Mini-Batch Spectral Clustering

Yufei Han
Symantec Research Labs
yfhan.hust@gmail.com

Maurizio Filippone
EURECOM
maurizio.filippone@eurecom.fr

Abstract

The cost of computing the spectrum of Laplacian matrices hinders the application of spectral clustering to large data sets. While approximations recover computational tractability, they can potentially affect clustering performance. This paper proposes a practical approach to learn spectral clustering based on adaptive stochastic gradient optimization. Crucially, the proposed approach recovers the exact spectrum of Laplacian matrices in the limit of the iterations, and the cost of each iteration is linear in the number of samples. Extensive experimental validation on data sets with up to half a million samples demonstrate its scalability and its ability to outperform state-of-the-art approximate methods to learn spectral clustering for a given computational budget.

1 Introduction

Over the past two decades, spectral clustering has established itself as one of the most prominent clustering methods [1, 2]. The effectiveness of spectral clustering in identifying complex structures in data is a direct consequence of its close connection with kernel machines [3]. Because of this connection, however, it is also apparent that spectral clustering inherits the scalability issues of kernel machines. In spectral clustering, the computational challenge is to determine the spectral properties of the so called Laplacian matrix [4]. Denoting by n the number of samples, storing the Laplacian matrix requires $\mathcal{O}(n^2)$ space while calculating the spectrum of the Laplacian requires $\mathcal{O}(n^3)$ computations.

Several approaches have been proposed to reduce the complexity of spectral clustering, such as employing power methods to identify the principal eigenvectors of the Laplacian [5]. While this approach is exact in the limit of iterations and does not require storing the Laplacian, the complexity is dominated by the iterative multiplication of the Laplacian matrix by vectors, leading to $\mathcal{O}(n^2)$ computations. In order to further reduce this complexity to $\mathcal{O}(n)$, a number of approximations are proposed in the literature. A popular technique based on the Nyström approximation relies on a small set of inducing points to approximate the spectrum of the Laplacian matrix [6]. Other recent approximations which attempt to compress the dataset appear in Yan et al. [7] and Li et al. [8]. These approximations recover tractability and make it possible to apply spectral clustering to large data sets. However, approximations can generally affect the quality of the clustering solution, as we illustrate in the experiments.

This paper proposes a novel iterative way to solve spectral clustering in $\mathcal{O}(n)$, while retaining exactness in the calculation of the spectrum of the Laplacian in the limit of the iterations. Denoting by L the Laplacian matrix, the idea hinges on the possibility to cast the algebraic problem of identifying its principal eigenvectors as the following trace optimization problem

$$\underset{W \in \mathcal{R}^{n \times k}}{\arg\min} \left\{ \mathrm{Tr}\left(-\frac{1}{2} W^\top L W \right) \right\} \qquad \text{subject to} \qquad W^\top W = I \tag{1}$$

We propose to solve the constrained optimization problem by means of stochastic gradient optimization. In view of the orthonormality constraint the elements of W lie on the so called Stiefel manifold, and appealing to theoretical guarantees of convergence of stochastic gradient optimization

on manifolds, we can prove that our proposal computes the exact spectrum of L in the limit of the iterations [9]. In order to simplify the tuning of the optimization procedure, we adapt Adagrad [10] for stochastic gradient optimization on the Stiefel manifold. The novelty of our proposal stems from the use of stochastic linear algebra techniques to compute stochastic gradients in $\mathcal{O}(n)$. This leads to computations of stochastic gradient that require processing of a limited number of columns of the full Laplacian matrix, motivating us to name our proposal Mini-Batch Spectral Clustering.

The results on a variety of clustering problems with n up to $580K$ give credence to the value of our proposal. We can tackle large scale spectral clustering problems achieving the same level of accuracy of the approach that uses the exact spectrum of L at a fraction of the computing time. We also compare against approximate spectral clustering methods and show that approximations lead to faster solutions that are suboptimal compared to what we can achieve with the proposed method, especially for large data sets[1].

Summary of contributions (i) We formulate the solution of spectral clustering as a constrained optimization problem that we solve using adaptive stochastic gradient optimization; (ii) We present a novel way of computing stochastic gradients linearly in the number of data that does not require storing the Laplacian matrix; (iii) We analyze the variance of the proposed estimator of the exact gradient to explain the impact of algorithm parameters. (iv) We demonstrate that our proposal allows us to tackle large-scale spectral clustering problems by reporting results on data sets of size up to $n = 580K$. Crucially, we can achieve clustering solutions of similar accuracy and orders of magnitude faster compared to the approach that computes the exact spectrum of L, and higher accuracy compared to approximate methods at a comparable cost.

2 Spectral clustering

2.1 Background

Define $X = \{\mathbf{x}_1, \ldots, \mathbf{x}_n\}$ to be a set of n samples. The formulation of spectral clustering introduces an undirected graph $\mathcal{G}$ based on X, where the n nodes of $\mathcal{G}$ represent the n input data in X, and the edges are weighted according to a similarity measure between the inputs. The graph $\mathcal{G}$ is expressed by an $n \times n$ adjacency matrix A, where each entry a_{ij} determines the weight associated with the edge connecting inputs i and j. Typically, the elements of the adjacency matrix are defined through off-the-shelf kernel functions, e.g., the Radial Basis Function (RBF) kernel [11].

Spectral clustering attempts to cluster the elements of X by analyzing the spectral properties of the graph $\mathcal{G}$. In particular, the objective of spectral clustering is to partition the graph so as to minimize some graph cut criterion, e.g., the normalized cut [1]. The graph cut problem is generally NP-hard, but its relaxation leads to the definition of the clustering problem as the solution of an algebraic problem [12]. In particular, following the spectral clustering algorithm proposed by Ng et al. [2], the graph Laplacian is defined as a normalized version of the adjacency matrix

$$L = D^{-\frac{1}{2}} A D^{-\frac{1}{2}} \tag{2}$$

where D is the diagonal matrix of the degrees of the n nodes. Spectral clustering represents each data point using the corresponding component of the top k eigenvectors of the Laplacian L, and computes the solution to the clustering problem by applying k-means in this representation. The difficulty in solving the graph cut problem then becomes calculating the spectrum L; this requires $\mathcal{O}(n^3)$ computations and $\mathcal{O}(n^2)$ space, making it prohibitive – if not unfeasible – for large data sets. The aim of this paper is to address this scaling issue of spectral clustering without sacrificing the accuracy of the solution, as explained next.

2.2 Spectral clustering as a constrained optimization problem

The first step to reduce the complexity in finding the top k eigenvectors of L, is to cast this algebraic operation as solving the constrained optimization problem in Eq. 1. This is an optimization problem involving $n \times k$ parameters representing the n components of the top k eigenvectors of L. The objective function rewards maximization of a score that, at convergence, is the sum of the k largest

[1] Code to reproduce all results in the paper is available upon request.

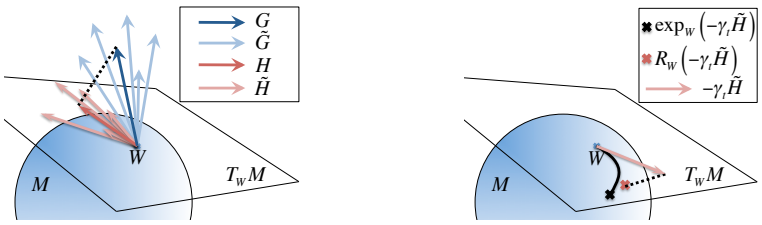

Figure 1: Left: Exact gradient G with a few $\tilde{G}$ and their projection onto the tangent space $T_W M$ of a manifold M at a given W giving H and $\tilde{H}$. Right: Retraction scheme that approximates the exponential map at W.

eigenvalues. The constraint $W^\top W = I$ gives rise to the so called Stiefel manifold in $\mathcal{R}^{n \times d}$; this imposes orthonormality on the columns of W which, at convergence, represent the eigenvectors associated with the k largest eigenvalues.

The constrained optimization problem in Eq. 1 can be solved by formulating standard optimization algorithms to deal with the Riemannian metric on the manifold [13], which typically rely on search operations along geodesics. Alternative schemes, such as the Cayley transform, have been proposed to tackle optimization problems on Riemannian manifolds [14]. All these optimization schemes require calculating the gradient of the objective function

$$G = \nabla_W \mathrm{Tr}\left(-\frac{1}{2} W^\top L W\right) = -LW \tag{3}$$

This costs $\mathcal{O}(n^2)$ and does not require storing L anymore, which is an improvement with respect to computing the full spectrum of L. However, while casting spectral clustering as a constrained optimization problem improves scalability, when n is large it may still take a prohibitive amount of time to be practical. In the next section we present our proposal to reduce the complexity to solve the constrained optimization problem to $\mathcal{O}(n)$ with the guarantee of obtaining the exact solution of the in the limit of the iterations.

3 Mini-Batch Spectral Clustering (MBSC)

The intuition behind our proposal is that it is possible to solve the constrained optimization problem in Eq. 1 relying exclusively on stochastic gradients [9]. By inspecting the expression of the exact gradient in Eq. 3, we show how stochastic linear algebra techniques can be employed to unbiasedly estimate exact gradients ($\mathcal{O}(n^2)$ cost) with stochastic gradients at $\mathcal{O}(n)$ cost.

3.1 Stochastic optimization on Stiefel manifolds

Stochastic gradient optimization on manifolds is based on the notion of Riemannian gradients, which are elements of the tangent space at a given W that determine the direction of steepest increase of the objective function on the manifold. For the Stiefel manifold, the Riemannian gradient is [9]

$$H = (I - WW^\top)G \tag{4}$$

In the case where an unbiased version $\tilde{G}$ of the gradient G is available, namely $\mathrm{E}[\tilde{G}] = G$, an intuitively sensible strategy is to perform stochastic gradient optimization on the manifold. Formally, this would amount in moving along geodesics for a given length based on the noisy version of the gradient. Defining γ_t as the equivalent of the usual step-sizes in stochastic gradient optimization, and $\exp_W()$ as the exponential map at W, the update equation would then be $W' = \exp_W\left(-\gamma_t \tilde{H}\right)$, where $\tilde{H} = (I - WW^\top)\tilde{G}$ is the unbiased Riemannian gradient of the objective function. This approach can be shown to converge to the exact solution of the constrained optimization problem in Eq. 1 [9]. However, computing the exponential map to simulate the trajectory of the solver on the manifold requires solving potentially expensive differential equations.

An alternative that avoids computing the exponential map altogether is to replace this calculation with an approximation $W' = R_W\left(-\gamma_t \tilde{H}\right)$ that is much easier to calculate. If the so called retraction

Algorithm 1: Stochastic Riemannian gradient on Stiefel Manifold using Mini-Batches

`Htilde(`L, p, $N_{\mathbf{r}}$, W`)`
Initialize $\tilde{G} \in R^{n\times k}$ with elements equal to zero
for $i = 1$ **to** $N_{\mathbf{r}}$ **do**
  Draw the components of $\mathbf{r}_i$
  $\tilde{G}$ += $\frac{1}{N_r} L\mathbf{r}_i\mathbf{r}_i^\top W$
end
Return $\tilde{H}$ = $(I - WW^\top)\tilde{G}$

Algorithm 2: Mini-Batch Spectral Clustering

Input: Normalized Laplacian Matrix $L \in R^{n\times n}$, number of clusters k, regularization factor ε
Parameters: Master step length λ, maximum iteration steps T
Output: Cluster labels of each data points
Initialize $W^{(0)} \in R^{n\times k}$ as a random orthonormal matrix
Initialize $M^{(0)} \in R^{n\times k}$ with elements equal to zero
for $t = 1$ **to** T **do**
  $\tilde{H}^{(t)}$ = `Htilde(`L, p, $N_{\mathbf{r}}$, $W^{(t-1)}$`)`
  $M_{ij}^{(t)} = M_{ij}^{(t-1)} + |\tilde{H}_{ij}^{(t)}|^2$
  $\hat{H}_{ij}^{(t)} = \dfrac{\tilde{H}_{ij}^{(t)}}{\varepsilon + \sqrt{M_{ij}^{(t)}}}$
  $W^{(t)} = W^{(t-1)} - \lambda\hat{H}^{(t)}$
  $W^{(t)} = \mathrm{QR}_Q(W^{(t)})$
end
Apply k-means on $W^{(T)}$ to get the cluster labels

R_W satisfies the property that $d(R_W(\delta v), \exp_W(\delta v)) = \mathcal{O}(\delta^2)$, where v is an alement of the tangent space, then it is still possible to prove convergence to the exact solution [9]. A simple and computationally convenient retraction function that satisfies this property is

$$W' = \mathrm{QR}_Q\left(W - \gamma_t(I - WW^\top)\tilde{G}\right) \tag{5}$$

where QR_Q extracts the orthonormal factor Q of a QR decomposition. This simple retraction moves the optimization in the direction of the stochastic Riemannian gradient $\tilde{H} = (I - WW^\top)\tilde{G}$ and applies an orhtonormalization step to ensure that the update is projected back onto the manifold. Under this choice of retraction, we can appeal to the theoretical results in Bonnabel [9] that ensure convergence to the exact solution in the limit of the iterations similarly to standard stochastic gradient optimization. An illustration of the retraction scheme is provided in Fig.1.

3.2 Calculation of stochastic gradients in $\mathcal{O}(n)$

The introduction of stochasticity in the calculation of $\tilde{G}$ follows on from ideas that have been proposed to calculate unbiased stochastic approximations to algebraic quantities, such as traces and log-determinants [15, 16]. In particular, we define a vector $\mathbf{r}$ such that $\mathrm{E}[\mathbf{r}\mathbf{r}^\top] = I$ and we rewrite the expensive matrix product as

$$G = LW = LIW = L\mathrm{E}[\mathbf{r}\mathbf{r}^\top]W = \mathrm{E}[L\mathbf{r}\mathbf{r}^\top W] \tag{6}$$

which suggests that we can replace the exact calculation of LW with the estimator

$$\tilde{G} = \frac{1}{N_r}\sum_{i=1}^{N_r} L\mathbf{r}_i\mathbf{r}_i^\top W \tag{7}$$

The key to making computations linear in n is to define the components of the random vectors $\mathbf{r}_i$ as drawn from the set $\{-p^{-\frac{1}{2}}, 0, +p^{-\frac{1}{2}}\}$ with probabilities $(p/2, 1-p, p/2)$ respectively. It is

straightforward to verify that $\mathrm{E}[\mathbf{r}_i\mathbf{r}_i^\top] = I$, and p can be chosen to enforce any proportion of zeros in the $\mathbf{r}_i$ vectors. With this mechanism to inject stochasticity in the calculation of the gradients, we are effectively ignoring some columns of the matrix L whenever there is a zero in the corresponding positions of the $\mathbf{r}_i$ vectors. This makes it possible to update the parameters W during the solution of the constrained optimization problem in Eq. 1, by only selecting a few columns of the full Laplacian. If the average number of non-zero elements is chosen to be independent of n, the calculation of the stochastic gradient is $\mathcal{O}(n)$, making the proposed iterative solver linear in the number of samples. The memory footprint of the algorithm is a distinctive feature of our proposal; if the degree matrix D is precomputed, calculating the columns of L requires to compute stochastic gradients requires evaluating and normalizing only $\mathcal{O}(n)$ elements of the adjacency matrix A.

Given that only a subset of the columns in L are used at each iteration to calculate stochastic gradients, we term our proposal Mini-Batch Spectral Clustering (MBSC). Instead of defining a probability p to select columns and to repeat this $N_\mathbf{r}$ times, we can fix the number and indices of columns that are selected at each iteration (size of the mini-batch) to be $m = lN_\mathbf{r}$ and interpret $p = l/n$. While this is intuitively sensible, fixing the indices of the mini-batches would violate the property that $\mathrm{E}[\mathbf{r}_i\mathbf{r}_i^\top] = I$. One easy way around this issue is to constantly change the way data are split into mini-batches, e.g., by shuffling the data, and this would recover the property $\mathrm{E}[\mathbf{r}_i\mathbf{r}_i^\top] = I$. Even though it is not the focus of the current paper, we envisage the possibility to develop a distributed version of the proposed MBSC algorithm based on our formulation. The proposed MBSC algorithm is sketched in Algorithms 1 and 2, where, for the sake of clarity, L is assumed to be stored. For memory constrained systems where storing the whole Laplacian matrix is unfeasible, our proposal can easily be adapted to avoid storing it, and we report on the performance of this variant in the experiments.

3.3 Variance of stochastic gradients

Here we are interested in quantifying the impact of the choice of p and $N_\mathbf{r}$ in the calculation of stochastic gradients; to this end, we analyze the variance of the proposed estimator of the exact gradients. Without loss of generality, we focus on the variance of a given column of the stochastic gradient $\tilde{G}$, namely the one associated with a given eigenvector, say $\mathbf{w} := W_{\cdot s}$. Recall that the exact gradient with respect to $\mathbf{w}$ would be $G_{\cdot s} = L\mathbf{w}$ and assume that we use a single vector $\mathbf{r}$ to unbiasedly estimate this as $\tilde{G}_{\cdot s} = L\mathbf{r}\mathbf{r}^\top\mathbf{w}$. The covariance of $\tilde{G}_{\cdot s}$ is

$$\mathrm{cov}\left(L\mathbf{r}\mathbf{r}^\top\mathbf{w}\right) = \mathrm{E}[(L\mathbf{r}\mathbf{r}^\top\mathbf{w})(L\mathbf{r}\mathbf{r}^\top\mathbf{w})^\top] - \left(\mathrm{E}[L\mathbf{r}\mathbf{r}^\top\mathbf{w}]\right)\left(\mathrm{E}[L\mathbf{r}\mathbf{r}^\top\mathbf{w}]\right)^\top$$

After some manipulations, that we leave to the supplementary material, we obtain

$$\mathrm{diag}\left[\mathrm{cov}\left(L\mathbf{r}\mathbf{r}^\top\mathbf{w}\right)\right] = \mathrm{diag}\left[G_{\cdot s}G_{\cdot s}^\top\right] + \left(\frac{1}{p} - 3\right)\mathrm{diag}\left[L\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top))L^\top\right] + \mathrm{diag}\left[LL^\top\right]$$

Given that $\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top)) < I$, then $\mathrm{diag}\left[L\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top))L^\top\right] \leq \mathrm{diag}\left[LL^\top\right]$ when L has positive elements. Also, since $LL^\top$ has positive diagonal entries, we can further bound

$$\mathrm{diag}\left[\mathrm{cov}\left(L\mathbf{r}\mathbf{r}^\top\mathbf{w}\right)\right] \leq \mathrm{diag}\left[G_{\cdot s}G_{\cdot s}^\top\right] + \frac{1}{p}\mathrm{diag}\left[LL^\top\right] \tag{8}$$

The first term contains the square of the components of the exact gradient that will vanish at convergence. The second term depends on the choice of p. We can rewrite the bound in terms of the mini-batch size $m = lN_\mathbf{r}$, for which $p = l/n$, and when $N_\mathbf{r}$ vectors $\mathbf{r}_i$ are used to calculate stochastic gradients as in Eq. 7.

$$\mathrm{diag}\left[\mathrm{cov}\left(L\mathbf{r}\mathbf{r}^\top\mathbf{w}\right)\right] \leq \frac{1}{N_\mathbf{r}}\mathrm{diag}\left[G_{\cdot s}G_{\cdot s}^\top\right] + \frac{n}{lN_\mathbf{r}}\mathrm{diag}\left[LL^\top\right] \tag{9}$$

This reveals that we have two ways of reducing the variance of stochastic gradients; one is to increase $N_\mathbf{r}$ and another is to increase l. Imagine that we fix the mini-batch size $m = lN_\mathbf{r}$; is it better to increase l and reduce $N_\mathbf{r}$, or the other way around? For the second term it does not matter. For the first instead, given that it depends only on $N_\mathbf{r}$, it is clear that we should favor increasing $N_\mathbf{r}$ and reducing l. This entails that we should consider averaging stochastic gradients over several subsets of a mini-batch instead of a few large ones. This result is interesting because in other popular mini-batch approaches increasing the mini-batch size or the number of repetitions is equivalent. In our proposed

Table 1: Summary of the characteristics of the data sets considered in the experiments.

Data set	# of samples	# of features	# of classes	σ
`Pendigits`	10992	16	10	223.61
`Shuttle`	58000	9	7	0.45
`MNIST`	60000	780	10	4.08
`Covtype-I`	100000	54	5	1.15
`Covtype-II`	581012	54	7	1.15

MBSC, because of the nonlinearity of the estimator with respect to the vectors $\mathbf{r}_i$, the bound on the variance shows an unintuitive asymmetry between $N_{\mathbf{r}}$ and l. A further consideration we can make is that this suggestion is most useful during the first phase of the optimization; towards convergence, the first term will be small and dominated by the second term that is inversely proportional to the mini-batch size m.

4 Experiments

Throughout the experiments, we make use of the Radial Basis Function (RBF) adjacency function:

$$a_{ij} = \exp\left(-\frac{\|\mathbf{x}_i - \mathbf{x}_j\|}{\sigma^2}\right) \qquad i \neq j$$

where we set $a_{ij} = 0$ when $i = j$. The RBF adjacency function assigns a large weight to the edge connection inputs that are close in the input space, whereas it assigns a small weight to edges connecting inputs that are far apart. The parameter σ determines the rate of decay of the adjacency function, which can be tuned, e.g., using local statistics on the distances between pairs of points [17].

We assess clustering performance using the normalized mutual information (NMI) score between the cluster labels and the ground truth class labels. To reliably measure computational cost of all methods involved in the comparison, we count the amount of floating-point addition and multiplication operations they require, given the affinity matrix, and we also report running time statistics. We implemented all the algorithms in Python using the `numpy` and `scikit-learn` packages. All our experiments are conducted under Ubuntu Linux 14.04 with 10-core CPU and 20GB memory.

Table 1 summaries the statistics of the data sets, taken from LibSVM [18], that we consider in the experiments. To construct the `Covtype-I` set, we randomly sample 14129 samples from the first two classes of the original `Covtype` data set and merge the data samples of classes 4, 5 and 6 into one single class. The purpose is to avoid severe imbalances between classes. We make use of all data samples of the original `Covtype` data set to build the `Covtype-II` set.

We organize the experimental study in two parts. In both parts, we aim to show the performance of the proposed MBSC algorithm against other state-of-the-art algorithms for solving spectral clustering. In particular, we compare our proposal with the solution of the power iteration-based algorithm in Boutsidis et al. [5] and the Nyström approximation-based spectral clustering in Fowlkes et al. [6]. In the first part, this comparison is carried out on moderately large data sets comprising $10K$, $58K$ and $60K$ samples. For these, we can also report the performance of the spectral clustering approach in Shi and Malik [1] where the spectrum of the normalized Laplacian is computed exactly (denoted as "Exact" in the plots). In the second part, we repeat the same comparison on two larger sized data sets, comprising $100K$ and $580K$ samples, where we cannot compute the exact spectrum of L.

4.1 Comparative evaluation

We conduct a comparative evaluation of the proposed MBSC with state-of-the-art spectral clustering algorithms on the `Pendigits`, `Shuttle` and `MNIST` data sets [18]. In the proposed MBSC approach, we experiment with different choices of the mini-batch size to assess its impact on performance. To comprehensively analyze the performance of the proposed method, the iterative stochastic gradient descent on the manifold runs until we make one full pass through the whole data set. In practice, the stochastic gradient descent process can be stopped either if the difference between the spectral embedding W derived at two successive iteration steps is less than a threshold, or if a fixed number of iteration is achieved. For the Nyström approximation, we report a few choices on the number of samples selected to construct the approximate eigenvectors, namely 10, 50, 100, 500 and 1000.

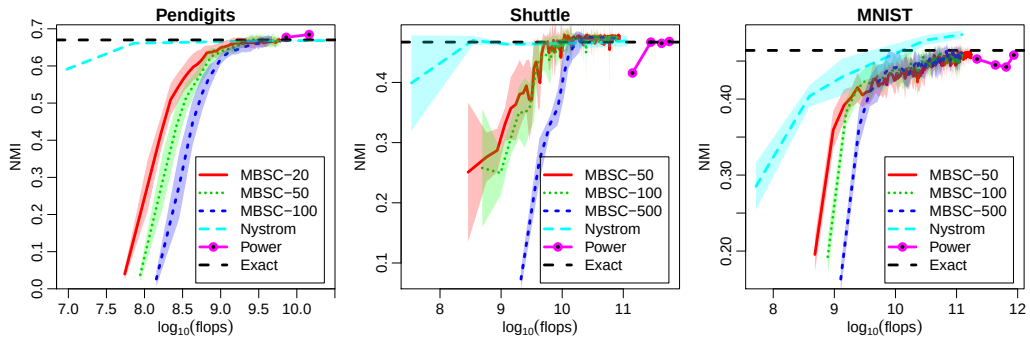

Figure 2: NMI versus counts of floating-point operations on the `Pendigits`, `Shuttle` and `MNIST` data sets. NMI for the exact spectral clustering is shown as a constant dashed line. MBSC with the largest mini-batches runs for $3.8s$, $107s$ and $112s$ to achieve an average NMI score of 0.67, 0.48 and 0.47 on the three data sets, whereas the power method takes $20s$, $935s$ and $987s$ to get a stable clustering output. The Nyström approximation needs 1000 samples to obtain the optimal clustering accuracy, which takes $15.2s$, $198s$ and $211s$ on the three data sets.

Figure 2 illustrates the NMI scores of the proposed MBSC algorithm, the power method, and the Nyström approximation versus the amount of floating-point operations. In the figure, for the proposed MSBC and the Nyström approximation we report the average plus and minus one standard deviation of the NMI score over 10 repetitions.

Recalling that m is the mini-batch size, k is the number of top eigenvectors and the number of clusters, each iteration of MBSC requires $2nkm + 6nk^2 - k^2$ floating point operations. The power method starts by generating an n-by-k Gaussian random matrix S, which costs $\mathcal{O}(nk)$ operations. It then computes a matrix product between the n-by-n affinity matrix and S, and iteratively applies the same multiplication for a total cost of $4n^2k - 2nk$ floating point operations. The final step of the power method performs Singular Value Decomposition (SVD) on an n-by-k matrix. Since $n \gg k$, we adopt the estimate of SVD complexity in [19], which costs $2nk^2 + 2k^3$ operations. Finally, in order to calculate the number of floating point operations for the Nyström approximation we follow the pseudo code in [6], for which the total count of floating point operations is $6nk + 8k^3 - 3k^2 + 4nk^2 - 3k + 2nkm + nm + 2nm^2 + m^2 + m^3 - n$.

In Figure 2, we can observe how the variance of the clustering performance of MSBC diminishes throughout iterations. Larger mini-batches lead to faster variance reduction of the stochastic gradient, thus producing faster convergence to the solution. Compared with the power method, the proposed MBSC needs distinctively less computations, while achieving higher or similar clustering accuracy. Another interesting observation is that the proposed MBSC achieves stable clustering accuracy before it makes a full pass through the whole data set. The Nyström approximation is computationally fast on all three data sets. Nevertheless, its time and space complexity increase drastically if the number of inducing points increases. On the `Pendigits` and `Shuttle` data sets, with less running time, the proposed MBSC method requires smaller mini batches to conduct clustering and achieves better clustering performance than the Nyström approximation. On the `MNIST` data set, instead, the Nyström approximation produces strikingly good clustering results when the number of selected samples is larger than 500. However, the proposed MBSC method converges to the clustering accuracy of the exact spectral clustering even when the size of the mini-batch size is small, e.g., less than 100.

4.2 Use case: spectral clustering on larger data sets

To demostrate that the proposed approach can tackle large-scale spectral clustering problems, we implemented the proposed MBSC algorithm without storing the Laplacian matrix, and applied it on two data sets comprising $100K$ and $580K$ samples. This variant of the code, that we denote as MBSC-E, computes the necessary columns of L to construct stochastic gradients on-the-fly.

Table 2 reports the overall running time and NMI of MBSC-E on the two data sets, where MBSC-E produces stable clustering results after 200 iterations. In addition, we compare against the Nyström approximation and the power method. In the comparison, the Nyström approximation selects a subset

Table 2: Running time comparison on Covtype-I and Covtype-II data. T is the number of iterations.

Covtype-I				Covtype-II			
Algorithm	T	NMI	time (s)	Algorithm	T	NMI	time (s)
MBSC-E-400	200	0.40	998	MBSC-E-1000	200	0.14	7500
MBSC-E-800	200	0.40	2100	MBSC-E-2000	200	0.14	12610
Nystrom-400	-	0.38	78	Nystrom-1000	-	0.09	2520
Nystrom-800	-	0.38	965	Nystrom-2000	-	0.11	11420
Power method	3	0.40	9300	Power Method	Too expensive		

of the same size of the mini-batch in MBSC-E. On the Covtype-II data set, the power method fails to obtain clustering results within an acceptable time, and we omit it from Table 2.

While running time is heavily dependent on implementation and system architecture, we argue that this is probably in favor of the Nyström approximation, for which we are using well optimized scientific computing packages. In any case, the purpose of this experiment is to demonstrate that the proposed MBSC algorithm, being exact in the limit of iterations, can achieve higher performance than approximate methods on large scale spectral clustering problems. Crucially, we demonstrate that this is possible at a comparable computational cost with approximate methods.

On both Covtype-I and Covtype-II data sets, MBSC achieves consistently better clustering accuracy than the Nyström approximation. Because the computational cost of the Nyström approximation rapidly increases with the size of the approximating set, it requires longer than MBSC-E to achieve a comparable clustering accuracy. Furthermore, compared with the power method, MBSC-E shows superior computational efficiency for large-scale spectral clustering problems. Remarkably, MBSC-E requires less than $1\mathrm{GB}$ and $3.6\mathrm{GB}$ memory to run on the two data sets, respectively.

5 Conclusions

With the aim of improving scalability of spectral clustering, in this work we formulated normalized cut spectral clustering as an optimization problem with an orthonormality constraint that we could solve using stochastic gradient optimization. We proposed a novel adaptive stochastic gradient optimization framework on Stiefel manifolds to compute the spectrum of Laplacian matrices, with computation of stochastic gradients linear in the number of samples. We provided theoretical justifications and empirical analyses to demonstrate how our proposal can tackle large-scale spectral clustering problems in a practical way.

The proposed stochastic optimization is characterized by attractive robustness to parameter selection and scalability properties, leading to the same clustering accuracy of spectral clustering approaches that use the exact spectrum of the Laplacian at a fraction of the cost. The results also support the motivation behind our proposal that approximate methods can potentially affect clustering performance. In cases where approximate methods perform well, as we reported in one of the experiments, we can see our proposal as a practical way to obtain the gold-standard of the "exact" approach at a reasonable cost. Furthermore, the proposed method does not need to load the whole Laplacian matrix into memory, making it especially suitable for handling large-scale spectral clustering with a limited memory footprint.

There are a number of extensions that we are currently investigating, such as the possibility to combine our framework with other approximate methods, for example to be able to afford more inducing points when performing spectral clustering using the Nyström approximation. Another extension is to leverage approximations to reduce the variance of the stochastic gradients without introducing any bias to accelerate stochastic gradient optimization. A Spark/TensorFlow implementation of the proposed algorithm is under development.

Acknowledgments

The Authors would like to thank Yun Shen from Symantec Research Labs for his support in setting up the experiments and optimizing the implementation of the algorithms.

References

[1] J. Shi and J. Malik. Normalized Cuts and Image Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905, 2000.

[2] A. Y. Ng, M. I. Jordan, and Y. Weiss. On Spectral Clustering: Analysis and an algorithm. In *Advances in Neural Information Processing Systems 14*, Cambridge, MA, 2002.

[3] I. S. Dhillon, Y. Guan, and B. Kulis. Kernel k-means: spectral clustering and normalized cuts. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 551–556, New York, NY, USA, 2004.

[4] U. von Luxburg. A tutorial on spectral clustering. *Statistics and Computing*, 17(4):395–416, 2007.

[5] C. Boutsidis, P. Kambadur, and A. Gittens. Spectral Clustering via the Power Method — Provably. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 40–48, 2015.

[6] C. Fowlkes, S. Belongie, F. Chung, and J. Malik. Spectral Grouping Using the Nyström Method. *IEEE Transactions on Pattern Analysis and Machince Intelligence*, 26(2):214–225, 2004.

[7] D. Yan, L. Huang, and M. I. Jordan. Fast Approximate Spectral Clustering. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 907–916, New York, NY, USA, 2009.

[8] Y. Li, J. Huang, and W. Liu. Scalable Sequential Spectral Clustering. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*, pages 1809–1815, 2016.

[9] S. Bonnabel. Stochastic gradient descent on Riemannian manifolds. *IEEE Transactions on Automatic Control*, 58(9):2217–2229, 2013.

[10] J. Duchi, E. Hazan, and Y. Singer. Adaptive Subgradient Methods for Online Learning and Stochastic Optimization. *Journal of Machine Learning Research*, 12:2121–2159, 2011.

[11] B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, Cambridge, MA, USA, 2001.

[12] F. R. K. Chung. *Spectral Graph Theory (CBMS Regional Conference Series in Mathematics, No. 92)*. American Mathematical Society, 1997.

[13] S. T. Smith. Optimization Techniques on Riemannian Manifolds, 2014. arXiv:1407.5965.

[14] Z. Wen and W. Yin. A feasible method for optimization with orthogonality constraints. *Mathematical Programming*, 142(1):397–434, 2013.

[15] J. Chen, M. Anitescu, and Y. Saad. Computing f(A)b via Least Squares Polynomial Approximations. *SIAM Journal on Scientific Computing*, 33(1):195–222, 2011.

[16] M Filippone and R. Engler. Enabling scalable stochastic gradient-based inference for Gaussian processes by employing the Unbiased LInear System SolvEr (ULISSE). In *Proceedings of the 32nd International Conference on Machine Learning*, pages 1015–1024, 2015.

[17] L. Zelnik-Manor and P. Perona. Self-tuning spectral clustering. In *Advances in Neural Information Processing Systems 17*, pages 1601–1608, 2004.

[18] C.-C. Chang and C.-J. Lin. LIBSVM: A library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, 2:27:1–27:27, 2011.

[19] G. H. Golub and C. F. Van Loan. *Matrix computations*. The Johns Hopkins University Press, 3rd edition, 1996.

A Derivation of the variance of stochastic gradients

Recall that the covariance of the stochastic gradient is:

$$\mathrm{cov}\left(L\mathbf{r}\mathbf{r}^\top\mathbf{w}\right) = \mathrm{E}[(L\mathbf{r}\mathbf{r}^\top\mathbf{w})(L\mathbf{r}\mathbf{r}^\top\mathbf{w})^\top] - \left(\mathrm{E}[L\mathbf{r}\mathbf{r}^\top\mathbf{w}]\right)\left(\mathrm{E}[L\mathbf{r}\mathbf{r}^\top\mathbf{w}]\right)^\top$$

By expanding the first term and realizing that the second term in the right hand side is just the outer product of the sth column of $\tilde{G}$ with itself, we obtain that the variance of the components of the stochastic gradient is:

$$\mathrm{diag}\left[\mathrm{cov}\left(L\mathbf{r}\mathbf{r}^\top\mathbf{w}\right)\right] = \mathrm{diag}\left[L\mathrm{E}\left[\mathbf{r}\mathbf{r}^\top\mathbf{w}\mathbf{w}^\top\mathbf{r}\mathbf{r}^\top\right]L^\top\right] - \mathrm{diag}\left[L\mathbf{w}\mathbf{w}^\top L^\top\right]$$

The focus of this supplement is to derive an expression for the expectation

$$\mathrm{E}\left[\mathbf{r}\mathbf{r}^\top\mathbf{w}\mathbf{w}^\top\mathbf{r}\mathbf{r}^\top\right]$$

needed to calculate the variance of the stochastic gradients. The ij element of the expectation is

$$\mathrm{E}(\mathbf{r}\mathbf{r}^\top\mathbf{w}\mathbf{w}^\top\mathbf{r}\mathbf{r}^\top)_{ij} = \sum_{kl}\sum_{r_i,r_k,r_l,r_j} r_i r_r w_k w_l r_s r_j P(r_i)P(r_k)P(r_l)P(r_j) \tag{10}$$

$$= \sum_{kl} w_k w_l \sum_{r_i,r_k,r_l,r_j} r_i r_k r_l r_j P(r_i)P(r_k)P(r_l)P(r_j) \tag{11}$$

We need to consider all possible n^4 cases for the indices going from 1 to n.

- $i \neq j \neq k \neq l$ - there are $\frac{n!}{(n-4)!}$ of these cases - all variables are independent and the expectation factorizes into the product of the expectations, which is zero.
- two pairs are equal - there are $3\frac{n!}{(n-2)!}$ of these cases. It is useful to realize the following expectation:

  $$\sum_{r_i,r_k,r_l,r_j} r_i r_k r_l r_j P(r_i)P(r_k)P(r_l)P(r_j) = 4\left(\frac{1}{\sqrt{p}}\right)^2\left(\frac{1}{\sqrt{p}}\right)^2\frac{p}{2}\frac{p}{2} = 1$$

  Within this set of cases we distinguish three cases, and there are $\frac{n!}{(n-2)!}$ for each of these:
  - $(i = j) \neq (k = l)$ - these are useful for the calculation of the diagonal of the expectation and they give $\sum_{k\neq i} w_k w_k$
  - $(i = k) \neq (j = l)$ - these give $w_i w_j$
  - $(i = l) \neq (k = j)$ - these give $w_i w_j$

  So for the off-diagonal elements of the expectation we have the sum of the last two cases above, giving $2w_i w_j$.
- two out of four are equal and the other two are different - there are $\frac{n!}{(n-3)!}\frac{4!}{(4-2)!2!}$ of these cases and they give a zero.
- three out of four are equal - there are $4\frac{n!}{(n-2)!}$ of these combinations - these cases give a zero.
- $i = j = k = l$ - there are n of these combinations - when the indices are all the same, we have nonzero in the two cases of v being $-\frac{1}{\sqrt{p}}$ or $+\frac{1}{\sqrt{p}}$, giving

  $$2w_i w_i\left(\frac{1}{\sqrt{p}}\right)^4\frac{p}{2} = \frac{w_i w_i}{p}$$

We are now ready to compute the expectation above. We distinguish two cases:

- $i \neq j$ - the off-diagonal of the expectation which is:

  $$2w_i w_j$$

- $i = j$ - the diagonal of the expectation which is:

$$\frac{w_i w_i}{p} + \sum_{k \neq i} w_{kk}$$

Because of these expressions, the expectation can be rewritten in matrix form as:

$$\mathrm{E}(\mathbf{r}\mathbf{r}^\top \mathbf{w}\mathbf{w}^\top \mathbf{r}\mathbf{r}^\top) = 2\mathbf{w}\mathbf{w}^\top - 2\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top)) + \frac{1}{p}\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top)) + \mathrm{Tr}(\mathbf{w}\mathbf{w}^\top)I - \mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top))$$

We realize that

$$\mathrm{Tr}(\mathbf{w}\mathbf{w}^\top) = \|\mathbf{w}\|^2 = 1$$

because of the orthonormality constraint, so after substituting this expression and gathering terms we obtain

$$\mathrm{E}(\mathbf{r}\mathbf{r}^\top \mathbf{w}\mathbf{w}^\top \mathbf{r}\mathbf{r}^\top) = 2\mathbf{w}\mathbf{w}^\top + \left(\frac{1}{p} - 3\right)\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top)) + I$$

Plugging this into the expression of the variance of the components of the stochastic gradient we obtain

$$\mathrm{diag}\left[\mathrm{cov}\left(L\mathbf{r}_i\mathbf{r}_i^\top \mathbf{w}\right)\right] = \mathrm{diag}\left[L\left(2\mathbf{w}\mathbf{w}^\top + \left(\frac{1}{p} - 3\right)\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top)) + I\right)L^\top\right] - \mathrm{diag}\left[L\mathbf{w}\mathbf{w}^\top L^\top\right]$$

This can be simplifed into

$$\mathrm{diag}\left[\mathrm{cov}\left(L\mathbf{r}_i\mathbf{r}_i^\top \mathbf{w}\right)\right] = \mathrm{diag}\left[L\left(\mathbf{w}\mathbf{w}^\top + \left(\frac{1}{p} - 3\right)\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top)) + I\right)L^\top\right]$$

and finally into

$$\mathrm{diag}\left[\mathrm{cov}\left(L\mathbf{r}_i\mathbf{r}_i^\top \mathbf{w}\right)\right] = \mathrm{diag}\left[L\mathbf{w}\mathbf{w}^\top L^\top + \left(\frac{1}{p} - 3\right)L\mathrm{diag}(\mathrm{diag}(\mathbf{w}\mathbf{w}^\top))L^\top + LL^\top\right]$$