Snake: a Stochastic Proximal Gradient Algorithm for Regularized Problems over Large Graphs

Adil Salim, Pascal Bianchi and Walid Hachem

***Abstract*—A regularized optimization problem over a large unstructured graph is studied, where the regularization term is tied to the graph geometry. Typical regularization examples include the total variation and the Laplacian regularizations over the graph. When the graph is a simple path without loops, efficient off-the-shelf algorithms can be used. However, when the graph is large and unstructured, such algorithms cannot be used directly. In this paper, an algorithm, referred to as "Snake", is proposed to solve such regularized problems over general graphs. The algorithm consists in properly selecting random simple paths in the graph and performing the proximal gradient algorithm over these simple paths. This algorithm is an instance of a new general stochastic proximal gradient algorithm, whose convergence is proven. Applications to trend filtering and graph inpainting are provided among others. Numerical experiments are conducted over large graphs.**

I. Introduction

Many applications in the fields of machine learning [1]–[3], signal and image restoration [4]–[6], or trend filtering [7]–[12] require the solution of the following optimization problem. On an undirected graph $G = (V, E)$ with no self loops, where $V = \{1, \dots, N\}$ represents a set of N nodes ($N \in \mathbb{N}^*$) and E is the set of edges, find

$$\min_{x \in \mathbb{R}^V} F(x) + R(x, \phi), \tag{1}$$

where F is a convex and differentiable function on $\mathbb{R}^V$ representing a data fitting term, and the function $x \mapsto R(x, \phi)$ represents a regularization term of the form

$$R(x, \phi) = \sum_{\{i,j\} \in E} \phi_{\{i,j\}}(x(i), x(j)) \,,$$

where $\phi = (\phi_e)_{e \in E}$ is a family of convex and symmetric $\mathbb{R}^2 \to \mathbb{R}$ functions. The regularization term $R(x, \phi)$ will be called a ϕ-regularization in the sequel. These ϕ-regularizations often promote the sparsity or the smoothness of the solution. For instance, when $\phi_e(x, x') = w_e|x - x'|$ where $w = (w_e)_{e \in E}$ is a vector of positive weights, the function $R(\cdot, \phi)$ coincides with the weighted Total Variation (TV) norm. This kind of regularization is often used in programming problems over a graph which are intended to recover a piecewise constant signal across adjacent nodes [8]–[15]. Another example is the Laplacian regularization $\phi_e(x, x') = (x - x')^2$, or its normalized version obtained by rescaling x and x' by the degrees of each node in e respectively. Laplacian regularization tends to smoothen the solution in accordance with the graph geometry [1], [2].

The Forward-Backward (or proximal gradient) algorithm is one of the most popular approaches towards solving Problem (1). This algorithm produces the sequence of iterates

$$x_{n+1} = \mathrm{prox}_{\gamma R(\cdot, \phi)}(x_n - \gamma \nabla F(x_n)) \,, \tag{2}$$

where $\gamma > 0$ is a fixed step, and where

$$\mathrm{prox}_g(y) = \arg\min_x \left(g(x) + \frac{1}{2}\|x - y\|^2 \right)$$

is the well-known proximity operator applied to the proper, lower semicontinuous (lsc), and convex function g (here $\|\cdot\|$ is the standard Euclidean norm). When F satisfies a smoothness assumption, and when γ is small enough, it is indeed well-known that the sequence (x_n) converges to a minimizer of (1), assuming this minimizer exists.

Implementing the proximal gradient algorithm requires the computation of the proximity operator applied to $R(\cdot, \phi)$ at each iteration. When N is large, this computation is in general affordable only when the graph exhibits a simple structure. For instance, when $R(\cdot, \phi)$ is the TV norm, the so-called *taut-string* algorithm is an efficient algorithm for computing the proximity operator when the graph is one-dimensional (1D) [16] (see Figure 1) or when it is a two-dimensional (2D) regular grid [13]. Similar observations can be made for the Laplacian regularization [17], where, *e.g.*, the discrete cosine transform can be implemented. When the graph is large and unstructured, these algorithms cannot be used, and the computation of the proximity operator is more difficult ([8], [18]).

This problem is addressed in this paper.[1] Towards obtaining a simple algorithm, we first express the functions $F(\cdot)$ and $R(\cdot, \phi)$ as the expectations of functions defined on a random walks in the graph, paving the way for a *randomized* version of the proximal gradient algorithm. Stochastic online algorithms in the spirit of this algorithm are often considered as simple and reliable procedures for solving high dimensional machine learning problems, including in the situations where the randomness is not inherent to these problems [20], [21]. One specificity of the algorithm developed here lies in that it

A. Salim and P. Bianchi are with the LTCI, Télécom ParisTech, Université Paris-Saclay, 75013, Paris, France (adil.salim, pascal.bianchi@telecom-paristech.fr).

W. Hachem is with the CNRS / LIGM (UMR 8049), Université Paris-Est Marne-la-Vallée, France (walid.hachem@u-pem.fr).

This work was supported by the Agence Nationale pour la Recherche, France, (ODISSEE project, ANR-13-ASTR-0030) and by the Labex Digiteo-DigiCosme (OPALE project), Université Paris-Saclay.

The authors are grateful to TeraLab DataScience for their material support.

[1] Note that a preliminary version of this work was published in [19], without proofs, and only focused on the TV-regularization problem. In comparison, the present paper provides proofs, extends the results to more general ϕ-regularizations, includes an arbitrary data-fitting term F, provides discussion about the complexity and the choice of hyperparameters and, finally, provides more numerical results and applicative contexts.

reconciles two requirements: on the one hand, the random versions of $R(\cdot, \phi)$ should be defined on *simple paths*, *i.e.*, on walks without loops (see Figure 1), in a way to make benefit of the power of the existing fast algorithms for computing the proximity operator. Owing to the existence of a procedure for selecting these simple paths, we term our algorithm as the "Snake" algorithm. On the other hand, the expectations of the functions handled by the optimization algorithm coincide with $F(\cdot)$ and $R(\cdot, \phi)$ respectively (up to a multiplicative constant), in such a way that the algorithm does not introduce any bias on the estimates.

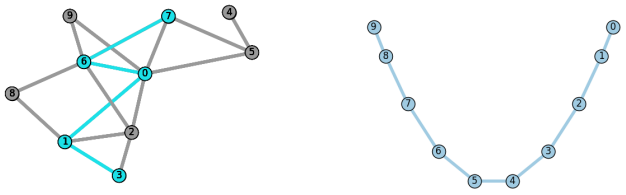

Fig. 1. Left: General graph on which is colored the simple path 3-1-0-6-7. Right: 1D-graph.

There often exists efficient methods to compute the proximity operator of ϕ-regularization over 1D-graphs. The algorithm Snake randomly selects simple paths in a general graph in order to apply the latter 1D efficient methods over a general graph.

Actually, the algorithm Snake will be an instance of a new general stochastic approximation algorithm that we develop in this paper. In some aspects, this general stochastic approximation algorithm is itself a generalization of the random Forward-Backward algorithm studied in [22].

Before presenting our approach, we provide an overview of the literature dealing with our problem. First consider the case where $R(\cdot, \phi)$ coincides with the TV norm. As said above, fast methods exist when the graph has a simple structure. We refer the reader to [13] for an overview of iterative solvers of Problem (1) in these cases. In [23], the author introduces a dynamical programming method to compute the proximity operator on a 1D-graph with a complexity of order $\mathcal{O}(N)$. Still in the 1D case, Condat [16] revisited recently an algorithm that is due to Mammen and Van De Geer [24] referred to as the taut-string algorithm. The complexity of this algorithm is $\mathcal{O}(N^2)$ in the worst-case scenario, and $\mathcal{O}(N)$ in the most realistic cases. The taut-string algorithm is linked to a total variation regularized problem in [25]. This algorithm is generalized to 2D-grids, weighted TV norms and ℓ^p TV norms by Barbero and Sra in [13]. To generalize to 2D-grids, the TV regularization can be written as a sum of two terms on which one can apply 1D methods, according to [26] and [27]. Over general graphs, there is no immediate way to generalize the taut string method. The problem of computing the TV-proximity operator over a general graph is addressed in [8].

The authors of [8] suggest to solve the problem using a projected Newton algorithm applied to the dual problem. It is observed that, empirically, this methods performs better than other concurrent approaches. As a matter of fact, this statement holds when the graph has a moderate size. As far as large graphs are concerned, the iteration complexity of the projected Newton method can be a bottleneck. To address this problem, the authors of [14] and [3] propose to solve the problem distributively over the nodes using the Alternating Direction Method of Multipliers (ADMM).

In [12] the authors propose to compute a decomposition of the graph in 1D-graphs and then solve Problem (1) by means of the TV-proximity operators over these 1D-graphs. Although the decomposition of the graph is fast in many applications, the algorithm [12] relies on an offline decomposition of the whole graph that needs a global knowledge of the graph topology. The Snake algorithm obtains this decomposition online. In [11], the authors propose a working set strategy to compute the TV-proximity operator. At each iteration, the graph is cut in two well-chosen subgraphs and a reduced problem of (1) is deduced from this cut. The reduced problem is then solved efficiently. This method has shown speed-ups when G is an image (*i.e* a two dimensional grid). Although the decomposition of the graph is not done during the preprocessing time, the algorithm [11] still needs a global knowledge of the graph topology during the iterations. On the contrary, the Snake algorithm only needs a local knowledge. Finally, in [9], the authors propose to replace the computation of the TV-proximity operator over the graph G by the computation of the TV-proximity operator over an 1D-subgraph of G well chosen. This produces an approximation of the solution whereas the Snake algorithm is proven to converge to the exact solution.

In the case where $R(\cdot, \phi)$ is the Laplacian regularization, the computation of the proximity operator of R reduces to the resolution of a linear system $(\mathcal{L} + \alpha I)x = b$ where $\mathcal{L}$ is the Laplacian matrix of the graph G and I the identity matrix. On an 1D-graph, the latter resolution can be done efficiently and relies on an explicit diagonalization of $\mathcal{L}$ ([17]) by means of the discrete cosine transform, which take $\mathcal{O}(N \log(N))$ operations. Over general graphs, the problem of computing the proximity operator of the Laplacian regularization is introduced in [2]. There exist fast algorithms to solve it due to [28]. They are based on recursively preconditioning the conjugate gradient method using graph theoretical results [18]. Nevertheless, the preconditioning phase which may be demanding over very large graphs. Compared to [18], our online method Snake requires no preprocessing step.

II. Outline of the approach and paper organization

The starting point of our approach is a new stochastic optimization algorithm that has its own interest. This algorithm will be presented succinctly here, and more rigorously in Sec. III below. Given an integer $L > 0$, let $\xi = (\xi^1, \ldots, \xi^L)$ be a random vector where the ξ^i are valued in some measurable space. Consider the problem

$$\min_x \sum_{i=1}^{L} \mathbb{E}_\xi \left[f_i(x, \xi^i) + g_i(x, \xi^i) \right] \tag{3}$$

where the $f_i(\cdot, \xi^i)$ are convex and differentiable, and the $g_i(\cdot, \xi^i)$ are convex. Given $\gamma > 0$, define the operator $\mathsf{T}_{\gamma,i}(x, s) = \mathrm{prox}_{\gamma g_i(\cdot,s)}(x - \gamma \nabla f_i(x, s))$. Given a sequence

(ξ_n) of independent copies of ξ, and a sequence of positive steps $(\gamma_n) \in \ell^2 \setminus \ell^1$, we consider the algorithm

$$x_{n+1} = \mathsf{T}_{\gamma_{n+1}}(x_n, \xi_{n+1}) \,, \tag{4}$$

where

$$\mathsf{T}_\gamma(\cdot, (s^1, \dots, s^L)) = \mathsf{T}_{\gamma,L}(\cdot, s^L) \circ \cdots \circ \mathsf{T}_{\gamma,1}(\cdot, s^1)$$

and where $\circ$ stands for the composition of functions: $f \circ g(\cdot) = f(g(\cdot))$. In other words, an iteration of this algorithm consists in the composition of L random proximal gradient iterations. The case where $L = 1$ was treated in [22].

Assuming that the set of minimizers of the problem is non empty, Th. 1 below states that the sequence (x_n) converges almost surely to a (possibly random) point of this set. The proof of this theorem which is rather technical is deferred to Sec. VII. It follows the same canvas as the approach of [22], with the difference that we are now dealing with possibly different functions (f_i, g_i) and non-independent noises ξ^i for $i \in \{1, \dots, L\}$.

We now want to exploit this stochastic algorithm to develop a simple procedure leading to a solution of Problem (1). This will be done in Sec. IV and will lead to the Snake algorithm. The first step is to express the function $R(\cdot, \phi)$ as the expectation of a function with respect to a finite random walk. Given an integer $M > 0$ and a finite walk $s = (v_0, v_1, \dots, v_M)$ of length M on the graph G, where $v_i \in V$ and $\{v_i, v_{i+1}\} \in E$, write

$$R(x, \phi_s) = \sum_{i=1}^{M} \phi_{\{v_{i-1}, v_i\}}(x(v_{i-1}), x(v_i)) \,.$$

Now, pick a node at random with a probability proportional to the degree (*i.e.*, the number of neighbors) of this node. Once this node has been chosen, pick another one at random uniformly among the neighbors of the first node. Repeat the process of choosing neighbors M times, and denote as $\xi \in V^{M+1}$ the random walk thus obtained. With this construction, we get that $\frac{1}{|E|} R(x, \phi) = \frac{1}{M} \mathbb{E}_\xi[R(x, \phi_\xi)]$ using some elementary Markov chain formalism (see Prop. 2 below).

In these conditions, a first attempt of the use of Algorithm (4) is to consider Problem (1) as an instance of Problem (3) with $L = 1$, $f_1(x, \xi) = \frac{1}{|E|} F(x)$, and $g_1(x, \xi) = \frac{1}{M} R(x, \phi_\xi)$. Given an independent sequence (ξ_n) of walks having the same law as ξ and a sequence (γ_n) of steps in $\ell^2 \setminus \ell^1$, Algorithm 4 boils down to the stochastic version of the proximal gradient algorithm

$$x_{n+1} = \mathrm{prox}_{\gamma_{n+1} \frac{1}{M} R(\cdot, \phi_{\xi_{n+1}})}(x_n - \gamma_{n+1} \frac{1}{|E|} \nabla F(x_n)) \,. \tag{5}$$

By Th. 1 (or by [22]), the iterates x_n converge almost surely to a solution of Problem (1).

However, although simpler than the deterministic algorithm (2), this algorithm is still difficult to implement for many regularization functions. As said in the introduction, the walk ξ is often required to be a simple path. Obviously, the walk generation mechanism described above does not prevent ξ from having repeated nodes. A first way to circumvent this problem would be to generate ξ as a loop-erased walk on the graph. Unfortunately, the evaluation of the corresponding distribution is notoriously difficult. The generalization of Prop. 2 to loop-erased walks is far from being immediate.

As an alternative, we identify the walk ξ with the concatenation of at most M simple paths of maximal length that we denote as $\xi^1, \dots, \xi^M$, these random variables being valued in the space of all walks in G of length at most M:

$$\xi = (\xi^1, \xi^2, \dots, \xi^M) \,.$$

Here, in the most frequent case where the number of simple paths is strictly less than M, the last ξ^i's are conventionally set to a trivial walk, *i.e.*, a walk with one node and no edge. We also denote as $\ell(\xi^i)$ the length of the simple path ξ^i, *i.e.*, the number of edges in ξ^i. We now choose $L = M$, and for $i = 1, \dots, L$, we set $f_i(x, \xi^i) = \frac{\ell(\xi^i)}{L|E|} F(x)$ and $g_i(x, \xi^i) = \frac{1}{L} R(x, \phi_{\xi^i})$ if $\ell(\xi^i) > 0$, and $f_i(x, \xi^i) = g_i(x, \xi^i) = 0$ otherwise. With this construction, we show in Sec. IV that $\frac{1}{|E|}(F(x) + R(x, \phi)) = \sum_{i=1}^{L} \mathbb{E}_\xi[f_i(x, \xi^i) + g_i(x, \xi^i)]$ and that the functions f_i and g_i fulfill the general assumptions required for the Algorithm (4) to converge to a solution of Problem (1). In summary, at each iteration, we pick up a random walk of length L according to the procedure described above, split it into simple paths of maximal length, and then we successively apply the proximal gradient algorithm to these simple paths.

After recalling the contexts of the taut-string and the Laplacian regularization algorithms (Sec. V), we simulate Algorithm (4) in several application contexts. First, we study the so called graph trend filtering [8], with the parameter k defined in [8] set to one. Then, we consider the graph inpainting problem [1], [2], [15]. These contexts are the purpose of Sec. VI. Finally, a conclusion and some future research directions are provided in Sec. VIII.

III. A General Stochastic Proximal Gradient Algorithm

Notations. We denote by $(\Omega, \mathcal{F}, \mathbb{P})$ a probability space and by $\mathbb{E}[\cdot]$ the corresponding expectation. We let $(\Xi, \mathcal{X})$ be an arbitrary measurable space. We denote X some Euclidean space and by $\mathscr{B}(\mathsf{X})$ its Borel σ-field. A mapping $f : \mathsf{X} \times \Xi \to \mathbb{R}$ is called a normal convex integrand if f is $\mathscr{B}(\mathsf{X}) \otimes \mathcal{X}$-measurable and if $f(\,.\,, s)$ is convex for all $s \in \Xi$ [29].

A. Problem and General Algorithm

In this section, we consider the general problem

$$\min_{x \in \mathsf{X}} \sum_{i=1}^{L} \mathbb{E}\left[f_i(x, \xi^i) + g_i(x, \xi^i)\right] \tag{6}$$

where L is a positive integer, the $\xi^i : \Omega \to \Xi$ are random variables (r.v.), and the functions $f_i : \mathsf{X} \times \Xi \to \mathbb{R}$ and $g_i : \mathsf{X} \times \Xi \to \mathbb{R}$ satisfy the following assumption:

Assumption 1. *The following holds for all $i \in \{1, \dots, L\}$:*

1) *The f_i and g_i are normal convex integrands.*
2) *For every $x \in \mathsf{X}$, $\mathbb{E}[|f_i(x, \xi^i)|] < \infty$ and $\mathbb{E}[|g_i(x, \xi^i)|] < \infty$.*
3) *For every $s \in \Xi$, $f_i(\cdot, s)$ is differentiable. We denote as $\nabla f_i(\cdot, s)$ its gradient w.r.t. the first variable.*

Remark. In this paper, we assume that the functions $g_i(\cdot,\xi)$ have a full domain for almost all ξ. This assumption can be relaxed with some effort, along the ideas developed in [22].

For every $i=1,\dots,L$ and every $\gamma>0$, we introduce the mapping $\mathsf{T}_{\gamma,i}:\mathsf{X}\times\Xi\to\mathsf{X}$ defined by

$$\mathsf{T}_{\gamma,i}(x,s)=\mathrm{prox}_{\gamma g_i(\cdot,s)}(x-\gamma\nabla f_i(x,s))\,.$$

We define $\mathsf{T}_\gamma:\mathsf{X}\times\Xi^L\to\mathsf{X}$ by

$$\mathsf{T}_\gamma(\cdot,(s^1,\dots,s^L))=\mathsf{T}_{\gamma,L}(\cdot,s^L)\circ\cdots\circ\mathsf{T}_{\gamma,1}(\cdot,s^1)\,.$$

Let ξ be the random vector $\xi=(\xi^1,\dots,\xi^L)$ with values in Ξ^L and let $(\xi_n:n\in\mathbb{N}^*)$ be a sequence of i.i.d. copies of ξ, defined on the same probability space $(\Omega,\mathcal{F},\mathbb{P})$. For all $n\in\mathbb{N}^*$, $\xi_n=(\xi_n^1,\dots,\xi_n^L)$. Finally, let (γ_n) be a positive sequence. Our aim is to analyze the convergence of the iterates (x_n) recursively defined by:

$$x_{n+1}=\mathsf{T}_{\gamma_{n+1}}(x_n,\xi_{n+1})\,, \tag{7}$$

as well as the intermediate variables $\bar{x}_{n+1}^i$ $(i=0,\dots,L)$ defined by $\bar{x}_{n+1}^0=x_n$, and

$$\bar{x}_{n+1}^i=\mathsf{T}_{\gamma_{n+1},i}(\bar{x}_{n+1}^{i-1},\xi_{n+1}^i)\,,\quad i=1,\dots,L\,. \tag{8}$$

In particular, $x_{n+1}=x_{n+1}^L=T_{\gamma_{n+1},L}(\bar{x}_{n+1}^{L-1},\xi_{n+1}^L)$.

In the special case where the functions g_i, f_i are all constant with respect to s (the algorithm is deterministic), the above iterations were studied by Passty in [30]. In the special case where $L=1$, the algorithm boils down to the stochastic Forward-Backward algorithm, whose detailed convergence analysis can be found in [22] (see also [31], and [32] as an earlier work). In this case, the iterates take the simpler form

$$x_{n+1}=\mathrm{prox}_{\gamma_{n+1}g_1(\cdot,\xi_{n+1})}(x_n-\gamma_{n+1}\nabla f_1(x_n,\xi_{n+1}))\,, \tag{9}$$

and converge a.s. to a minimizer of $\mathbb{E}[f_1(x,\xi)+g_1(x,\xi)]$ under the convenient hypotheses.

It is worth noting that the present algorithm (7) cannot be written as an instance of (9). Indeed, the operator T_γ is a composition of L (random) operators, whereas the stochastic forward backward algorithm (9) has a simpler structure. This composition raises technical difficulties that need to be specifically addressed. Among these difficulties is the dependency of the intermediate variables.

B. Almost sure convergence

We make the following assumptions.

Assumption 2. *The positive sequence (γ_n) satisfies the conditions*

$$\sum\gamma_n=+\infty\ \ \textit{and}\ \ \sum\gamma_n^2<\infty\,,$$

*(*i.e.*, $(\gamma_n)\in\ell^2\setminus\ell^1$). Moreover, $\frac{\gamma_{n+1}}{\gamma_n}\to 1$*

Assumption 3. *The following holds for all $i\in\{1,\dots,L\}$:*

1) *There exists a measurable map $K_i:\Xi\to\mathbb{R}_+$ s.t. the following holds $\mathbb{P}$-a.e.: for all x,y in X,*

$$\|\nabla f_i(x,\xi_i)-\nabla f_i(y,\xi_i)\|\le K_i(\xi_i)\|x-y\|\,.$$

2) *For all $\alpha>0$, $\mathbb{E}[K_i(\xi_i)^\alpha]<\infty$.*

We denote by $\mathcal{Z}$ the set of minimizers of Problem (6). Thanks to Ass. 1, the qualification conditions hold, ensuring that a point $x_\star$ belongs to $\mathcal{Z}$ *iff*

$$0\in\sum_{i=1}^L\nabla\mathbb{E}[f_i(x_\star,\xi^i)]+\partial\mathbb{E}[g_i(x_\star,\xi^i)]\,.$$

The (sub)differential and the expectation operators can be interchanged [33], and the above optimality condition also reads

$$0\in\sum_{i=1}^L\mathbb{E}[\nabla f_i(x_\star,\xi^i)]+\mathbb{E}[\partial g_i(x_\star,\xi^i)]\,, \tag{10}$$

where $\mathbb{E}[\partial g_i(x_\star,\xi^i)]$ is the Aumann expectation of the random set $\partial g_i(x_\star,\xi^i)$, defined as the set of expectations of the form $\mathbb{E}[\varphi_i(\xi^i)]$, where $\varphi_i:\Xi\to\mathsf{X}$ is a measurable map s.t. $\varphi_i(\xi^i)$ is integrable and

$$\varphi_i(\xi^i)\in\partial g_i(x_\star,\xi^i)\ \mathbb{P}\text{-a.e.},\ \forall i. \tag{11}$$

Therefore, the optimality condition (10) means that there exist L integrable mappings $\varphi_1,\dots,\varphi_L$ satisfying (11) and s.t.

$$0=\sum_{i=1}^L\mathbb{E}[\nabla f_i(x_\star,\xi^i)]+\mathbb{E}[\varphi_i(\xi^i)]\,. \tag{12}$$

When (11)-(12) hold, we say that the family $(\nabla f_i(x_\star,\xi^i),\varphi_i(\xi^i))_{i=1,\dots,L}$ is a *representation* of the minimizer $x_\star$. In addition, if for some $\alpha\ge1$ and every $i=1,\dots,L$, $\mathbb{E}[\|\nabla f_i(x_\star,\xi^i)\|^\alpha]<\infty$ and $\mathbb{E}[\|\varphi(\xi^i)\|^\alpha]<\infty$, we say that the minimizer $x_\star$ admits a α-integrable representation.

Assumption 4. 1) *The set $\mathcal{Z}$ is not empty.*

2) *For every $x_\star\in\mathcal{Z}$, there exists $\varepsilon>0$ s.t. $x_\star$ admits a $(2+\varepsilon)$-integrable representation $(\nabla f_i(x_\star,\xi^i),\varphi_i(\xi^i))_{i=1,\dots,L}$.*

We denote by $\partial g_i^0(x,\xi^i)$ the least norm element in $\partial g_i(x,\xi^i)$.

Assumption 5. *For every compact set $\mathcal{K}\subset\mathsf{X}$, there exists $\eta>0$ such that for all $i=1,\dots,L$,*

$$\sup_{x\in\mathcal{K}}\mathbb{E}[\|\partial g_i^0(x,\xi^i)\|^{1+\eta}]<\infty\,.$$

We can now state the main result of this section, which will be proven in Sec. VII.

Theorem 1. *Let Ass. 1–5 hold true. There exists a r.v. $X_\star$ s.t. $\mathbb{P}(X_\star\in\mathcal{Z})=1$ and s.t. (x_n) converges a.s. to $X_\star$ as $n\to\infty$. Moreover, for every $i=0,\dots,L-1$, $\bar{x}_n^i$ converges a.s. to $X_\star$.*

IV. The Snake Algorithm

A. Notations

Let $\ell\ge1$ be an integer. We refer to a walk of length ℓ over the graph G as a sequence $s=(v_0,v_1,\dots,v_\ell)$ in $V^{\ell+1}$ such that for every $i=1,\dots,\ell$, the pair $\{v_{i-1},v_i\}$ is an edge of the graph. A walk of length zero is a single vertex.

We shall often identify s with the graph $\mathcal{G}(s)$ whose vertices and edges are respectively given by the sets $\mathcal{V}(s) = \{v_0, \dots, v_\ell\}$ and $\mathcal{E}(s) = \{\{v_0, v_1\}, \dots, \{v_{\ell-1}, v_\ell\}\}$.

Let $L \geq 1$. We denote by Ξ the set of all walks over G with length $\leq L$. This is a finite set. Let $\mathcal{X}$ be the set of all subsets of Ξ. We consider the measurable space $(\Xi, \mathcal{X})$.

Let $s = (v_0, v_1, \dots, v_\ell) \in \Xi$ with $0 < \ell \leq L$. We abusively denote by ϕ_s the family of functions $(\phi_{\{v_{i-1}, v_i\}})_{i=1,\dots,\ell}$. We refer to the ϕ_s-regularization of x as the ϕ_s-regularization on the graph s of the restriction of x to s that is

$$R(x, \phi_s) = \sum_{i=1}^{\ell} \phi_{\{v_{i-1}, v_i\}}(x(v_{i-1}), x(v_i))\,.$$

Besides, $R(x, \phi_s)$ is defined to be 0 if s is a single vertex (that is $\ell = 0$).

We say that a walk is a *simple path* if there is no repeated node *i.e*, all elements in s are different or if s is a single vertex. Throughout the paper, we assume that when s is a simple path, the computation of $\mathrm{prox}_{R(.,\phi_s)}$ can be done easily.

B. Writing the Regularization Function as an Expectation

One key idea of this paper is to write the function $R(x, \phi)$ as an expectation in order to use a stochastic approximation algorithm, as described in Sec. III.

Denote by $\deg(v)$ the degree of the node $v \in V$, *i.e.*, the number of neighbors of v in G. Let π be the probability measure on V defined as

$$\pi(v) = \frac{\deg(v)}{2|E|}, \quad v \in V\,.$$

Define the probability transition kernel P on V^2 as $P(v, w) = \mathbb{1}_{\{v,w\}\in E}/\deg(v)$ if $\deg(v) > 0$, and $P(v, w) = \mathbb{1}_{v=w}$ otherwise, where $\mathbb{1}$ is the indicator function.

We refer to a Markov chain (indexed by $\mathbb{N}$) over V with initial distribution π and transition kernel P as an infinite random walk over G. Let $(v_k)_{k\in\mathbb{N}}$ be an infinite random walk over G defined on the canonical probability space $(\Omega, \mathcal{F}, \mathbb{P})$, with $\Omega = V^{\mathbb{N}}$. The first node v_0 of this walk is randomly chosen in V according to the distribution π. The other nodes are drawn recursively according to the conditional probability $\mathbb{P}(v_{k+1} = w \,|\, v_k) = P(v_k, w)$. In other words, conditionally to v_k, the node v_{k+1} is drawn uniformly from the neighborhood of v_k. Setting an integer $L \geq 1$, we define the random variable ξ from $(v_k)_{k\in\mathbb{N}}$ as $\xi = (v_0, v_1, \dots, v_L)$.

Proposition 2. *For every $x \in \mathbb{R}^V$,*

$$\frac{1}{|E|} R(x, \phi) = \frac{1}{L}\, \mathbb{E}[R(x, \phi_\xi)]\,. \tag{13}$$

Proof. It is straightforward to show that π is an invariant measure of the Markov chain $(v_k)_{k\in\mathbb{N}}$. Moreover, $\mathbb{P}(v_k = w, v_{k-1} = v) = \pi(v)P(v, w) = \mathbb{1}_{\{v,w\}\in E}/(2|E|)$, leading to the identity

$$\mathbb{E}\left[\phi_{\{v_{k-1}, v_k\}}(x(v_{k-1}), x(v_k))\right] = \frac{1}{|E|} R(x, \phi)\,,$$

which completes the proof by symmetry of $\phi_e, \forall e \in E$. □

This proposition shows that Problem (1) is written equivalently

$$\min_{x\in\mathbb{R}^V} \frac{1}{|E|} F(x) + \mathbb{E}[\frac{1}{L} R(x, \phi_\xi)]. \tag{14}$$

Hence, applying the stochastic proximal gradient algorithm to solve (14) leads to a new algorithm to solve (1), which was mentioned in Sec. II, Eq. (5):

$$x_{n+1} = \mathrm{prox}_{\gamma_{n+1}\frac{1}{L} R(\cdot, \phi_{\xi_{n+1}})}(x_n - \gamma_{n+1} \frac{1}{|E|} \nabla F(x_n))\,. \tag{15}$$

Although the iteration complexity is reduced in (15) compared to (2), the computation of the proximity operator of the ϕ-regularization over the random subgraph ξ_{n+1} in the algorithm (15) can be difficult to implement. This is due to the possible presence of loops in the random walk ξ. As an alternative, we split ξ into several simple paths. We will then replace the proximity operator over ξ by the series of the proximity operators over the simple paths induced by ξ, which are efficiently computable.

C. Splitting ξ into Simple Paths

Let $(v_k)_{k\in\mathbb{N}}$ be an infinite random walk on $(\Omega, \mathcal{F}, \mathbb{P})$. We recursively define a sequence of stopping time $(\tau_i)_{i\in\mathbb{N}}$ as $\tau_0 = 1$ and for all $i \geq 0$,

$$\tau_{i+1} = \min\{k \geq \tau_i :\, v_k \in \{v_{\tau_i - 1}, \dots, v_{k-1}\}\}$$

if the above set is nonempty, and $\tau_{i+1} = +\infty$ otherwise. We now define the stopping times t_i for all $i \in \mathbb{N}$ as $t_i = \min(\tau_i, L + 1)$. Finally, for all $i \in \mathbb{N}^*$ we can consider the random variable ξ^i on $(\Omega, \mathcal{F}, \mathbb{P})$ with values in $(\Xi, \mathcal{X})$ defined by

$$\xi^i = (v_{t_{i-1}-1}, v_{t_{i-1}}, \dots, v_{t_i - 1}).$$

We denote by N the smallest integer n such that $t_n = L + 1$. We denote by $\ell(\xi^i)$ the length of the simple path ξ^i.

Example. Given a graph with vertices $V = \{a, b, c, \dots, z\}$ and a given edge set that is not useful to describe here, consider $\omega \in \Omega$ and the walk $\xi(\omega) = (c, a, e, g, a, f, a, b, h)$ with length $L = 8$. Then, $t_0(\omega) = 1$, $t_1(\omega) = 4$, $t_2(\omega) = 6$, $t_3(\omega) = t_4(\omega) = \ldots = 9$, and $\xi(\omega)$ can be decomposed into $N(\omega) = 3$ simple paths and we have $\xi^1(\omega) = (c, a, e, g)$, $\xi^2(\omega) = (g, a, f)$, $\xi^3(\omega) = (f, a, b, h)$ and $\xi^4(\omega) = \ldots = \xi^8(\omega) = (h)$. Their respective lengths are $\ell(\xi^1(\omega)) = 3$, $\ell(\xi^2(\omega)) = 2$, $\ell(\xi^3(\omega)) = 3$ and $\ell(\xi^i(\omega)) = 0$ for all $i = 4, \dots, 8$. We identify $\xi(\omega)$ with $(\xi^1(\omega), \dots, \xi^8(\omega))$.

It is worth noting that, by construction, ξ^i is a simple path. Moreover, the following statements hold:

- We have $1 \leq N \leq L$ a.s.
- These three events are equivalent for all i: $\{\xi^i$ is a single vertex$\}$, $\{\ell(\xi^i) = 0\}$ and $\{i \geq N + 1\}$
- The last element of ξ^N is a.s. v_L
- $\sum_{i=1}^{L} \ell(\xi^i) = L$ a.s.

In the sequel, we identify the random vector $(\xi^1, \dots, \xi^L)$ with the random variable $\xi = (v_0, \dots, v_L)$. As a result, ξ is seen as a r.v with values in Ξ^L.

Our notations are summarized in Table I. For every $i =$

TABLE I
USEFUL NOTATIONS

$G=(V,E)$	Graph with no self-loop
s	walk on G
(v_i)	infinite random walk
$\xi=(\xi^1,\dots,\xi^L)$	random walk of length L
ξ^i	random simple path
$\ell(\xi^i)$	length of ξ^i
$R(x,\phi)$	$\phi-$regularization of x on G
$R(x,\phi_s)$	$\phi-$regularization of x along the walk s

$1,\dots,L$, define the functions f_i, g_i on $\mathbb{R}^V \times \Xi$ in such a way that

$$f_i(x,\xi^i) = \frac{\ell(\xi^i)}{L|E|} F(x) \tag{16}$$

$$g_i(x,\xi^i) = \frac{1}{L} R(x,\phi_{\xi^i}) \,. \tag{17}$$

Note that when $i > N(\omega)$ then $f_i(x,\xi^i(\omega)) = g_i(x,\xi^i(\omega)) = 0$.

Proposition 3. *For every $x \in \mathbb{R}^V$, we have*

$$\frac{1}{|E|}(F(x)+R(x,\phi)) = \sum_{i=1}^{L} \mathbb{E}\left[f_i(x,\xi^i)+g_i(x,\xi^i)\right] \,. \tag{18}$$

Proof. For every $\omega \in \Omega$ and every $x \in \mathbb{R}^V$,

$$\frac{1}{L} R(x,\phi_{\xi(\omega)}) = \frac{1}{L} \sum_{i=1}^{N(\omega)} R(x,\phi_{\xi^i(\omega)}) = \sum_{i=1}^{L} g_i(x,\xi^i(\omega)) \,.$$

Integrating, and using Prop. 2, it follows that $\sum_{i=1}^{L} \mathbb{E}[g_i(x,\xi^i)] = \frac{1}{|E|} R(x,\phi)$. Moreover, $\sum_{i=1}^{L} f_i(x,\xi^i(\omega)) = \frac{1}{|E|} F(x)$. This completes the proof. □

D. Main Algorithm

Prop. 3 suggests that minimizers of Problem (1) can be found by minimizing the right-hand side of (18). This can be achieved by means of the stochastic approximation algorithm provided in Sec. III. The corresponding iterations (7) read as $x_{n+1} = \mathsf{T}_{\gamma_{n+1}}(x_n, \xi_{n+1})$ where (ξ_n) are iid copies of ξ. For every $i = 1,\dots,L-1$, the intermediate variable $\bar{x}^i_{n+1}$ given by Eq. (8) satisfies

$$\bar{x}^i_{n+1} = \mathrm{prox}_{\gamma_n g_i(\,.\,,\xi^i_{n+1})}(\bar{x}^{i-1}_n - \gamma_n \nabla f_i(\bar{x}^{i-1}_n, \xi^i_{n+1})) \,.$$

Theorem 4. *Let Ass. 2 hold true. Assume that the convex function F is differentiable and that ∇F is Lipschitz continuous. Assume that Problem* (1) *admits a minimizer. Then, there exists a r.v. $X_\star$ s.t. $X_\star(\omega)$ is a minimizer of* (1) *for all ω $\mathbb{P}$-a.e., and s.t. the sequence (x_n) defined above converges a.s. to $X_\star$ as $n \to \infty$. Moreover, for every $i = 0,\dots,L-1$, $\bar{x}^i_n$ converges a.s. to $X_\star$.*

Proof. It is sufficient to verify that the mappings f_i, g_i defined by (16) and (17) respectively fulfill Ass. 1–5 of Th. 1. Then, Th. 1 gives the conclusion. Ass. 1 and 3 are trivially satisfied. It remains to show, for every minimizer $x_\star$, the existence of a $(2+\varepsilon)$-representation, for some $\varepsilon > 0$. Any such $x_\star$ satisfies Eq. (12) where φ_i satisfies (11). By definition of f_i and g_i, it is straightforward to show that there exists a deterministic constant $C_\star$ depending only on $x_\star$ and the graph G, such that $\|\nabla f_i(x_\star,\xi^i)\| < C_\star$ and $\|\varphi_i(\xi^i)\| < C_\star$. This proves Ass. 4. Ass. 5 can be easily checked by the same arguments. □

TABLE II
PROPOSED SNAKE ALGORITHM.

procedure SNAKE(x_0, L)
  $z \leftarrow x_0$
  $e \leftarrow$RND_ORIENTED_EDGE
  $n \leftarrow 0$
  $\ell \leftarrow L$
  while stopping criterion is not met **do**
    $c, e \leftarrow$ SIMPLE_PATH(e, ℓ)
    $z \leftarrow$ PROX1D($z - \gamma_n \frac{\text{LENGTH}(c)}{L|E|} \nabla F(z), c, \frac{1}{L}\gamma_n$)
    $\ell \leftarrow \ell -$ LENGTH(c)
    if $\ell = 0$ **then**
      $e \leftarrow$ RND_ORIENTED_EDGE
      $\ell \leftarrow L$
      $n \leftarrow n+1$ ▷ $x_n$is z at this step
    end if
  end while
  return z
end procedure

TABLE III
SIMPLE_PATH PROCEDURE.

procedure SIMPLE_PATH(e, ℓ)
  $c \leftarrow e$
  $w \leftarrow$ UNIFORM_NEIB($e[-1]$)
  while $w \notin c$ and LENGTH(c)$< \ell$ **do**
    $c \leftarrow [c, w]$
    $w \leftarrow$ UNIFORM_NEIB(w)
  end while
  return $c, [c[-1], w]$
end procedure

Consider the general ϕ-regularized problem (1), and assume that an efficient procedure to compute the proximity operator of the ϕ-regularization over an 1D-graph is available. The sequence (x_n) is generated by the algorithm SNAKE (applied with the latter 1D efficient procedure) and is summarized in Table II. Recall the definition of the probability π on V and the transition kernel P on V^2. The procedure presented in this table calls the following subroutines.

- If c is a finite walk, $c[-1]$ is the last element of c and LENGTH(c) is its length as a walk that is $|c|-1$.
- The procedure RND_ORIENTED_EDGE returns a tuple of two nodes randomly chosen (v,w) where $v \sim \pi$ and $w \sim P(v,.)$.
- For every $x \in \mathbb{R}^V$, every simple path s and every $\alpha > 0$, PROX1D(x,s,α) is any procedure that returns the quantity $\mathrm{prox}_{\alpha R(.,\phi_s)}(x)$.
- The procedure UNIFORM_NEIB(v) returns a random vertex drawn uniformly amongst the neighbors of the vertex v that is with distribution $P(v,.)$.
- The procedure SIMPLE_PATH(e,ℓ), described in Table III, generates the first steps of a random walk on G with transition kernel P initialized at the vertex $e[-1]$, and prefaced by the first node in e. It represents the ξ^i's of the previous section. The random walk is stopped when one node is repeated, or until the maximum number of samples $\ell+1$ is reached. The procedure produces two

outputs, the walk and the oriented edge $c, (c[-1], w)$. In the case where the procedure stopped due to a repeated node, c represents the simple path obtained by stopping the walk before the first repetition occurs, while w is the vertex which has been repeated (referred to as the pivot node). In the case where no vertex is repeated, it means that the procedure stopped because the maximum length was achieved. In that case, c represents the last simple path generated, and the algorithm doesn't use the pivot node w.

Remark. Although Snake converges for every value of the hyperparameter L, a natural question is about the influence of L on the behavior of the algorithm. In the case where $R(\,\cdot\,, \phi)$ is the TV regularization, [16] notes that, empirically, the taut-string algorithm used to compute the proximity operator has a complexity of order $O(L)$. The same holds for the Laplacian regularization. Hence, parameter L controls the complexity of every iteration. On the other hand, in the reformulation of Problem (1) into the stochastic form (13), the random variable $|E|R(x, \phi_\xi)/L$ is an unbiased estimate of $R(x, \phi)$. By the ergodic theorem, the larger L, the more accurate is the approximation. Hence, there is a trade-off between complexity of an iteration and precision of the algorithm. This trade-off is standard in the machine learning literature. It often appears while sampling mini-batches in order to apply the stochastic gradient algorithm to a deterministic optimization problem(see [20], [21]). The choice of L is somehow similar to the problem of the choice of the length of the mini-batches in this context.

Providing a theoretical rule that would optimally select the value of L is a difficult task that is beyond the scope of this paper. Nevertheless, in Sec. VI, we provide a detailed analysis of the influence of L on the numerical performance of the algorithm.

V. Proximity operator over 1D-graphs

We now provide some special cases of ϕ-regularizations, for which the computation of the proximity operator over 1D-graphs is easily tractable. Specifically, we address the case of the total variation regularization and the Laplacian regularization which are particular cases of ϕ-regularizations.

A. Total Variation norm

In the case where $\phi_{\{i,j\}}(x, x') = w_{\{i,j\}}|x - x'|$, $R(x, \phi)$ reduces to the weighted TV regularization

$$R(x, \phi) = \sum_{\{i,j\}\in E} w_{\{i,j\}}|x(i) - x(j)|$$

and in the case where $\phi_{\{i,j\}}(x, x') = |x-x'|$, $R(x, \phi)$ reduces to the its unweighted version

$$R(x, \phi) = \sum_{\{i,j\}\in E} |x(i) - x(j)|\,.$$

As mentioned above, there exists a fast method, the taut string algorithm, to compute the proximity operator of these $\phi-$regularizations over a 1D-graph ([13], [16]).

B. Laplacian regularization

In the case where $\phi_{\{i,j\}}(x, x') = w_{\{i,j\}}(x - x')^2$, $R(x, \phi)$ reduces to the Laplacian regularization that is

$$R(x, \phi) = \sum_{\{i,j\}\in E} w_{\{i,j\}}(x(i) - x(j))^2.$$

Its unweighted version is

$$\sum_{\{i,j\}\in E} (x(i) - x(j))^2 = \|\nabla x\|^2 = x^*\mathcal{L}x.$$

In the case where $\phi_{\{i,j\}}(x, x') = w_{\{i,j\}}(x/\sqrt{\deg(i)} - x'/\sqrt{\deg(j)})^2$,

$$R(x, \phi) = \sum_{\{i,j\}\in E} w_{\{i,j\}} \left(\frac{x(i)}{\sqrt{\deg(i)}} - \frac{x'(i)}{\sqrt{\deg(j)}}\right)^2$$

is the normalized Laplacian regularization.

We now explain one method to compute the proximity operator of the unweighted Laplacian regularization over an 1D-graph. The computation of the proximity operator of the normalized Laplacian regularization can be done similarly. The computation of the proximity operator of the weighted Laplacian regularization over an 1D-graph is as fast as the computation the proximity operator of the unweighted Laplacian regularization over an 1D-graph, using for example Thomas' algorithm.

The proximity operator of a fixed point $y \in \mathbb{R}^{\ell+1}$ is obtained as a solution to a quadratic programming problem of the form:

$$\min_{x\in\mathbb{R}^{\ell+1}} \frac{1}{2}\|x - y\|^2 + \lambda\sum_{k=1}^{\ell}(x(k-1) - x(k))^2\,,$$

where $\lambda > 0$ is a scaling parameter. Writing the first order conditions, the solution x satisfies

$$(I + 2\lambda\mathcal{L})x = y \tag{19}$$

where $\mathcal{L}$ is the Laplacian matrix of the 1D-graph with $\ell + 1$ nodes and I is the identity matrix in $\mathbb{R}^{\ell+1}$. By [17], $\mathcal{L}$ can be diagonalized explicitely. In particular, $I + 2\lambda\mathcal{L}$ has eigenvalues

$$1 + 4\lambda\left(1 - \cos\left(\frac{\pi k}{\ell + 1}\right)\right),$$

and eigenvectors $e_k \in \mathbb{R}^{\ell+1}$

$$e_k(j) = \frac{1}{2(\ell + 1)}\cos\left(\pi\frac{kj}{\ell + 1} - \pi\frac{k}{2(\ell + 1)}\right),$$

for $0 \le k < n$. Hence, $x = C^*\Lambda^{-1}Cy$, where Λ gathers the eigenvalues of $I + 2\lambda\mathcal{L}$ and the operators C and C^* are the discrete cosine transform operator and the inverse discrete cosine transform. The practical computation of x can be found in $O(\ell\log(\ell))$ operations.

VI. Examples

We now give some practical instances of Problem (1) by particularizing F and the ϕ-regularization in (1). We also provide some simulations to compare our method to existing algorithms. The code is available at the address https://github.com/adil-salim/Snake.

A. Trend Filtering on Graphs

Consider a vector $y \in \mathbb{R}^V$. The Graph Trend Filtering (GTF) estimate on V with parameter k set to one is defined in [8] by

$$\hat{y} = \arg\min_{x \in \mathbb{R}^V} \frac{1}{2}\|x - y\|^2 + \lambda \sum_{\{i,j\} \in E} |x(i) - x(j)|. \quad (20)$$

where $\lambda > 0$ is a scaling parameter. In the GTF context, the vector y represents a sample of noisy data over the graph G and the GTF estimate represents a denoised version of y. When G is an 1D or a 2D-graph, the GTF boils down to a well known context [4], [7]. When G is a general graph, the GTF estimate is studied in [8] and [10]. The estimate $\hat{y}$ is obtained as the solution of a TV-regularized risk minimization with $F(x) = \frac{1}{2}\|x - y\|^2$ where y is fixed. We address the problem of computing the GTF estimate on two real life graphs from [34] and one sampled graph. The first one is the Facebook graph which is a network of 4039 nodes and 88234 edges extracted from the Facebook social network. The second one is the Orkut graph with 3072441 nodes and 117185083 edges. Orkut was also an on-line social network. The third graph is sampled according to a Stochastic Block Model (SBM). Namely we generate a graph of 4000 nodes with four well-separated clusters of 1000 nodes (also called "communities") as depicted in Fig. (2). Then we draw independently N^2 Bernoulli r.v. $E(i,j)$, encoding the edges of the graph (an edge between nodes i and j is present iff $E(i,j) = 1$), such that $\mathbb{P}\{E(i,j) = 1\} = P(c_i, c_j)$ where c_i denotes the community of the node i and where

$$\begin{cases} P(c,c') = & .1 \text{ if } c = c' \\ P(c,c') = & .005 \text{ otherwise} \end{cases}$$

This model is called the stochastic block model for the matrix P [35]. It amounts to a blockwise Erdös-Rényi model with parameters depending only on the blocks. It leads to 81117 edges.

We assume that every node is provided with an unknown value in $\mathbb{R}$ (the set of all these values being referred to as the *signal* in the sequel). In our example, the value $y(i)$ at node i is generated as $y(i) = l(c_i) + \sigma\epsilon_i$ where l is a mapping from the communities to a set of levels (in Fig. 2, $l(i)$ is an integer in $[0, 255]$), and ϵ denotes a standard Gaussian white noise with $\sigma > 0$ as its standard deviation. In Figure 2 we represent an example of the signal y (left figure) along with the "initial" values $l(c_i)$ represented in grayscale at every node.

Over the two real life graphs, the vector y is sampled according to a standard Gaussian distribution of dimension $|V|$. The parameter λ is set such that $\mathbb{E}[\frac{1}{2}\|x - y\|^2] = \mathbb{E}[\lambda \sum_{\{i,j\} \in E} |x(i) - x(j)|]$ if x, y are two independent r.v with standardized Gaussian distribution. The initial guess x_0 is set equal to y. The step size γ_n set equal to $|V|/(10n)$ for the two real life graphs and $|V|/(5n)$ for the SBM realization graph. We ran the Snake algorithm for different values of L, except over the Orkut graph where $L = |V|$.

The dual problem of (20) is quadratic with a box constraint. The Snake algorithm is compared to the well-known projected gradient (PG) algorithm for the dual problem. To solve the dual problem of (20), we use L-BFGS-B [36] as suggested in [8]. Note that, while running on the Orkut graph, the algorithm L-BFGS-B leads to a memory error from the solver [36] in SciPy (using one thread of a 2800 MHz CPU and 256GB RAM).

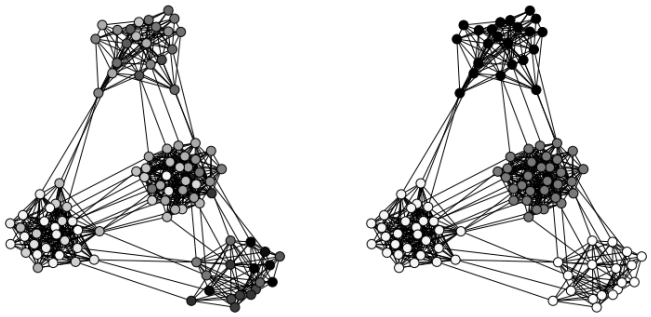

Fig. 2. The signal is the grayscale of the node The graph is sampled according to a SBM. Left: Noised signal over the nodes. Right: Sought signal.

Figures 3, 4 and 5 show the objective function as a function of time for each algorithm.

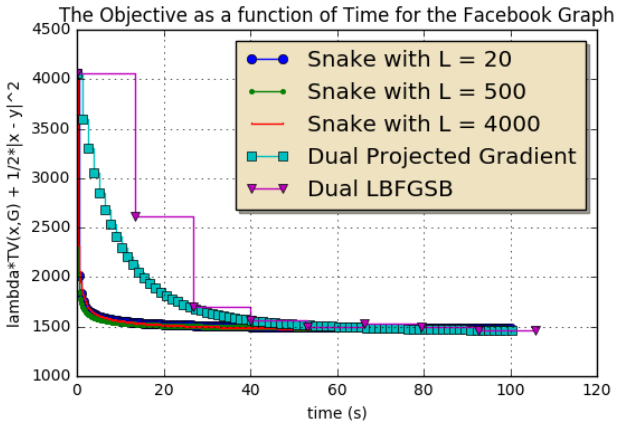

Fig. 3. Snake applied to the TV regularization over the Facebook Graph

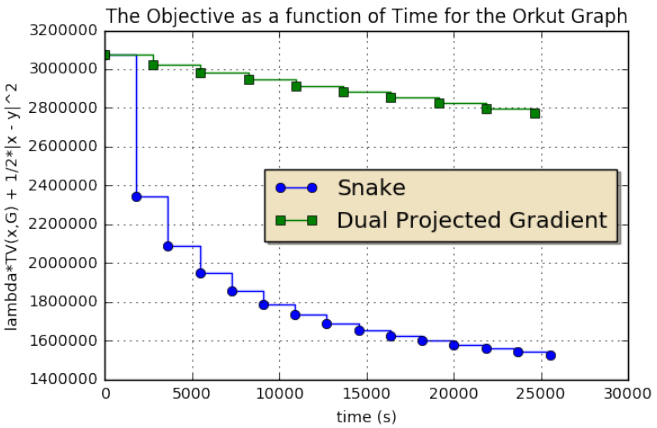

Fig. 4. Snake applied to the TV regularization over the Orkut Graph

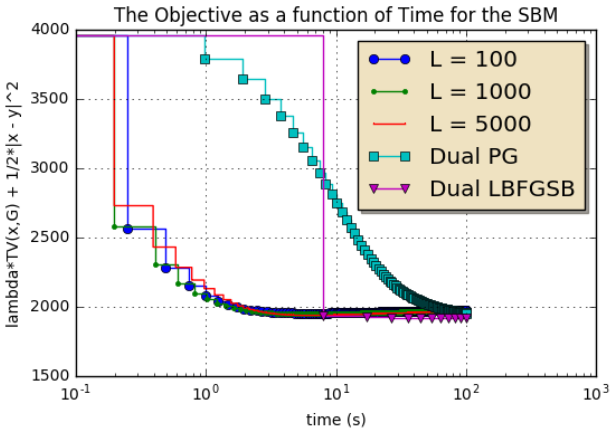

Fig. 5. Snake applied to the TV regularization over the SBM realization graph, in log scale

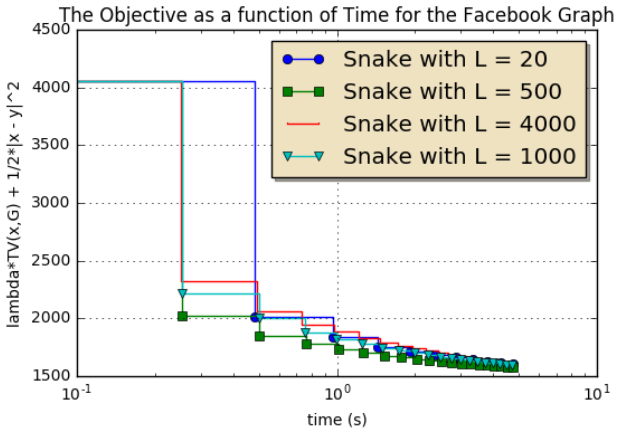

Fig. 6. Snake applied to the TV regularization over the Facebook graph, in log scale

In the case of the TV regularization, we observe that Snake takes advantage of being an online method, which is known to be twofold ([20], [21]). First, the iteration complexity is controlled even over large general graphs: the complexity of the computation of the proximity operator is empirically linear [16]. On the contrary, the projected gradient algorithm involves a matrix-vector product with complexity $O(|E|)$. Hence, *e.g* the projected gradient algorithm has an iteration complexity of at least $O(|E|)$. The iteration complexity of Snake can be set to be moderate in order to frequently get iterates while running the algorithm. Then, Snake is faster than L-BFGS-B and the projected gradient algorithms for the dual problem in the first iterations of the algorithms.

Moreover, for the TV regularization, Snake seems to perform globally better than L-BFGS-B and the projected gradient. This is because Snake is a proximal method where the proximity operator is efficiently computed ([37]).

The parameter L seems to have a minor influence on the performance of the algorithm since, in Figure 3 the curves corresponding to different values of L are closely superposed. The log scale in Figure 5 allows us to see that the curve corresponding Snake with $L = 1000$ performs slightly better that the others. Figure 6 shows supplementary curves in log scale where Snake is run over the Facebook graph with different values of L.

In Figure 6, the best performing value is $L = 500$.

Over the three graphs, the value $L = O(|V|)$ is a good value, if not the best value to use the Snake algorithm. One can show that, while sampling the first steps of the infinite random walk over G from the node, say v, the expected time of return to the random node v is $|V|$. Hence, the value $L = |V|$ allow Snake to significantly explore the graph during one iteration.

B. Graph Inpainting

The problem of graph inpainting has been studied in [1], [2], [15] and can be expressed as follows. Consider a vector $y \in \mathbb{R}^V$, a subset $O \subset V$. Let $\bar{O}$ be its complementary in V. The harmonic energy minimization problem is defined in [2] by

$$\begin{aligned} \min_{x \in \mathbb{R}^V} \quad & \sum_{\{i,j\} \in E} (x(i) - x(j))^2 \\ \text{subject to} \quad & x(i) = y(i), \forall i \in O. \end{aligned}$$

This problem is interpreted as follows. The signal $y \in \mathbb{R}^V$ is partially observed over the nodes and the aim is to recover y over the non observed nodes. The subset $O \subset V$ is the set of the observed nodes and $\bar{O}$ the set of unobserved nodes. An example is shown in Figure 7.

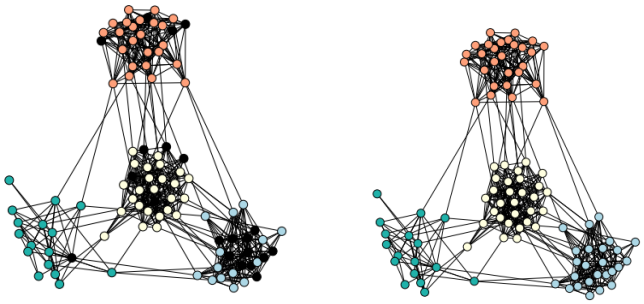

Fig. 7. Left: Partially observed data (unobserved nodes are black, data is the color of nodes). Right: Fully observed data (the color is observed for all nodes).

Denote by $G_{\bar{O}} = (\bar{O}, E_{\bar{O}})$ the subgraph of G induced by $\bar{O}$. Namely, $\bar{O}$ is the set of vertices, and the set $E_{\bar{O}}$ is formed by the edges $\{i,j\} \in E$ s.t. $i \in \bar{O}$ and $j \in \bar{O}$. The harmonic energy minimization is equivalent to the following Laplacian regularized problem over the graph $G_{\bar{O}}$:

$$\min_{x \in \mathbb{R}^{\bar{O}}} \quad F(x) + \sum_{\substack{\{i,j\} \in E_{\bar{O}} \\ i<j}} (x(i) - x(j))^2 \tag{21}$$

where

$$F(x) = \sum_{\substack{i \in \bar{O}, j \in O \\ \{i,j\} \in E}} (x(i) - y(j))^2 .$$

The signal y is sampled according to a standardized Gaussian distribution of dimension $|V|$. We compared the Snake algorithm to existing algorithm over the Orkut graph. The set V is divided in two parts of equal size to define O and $\bar{O}$. The initial guess is set equal to zero over the set of unobserved nodes $\bar{O}$, and to the restriction of y to O over the set of observed nodes O. We compare our algorithm with the conjugate gradient

Figures 8 and 9 represent the objective function $\sum_{\{i,j\}\in E}(x(i) - x(j))^2$ as a function of time. Over the Facebook graph, the parameter L is set equal to $|V|/10$. The step size γ_n are set equal to $|V|/(10n)$. Over the Orkut graph, L is set equal to $|V|/50$. The step size are set equal to $|V|/(5\sqrt{n})$ on the range displayed in Figure 8. Even if the sequence $(|V|/(5\sqrt{n}))_{n\in\mathbb{N}}$ does not satisfies the Ass. 2, it is a standard trick in stochastic approximation to take a slowly decreasing step size in the first iterations of the algorithm ([38]). It allows the iterates to be quickly close to the set of solutions without converging to the set of solutions. Then, one can continue the iterations using a step size satisfying Ass. 2 to make the algorithm converging. There is a trade-off between speed and precision while choosing the step-size.

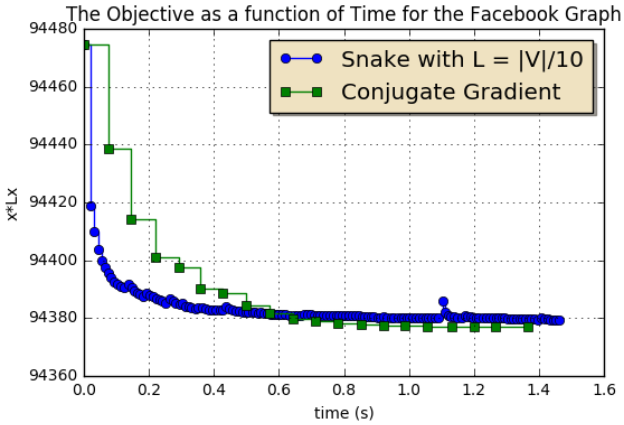

Fig. 8. Snake applied to the Laplacian regularization over the Facebook Graph

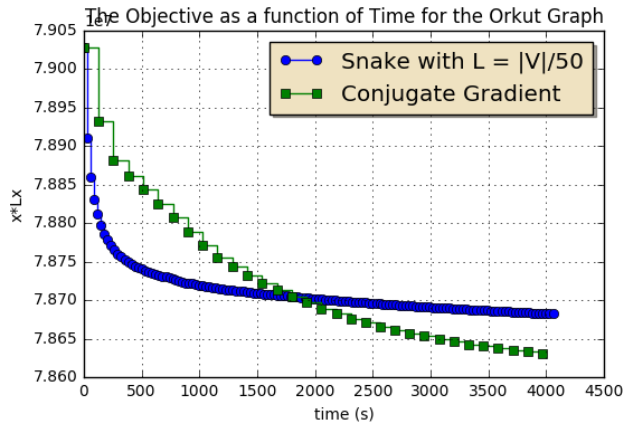

Fig. 9. Snake applied to the Laplacian regularization over the Orkut Graph

Snake turns out to be faster in the first iterations. Moreover, as an online method, it allows the user to control the iteration complexity of the algorithm. Since a discrete cosine transform is used, the complexity of the computation of the proximity operator is $O(L\log(L))$. In contrast, the iteration complexity of the conjugate gradient algorithm can be a bottleneck (at least $O(|E|)$) as far as very large graphs are concerned.

Besides, Snake for the Laplacian regularization does not perform globally better than the conjugate gradient. This is because the conjugate gradient is designed to fully take advantage of the quadratic structure. On the contrary, Snake is not specific to quadratic problems.

C. Online Laplacian solver

Let $\mathcal{L}$ the Laplacian of a graph $G = (V, E)$. The resolution of the equation $\mathcal{L}x = b$, where b is a zero mean vector, has numerous applications ([18], [39]). It can be found by minimizing the Laplacian regularized problem

$$\min_{x\in\mathbb{R}^V} -b^*x + \frac{1}{2}x^*\mathcal{L}x.$$

In our experiment, the vector b is sampled according to a standardized Gaussian distribution of dimension $|V|$. We compare our algorithm with the conjugate gradient over the Orkut graph.

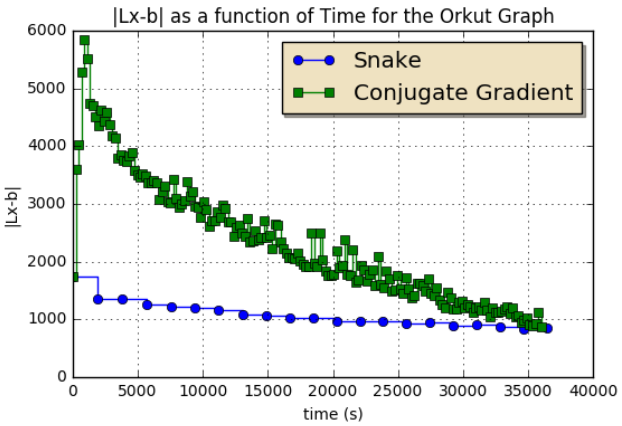

Fig. 10. Snake applied to the resolution of a Laplacian system over the Orkut graph

Figure 10 represents the quantity $\|\mathcal{L}x_n - b\|$ as a function of time, where x_n is the iterate provided either by Snake or by the conjugate gradient method. The parameter L is set equal to $|V|$. The step size γ_n are set equal to $|V|/(2n)$. Snake appears to be more stable than the conjugate gradient method, has a better performance at start up.

VII. Proof of Th. 1

We start with some notations. We endow the probability space $(\Omega, \mathscr{F}, \mathbb{P})$ with the filtration $(\mathscr{F}_n)$ defined as $\mathscr{F}_n = \sigma(\xi_1, \ldots, \xi_n)$, and we write $\mathbb{E}_n = \mathbb{E}[\cdot\,|\,\mathscr{F}_n]$. In particular, $\mathbb{E}_0 = \mathbb{E}$. We also define $G_i(x) = \mathbb{E}[g_i(x, \xi^i)]$ and $F_i(x) = \mathbb{E}[f_i(x, \xi^i)]$ for every $x \in \mathsf{X}$. We denote by μ_i and μ the probability laws of ξ^i and ξ respectively. Finally, C and η will refer to positive constants whose values can change from an

equation to another. The constant η can be chosen arbitrarily small.

In [22], the case $L = 1$ is studied (algorithm (9)). Here we shall reproduce the main steps of the approach of [22], only treating in detail the specificities of the case $L \geq 1$. We also note that in [22], we considered the so-called maximal monotone operators, which generalize the subdifferentials of convex functions. This formalism is not needed here.

The principle of the proof is the following. Given $a \in \mathsf{X}$, consider the so called differential inclusion (DI) defined on the set of absolutely continuous functions from $\mathbb{R}_+ = [0, \infty)$ to X as follows:

$$\left\{ \begin{array}{lcl} \dot{\mathsf{z}}(t) & \in & -\sum_{i=1}^{L} (\nabla F_i(\mathsf{z}(t)) + \partial G_i(\mathsf{z}(t))) \\ \mathsf{z}(0) & = & a\,. \end{array} \right. \tag{22}$$

It is well known that this DI has a unique solution, *i.e.*, a unique absolutely continuous mapping $\mathsf{z} : \mathbb{R}_+ \to \mathsf{X}$ such that $\mathsf{z}(0) = a$, and $\dot{\mathsf{z}}(t) \in -\sum(\nabla F_i(\mathsf{z}(t)) + \partial G_i(\mathsf{z}(t)))$ for almost all $t > 0$. Consider now the map $\Phi : \mathsf{X} \times \mathbb{R}_+ \to \mathsf{X}$, $(a, t) \mapsto \mathsf{z}(t)$, where $\mathsf{z}(t)$ is the DI solution with the initial value $\mathsf{z}(0) = a$. Then, Φ is a semi-flow [40], [41].

Let us introduce the following function I from $\mathsf{X}^{\mathbb{N}}$ to the space of $\mathbb{R}_+ \to \mathsf{X}$ continuous functions. For $u = (u_n) \in \mathsf{X}^{\mathbb{N}}$, the function $\mathsf{u} = \mathsf{I}(u)$ is the continuous interpolated process obtained from u as

$$\mathsf{u}(t) = u_n + \frac{u_{n+1} - u_n}{\gamma_{n+1}}(t - \tau_n)$$

for $t \in [\tau_n, \tau_{n+1})$, where $\tau_n = \sum_1^n \gamma_k$. Consider the interpolated function $\mathsf{x} = \mathsf{I}((x_n))$. We shall prove the two following facts:

- The sequence $(\|x_n - x_\star\|)$ is almost surely convergent for each $x_\star \in \mathcal{Z}$ (Prop. 6);
- The process $\mathsf{x}(t)$ is an almost sure Asymptotic Pseudo Trajectory (APT) of the semi-flow Φ, a concept introduced by Benaïm and Hirsch in the field of dynamical systems [42]. Namely, for each $T > 0$,

$$\sup_{u \in [0,T]} \|\mathsf{x}(t + u) - \Phi(\mathsf{x}(t), u)\| \xrightarrow[t\to\infty]{\text{a.s.}} 0, \tag{23}$$

Taken together, these two results lead to the a.s. convergence of (x_n) to some r.v. $X^\star$ supported by the set $\mathcal{Z}$, as is shown by [22, Cor. 3.2]. The convergence of the $(\bar{x}_n^i)_n$ stated by Th. 1 will be shown in the course of the proof.

Before entering the proof, we recall some well known facts relative to the so called Moreau envelopes. For more details, the reader is referred to *e.g.* [40, Ch. 2], or [37, Ch. 12]. The Moreau envelope of parameter γ of a convex function h with domain X is the function

$$h^\gamma(x) = \min_{w \in \mathsf{X}} h(w) + (2\gamma)^{-1}\|w - x\|^2\,.$$

The function h^γ is a differentiable function on X, and its gradient is given by the equation

$$\nabla h^\gamma(x) = \gamma^{-1}(x - \mathrm{prox}_{\gamma h}(x))\,. \tag{24}$$

This gradient is a γ^{-1}-Lipschitz continuous function satisfying the inequality $\|\nabla h^\gamma(x)\| \leq \|\partial h^0(x)\|$, where $\partial h^0(x)$ is the least-norm element of $\partial h(x)$. Finally, for all $(x, u) \in \mathsf{X} \times \mathsf{X}$ and for all $v \in \partial h(u)$, the inequality

$$\langle \nabla h^\gamma(x) - v, \mathrm{prox}_{\gamma h}(x) - u \rangle \geq 0 \tag{25}$$

holds true. With the formalism of the Moreau envelopes, the mapping $\mathsf{T}_{\gamma,i}$ can be rewritten as

$$\mathsf{T}_{\gamma,i}(x, s) = x - \gamma\nabla f_i(x, s) - \gamma\nabla g_i^\gamma(x - \gamma\nabla f_i(x, s), s) \tag{26}$$

thanks to (24), where $\nabla g_i^\gamma(\cdot, s)$ is the gradient of the Moreau envelope $g_i^\gamma(\cdot, s)$. We shall adopt this form in the remainder of the proof.

The following lemma is proven in Appendix A-A.

Lemma 5. *For $i = 1, \ldots, L$, let*

$$\bar{x}^i = (T_{\gamma,i}(\cdot, s^i) \circ \cdots \circ T_{\gamma,1}(\cdot, s^1))(x).$$

Then, with Ass. 3, there exists a measurable map $\kappa : \Xi^L \to \mathbb{R}_+$ s.t. $\mathbb{E}[\kappa(\xi)^\alpha] < \infty$ for all $\alpha \geq 1$ and s.t. for all $\bar{s} = (s^1, \ldots, s^L) \in \Xi^L$,

$$\|\nabla f_i(\bar{x}^{i-1}, s^i)\| \leq \kappa(\bar{s}) \sum_{k=1}^{i} \|\nabla f_k(x, s^k)\| + \|\nabla g_k^\gamma(x, s^k)\|$$

$$\|\nabla g_i^\gamma(\bar{x}^{i-1} - \gamma\nabla f_i(\bar{x}^{i-1}, s^i), s^i)\| \leq \kappa(\bar{s}) \sum_{k=1}^{i} \|\nabla f_k(x, s^k)\| + \|\nabla g_k^\gamma(x, s^k)\|.$$

Recall that we are studying the iterations $\bar{x}_{n+1}^i = \mathsf{T}_{\gamma_{n+1},i}(\bar{x}_{n+1}^{i-1}, \xi_{n+1}^i)$, for $i = 1, \ldots, L$, $n \in \mathbb{N}^*$, with $\bar{x}_{n+1}^0 = x_n$ and $x_{n+1} = \bar{x}_{n+1}^L$. In this section and in Appendix A, we shall write for conciseness, for any $x_\star \in \mathcal{Z}$,

$$\begin{aligned} \nabla g_i^\gamma &= \nabla g_i^{\gamma_{n+1}}(\bar{x}_{n+1}^{i-1} - \gamma_{n+1}\nabla f_i(\bar{x}_{n+1}^{i-1}, \xi_{n+1}^i), \xi_{n+1}^i), \\ \mathrm{prox}_{\gamma g_i} &= \mathrm{prox}_{\gamma g_i(\cdot, \xi_{n+1}^i)}(\bar{x}_{n+1}^{i-1} - \gamma_{n+1}\nabla f_i(\bar{x}_{n+1}^{i-1}, \xi_{n+1}^i)), \\ \nabla f_i &= \nabla f_i(\bar{x}_{n+1}^{i-1}, \xi_{n+1}^i), \\ \nabla f_i^\star &= \nabla f_i(x_\star, \xi_{n+1}^i) \text{ where } x_\star \in \mathcal{Z}, \\ \varphi_i &= \varphi_i(\xi_{n+1}^i), \text{ (see Ass. 4) and} \\ \gamma &= \gamma_{n+1}. \end{aligned}$$

The following proposition is analoguous to [31, Prop. 1] or to [22, Prop. 6.1]:

Proposition 6. *Let Ass. 2–4 hold true. Then the following facts hold true:*

1) *For each $x_\star \in \mathcal{Z}$, the sequence $(\|x_n - x_\star\|)$ converges almost surely.*
2) $\mathbb{E}\left[\sum_{i=1}^{L}\sum_{n=1}^{\infty} \gamma^2(\|\nabla g_i^\gamma\|^2 + \|\nabla f_i\|^2)\right] < \infty$.
3) *For each i, $\bar{x}_{n+1}^i - x_n \to 0$ almost surely.*

This proposition is shown in Appendix A-B. It remains to establish the almost sure APT to prove Th. 1. We just provide here the main arguments of this part of the proof, since it is similar to its analogue in [22].

Let us write

$$x_{n+1} = x_n - \gamma_{n+1}\sum_{i=1}^{L}\Big(\nabla f_i(\bar{x}_{n+1}^{i-1}, \xi_{n+1}^i) + \nabla g_i^{\gamma_{n+1}}(\bar{x}_{n+1}^{i-1} - \gamma_{n+1}\nabla f_i(\bar{x}_{n+1}^{i-1}, \xi_{n+1}^i), \xi_{n+1}^i)\Big)\,,$$

and let us also define the function

$$H_\gamma(x,(s^1,\dots,s^L)) = \sum_{i=1}^{L} \big[\nabla f_i(\bar{x}^{i-1},s^i) + \nabla g_i^\gamma(\bar{x}^{i-1}-\gamma\nabla f_i(\bar{x}^{i-1},s^i),s^i)\big]$$

where we recall the notation $\bar{x}^i = (T_{\gamma,i}(\cdot,s^i)\circ\cdots\circ T_{\gamma,1}(\cdot,s^1))(x)$. By Lem. 5 and Ass. 3, 4 and 5, $\mathbb{E}[\|H_\gamma(x,\xi)\|]<\infty$ and we define:

$$h_\gamma(x) = \mathbb{E}[H_\gamma(x,\xi)]\,.$$

Note that $x_{n+1} = x_n - \gamma_{n+1}H_{\gamma_{n+1}}(x_n,\xi_{n+1})$. Defining the $(\mathscr{F}_n)$ martingale

$$M_n = \sum_{k=1}^{n} x_k - \mathbb{E}_{k-1}[x_k]$$

it is clear that $x_{n+1} = x_n - \gamma_{n+1}h_{\gamma_{n+1}}(x_n) + (M_{n+1}-M_n)$. Let us rewrite this equation in a form involving the continuous process $\mathsf{x} = \mathsf{I}((x_n))$. Defining $\mathsf{M} = \mathsf{I}((M_n))$, and writing

$$r(t) = \max\{k\geq 0 \,:\, \tau_k \leq t\},\quad t\geq 0,$$

we obtain

$$\mathsf{x}(\tau_n+t) - \mathsf{x}(\tau_n) = -\int_0^t h_{\gamma_{r(\tau_n+u)+1}}(x_{r(\tau_n+u)})\,du + \mathsf{M}(\tau_n+t) - \mathsf{M}(\tau_n)\,. \tag{27}$$

The first argument of the proof of the almost sure APT is a compactness argument on the sequence of continuous processes $(\mathsf{x}(\tau_n+\cdot))_n)$. Specifically, we show that on a $\mathbb{P}$-probability one set, this sequence is equicontinuous and bounded. By Ascoli's theorem, this sequence admits accumulation points in the topology of the uniform convergence on the compacts of $\mathbb{R}_+$. As a second step, we show that these accumulation points are solutions to the differential inclusion (22), which is in fact a reformulation of the almost sure APT property (23).

Since

$$\begin{aligned}
&\mathbb{E}\left[\|x_{n+1}-\mathbb{E}_n x_{n+1}\|^2\right]\\
&= \gamma^2\mathbb{E}\left[\Big\|\sum_{i=1}^{L}(\nabla f_i - \mathbb{E}_n\nabla f_i) + \sum_{i=1}^{L}(\nabla g_i^\gamma - \mathbb{E}_n\nabla g_i^\gamma)\Big\|^2\right]\\
&\leq C\gamma^2\mathbb{E}\left[\sum_{i=1}^{L}(\|\nabla f_i\|^2 + \|\nabla g_i^\gamma\|^2)\right],
\end{aligned}$$

we obtain by Prop. 6–2) that $\sup_n \mathbb{E}[\|M_n\|^2]<\infty$. Thus, the martingale M_n converges almost surely, which implies that the sequence $(\mathsf{M}(\tau_n+\cdot)-\mathsf{M}(\tau_n))_n$ converges almost surely to zero, uniformly on $\mathbb{R}_+$.

By Ass. 3 and 4, $\sup_{x\in\mathcal{K}}\int\|\nabla f_i(x,s)\|^2\mu_i(ds)<\infty$ for each compact $\mathcal{K}\subset\mathsf{X}$ and each i. By Ass. 5, we also have

$$\sup_{x\in\mathcal{K}}\int\|\nabla g_i^\gamma(x,s)\|^{1+\varepsilon}\mu_i(ds) \leq \sup_{x\in\mathcal{K}}\int\|\partial g_i^0(x,s)\|^{1+\varepsilon}\mu_i(ds) < \infty\,.$$

Thus by Lem. 5 and Hölder inequality, and using the fact that the sequence (x_n) is almost surely bounded by Prop. 6–1), it can be shown that

$$\sup_n \|h_{\gamma_{n+1}}(x_n)\|<\infty \quad \text{w.p. } 1\,,$$

Inspecting (27), we thus obtain that the sequence $(\mathsf{x}(\tau_n+\cdot))_n$ is equicontinuous and bounded with probability one.

In order to characterize its cluster points, choose $T>0$, and consider an elementary event on the probability one set where x is equicontinuous and bounded on $[0,T]$. With a small notational abuse, let (n) be a subsequence along which $(\mathsf{x}(\tau_n+\cdot))_n$ converges on $[0,T]$ to some continuous function $\mathsf{z}(t)$. This function then is written as

$$\mathsf{z}(t)-\mathsf{z}(0) = -\lim_{n\to\infty}\int_0^t du\int_\Xi \mu(ds)\,H_{\gamma_{r(\tau_n+u)+1}}(x_{r(\tau_n+u)},s).$$

By the boundedness of (x_n) (Prop. 6-1)), Lem. 5, and Ass. 3, 4 and 5, the sequence of functions $(H_{\gamma_{r(\tau_n+u)+1}}(x_{r(\tau_n+u)},s))_n$ in the parameters (u,s) is bounded in the Banach space $\mathcal{L}^{1+\varepsilon}(du\otimes\mu)$, for some $\varepsilon>0$, where du is the Lebesgue measure on $[0,T]$. Since the unit ball of $\mathcal{L}^{1+\varepsilon}(du\otimes\mu)$ is weakly compact in this space by the Banach-Alaoglu theorem, since this space is reflexive, we can extract a subsequence (still denoted as (n)) such that $H_{\gamma_{r(\tau_n+u)+1}}(x_{r(\tau_n+u)},s)$ converges weakly in $\mathcal{L}^{1+\varepsilon}(du\otimes\mu)$, as $n\to\infty$, to a function $Q(u,s)$. The remainder of the proof consists in showing that Q can be written as

$$Q(u,s) = \sum_{i=1}^{L}\big(b_i(u,s^i)+p_i(u,s^i)\big)\,,$$

where $b_i(u,s^i) = \nabla f_i(\mathsf{z}(u),s^i)$ and $p_i(u,s^i)\in\partial g_i(\mathsf{z}(u),s^i)$ for $du\otimes\mu_i$-almost all (u,s^i). Indeed, once this result is established, it becomes clear that $\mathsf{z}(t)$ is an absolutely continuous function whose derivative satisfies almost everywhere the inclusion $\dot{\mathsf{z}}(t)\in-\sum_i(\nabla F_i(\mathsf{z}(t))+\partial G_i(\mathsf{z}(t)))$, noting that we can exchange the integration and the differentiation (resp. the subdifferentiation) in the expression of ∇F (resp. of ∂G).

We just provide here the main argument of this part of the proof, since it is similar to its analogue in [22]. Let $i\in\{1,\dots,L\}$. Let us focus on the sequence of functions of $(u,s)\in[0,T]\times\Xi$ defined by

$$\nabla g_i^{\gamma_{r(\tau_n+u)+1}}(\bar{x}^{i-1}_{r(\tau_n+u)+1} - \gamma_{r(\tau_n+u)+1}\nabla f_i(\bar{x}^{i-1}_{r(\tau_n+u)+1},s),s)$$

and indexed by n. This sequence is bounded in $\mathcal{L}^{1+\varepsilon}(du\otimes\mu_i)$ on a probability one set, as a function of (u,s), for the same reasons as those explained above for $(H_{\gamma_{r(\tau_n+u)+1}}(x_{r(\tau_n+u)},s))_n$. We need to show that any weak limit point $p_i(u,s)$ of this sequence satisfies $p_i(u,s)\in\partial g_i(\mathsf{z}(u),s)$ for $du\otimes\mu_i$-almost all (u,s). Using the fact that $\mathsf{x}(\tau_n+\cdot)\to\mathsf{z}(\cdot)$ almost surely, along with the inequality $\langle\nabla g_i^\gamma(x,s)-w,\mathrm{prox}_{\gamma g_i(\cdot,s)}(x)-v\rangle\geq 0$, valid for all $x,v\in\mathsf{X}$ and $w\in\partial g_i(v,s)$, we show that $\langle p_i(u,s)-w,\mathsf{z}(u)-v\rangle\geq 0$ for $du\otimes\mu_i$-almost all (u,s). Since $v\in\mathsf{X}$ and $w\in\partial g_i(v,s)$ are arbitrary, we get that $p_i(u,s)\in\partial g_i(\mathsf{z}(u),s)$ by a well known property of the subdifferentials of $\Gamma_0(\mathsf{X})$ functions.

VIII. Conclusion

A fast regularized optimization algorithm over large unstructured graphs was introduced in this paper. This algorithm is a variant of the proximal gradient algorithm that operates on randomly chosen simple paths. It belongs to the family of stochastic approximation algorithms with a decreasing step size.

One future research direction consists in a fine convergence analysis of this algorithm, hopefully leading to a provably optimal choice of the total walk length L. Another research direction concerns the constant step analogue of the described algorithm, whose transient behavior could be interesting in many applicative contexts in the fields of statistics and learning.

APPENDIX A
PROOFS FOR SEC. VII

A. Proof of Lem. 5

We start by writing $\|\nabla f_i(\bar{x}^{i-1}, s^i)\| \le \|\nabla f_i(\bar{x}^{i-2}, s^i)\| + K_i(s^i)\|\bar{x}^{i-1} - \bar{x}^{i-2}\|$, where $K_i(s^i)$ is provided by Ass. 3. Using the identity $\bar{x}^{i-1} = \mathsf{T}_{\gamma,i-1}(\bar{x}^{i-2})$, where $\mathsf{T}_{\gamma,i}$ is given by (26), and recalling that $\nabla g_i^\gamma(\cdot, s^i)$ is γ^{-1}-Lipschitz, we get

$$\begin{aligned}&\|\nabla f_i(\bar{x}^{i-1}, s^i)\| \le \|\nabla f_i(\bar{x}^{i-2}, s^i)\| \\ &+\gamma K_i(s^i)(2\|\nabla f_{i-1}(\bar{x}^{i-2}, s^{i-1})\| + \|\nabla g_{i-1}^\gamma(\bar{x}^{i-2}, s^{i-1})\|).\end{aligned}$$

Similarly,

$$\begin{aligned}&\|\nabla g_i^\gamma(\bar{x}^{i-1} - \gamma\nabla f_i(\bar{x}^{i-1}, s^i), s^i)\| \\ &\le \|\nabla f_i(\bar{x}^{i-1}, s^i)\| + 2\|\nabla f_{i-1}(\bar{x}^{i-2}, s^{i-1})\| \\ &\quad + \|\nabla g_i^\gamma(\bar{x}^{i-2}, s^i)\| + \|\nabla g_{i-1}^\gamma(\bar{x}^{i-2}, s^{i-1})\|.\end{aligned}$$

Iterating down to $\bar{x}^0 = x$, we get the result since for every i, the $K_i(\xi^i)$ admit all their moments.

B. Proof of Prop. 6

Let $x_\star$ be an arbitrary element of $\mathcal{Z}$. Let $i \in \{1, \dots, L\}$. We start by writing

$$\begin{aligned}\|\bar{x}_{n+1}^i - x_\star\|^2 &= \|\bar{x}_{n+1}^i - \bar{x}_{n+1}^{i-1}\|^2 + \|\bar{x}_{n+1}^{i-1} - x_\star\|^2 \\ &\quad + 2\langle \bar{x}_{n+1}^i - \bar{x}_{n+1}^{i-1}, \bar{x}_{n+1}^{i-1} - x_\star\rangle \\ &= \|\bar{x}_{n+1}^{i-1} - x_\star\|^2 + \gamma^2\|\nabla f_i + \nabla g_i^\gamma\|^2 \\ &\quad - 2\gamma\langle \nabla f_i - \nabla f_i^\star, \bar{x}_{n+1}^{i-1} - x_\star\rangle \\ &\quad - 2\gamma\langle \nabla g_i^\gamma - \varphi_i, \bar{x}_{n+1}^{i-1} - x_\star\rangle \\ &\quad - 2\gamma\langle \nabla f_i^\star + \varphi_i, \bar{x}_{n+1}^{i-1} - x_\star\rangle \\ &= \|\bar{x}_{n+1}^{i-1} - x_\star\|^2 + A_1 + A_2 + A_3 + A_4.\end{aligned}$$

Most of the proof consists in bounding the A_i's. We shall repeatedly use Young's inequality $|\langle a, b\rangle| \le \eta\|a\|^2 + C\|b\|^2$, where $\eta > 0$ is a constant chosen as small as desired, and $C > 0$ is fixed accordingly. Starting with A_1, we have

$$A_1 \le \gamma^2(1+\eta)\|\nabla g_i^\gamma\|^2 + C\gamma^2\|\nabla f_i\|^2.$$

We have $A_2 \le 0$ by the convexity of f_L. We can write

$$\begin{aligned}A_3 &= -2\gamma\langle \nabla g_i^\gamma - \varphi_i, \mathrm{prox}_{\gamma g_i} - x_\star\rangle \\ &\quad - 2\gamma\langle \nabla g_i^\gamma - \varphi_i, \bar{x}_{n+1}^{i-1} - \gamma\nabla f_i - \mathrm{prox}_{\gamma g_i}\rangle \\ &\quad - 2\gamma\langle \nabla g_i^\gamma - \varphi_i, \gamma\nabla f_i\rangle\end{aligned}$$

By (25), the first term at the right hand side is ≤ 0. By (24), $\bar{x}_{n+1}^{i-1} - \gamma\nabla f_i - \mathrm{prox}_{\gamma g_i} = \gamma\nabla g_i^\gamma$. Thus,

$$\begin{aligned}A_3 &\le -2\gamma^2\|\nabla g_i^\gamma\|^2 + 2\gamma^2\langle \varphi_i, \nabla g_i^\gamma + \nabla f_i\rangle - 2\gamma^2\langle \nabla g_i^\gamma, \nabla f_i\rangle \\ &\le -(2-\eta)\gamma^2\|\nabla g_i^\gamma\|^2 + C\gamma^2\|\nabla f_i\|^2 + C\gamma^2\|\varphi_i\|^2\end{aligned}$$

As regards A_4, we have

$$\begin{aligned}A_4 &= -2\gamma\langle \nabla f_i^\star + \varphi_i, x_n - x_\star\rangle \\ &\quad - 2\gamma\langle \nabla f_i^\star + \varphi_i, \bar{x}_{n+1}^{i-1} - x_n\rangle.\end{aligned}$$

Gathering these inequalities, we get

$$\begin{aligned}\|\bar{x}_{n+1}^i - x_\star\|^2 &\le \|\bar{x}_{n+1}^{i-1} - x_\star\|^2 - (1-\eta)\gamma^2\|\nabla g_i^\gamma\|^2 \\ &\quad + C\gamma^2\|\nabla f_i\|^2 + C\gamma^2\|\varphi_i\|^2 \\ &\quad - 2\gamma\langle \nabla f_i^\star + \varphi_i, x_n - x_\star\rangle \\ &\quad - 2\gamma\langle \nabla f_i^\star + \varphi_i, \bar{x}_{n+1}^{i-1} - x_n\rangle\end{aligned}$$

Iterating over i, we get

$$\begin{aligned}\|\bar{x}_{n+1}^i - x_\star\|^2 &\le \|x_n - x_\star\|^2 - (1-\eta)\gamma^2\sum_{k=1}^i \|\nabla g_k^\gamma\|^2 \\ &\quad + C\gamma^2\sum_{k=1}^i \|\nabla f_k\|^2 + C\gamma^2\sum_{k=1}^i \|\varphi_k\|^2 \\ &\quad - 2\gamma\sum_{k=1}^i \langle \nabla f_k^\star + \varphi_k, x_n - x_\star\rangle \\ &\quad - 2\gamma\sum_{k=1}^i \langle \nabla f_k^\star + \varphi_k, \bar{x}_{n+1}^{k-1} - x_n\rangle.\end{aligned}$$

The summand in the last term can be written as

$$\begin{aligned}&- 2\gamma\langle \nabla f_k^\star + \varphi_k, \bar{x}_{n+1}^{k-1} - x_n\rangle \\ &= -2\gamma\sum_{\ell=1}^{k-1}\langle \nabla f_k^\star + \varphi_k, \bar{x}_{n+1}^\ell - \bar{x}_{n+1}^{\ell-1}\rangle \\ &= -2\gamma^2\sum_{\ell=1}^{k-1}\langle \nabla f_k^\star + \varphi_k, \nabla f_\ell + \nabla g_\ell^\gamma\rangle \\ &\le \gamma^2 C\|\nabla f_k^\star\|^2 + \gamma^2 C\|\varphi_k\|^2 \\ &\quad + \gamma^2 C\sum_{\ell=1}^{k-1}\|\nabla f_\ell\|^2 + \gamma^2\eta\sum_{\ell=1}^{k-1}\|\nabla g_\ell^\gamma\|^2.\end{aligned}$$

where we used $|\langle a, b\rangle| \le \eta\|a\|^2 + C\|b\|^2$ as above. Therefore, for all $i = 1, \dots, L$,

$$\begin{aligned}\|\bar{x}_{n+1}^i - x_\star\|^2 &\le \|x_n - x_\star\|^2 - (1-\eta)\gamma^2\sum_{k=1}^i \|\nabla g_k^\gamma\|^2 \\ &\quad + C\gamma^2\sum_{k=1}^i \|\nabla f_k^\star\|^2 + C\gamma^2\sum_{k=1}^i \|\varphi_k\|^2 \\ &\quad + C\gamma^2\sum_{k=1}^i \|\nabla f_k\|^2 \\ &\quad - 2\gamma\langle \sum_{k=1}^i \nabla f_k^\star + \varphi_k, x_n - x_\star\rangle. \qquad (28)\end{aligned}$$

We consider the case $i = L$. Using Ass. 4,

$$\begin{aligned}\mathbb{E}_n\left[\|\bar{x}_{n+1}^L - x_\star\|^2\right] \leq \|x_n - x_\star\|^2 \\ - (1-\eta)\gamma^2\mathbb{E}_n\left[\sum_{k=1}^L \|\nabla g_k^\gamma\|^2\right] \\ + C\gamma^2 + C\gamma^2\sum_{k=1}^L \mathbb{E}_n[\|\nabla f_k\|^2] \\ - 2\gamma\mathbb{E}_n\left[\langle\sum_{k=1}^L \nabla f_k^\star + \varphi_k, x_n - x_\star\rangle\right].\end{aligned}$$

The last term at the right hand side is zero since

$$\begin{aligned}&\mathbb{E}_n\left[\langle\sum_{k=1}^L \nabla f_k^\star + \varphi_k, x_n - x_\star\rangle\right] \\ =&\langle\mathbb{E}\left[\sum_{k=1}^L \nabla f_k^\star + \varphi_k\right], x_n - x_\star\rangle = 0\end{aligned}$$

by definition of $\nabla f_k^\star$ and φ_k. Besides, using Ass. 3, for all k we have

$$\mathbb{E}_n[\|\nabla f_k\|^2] \leq C\mathbb{E}_n[\|\nabla f_k^\star\|^2] + C\mathbb{E}_n[K_k^2(\xi_{n+1}^k)\|\bar{x}_{n+1}^{k-1} - x_\star\|^2].$$

Then,

$$\begin{aligned}\mathbb{E}_n[\|x_{n+1} - x_\star\|^2] \leq \|x_n - x_\star\|^2 + C\gamma^2 \\ - (1-\eta)\gamma^2\mathbb{E}_n\left[\sum_{k=1}^L \|\nabla g_k^\gamma\|^2\right] \\ + C\gamma^2\sum_{k=1}^L \mathbb{E}_n\left[K_k^2(\xi_{n+1}^k)\|\bar{x}_{n+1}^{k-1} - x_\star\|^2\right]\end{aligned} \tag{29}$$

We shall prove by induction that for all r.v P_k which is a monomial expression of the r.v $K_k^2(\xi_{n+1}^k), \ldots, K_L^2(\xi_{n+1}^L)$, there exists $C > 0$ such that

$$\mathbb{E}_n\left[P_k\|\bar{x}_{n+1}^{k-1} - x_\star\|^2\right] \leq C(1 + \|x_n - x_\star\|^2), \tag{30}$$

for all $k = 1, \ldots, L$. Note that such a r.v P_k is independent of $\mathscr{F}_n$, non-negative and for all $\alpha > 0$, $\mathbb{E}[P_k^\alpha] < \infty$ by Ass. 3. Using Ass. 3, the induction hypothesis 30 is satisfied if $k = 1$. Assume that it holds true until the step $k-1$ for some $k \leq L$. Using 28 and Ass. 3,

$$\begin{aligned}\mathbb{E}_n\left[P_k\|\bar{x}_{n+1}^{k-1} - x_\star\|^2\right] \leq C\|x_n - x_\star\|^2 \\ + C\gamma^2\mathbb{E}_n\left[P_k\sum_{\ell=1}^{k-1}\|\nabla f_\ell\|^2\right] \\ + C\gamma^2\mathbb{E}_n\left[P_k\sum_{\ell=1}^{k-1}\|\varphi_\ell\|^2 + \|\nabla f_\ell^\star\|^2\right] \\ - 2\gamma\mathbb{E}_n P_k\langle\sum_{\ell=1}^{k-1}\nabla f_\ell^\star + \varphi_\ell, x_n - x_\star\rangle.\end{aligned} \tag{31}$$

The last term at the right hand side can be bounded as

$$\begin{aligned}&- 2\gamma\mathbb{E}_n P_k\langle\sum_{\ell=1}^{k-1}\nabla f_\ell^\star + \varphi_\ell, x_n - x_\star\rangle \\ \leq& C\|x_n - x_\star\|^2 + C\mathbb{E}_n\left[P_k\sum_{\ell=1}^{k-1}\|\nabla f_\ell^\star\|^2 + \|\varphi_\ell\|^2\right] \\ \leq& C\|x_n - x_\star\|^2 + C\end{aligned} \tag{32}$$

using Hölder inequality and Ass. 4. For all $\ell = 1, \ldots, k-1$,

$$\begin{aligned}\mathbb{E}_n[P_k\|\nabla f_\ell\|^2] \leq C\mathbb{E}_n[P_k\|\nabla f_\ell^\star\|^2] \\ + C\mathbb{E}_n\left[P_k K_\ell^2(\xi_{n+1}^\ell)\|\bar{x}_{n+1}^{\ell-1} - x_\star\|^2\right] \\ \leq C(1 + \|x_n - x_\star\|^2)\end{aligned} \tag{33}$$

where we used Hölder inequality and Ass. 4 for the first term at the right hand side and the induction hypothesis (30) at the step ℓ with the r.v $P_\ell := P_k K_\ell^2(\xi_{n+1}^\ell)$ for the second term.

Plugging (32) and (33) into (31) and using again Hölder inequality and Ass. 4 we find that (30) holds true at the step k. Hence (30) holds true for all $k = 1, \ldots, L$. Finally, plugging (30) into (29) with $P_k = K_k^2(\xi_{n+1}^k)$ for all $k = 1, \ldots, L$ we get

$$\begin{aligned}\mathbb{E}_n[\|x_{n+1} - x_\star\|^2] \leq (1 + C\gamma^2)\|x_n - x_\star\|^2 + C\gamma^2 \\ - (1-\eta)\gamma^2\mathbb{E}_n\left[\sum_{k=1}^L \|\nabla g_k^\gamma\|^2\right].\end{aligned}$$

By the Robbins-Siegmund lemma [43], used along with $(\gamma_n) \in \ell^2$, we get that $(\|x_n - x_\star\|)$ converges almost surely, showing the first point.

By taking the expectations at both sides of this inequality, we also obtain that $(\mathbb{E}\|x_n - x_\star\|^2)$ converges, $\sup_n \mathbb{E}\|x_n - x_\star\|^2 < \infty$, and $\mathbb{E}\sum_n \gamma_{n+1}^2 \sum_{i=1}^L \|\nabla g_i^\gamma\|^2 < \infty$. As $\sup_n \mathbb{E}\|x_n - x_\star\|^2 < \infty$, we have by Ass. 3 that $\sup_n \mathbb{E}\|\nabla f_1\|^2 < \infty$. Using Lem. 5 and iterating, we easily get that $\mathbb{E}\sum_n \gamma_{n+1}^2 \sum_{i=1}^L \|\nabla f_i\|^2 < \infty$ for all i.

Since $\|\bar{x}_{n+1}^1 - x_n\| \leq \gamma\|\nabla f_1\| + \gamma\|\nabla g_1^\gamma\|$, we get that $\sum_n \mathbb{E}\|\bar{x}_{n+1}^1 - x_n\|^2 < \infty$. By Borel-Cantelli's lemma, we get that $\bar{x}_{n+1}^1 - x_n \to 0$ almost surely. The almost sure convergence of $\bar{x}_{n+1}^i - x_n$ to zero is shown similarly, and the proof of Prop. 6 is concluded.

REFERENCES

[1] A. El Alaoui, X. Cheng, A. Ramdas, M. J. Wainwright, and M. I. Jordan, "Asymptotic behavior of ℓ_p-based Laplacian regularization in semi-supervised learning," in *COLT*, 2016, pp. 879–906.

[2] X. Zhu, Z. Ghahramani, and J. Lafferty, "Semi-supervised learning using gaussian fields and harmonic functions," in *ICML*, 2003, pp. 912–919.

[3] D. Hallac, J. Leskovec, and S. Boyd, "Network lasso: Clustering and optimization in large graphs," in *SIGKDD*, 2015, pp. 387–396.

[4] A. Chambolle, V. Caselles, D. Cremers, M. Novaga, and T. Pock, "An introduction to total variation for image analysis," *Theoretical foundations and numerical methods for sparse recovery*, vol. 9, pp. 263–340, 2010.

[5] W. Hinterberger, M. Hintermüller, K. Kunisch, M. Von Oehsen, and O. Scherzer, "Tube methods for bv regularization," *Journal of Mathematical Imaging and Vision*, vol. 19, no. 3, pp. 219–235, 2003.

[6] Z. Harchaoui and C. Lévy-Leduc, "Multiple change-point estimation with a total variation penalty," *Journal of the American Statistical Association*, 2012.

[7] R. J. Tibshirani, "Adaptive piecewise polynomial estimation via trend filtering," *The Annals of Statistics*, vol. 42, no. 1, pp. 285–323, 2014.

[8] Y.-X. Wang, J. Sharpnack, A. Smola, and R. J. Tibshirani, "Trend filtering on graphs," *Journal of Machine Learning Research*, vol. 17, no. 105, pp. 1–41, 2016.

[9] O. H. M. Padilla, J. G. Scott, J. Sharpnack, and R. J. Tibshirani, "The dfs fused lasso: nearly optimal linear-time denoising over graphs and trees," *arXiv preprint arXiv:1608.03384*, 2016.

[10] J.-C. Hütter and P. Rigollet, "Optimal rates for total variation denoising," *arXiv preprint arXiv:1603.09388*, 2016.

[11] L. Landrieu and G. Obozinski, "Cut pursuit: Fast algorithms to learn piecewise constant functions," in *AISTATS*, 2016, pp. 1384–1393.

[12] W. Tansey and J. G. Scott, "A fast and flexible algorithm for the graph-fused lasso," *arXiv preprint arXiv:1505.06475*, 2015.

[13] A. Barbero and S. Sra, "Modular proximal optimization for multidimensional total-variation regularization," *arXiv preprint arXiv:1411.0589*, 2014.

[14] W. Ben-Ameur, P. Bianchi, and J. Jakubowicz, "Robust distributed consensus using total variation," *IEEE Transactions on Automatic Control*, vol. 61, no. 6, pp. 1550–1564, 2016.

[15] S. Chen, A. Sandryhaila, G. Lederman, Z. Wang, J. M. Moura, P. Rizzo, J. Bielak, J. H. Garrett, and J. Kovačević, "Signal inpainting on graphs via total variation minimization," in *ICASSP*, 2014, pp. 8267–8271.

[16] L. Condat, "A direct algorithm for 1d total variation denoising," *IEEE SPL*, vol. 20, no. 11, pp. 1054–1057, 2013.

[17] F. R. Chung, *Spectral graph theory*. American Mathematical Soc., 1997, vol. 92.

[18] D. A. Spielman, "Algorithms, graph theory, and linear equations in laplacian matrices," in *Proceedings of the ICM*, vol. 4, 2010, pp. 2698–2722.

[19] A. Salim, P. Bianchi, W. Hachem, and J. Jakubowicz, "A stochastic proximal point algorithm for total variation regularization over large scale graphs," *IEEE CDC*, 2016.

[20] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *COMPSTAT'2010*, 2010, pp. 177–186.

[21] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *arXiv preprint arXiv:1606.04838*, 2016.

[22] P. Bianchi and W. Hachem, "Dynamical behavior of a stochastic forward–backward algorithm using random monotone operators," *Journal of Optimization Theory and Applications*, vol. 171, no. 1, pp. 90–120, 2016.

[23] N. A. Johnson, "A dynamic programming algorithm for the fused lasso and l 0-segmentation," *Journal of Computational and Graphical Statistics*, vol. 22, no. 2, pp. 246–260, 2013.

[24] E. Mammen and S. van de Geer, "Locally adaptive regression splines," *The Annals of Statistics*, vol. 25, no. 1, pp. 387–413, 1997.

[25] P. L. Davies and A. Kovac, "Local extremes, runs, strings and multiresolution," *The Annals of Statistics*, pp. 1–48, 2001.

[26] P. L. Combettes, "Iterative construction of the resolvent of a sum of maximal monotone operators," *Journal of Convex Analysis*, vol. 16, no. 4, pp. 727–748, 2009.

[27] S. Jegelka, F. Bach, and S. Sra, "Reflection methods for user-friendly submodular optimization," in *Advances in NIPS*, 2013, pp. 1313–1321.

[28] D. A. Spielman and S.-H. Teng, "Nearly linear time algorithms for preconditioning and solving symmetric, diagonally dominant linear systems," *SIAM Journal on Matrix Analysis and Applications*, vol. 35, no. 3, pp. 835–885, 2014.

[29] R. T. Rockafellar, "Measurable dependence of convex sets and functions on parameters," *Journal of Mathematical Analysis and Applications*, vol. 28, no. 1, pp. 4–25, 1969.

[30] G. B. Passty, "Ergodic convergence to a zero of the sum of monotone operators in hilbert space," *Journal of Mathematical Analysis and Applications*, vol. 72, no. 2, pp. 383–390, 1979.

[31] P. Bianchi, "Ergodic convergence of a stochastic proximal point algorithm," *SIAM Journal on Optimization*, vol. 26, no. 4, pp. 2235–2260, 2016.

[32] M. Wang and D. P. Bertsekas, "Incremental constraint projection methods for variational inequalities," *Mathematical Programming*, vol. 150, no. 2, pp. 321–363, 2015.

[33] R. T. Rockafellar and R. J. Wets, "On the interchange of subdifferentiation and conditional expectation for convex functionals," *Stochastics: An International Journal of Probability and Stochastic Processes*, vol. 7, no. 3, pp. 173–182, 1982.

[34] J. Leskovec and A. Krevl, "SNAP Datasets: Stanford large network dataset collection," http://snap.stanford.edu/data, Jun. 2014.

[35] P. W. Holland, K. B. Laskey, and S. Leinhardt, "Stochastic blockmodels: First steps," *Social networks*, vol. 5, no. 2, pp. 109–137, 1983.

[36] R. H. Byrd, P. Lu, J. Nocedal, and C. Y. Zhu, "A limited memory algorithm for bound constrained optimization," *SIAM Journal on Scientific Computing*, vol. 16, no. 5, pp. 1190–1208, 1995.

[37] H. H. Bauschke and P. L. Combettes, *Convex analysis and monotone operator theory in Hilbert spaces*, ser. CMS Books in Mathematics/Ouvrages de Mathématiques de la SMC. New York: Springer, 2011. [Online]. Available: http://dx.doi.org/10.1007/978-1-4419-9467-7

[38] E. Moulines and F. R. Bach, "Non-asymptotic analysis of stochastic approximation algorithms for machine learning," in *Advances in NIPS*, 2011, pp. 451–459.

[39] N. K. Vishnoi, "Laplacian solvers and their algorithmic applications," *Theoretical Computer Science*, vol. 8, no. 1-2, pp. 1–141, 2012.

[40] H. Brézis, *Opérateurs maximaux monotones et semi-groupes de contractions dans les espaces de Hilbert*, ser. North-Holland mathematics studies. Burlington, MA: Elsevier Science, 1973. [Online]. Available: http://cds.cern.ch/record/1663074

[41] J.-P. Aubin and A. Cellina, *Differential inclusions*, ser. Grundlehren der Mathematischen Wissenschaften [Fundamental Principles of Mathematical Sciences]. Springer-Verlag, Berlin, 1984, vol. 264, set-valued maps and viability theory. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-69512-4

[42] M. Benaïm and M. W. Hirsch, "Asymptotic pseudotrajectories and chain recurrent flows, with applications," *Journal of Dynamics and Differential Equations*, vol. 8, no. 1, pp. 141–176, 1996.

[43] H. Robbins and D. Siegmund, "A convergence theorem for non negative almost supermartingales and some applications," in *Optimizing Methods in Statistics*. Academic Press, New York, 1971, pp. 233–257.

Adil Salim was born in 1991 in L'Hay-les-Roses, France. He received the M.Sc. degree of the University of Paris XI and the ENSAE ParisTech in 2015. Then he joined Telecom ParisTech as a Ph.D. student in the Signal, Statistics, Learning group. His research interests are focused on optimization algorithms for machine learning.

Pascal Bianchi was born in 1977 in Nancy, France. He received the M.Sc. degree of the University of Paris XI and Supélec in 2000 and the Ph.D. degree of the University of Marne-la-Vallée in 2003. From 2003 to 2009, he was with the Telecommunication Department of Centrale-Supélec. He is now working as a full Professor in the Signal, Statistics, Learning group at Telecom ParisTech. His current research interests are in the area of numerical optimization, stochastic approximations, signal processing and distributed systems.

Walid Hachem was born in Bhamdoun, Lebanon, in 1967. He received the Engineering degree in telecommunications from St Joseph University (ESIB), Beirut, Lebanon, in 1989, the Masters degree from Telecom ParisTech, France, in 1990, the PhD degree in signal processing from the Université Paris-Est Marne-la-Vallée in 2000 and the Habilitation à diriger des recherches from the Université Paris-Sud in 2006. Between 1990 and 2000 he worked in the telecommunications industry as a signal processing engineer. In 2001 he joined the academia as a faculty member at Supélec, France. In 2006, he joined the CNRS (Centre national de la Recherche Scientifique), where he is now a research director based at the Université Paris-Est. His research themes consist mainly in the large random matrix theory and its applications in statistical estimation and in communication theory, and in the optimization algorithms in random environments.
He served as an associate editor for the IEEE Transactions on Signal Processing between 2007 and 2010.