Journal of Machine Learning Research 18 (2018) 1-27 Submitted 11/15; Revised 1/18; Published 4/18

Gradient Estimation with Simultaneous Perturbation and Compressive Sensing

Vivek S. Borkar BORKAR.VS@GMAIL.COM
Department of Electrical Engineering
Indian Institute of Technology Bombay
Mumbai 400076, India

Vikranth R. Dwaracherla VIKRANTHA@STANFORD.EDU
Department of Electrical Engineering
Stanford, USA

Neeraja Sahasrabudhe NEERAJA@IISERMOHALI.AC.IN
Department of Mathematical Sciences
Indian Institute of Science Education and Research, Mohali
SAS Nagar 140306, India

Editor: Sujay Sanghavi

Abstract

We propose a scheme for finding a "good" estimator for the gradient of a function on a high-dimensional space with few function evaluations, for applications where function evaluations are expensive and the function under consideration is not sensitive in all coordinates locally, making its gradient almost sparse. Exploiting the latter aspect, our method combines ideas from Spall's Simultaneous Perturbation Stochastic Approximation with compressive sensing. We theoretically justify its computational advantages and illustrate them empirically by numerical experiments. In particular, applications to estimating gradient outer product matrix as well as standard optimization problems are illustrated via simulations.

Keywords: Gradient estimation; Compressive sensing; Sparsity; Gradient descent; Gradient outer product matrix.

1. Introduction

Estimating the gradient of a given function (with or without noise) is often an important part of problems in reinforcement learning, optimization and manifold learning. In reinforcement learning, policy-gradient methods are used to obtain an unbiased estimator for the gradient. The policy parameters are then updated with increments proportional to the estimated gradient (Sutton et. al, 2000). The objective is to learn a locally optimum policy. REINFORCE and PGPE methods (policy gradients with parameter-based exploration) are popular instances of this approach (Zhao et. al, 2012) for details and comparisons, (Grondman et. al, 2012) for a survey on policy gradient methods in the context of actor-critic algorithms). In manifold learning, various finite difference methods have been explored for gradient estimation (Mukherjee, Wu and Zhou, 2010; Wu et. al, 2010). The idea is to use the estimated gradient to find the lower dimensional manifold where the given function actually lives. Optimization, i.e., finding maximum or minimum of a function, is a ubiquitous problem that appears in many fields wherein one seeks zeroes of the gradient. But the gradient itself might be hard to compute. Gradient estimation techniques prove particularly useful in such scenarios.

A further theoretical justification is facilitated by the results of Austin (2016). In Austin (2016), it was shown that given a connected and locally connected metric probability space (X, d, μ) (i.e., X is a compact metric space with metric d and μ is a probability measure on the Borel σ-algebra of (X, d)), under suitable conditions, any function $f : X^n \mapsto \mathbb{R}$ is close (in $L^1(\mu^n)$) to a function on a lower dimensional factor space obtained by averaging out the remaining arguments w.r.t. the corresponding product of μ (see Austin, 2016, Theorem 1.1). As a special case, a similar fact can be proved for real-values 1-Lipschitz functions on $\mathbb{R} \setminus \mathbb{Z}$ with metric $|\cdot|_\infty^n$ (see Austin, 2016, Theorem 1.2). This suggests that sparse gradients can be expected for functions on high dimensional spaces with adequate regularity conditions.

Over the years gradient estimation has also become an interesting problem in its own right. One would expect that the efficiency of a given method for gradient estimation also depends on the properties of function f. We consider one such class of problems in this paper. Suppose we have a continuously differentiable function $f : \mathbb{R}^n \mapsto \mathbb{R}$ where n is large, such that the gradient ∇f lives mostly in a lower dimensional subspace. This means that one can throw out most of the coordinates of ∇f in a suitable local basis without incurring too much error. In this case, computing $\frac{\partial f}{\partial x_i}$ $\forall i$ is clearly a waste of means. If in addition the function evaluations are expensive, most gradient estimation methods become inefficient. Such is the case, e.g., if a single function evaluation is the output of a large time-consuming simulation. This situation is our specific focus. The problem of expensive function evaluations does not seem to have attracted much attention in machine learning literature, though there has been quite a lot of work on this theme in other communities such as engineering and operations research (Joseph and Murthy, 2017; Xu, Caramanis and Mannor, 2016). Most methods, however, focus on learning a good surrogate for the original function (see Jones, Schonlau and Welch, 1998; Pandita, Bilionis and Panchal, 2016; Shan and Wang, 2010).

To handle the first issue, ideas from compressive sensing can be applied. Compressive sensing theory tells us that an s-sparse vector can be reconstructed from $m \sim s \log(n/s)$ measurements. This means that one does not need the information about ∇f in all n directions, a much smaller number of measurements would suffice. These ideas are frequently used in signal as well as image processing (see Chan et. al, 2008; Duarte et. al, 2008). To remedy the latter difficulty, we use an idea from Simultaneous Perturbation Stochastic Approximation (SPSA) due to Spall (Spall, 1992), viz., the Simultaneous Perturbation (SP).

We begin by explaining the proposed method for gradient estimation. Important ideas and results from compressive sensing and SPSA that are relevant to this work are discussed in Section 2.1 and Section 2.2 respectively. We state the main result in Section 2.3. Section 3 presents applications to manifold learning and optimization with simulated examples.

Some notational preliminaries are as follows. By $\|\cdot\|_1$ and $\|\cdot\|$ we denote the l_1 and l_2 norms in $\mathbb{R}^n$ respectively. By abuse of notation, we also denote the Frobenius norm for matrices over $\mathbb{R}$ by $\|\cdot\|$. Throughout, 'a.s.' stands for 'almost surely', i.e., with probability one.

2. Gradient Estimation: Combining Compressive Sensing and SP

As mentioned above, if function evaluations are expensive, SP works well to avoid the problem of computing function multiple times. However, if the gradient is sparse it makes sense to use the ideas of compressive sensing to our advantage. Combining these two techniques helps us overcome the problem of too many function evaluations and also exploit the sparse structure of the gradient. The idea is to use SP to get sufficient number of observations to be able to recover the gradient via l_1-minimization. We describe the method in detail in the following sub-sections.

2.1 Compressive Sensing

Assume that $\nabla f \in \mathbb{R}^n$ is an approximately sparse vector. The idea of compressive sensing is based on the fact that typically a sparse vector contains much less information or complexity than its apparent dimension. Therefore one should be able to reconstruct ∇f with considerable accuracy with much less information than that of order n. We will make these ideas more precise in the forthcoming discussion on compressive sensing. We state all the results for vectors in $\mathbb{R}^n$. All of these results also hold for vectors over $\mathbb{C}$. We start by defining what we mean by sparse vectors.

Definition 1 (Sparsity) *The support of a vector $x \in \mathbb{R}^n$ is defined as:*

$$supp(x) := \{j \in [n] : x_j \neq 0\}.$$

where $[n] = \{1, 2, \dots, n\}$. The vector $x \in \mathbb{R}^n$ is called s-sparse if at most s of its entries are nonzero, i.e., if

$$\|x\|_0 := card(supp(x)) \leq s.$$

We assume that the observed data $y \in \mathbb{R}^m$ is related to the original vector $x \in \mathbb{R}^n$ via $Ax = y$ for some matrix $A \in \mathbb{R}^{m \times n}$, where $m < n$. In other words, we have a linear measurement process for observing x. The theory of compressive sensing tells us that if x is sparse, then it can be recovered from y by solving a convex optimization problem. In particular, given a suitable matrix A and appropriate m, the following l_1-minimization problem recovers x exactly.

$$\min_{z \in \mathbb{R}^n} \|z\|_1 \text{ subject to } y = Az \tag{1}$$

where $y = Ax$ are the m observations. These ideas were introduced by E. Candés and T. Tao in their seminal paper on near-optimal signal reconstruction (Candés and Tao, 2006). In this paper, the authors proved that the matrices suitable for the recovery need to have what is called the restricted isometry property (RIP). A large class of random matrices satisfy the RIP with quantifiable 'high probability' and are therefore suitable for reconstruction via l_1-minimization. In particular, subgaussian matrices have been shown to have RIP with high probability and are suitable for the aforementioned reconstruction scheme for $m \sim s \log(n/s)$. This gives the explicit relationship between the sparsity level s, the dimension of the original vector n and the dimension of the observed data m. In recent times some work has been done to construct deterministic matrices with this restricted isometry property (Bandeira et. al, 2013). The current known lower bound on m for deterministic matrices is of the order of s^2 where s is the sparsity. Thus random matrices are a better choice for linear measurement for reconstruction via compressive sensing if one is willing to settle for probabilistic guarantees.

For the scope of this paper, we consider robust recovery options using Gaussian random matrices, i.e., matrices whose entries are realizations of independent standard normal random variables.

Remark 2 *Matrices with more structure such as random partial Fourier matrix or more generally, bounded orthonormal systems $\{\phi_i\}_{i=1}^N$ can also be used as meaurement matrices for compressive sensing techniques. Given a random draw of such a matrix with associated constant $K_0 \geq 1$ (where K_0 is the bound on $\|\phi_i\|_\infty$), a fixed s-sparse vector x can be reconstructed via l_1-minimization with high probability provided $m \geq CK_0^2 s \log n$. For more details on random sampling matrices in compressive sensing (see Foucart and Rauhut, 2013, Chap. 12).*

The crucial point here is that it is enough that the given vector is sparse in some basis. A more detailed discussion on various aspects of compressive sensing can be found in Foucart and Rauhut (2013). In real-life situations the measurements are almost always noisy. A more general statement of problem in (1), that takes into account bounded noise in measurement, bounded in norm by η, is given by:

$$\min_{z \in \mathbb{R}^n} \|z\|_1 \text{ subject to } \|Az - y\| \leq \eta \tag{2}$$

It may also happen that the original vector x is not sparse but is close to a sparse vector. In other words, we would like the reconstruction scheme to be robust and stable. Foucart and Rauhut (2013, Theorem 9.13) gives explicit error bounds for stable and robust recovery where A is a subgaussian matrix. The bound is expressed in terms of $\sigma_s(x) := \inf\{\|x - z\| : z \in \mathbb{R}^n \text{ is } s\text{-sparse}\}$, the distance of x from the nearest s-sparse vector, and the measurement error. See Candés, Romberg and Tao (2006); Candés et. al (2005); Kabanava and Rauhut (2015) for more on robust and stable recovery via compressive sensing.

We assume that our observations $y = (y_1, \ldots, y_m)$ are noisy. The following theorem gives an error bound on the reconstruction from noisy measurements using a Gaussian matrix.

Theorem 3 (Theorem 9.20 in (Foucart and Rauhut, 2013)) *Let $x \in \mathbb{R}^n$ be a s-sparse vector. Let $M \in \mathbb{R}^{m \times n}$ be a randomly drawn Gaussian. Assume that noisy measurements $y = Mx + \xi$ are taken with $\|\xi\| \leq \eta$. If for $0 < \epsilon < 1$ and some $\tau > 0$,*

$$\frac{m^2}{m+1} \geq 2s\left(\sqrt{\log(en/s)} + \sqrt{\frac{\log(\epsilon^{-1})}{s}} + \frac{\tau}{\sqrt{s}}\right)^2, \tag{3}$$

then with probability at least $1 - \epsilon$ every minimizer $\hat{x}$ of $\|z\|_1$ subject to $\|Mz - Mx\| \leq \eta$ satisfies

$$\|x - x^*\| \leq \frac{2\eta}{\tau}.$$

See Theorem 9.29 in Foucart and Rauhut (2013) for a statement for stable and robust recovery via Gaussian matrices.

2.2 Simultaneous Perturbation Stochastic Approximation

As discussed above we have a fairly good reconstruction of a sparse gradient ∇f given a sufficient number of observations $\{y_i\}$. However, as mentioned before, the problem often is the unavailability of these observations. Even though observations for ∇f are not readily available, one may compute y_i's using the available information, that is, noisy measurements of the function f. Note that we have, however, assumed that the function evaluations are computationally expensive. We will now address this issue of estimating ∇f with low computational overheads.

Let e_i denote the i^{th} coordinate direction for $1 \leq i \leq n$. We consider the finite difference approximation

$$\frac{\partial f(x(k))}{\partial x_i} \approx \frac{f(x(k) + \delta e_i) - f(x(k) - \delta e_i)}{2\delta}$$

where $x(k) = (x_1(k), \dots, x_n(k))$ and $\delta > 0$. By Taylor's theorem, the error of estimation is $O(\delta \|\nabla^2 f(x(k))\|)$ where $\nabla^2 f$ denotes the Hessian. This estimate requires $2n$ function evaluations. Replacing the 'two sided differences' $(f(x(k) + \delta e_i) - f(x(k) - \delta e_i))/2$ above by 'one sided differences' $(f(x(k) + \delta e_i) - f(x(k)))$ reduces this to $n+1$, which is still large for large n. Given that we have assumed f to be such that the function evaluations are computationally expensive, an alternative method is desirable. We use the method devised by Spall (Spall, 1992) in the context of stochastic gradient descent, known as Simultaneous Perturbation Stochastic Approximation (SPSA).

Recall the stochastic gradient descent scheme (Borkar, 2008)

$$x(k+1) = x(k) + a(k)\left[-\nabla f(x(k)) + M(k+1)\right], \tag{4}$$

where:

- $\{M(k)\}$ is a square-integrable martingale difference sequence, viz., a sequence of zero mean random variables with finite second moment satisfying

  $$E\left[M(k+1) | x(m), M(m), m \leq k\right] = 0 \ \forall \ k \geq 0,$$

  i.e., it is uncorrelated with the past. We assume that it also satisfies

  $$\sup_k E\left[\|M(k+1)\|^2 | x(m), M(m), m \leq k\right] < \infty, \tag{5}$$

- $\{a(k)\}$ are step-sizes satisfying

  $$a(k) > 0 \ \forall k, \ \sum_k a(k) = \infty, \ \sum_k a(k)^2 < \infty. \tag{6}$$

The term in square bracket in (4) stands for a noisy measurement of the gradient. Under mild technical conditions, $x(k)$ can be shown to converge a.s. to a local minimum of f (Borkar, 2008). The idea is that the incremental adaptation due to the slowly decreasing step-size $a(k)$ averages out the noise $\{M(k)\}$, rendering this a close approximation of the classical gradient descent with vanishing error (Borkar, 2008). In practice the noisy gradient is often unavailable and one has to use an approximation $\widehat{\nabla f}$ thereof using noisy evaluations of f, e.g., the aforementioned finite difference approximations, which lead to the Kiefer-Wolfowitz scheme. That is where the SP scheme comes in. We describe this next.

Let $\{\Delta_i(k), 1 \leq i \leq n, k \geq 0\}$ be i.i.d. zero mean random variables such that

- $\Delta(k) = (\Delta_1(k), \ldots \Delta_n(k))$ is independent of $M(\ell), \ell \leq k+1$.
- $P(\Delta_i(k) = 1) = P(\Delta_i(k) = -1) = 1/2$.

Then by Taylor's theorem, we have that for $\delta > 0$:

$$\frac{f(x(k) + \delta\Delta(k)) - f(x(k))}{\delta\Delta_i(k)} \approx \frac{\partial f}{\partial x_i}(x(k)) + \sum_{j \neq i} \frac{\partial f}{\partial x_i}(x(k)) \frac{\Delta_j}{\Delta_i}. \tag{7}$$

Note that since Δ_j's are i.i.d. zero mean random variables, we have for $j \neq i$,

$$\mathbb{E}\left[\frac{\partial f}{\partial x_i}(x(k)) \frac{\Delta_j}{\Delta_i} \Big| x(m), M(m), m \leq k-1\right] = 0.$$

Hence for the purpose of stochastic gradient descent, the second term in (7) acts as a zero mean noise (i.e., martingale difference) term that can be clubbed with $M(k+1)$ as martingale difference noise and gets averaged out by the iteration. This serves our purpose, since the above scheme requires only two function evaluations per iterate given by

$$x_i(k+1) = x_i(k) + a(k) \left[-\frac{f(x(k) + \delta\Delta(k)) - f(x(k))}{\delta\Delta_i(k)}\right] + M_i(k+1).$$

Our idea is to generate $\widetilde{\nabla f}$ according to the scheme discussed above.

It should be mentioned that Spall also introduced another approximation based on a single function evaluation (see Borkar, 2008, chap. 10). But this suffers from numerical issues due to the 'small divisor' problem, so we do not pursue it here.

2.3 Main result

As mentioned in the introduction, the idea is to combine the SP and compressive sensing to obtain a sparse approximation of ∇f. Note that while SP gives an estimate with zero-mean error, the final estimate of gradient obtained after compressive sensing may not be unbiased. To avoid the error from piling up we need to average out the error at SP stage. We propose the following algorithm for estimating gradient of f. Let a_i denote the row vectors of A.

Algorithm 1 Gradient Estimation at some $x \in \mathbb{R}^n$ with SP and Compressive Sensing

```
Initialization:
$A = (a_{ij})_{m\times n} \leftarrow$ random Gaussian matrix.

$\delta \leftarrow$ small positive scalar.

• $y_i^\ell = \frac{f(x+\delta\sum_j^m \Delta_j^\ell a_j)-f(x)}{\delta\Delta_i^\ell}$ for $i = 1,\dots,m;\ \ell = 1,\dots,k$,
  where $\Delta_j^\ell$ are i.i.d. zero mean Bernoulli random variables taking values in $\{-1,1\}$.

• Set $\bar{y}_i := \frac{\sum_{\ell=1}^k y_i^\ell}{k},\ i = 1,\dots,m$.

• $y = (\bar{y}_1,\dots,\bar{y}_m) = A\nabla f(x) + \zeta$ where $\zeta$ denotes the bounded error with bound $\|\zeta\| \le \eta$.

• Solve the $l_1$-minimization problem stated below to obtain $\widetilde{\nabla f}$:

      minimize $\|z\|_1$ subject to $\|Az - y\| \le \eta$

Output: estimated gradient $\widetilde{\nabla f}(x)$.
```

There are several algorithms for the l_1-minimization problem that appears in the last step of the algorithm above. A detailed discussion of these algorithms, including the Homotopy method (used in Section 3 for the l_1-minimization step), can be found in Yang et. al (2010) and Foucart and Rauhut (2013, Chap. 15).

The following theorem states that with high probability such an approximation is "close" to the actual gradient.

Theorem 4 *Let $f : \mathbb{R}^n \mapsto \mathbb{R}$ be a continuously differentiable function with bounded sparse gradient. Then for $m \in \mathbb{N}$ such that it satisfies the bound in (3), $0 < \epsilon << \frac{1}{m}$, and given $\delta > 0$ (as in (7)) and $\tau > 0$ (as in Theorem 3), ∇f can be estimated by a sparse vector $\widetilde{\nabla f}$ such that with probability at least $1 - \epsilon m$,*

$$\|\widetilde{\nabla f} - \nabla f\| < \frac{2t}{\tau}$$

where $t > 2mO(\delta)$.

Proof Let $A \in \mathbb{R}^{m\times n}$ be a Gaussian matrix such that m satisfies (3). Then, following the same idea as in (7), we have:

$$\begin{aligned} y_i &= \frac{f(x + \delta \sum\limits_{j=1}^{m} \Delta_j a_j) - f(x)}{\delta \Delta_i} \\ &= \langle \nabla f(x), a_i \rangle + \sum_{j \neq i} \frac{\Delta_j \langle \nabla f, a_i \rangle}{\Delta_i} + O(\delta). \end{aligned}$$

So we get

$$y = A\nabla f + \text{ `error'}, \tag{8}$$

where we quantify the 'error' below.

The above computation is carried out k times independently, keeping the matrix A fixed and choosing the random vector Δ according to the distribution defined in Section 2.2. The reason for this additional averaging is as follows. The reconstruction in compressive sensing need not give an unbiased estimate, since it performs a nonlinear (minimization) operation. Thus it is better to do some pre-processing of the SP estimate (which is nearly, i.e., modulo the $O(\delta)$ term, unbiased) to reduce its variance. We do so by repeating it k times with independent perturbations and taking its arithmetic mean. This may seem to defeat our original objective of reducing function evaluations, but the k required to get reasonable error bounds is not large as our analysis shows later, and the computational saving is still significant (see 'Remark 5' below).

Denote by y^l the measurement obtained at l^{th} iteration of SP. The error for a single iteration is given by

$$\eta^l = \left(\sum_{j \neq 1} \frac{\Delta_j^l \langle \nabla f, a_1 \rangle}{\Delta_1^l} + O(\delta), \ldots, \sum_{j \neq m} \frac{\Delta_j^l \langle \nabla f, a_m \rangle}{\Delta_m^l} + O(\delta) \right).$$

Denote by $X_{ij}^l = \frac{\Delta_j^l \langle \nabla f, a_i \rangle}{\Delta_i^l}$, $j \neq i$. X_{ij}^l are zero-mean conditionally (given past iterates) independent random variables.

The error vector after k iterations is given by

$$\begin{aligned}
\eta &= \frac{1}{k} \sum_{l=1}^{k} \eta^l \\
&= \frac{1}{k} \sum_{l=1}^{k} \left(\sum_{j \neq 1} X_{1j}^l + O(\delta), \ldots, \sum_{j \neq m} X_{mj}^l + O(\delta) \right) \\
&= \left(\frac{1}{k} \sum_{l=1}^{k} \sum_{j \neq 1} X_{1j}^l + O(\delta), \ldots, \frac{1}{k} \sum_{l=1}^{k} \sum_{j \neq m} X_{mj}^l + O(\delta) \right). \qquad (9)
\end{aligned}$$

In order to apply the ideas from compressive sensing as in Theorem 3, we need to have a bound on the error $\|\eta\|$. This is obtained as follows. Let $K > 0$ be a constant such that the $O(\delta)$ term above is bounded in absolute value by $K\delta$. K can, e.g., be a bound on $\|\nabla^2 f\| \|A\|$ by the mean value theorem, where we use the Frobenius norm. Choose $C \geq \sup |\langle \nabla f, a_i \rangle|$

and $t > 2mK\delta$. Then, by Hoeffding's inequality we have,

$$\begin{aligned} P(\|\eta\| \geq t) &\leq \sum_{i=1}^{m} P\left(\Big| \frac{1}{k} \sum_{l=1}^{k} \sum_{j \neq i} X_{ij}^{l} + O(\delta) \Big| > t/m \right) \\ &\leq \sum_{i=1}^{m} P\left(\Big| \sum_{l=1}^{k} \sum_{j \neq i} X_{ij}^{l} \Big| > kt/2m \right) \\ &\leq 2me^{-\frac{kt^2}{2m^2(m-1)C^2}}. \end{aligned}$$

Choose the number of iterations, $k > \frac{2m^3C^2}{t^2} \log\left(\frac{2}{\epsilon}\right)$. Then,

$$P(\|\eta\| \geq t) \leq \epsilon m. \tag{10}$$

We define $\widetilde{\nabla f}$ to be the reconstruction of the gradient using m measurements. That is, $\widetilde{\nabla f}$ solves the following optimization problem:

$$\min_{z \in \mathbb{R}^n} \|z\|_1 \text{ subject to } \|Az - y\| \leq t,$$

where y is as in (8). Our claim then follows from the bound in (10) and Theorem 3. ■

Remark 5 *Note that the minimum number of iterations of SP required to obtain a "good" estimate of ∇f is given by*

$$\begin{aligned} k &> \frac{2m^3C^2}{t^2} \log\left(\frac{2}{\epsilon}\right) \\ &\geq \frac{mC^2}{2K^2\delta^2} \log\left(\frac{2}{\epsilon}\right) \\ &\geq \frac{s\tilde{C}}{\delta^2} \log\left(\frac{n}{s}\right) \log\left(\frac{2}{\epsilon}\right). \end{aligned}$$

for a suitable constant $\tilde{C}$.

The above $\widetilde{\nabla f}$ can now be used as an effective gradient in various problems involving gradients of high-dimensional functions. Three such applications are discussed in the next section.

3. Applications

We consider the application of our method to manifold learning and optimization problems. The gradient estimates obtained using our method can be used to estimate the gradient outer product matrix or can be plugged into an optimization scheme. In the former case, along with an example, we also provide error bounds on the estimated and actual gradient outer product matrix. For the latter case, we look at an example and provide suitable modifications to existing algorithms to achieve faster convergence. Algorithm 1 described in Section 2.3 is used for gradient estimation.

As mentioned before, there are various algorithms available for carrying out the l_1-minimization. Here we use the homotopy method (see Donoho and Tsaig, 2008; Foucart and Rauhut, 2013; Yang et. al, 2010). Homotopy method solves the quadratically constrained l_1-minimization problem (2) by considering the following l_1-regularized least squares functional:

$$F_\lambda(x) = \frac{1}{2}\|Az - y\| + \lambda\|z\|_1 \quad \text{for } \lambda > 0 \tag{11}$$

The algorithm traces the piece-wise linear and continuous solution path $\lambda \mapsto x_\lambda$ and at each step, an element is added or removed from the support set of the current minimizer. If the minimizer x^* of (1) is unique then the minimizer x_λ of (11) converges to x^*. The idea is to start with a large λ such that the minimizer x_λ of (11) is zero and then trace the solution trajectory in the direction of decreasing λ.

We consider a Gaussian matrix A and get $y(n) \quad \leftarrow A\nabla f(x(n))$ + error as obtained in equation (8). The last step is to obtain $\widetilde{\nabla} f(x)$ via l_1-recovery from observations y and Gaussian random matrix A using the homotopy method described above. All the simulations were performed on MATLAB using the available toolbox for l_1-minimization. (Berkeley database: http://www.eecs.berkeley.edu/ yang/software/ l1benchmark/).

Consider a function $f : \mathbb{R}^{25000} \mapsto \mathbb{R}$ given by $f(x) = x^T M M^T x$ where, M is 25000×3-dimensional matrix with 3 non-zero elements per column. Let A be a random Gaussian matrix that is used for measurement. We consider $m = 50$ measurements.

Figure 1 shows the performance of the proposed method with varying number of SP iterations. Figures 2 and 3 show the comparison between our method and naive SP for estimating gradient with gradually increasing number of iterations for averaging over SP (The quantity 'k' in (9)). As mentioned earlier, since the gradient is assumed to be sparse, using naive SP to compute derivative in each direction seems wasteful. Although the error diminishes as the number of iterations for SP increase, the proposed method combining compressive sensing with SP consistently performs better.

Figures 4 and 5 show that the proposed method works well with higher sparsity levels too. It also shows that with higher s, performance of naive SP improves. This is expected. The extremely high error in the naive SP method (especially for small s) is owing to the fact that the actual gradient is extremely sparse and in the beginning SP method ends up populating almost all the coordinates. That contributes to the high percentage of error as seen in the aforementioned figures.

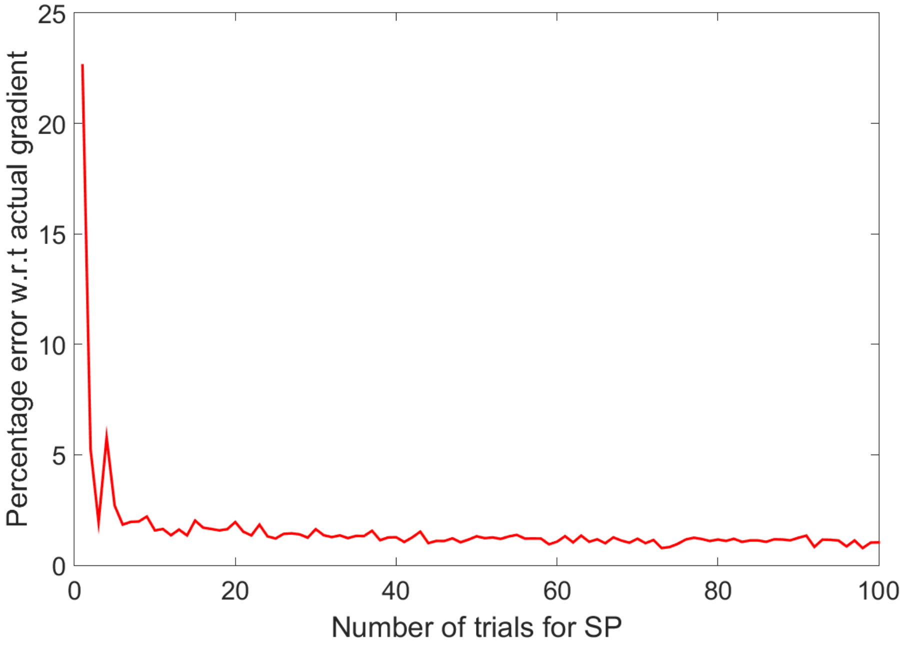
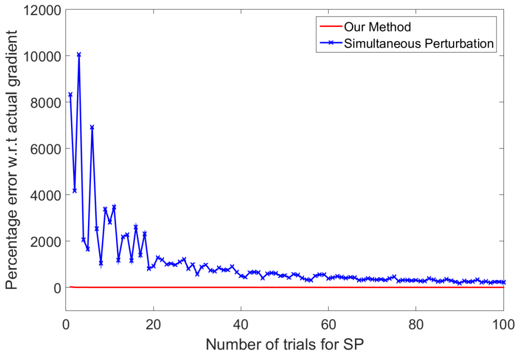


Figure 1: Percentage error of $\|\nabla f - \widetilde{\nabla f}\|$ in our method, with varying number of iterations k for SP. Here, $n = 25000, s = 3$ and $m = 50$.



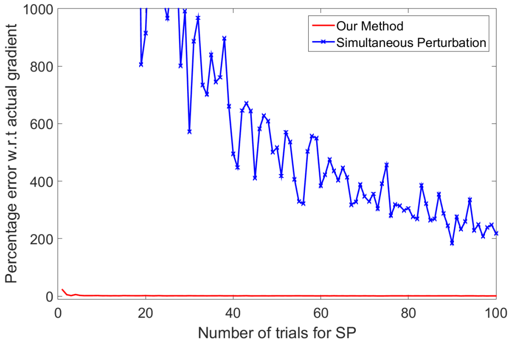


Figure 2: Performance of the proposed algorithm vs. the SP method with varying number iterations k at SP step. Here $n = 25000, s = 3$ and $m = 50$

Figure 3: A closer look at Figure 2 : Performance of the proposed algorithm vs. the SP method with varying number iterations k at SP step. Here $n = 25000, s = 3$ and $m = 50$.

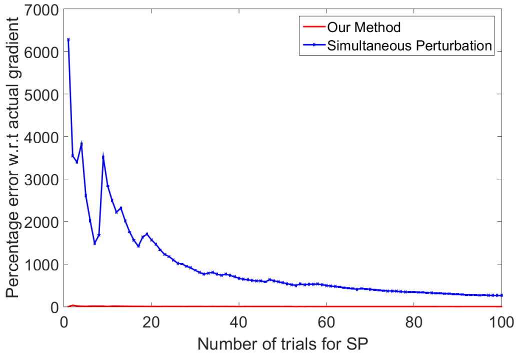


Figure 4: Performance of the proposed algorithm vs. the SP method with varying number iterations k at SP step. Here $s = 50$ and $m = 500$.

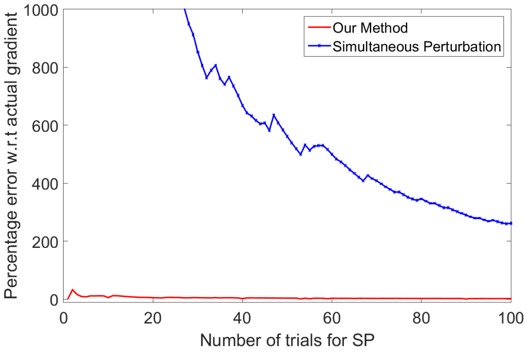


Figure 5: A closer look at Figure 4: Performance of the proposed algorithm vs. the SP method with varying number iterations k at SP step. Here $s = 50$ and $m = 500$.

Before we consider specific applications, we illustrate how the percentage error of estimated gradient with varying k for different sparsity levels s. For appropriately large m, for small k the error is high (this matches with the discussion in Remark 5). As k increases the error is much less. As long as m satisfies (3), the compressive sensing results apply. Figures 6 and 7 show the behaviour of the proposed method with variation in the sparsity, but with constant number of observations. We consider 10000-dimensional vector with $m = 50$ observations. As expected, for a fixed m, as the sparsity increases, increasing k no longer helps as the compressive sensing results do not apply and the error increases. $f(x) = x(1)^2 + \ldots + x(s)^2$ was used as a test function for both the simulations.

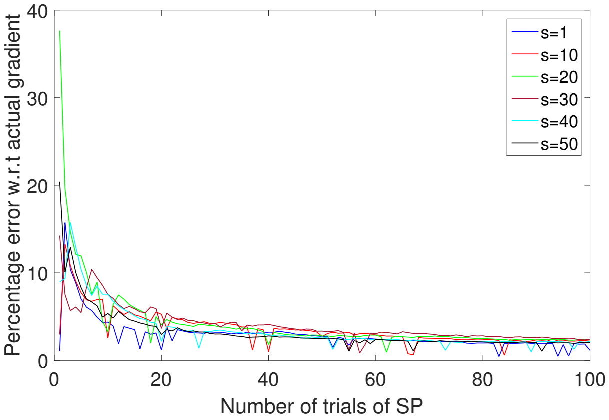


Figure 6: Performance of the proposed algorithm with variation k for different sparsity levels.

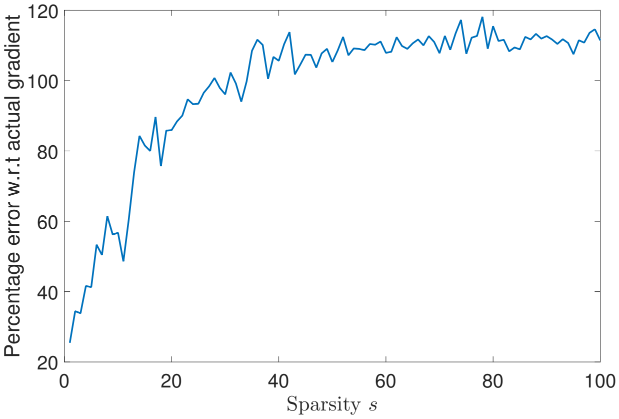


Figure 7: Performance of the proposed algorithm with variation in sparsity.

3.1 Manifold Learning: Estimating e.d.r. space

Consider the following semi-parametric model

$$Y = f(X) + \epsilon$$

where ϵ is noise and f is a smooth function $\mathbb{R}^n \mapsto \mathbb{R}^m$ of the form $f(X) = g(b_1^T X, \dots, b_d^T X)$. Define by B the matrix $(b_1, b_2, \dots, b_d)^T$. B maps the data to a d-dimensional relevant subspace. This means that the function f depends on a subspace of smaller dimension given by Range(B) (Note that this is essentially the local view in manifold learning : B can vary with location.). The vectors or the directions given by the vectors b_i are called the effective dimension reducing directions or e.d.r. The question is: how to find the matrix B? It turns out that if f doesn't vary in some direction v, then $v \in Null(E_X[G])$ where G is the gradient outer product matrix defined as

$$G = [[G_{ij}]] \text{ where } G_{ij} = \left\langle \frac{\partial f}{\partial x_i}(X), \frac{\partial f}{\partial x_j}(X) \right\rangle$$

and $E_X[\,\cdot\,]$ denotes the expectation over X. Lemma 1 from (Wu et. al, 2010) stated below implies that to find the e.d.r. directions it is enough to compute $E_X[G]$.

Lemma 6 *Consider the semi-parametric model*

$$Y = g(b_1^T X, \dots, b_d^T X) + \epsilon, \tag{12}$$

where ϵ represents zero mean finite variance noise. Then the expected gradient outer product (EGOP) matrix G is of rank at most d. Furthermore, if $\{v_1, \dots, v_d\}$ are the eigenvectors associated to the nonzero eigenvalues of G, the following holds:

$$Span(B) = Span(v_1, \dots, v_d).$$

Clearly, calculating $E_X[G(X)]$ is computationally heavy. We therefore try to estimate this matrix. Several methods are known for estimating the EGOP and this has been a very popular problem in statistics for a while. The idea of using EGOP for obtaining e.d.r. originated in Li (1991). While there are other methods based on inverse regression etc., most of the efforts have been directed towards getting an efficient way to estimate gradients in order to finally estimate EGOP (see Xia et. al, 2002). In Mukherjee, Wu and Zhou (2010), the authors use their method of gradient estimation for this purpose. The idea is to use sample observations $\{f(x_i)\}$ for $\{x_i\}$ in a neighborhood of the given point x and minimize over z the error

$$\frac{1}{n^2}\sum_{i,j=1}^{n} w_{ij}[y_i - f(x_j) - \langle z, (x_i - x_j)\rangle]^2,$$

where $w_{ij} \geq 0$ are weights ('kernel') that favor locality $x_i \approx x$ and are typically Gaussian, with regularization in a reproducing kernel Hilbert space (RKHS). The minimizer then is the desired estimate. In Trivedi et. al (2014) a rather simple rough estimator using directional derivative along each coordinate direction is provided. The authors demonstrate that for the purpose of finding e.d.r., a rough estimate such as theirs suffices. We also propose a method via gradient estimation. Take $\widehat{G}$ to be the matrix defined by

$$\widehat{G}_{ij} = \Big\langle \widetilde{\frac{\partial f}{\partial x_i}}, \widetilde{\frac{\partial f}{\partial x_j}} \Big\rangle.$$

In other words, $\widehat{G} = \widetilde{\nabla f}\widetilde{\nabla f}^T$, where $\widetilde{\nabla f}$ denotes the estimate of ∇f obtained by algorithm 1. We impose our previous restrictions on f. That is, the function evaluations at any point are expensive and the gradient of f is sparse. In this case we propose an estimate for $E_X[G]$ by the mean of $\widehat{G}$ over a sample of r points given by the set $\chi = \{(x_i, f(x_i))\}_{1\leq i\leq r}$. By $\langle\ \cdot\ \rangle$, we shall denote the empirical mean over the sample set χ. Thus,

$$\langle G(X)\rangle = \frac{1}{r}\sum_{x_i\in\chi} \nabla f(x_i)\nabla f(x_i)^T$$

and

$$\langle \widehat{G}(X)\rangle = \frac{1}{r}\sum_{x_i\in\chi} \widetilde{\nabla f}(x_i)\widetilde{\nabla f}(x_i)^T.$$

Theorem 7 *Let $f : \mathbb{R}^n \mapsto \mathbb{R}^k$ from the semi-parametric model in* (12) *be a continuously differentiable function with bounded sparse gradient. Then, for $0 < \epsilon << \frac{1}{m}$ and some $\tau > 0$, with probability at least $1 - \epsilon m$,*

$$\left\| E_X[G] - \langle \widehat{G}\rangle \right\| < \frac{6R^2}{\sqrt{r}}\left(\sqrt{\ln n} + \sqrt{\ln\frac{1}{\epsilon}}\right) + \frac{2t}{\tau}\left(\frac{2t}{\tau} + 2R\right)$$

where r is the sample size, R is such that $\|\nabla f\| \leq R$, t is as in Theorem 4 and $m \in \mathbb{N}$ is such that it satisfies the bound in (3).

The proof closely follows the line of argument in Trivedi et. al (2014).

Proof Note that,

$$\|E_X[G(X)] - \langle \widehat{G}(X)\rangle\| \le \|E_X[G(X)] - \langle G(X)\rangle\| + \|\langle G(X)\rangle - \langle \widehat{G}(X)\rangle\|.$$

The idea is to bound each term. We use concentration inequality for sum of random matrices (see Trivedi et. al, 2014, Lemma 1) and (Tropp, 2012) for more general results, to claim that for $\epsilon > 0$,

$$\|E_X[G(X)] - \langle G(X)\rangle\| \le \frac{6R^2}{\sqrt{r}}\left(\sqrt{\ln n} + \sqrt{\ln \frac{1}{\epsilon}}\right)$$

with probability $\ge 1 - \epsilon$. For the second term, it is enough to show that it is bounded for any single sample point x. Observe that for any two vectors v and w, $\|vv^T - ww^T\| \le \|(v-w)(v+w)^T\|$ Using this we get, for a fixed x,

$$\begin{aligned}
\|G(x) - \widehat{G}(x)\| &= \|\nabla f(x)\nabla f(x)^T - \widetilde{\nabla f}(x)\widetilde{\nabla f}(x)^T\| \\
&\le \|\nabla f(x) - \widetilde{\nabla f}(x)\|\|\nabla f(x) + \widetilde{\nabla f}(x)\| \\
&\le \|\nabla f(x) - \widetilde{\nabla f}(x)\|\left(\|\nabla f(x) - \widetilde{\nabla f}(x)\| + 2\|\nabla f(x)\|\right) \\
&\le \frac{2t}{\tau}\left(\frac{2t}{\tau} + 2R\right)
\end{aligned}$$

with probability $\ge 1 - \epsilon m$, where the last inequality is obtained by applying the bound from Theorem 4. ■

We now simulate an example to illustrate the decay of the error $\|\langle G(X)\rangle - \langle \widehat{G}(X)\rangle\|_F$ (See Figure 8). Consider a function $f : R^{25000} \mapsto R^3$ given by: $f_i(x) = x^T M_i M_i^T x$ where, M_i is a 25000-dimensional vector with 3 non-zero elements and $f_i(x)$ corresponds to the i^{th} dimension of $f(x)$. A 25000×100 Gaussian random matrix is used for the compressive sensing part of the algorithm. The plot of percentage in normed error between $\langle G(X)\rangle$ and $\langle \widehat{G}(X)\rangle$, i.e. $||\langle G(X)\rangle - \langle \widehat{G}(X)\rangle||_F^2$ is shown below by varying number of samples $r = 1$ to 25. Remember that due to the bias at compressive sensing step, we need to average out the gradient estimation error at SP step. This is done in $k = 100$ iterations.

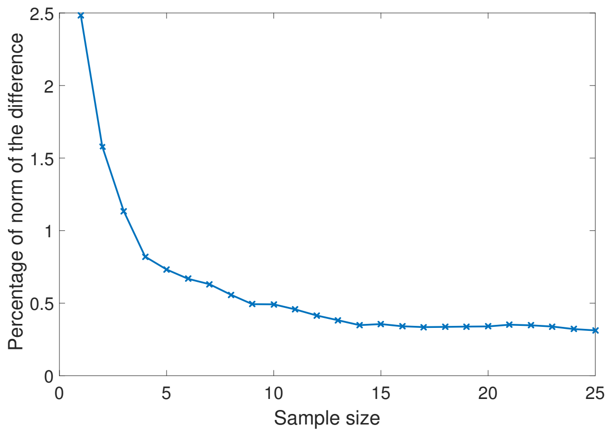


Figure 8: Percentage error in $\|\langle G(X)\rangle - \langle \widehat{G}(X)\rangle\|$ with number of samples r varying from 1 to 25. Here, $n = 25000, s = 3, m = 100$ and $k = 100$.

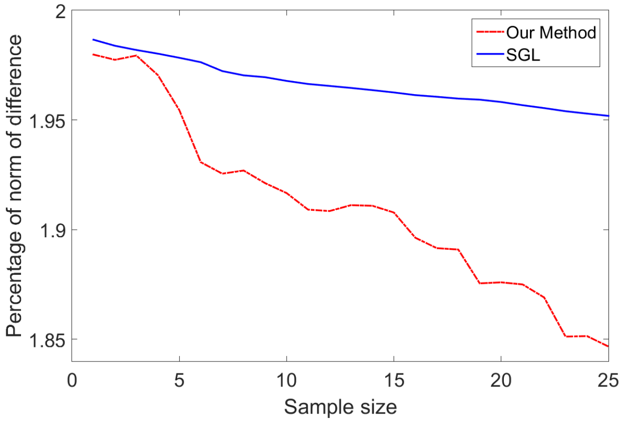


Figure 9: Comparison of percentage error of $\|\langle G(X)\rangle - \langle \widehat{G}(X)\rangle\|$ computed by plugging gradient estimates by proposed method vs SGL method. Here, $n = 10000, s = 30, m = 100$ and $k = 100$. Number of samples for SGL method are 100.

Learning e.d.r. by estimating the gradient using the method proposed in this paper was compared with the SGL (Sparse Gradient Learning) method proposed in (Mukherjee, Wu and Zhou, 2010) using the same function as above and an exponential kernel (See `http://www2.stat.duke.edu/~sayan/soft.html` for details). This is illustrated in Figure 9. Here, $n = 10000$ and the measurement matrix is a 10000×100 Gaussian matrix. The SP step is averaged over 100 iterations. 100 samples were considered for the SGL method with the neighborhood radius of of 0.05. Sparsity of the gradient vector is 30.

3.2 Optimization

We consider next a typical problem of function minimization, but only consider a function with sparse gradient. In other words, we want to minimize $f(x)$ where

$$f : \mathbb{R}^n \mapsto \mathbb{R}$$

is a continuously differentiable real-valued Lipschitz function such that function evaluation at a point in $\mathbb{R}^n$ is typically expensive. We also assume that n is large and that ∇f is sparse. In addition, we assume that the critical points of f (i.e., the zeros of ∇f) are isolated. (This is generically true unless there is overparametrization.) The idea is to use the stochastic gradient scheme (4) with the standard assumptions (5), (6). It follows from the theory of stochastic approximation (see Borkar, 2008, chap. 2) that under above conditions, the solution of the random difference equation (4) tracks with probability one the trajectory of the solution of a limiting o.d.e. as long as the iterates remain bounded, which they do under mild additional conditions on f. Following (Borkar, 2008, chap. 2), we use this so called 'o.d.e. approach' which states that the algorithm will a.s. converge to the equilibria of the limiting o.d.e., which is

$$\dot{x}(t) = -\nabla f(x(t)). \tag{13}$$

For this, f itself serves as the Lyapunov function, leading to the conclusion that the trajectories of (13) and therefore a.s., the iterates of (4) will converge to one of its equilibria, viz., the critical points of f. In fact under additional conditions on the noise, it will converge to a (possibly random) stable equilibrium thereof, viz., a local minimum (*ibid.*, Chapter 4).

The stochastic gradient scheme requires $\nabla f(x)$ at each iteration. The problem often is the unavailability of $\nabla f(x)$, as already noted. It is therefore important to have a good method for estimating the gradient. Typically one would obtain noisy measurements and hence the estimate will have a non-zero error η. It is known that if the error remains small, the iterates converge a.s. to a small neighbourhood of some point in the set of equilibria of (13). We analyze the resultant error below. Also, the error obtained in SP is zero-mean modulo higher order terms, so one can even take an empirical average over a few separate estimates in order to reduce variance. For high dimensional problems, the number of function evaluations remains still small as compared with, e.g., the classical Kiefer-Wolfowitz scheme. We use Theorem 4 to justify using SP (Simultaneous Perturbation Stochastic Approximation) combined with compressed sensing to obtain an approximation for the gradient and then use the above scheme to minimize f.

Consider the following stochastic approximation scheme:

$$x_{n+1} = x_n + a(n)[-\nabla f(x_n) + M_{n+1} + \eta(n)] \tag{14}$$

where $\{\eta(n)\}$ is the additional error arising due to the error in gradient estimation. That is, $\widetilde{\nabla f}(x_n) = \nabla f(x_n) - \eta(n)$. If $\sup_n \|\eta(n)\| < \epsilon_0$ for some small ϵ_0, then the iterates of (14) converge to a small neighbourhood A of some point x^* in $H = \{x : \nabla f(x) = 0\}$ (see Tadic and Doucet, 2017) and (Borkar, 2008, chap. 10)). This is ensured by a Lyapunov argument as follows. The limiting o.d.e. is of the form

$$\dot{x}(t) = -\nabla f(x(t)) + \breve{\eta}(t)$$

for some measurable $\breve{\eta}(\cdot)$ with $\|\breve{\eta}(t)\| \leq \epsilon_0 \; \forall t$. Then

$$\frac{d}{dt} f(x(t)) = -\|\nabla f(x(t))\|^2 + \langle \nabla f(x(t)), \breve{\eta}(t) \rangle,$$

which is < 0 as long as $\|\nabla f(x(t))\|^2 > |\langle \nabla f(x(t)), \breve{\eta}(t) \rangle|$. Therefore $x(t)$ will converge to the set

$$\{x : \|\nabla f(x)\| \leq \epsilon_0\}.$$

Assume that the Hessian $\nabla^2 f(x^*)$ is positive definite, which is generically so for isolated local minima. Then for A small enough, the lowest eigenvalue $\lambda_m(x)$ of $\nabla^2 f(x)$ for $x \in A$ is > 0. By mean value theorem, $\nabla f(x) = \nabla^2 f(x')(x - x^*)$ for some $x' \in A$, so $\|\nabla f(x)\| \geq \lambda_m(x')\|x - x^*\|$. Thus there is convergence to a ball of radius $\frac{\epsilon_0}{\lambda_m}$ around x^*. (A statement to this result without the estimate on the radius of the ball is contained in Theorem 1 of (Hirsch, 1989).) Thus we have:

Theorem 8 *The stochastic gradient scheme*

$$x_{n+1} = x_n + a(n)[-\widetilde{\nabla f}(x_n) + M_{n+1}]$$

a.s. converges to a ball of radius $O(\epsilon_0)$ centered at some local minimum of f, where $\widetilde{\nabla f}$ is the reconstructed gradient as in Theorem 4 and ϵ_0 is a bound on $\|\widetilde{\nabla f} - \nabla f\|$.

Proof The claim is immediate from the above observations about the perturbed differential equation and Theorem 6, pp. 58-59, (Borkar, 2008). □

See Tadic and Doucet (2017) for a finer analysis. Also, observe that we have only discussed asymptotic convergence above. For real-life optimization problems, however, we must ensure that the scheme in (14) converges to a neighbourhood of x^* in finite time. This is indeed true and recent concentration-type results (see Kamal, 2010; Thoppe and Borkar , 2015)) strengthen the theoretical basis for plugging $\widetilde{\nabla f}(x)$ in place of $\nabla f(x)$ in stochastic gradient descent schemes. The results in Kamal (2010) involve estimates on lock-in probability, i.e., the probability of convergence to a stable equilibrium given that the iterates visit its domain of attraction. An estimate on the number of steps needed to be within a prescribed neighborhood of the desired limit set with a prescribed probability is also obtained. Specifically, the result states that if the n_0th iterate is in the domain of attraction of a stable equilibrium x^*, then after a certain number of additional steps, the iterates remain in a small tube around the differential equation trajectory converging to x^* with probability exceeding

$$1 - \mathrm{O}\left(e^{-\frac{C}{(\sum_{m=n_0}^{\infty} a(m)^2)^{\frac{1}{4}}}} \right),$$

ipso facto implying an analogous claim for the probability of remaining in a small neighborhood of x^* after a certain number of iterates. We refer the reader to Kamal (2010) for details. In Thoppe and Borkar (2015), an improvement on this estimate is proved under additional regularity conditions on ∇f (twice continuous differentiability) using Alekseev's formula. We have omitted the details of both the cases as it needs much additional notation to replicate them here. These would, however, apply to the exact stochastic gradient descent. Since we have an additional error due to approximate gradient as in the preceding theorem, we need to combine the results of *ibid.* with the above theorem to make a weaker claim regarding how small the neighborhood of x^* in question can be. Furthermore, these claims are about iterates which are in the domain of attraction of a stable equilibrium. This, however, is not a problem, as 'avoidance of traps' results as in Section 4.3 of Borkar (2008) (also see Benaim, 1996; Brandiére and Duflo, 1996; Pemantle, 1990) ensure that if the noise is rich enough in a certain precise sense, unstable equilibria are avoided with probability one.

Remark 9 *Note that the gradient descent is a stochastic approximation scheme which itself averages out the noise. So in principle the averaging over k steps at the SP stage in the original algorithm can be skipped. This means that for a stochastic gradient descent scheme, we cut down the cost of function evaluation even further. The simulations in the next section confirm that good results are obtained without averaging over SP iterations. There is, however, a standard trade-off involved between per step computation / speed of convergence, and fluctuations (equivalently, variance) of the estimates: any additional averaging improves the latter at the expense of the former.*

3.3 Numerical experiments

We compare following three algorithms.

1. *Actual Gradient Descent*

   This is the classical stochastic gradient descent with exact gradient.

Algorithm 2 Stochastic Gradient Decent with Compressive Sensing

```
Initialization:
$x(0) = x_{initial}, A \leftarrow random\ Gaussian\ matrix$
$a(n)$ be a sequence that satisfies the properties of stepsize listed above.

Iteration: Repeat until convergence criteria is met at $n = n^{\#}$. At $n^{th}$ iteration:
    $y(n) \leftarrow A\nabla f(x(n))\ +\ error$
    $\widetilde{\nabla f}(x(n)) \leftarrow l_1 -\ minimization\ with\ Homotopy(y(n), A)$
    $x(n+1) \leftarrow x(n) - a(n)[\widetilde{\nabla f}(x(n))]$
Output: Approximate minimizer of $f$ i.e. $x(n^{\#})$
```

Here $Homotopy(y(n), A)$ denotes the l_1-recovery from observations $y(n)$ and Gaussian random matrix A using the homotopy method.

2. *Accelerated Gradient Method*

   Accelerated gradient scheme was proposed by Nesterov (Nesterov, 1983). While Gradient Descent algorithm has a rate of convergence of order $1/s$ after s steps, Nesterov's method achieves a rate of order $1/s^2$. We implement the method here to achieve an improvement in the time complexity further. The idea is to replace the n^{th} iteration above by the following.

```
At $n^{th}$ iteration:
  $y(n) \leftarrow A\nabla f(x(n)) + \text{error}$
  $\widetilde{\nabla f}(x(n)) \leftarrow l_1-$ minimization with $Homotopy(y(n), A)$
  $z(n+1) \leftarrow x(n) - a(n)[\widetilde{\nabla f}(x(n))$
  $x(n+1) \leftarrow (1-\gamma(n))z(n+1) + \gamma(n)z(n)$
where, $\lambda$ and $\gamma$ are as follows:
```

$$\lambda(0) = 0,\ \lambda(n) = \frac{1+\sqrt{1+4\lambda^2(n-1)}}{2},\ \text{and } \gamma(n) = \frac{1-\lambda(n)}{\lambda(n+1)}.$$

   This gives us faster convergence towards the minimum.

3. *Adaptive Method*

   Another way to achieve a faster convergence rate is to perform the l_1-minimization adaptively with the gradient descent. The idea is to again use the homotopy method for l_1-minimization but this part of the algorithm is run for very few iterations. The intermediate approximation of ∇f is then used for performing the stochastic gradient descent. As expected, the errors are high in the beginning but the convergence is faster.

We consider the following function to test our algorithms:

$$f(x) = (x^T M_1^T M_1 x)^3 + (x^T M_2^T M_2 x)^2 + x^T M_1^T M_2 x \tag{15}$$

where, function $f : \mathbb{R}^n \mapsto \mathbb{R}$ and M_1, M_2 are $n \times s$ random matrices. This is to ensure sparsity of the gradient. Here, $n = 25000$ and number of non-zero entries in each column of M_1 and M_2 are $s = 3$. Number of measurements, $m = 500$. A is a $n \times m$ random Gaussian matrix.

Figure 10 show the comparisons between various algorithms described above for the same function.

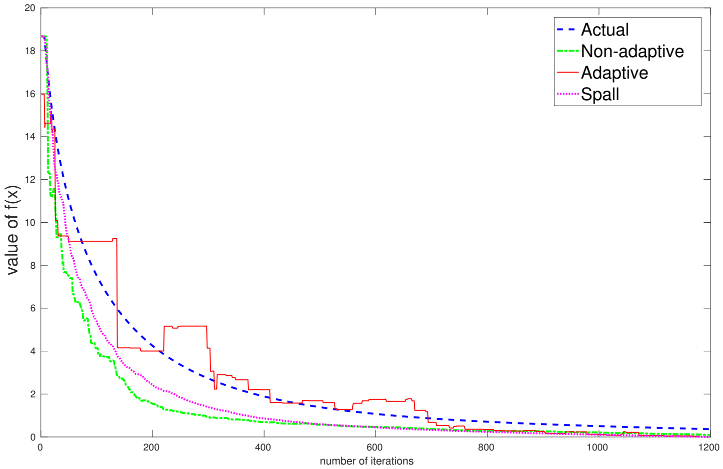


Figure 10: Comparison of Gradient descent with actual gradient ∇f and estimated gradient $\widetilde{\nabla f}$ using non-adaptive, adaptive schemes and Spall's SPSA. Here, $n = 25000, s = 3$ and $m = 500$

Algorithm Used	n = 1000 (sec)	n = 10000 (sec)	n = 25000 (sec)
Adaptive Method	4.3	62.3	85.4
Without Adaptive	50.572	612.3	1004.5
Spall's SPSA	25.0	459.8	864.5
Actual Gradient Method	64.1	4641.12	7526

As expected, adaptive method turns out to be faster compared to the non-adaptive method which in turn is much faster than the algorithm that computes actual gradients. Incidentally, the classical scheme all but converges in under 1000 iterations. Even so it takes more time than the other two which take more iterations. This is because of the heavy per iterate computation for the classical scheme. From the above table it is clear that as the dimensionality of the problem increases, adaptive method proves more and more useful compared to the other two algorithms.

We also compared our method with the method proposed in (Mukherjee, Wu and Zhou, 2010) (See Figure 11). The function in (15) is used for the comparison with $n = 10000$ and $s = 50$. Figure 12 is a scaled down version with $n = 10000, s = 100$ and $m = 500$. Number of samples for the SGL method were 10, chosen with neighbourhood radius 0.05.

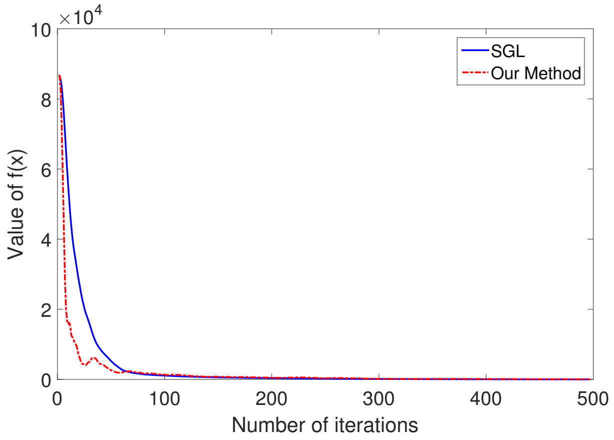


Figure 11: Comparison of Gradient descent using estimated gradient from proposed method vs the SGL method proposed in (Mukherjee, Wu and Zhou, 2010). Here, $n = 10000, s = 50$ and $m = 500$. Number of samples for the SGL method = 10.

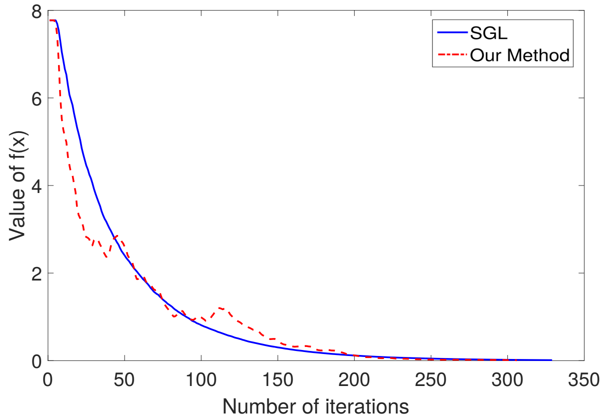


Figure 12: Comparison of Gradient descent using estimated gradient from proposed method vs the SGL method proposed in (Mukherjee, Wu and Zhou, 2010). Here, $n = 10000, s = 100$ and $m = 500$. Number of samples for the SGL method = 10.

The time taken by the SGL method and our method was 993 and 83 seconds respectively. As mentioned earlier, the aim of this paper is to provide a good estimation of gradient when the function evaluations are expensive. In such cases, our method would provide a significant gain in terms of function evaluations needed. While in this example we do see a significant improvement in time taken for the estimation, there is no a priori reason to always expect it. It will indeed be the case when the function evaluations are 'expensive' in terms of the time they take. One expects this to be so when the ambient dimension is high.

3.3.1 An Example: Longitude Estimation

In this section, we test our method on real data. Gradient estimation technique proposed in this paper is applied on UJIIndoorLoc Data Set (Torres-Sospedra et. al, 2014) to estimate the longitude information from signal strengths of 520 wireless access points. The Data-set has 19937 samples for training and 1111 samples for testing. We assume that longitude l is linearly dependent on signal strengths of access points x. Let $\theta \in \mathbb{R}^n$ be the vector assigning weights of signal strengths to each access points. Then, $l(\theta) = \theta^T x$. We recover θ by minimizing regularized l_1 norm of the error:

$$f(\theta) = \sum_{i=1}^{M} \frac{1}{M} \|l_{actual} - l(\theta)\|_1 + \lambda \|\theta\|$$

where, l_{actual} denotes the actual longitude. In this example, $M = 19937$, $n = 520$ and $m = 100$, where m is as in theorem 2.3.

Note that we do not have a closed form expression for the gradient of $f(\theta)$. We compare the proposed method with Spall's SPSA. Parameters like k (number of trials of SP) and iteration for l_1 minimization are chosen so that both methods converge empirically in least number of iterations. Taking into account the higher error in SPSA, we take number of trials of SP to be 20 for Spall's SPSA and 10 for our method. Maximum number of iterations in Homotopy method are limited to 50.

Algorithm:	Proposed Method	Spall's SPSA
Mean percentage error on training data	5.43	5.54
Mean percentage error on test data	5.55	5.48
Time taken to train (sec)	16.77	84.81

We can see that both methods obtain similar performance but the method proposed in this paper is much faster. Figure 13 shows the training performance of both methods. Figures 14, 15 show percentage error in reconstruction of training and test data respectively. Percentage error is sorted for better visualization.

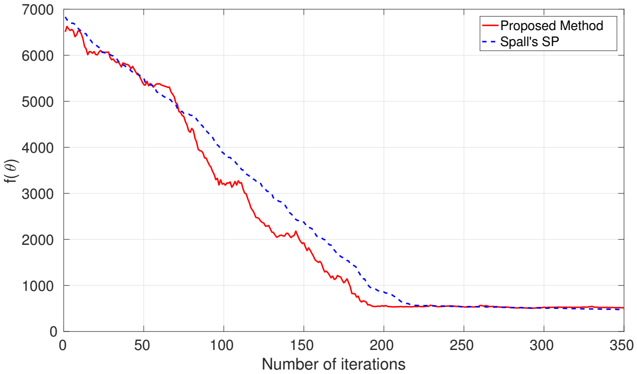


Figure 13: Optimizing $f(\theta)$ using Spall's SPSA and proposed method.

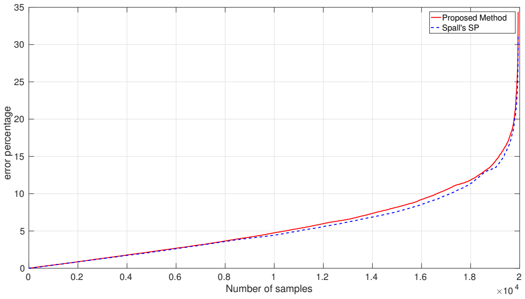


Figure 14: Sorted percentage error on train data.

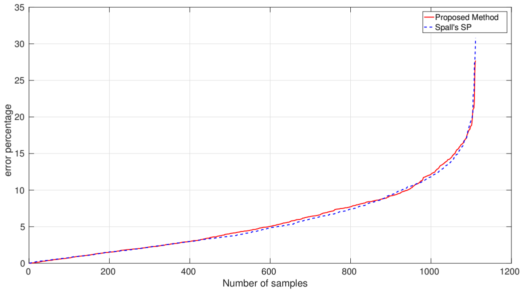


Figure 15: Sorted percentage error on test data.

4. Concluding remarks

We have proposed an estimation scheme for gradient in high dimensions that combines ideas from Spall's SPSA with compressive sensing and thereby tries to economize on the number of function evaluations. This has theoretical justification by the results of (Austin, 2016). Our method can be extremely useful when the function evaluation is very expensive, e.g., when a single evaluation is the output of a long simulation. This situation does not seem to have been addressed much in literature. In very high dimensional problems with sparse gradient, computing estimates for partial derivatives in every direction is inefficient because of the large number of function evaluations needed. SP simplifies the problem of repeated function evaluation by concentrating on a single *random* direction at each step. When the gradient vectors in such cases live in a lower dimensional subspace, it also makes sense to exploit ideas from compressive sensing. We have computed the error bound in this case and have also shown theoretically that this kind of estimation of gradient works well with high probability for the gradient descent problems and in other high dimensional problems such as estimating EGOP in manifold learning where gradients are actually low-dimensional and gradient estimation is relevant. Simulations show that our method works much better than pure SP.

Acknowledgments

We thank GPU Centre of Excellence, IIT Bombay for providing us with the facility to carry out simulations and Prof. Chandra Murthy of Indian Institute of Science for helpful discussions regarding compressive sensing.

References

T. Austin. On the failure of concentration for the ℓ_∞-ball. *Israel Journal of Mathematics* 211(1):221-238, 2016.

S. A. Bandeira, M. Fickus, G. D. Mixon, and P. Wong. The road to deterministic matrices with the restricted isometry property. *Journal of Fourier Analysis and Applications* 19(6):1123-1149, 2013.

M. Benaim, A dynamical system approach to stochastic approximation, *SIAM Journal of Control and Optimization* 34(2):437-472, 1996.

O. Brandiére and M. Duflo, Les algorithmes stochastiques contourment-ils les pièges?, *Annals de l'Institut Henri Poincaré* 32(3):395-427, 1996.

Vivek S. Borkar. *Stochastic Approximation: A Dynamical Systems Viewpoint.* Hindustan Book Agency, New Delhi, and Cambridge University Press, Cambridge, UK, 2008.

E. Candés and T. Tao. Near optimal signal recovery from random projections: universal encoding strategies? *IEEE Transactions on Information Theory*, 52(12):5406-5425, 2006.

E. Candés, J. Romberg J. and T. Tao. Stable signal recovery from incomplete and inaccurate measurements, *Comm. Pure Appl. Math.*, 59(8):1207-1223, 2006.

E. Candés, M. Rudelson, T. Tao and R. Vershynin. Error correction via linear programming. In *Proceedings of the 46th Annual IEEE Symposium on Foundations of Computer Science (FOCS)*, 295-308, 2005.

L. W. Chan, K. Charan, D. Takhar, K. F. Kelly, R. G. Baraniuk, G. Richard and D. M. Mittleman. A single-pixel terahertz imaging system based on compressed sensing. *Applied Physics Letters*, 93(12):121105-121105-3, 2008.

D. L. Donoho and Y. Tsaig. Fast solution of l_1-norm minimization problems when the solution may be sparse, *IEEE Transactions on Information Theory*, 54:4789-4812, 2008.

M. F. Duarte, M. A. Davenport, T. Dharmpal, J. N. Laska, T. Sun, K. F. Kelly and R. G. Baraniuk. Single-pixel imaging via compressive sampling. *IEEE Signal Processing Magazine*, 25(2):83-91, March 2008.

S. Foucart and H. Rauhut. *A Mathematical Introduction to Compressive Sensing*, Birkhäuser, New York, 2013.

I. Grondman, L. Busoniu, G. A. D. Lopes and R. Babuska. A Survey of actor-critic reinforcement learning: standard and natural policy gradients. IEEE Transactions on Systems, Man, and Cybernetics, Part C: Applications and Reviews, 42(6):1291-1307, Nov. 2012.

M. W. Hirsch. Convergent Activation Dynamics in Continuous Time Networks. *Neural Networks*, 2(5): 331-349, 1989.

D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black box functions. *Journal of Global Optimization*, 13(4):455-492, 1998.

G. Joseph and C. R. Murthy. A Non-iterative Online Bayesian Algorithm for the Recovery of Temporally Correlated Sparse Vectors. *IEEE Transactions on Signal Processing*, 65(20):5510-5525, 2017.

M. Kabanava and H. Rauhut. Analysis l_1-recovery with frames and gaussian measurements. *Acta Applicandae Mathematicae*, 140(1):173-195, 2015.

S. Kamal. On the convergence, lock-in probability, and sample complexity of stochastic approximation. *SIAM Journal on Control and Optimization*, 48(8):5178-5192, 2010.

K. C. Li. Sliced inverse regression for dimension reduction. *Journal of the American Statistical Association*, 86(414):316-327, 1991.

S. Mukherjee, Q. Wu, and D. X. Zhou. Learning gradients on manifolds. *Bernoulli*, 16(1):181-207, 2010.

Y. Nesterov. A method of solving a convex programming problem with convergence rate $O(1/k^2)$. *Soviet Mathematics Doklady*, 27(2):372-376, 1983.

P. Pandita, I. Bilionis and J. Panchal. Extending Expected Improvement for High-dimensional Stochastic Optimization of Expensive Black-Box Functions. *J*ournal of Mechanical Design 138(11):111412, 2016.

R. Pemantle. Nonconvergence to unstable points in urn models and stochastic approximation. *Annals of Probability*, 18(2):698-712, 1990.

S. Shan and G. Gary Wang. Survey of modeling and optimization strategies to solve high-dimensional design problems with computationally-expensive black-box functions. *Structural and Multidisciplinary Optimization*, 41(2), 219, 2010.

J. C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332-341, 1992.

R. S. Sutton, D. McAllester, S. Singh and Y. Mansour. Policy gradient methods for reinforcement learning with function approximation. *Advances in Neural Information Processing Systems*, 12:1057-1063, MIT Press, 2000.

V. B. Tadic and A. Doucet. Asymptotic bias of stochastic gradient search. *The Annals of Applied Probability* 27(6):3255-3304, 2017.

G. Thoppe and V. S. Borkar. A concentration bound for stochastic approximation via Alexeev's formula. *arXiv*:1506-08657v2 [math.OC], 2015.

J. Torres-Sospedra, R. Montoliu, A. Martnez-Us, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta. UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems. *Indoor Positioning and Indoor Navigation (IPIN), 2014 International Conference*, 261-270, 2014.

S. Trivedi, J. Wang, S. Kpotufe and G. Shakhnarovich. A consistent estimator of the expected gradient outerproduct. *Proceedings of the Thirtieth Conference on Uncertainty in Artificial Intelligence*:819-828, July 2014.

J. A. Tropp. User-friendly tail bounds for sums of random matrices. *Foundations of Computational Mathematics*, 12(4):389-434, 2012.

Q. Wu, J. Guinney, M. Maggioni and S. Mukherjee. Learning gradients : predictive models that infer geometry and statistical dependence. *Journal of Machine Learning Research*, 11(1922):2175-2198, 2010.

Y. Xia, H. Tong, W. Li, and L.-X. Zhu. An adaptive estimation of dimension reduction space. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 64(3):363-410, 2002.

H. Xu, C. Caramanis and S. Mannor. Statistical optimization in high dimensions. *Operations research*, 64(4):958-979, 2016.

A. Yang, A. Ganesh, S. Sastry, and Y. Ma. Fast l_1-minimization algorithms and an application in robust face recognition: a review. *ICIP*, 2010.

T. Zhao, H. Hachiya, G. Niu and M. Sugiyama. Analysis and improvement of policy gradient estimation. *Neural Networks*, 26:118-129, 2012.