algorithm. Indeed, we seek to equalize
the eigenvalues of the extracted matrix $B$, constraining the
spectrum of $BB'$ to be as narrow as possible:

**Problem 3 (Minimization of the spectrum’s spread). Find:**
argmin \sum_{i=1}^{d} (\lambda_i(BB') - \bar{\lambda})^2 \tag{11}

where \( A \) has normalized columns, \( \lambda_i(BB'), 1 \leq i \leq d \) are the \( d \) eigenvalues of \( BB' \), and \( \bar{\lambda} \) is the common value to reach.

We now describe an algorithm able to promote equalization of the eigenvalues of \( BB' \) during the selection of columns, and solve Pb. 3.

A. Initialization
The initialization step consists in producing a square matrix \( B_d \) of size \( d \times d \) using any existing algorithm of volume maximization, for example an exhaustive search if the dimension \( d \) is low enough, or the MAXVOL algorithm [9].

B. Key principle: constrain the spread of the spectrum
\( m - d \) columns are next added to \( B_d \) to form \( B_m \). For the sake of conciseness, we denote the covariance matrices \( \Sigma_d = B_dB_d' \), \( \Sigma_m = B_mB_m' \), and their eigenvalues \( \lambda_1, ..., \lambda_d \), and \( \lambda_1(\Sigma_m), ..., \lambda_d(\Sigma_m) \), respectively. The effect of concatenating to \( B_d \) one or more columns, in order to build the matrix \( B_m \), can be characterized by using the decomposition in terms of rank one operators:

\[
\Sigma_m = \sum_{i=1}^{m} b_i b_i' = \Sigma_d + \sum_{i=d+1}^{m} b_i b_i'
\tag{12}
\]

Concatenating \( m - d \) columns to \( B_d \) is thus equivalent to adding \( m - d \) rank one operators to \( \Sigma_d \). From [7], [15], if \( u_i \) is a normalized eigenvector related to \( \lambda_i \), then, for any column \( b \), we have that

\[
\lambda_i \leq \lambda_i(\Sigma_d + bb') \leq \lambda_i + 1
\tag{13}
\]

where equality occurs in the second inequality if and only if \( b \) is colinear to \( u_i \). In other words, the eigenvalues of \( \Sigma_m \) are obtained from those of \( \Sigma_d \) by summing the \( m - d \) contributions from all the rank one perturbations:

\[
\lambda_i(\Sigma_m) = \lambda_i + \epsilon_i, \; i = 1, ..., d
\tag{14}
\]

where \( 0 \leq \epsilon_i \leq m - d \).

Interestingly, a continuous relaxation of Pb. 3 admits a closed form solution. Indeed, consider the problem

\[
\begin{align*}
\min_{\epsilon_i} & \sum_{i=1}^{d} (\lambda_i + \epsilon_i - \bar{\lambda})^2 \\
\text{s.t.} & \sum_{i=1}^{d} \epsilon_i = m - d; \; \epsilon_i \geq 0
\end{align*}
\tag{15}
\]

where the equality constraint follows (8) and \( \sum_{i=1}^{d} \epsilon_i = \text{Tr}(\Sigma_m) - \text{Tr}(\Sigma_d) = m - d \). The Lagrangian is

\[
\sum_{i=1}^{d} (\lambda_i + \epsilon_i - \bar{\lambda})^2 + \alpha \left( \sum_{i=1}^{d} \epsilon_i - m + d \right) - \sum_{i=1}^{d} (\beta_i \epsilon_i)
\]

and the first order conditions are obtained by differentiating \( L \) with respect to \( \epsilon_i, \alpha \) and \( \beta_i \):

\[
\begin{align*}
-\alpha/2 & = \lambda_i - \bar{\lambda} + \epsilon_i - \beta_i, \; \forall i \\
\sum_i \epsilon_i & = m - d \\
\epsilon_i & = 0 \text{ or } \beta_i = 0, \; \forall i
\end{align*}
\tag{16}
\]

The two first equations of (16) give

\[
\begin{align*}
\bar{\lambda} - \alpha & > \lambda_i \quad & \text{if } \beta_i > 0, \\
\epsilon_i = (\bar{\lambda} - \alpha) - \lambda_i & \quad & \text{if } \epsilon_i > 0.
\end{align*}
\tag{17}
\tag{18}
\]

Therefore, in the continuous relaxation problem the common achieved singular value is \( \tilde{\lambda} - \alpha \).

C. A discrete water-filling solution
The solution (18) to the continuous relaxed problem (15) is similar to the waterfilling solution used in power allocation for multiple channel communication [5]. For the latter problem, the optimum is achieved, when the total power is evenly distributed among each channel. Similarly, in our problem, the goal is to add \( m \) columns to the \( d \times d \) matrix such that the eigenvalues of the obtained matrix are all equal to a common value \( \tilde{\lambda} \), except for the eigenvalues \( \lambda_i \) that are already above the threshold \( \bar{\lambda} \).

For the sake of computational efficiency, we propose an algorithm that allows to select \( m \) columns all at ones. To do so, we order the columns in \( A \) according to their distance to the eigenvectors \( u_i \) of \( \Sigma_d \). More formally,

\[
\forall i, \; A_i = (b_{i1}(i), b_{i2}(i), ..., b_{in}(i)), \quad \text{where}
\]

\[
||u_i - b_{i1}(i)||^2 \leq ||u_i - b_{i2}(i)||^2 \leq ... \leq ||u_i - b_{in}(i)||^2
\tag{19}
\]

Second, to equalize the spectrum, we first determine the continuous increase \( \epsilon_i \) for each eigenvalue with (18), where the threshold is approximated by \( \tilde{\lambda} - \alpha = \frac{m}{d} \). Indeed, there are \( d \) eigenvalues, and due to the norm constraint on the columns of \( A \) and from (8), the sum of the eigenvalues is \( \sum_{i=1}^{d} \lambda_i(\Sigma_m) = m \). Then, we round \( \epsilon_i \) to determine how many columns should contribute to each eigenvalue:

\[
c_i = \left[ \frac{(m-d)\epsilon_i}{||\epsilon_i||_1} \right], \; i = 1, ..., d
\tag{20}
\]

Note that by doing so, we consider that adding a vector \( b \in A_i \) to \( B_d \) modifies the eigenvalues of the new matrices into:

\[
\begin{align*}
\lambda_i(\Sigma_d + bb') & = \lambda_i + 1, \quad \forall j \neq i \\
\lambda_j(\Sigma_d + bb') & = \lambda_j, \quad \forall j \neq i
\end{align*}
\tag{21a}
\tag{21b}
\]

In other words, we neglect the angle between \( b \) and \( u_i \). Then, to meet the constraint \( \sum_{i=1}^{d} c_i = m - d \), the update of \( c_i \) is processed iteratively: \( c_d \) is first evaluated, then \( c_{d-1} \), etc. until the \( (m - d) \) columns are attributed, where we assume that the eigenvalues are ordered by decreasing order \( \lambda_1 \geq \lambda_2, ... \). This allows to favor the smallest eigenvalues, that need to be increased. Finally, when \( c_i \) (the number of vectors in \( A_i \) to be selected) is set, the columns chosen are the closest ones to \( u_i \) in \( A_i \). To favor the smallest eigenvalues, we first select vectors in \( A_d \), and remove the selected columns from all other sets \( A_i \) to \( A_{d-1} \). Then we proceed with \( u_{d-1} \).

D. Implementation and complexity
We now detail the complexity of each step of the proposed algorithm:
We consider a subclass of DPPs, called $L$-ensembles, and well suited to our framework. Indeed it is characterized by a symmetric semi-definite matrix indexed by the elements of the dataset. In our model, this kernel matrix is the Gram matrix $G = B'B$. The DPP is the probability distribution over all subsets of the items such that:

$$
\mathbb{P}(S) = \frac{\det(G)}{\det(A'A + I)} \propto \text{vol}(B)^2
$$

where $G = B'B$ and $B$ is the extracted matrix from $A$ whose column indices belong to $S: B = (b_i)_{i \in S}$. But $G$ is a $n \times n$ matrix of rank $d < n$, so that $\det(B'B) = 0$. For (22) to define a probability distribution, the kernel function $G$ must be modified in such a way that it becomes semi definite positive.

Here we consider two kernel functions. First, $G$ is defined by the kernel function of a modified Gram matrix $G = B'B + \delta I$, where $\delta > 0$ is a real positive number and $G$ is then definite positive.

The second kernel is defined through an embedding into a nonlinear feature function $\phi: \mathbb{R}^d \rightarrow \mathbb{R}^n$ such that the family $(\phi(b_i); i = 1, ..., n)$ is linearly independent. Then, the kernel matrix is defined by the Euclidean dot product

$$
G_{ij} = \langle \phi(b_i), \phi(b_j) \rangle
$$

for each pair of columns $b_i, b_j$.

In addition to ensure linear independency, the function $\phi$ has to reflect the way initial column vectors contribute to the volume of the extracted matrix. So instead of the traditional basis radial function kernel, we choose a cosine kernel which is maximum when the variables are orthogonal and zero when they are colinear. Thanks to the kernel trick, We do not need to explicit the function $\phi$, but just to define $G_{ij} = \cos(b_i, b_j)$.

The implementation of DPPs is made with MATLAB using the code of Alex Kulesza. Our implementation uses also the MAXVOL algorithm [9] during the initialization step. MAXVOL algorithm is available in MATLAB as part of the TT-Toolbox of Ivan Oseledets.

We consider a subclass of DPPs, called $L$-ensembles, and well suited to our framework. Indeed it is characterized by a symmetric semi-definite matrix indexed by the elements of the dataset. In our model, this kernel matrix is the Gram matrix $G = B'B$. The DPP is the probability distribution over all subsets of the items such that:

$$
\mathbb{P}(S) = \frac{\det(G)}{\det(A'A + I)} \propto \text{vol}(B)^2
$$

where $G = B'B$ and $B$ is the extracted matrix from $A$ whose column indices belong to $S: B = (b_i)_{i \in S}$. But $G$ is a $n \times n$ matrix of rank $d < n$, so that $\det(B'B) = 0$. For (22) to define a probability distribution, the kernel function $G$ must be modified in such a way that it becomes semi definite positive.

Here we consider two kernel functions. First, $G$ is defined by the kernel function of a modified Gram matrix $G = B'B + \delta I$, where $\delta > 0$ is a real positive number and $G$ is then definite positive.

The second kernel is defined through an embedding into a nonlinear feature function $\phi: \mathbb{R}^d \rightarrow \mathbb{R}^n$ such that the family $(\phi(b_i); i = 1, ..., n)$ is linearly independent. Then, the kernel matrix is defined by the Euclidean dot product

$$
G_{ij} = \langle \phi(b_i), \phi(b_j) \rangle
$$

for each pair of columns $b_i, b_j$.

In addition to ensure linear independency, the function $\phi$ has to reflect the way initial column vectors contribute to the volume of the extracted matrix. So instead of the traditional basis radial function kernel, we choose a cosine kernel which is maximum when the variables are orthogonal and zero when they are colinear. Thanks to the kernel trick, We do not need to explicit the function $\phi$, but just to define $G_{ij} = \cos(b_i, b_j)$.

The implementation of DPPs is made with MATLAB using the code of Alex Kulesza. Our implementation uses also the MAXVOL algorithm [9] during the initialization step. MAXVOL algorithm is available in MATLAB as part of the TT-Toolbox of Ivan Oseledets.

We consider a subclass of DPPs, called $L$-ensembles, and well suited to our framework. Indeed it is characterized by a symmetric semi-definite matrix indexed by the elements of the dataset. In our model, this kernel matrix is the Gram matrix $G = B'B$. The DPP is the probability distribution over all subsets of the items such that:

$$
\mathbb{P}(S) = \frac{\det(G)}{\det(A'A + I)} \propto \text{vol}(B)^2
$$

where $G = B'B$ and $B$ is the extracted matrix from $A$ whose column indices belong to $S: B = (b_i)_{i \in S}$. But $G$ is a $n \times n$ matrix of rank $d < n$, so that $\det(B'B) = 0$. For (22) to define a probability distribution, the kernel function $G$ must be modified in such a way that it becomes semi definite positive.

Here we consider two kernel functions. First, $G$ is defined by the kernel function of a modified Gram matrix $G = B'B + \delta I$, where $\delta > 0$ is a real positive number and $G$ is then definite positive.

The second kernel is defined through an embedding into a nonlinear feature function $\phi: \mathbb{R}^d \rightarrow \mathbb{R}^n$ such that the family $(\phi(b_i); i = 1, ..., n)$ is linearly independent. Then, the kernel matrix is defined by the Euclidean dot product

$$
G_{ij} = \langle \phi(b_i), \phi(b_j) \rangle
$$

for each pair of columns $b_i, b_j$.

In addition to ensure linear independency, the function $\phi$ has to reflect the way initial column vectors contribute to the volume of the extracted matrix. So instead of the traditional basis radial function kernel, we choose a cosine kernel which is maximum when the variables are orthogonal and zero when they are colinear. Thanks to the kernel trick, We do not need to explicit the function $\phi$, but just to define $G_{ij} = \cos(b_i, b_j)$.

The implementation of DPPs is made with MATLAB using the code of Alex Kulesza. Our implementation uses also the MAXVOL algorithm [9] during the initialization step. MAXVOL algorithm is available in MATLAB as part of the TT-Toolbox of Ivan Oseledets.

![Image](image.png)

**Fig. 2.** Maximum volume comparison of 6 algorithms, as a function of $m$ varying from 5 to 20. The average of 10 independent samples of the logarithm of volume are given in ordinate. Blue dots: two DPPs algorithms (Gram $L$-ensemble and cosine kernel). Green: real maximum by exhaustive search. Grey: uniform random choice. Black: RECT_MAXVOL algorithm of [14]. Red: our algorithm.
The algorithm performs well for the first half values of $m$. When $m$ increases and comes closer to $n$, the possible choices of columns decrease and all methods give approximatively the same results.

**B. Spectrum equalization of the proposed algorithm**

The effect of the algorithm can be visualized in Fig. 4, in the case of random Gaussian i.i.d. matrices. 100 independent samples of an $d \times n$ standard Gaussian matrix $A$ are generated. For each dimension $m$ varying in $[d, n]$ (corresponding to one vertical line), we extract a $d \times m$ random submatrix $C$ and draw its spectrum in grey. We also extract a $d \times m$ submatrix $B$ by using the WaterMaxVol algorithm and draw its spectrum in red. The narrowing of the red tube around the theoretical mean curve $\sqrt{m/d}$ illustrates the performances of the optimization for the first half values of $m$. This results remain licit for any random vectors whose distribution is either spherical symmetric, either coordinates independent.

**V. Conclusion**

In this work, we proposed an algorithm to extract a maximum volume rectangular submatrix of size greater than the dataset matrix rank. Thanks to a continuous relaxation which admits a closed form solution, we propose to equalize the singular values from the extracted matrix. The algorithm can therefore be interpreted as a water-filling technique used in telecommunications to equalize the allocated powers.

Simulations showed that the proposed algorithm outperforms DPPs methods. Moreover, the proposed algorithm, the exhaustive search, and the best known algorithm (RECT_MAX_VOL) have all similar performances. However, our approach achieves the maximum with a much lower complexity.

**REFERENCES**


