Available online at www.sciencedirect.com

ScienceDirect

IFAC Papers Online CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 50-1 (2017) 3899–3904

Projective Approximation Based Gradient Descent Modification

Alexander Senov * ** , **Oleg Granichin** * ** ***

* *Saint Petersburg State University (Faculty of Mathematics and Mechanics), 7-9, Universitetskaya Nab., St. Petersburg, 199034, Russia (*alexander.senov@gmail.com, o.granichin@spbu.ru*)*

** *Institute of Problems in Mechanical Engineering, Russian Academy of Sciences (IPME RAS), 61, Bolshoy pr., St. Petersburg, Russia*

*** *ITMO University, St. Petersburg, Russia*

Abstract: We present a new modification of the gradient descent algorithm based on the surrogate optimization with projection into low-dimensional space. It iteratively approximates the target function in low-dimensional space and takes the approximation optimum point mapped back to original parameter space as next parameter estimate. Main contribution of the proposed method is in application of projection idea in approximation process. Major advantage of the proposed modification is that it does not change the gradient descent iterations, thus it can be used with some other variants of the gradient descent. We give a theoretical motivation for the proposed algorithm and a theoretical lower bound for its accuracy. Finally, we experimentally study its properties on modelled data.

Keywords: Mathematical programming, Parameter estimation, Steepest descent, Least-squares, Function approximation, Convex optimization, Model approximation, Iterative methods, Quadratic programming, Projective methods.

1. INTRODUCTION

Many science, engineering and control problems could be represented as an optimization (a.k.a. mathematical optimization, mathematical programming) problems. Particularly, both machine learning and optimization became very popular and widely used in today's control applications (see, for example Boyd (2012); Granichin et al. (2015)). At the same time, most part of the machine learning problems can be translated to the optimization task of real-valued parametrized function known as *loss function* Bottou (1998). Moreover, optimization plays a crucial role in deep learning, particularly in image-related problems (e.g., image classification Hinton and Salakhutdinov (2006), image recognition Dong et al. (2016)). This is due to the fact that almost every process is essentially about maximising or minimising some quantity: maximising revenue, accuracy, efficiency, performance or minimising errors, expenses, downtime. Most part of optimization problems can be formulated as follows

$$f \to \min_{\boldsymbol{x}\in\mathcal{X}},$$

where $f : \mathbb{X} \to \mathbb{R}$ and $\mathcal{X} \subset \mathbb{X}$.

One of the most extensively studied sub-fields of optimization is convex programming which study convex functions optimization under the most convex constraints (see Boyd and Vandenberghe (2004) for an extensive overview). Assuming function convexity and smoothness we can choose amongst many first-order iterative optimization algorithms, e.g. *gradient descent* algorithm also known as *steepest descent* originally proposed in Cauchy (1847). The general idea of first-order iterative optimization algorithms is to gradually improve current function optimum point estimate by moving it in the direction opposite to the function gradient. The first-order iterative optimization methods are especially important in problems with high-dimensional parameter space, where high-order methods are inapplicable due to the curse of dimensionality, e.g., such problems as object detection and image super-resolution might have millions of parameters (e.g, see Hinton and Salakhutdinov (2006); Dong et al. (2016)).

Plenty of various modifications and extensions of the gradient descent algorithm exists. Some algorithms exploit specific function properties to improve convergence. E.g., for *separable* functions it is possible to reduce total number of evaluations (see Davidon (1976); Boyd et al. (2011)). Learning rate adaptation is another popular direction for modifications (see, e.g., Duchi et al. (2011); Kingma and Ba (2014)). Momentum methods is another approach that reduce estimates oscillation effect by accumulating so-called "inertion" of the estimates (see Qian (1999); Nesterov (1983)) . However most modifications share one but yet important drawback: at each iteration they explicitly consider only current and few previous points and function values ignoring preceding history, thus, loosing potentially important information.

⋆ This work was supported by Russian Foundation for Basic Research, projects number 17–51–53053 and 16–01–00759, and by the H&W program of the University of Brescia under the project "Classificazione della fibrillazione ventricolare a supporto della decisione terapeutica — CLAFITE".

10.1016/j.ifacol.2017.08.362

In contrast, another type of optimization algorithms — *surrogate methods* — do use history of points and function values. Surrogate optimization methods approximating the objective function by so-called *surrogate* function on the basis of set of points and corresponding function values and takes optimum estimate based on obtained surrogate.

Most types of surrogate optimization methods consequently improve function approximation by taking each next optimum estimate into account in iterative fashion. General surrogate-based optimization model consist of the following steps: design of experiment, sampling new observations, building surrogate model, evaluating surrogate model and returning back to design of experiments (see Forrester and Keane (2009)). Among many surrogate models we can mention several most common ones: radial basis functions (Broomhead and Lowe (1988)), krigging (Krige (1951)), support vector regression (Vapnik (1995)) and response surface methodology (Box et al. (1987)).

Despite many advantages, most surrogate models share common drawbacks: they are memory and time consuming and, what is most important, their quality depends on the chosen surrogate model adequacy with respect to original function (see, e.g., Forrester and Keane (2009)). As example, it is impossible to approximate high-order polynomial with linear surrogate function.

In this paper we propose a new method which is essentially an incorporation of quadratic response surface methodology into the gradient descent algorithm. To neutralize memory footprint of quadratic polynomial we use the projection trick. The general idea of the proposed algorithm is to use a sequence of points obtained from gradient descent iterations as follows:

1. q points used to construct orthogonal *projection matrix* $\mathbf{P}$ using Gram-Schmidt orthogonalization.
2. K points used to collect training set in *projected space* obtained by multiplication on the projection matrix $\mathbf{P}$.
3. Quadratic polynomial $\widehat{g}$ fitted to collected training set and its argmin returned back to original space used as next estimate of argmin f.

These steps are executed in loop producing next estimate from which gradient descent proceeds every $q + K$ iterations. One of the main advantages of the proposed method is that it can be used with almost any gradient descent modification mentioned above.

The contribution of this paper in general is in application of the projective approximation idea in optimization. Most surrogate optimization methods use the approximation idea without projection. Additionally, there are few works regarding the projection idea: in Krause (2010) authors proposed SFO method which keeps gradient and estimates vectors not in original but projects it into low-dimensional space. Compressed sensing is another powerful technique for a signal recovery solely based on the projection idea (see Donoho (2006)). As far as we know, there is currently no work on applying the projective approximation idea for improving zero- or first-order iterative optimization methods convergence.

The paper is organized as follows. In Section 2 we cover some basics and previous results required for further explanation and formulating the problem statement. In Section 3 we propose a hybrid algorithm which improves the gradient descent by use of the projective quadratic response surface methodology and provide theoretical motivation behind it. Further, in Section 4 we perform an experiment on modelled data and discuss its results. Finally, Section 5 the conclusion is given.

2. PRELIMINARIES & PROBLEM STATEMENT

In this Section we briefly introduce some methods and concepts necessary for further explanation and ending it with the problem statement.

2.1 Notation remarks

We use small light symbols x for scalars and indexes (mainly, small bold symbols $\boldsymbol{x}$ for vectors, capital light symbols X for constants and sets (except parameter matrix Θ), capital bold symbols $\mathbf{X}$ for matrices. Specifically we denote t as iteration index, T as total number of iterations, d as the original space dimensionality, q as dimensionality of the projected space, $\mathbf{P} \in \mathbb{R}^{q \times d}$ as orthogonal projection matrix, $\mathbf{P}^{-1}\boldsymbol{z}$ as a set of points $\{\boldsymbol{x} \in \mathbb{R}^d : \boldsymbol{z} = \mathbf{P}\boldsymbol{x}\}$, $\mathbf{I}$ as identity matrix (its size follows from the context), $^\top$ as transpose sign, $\widehat{}$ as estimate sign, f as objective function and $\nabla_{\boldsymbol{x}} f$ as function gradient with respect to parameter vector $\boldsymbol{x}$.

2.2 Quadratic response surface methodology

Quadratic response surface methodology (QRSM) is a surrogate optimization method where surrogate is a 2nd order polynomial constructed via polynomial regression. It approximates a set of points and corresponding objective function values using polynomial least squares technique. Then, argmin of obtained 2nd order polynomial is used as next objective function optimum estimate. Figure 1 illustrates the idea.

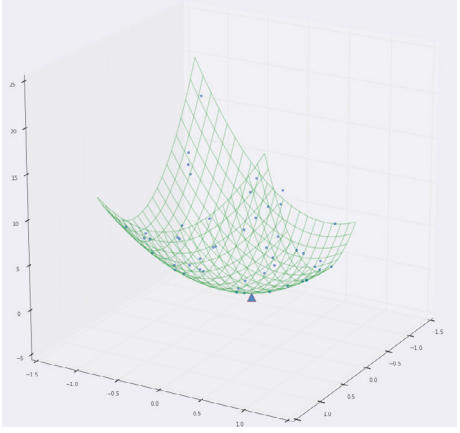

Fig. 1. Quadratic response surface example. Blue circles represents the set of points. Green gird represent quadratic polynomial approximates these points. Blue triangle represent the argmin of constructed polynomial.

An important drawback of the quadratic response surface methodology is infeasibility in high-dimensional problems (see, e.g., Box et al. (1987)): straightforward second order polynomial reconstruction requires $\mathcal{O}\left(Nd^2 + d^4\right)$ in memory and $\mathcal{O}\left(d^6 + d^4N\right)$ in time. This is clearly inappropriate in case of big d.

2.3 Gradient descent

Algorithm 1 describes the simple variant of the gradient descent. We use the most primitive setup: static learning rate and upper bound of total number of iterations used as a stop rule, since it helps us to simplify the description. Moreover, the chosen stop rule might be the best one if number of function and/or gradient evaluations is limited. For choice of the stop rule and learning rate strategy we refer to Polyak (1987) and Bertsekas (1999).

Algorithm 1 Gradient Descent $(f, \boldsymbol{x}_0, \lambda, T)$

for $t \leftarrow 1$ to T **do**
  $\boldsymbol{x}_t \leftarrow \boldsymbol{x}_{t-1} - \lambda \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_{t-1})$
end for
return $\boldsymbol{x}_T$

Despite its simplicity, gradient descent algorithm suffers from variety of drawbacks. On of the main problem relates to the learning rate parameter: if it is chosen to be too big the estimate will dangle around the optimal point, not approaching it. On the other hand, if it is too small, convergence might take vast amount of iterations and we won't approach optimum for T iterations. Second problem relates to the second parameter — initial estimate $\boldsymbol{x}_0$: if it chosen far away from the optimum the path would be very long and we will need a lot of iterations to converge.

In this paper we address the first two problems. Both of them can be reformulated as one: how to improve the convergence of gradient descent algorithm.

2.4 Problem statement

Assume that starting point choice was unsuccessful and/or learning rate is to small. This will result in large number of iterations. The problem statement is:
how to reduce total number of gradient descent iterations without modification of learning rate and gradient update policy at each gradient descent step?

3. ALGORITHM

Any variation of the gradient descent algorithm generates a sequence of optimum point estimates $\{\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_n, \ldots\}$ that can converge to local or even global minimum of f. At the same time, most iterative optimization algorithms use only few previous points at each step explicitly, thus, loosing potentially important information.

One can try to fix it by retaining last K points in addition to the current one (e.g. points $\boldsymbol{x}_{t-K}, \ldots, \boldsymbol{x}_t$ at iteration number t) and use them to obtain next estimate through some estimator G:

$$\boldsymbol{x}_{t+1} \leftarrow G(f, \{\boldsymbol{x}_{t-K}, \ldots, \boldsymbol{x}_t\})$$

However, if $\boldsymbol{x}$ is high-dimensional, then maintaining $K+1$ points would require $\mathcal{O}(dK)$ memory, which might be critical in some situations. This is even more important if G produce estimate using quadratic response surface methodology, hence amount of required memory will raise to $\mathcal{O}\left(d^2K\right)$.

In this paper we suggest an algorithm which use the same strategy of retaining last K points, but projected in low-dimensional space, thus significantly reducing memory footprint.

3.1 Algorithm description

Algorithm 2 Proposed gradient descent modification

Require:
f — function to be optimized
$\boldsymbol{x}_0$ — initial parameter value
λ — step size
T — number of iterations
Additional parameters:
q — projected space dimensionality
K — number of points for surrogate construction

$t \leftarrow 1$
while $t \leq T$ **do**
  $\mathbf{P} \leftarrow \boldsymbol{x}_t^\top$ ▷ projection matrix initialization
  for $k \leftarrow 1$ to $q-1$ **do**
    $\boldsymbol{x}_t \leftarrow \boldsymbol{x}_{t-1} - \lambda \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_{t-1})$
    $\mathbf{P} \leftarrow \left[\mathbf{P}^\top; \boldsymbol{x}_t\right]^\top$
    $t \leftarrow t+1$
  end for
  $\mathbf{P} \leftarrow$ Gram-Schmidt $(\mathbf{P}_{1,\cdot}, \ldots, \mathbf{P}_{q,\cdot})$ ▷ orthogonalization
  $\overline{\boldsymbol{x}} \leftarrow 0_d$
  for $k \leftarrow 1$ to K **do**
    $\boldsymbol{x}_t \leftarrow \boldsymbol{x}_{t-1} - \lambda \nabla_{\boldsymbol{x}} f(\boldsymbol{x}_{t-1})$
    $\overline{\boldsymbol{x}} \leftarrow \overline{\boldsymbol{x}} + \boldsymbol{x}_t$
    $\boldsymbol{z}_k \leftarrow \mathbf{P}\boldsymbol{x}_t$
    $y_k \leftarrow f(\boldsymbol{x}_t)$
    $t \leftarrow t+1$
  end for
  $\overline{\boldsymbol{x}} \leftarrow \overline{\boldsymbol{x}}/K$
  $\widehat{\mathbf{A}}, \widehat{\boldsymbol{b}}, \widehat{c} \leftarrow$ QuadraticLeastSquares $\left(\{\boldsymbol{z}_k, y_k\}_1^K\right)$
  if $\widehat{\mathbf{A}}$ positive definite **then**
    $\widehat{\boldsymbol{z}} \leftarrow -\frac{1}{2}\widehat{\mathbf{A}}^{-1}\widehat{\boldsymbol{b}}$
    $\boldsymbol{x}_t \leftarrow -\frac{1}{2}\mathbf{P}^\top\widehat{\boldsymbol{z}} + (\mathbf{I} - \mathbf{P}^\top\mathbf{P})\overline{\boldsymbol{x}}$ ▷ "backward" projection
  end if
end while

See Algorithm 2 for pseudo-code of our proposed algorithm. It modifies a gradient descent algorithm by adding a projective quadratic response surface methodology procedure on top of gradient descent iterations. Keeping in mind that gradient descent iterations produce sequence of optimum point estimates we now describe proposed modification in details.

(0) Initially, we run a gradient descent algorithm and start obtaining optimum point estimates $\boldsymbol{x}_t$, forming a sequence $\boldsymbol{x}_0, \boldsymbol{x}_1, \ldots, \boldsymbol{x}_t, \ldots$. Assume that we have just obtained a point $\boldsymbol{x}_{t+1}$.
(1) First, q points from $\boldsymbol{x}_{t+1}$ to $\boldsymbol{x}_{t+q}$ are used to construct projection matrix $\mathbf{P} \in \mathbb{R}^{q \times p}$ using Gram-Schmidt

orthogonalization.
Update $t \leftarrow t + q$.

(2) Next, K points together with corresponding function values used to collect an average $\overline{\boldsymbol{x}} = \frac{1}{K}\sum_1^K \boldsymbol{x}_{t+k}$ and a training set in projected space: $\{\boldsymbol{z}_k, y_k\}_{k=1}^K$, where $\boldsymbol{z}_k = \mathbf{P}\boldsymbol{x}_{t+k}$ and $y_k = f(\boldsymbol{x}_{t+k})$.
Update $t \leftarrow t + K$.

(3) A quadratic surrogate function fitted to the training set in projected space $\widehat{g}(z) = \boldsymbol{z}^\top \widehat{\mathbf{A}}\boldsymbol{z} + \widehat{\boldsymbol{b}}^\top \boldsymbol{z} + \widehat{c}$ using quadratic least squares estimate.

(4) If matrix $\widehat{\mathbf{A}}$ turns to be positive definite, do
   (a) Calculate surrogate minimum in projected space: $\widehat{\boldsymbol{z}} \leftarrow \operatorname{argmin} \widehat{g}$.
   (b) Return the surrogate minimum back to original space: $\widehat{\boldsymbol{x}} \leftarrow \mathbf{P}^\top \widehat{\boldsymbol{z}} + \overline{\boldsymbol{x}}$.
   (c) Use $\widehat{\boldsymbol{x}}$ as next gradient descent optimum point estimate.

(5) Go back to (1)

One can be confused by "backward projection" step of transforming $\widehat{\boldsymbol{z}}$ from low-dimensional space to $\widehat{\boldsymbol{x}}$ in high-dimensional space (line 28 in Algorithm 2). This transform motivated by Proposition 1. As for the choice of the consecutive points orthogonalization as a tool for the orthogonal projection construction, this choice is rather motivated by intuition and practice.

It is worth noting, that described modification does not violate gradient descent iterations, but produces an additional optimum point estimate every qK iterations (actually, it may fail, since matrix $\widehat{A}$ might be not a positive definite one). Accordingly, its not a modification but more a complement on top of gradient descent. Thereby, gradient descent steps might be replaced by any variation of it: stochastic gradient descent, AdaGrad, momentum method, etc.

As one can see, this modification utilize:

- $\mathcal{O}\left(q^2 d\right)$ operations and $\mathcal{O}\left(qd\right)$ memory at step (1),
- $\mathcal{O}\left(Kqd\right)$ operations and $\mathcal{O}\left(Kq\right)$ memory at step (2),
- $\mathcal{O}\left(Kq^2 + q^3\right)$ operations and $\mathcal{O}\left(Kq + q^2\right)$ memory at step (3).

Hence, maximum addition per single gradient descent iteration with the modification described above is $\mathcal{O}\left(qd + q^3\right)$ in number of operations and $\mathcal{O}\left(q^2 + qd + Kq\right)$ in memory consumption.

3.2 Theoretical ground

In this Section we give several results regarding motivation of the proposed algorithm and its quality.

Lets start with abstracting from particular problem. Consider an orthogonal projection matrix $\mathbf{P} \in \mathbb{R}^{q\times d}$ and a sequence of points $\{\boldsymbol{x}_1, \dots, \boldsymbol{x}_K\} \subset \mathbb{R}^d$ representing some kind of "trace" (e.g. a gradient descent estimates) together with corresponding function values $\{y_t\}_1^K$, where $y_t = f(\boldsymbol{x}_t)$. Applying projection we obtain points images in low-dimensional space: $\{\boldsymbol{z}_1, \dots, \boldsymbol{z}_K\}$, $\boldsymbol{z}_t = \mathbf{P}\boldsymbol{x}_t$. Then, we obtain a continuation of this sequence $\widehat{\boldsymbol{z}}$ using quadratic least squares. Two questions arises here.

1 How to perform "backward projection" of the estimate $\widehat{\boldsymbol{z}}$ from the low-dimensional space back to the high-dimensional space $\mathbb{R}^d$?
2 How projection procedure (both forward & backward) affects optimum point estimate accuracy?

The answer on the first question is given by Propositions 1 and 2. The second question is answered by Theorem 3.

If the matrix $\mathbf{P}$ is invertible, we may simply set $\widehat{\boldsymbol{x}} = \mathbf{P}^{-1}\widehat{\boldsymbol{z}}$. Unfortunately, it isn't: any point $\boldsymbol{z} \in \mathbb{R}^q$ corresponds to the entire set $\{\boldsymbol{x} \in \mathbb{R}^d : \mathbf{P}\boldsymbol{x} = \boldsymbol{z}\}$. Hence we need to impose additional restrictions to pick some specific point from this set. Since we extending original sequence $\{\boldsymbol{x}_t\}_1^{K-1}$ the following idea sounds reasonable: lets pick a point closest to the original sequence in terms of euclidean distance:

$$\widehat{\boldsymbol{x}} \leftarrow \operatorname*{argmin}_{\{\boldsymbol{x}\in\mathbb{R}^d\,:\,\mathbf{P}\boldsymbol{x}=\widehat{\boldsymbol{z}}\}} \sum_{t=1}^{K} \|\boldsymbol{x}_t - \boldsymbol{x}\|_2^2 . \tag{1}$$

Proposition 1. In the notation described above, $\widehat{\boldsymbol{x}}$ from (1) can be explicitly expressed as:

$$\widehat{\boldsymbol{x}} = \left(\mathbf{I} - \mathbf{P}^\top\mathbf{P}\right)\frac{1}{K}\sum_{t=1}^{K}\boldsymbol{x}_t + \mathbf{P}^\top\widehat{\boldsymbol{z}}.$$

See the Proposition proof in appendix A. This Proposition provides a recipe how to choose an estimate $\widehat{\boldsymbol{x}}$ in original space $\mathbb{R}^d$ from set $\mathbf{P}^{-1}\widehat{\boldsymbol{z}}$ if low-dimensional estimate $\widehat{\boldsymbol{z}}$ is already known.

Further, to measure the effect that projection procedure have on the optimum point estimation accuracy we need a ground true answer to compare with. Assume that function f is itself a convex polynomial of degree two: $f(\boldsymbol{x}) := \boldsymbol{x}^\top\mathbf{A}\boldsymbol{x} + \boldsymbol{b}^\top\boldsymbol{x} + c$. Therefore, its argmin is $-\frac{1}{2}\mathbf{A}^{-1}\boldsymbol{b}$. Despite the fact, that this case is trivial, it serves as a good example and function approximation in small neighbourhood of particular point. Keeping in mind that $\boldsymbol{x}$ can be represented as a sum $\mathbf{P}^\top\mathbf{P}\boldsymbol{x} + (\mathbf{I} - \mathbf{P}^\top\mathbf{P})\boldsymbol{x}$, and function f can be considered as a function of $\boldsymbol{z} = \mathbf{P}\boldsymbol{x}$, the following Proposition gives explicit expression for its argmin in terms of z.

Proposition 2. Consider function $f(\boldsymbol{x}) = \boldsymbol{x}^\top\mathbf{A}\boldsymbol{x} + \boldsymbol{b}^\top\boldsymbol{x} + c$, where $\mathbf{A} \in \mathbb{R}^{d\times d}$, $\mathbf{A} \succ 0$, $\boldsymbol{b} \in \mathbb{R}^d$, $c \in \mathbb{R}$ and orthogonal projection matrix $\mathbf{P} \in \mathbb{R}^{q\times d}$, $q < d$. Define $\boldsymbol{z} = \mathbf{P}\boldsymbol{x}$, $\boldsymbol{z} = (\mathbf{I} - \mathbf{P}^\top\mathbf{P})\boldsymbol{x}$ and $g(\boldsymbol{z} \mid \boldsymbol{v}) = f(\boldsymbol{x}) = f(\mathbf{P}^\top\boldsymbol{z} + \boldsymbol{v})$.

Then, the following holds true:

- $\operatorname*{argmin}_{\boldsymbol{z}} g(\boldsymbol{z} \mid \boldsymbol{v})$ does not depend on v;
- $\operatorname*{argmin}_{\boldsymbol{z}} g(\boldsymbol{z} \mid \boldsymbol{v}) = -\frac{1}{2}\mathbf{P}\mathbf{A}^{-1}\boldsymbol{b}$;

See the Proposition proof in appendix A

Finally, the following Theorem provides the main theoretical result of the paper: an estimate of the difference between the true argmin of f and its estimate obtained with the proposed projection procedure (single loop of an Algorithm 2, lines 9–30).

Theorem 3. Consider function $f(\boldsymbol{x}) = \boldsymbol{x}^\top\mathbf{A}\boldsymbol{x} + \boldsymbol{b}^\top\boldsymbol{x} + c$, where $\mathbf{A} \in \mathbb{R}^{d\times d}$, $\boldsymbol{b} \in \mathbb{R}^d$, $c \in \mathbb{R}$ and $\mathbf{A} \succ 0$, orthogonal projection matrix $\mathbf{P} \in \mathbb{R}^{q\times d}$, $q < d$, sequence of gradient descent estimates $\{\boldsymbol{x}_t\}_1^K \subset \mathbb{R}^d$, their projections $\{\boldsymbol{z}_t\}_1^K \subset$

$\mathbb{R}^q$, $\boldsymbol{z}_t = \mathbf{P}\boldsymbol{x}_t$ and corresponding function values $\{y_t\}_1^K$, $y_t = f(\boldsymbol{x}_t)$.
Then, the difference between argmin f and its estimate $\widehat{\boldsymbol{x}}$ obtained via single loop of the proposed Algorithm 2 is equal to:

$$\|\text{argmin}\, f - \widehat{\boldsymbol{x}}\|_2^2 = \left\|(\mathbf{I} - \mathbf{P}^\top\mathbf{P})\left(\frac{1}{2}\mathbf{A}^{-1}\boldsymbol{b} - \overline{\boldsymbol{x}}\right)\right\|_2^2.$$

See the Theorem proof in appendix A

This Theorem implies the following important facts.

1 The difference between the optimum point and obtained estimate lies in kernel of orthogonal projection $\mathbf{P}$. Hence, estimate $\mathbf{P}\widehat{\boldsymbol{x}}$ is a best estimate in terms of this projection.
2 The closer the averaged gradient descent estimates to the optimum points, the smaller the error. Hence, we $\widehat{\boldsymbol{x}}$ benefits from precision of gradient descent estimates.

4. MODELLING

This Section contains experiments results and analysis. First of all, we describe the modelling strategy.

For modelling purposes we use the following defaults:

- $f(\boldsymbol{x}) = \boldsymbol{x}^\top\mathbf{I}_d - \mathbf{1}_d^T\boldsymbol{x}$, where d is a variable parameter;
- $d \leftarrow 10$, $T \leftarrow 50$, $x_0 \sim \mathrm{U}\,[0,1]^d$, $\lambda = 10^{-5}$;
- $q \leftarrow 1$, $q \leftarrow 10$, $K \leftarrow 5$.

We vary parameters T, d, q and K independently (with other parameters fixed) in following ranges:

- $T \in [25, 30, 50, 100]$;
- $d \in [5, 10, 20, 50, 100]$;
- $K \in [2, 5, 10, 20]$.

For every parameters values combination we execute Algorithms 1 and 2 with same initial estimate x_0 for 10^3 times and calculate their errors as euclidean distance from real optimum to point to algorithm estimate. Then we calculate simple statistics ξ — a ratio of times when the modified gradient descent error was less than error of the original gradient descent.

Table 1. Modelling results

Parameter name	Parameter value	ξ
T	25	0.90
T	30	0.89
T	50	0.81
T	100	0.53
d	5	0.82
d	10	0.80
d	20	0.81
d	50	0.79
d	100	0.81
K	2	0.11
K	3	0.81
K	5	0.82
K	10	0.79

Table 1 contains results of the experiment described above. We will discuss them per parameter varied.

- Results on the parameter T may seem confusing: ξ monotonically decreases with T increased. It can be explained by the fact that the proposed gradient descent modification doing a good job at start (when gradient descent steps are large and function is steep), but it fails to build a surrogate model when gradient descent oscillates near the optimum point.
- Situation with the parameter q is even more surprising: there no strict dependency. This might be explained by the fact that in case of the particular function f, gradient descent estimates lie on a straight line. Thereby, they are perfectly described by a single dimension.
- K is another parameter which results may seem confusing at first, but then become quite obvious: huge difference in quality between $K = 2$ and $K = 3$ is explained by the fact that one need at least three points in one-dimensional space to construct the second order polynomial.

5. CONCLUSION

We propose a novel modification of the gradient descent based on the quadratic response surface methodology with the projection trick. We provide few theoretical results regarding the proposed modification and perform an experimental study of it. We show that the proposed modification provides the best optimum approximation with respect to given projection matrix. Experiments on toy example demonstrates that the modified gradient descent might be superior to the original one in terms of the the optimum point estimation error. This modification is potentially rather important, because it may be used with almost every gradient descent variation . Further developments may include approbation of the proposed modification in real world problems, its adaptation to other iterative optimization algorithms and study of the projective response surface methodology applicability to other optimization problems. An application of the proposed modification on top of the stochastic approximation algorithms (see Granichin and Polyak (2003)) in context of object detection and image super-resolution problems seems to be the most promising one.

REFERENCES

Bertsekas, D.P. (1999). *Nonlinear programming*. Athena scientific Belmont.

Bottou, L. (1998). Online learning and stochastic approximations. *On-line learning in neural networks*, 17(9), 142.

Box, G.E., Draper, N.R., et al. (1987). *Empirical model-building and response surfaces*. John Wiley & Sons, New York.

Boyd, S. (2012). Global optimization in control system analysis and design. *Control and Dynamic Systems V53: High Performance Systems Techniques and Applications: Advances in Theory and Applications*, 53, 1.

Boyd, S., Parikh, N., Chu, E., Peleato, B., and Eckstein, J. (2011). Distributed optimization and statistical learning via the alternating direction method of multipliers. *Foundations and Trends® in Machine Learning*, 3(1), 1–122.

Boyd, S. and Vandenberghe, L. (2004). *Convex Optimization*. Cambridge university press.

Broomhead, D. and Lowe, D. (1988). Radial basis functions, multi-variable functional interpolation and adaptive networks. *Complex Systems*, 2, 321–355.
Cauchy, A. (1847). Méthode générale pour la résolution des systemes déquations simultanées. *Comp. Rend. Sci. Paris*, 25(1847), 536–538.
Davidon, W.C. (1976). New least-square algorithms. *Journal of Optimization Theory and Applications*, 18(2), 187–197.
Dong, C., Loy, C.C., He, K., and Tang, X. (2016). Image super-resolution using deep convolutional networks. *IEEE transactions on pattern analysis and machine intelligence*, 38(2), 295–307.
Donoho, D.L. (2006). Compressed sensing. *IEEE Transactions on information theory*, 52(4), 1289–1306.
Duchi, J., Hazan, E., and Singer, Y. (2011). Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research*, 12, 2121–2159.
Forrester, A. and Keane, A. (2009). Recent advances in surrogate-based optimization. *Progress in Aerospace Sciences*, 45(1), 50–79.
Granichin, O., Volkovich, V., and Toledano-Kitai, D. (2015). *Randomized Algorithms in Automatic Control and Data Mining*. Springer.
Granichin, O. and Polyak, B. (2003). *Randomized Algorithms of Estimation and Optimization Under Almost Arbitrary Noise*. M.: Nauka.
Hinton, G.E. and Salakhutdinov, R.R. (2006). Reducing the dimensionality of data with neural networks. *science*, 313(5786), 504–507.
Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.
Krause, A. (2010). Sfo: A toolbox for submodular function optimization. *Journal of Machine Learning Research*, 11(Mar), 1141–1144.
Krige, D. (1951). A statistical approach to some basic mine valuation problems on the witwatersrand. *Journal of Chemical, Metallurgical, and Mining Society of South Africa*, 52(6), 119–139.
Nesterov, Y. (1983). A method for unconstrained convex minimization problem with the rate of convergence o(1/k2). In *Doklady an SSSR*, volume 269, 543–547.
Polyak, B.T. (1987). Introduction to optimization. translations series in mathematics and engineering. *Optimization Software*.
Qian, N. (1999). On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1), 145–151.
Vapnik, V.N. (1995). The nature of statistical learning theory.

Appendix A. PROOFS

Proof. (of Proposition 1)
Since matrix $(\mathbf{I}-\mathbf{P}^\top\mathbf{P})$ projects $\mathbb{R}^d$ into null space of $\mathbf{P}$, (1) can be reformulated as follows:

$$\widehat{\boldsymbol{x}} \leftarrow \underset{\boldsymbol{x}\in\mathbb{R}^d}{\operatorname{argmin}} \sum_{t=1}^{K} \left\|\boldsymbol{x}_t - \left(\widehat{\boldsymbol{z}} + \left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\boldsymbol{x}\right)\right\|_2^2.$$

Differentiating the sum above with respect to $\boldsymbol{x}$ we obtain:

$$\begin{aligned}
&\partial_{\boldsymbol{x}} \sum_{t=1}^{K} \left\|\boldsymbol{x}_t - \left(\widehat{\boldsymbol{z}} + \left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\boldsymbol{x}\right)\right\|_2^2 \\
&= \sum_{t=1}^{K} 2(\mathbf{I}-\mathbf{P}^\top\mathbf{P})\left(\boldsymbol{x}_t - \left(\widehat{\boldsymbol{z}} + \left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\boldsymbol{x}\right)\right) \\
&= \sum_{t=1}^{K} 2(\mathbf{I}-\mathbf{P}^\top\mathbf{P})\left((\mathbf{I}-\mathbf{P}^\top\mathbf{P})\boldsymbol{x}_t - \left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\boldsymbol{x}\right) \\
&= 2(\mathbf{I}-\mathbf{P}^\top\mathbf{P})\sum_{t=1}^{K}\boldsymbol{x}_t - 2K\left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\boldsymbol{x}
\end{aligned}$$

Equating the derivative to zero we obtain:

$$(\mathbf{I}-\mathbf{P}^\top\mathbf{P})\boldsymbol{x} = (\mathbf{I}-\mathbf{P}^\top\mathbf{P})\frac{1}{K}\sum_{1}^{K}\boldsymbol{x}_t.$$

Since $(\mathbf{I}-\mathbf{P}^\top\mathbf{P})$ is not invertible the above equation has infinite number of solutions. Hence, we are free to choose any one of them, e.g. $\boldsymbol{x} = \frac{1}{K}\sum_1^K \boldsymbol{x}_t$.

□

Proof. (of Proposition 2)

$$\begin{aligned}
f(\boldsymbol{x}) &= \left(\mathbf{P}^\top\boldsymbol{z}+\boldsymbol{v}\right)^\top\mathbf{A}\left(\mathbf{P}^\top\boldsymbol{z}+\boldsymbol{v}\right) + \boldsymbol{b}^\top\left(\mathbf{P}^\top\boldsymbol{z}+\boldsymbol{v}\right) + c \\
&= \boldsymbol{z}^\top\mathbf{P}\mathbf{A}\mathbf{P}^\top\boldsymbol{z} + \boldsymbol{b}^\top\boldsymbol{z} + \boldsymbol{v}^\top\mathbf{A}\mathbf{P}^\top\boldsymbol{z} - \boldsymbol{b}^\top\boldsymbol{v} + \boldsymbol{v}^\top\mathbf{A}\boldsymbol{v} + c \\
&= \boldsymbol{z}^\top\mathbf{P}\mathbf{A}\mathbf{P}^\top\boldsymbol{z} + \left(\boldsymbol{b}^\top + \boldsymbol{v}^\top\mathbf{A}\right)\mathbf{P}^\top\boldsymbol{z} \\
&\quad + \left(\boldsymbol{v}^\top\mathbf{A}\boldsymbol{v} - \boldsymbol{b}^\top\boldsymbol{v} + c\right)
\end{aligned}$$

Substituting $\boldsymbol{v}$ back and taking derivative with respect to $\boldsymbol{z}$, we've got:

$$\begin{aligned}
0 = \;& 2\mathbf{P}\mathbf{A}\mathbf{P}^\top\widehat{\boldsymbol{z}} + \left(\boldsymbol{b}^\top + \boldsymbol{x}^\top\left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)^\top\mathbf{A}\right)\mathbf{P}^\top \\
\rightsquigarrow \widehat{\boldsymbol{z}} &= -\frac{1}{2}\left(\mathbf{P}\mathbf{A}\mathbf{P}^\top\right)^{-1}\left(\left(\boldsymbol{b}^\top + \boldsymbol{x}^\top\left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)^\top\mathbf{A}\right)\mathbf{P}^\top\right)^\top \\
&= -\frac{1}{2}\mathbf{P}\mathbf{A}^{-1}(\mathbf{P}^\top\mathbf{P})^{-1}\mathbf{P}^\top\mathbf{P}\left(\boldsymbol{b} + \mathbf{A}\left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\boldsymbol{x}\right) \\
&= -\frac{1}{2}\mathbf{P}\left(\mathbf{A}^{-1}\boldsymbol{b} + \left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\boldsymbol{x}\right) \\
&= -\frac{1}{2}\mathbf{P}\mathbf{A}^{-1}\boldsymbol{b}
\end{aligned}$$

□

Proof. (of Theorem 3)
As mentioned above, $\operatorname{argmin} f = -\frac{1}{2}\mathbf{A}^{-1}\boldsymbol{b}$. From Proposition 2 we know that $\widehat{\boldsymbol{z}}$ obtained from at line 27 of Algorithm 2 equals to $-\frac{1}{2}\mathbf{P}\mathbf{A}^{-1}\boldsymbol{b}$. Moreover, using the result of Proposition 1 we now that the best $\boldsymbol{x}$ in $\mathbf{P}^{-1}\widehat{z} = \left\{\mathbf{P}^\top\widehat{z} + (\mathbf{I}-\mathbf{P}^\top\mathbf{P})\boldsymbol{x} : \boldsymbol{x}\in\mathbb{R}^d\right\}$ is $\boldsymbol{x} = \overline{\boldsymbol{x}}$.

Hence $\widehat{\boldsymbol{x}} = \left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\overline{\boldsymbol{x}} - \frac{1}{2}\mathbf{P}^\top\mathbf{P}\mathbf{A}^{-1}\boldsymbol{b}$, and

$$\begin{aligned}
&\|\operatorname{argmin} f - \widehat{\boldsymbol{x}}\|_2^2 = \\
&\quad = \left\|\frac{1}{2}\mathbf{P}\mathbf{A}^{-1}\boldsymbol{b} - \widehat{\boldsymbol{x}}\right\|_2^2 \\
&\quad = \left\|-\left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\overline{\boldsymbol{x}} + \frac{1}{2}\mathbf{P}^\top\mathbf{P}\mathbf{A}^{-1}\boldsymbol{b} - \frac{1}{2}\mathbf{A}^{-1}\boldsymbol{b}\right\|_2^2 \\
&\quad = \left\|\left(\mathbf{I}-\mathbf{P}^\top\mathbf{P}\right)\left(\frac{1}{2}\mathbf{A}^{-1}\boldsymbol{b} - \overline{\boldsymbol{x}}\right)\right\|_2^2.
\end{aligned}$$

□