A Stochastic Proximal Point Algorithm for Total Variation Regularization over Large Scale Graphs

Adil Salim, Pascal Bianchi, Walid Hachem and Jérémie Jakubowicz

***Abstract*— The total-variation (TV) regularizer is often used to promote the structured sparsity of a given real function over the vertices of a non-directed graph. Indeed, the proximity operator associated with TV regularizer promotes sparsity of the function discrete gradient. Although quite affordable in the special case of one-dimensional (1D) graphs, the computation of the proximity operator for general large scale graphs can be demanding. In this paper, we propose a stochastic algorithm for solving this problem over large graphs with a moderate iteration complexity. The algorithm consists in properly selecting random paths in the graph and computing 1D-proximity operators over these paths. Convergence of the algorithm is related to recent results on stochastic proximal point algorithms.**

I. Introduction

In applications such as trend filtering [1], [2], [3] or image restoration and signal processing [4], [5], [6], a usual task is to solve on some Euclidean space $\mathcal{X}$ an optimization problem of the form $\min_{x\in\mathcal{X}} L(x)+R(x)$ where $L(x)$ is a data fitting term and $R(x)$ is a convex proper lower semicontinuous regularization term which promotes the structured sparsity of the solution. Many solvers are based on the computation of the so-called proximity operator of R defined by

$$\mathrm{prox}_R(y) = \arg\min_{x\in\mathcal{X}} \frac{1}{2}\|x-y\|^2 + R(x)$$

for every $y \in \mathcal{X}$, where $\|\,.\,\|$ stands for the standard Euclidean norm. Hence, the performance of the solvers is intrinsically related to the efficiency of the computation of the proximity operator.

In this paper, we consider an undirected graph $G=(V,E)$ with no self loops, where V is a finite set of nodes and E is the set of edges. The (anisotropic) *total variation* (TV) of a function $x \in \mathbb{R}^V$ on the graph G is defined by

$$\mathrm{TV}(x,G) = \sum_{\{i,j\}\in E} |x(i)-x(j)|\,.$$

The total variation is a seminorm on $\mathbb{R}^V$ which is often used as a regularizer in the programming problems over a graph where the sparsity of the discrete gradient of the solution is promoted. The aim of this paper is to compute the TV-proximity operator $\mathrm{prox}_{\lambda \mathrm{TV}(\,.\,,G)}(y)$ at a given point $y \in \mathbb{R}^V$, where $\lambda > 0$ is an arbitrary scaling parameter. Otherwise stated, the aim is to solve

$$\min_{x\in\mathbb{R}^V} \frac{1}{2}\|x-y\|^2 + \lambda \mathrm{TV}(x,G). \tag{1}$$

We refer the reader to [7] for an overview of efficient iterative algorithms which can be used for that sake. Although the programming problem can be difficult to solve for general graphs, fast methods exist for structured graphs. When G is a 1D-graph *i.e.*, a simple path with no repeated node, efficient algorithms exist. Recently, Condat [8] revisited an algorithm that is due to Mammen and Van De Geer [9] referred to as the *taut-string* algorithm, and that has complexity $O(n^2)$ in the worst-case scenario, and $O(n)$ in the most realistic cases, where n denotes the number of nodes. In [9] the taut-string is linked to a total variation regularization problem. The taut-string algorithm is also supported by a rich literature and can be traced back to several decades ago, ranging from problems of *isotonic regression* [10], to *unimodality tests* [11], and to non-parametric statistics [9], [12] (see also [5], [13], [14]). This algorithm is generalized to 2D-grids, weighted TV norms and L^p TV norms by A. Barbero and S. Sra in [15]. To generalize to 2D-grids, they noted that the TV regularization can be written as a sum of two terms on which one can apply 1D methods, according to [16] and [17]. Over general graphs, there is no immediate way to generalize the taut string method. The problem of computing the TV-proximity operator is addressed in [1], [2], [3] where a primal-dual interior point method is proposed, and in [18] and where a path algorithm is proposed. We also mention [19] which proposes to solve the problem distributively over the nodes using the Alternating Direction Method of Multipliers (ADMM).

The iteration complexity of the above methods can be a bottleneck as far as very large graphs are concerned. In this paper, we provide a stochastic approach to solve (1), where the iteration complexity is controlled by the user. Our algorithm combines two ingredients. The first one is a stochastic version of the celebrated proximal point algorithm which is grounded in the recent works [20], [21], [22]. The second one is the taut-string optimization method mentioned above, which is known to be particularly efficient to solve the TV-proximity problem in the special case of a 1D-graph. Specifically, our algorithm consists in properly selecting random paths in the graph and efficiently computing 1D-proximity operators over these paths.

The paper is organized as follows. First, we provide a brief description of the proximal stochastic methods in

The first three authors are with LTCI, CNRS, Telecom ParisTech, Universit Paris-Saclay, 75013 Paris, France. The fourth author is with Telecom Sud Paris, RST, 91011 Evry, France. E-mails: adil.salim, pascal.bianchi, walid.hachem@telecom-paristech.fr, jeremie.jakubowicz@telecom-sudparis.eu

This research was partially supported by Labex DigiCosme (project Idex Paris-Saclay ANR-11-IDEX- 0003-02 operated by ANR as part of the program "Investissement d'Avenir").

Section II. Then, in Section III, we describe the proposed algorithm. Finally, Section IV is dedicated to some numerical experiments.

II. Background on Stochastic Approximation

A. The Stochastic Proximal Point Algorithm

In this section, we briefly recall the stochastic proximal point algorithm and the related convergence results. The aim is to minimize some integral functional

$$\min_{x\in\mathcal{X}} \mathbb{E}(f(x,\xi)) \tag{2}$$

where ξ is a random variable (r.v.) on some probability space $(\Omega,\mathcal{F},\mathbb{P})$ with the expectation $\mathbb{E}$, the mapping $f(\,.\,,s)$ is, for almost every s, a proper lower-semicontinuous (l.s.c.) convex function, and $\mathcal{X}$ is an Euclidean space. We are interested in the case where the above integral is difficult to compute. Stochastic approximation methods, such as the celebrated stochastic (sub)gradient algorithm, consist in iteratively updating an estimate x_n of a minimizer using (typically independent) copies $(\xi_n)_{n\in\mathbb{N}^*}$ of the r.v. ξ. Regarding the rich literature on such methods, we refer for instance to [23].

The stochastic proximal point algorithm is written

$$x_{n+1} = \mathrm{prox}_{\gamma_{n+1} f(\,.\,,\xi_{n+1})}(x_n)$$

where $(\gamma_n)_{n\in\mathbb{N}^*}$ is a sequence of positive step sizes. An early version of the algorithm was proposed in [24] in the case where the integral functional in (2) is reduced to a to a sum (which amounts to choosing ξ as a simple uniformly distributed r.v.). In addition, the prox operators are chosen cyclically in [24] instead of randomly. The random case was studied in [20], and a detailed convergence analysis of the method can be found in [21], [22] under more general hypotheses. The assumptions below will be enough for the present paper:

Theorem 2.1 ([22]): Assume the following:

- The function f is a normal integrand [25] on $\mathcal{X}\times\Omega$ such that $f(\cdot,s)$ is a l.s.c. convex function with domain $\mathcal{X}$ for $\mathbb{P}$-almost every s.
- $\forall x\in\mathcal{X},\ \int_\Omega |f(x,s)|\,\mathbb{P}(ds) < \infty$.
- For every compact set $K\subset\mathcal{X}$,

$$\sup_{x\in K}\int \|\partial f_0(x,s)\|^2\,\mathbb{P}(ds) \ < \ \infty,$$

  where $\partial f_0(\cdot,s)$ is the least-norm element of the subdifferential of $f(\cdot,s)$.
- The set $\mathcal{Z}$ of minimizers of (2) is nonempty. For any $x\in\mathcal{Z}$, there exists a measurable function $M_x:\Omega\to\mathbb{R}$ such that $\sup\{\|z\| \ : \ z\in\partial f(x,s)\}\le M_x(s)$, and $\int_\Omega M_x(s)^2\,\mathbb{P}(ds)<\infty$.
- $(\gamma_n)\in\ell^2\setminus\ell^1$.

Then, there exists a random variable $x_\star$ supported by $\mathcal{Z}$ such that x_n converges almost surely to $x_\star$.

The assumption $\sum_n \gamma_n^2<\infty$ implies that the sequence γ_n tends to zero, as often assumed in the context of stochastic optimization methods. The assumption is needed to ensure the almost sure convergence of the algorithm. The case of a (non vanishing) step size is as well important in practice, although not as well explored from a theoretical point of view. In this case, it is expected that, in its stationary regime, the algorithm ends up in the vicinity of the set of minimizers.

B. A Stochastic version of Passty's Algorithm

In this paper, we will use the following variant of the stochastic proximal point algorithm, which we shall refer to as the *stochastic Passty's algorithm*. Consider the case where the mapping f is written as the following sum

$$f(x,s)=\sum_{i=1}^{L} f_i(x,s)$$

where $L\ge 1$ is an integer and where for every $i=1,\dots,L$ and $\mathbb{P}$-almost every s, the mapping $f_i(\,.\,,s)$ is proper convex l.s.c. The stochastic Passty's algorithm consists in the iterations

$$x_{n+1} = T_{\gamma_{n+1}}(x_n,\xi_{n+1})$$

where for every $\gamma>0$ and every s, we set

$$T_\gamma(\,.\,,s) = \mathrm{prox}_{\gamma f_L(\,.\,,s)}\circ\cdots\circ\mathrm{prox}_{\gamma f_1(\,.\,,s)}$$

where $\circ$ stands for composition. In the special case where the functions f_i are constant with respect to s (the algorithm is deterministic), the above iterations boil down to the standard Passty's algorithm [24]. In the case where $L=1$, the methods reduces to the stochastic proximal point algorithm. In the general case, the convergence of the method can be studied using similar tools as in [21], [22]. The problem is addressed in [26].

III. Main Algorithm

A. Notations

Let $L\ge 0$ be an integer. L will control the complexity of an iteration of the algorithm. We refer to a walk of length L over the graph G as a sequence $s=(v_1,v_2,\dots,v_{L+1})$ in V^{L+1} such that for every $i=1,\dots,L$, the pair $\{v_i,v_{i+1}\}$ 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 $\mathcal{V}(s)=\{v_1,\dots,v_{L+1}\}$ and $\mathcal{E}(s)=\{\{v_1,v_2\},\dots,\{v_L,v_{L+1}\}\}$. For every $x\in\mathbb{R}^V$, we denote by $x_s=(x(v_1),\dots,x(v_{L+1}))$ the restriction of x to s, and by $x_{\overline{s}}$ the restriction of x to the complementary set of s in V. We refer to the total variation of x *along the walk* s as

$$\mathrm{TV}(x_s,\mathcal{G}(s)) = \sum_{i=1}^{L} |x(v_{i+1})-x(v_i)|\,.$$

For every walk s and every $u\in\mathbb{R}^{\mathcal{V}(s)}$, we define

$$\Pi_{\mathrm{TV}}(u,s,\rho) = \mathrm{prox}_{\rho\mathrm{TV}(\,.\,,\mathcal{G}(s))}(u)\,.$$

We say that a walk is a *simple path* if there is no repeated node that is, all elements in s are different. When s is a simple path, the quantity $\Pi_{\mathrm{TV}}(u,s,\rho)$ can be efficiently computed using *e.g.* the taut-string method of [8].

Our notations are summarized in Table I

$G=(V,E)$	Graph with no self-loop
$d(v)$	Degree of a vertex $v \in V$
x, y	Functions on $V \to \mathbb{R}$. Function y is fixed.
s	Walk in G
x_s, $x_{\overline{s}}$	Restriction of x to s (resp. $V \backslash s$)
$\mathrm{TV}(x, G)$	Total variation of x on G
$\Pi_{\mathrm{TV}}(\,.\,, s, \rho)$	TV-proximity operator on the walk s

TABLE I
USEFUL NOTATIONS

B. TV as an Expectation over a Random Walk

From now on, $y \in \mathbb{R}^V$ represents a fixed function on $V \to \mathbb{R}$ and we seek to solve the programming problem (1). The key part of this paper is to write the total variation of a function $x \in \mathbb{R}^V$ as an expectation in order to apply a stochastic algorithm as described in Section II.

Let $L \geq 1$ be an integer. We build a simple random walk on the graph G, $\xi = (\boldsymbol{v}_1, \ldots, \boldsymbol{v}_{L+1})$ of length L as follows. The value of the starting node $\boldsymbol{v}_1$ is randomly chosen in V according to the distribution

$$\mathbb{P}(\boldsymbol{v}_1 = v) = \frac{d(v)}{2|E|} \tag{3}$$

for all $v \in V$, where $d(v)$ is the degree of v (the number of its neighbors) and where $|E|$ is the number of edges. Recursively for every $i = 1, \ldots, L$, the next node $\boldsymbol{v}_{i+1}$ is randomly selected according to

$$\mathbb{P}(\boldsymbol{v}_{i+1} = w \,|\, \boldsymbol{v}_i = v) = \frac{1}{d(v)} \tag{4}$$

if w is a neighbor of v and $\mathbb{P}(\boldsymbol{v}_{i+1} = w \,|\, \boldsymbol{v}_i = v) = 0$ otherwise. Of course, ξ has no reason to be a simple path.

Proposition 3.1: For every $x \in \mathbb{R}^V$,

$$\mathrm{TV}(x, G) = \frac{|E|}{L}\, \mathbb{E}(\mathrm{TV}(x_\xi, \mathcal{G}(\xi)))\,.$$

Proof: First, let $(\boldsymbol{v}_i)_i$ the stochastic process whose law is specified by the equations (3) and (4). This process is a Markov chain on V, and it is easy to show that the function $\frac{d}{2|E|}$ describes an invariant probability for this Markov chain. Since the initial distribution of $(\boldsymbol{v}_i)_i$ is $\frac{d}{2|E|}$, the Markov chain $(\boldsymbol{v}_i)_i$ is stationary, and for all i,

$$\boldsymbol{v}_i \sim \frac{d}{2|E|}$$

Then, for all i,

$$\{\boldsymbol{v}_i, \boldsymbol{v}_{i+1}\} \sim \mathrm{Uniform}(E).$$

Actually, for all edge $e = \{v, w\}$,

$$\begin{aligned}
&\mathbb{P}(\{\boldsymbol{v}_i, \boldsymbol{v}_{i+1}\} = e) \\
&\quad = \mathbb{P}(\boldsymbol{v}_i = v, \boldsymbol{v}_{i+1} = w) + \mathbb{P}(\boldsymbol{v}_i = w, \boldsymbol{v}_{i+1} = v) \\
&\quad = \frac{d(v)}{2|E|}\frac{1}{d(v)} + \frac{d(w)}{2|E|}\frac{1}{d(w)} \\
&\quad = \frac{1}{|E|}.
\end{aligned}$$

Finally, for all i,

$$\mathbb{E}\left(|x(\boldsymbol{v}_i) - x(\boldsymbol{v}_{i+1})|\right) = \frac{1}{|E|}\mathrm{TV}(x, G)$$

and

$$\mathbb{E}(\mathrm{TV}(x_\xi, \mathcal{G}(\xi))) \;= \frac{L}{|E|}\mathrm{TV}(x, G)$$

by linearity, since the length of $\mathcal{G}(\xi)$ is L. ∎

By Proposition 3.1, one is able to cast the proximity problem (1) into the form (2) where $\mathcal{X} = \mathbb{R}^V$ and the function f is chosen as

$$f(x, s) = \frac{1}{2}\|x - y\|^2 + \frac{\lambda|E|}{L}\mathrm{TV}(x_s, \mathcal{G}(s))\,. \tag{5}$$

In order to minimize $\mathbb{E}(f(x, \xi))$, it is tempting to apply the stochastic proximal point algorithm. By doing so, the resulting algorithm would consist in the following two steps at each iteration $n+1$:

- Generate a random walk ξ_{n+1} as described above,
- Update the estimate $x_{n+1} = \mathrm{prox}_{\gamma_{n+1} f(\,.\,, \xi_{n+1})}(x_n)\,.$

Of course, the above algorithm is practical only if one is able to easily implement the proximity operator in the second step. The following lemma whose proof is left to the reader sheds some light on that point.

Lemma 3.2: Consider $x \in \mathbb{R}^V$ and a path s in G. Set $\alpha, \beta, \gamma > 0$ and define $p = \mathrm{prox}_{\gamma g}(x)$ where g is the function given by

$$g(x) = \frac{\alpha}{2}\|x - y\|^2 + \beta\mathrm{TV}(x_s, \mathcal{G}(s))\,.$$

Then, separating the components that pertain to nodes on s from the complementary set $\overline{s}$, it holds that $p = (p_s, p_{\overline{s}})$, where

$$\begin{aligned}
p_s &= \Pi_{\mathrm{TV}}\left(\frac{x_s + \gamma\alpha y_s}{1 + \gamma\alpha}, s, \frac{\gamma\beta}{1 + \gamma\alpha}\right), \text{ and} \\
p_{\overline{s}} &= \frac{x_{\overline{s}} + \gamma\alpha y_{\overline{s}}}{1 + \gamma\alpha}\,.
\end{aligned} \tag{6}$$

It follows from the above Lemma that the computation of the stochastic proximity operator $\mathrm{prox}_{\gamma f(\,.\,, \xi)}$ boils down to the evaluation of the TV-proximity operator along the walk ξ. In order to have a fast algorithm, we propose to use the taut-string method alluded to above. However, the taut-string method is easy to implement on *simple* paths, that is, when the walk ξ contains no repeated node. Of course, the random walk ξ has no reason to coincide with a simple path.

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 Proposition 3.1 to loop-erased paths is far from immediate.

As an alternative, we propose to split the simple random walk ξ into several simple paths.

C. Splitting the Walk ξ into Simple Paths

One can always decompose a walk $s = (v_1, v_2, \dots, v_{L+1})$ of length L into a sequence of simple paths. There exists several methods to do so. For convergence speed purposes, these simple paths have to be as long as possible. But, in order to save memory while using the algorithm, the main requirement is that it must be possible to split s online. Thus, after each splitting, we can use and then forget the visited nodes. In other words, thinking s as the realization of a stochastic process, the times of splitting must be stopping times. We propose the following method, although the convergence of the algorithm still holds with another method of splitting.

Formally, one way is to recursively define a sequence of integers $(t_i(s))_{i\in\mathbb{N}}$ as $t_0(s) = 1$ and for all $i \geq 0$,

$$t_{i+1}(s) = \inf\{k \in \{t_i + 1, \dots, L\} \,:\, v_{k+1} \in \{v_{t_i}, \cdots, v_k\}\}$$

if the above set is nonempty, and $t_{i+1} = L+1$ otherwise. We denote by $N(s)$ the smallest integer n such that $t_n = L+1$. It is clear that $1 \leq N(s) \leq L$. For every $i = 1, \dots, N(s)$, we define

$$C_i(s) = (v_{t_{i-1}(s)}, v_{t_{i-1}(s)+1}, \dots, v_{t_i(s)})\,.$$

By construction, $C_i(s)$ is a simple path. We denote by $\ell_i(s)$ the length of $C_i(s)$. We remark that $\sum_{i=1}^{N(s)} \ell_i(s) = L$ and moreover

$$\mathrm{TV}(x_s, \mathcal{G}(s)) = \sum_{i=1}^{N(s)} \mathrm{TV}(x_{C_i(s)}, \mathcal{G}(C_i(s))) \qquad (7)$$

for every $x \in \mathbb{R}^V$.

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 the walk $s = (c, a, e, g, a, f, a, b, h)$ with length $L = 8$. Then, $t_1 = 4$, $t_2 = 6$, $t_3 = t_4 = \cdots = 9$, and s can be decomposed into $N(s) = 3$ simple paths $C_1(s) = (c, a, e, g)$, $C_2(s) = (g, a, f)$ and $C_3(s) = (f, a, b, h)$ with respective lengths $\ell_1(s) = 3$, $\ell_2(s) = 2$ and $\ell_3(s) = 3$.

Let us return to the programming problem (1). From (7), the function f defined in (5) can be decomposed as

$$Lf(x, s) = \sum_{i=1}^{L} f_i(x, s)$$

where we set

$$f_i(x, s) = \frac{\ell_i(s)}{2}\|x - y\|^2 + \lambda|E|\mathrm{TV}(x_{C_i(s)}, \mathcal{G}(C_i(s)))$$

for all $1 \leq i \leq N(s)$ and $f_i(x, s) = 0$ whenever $i > N(s)$. Hence, Problem (1) can be written as

$$\min_{x\in\mathbb{R}^V} \mathbb{E}\left(\sum_{i=1}^{L} f_i(x, \xi)\right)$$

and one can use the stochastic Passty's algorithm described in Section II-B.

procedure PROXTV(y)
  $x \leftarrow x_0$ ▷ Initialization
  $n \leftarrow 1$
  while Convergence is not reached **do**
    Choose a node $\boldsymbol{v}_1$ according to (3).
    Generate a walk $\xi = (\boldsymbol{v}_1 \dots \boldsymbol{v}_L)$ according to (4).
    Split ξ into $N = N(\xi)$ simple paths $C_1, \dots, C_N$
    for $i = 1 \dots N$ **do**
      $\ell \leftarrow \mathrm{length}(C_i)$
      $x \leftarrow \frac{x+\gamma_n \ell y}{1+\gamma_n \ell}$
      $x_{C_i} \leftarrow \Pi_{\mathrm{TV}}\left(x_{C_i}, C_i, \frac{\gamma_n \lambda|E|}{1+\gamma_n \ell}\right)$ ▷ Taut-string (1D)
    end for
    $n \leftarrow n + 1$
  end while
  return x
end procedure

TABLE II
PROPOSED ALGORITHM.

D. Algorithm

The algorithm is summarized in Table II.

Denote by x_n the iterate at time n. Consider a positive step size sequence $(\gamma_n)_{n\in\mathbb{N}^*}$ as introduced in Section II. Let $(\xi_n)_{n\in\mathbb{N}^*}$ be independent copies of the simple random walk ξ whose distribution is given by (3) and (4). Define $N_{n+1} = N(\xi_{n+1})$ and for every $i = 1 \dots N_{n+1}$, $c^i_{n+1} = C_i(\xi_{n+1})$ and $\ell^i_{n+1} = \ell_i(\xi_{n+1})$.

We introduce the variables $\tilde{x}^i_{n+1}$ for $i = 0 \dots N_{n+1}$ recursively defined by $\tilde{x}^0_{n+1} = x_n$ and for every $i \geq 0$,

$$\tilde{x}^{i+1}_{n+1} = \mathrm{prox}_{\gamma_{n+1} f_{i+1}(\,\cdot\,, \xi_{n+1})}(\tilde{x}^i_{n+1})$$

The above iteration can be explicited as follows. Let

$$z^{i+1}_{n+1} = \frac{x^i_{n+1} + \gamma_{n+1}\ell^i_{n+1} y}{1 + \gamma_{n+1}\ell^i_{n+1}}\,.$$

By Lemma 3.2, one has for all $v \notin c^i_{n+1}$

$$\tilde{x}^{i+1}_{n+1}(v) = z^{i+1}_{n+1}(v)$$

whereas the restriction of $\tilde{x}^{i+1}_{n+1}$ to the simple path c^i_{n+1} is given as a the following TV-1D proximity operator:

$$\Pi_{\mathrm{TV}}\left((z^{i+1}_{n+1}(v) : v \in c^{i+1}_{n+1}), c^{i+1}_{n+1}, \rho^{i+1}_{n+1}\right)\,. \qquad (8)$$

where $\rho^{i+1}_{n+1} = \frac{\gamma_{n+1}\lambda|E|}{1+\gamma_{n+1}\ell^{i+1}_{n+1}}$. Finally, we set $x_{n+1} = \tilde{x}^{N_{n+1}}_{n+1}$. In practice, the computation of (8). is achieved by means of an efficient black box. Here we propose to use Condat's algorithm [8] implemented by Sra and Barbero [15], [7].

IV. NUMERICAL EXPERIMENTS

In order to demonstrate the denoising power of the proposed algorithm we start from a "cartoon"-like model over graphs. Namely we generate a graph with well-separated clusters as depicted in Fig. 1. More precisely, we consider N nodes and a C clusters (also called "communities") and we randomly pick a community c_v for each node v uniformly over $\{1, \dots, C\}$. The r.v. c_v are independent. Then we draw independently N^2 Bernoulli r.v. $E(v, w)$, encoding the edges

of the graph (an edge between v and w is present iff $E(v,w)=1$), such that $\mathbb{P}\{E(v,w)=1\}=P(c_v,c_w)$ where

$$\begin{cases} p(c,c') = & .1 \text{ if } c=c' \\ p(c,c') = & .01 \text{ otherwise} \end{cases}$$

This model is called the stochastic block model for the matrix P [27]. It amounts to a blockwise Erdös-Rényi model with parameters depending only on the blocks. Having defined

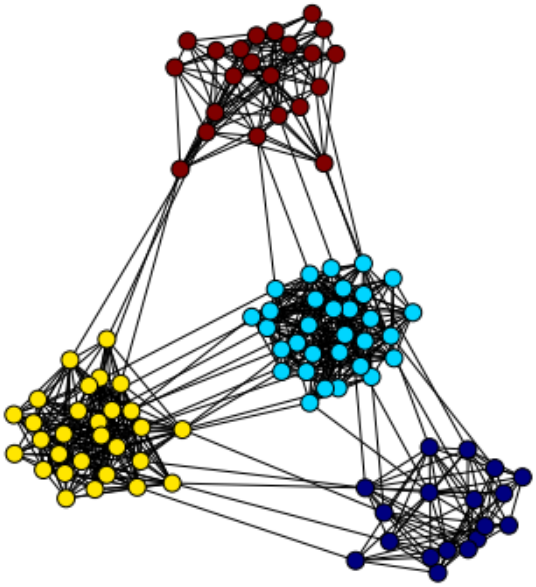

Fig. 1. Realization of a stochastic block model with 4 communities and 100 nodes. Each community is represented with a different color.

the graph as a realization of a stochastic block model, we define the data term y as $y(v)=l(c_v)+\sigma(c_v)\epsilon_v$ where $l(c)$ is a mapping from the communities to the levels (in our experiments $0\leq l\leq 255$), and ϵ_v denotes a standard Gaussian white noise with $\sigma(c)>0$ as its standard deviation. In Figure 2 we represent an example of the signal y along with the sought "signal" $l(c_v)$. On figure 3, our method is compared with two methods: the L-BFGS-B [28] for the dual problem and the well-known projected gradient method for the dual problem. The three methods are used with a constant step size. We sampled a graph according to a stochastic block model with $N=20000$ nodes and $C=10$ communities. It led to about 3.8×10^6 edges. We set the parameter λ at 0.01 in order to ensure that the two terms of the objective function are significant. We sampled the signal y with the same scheme as above, *i.e* $y(v)=l(c_v)+\sigma(c_v)\epsilon_v$. We chose the coordinate of the vector l uniformly in $[-5,5]$ and the coordinate of the vector σ uniformly in $[0,1]$. In Figure 3, the curves with triangles, squares and circles illustrate the performance of L-BFGS-B in the dual space, the projected gradient in the dual space, and our algorithm respectively.

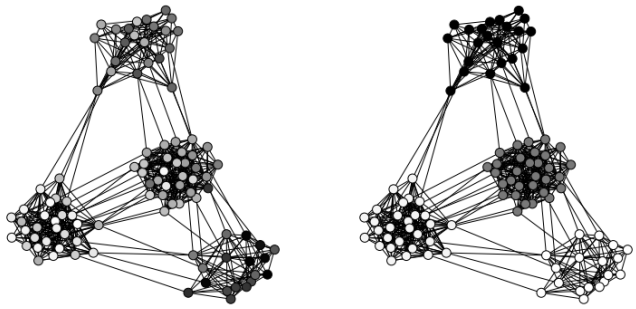

Fig. 2. Left: Signal plus noise function $y=l+\epsilon$. Right: Cartoon-like signal l.

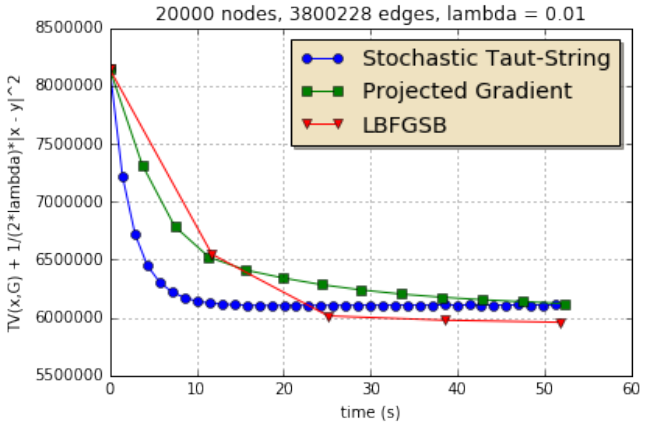

Fig. 3. The objective as a function of the time in seconds

Since we are experimenting these algorithms with constant step size, they reach a long term value in the vicinity of the set of minimizers. This long term value converges to the set of minimizers when the step size converges to zero (this assertion has not been proved yet for stochastic Passty's algorithm with constant step size, but is a usual behaviour). We observe that our method reaches its long term value faster than the other two algorithms. However, the other algorithms long term values are slightly better. One can take advantage of this method when one has a limited time to compute the proximal operator.

In Figure 4 are plotted two histograms. The red one (Initialization) is the histogram of the signal at the initialization. The blue one is the resulting iterate, after applying our algorithm. It exhibits a piecewise constant signal across the nodes, due to the sparsity of the discrete gradient of the signal. The evolution of the TV norm while running the three algorithms is plotted in figure 5.

We observed the above behaviour on several graphs. For example with a bigger graph : $N=10^5$ nodes and about 25×10^6 edges, figure 6 shows similar the results as figure 3 (the legend is the same as in 3).

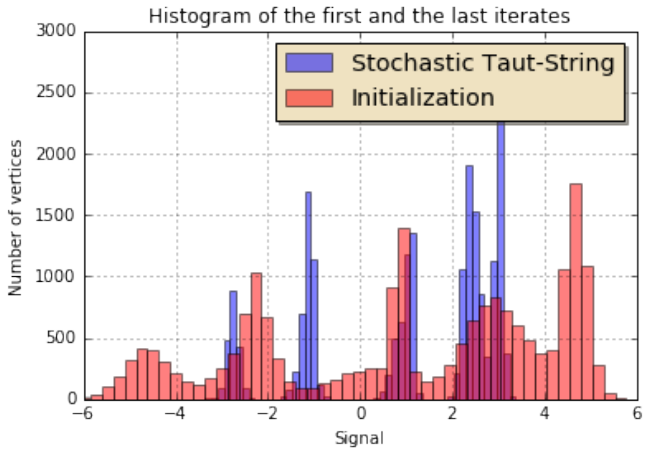

Fig. 4. Signals across the nodes before and after our algorithm

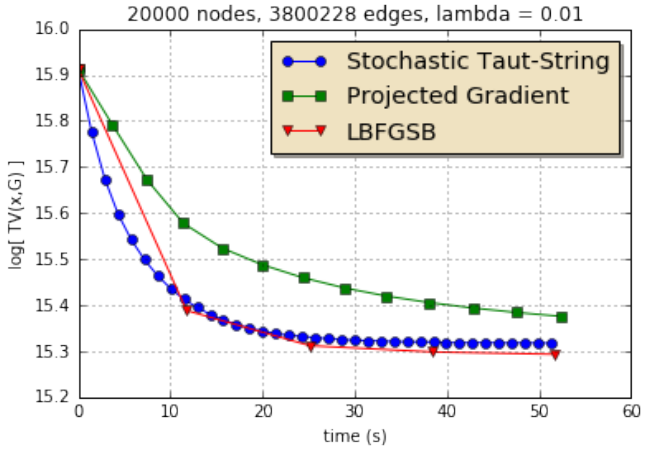

Fig. 5. The logarithm of the TV norm as a function of the time in seconds

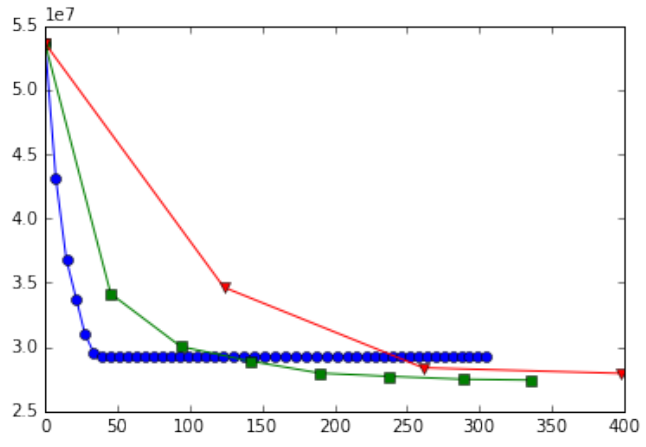

Fig. 6. The objective as a function of the time in seconds

REFERENCES

[1] S-J. Kim, K. Koh, S. Boyd, and D. Gorinevsky, "ℓ_1 trend filtering," *SIAM review*, vol. 51, no. 2, pp. 339–360, 2009.

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

[3] Y-X. Wang, J. Sharpnack, A. Smola, and R. J. Tibshirani, "Trend filtering on graphs," *arXiv preprint arXiv:1410.7690*, 2014.

[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, no. 263-340, pp. 227, 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] A. Barbero and S. Sra, "Fast newton-type methods for total variation regularization.," in *ICML*, 2011, pp. 313–320.

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

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

[10] R. E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. D. Brunk, *Statistical inference under order restrictions: The theory and application of isotonic regression*, Wiley New York, 1972.

[11] J. A. Hartigan and P. M. Hartigan, "The dip test of unimodality," *The Annals of Statistics*, pp. 70–84, 1985.

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

[13] G. Steidl, S. Didas, and J. Neumann, "Relations between higher order tv regularization and support vector regression," in *Scale Space and PDE Methods in Computer Vision*, pp. 515–527. Springer, 2005.

[14] M. Grasmair, "The equivalence of the taut string algorithm and bv-regularization," *Journal of Mathematical Imaging and Vision*, vol. 27, no. 1, pp. 59–66, 2007.

[15] A. Barbero and S. Sra, "Modular proximal optimization for multidimensional total-variation regularization," 2014.

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

[17] S. Jegelka, F. Bach, and S. Sra, "Reflection methods for user-friendly submodular optimization," in *Advances in Neural Information Processing Systems*, 2013, pp. 1313–1321.

[18] R.J. Tibshirani, J. E. Taylor, E.J. Candes, and T. Hastie, *The solution path of the generalized lasso*, Stanford University, 2011.

[19] W. Ben-Ameur, P. Bianchi, and J. Jakubowicz, "Robust distributed consensus using total variation," *IEEE Trans. on Autmatic Control, to appear*, 2015.

[20] M. Wang and D. P Bertsekas, "Incremental constraint projection-proximal methods for nonsmooth convex optimization," Tech. Rep., Massachusetts Institute of Technology, 2013.

[21] P. Bianchi, "Ergodic convergence of a stochastic proximal point algorithm," *ArXiv e-prints, 1504.05400*, Apr. 2015.

[22] P. Bianchi and W. Hachem, "Dynamical behavior of a stochastic forward-backward algorithm using random monotone operators," *arXiv preprint arXiv:1508.02845*, 2015.

[23] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, "Robust stochastic approximation approach to stochastic programming," *SIAM Journal on Optimization*, vol. 19, no. 4, pp. 1574–1609, 2009.

[24] G. B. Passty, "Ergodic convergence to a zero of the sum of monotone operators in Hilbert space," *J. Math. Anal. Appl.*, vol. 72, no. 2, pp. 383–390, 1979.

[25] R. T. Rockafellar, "Measurable dependence of convex sets and functions on parameters," *J. Math. Anal. Appl.*, vol. 28, pp. 4–25, 1969.

[26] A. Salim, P. Bianchi, W. Hachem, and J. Jakubowicz, "A stochastic version of passty's algorithm," Tech. Rep., Work in progress, see http://perso.telecom-paristech.fr/ hachem/, 2016.

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

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