# Distributed and Asynchronous Methods for Semi-supervised Learning

Konstantin Avrachenkov[1], Vivek S. Borkar[2], Krishnakant Saboo[2]

[1] Inria Sophia Antipolis, France
k.avrachenkov@inria.fr
[2] IIT Bombay, India
{borkar.vs,kishansaboo.2004}@gmail.com

**Abstract.** We propose two asynchronously distributed approaches for graph-based semi-supervised learning. The first approach is based on stochastic approximation, whereas the second approach is based on randomized Kaczmarz algorithm. In addition to the possibility of distributed implementation, both approaches can be naturally applied online to streaming data. We analyse both approaches theoretically and by experiments. It appears that there is no clear winner and we provide indications about cases of superiority for each approach.

## 1 Introduction

Semi-supervised learning (SSL) is a type of learning which uses both labelled and unlabelled data for training [8]. In many practical cases, the amount of labelled data is much less compared to the unlabelled data, making SSL a powerful tool for processing massive data. The present work focuses on graph based SSL [8,11,20,24,29,30]. Consider a (weighted) graph in which nodes belong to one of $K$ classes and the true class of a few nodes is given. An edge and its weight in the graph indicate a similarity and similarity degree between two nodes, respectively. Hence such a graph is called a similarity graph. SSL aims to estimate the class of each of the unlabelled nodes by using the true class information of few labelled nodes and the structure of the graph.

Being of practical importance [11,29], graph based SSL has been widely studied. Most though not all (see e.g., [13,21]) methods formulate SSL as a quadratic optimization problem [2,3,4,26,27,28] for feature vectors and then solve it or the associated stationary point equation iteratively. Iterative solutions become computationally intensive because they involve matrix operations which grow in size as a polynomial with the graph size, though the growth can be linear or quasi-linear in case of sparse graphs. The data can be distributed over a network (as in sensor networks [19] or in the internet of things) or can be stored and processed in distributed manner as is the case in data centers. These motivate us to look for distributed algorithms. There are a variety of algorithms that address this issue, see, e.g., the classic text of [5] or a more recent comparative survey of [14]. We apply variants of two of these for the solution of the stationary point equation for

the feature vectors and conduct a comparative analysis. The first, more original, is based on gossip-type stochastic approximation, whereas the second is based on the well known randomized Kaczmarz algorithm. The randomized Kaczmarz algorithm has drawn much attention in recent years, beginning with the seminal work of [23] - see [17,18,31] for some subsequent work ([31] is the closest in spirit to ours). Both approaches are asynchronously distributed and can also be applied online to streaming data (e.g., as in stream image classification tasks [25]). There is no clear winner with respect to accuracy and we provide indications about cases of superiority for each approach. With respect to computational efficiency, the stochastic approximation approach appears to be generally more efficient than the randomized Kaczmarz approach on all our numerical examples. In the case of online data processing, both approaches are simpler and computationally lighter than the method of [25] based on the Nyström approach which is not distributed. We would also like to mention that the recent well performing SSL methods [20] and [24] use the Jacobi method which can be straightforwardly distributed in synchronous manner. However, we would like to note that the convergence conditions for the asynchronous version of the Jacobi method can be more stringent than the convergence conditions for our approaches [5].

As shall be seen later, our only requirements are: global knowledge on the number of classes and the ability of nodes to pass information to their immediate neighbors. The adjacency matrix of the network is defined by the support of the similarity matrix. Clearly, our approach will be particularly efficient when the similarity matrix is sparse. Regarding mapping data points to agents, our basic scenario is: each node corresponds to a different agent and needs to compute the elements of the classification functions only corresponding to itself. For instance, this is a natural setting in wireless sensor networks. Our approaches can be extended in a straightforward way to the case when one agent is responsible for several nodes. The computation at each node happens in an asynchronous fashion and the information needs to be exchanged only among the neighbours. This highlights the distributed nature of the approaches.

Rest of the article is organised as follows: the next section introduces the problem formally. Section 3 introduces and analyses two distributed approaches for graph-based SSL. Section 4 shows that a number of established SSL methods can be mapped to our general methodology. Section 5 investigates and compares the two approaches by numerical experiments.

## 2 Definitions and problem formulation

Consider a graph with (weighted) adjacency matrix $A$ with $N$ nodes, each belonging to one of $K$ classes. (We discuss online formulation of the problem later.) The class information is given for some nodes, referred to as labelled nodes. Define $D$ as a diagonal matrix with $D_{ii} = d(i)$ for $d(i) :=$ the (weighted) degree of node $i$. We also use the standard graph Laplacian $L = D - A$. Define $N \times K$

matrix $Y$ containing information about the labelled nodes by:

$$Y_{ij} = \begin{cases} 1 & \text{if labelled node } i \text{ belongs to class } j, \\ 0 & \text{otherwise.} \end{cases}$$

$F = \{F_{ik}\}$ is a $N \times K$ matrix with $F_{ik}$ representing the 'belongingness' of node $i$ to class $k$. The vector $F_{*k}$ is referred to as 'classification function' or 'feature vector'. The aim of the SSL problem is to find $F_{*k}$ such that it is close to the labelling function and it varies smoothly over the graph. The class of node $i$ is calculated from $F$ by assigning class $k$ to node $i$ if $F_{ik} > F_{ij}, \quad \forall\ j \neq k$. The optimization problem associated with the above stated requirements is to minimize

$$Q(F) = \sum_{k=1}^{K} F_{*k}^T \mathcal{A} F_{*k} + \mu \sum_{k=1}^{K} (F_{*k} - Y_{*k})^T \mathcal{B} (F_{*k} - Y_{*k}), \tag{1}$$

where $\mathcal{A}$ is the positive (semi-)definite graph kernel and $\mathcal{B}$ the cost of deviation from the labels. Typically, the support of matrix $\mathcal{A}$ coincides with the support of the adjacency matrix $A$ and matrix $\mathcal{B}$ is diagonal. $\mu > 0$ is the regularization parameter. Majority of all existing graph-based semi-supervised learning methods can be cast into the optimization formulation (1). A few important examples are: Choosing $\mathcal{A} = D^{\sigma-1} L D^{\sigma-1}$ and $\mathcal{B} = D^{2\sigma-1}$ [2], we obtain semi-supervised methods based on standard Laplacian ($\sigma = 1$) [27], normalized Laplacian ($\sigma = 1/2$) [26], and PageRank ($\sigma = 0$) [1]. If $\mathcal{A} = L$ and $\mathcal{B} = I$, we retrieve the semi-supervised method based on the regularized Laplacian [3,9,22], which can be viewed as a Lagrangian relaxation of the method based on harmonic functions [28]. If $\mathcal{A} = \gamma D - A$ and $\mathcal{B} = I$, we obtain the method based on the modified regularized Laplacian [15]. A good comparative overview of the graph-based semi-supervised learning methods can be found in [11]. Nearly all methods described in [11] can be represented in optimization formulation (1). In [4,20,24] the authors discussed slightly different quadratic optimization formulations of the semi-supervised learning methods with an additional quadratic extra term. That additional term can be included in the first or second terms of (1).

Except for the harmonic functions method which puts zeros for diagonal elements of $\mathcal{B}$ when labels are unavailable, we used zero as default label when unavailable, a choice neutral to the classes. This gives a bias towards zero for learned classification functions, but because we are interested only in their ordinal comparison, the classification is robust to this choice.

The above problem is a convex quadratic optimization problem. Applying the first order optimality condition and solving for $F_{*k}$ gives the stationary point equation:

$$\left[(\mathcal{A} + \mathcal{A}^T) + \mu(\mathcal{B} + \mathcal{B}^T)\right] F_{*k} = \mu(\mathcal{B} + \mathcal{B}^T) Y_{*k}. \tag{2}$$

## 3 Distributed approaches

In this section we describe two approaches for solution of (2) which can be naturally distributed (in asynchronous fashion) and can also be applied in a scenario

with streaming data. We prove the convergence of the proposed approaches and comment on their rate of convergence.

### 3.1 Stochastic approximation approach

The first solution is based on Stochastic Approximation (SA-approach). Consider the general problem of finding a unique solution $x^*$ of the system

$$x = G(x) = \widetilde{B}x + \widetilde{Y}, \tag{3}$$

where $x, \widetilde{Y} \in \mathcal{R}^d$ and $\widetilde{B} = \{b(i,j)\} \in \mathcal{R}^{d\times d}$ is irreducible non-negative. In Section 4 we show that most semi-supervised learning methods can be written in this form for appropriate $\widetilde{B}$ and $\widetilde{Y}$; see, e.g., (7), (9). Define $H$ as a diagonal matrix with elements as row sums of $\widetilde{B}$, i.e., $H_{ii} = \sum_j \widetilde{B}_{ij}$ Define $P$ as $P = H^{-1}\widetilde{B}$, viewed as the transition probability matrix on the graph, and $Q$ is its irreducible counterpart as in the PageRank algorithm: $Q = (1-\epsilon)\, P + \epsilon/N\, E$, where $E$ is an $N \times N$ matrix with all 1's. Let $X_t, t \geq 0$, be a Markov chain with transition matrix $Q$ and $\{\eta_t\}_{t\geq 0}$ a positive step-size sequence satisfying $\sum_{t\geq 0} \eta_t = \infty$ and $\sum_{t\geq 0} \eta_t^2 < \infty$. The stochastic approximation scheme to solve (3) is:

$$x_i^{t+1} = x_i^t + \eta_t I\{X_t = i\} \frac{P(i, X_{t+1})}{Q(i, X_{t+1})} \left( H_{ii} x^t_{X_{t+1}} - x_i^t + \widetilde{Y}_i \right). \tag{4}$$

**Convergence analysis** In equation (3), if $\widetilde{B}$ has its Perron-Frobenius eigenvalue $\lambda \in (0,1)$ with the normalized positive eigenvector $w = [w_1, ...w_d]^T$, then the following hold. (Proofs will appear in the journal version of the paper.) Define the weighted norm $\|x\|_w = \max_i |x_j/w_i|$ .

**Lemma** Map $G$ is a contraction w.r.t. $\|x\|_w$, i.e., $\|G(x) - G(y)\|_w \leq \lambda \|x - y\|_w$.

Using the above lemma, we can establish the following.

**Theorem** Almost surely, $x^t \to x^*$ as $t \to \infty$.

**Convergence rate** Section 4.2 of [6] gives results on sample complexity of a synchronous stochastic approximation. The result broadly implies that, at time $n$, the probability of remaining within a prescribed small neighborhood of $x^*$ after $n+\tau$ iterates is greater than $1 - O\left(e^{-\frac{C}{\sum_{t\geq n}\eta_t^2}}\right)$, $C > 0$. For our case where, $\eta_t = \Theta\left(\frac{1}{t}\right)$, the decay of probability of ever escaping from this neighborhood after $n+\tau$ iterations is exponential. Here $\tau$ is a quantity specified in terms of problem parameters.

We used Markov chain sampling of nodes above. For more general asynchronous schemes, the following considerations apply. We can use the step size $\eta_{\nu(t,i)}$ for the $i^{th}$ node where $\nu(t,i)$ = the number of updates node $i$ has made till time $t$ i.e., its local clock. Also, $\eta_t$ has to satisfy additional conditions as in

Chapter 7 of [6]. We then get a time scaled version of the same limiting O.D.E. in the asynchronous case as the synchronous case and the asymptotic behavior of the algorithm is the same as that for the synchronous case.

### 3.2 Randomized Kaczmarz approach

Another asynchronously distributed approach is to apply the randomized Kaczmarz algorithm (RK-approach) to (2). We solve the linear system (2) of the form $\underline{A}x = \underline{b}$, where $\underline{A} = (\mathcal{A}+\mathcal{A}^T)+\mu(\mathcal{B}+\mathcal{B}^T)$ and $\underline{b} = \mu(\mathcal{B}+\mathcal{B}^T)Y$. Let $a_i :=$ the $i^{th}$ row of $\underline{A}$, $\hat{a}_i :=$ the unit normalized row $a_i$ and $p_i :=$ the probability of sampling it for the Kaczmarz update (e.g., $p_i = \frac{1}{N}$ for uniform sampling). The update rule at step $t$ is given by

$$x(t+1) = x(t) + \sum_i I\{\xi(t) = i\}\frac{b(i) - \langle a_i, x(t)\rangle}{\|a_i\|_2^2}a_i^T, \tag{5}$$

where $Prob(\xi(t) = i) = p_i \ \forall \ i \in \{1, 2, ..., N\}$. Let $\lambda_{min}$ be the smallest non-negative eigenvalue of the matrix $\sum_i p_i \hat{a}_i^T \hat{a}_i$. From [7], we have that for $\lambda_{min} \in (0, 1)$, $x(t) - x^* \to 0$ almost surely and $E[\|x(t) - x^*\|^2] \to 0$ exponentially with rate $(1 - \lambda_{min})$ where $x^*$ is such that $\underline{A}x^* = \underline{b}$.

The randomized Kaczmarz scheme is an inherently distributed and asynchronous scheme because only one component (corresponding to a single node) is being updated at a time, using local information from the node's neighbours. This sampling is done probabilistically in an i.i.d. fashion with probability vector $p = [p_1, ..., p_N]^T$. We may drop the assumption of identical distribution sampling and still retain exponential decay of mean square error as long as $\lambda_{min}$ remains bounded away from 1 from above for time-varying $p$. In fact, one may drop the independence assumption as well, replacing it with the above condition on the conditional sampling probabilities given the history. The exponential decay rate of mean square error then is $1 - \lambda^*$, where $\lambda^* < 1$ is the aforementioned upper bound. For normal matrices, the condition number of a matrix $M$ is given by $\kappa(M) = \left|\frac{\lambda_{max}(M)}{\lambda_{min}(M)}\right|$, where $\lambda_{min}(M)$ and $\lambda_{max}(M)$ are its minimum and maximum eigenvalues, respectively. Hence a high condition number would imply a very small $\lambda_{min}$ resulting in very slow convergence.

### 3.3 Comments on implementation

For the asynchronous implementation of (4) and (5) we have a Poisson local clock at each node. The node updates its classification function once the local clock ticks. We simulate this by performing a coin toss at each instant of the global clock. The coin has a low probability $p_H$ of turning up heads. The node updates its classification function if it gets head at that time instant. There can be multiple nodes updating at a given global instant while there can also be no nodes updating at some other instant. This leads to the nodes updating their classification function asynchronously.

We emphasize that both our approaches, SA and RK, require the information only from the neighbors of the node. In fact, for any update, SA needs the information only from one random neighbor. Updating each node independent of the other nodes with only local information implies the distributed nature of the updates. In theory, we can have different $p_H$ for each node and convergence is assured if $p_H$ for all nodes is bounded away from zero. What we find particularly interesting is that the unlabelled nodes do not need to know the location of the labelled nodes. The information from the labelled nodes propagates through the network from neighbor to neighbor.

## 4 Application to specific SSL methods

We discuss the application of the two approaches to several well known SSL methods.

### 4.1 Normalized Laplacian-type methods

For $\mathcal{A} = D^{\sigma-1}LD^{\sigma-1}$ and $\mathcal{B} = D^{2\sigma-1}$ in (2), we obtain the normalized Laplacian-type methods [1,2,26,27] which yields the following equation on simplification:

$$\left(I - \frac{1}{1+\mu}D^{-\sigma}AD^{\sigma-1}\right)F_{*k} = \frac{\mu}{1+\mu}Y_{*k}. \tag{6}$$

This equation can be solved using RK-approach or can be recast to resemble (3) as follows:

$$F_{*k} = \frac{1}{1+\mu}D^{-\sigma}AD^{\sigma-1}F_{*k} + \frac{\mu}{1+\mu}Y_{*k}. \tag{7}$$

In case of normalized Laplacian-type methods, $\widetilde{B} = \frac{1}{1+\mu}D^{-\sigma}AD^{\sigma-1}$ is similar to and a scaled version of the transition probability matrix $D^{-1}A$. Hence its top eigenvalue is less than 1. Thus a stochastic approximation of the form of (4) written for (7) will converge to its solution.

### 4.2 Regularized Laplacian method

$\mathcal{A} = L, \mathcal{B} = I$ in (2) yield the regularized Laplacian method [3,9,22] which on simplification gives:

$$(L + \mu I)F_{*k} = \mu Y_{*k}. \tag{8}$$

This can be solved using RK-approach or can be rewritten in a form similar to equation (3) which leads to the stochastic approximation scheme:

$$F_{*k} = (D + \mu I)^{-1}AF_{*k} + \mu(D + \mu I)^{-1}Y_{*k}. \tag{9}$$

In case of regularized Laplacian, $\widetilde{B} = (D+\mu I)^{-1}A$. Its row sums are $< 1$, making $\widetilde{B}$ sub-stochastic with its top eigenvalue $< 1$. Hence a stochastic approximation for (9) of the form (4) will converge to its solution.

### 4.3 Harmonic functions method

Finally, substituting in (2) $\mathcal{A} = L$ and $\mathcal{B} = diag(I, 0)$, where the non-zero elements correspond to the labelled points, we get the harmonic functions method [28] which on simplification gives:

$$(L + \mu\, diag(I, 0))F_{*k} = \mu Y_{*k}. \tag{10}$$

A direct application of SA-approach may not be possible for this, but application of the RK-approach is straightforward.

## 5 Experiments

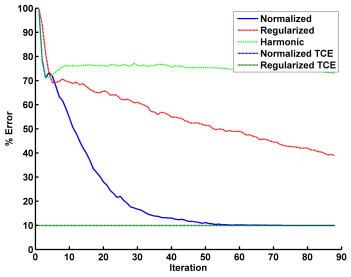
We test our distributed approaches on the three methods (normalized Laplacian with $\sigma = 1/2$ [26], regularized Laplacian [3,9,22] and harmonic functions [28]) applied to synthetic as well as real world graphs. As performance metric, we use classification error, i.e., the percentage of nodes wrongly classified. For all the experiments we took $\mu = 0.5$ as the regularization parameter (the results are robust for reasonable values of $\mu$). For comparison purposes, we take 1 iteration = $N$ steps, where $N$ is the number of nodes in the graph and one step is defined as one application of formula (5) for RK-approach and formula (4) for SA-approach, respectively. In most experiments, a uniform distribution is used for sampling rows in RK-approach. A decreasing step size for node $i$ of $\frac{1}{2+\nu(t,i)}$ where $\nu(t, i)$ is the local clock at node $i$ is used in (4).

### 5.1 WebKB graph

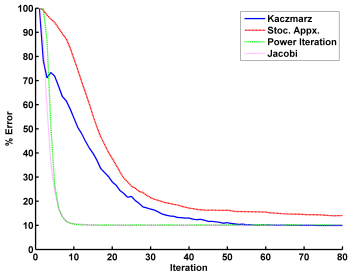
We look at the classification of webpages of 4 universities - Cornell, Texas, Washington and Wisconsin - corresponding to the popular WebKB dataset [10]. The graph formed by the hyperlinks connecting these pages is taken such that only webpages with hyperlinks to webpages within the dataset are considered. Self-links are removed. Clusters thus formed are: Cornell (676), Texas (590), Washington (982) and Wisconsin (613). The highest degree node from each class (university main web page) is labelled. Figure 1a shows the error evolution for the Kaczmarz implementation of the three SSL methods. Convergence only occurs for normalized Laplacian for the number of iterations shown while both other methods have a negative slope. Convergence occurs for regularized Laplacian method after 200 iterations. All the methods seem to have the same initial rate of convergence however they change drastically after around 3 iterations. Theoretical Classification Error (TCE) for all the methods is almost the same. Figure 1b shows the comparison of RK-approach, SA-approach, distributed Jacobi and power iteration for normalized Laplacian. Power iteration is simply the repeated application of (7) with one iteration defined as multiplication with $\frac{1}{1+\mu} D^{-1/2} A D^{-1/2}$ once. RK-approach's performance is better than that of SA-approach for this graph in terms of error while convergence occurs faster for the latter. Convergence is fastest for power iterations (PI) implementation since

in each iteration, the classification function of all the nodes is being updated and as a result, the update performed in the consequent iteration would be with the updated classification functions of the neighbors. This is unlike SA and RK approach where the update might use the non-updated classification function of its neighbor.

Let a call be defined as the transfer of information to a node from its neighbor. This information is used for updating the classification function. As Figure 1b shows, the number of iterations is almost the same for RK and SA. While updating the classification function of a node in one step of SA, only one of its neighbors is called, whereas all neighbors are called in RK and Jacobi. As a result, RK and Jacobi have more exchange of information and is computationally more expensive than SA.




(a) Comparison of Kaczmarz implementation of all the three methods. In dotted is shown the theoretical classification error (TCE).



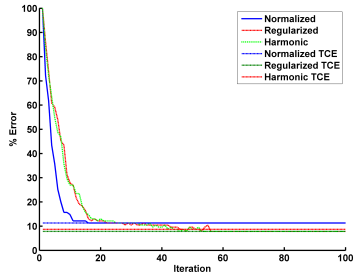

(b) Comparison of Kaczmarz, power iteration, Jacobi and stochastic approximation implementation of normalized Laplacian.

Fig. 1: Performance on the WebKB graph

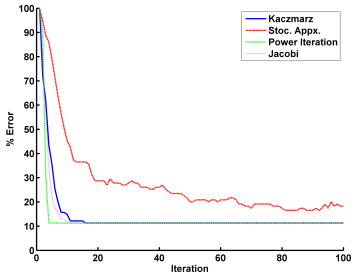
### 5.2 US football graph

We next see the classification in US college football network [12]. Nodes in the graph represent colleges that participated in the Division 1 games for the 2000 season. Edges between nodes represent games between the two teams they connect. The classes and the nodes corresponding to each class are known for this graph. There are 12 classes with each class consisting of 8-12 teams. Teams within the same class, called 'conference', tend to play more games with each other than with teams from another conference. The highest degree node from each class (conference) is labelled. Figure 2a shows the error evolution for the RK-approach implementation of the three methods. The convergence is fastest for the normalized Laplacian while the error is less for regularized Laplacian and harmonic functions methods. Figure 2b shows the comparison of RK-approach, SA-approach, distributed Jacobi and power iteration for normalized Laplacian.

In this example, RK and Jacobi show similar behaviour. We recall again that RK and Jacobi use more information and computationally more expensive than SA.




(a) Comparison of Kaczmarz implementation of all the three methods. In dotted is shown the theoretical classification error (TCE).




(b) Comparison of Kaczmarz, power iteration, Jacobi and stochastic approximation implementation of normalized Laplacian.

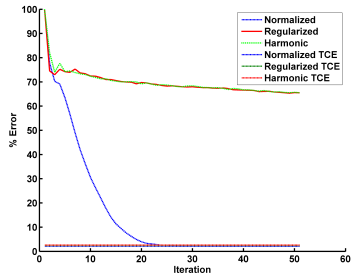
Fig. 2: Performance on the US Football graph

### 5.3 Gaussian mixture model graph

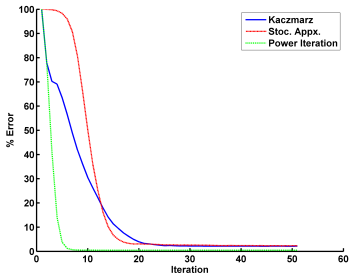
A Gaussian mixture model graph of 10000 nodes with 3 classes was created with the probability of a node belonging to either of the three classes being equal. Each class was generated on a Gaussian kernel. Nodes within a given radius of each other shared edges. Two nodes with the highest degree from each class were labelled. Figure 3a compares the error evolution of RK-approach for the three methods. Convergence only occurs for normalized Laplacian method for the number of iterations shown, while the other methods have a negative slope. All the methods seem to have the same initial rate of convergence which changes drastically after a few iterations. Figure 3b compares the RK-approach, SA-approach and power iteration for normalized Laplacian. The performance of PI is best in terms of both error and rate of convergence. We would like to emphasize that we sacrifice the rate of convergence for the distributed nature of algorithms. However, the performance for RK and SA approaches for this graph is different as compared to WebKB graph, Figure 1b. RK-approach has a higher error as well as convergence time as compared to SA-approach.

### 5.4 Online learning

In the RK-approach as well as SA-approach, the classification function is updated only for one or few nodes in one step. In other words, only local information is used each time while updating, allowing for natural application of our approaches to dynamic setting with streaming data. To illustrate the performance of our




(a) Comparison of Kaczmarz implementation of all the three methods. In dotted is shown the theoretical classification error (TCE).



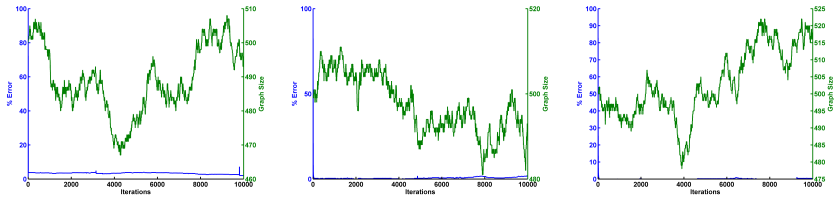

(b) Comparison of Kaczmarz, power iteration and stochastic approximation implementation of normalized Laplacian.

Fig. 3: Performance on Gaussian mixture model graph of 10000 nodes.

approaches in the dynamic setting, we consider a dynamic stochastic block model graph in which nodes enter and leave the graph. Upon arrival, a node is connected with another node of the same class with probability $p_{in} = 0.15$ and node from a different class with probability $p_{out} = 0.01$; $p_{in} >> p_{out}$. Nodes arrive into the graph according to the Poisson process with rate $\lambda_{arr}$. The class of the arriving node is chosen according to a pre-specified probability distribution. Each node stays in the graph for a random time that is exponentially distributed with mean $1/\mu_{dep}$, after which it leaves. The maximum number of nodes in the graph is limited to $K$, i.e., nodes arriving when the number of nodes in the graph is $K$ do not become a part of the graph. This system can be modelled as an M/M/K/K queue. As a result, irrespective of the number of nodes that the graph has initially, the average of the number of nodes in the graph will reach a steady state value given approximately by $\frac{\lambda_{arr}}{\mu_{dep}}$. In the considered example, the graph has 3 classes and an incoming node could belong to either of the classes with equal probability. We choose $K = 1000$, $\lambda_{arr} = 1/(2 \times 10^4)$ and $\mu_{dep} = 1/10^7$. Therefore, $\frac{\lambda_{arr}}{\mu_{dep}} = 500$. Two nodes with the maximum degree from each class were chosen as the labelled nodes during initialization. In case a labelled node left, then a random neighbor was labelled. In the plots, $K$ steps are considered as one iteration.

Figures 4a, 4b and 4c show the error evolution of the RK-approach implementation of normalized Laplacian, regularized Laplacian and harmonic functions methods, respectively, for the dynamic stochastic block model graph. The variation of the graph size is also shown in the same figures. In terms of accuracy, the performances of regularized Laplacian and harmonic function methods are similar, being between 0-1.5% during steady state. Interestingly, normalized Laplacian method has a higher error compared to the other two, being close to 3.5%. Figure 5 shows a zoom of the error evolution from Figures 4a, 4b and 4c. From this figure, it can be seen that the convergence is faster for the normalized

Laplacian compared to the other two methods, both of which have almost the same convergence time.




(a) Normalized Laplacian. (b) Regularized Laplacian. (c) Harmonic functions.

Fig. 4: Error evolution and graph size variation for a dynamic stochastic block model graph for Kaczmarz implementation of various methods.

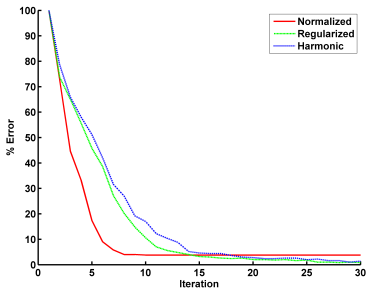



Fig. 5: This plot shows the comparison of Kaczmarz implementation of different methods for a dynamic stochastic block model.

### 5.5 Faster convergence for normalized Laplacian

The convergence was faster for the normalized Laplacian method compared to the regularized Laplacian and harmonic functions methods in all the cases of the RK-approach application. This can be understood from the condition number values for each method on different graphs as shown in Table 1 and invoking theoretical observations from Section 3.2. The condition number is the smallest for the normalized Laplacian method and the highest for the harmonic functions method in all the cases. $\lambda_{max}$ being less than 1 for all the methods, the $\lambda_{min}$ must be very small for the harmonic functions method as compared to normalized Laplacian to explain the large condition number. This leads to their large convergence times.

Table 1: Condition number values.

| | Normalized | Regularized | Harmonic |
|---|---|---|---|
| US Football | 3.88 | 32.32 | 314.55 |
| WebKB | 5 | 300 | $5.6 \times 10^{18}$ |
| Gaussian | 4.46 | $2.8 \times 10^3$ | $4.6 \times 10^6$ |

## 6 Conclusion

We proposed two asynchronously distributed approaches for graph-based semi-supervised learning. The first approach is based on stochastic approximation, whereas the second is based on the randomized Kaczmarz algorithm. We demonstrated that both the approaches can be naturally applied online to streaming data. Both were analysed theoretically and by experiments. Our main conclusions: there is no clear winner in terms of accuracy but the SA-approach generally outperformed the RK-approach in terms of operations count. In terms of accuracy, RK-approach performed better on real world datasets (US football and WebKB) while SA-approach performed better on the synthetic Gaussian mixture model. When using RK-approach, normalized Laplacian method showed much faster convergence as compared to regularized Laplacian and harmonic functions owing to its low condition number.

## Acknowledgments

This work was partially supported by Indo-French CEFIPRA Collaboration Grant No.5100-IT1 "Monte Carlo and Learning Schemes for Network Analytics" and Inria Bell Labs Joint Initiative.

## References

1. Avrachenkov, K., Dobrynin, V., Nemirovsky, D., Pham, S.K. and Smirnova, E. PageRank based clustering of hypertext document collections. In *Proceedings of ACM SIGIR* 2008.
2. Avrachenkov, K., Gonçalves, P., Mishenin, A., and Sokol, M. Generalized optimization framework for graph-based semi-supervised learning. In *Proceedings of SDM* 2012.
3. Avrachenkov, K., Chebotarev, P. and Mishenin, A. Semi-supervised learning with regularized laplacian. Accepted in *Optimization Methods & Software*, 2016.
4. Bengio, Y., Delalleau, O., and Le Roux, N. Label propagation and quadratic criterion. Ch.10 in *Semi-supervised learning*, 2006.
5. Bertsekas, D.P. and Tsitsiklis, J.N. *Parallel and distributed computation: numerical methods*. Englewood Cliffs, NJ: Prentice hall, 1989.
6. Borkar, V.S. *Stochastic approximation: A Dynamical Systems Viewpoint*. Hindustan Publishing Agency, New Delhi, and Cambridge Uni. Press, Cambridge, UK, 2008.

7. Borkar, V.S., Karamchandani, N., and Mirani, S. Randomized Kaczmarz for Rank Aggregation from Pairwise Comparisons. *IEEE ITW* 2016.
8. Chapelle, O., Schölkopf, B. and Zien A. *Semi-supervised learning*, MIT Press, 2006.
9. Chebotarev, P. and Shamis, E. The matrix-forest theorem and measuring relations in small social groups. *Automation and Remote Control*. v.58(9), pp.1505-1514, 1997.
10. Craven, M., McCallum, A., PiPasquo, D., Mitchell, T., and Freitag, D. (1998). *Learning to extract symbolic knowledge from the World Wide Web* (No. CMU-CS-98-122). Carnegie-Mellon Univ, Pittsburgh, PA, School of computer Science.
11. Fouss, F., Francoisse, K., Yen, L., Pirotte A., and Saerens, M. An experimental investigation of kernels on graphs for collaborative recommendation and semisupervised classification. *Neural Networks*, 31, pp.53-72, 2012.
12. Girvan, M. and Newman, M.E.J. Community structure in social and biological networks. *PNAS*. USA 99, 7821-7826, 2002.
13. Gleich, D.F. and Mahoney, M.W. Using local spectral methods to robustify graph-based learning algorithms. In *Proceedings of ACM SIGKDD* 2015.
14. Gower, R.M. and Richtárik, P. Randomized iterative methods for linear systems. *SIAM Journal on Matr. Anal. and Appl.*, v.36(4), pp.1660-1690, 2015.
15. Ito, T., Shimbo, M., Kudo, T. and Matsumoto, Y. Application of kernels to link analysis. In *Proceedings of ACM SIGKDD* 2005.
16. Knuth, D.E. *The Stanford GraphBase: A Platform for Combinatorial Computing*, Reading: Addison-Wesley, 1993.
17. Liu, J., Wright, S.J. and Sridhar, S. An asynchronous parallel randomized Kaczmarz algorithm. ArXiv preprint 1401.4780, 2014.
18. Needell, D., Ward, R. and Srebro, N. Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm. In *Proceedings of NIPS* 2014.
19. Pan, J.J., Pan, S.J., Yin, J., Ni, L.M. and Yang, Q. Tracking mobile users in wireless networks via semi-supervised colocalization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v.34(3), pp.587-600, 2012.
20. Ravi, S. and Diao, Q. Large Scale Distributed Semi-Supervised Learning Using Streaming Approximation. In *Proceedings of AISTATS* 2016.
21. Shivanna, R., Chatterjee, B.K., Sankaran, R., Bhattacharyya, C. and Bach, F. Spectral norm regularization of orthonormal representations for graph transduction. *Advances in Neural Information Processing Systems*, pp.2215-2223, 2015.
22. Smola, A.J. and Kondor, R. Kernels and regularization on graphs. In *Proceedings of COLT* 2003.
23. Strohmer, T. and Vershynin, R. A randomized Kaczmarz algorithm with exponential convergence. *Journal of Fourier Analysis and Applications*, v.15(2), pp.262-278, 2009.
24. Talukdar, P.P. and Crammer, K. New regularized algorithms for transductive learning. In *Proceedings of ECML PKDD* 2009.
25. Valko, M., Kveton, B., Huang, L. and Ting, D. Online semi-supervised learning on quantized graphs. In *Proceedings of UAI* 2010.
26. Zhou, D., Bousquet, O., Lal, T.N., Weston, J. and Schölkopf, B. Learning with local and global consistency. *Advances in Neural Information Processing Systems*, v.16, pp.321-328, 2004.
27. Zhou, D. and Burges, C.J. Spectral clustering and transductive learning with multiple views. In *Proceedings of ICML* 2007.
28. Zhu, X., Ghahramani, Z. and Lafferty, J. Semi-supervised learning using Gaussian fields and harmonic functions. In *Proceedings of ICML* 2003.

29. Zhu, X. Semi-supervised learning: literature survey. University of Wisconsin-Madison Research Report TR 1530, 2005.
30. Zhu, X. and Goldberg, A.B. *Introduction to Semi-Supervised Learning*. Morgan & Claypool, 2009.
31. Zouzias, A. and Freris, N.M. Randomized gossip algorithms for solving Laplacian systems. In *Proceedings of ECC* 2015.