Piece-wise quadratic lego set for constructing data approximation potentials and their fast optimization

Alexander N. Gorban[1], Evgeny M. Mirkes[1], Andrei Zinovyev[2]

[1] *Department of Mathematics, University of Leicester, University Road, Leicester LE1 7RH, UK*
[2] *Institut Curie, PSL Research University, Mines Paris Tech, Inserm, U900, F-75005, Paris, France.*

Abstract

Data dimension reduction by constructing low-dimensional approximators is one of the most fundamental approaches in data mining. Most efficient approximators based on quadratic error functional are not flexible enough in many circumstances and suffer from sensitivity to outliers and dimensionality curse, which led to introducing other functional forms of error potential that are usually computationally expensive to minimize. We suggest using piece-wise quadratic error potentials of subquadratic growth (PQSQ potentials) that can imitate a variety of approximation metrics used in practice, such as the popular $L1$-norm or even sub-linear potentials corresponding to fractional norm. A family of subquadratic piece-wise potentials are almost as computationally efficient in numerical optimization as quadratic ones for the most popular data approximators (k-means, principal components, principal manifolds and graphs), has guaranteed convergence to the global or local error minimum, allows flexible choice of data approximation metrics and can be naturally robust to outlier data points. We introduce this family of potentials, provide implementations of several popular data approximators exploiting them and do benchmarking.

Keywords: data approximation, nonquadraic potential, principal components, clustering

1. Introduction

Data dimension reduction by constructing explicit low-dimensional approximators of a finite set of vectors is one of the most fundamental approach in data analysis. Starting from the classical data approximators such as k-means [1] and linear principal components (PCA) [2], multiple generalizations have been suggested in the last decades (self-organizing maps, principal curves, principal manifolds, principal graphs, principal trees, etc.)[3, 4] in order to make the data approximators more flexible and suitable for complex data struc-

tures.

We solve the problem of approximating a finite set of vectors $\vec{x}_i \in R^m, i = 1...N$ (data set) by a simpler object L embedded into the data space, such that for each point $\vec{x}_i$ an approximation error $err(\vec{x}_i, L)$ function can be defined. We assume this function in the form

$$err(\vec{x}_i, L) = \min_{y \in L} \sum_k u(x_i^k - y^k), \qquad (1)$$

where the upper $k = 1...m$ stands for the coordinate index, and $u(x)$ is a monotonously growing symmetrical function, which we will be calling the error potential. By data approximation we mean that the embedment of L in the data space minimizes the error

$$\sum_i err(\vec{x}_i, L) \to \min .$$

Note that our definition of error function is coordinate-wise (it is a sum of error potential over all coordinates).

The simplest form of the error potential is quadratic $u(x) = x^2$, which leads to the most known data approximators: mean point (L is a point), principal points (L is a set of points) [5], principal components (L is a line or a hyperplane) [2]. In more advanced cases, L can posses some regular properties leading to principal curves (L is a smooth line or spline) [6], principal manifolds (L is a smooth low-dimensional surface) and principal graphs (eg., L is a pluri-harmonic graph embedment) [7, 3].

There exist multiple advantages of using quadratic potential $u(x)$, because it leads to the most computationally efficient algorithms usually based on the splitting schema, a variant of Expectation-Minimization approach [3]. For example, k-means algorithm solves the problem of finding the set of principal points and the standard iterative Singular Value Decomposition finds principal components. However, quadratic potential is known to be sensitive to outliers in the data set. Also, purely quadratic potentials can suffer from the curse of dimensionality, not being able to robustly discriminate "close" and "distant" point neighbours in a high-dimensional space [8].

There exist several widely used ideas for increasing approximator's robustness in the presence of strong noise in data such as: (1) using medians instead of mean values, (2) substituting quadratic norm by L1 norm (e.g. [9, 10]), (3) outliers exclusion or fixed weighting or iterative reweighting during optimizing the data approximators (e.g. [11, 12, 13]), and (4) regularizing the PCA vectors by L1 norm [14, 15, 16]. In some works, it was suggested to utilize "trimming" averages, e.g. in the context of the k-means clustering or some generalizations of PCA [17, 10]). In the context of regression, iterative reweighting is exploited to mimic the properties of L1 norm [18]. Several algorithms for constructing PCA with L1 norm have been suggested [19, 20, 21] and systematically benchmarked [22, 23]. Some authors go even beyond linear metrics and suggests that fractional norms (Lp metrics with $p < 1$) can be more appropriate in high-dimensional data approximation [8].

However, most of the suggested approaches

exploiting properties of non-quadratic metrics either represent useful but still arbitrary heuristics or are not sufficiently scalable. The standard approach for minimizing L1-based norm consists in solving a linear programming task. Despite existence of many efficient linear programming optimizer implementations, by their nature these computations are much slower than the iterative methods used in the standard SVD algorithm or k-means.

In this paper, we exploit the fact that finding a minimum of a piece-wise quadratic function, or, in other words, a function which is the *minorant of a set of quadratic functionals*, can be almost as computationally efficient as optimizing the standard quadratic potential. Therefore, if a given arbitrary potential (such as L1-based or even fractional norm-based) can be approximated by a piece-wise quadratic function, this should lead to relatively efficient and simple optimization algorithms. It appears that only potentials of quadratic or subquadratic growth are possible in this approach: however, these are the most usefull ones in data analysis. We introduce a rich family of piece-wise quadratic potentials of subquadratic growth (PQSQ-potentials), suggest general approach for their optimization, prove convergence of a simple iterative algorithm in the most general case, and provide implementations of the standard data approximators (mean point, k-means, principal components) using a PQSQ potential. In addition, we discuss exploiting similar approach in non-linear data approximation (i.e. principal manifolds and graphs) and regularized regression (i.e., lasso and elastic nets).

2. Piecewise quadratic potential of subquadratic growth (PQSQ)

2.1. Definition of the PQSQ potential

Let us split all non-negative numbers $x \in R_{\geq 0}$ into $p+1$ non-intersecting intervals $R_0 = [0; r_1), R_1 = [r_1; r_2), ..., R_k = [r_k; r_{k+1}), ..., R_p = [r_p; \infty)$, for a set of thresholds $r_1 < r_2 < ... < r_p$. For convenience, let us denote $r_0 = 0, r_{p+1} = \infty$. Piecewise quadratic potential is a continuous monotonously growing function $u(x)$ constructed from pieces of centered at zero parabolas $y = b_k + a_k x^2$, defined on intervals $x \in [r_k, r_{k+1})$ in the following way (see Figure 1):

$$u(x) = b_k + a_k x^2, \text{if } r_k \leq |x| < r_{k+1}, k = 0...p, \tag{2}$$

$$a_k = \frac{f(r_k) - f(r_{k+1})}{r_k^2 - r_{k+1}^2}, \tag{3}$$

$$b_k = \frac{f(r_{k+1}) r_k^2 - f(r_k) r_{k+1}^2}{r_k^2 - r_{k+1}^2} \tag{4}$$

where $f(x)$ is a majorating function, which is to be approximated (imitated) by $u(x)$. For example, in the simplest case $f(x)$ can be a linear function: $f(x) = x$, in this case, $\sum_k u(x^k)$ will approximate the $L1$-based error function.

Note that accordingly to (3,4), $b_0 = 0, a_p = 0, b_p = f(r_p)$. Therefore, the choice of r_p can naturally create a "trimmed" version of error potential $u(x)$ such that some data points (outliers) do not have any contribution to the gradient of $u(x)$, hence, will not affect the

optimization procedure. However, this set of points can change during minimizaton of the potential.

The condition of subquadratic growth consists in the requirement $a_{k+1} \leq a_k$ and $b_{k+1} \geq b_k$. To guarantee this, the following simple condition on $f(x)$ should be satisfied:

$$f' > 0, \quad f''x \leq f', \tag{5}$$

i.e., $f(x)$ should grow not faster than any parabola $ax^2 + cx, c > 0$.

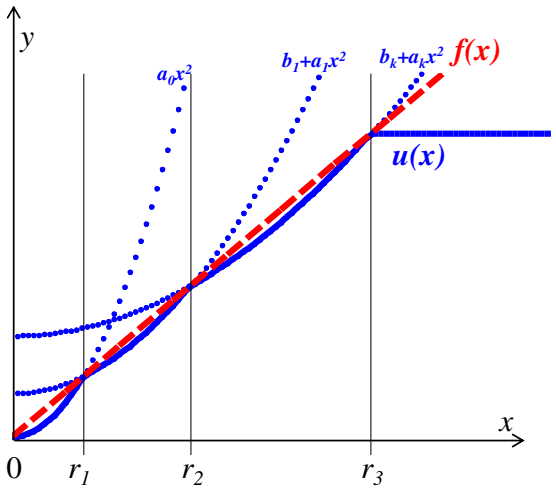

Figure 1: Trimmed piecewise quadratic potential of subquadratic growth $u(x)$ (solid blue line) defined for the majorating function $f(x)$ (red dashed line) and several thresholds r_k. Dotted lines show the parabolas which fragments are used to construct $u(x)$. The last parabola is flat ($a_p = 0$) which corresponds to trimmed potential.

2.2. Basic approach for optimization

In order to use the PQSQ potential in an algorithm, two ingredients should be precomputed:

1) set of p interval thresholds $r_s^k, s = 1...p$ for each coordinate $k = 1...m$.

2) Matrix of a-coefficients defined by (3) based on interval definitions: a_s^k, $s = 0...p$, $k = 1...m$ separately for each coordinate k.

Minimization of PQSQ-based functional consists in two steps:

1) For each coordinate k, split all data point indices into non-overlapping sets $\mathcal{R}_s^k$:

$$\mathcal{R}_s^k = \{i : r_s^k \leq |x_i^k - \beta_i^k| < r_{s+1}^k\}, s = 0...p, \tag{6}$$

where β is a matrix which depends on the nature of the algorithm.

2) Minimize PQSQ-based functional where each set of points $\{x_{i \in \mathcal{R}_s^k}\}$ contributes to the functional quadratically with coefficient a_s^k. This is a quadratic optimization task.

3) Repeat (1)-(2) till convergence.

3. General theory of the piece-wise convex potentials as the cone of minorant functions

In order to deal in most general terms with the data approximation algorithms based on PQSQ potentials, let us consider a general case where a potential can be constructed from a set of functions $\{q_i(x)\}$ with only two requirements: 1) that each $q_i(x)$ has a (local) minimum; 2) that the whole set of all possible $q_i(x)$s forms a cone. In this case, instead of the operational definition (2) it is convenient to define the potential $u(x)$ as the minorant function for a set of functions as follows. For convenience, in this section, x will notify a vector $\vec{x} \in R^m$.

Let us consider *a generating cone of functions* Q. We remind that the definition of a cone implies that for any $q(x) \in Q, p(x) \in Q$, we have $\alpha q(x)+\beta p(x) \in Q$, where $\alpha \geq 0, \beta \geq 0$.

For any finite set of functions $q_1(x) \in Q, q_2(x) \in Q, ..., q_s(x) \in Q$, we define the minorant function (Figure 2):

$$u_{q_1,q_2,...,q_s}(x) = \min(q_1(x), q_2(x), ..., q_s(x)). \quad (7)$$

It is convinient to introduce a multiindex $I_{q_1,q_2,...,q_s}(x)$ indicating which particular function(s) q_i corresponds to the value of $u(x)$, i.e.

$$I_{q_1,q_2,...,q_s}(x) = \{i | u_{q_1,q_2,...,q_s}(x) = q_i(x)\}. \quad (8)$$

For a cone Q let us define a set of all possible minorant functions $\mathbb{M}(Q)$

$$\mathbb{M}(Q) = \{u_{q_{i_1},q_{i_2},...,q_{i_n}} | q_{i_1} \in Q, q_{i_2} \in Q, q_{i_n} \in Q, n = 1, 2, 3, \dots\}. \quad (9)$$

Proposition 1. $\mathbb{M}(Q)$ *is a cone.*

Proof For any two minorant functions $u_{q_{i_1},q_{i_2},...,q_{i_k}}, u_{q_{j_1},q_{j_2},...,q_{j_s}} \in \mathbb{M}(Q)$ we have

$$\alpha u_{q_{i_1},q_{i_2},...,q_{i_k}} + \beta u_{q_{j_1},q_{j_2},...,q_{j_s}} = u_{\{\alpha q_{i_p}+\beta q_{j_r}\}} \in \mathbb{M}(Q), \quad p = 1, \dots, k, r = 1, \dots, s, \quad (10)$$

where $\{\alpha q_{i_p} + \beta q_{j_r}\}$ is a set of all possible linear combinations of functions from $\{q_{i_1}, q_{i_2}, ..., q_{i_k}\}$ and $\{q_{j_1}, q_{j_2}, ..., q_{j_s}\}$.

Proposition 2. *Any restriction of* $\mathbb{M}(Q)$ *onto a linear manifold* L *is a cone.*

Proof Let us denote $q(x)|_L$ a restriction of $q(x)$ function onto L, i.e. $q(x)|_L = \{q(x)|x \in L\}$. $q(x)|_L$ is a part of Q. Set of all $q(x)|_L$ forms a restriction $Q|_L$ of Q onto L. $Q|_L$ is a cone, hence, $\mathbb{M}(Q)|_L = \mathbb{M}(Q|_L)$ is a cone (Proposition 1).

Definition *Splitting algorithm* minimizing $u_{q_1,q_2,...,q_n}(x)$ is defined as **Algorithm 1**.

Algorithm 1 Finding local minimum of a minorant function $u_{q_1,q_2,...,q_n}(x)$

procedure MINIMIZING MINORANT FUNCTION
    initialize $x \leftarrow x_0$
repeat utill stopping criterion has been met:
    compute multiindex $I_{q_1,q_2,...,q_s}(x)$
    for all $i \in I_{q_1,q_2,...,q_s}(x)$
    $x_i = \arg\min\ q_i(x)$
    end for
    select optimal x_i :
    $x_{opt} \leftarrow \arg\min_{x_i} u(x_i)$
    $x \leftarrow x_{opt}$
    stopping criteria: check if the multiindex $I_{q_1,q_2,...,q_s}(x)$ does not change compared to the previous iteration
end repeat:

Theorem 3.1. *Splitting algorithm (**Algorithm 1**) for minimizing* $u_{q_1,q_2,...,q_n}(x)$ *converges in a finite number of steps.*

Proof Since the set of functions $\{q_1, q_2, ..., q_n\}$ is finite then we only have to show that at each step the value of the function $u_{q_1,q_2,...,q_n}(x)$ can not increase. For any x and the value $x' = \arg\min q_i(x)$ for $i \in I_{q_1,q_2,...,q_s}(x)$ we can have only two cases:

(1) Either $I_{q_1,q_2,...,q_s}(x) = I_{q_1,q_2,...,q_s}(x')$ (convergence, and in this case $q_{i'}(x') = q_i(x)$ for any $i' \in I_{q_1,q_2,...,q_s}(x')$);

(2) Or $u_{q_1,q_2,...,q_n}(x) < u_{q_1,q_2,...,q_n}(x')$ since, accordingly to the definition (7), $q_{i'}(x') < q_i(x)$, for any $i' \in I_{q_1,q_2,...,q_s}(x'), i \in I_{q_1,q_2,...,q_s}(x)$ (see Figure 2).

Note that in **Algorithm 1** we do not specify exactly the way to find the local minimum of $q_i(x)$. To be practical, the cone Q should contain only functions for which finding a local minimum is fast and explicit. Evident candidates for this role are positively defined quadratic functionals $q(x) = q_0 + (\vec{q_1}, x) + (x, \mathbb{Q}_2 x)$, where $\mathbb{Q}_2$ is a positively defined symmetric matrix. Any minorant function (7) constructed from positively defined quadratic functions will automatically provide subquadratic growth, since the minorant can not grow faster than any of the quadratic forms by which it is defined.

Operational definition of PQSQ given above (2), corresponds to a particular form of the quadratic functional, with $\mathbb{Q}_2$ being diagonal matrix. This choice corresponds to coordinate-wise definition of data approximation error function (1) that are particularly simple to minimize. This circumstance is used in **Algorithms 2,3**.

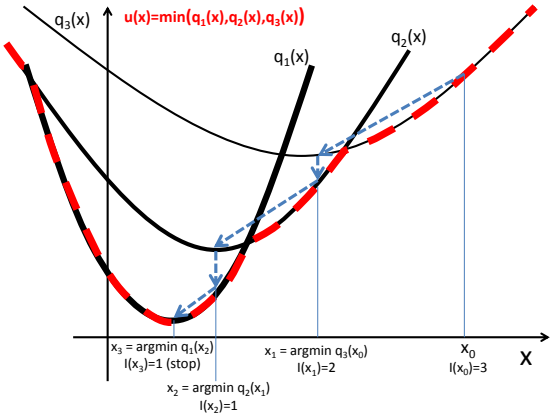

Figure 2: Optimization of a one-dimensional minorant function $u(x)$, defined by three functions $q_1(x), q_2(x), q_3(x)$ each of which has a local minimum. Each optimization step consists in determining which $q_{I(x)}(x) = u(x)$ and making a step into the local minimum of $q_{I(x)}$.

4. Commonly used data approximators with PQSQ potential

4.1. Mean value and k-means clustering in PQSQ approximation measure

Mean vector $\bar{X}_L$ for a set of vectors $X = \{x_i^k\}$, $i = 1...N, k = 1...m$ and an approximation error defined by potential $u(x)$ can be defined as a point minimizing the mean error potential for all points in X:

$$\sum_i \sum_k u(x_i^k - \bar{X}^k) \to \min . \qquad (11)$$

For Euclidean metrics L_2 ($u(x) = x^2$) it is the usual arithmetic mean.

For L_1 metrics ($u(x) = |x|$), (11) leads to the implicit equation $\#(x_i^k > \bar{X}^k) = \#(x_i^k < \bar{X}^k)$, where $\#$ stands for the number of points, which corresponds to the definition of median. This equation can not have

a unique solution in case of even number of points or when some data point coordinates coincide: therefore, definition of median is usually accompanied by heuristics used for breaking ties, i.e. to deal with non-uniquely defined rankings. This situation reflects the general situation of existence of multiple local minimuma and possible non-uniqueness of global minimum of (11) (Figure 3).

For PQSQ approximation measure (2) it is difficult to write down an explicit formula for computing the mean value corresponding to the global minimum of (11). In order to find a point $\bar{X}_{PQSQ}$ minimizing mean $PQSQ$ potential, a simple iterative algorithm can b used:

Algorithm 2 Computing PQSQ mean valu

1: **procedure** PQSQ MEAN VALUE
2: *define intervals* $r_s^k, s = 0...p, k = 1...m$
3: *compute coefficients* a_s^k
4: *initialize* $\bar{X}_{PQSQ}$ *eg., by arithmetic mean*
5: *repeat till convergence of* $\bar{X}_{PQSQ}$:
6: **for each** *coordinate* k
7: *define sets of indices*

$$\mathcal{R}_s^k = \{i : r_s^k \le |x_i^k - \bar{X}_{PQSQ}^k| < r_{s+1}^k\}, \quad s = 0, \ldots, p$$

8: *compute new approximation fo* $\bar{X}_{PQSQ}$:
9: $\bar{X}_{PQSQ}^k \leftarrow \frac{\sum_{s=1...p} a_s^k \sum_{i \in \mathcal{R}_s^k} x_i^k}{\sum_{s=1...p} a_s^k |\mathcal{R}_s^k|}$
10: **end for**
11: **goto** *repeat till convergence*

The suggested algorithm converges to the local minimum which depends on the initial point approximation.

Based on the PQSQ approximation measure and the algorithm for computing the PQSQ mean value (2), one can construct the PQSQ-based k-means clustering procedure in the usual way, splitting estimation of cluster centroids given partitioning of the data points into k disjoint groups, and then re-calculating the partitioning using the PQSQ-based proximity measure.

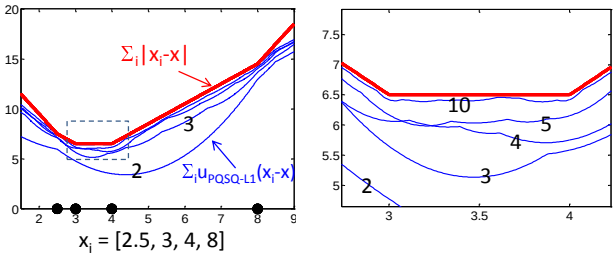

Figure 3: Minimizing the error to a point (finding the mean value) for a set of 4 points (shown by black circles). Solid red line corresponds to L1-based error. Thing blue lines correspond to PQSQ error potential imitating the L1-based error. Several choices of PQSQ potential for different numbers of intervals (indicated by a number put on top of the line) is illustrated. On the right panel a zoom of a particular region of the left plot is shown. Neither function (L1-based or PQSQ-based) possesses a unique local minimum. Moreover, L1-based error function has infinite number of points corresponding to the global minimum (any number between 3 and 4), while PQSQ error function has several local minimuma in [3;4] interval which exact positions are sensitive to the concrete choice of PQSQ parameters (interval definitions).

4.2. Principal Component Analysis (PCA) in PQSQ metrics

Accordingly to the classical definition of the first principal component, it is a line best fit to the data set X [2]. Let us define a line in the parametric form $\vec{y} = \vec{V}u + \vec{\delta}$, where $u \in R^1$ is the parameter. Then the first principal component will be defined by vectors $\vec{V}, \vec{\delta}$ satisfying

$$\sum_i \sum_k u(x_i^k - V^k u_i - \delta^k) \to \min, \qquad (12)$$

where

$$u_i = \arg\min_s \sum_k u(x_i^k - V^k s - \delta^k). \qquad (13)$$

The standard first principal component (PC1) corresponds to $u(x) = x^2$ when the vectors $\vec{V}, \vec{\delta}$ can be found by a simple iterative splitting algorithm for Singular Value Decomposition (SVD). If X does not contain missing values then $\vec{\delta}$ is the vector of arithmetic mean values. By contrast, computing $L1$-based principal components ($u(x) = |x|$) represents a much more challenging optimization problem [21]. Several approximative algorithms for computing $L1$-norm PCA have been recently suggested and benchmarked [19, 20, 21, 22, 23]. To our knowledge, there have not been a general efficient algorithm suggested for computing PCA in case of arbitrary approximation measure for some monotonous function $u(x)$.

Computing PCA based on PQSQ approximation error is only slightly more complicated than computing the standard $L2$ PCA by SVD. Here we provide a pseudo-code (**Algorithm 3**) of a simple iterative algorithm (similar to **Algorithm 2**) with guaranteed convergence (see Section 3).

Computation of second and further principal components follows the standard deflation approach: projections of data points onto the previously computed component are subtracted from the data set, and the algorithm is applied to the residues. However, as it is the case in any non-quadratic metrics, the resulting components can be non-orthogonal or even not invariant with respect to the dataset rotation. Moreover, unlike $L2$-based principal components, the **Algorithm 3** do not always converge to a unique global minimum; the computed components can depend on the initial estimate of $\vec{V}$. The situation is somewhat similar to the standard k-means algorithm. Therefore, in order to achieve the least possible approximation error to the linear subspace, $\vec{V}$ can be initialized randomly or by data vectors $\vec{x}_i$ many times and the deepest in PQSQ approximation error (1) minimum should be selected.

How does the **Algorithm 1** serve a more abstract version of the **Algorithms 2,3**? For example, the "variance" function $m(\vec{x}) = \frac{1}{N}\sum_j u(\vec{x}_j - \vec{x})$ to be minimized in **Algorithm 2** uses the generating functions in the form $Q = \{b_{ji}^k + \sum_k a_{ji}^k (x^k - x_j^k)^2\}$, where i is the index of the interval in (2). Hence, $m(x)$ is a minorant function, belonging to the cone $\mathbb{M}(Q)$, and must converge (to a local minimum) in a finite number of steps accordingly to Theorem 3.1.

Algorithm 3 Computing PQSQ PCA

procedure PQSQ FIRST PRINCIPAL COMPONENT
  define intervals $r_s^k, s = 0...p, k = 1...m$
  compute coefficients a_s^k
  $\vec{\delta} \leftarrow \bar{X}_{PQSQ}$
  initialize $\vec{V}$ *: eg., by L2-based PC1*
  initialize $\{u_i\}$ *: eg., by* $u_i = \frac{\sum_k V^k(x_i^k - \delta^k)}{\sum_k (V^k)^2}$
repeat till convergence of $\vec{V}$:
  normalize $\vec{V}$ *:* $\vec{V} \leftarrow \frac{\vec{V}}{||\vec{V}||}$
  for each *coordinate* k
  define sets of indices

$$\mathcal{R}_s^k = \{i : r_s^k \le |x_i^k - V^k u_i - \delta^k| < r_{s+1}^k\}, \quad s = 0...p$$

  end for
  for each *data point* i and *coordinate* k
  find all $s_{i,k}$ *such that* $i \in \mathcal{R}_{s_{i,k}}^k$
  if *all* $a_{s_{i,k}}^k = 0$ ***then*** $u_i' \leftarrow 0$ **else**

$$u_i' \leftarrow \frac{\sum_k a_{s_{i,k}}^k V^k (x_i^k - \delta^k)}{\sum_k a_{s_{i,k}}^k (V^k)^2}$$

  end for
  for each *coordinate* k

$$V^k \leftarrow \frac{\sum_s a_s^k \sum_{i \in \mathcal{R}_s^k} (x_i^k - \delta^k) u_i}{\sum_s a_s^k \sum_{i \in \mathcal{R}_s^k} (u_i)^2}$$

  end for
  for each i :
  $u_i \leftarrow u_i'$
  end for
  goto *repeat till convergence*

4.3. Nonlinear methods: PQSQ-based Principal Graphs and Manifolds

In a series of works, the authors of this article introduced a family of methods for constructing principal objects based on graph approximations (e.g., principal curves, principal manifolds, principal trees), which allows constructing explicit non-linear data approximators (and, more generally, approximators with non-trivial topologies, suitable for approximating, e.g., datasets with branching or circular topology) [24, 25, 26, 27, 7, 4, 3, 28]. The methodology is based on optimizing a piece-wise quadratic *elastic energy* functional (see short description below). A convenient graphical user interface was developed with implementation of some of these methods [29].

Let G be a simple undirected graph with set of vertices Y and set of edges E. For $k \ge 2$ a k-star in G is a subgraph with $k+1$ vertices $y_{0,1,...k} \in Y$ and k edges $\{(y_0, y_i) \mid i = 1, \dots k\} \subset E$. Suppose for each $k \ge 2$, a family S_k of k-stars in G has been selected. We call a graph G with selected families of k-stars S_k an *elastic graph* if, for all $E^{(i)} \in E$ and $S_k^{(j)} \in S_k$, the correspondent elasticity moduli $\lambda_i > 0$ and $\mu_{kj} > 0$ are defined. Let $E^{(i)}(0), E^{(i)}(1)$ be vertices of an edge $E^{(i)}$ and $S_k^{(j)}(0), \dots S_k^{(j)}(k)$ be vertices of a k-star $S_k^{(j)}$ (among them, $S_k^{(j)}(0)$ is the central vertex).

For any map $\phi : Y \to R^m$ the *energy of the graph* is defined as

$$U^{\phi}(G) := \sum_{E^{(i)}} \lambda_i \left\| \phi(E^{(i)}(0)) - \phi(E^{(i)}(1)) \right\|^2 + \\ + \sum_{S_k^{(j)}} \mu_{kj} \left\| \sum_{i=1}^{k} \phi(S_k^{(j)}(i)) - k\phi(S_k^{(j)}(0)) \right\|^2 .$$

For a given map $\phi : Y \to R^m$ we divide the dataset D into node neighborhoods K^y, $y \in Y$. The set K^y contains the data points for which the node $\phi(y)$ is the closest one in $\phi(y)$. The *energy of approximation* is:

$$U_A^{\phi}(G, D) = \sum_{y \in Y} \sum_{x \in K^y} w(x) \|x - \phi(y)\|^2, \quad (14)$$

where $w(x) \geq 0$ are the point weights. Simple and fast algorithm for minimization of the energy

$$U^{\phi} = U_A^{\phi}(G, D) + U^{\phi}(G) \quad (15)$$

is the splitting algorithm, in the spirit of the classical k-means clustering: for a given system of sets $\{K^y \mid y \in Y\}$ we minimize U^{ϕ} (optimization step, it is the minimization of a positive quadratic functional), then for a given ϕ we find new $\{K^y\}$ (re-partitioning), and so on; stop when no change.

Application of PQSQ-based potential is straightforward in this approach. It consists in replacing (14) with

$$U_A^{\phi}(G, D) = \sum_{y \in Y} \sum_{x \in K^y} w(x) \sum_k u(x^k - \phi(y^k)),$$

where u is a chosen PQSQ-based error potential. Partitioning of the dataset into $\{K^y\}$ can be also based on calculating the minimum PQSQ-based error to y, or can continue enjoying nice properties of $L2$-based distance calculation.

5. Numerical examples

5.1. Practical choices of parameters

The main parameters of PQSQ are (a) majorating function $f(x)$ and (b) decomposition of each coordinate range into $p + 1$ non-overlapping intervals. Depending on these parameters, various approximation error properties can be exploited, including robustness to outlier data points.

When defining the intervals $r_j, j = 1 \dots p$, it is desirable to achieve a small difference between $f(\Delta x)$ and $u(\Delta x)$ for expected argument values Δx (differences between an estimator and the data point), and choose the suitable value of the potential trimming threshold r_p in order to achieve the desired robustness properties. If no trimming is needed, then r_p should be made larger than the maximum expected difference between coordinate values (maximum Δx).

In our numerical experiments we used the following definition of intervals. For any data coordinate k, we define a characteristic difference D^k, for example

$$D^k = \alpha_{scale}(max_i(x_i^k) - min_i(x_i^k)), \quad (16)$$

where α_{scale} is a scaling parameter, which can be put at 1 (in this case, the approximating potential will not be trimmed). In case of existence of outliers, for defining D^k, instead

of amplitude one can use other measures such as the median absolute deviation (MAD):

$$D^k = \alpha_{scale} median_i(|x_i^k - median(\{x_i^k\}|); \quad (17)$$

in this case, the scaling parameter should be made larger, i.e. $\alpha_{scale} = 10$, if no trimming is needed.

After defining D^k we use the following definition of intervals:

$$r_j^k = D^k \frac{j^2}{p^2}, j = 0 \dots p. \quad (18)$$

More sophisticated approaches are also possible to apply such as, given the number of intervals p and the majorating function $f(x)$, choose $r_j, j = 1 \dots p$ in order to minimize the integral difference

$$\int_0^{r_p} (f(x) - u(x))^2 dx \to \min .$$

In further examples, we use (16) and (18) to define intervals in (2).

5.2. Implementation

We provide Matlab implementation of PQSQ approximators (in particular, PCA) at https://github.com/auranic/PQSQ-DataApproximators. Preliminary version of the Java code implementing also elastic graph-based non-linear approximator implementations are available from the authors on request.

5.3. Simple benchmark test

In order to compare the computation time and precision of PQSQ-based PCA algorithm for the case $u(x) = |x|$ with existing R-based implementations of $L1$-based PCA methods (pcaL1 package), we follow the benchmark described in [22]. We compare performance of PQSQ-based PCA based on **Algorithm 3** with several $L1$-based PCA algorithms: L1-PCA* [21], L1-PCA [19], PCA-PP [30], PCA-L1 [20]. As a reference point, we use the standard PCA algorithm based on quadratic norm and computed using the standard SVD iterations.

The idea of benchmarking is to generate a series of datasets of the same size ($N = 1000$ objects in $m = 10$ dimensions) such that the first 5 dimensions would be sampled from a uniform distribution $U(-10, 10)$. Therefore, the first 5 dimensions represent "true manifold" sampled by points.

The values in the last 5 dimensions represent "noise+outlier" signal. The background noise is represented by Laplacian distribution of zero mean and 0.1 variance. The outlier signal is characterized by mean value μ, dimension p and frequency ϕ. Then, for each data point with a probability ϕ, in the first p outlier dimensions a value is drawn from $Laplace(\mu, 0.1)$. The rest of the values is drawn from background noise distribution.

As in [22], we've generated 1300 test sets corresponding to $\phi = 0.1$, with 100 examples for each combination of $\mu = 1, 5, 10, 25$ and $p = 1, 2, 3$.

For each test set 5 first principal components $\vec{V}_1, \dots, \vec{V}_5$ of unit length were computed, with corresponding point projection distributions $U^1, \dots, U^5$ and the mean vector $\vec{C}$. Therefore, for each initial data point $\vec{x}_i$,

we have the "restored" data point

$$P(\vec{x}_i) = \sum_{k=1...5} U_i^k \vec{V}_k + \vec{C}.$$

For computing the PQSQ-based PCA we used 5 intervals without trimming. Changing the number of intervals did not significantly varied the benchmarking results.

Two characteristics were measured: (1) computation time measured as a ratio to the computation of 5 first principal components using the standard $L2$-based PCA and (2) the sum of absolute values of the restored point coordinates in the "outlier" dimensions normalized on the number of points:

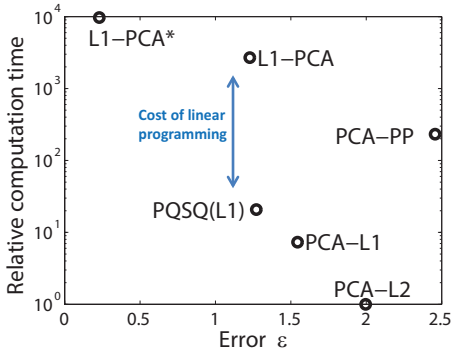

Figure 4: Simple benchmarking of several algorithms for constructing $L1$-based PCA. The computational cost of application of linear programming methods instead of simpler iterative methods is approximately shown by an arrow.

$$\epsilon = \frac{1}{N} \sum_{i=1...N} \sum_{k=6...10} |P^k(\vec{x}_i)|. \qquad (19)$$

Formally speaking, ϵ is $L1$-based from a point projection into the 5 principal components to the "true" subspace. In simple terms, larger values of ϵ correspond to the situation when the first 5 principal components do not represent well the first "true" dimensions, having significant loadings into the "outlier dimensions". Only if the first 5 components do not have any non-zero loadings in the dimensions $6 \ldots 10$ then $\epsilon = 0$.

The results of the comparison, averaged over all 1300 test sets, are shown in Figure 4. As one can see, PQSQ-based computation of PCA outperforms by accuracy the existing heuristics such as PCA-L1 but remains computationally efficient being 100 times faster than L1-PCA giving almost the same accuracy and almost 500 times faster than the L1-PCA* algorithm which is, however, significantly gains in accuracy. From Figure 4, one can see that PQSQ-based approach performs the best in terms of the accuracy in the family of fast iterative methods.

The detailed tables of comparison for all combinations of parameters are provided in Supplementary Table 1. The scripts used to generate the datasets and compare the results can be found at https://github.com/auranic/PQSQ-DataApproximators.

6. Conclusion

In this paper we propose a method of constructing the standard data approximators (mean value, k-means clustering, principal components, principal graphs) for arbitrary non-quadratic approximation error with subquadratic growth by using a piecewise-quadratic error functional (PQSQ potential). These approximators can be computed by

applying quasi-quadratic optimization procedures, which are simple adaptations of the previously described standard and computationally efficient algorithms.

The suggested methodology have several advantages over existing ones:

(a) *Scalability*: the algorithms are computationally efficient and can be applied to large data sets containing millions of numerical values.

(b) *Flexibility*: the algorithms can be adapted to any type of data metrics with subquadratic growth, even if the metrics can not be expressed in explicit form. For example, the error potential can be chosen as adaptive metrics [31, 32].

(c) *Built-in (trimmed) robustness*: choice of intervals in PQSQ can be done in the way to achieve a trimmed version of the standard data approximators, when points distant from the approximator do not affect to the error minimization during the current optimization step.

(d) *Guaranteed convergence*: the suggested algorithms converge to local or global minimum just as the corresponding predecessor algorithms based on quadratic optimization and expectation/minimization-based splitting approach.

One of the application of the suggested methodology is approximating the popular in data mining $L1$ metrics. We show by numerical simulations that PQSQ-based approximators perform as fast as the fast heuristical algorithms for computing $L1$-based PCA but achieve better accuracy in a previously suggested benchmark test. PQSQ-based approximators are less accurate than the exact algorithms for optimizing $L1$-based functions utilizing linear programming: however, they are several orders of magnitude faster.

At the same time, PQSQ-based approximators can imitate a variety of subquadratic error potentials, including fractional ones. In Table 1 we give a possible range of applications of PQSQ-based approximators. Evidently, PSQS potential can be applied in the task of regression, replacing the classical Least Squares or L1-based Least Absolute Deviation methods. Moreover, PQSQ potential can be easily adapted to the problems of regularized regression with non-quadratic penalty onto the regression coefficients, such as LASSO [33] or Elastic Net [34].

References

[1] S. Lloyd, Last square quantization in pcms, Bell Telephone Laboratories Paper (1957).

[2] K. Pearson, On lines and planes of closest fit to systems of points in space, Philos. Mag. 2 (1901) 559–572.

[3] A. N. Gorban, A. Zinovyev, Principal graphs and manifolds, In Handbook of Research on Machine Learning Applications and Trends: Algorithms, Methods and Techniques, eds. Olivas E.S., Guererro J.D.M., Sober M.M., Benedito J.R.M., Lopes A.J.S. (2009).

[4] A. Gorban, B. Kegl, D. Wunsch, A. Zinovyev (Eds.), Principal Manifolds for Data Visualisation and Dimension Reduction, LNCSE 58, Springer, 2008.

Data approximation/Clustering/Manifold learning	
Principal Component Analysis	Includes robust trimmed version of PCA, $L1$-based PCA, regularized PCA, and many other PCA modifications
Principal curves and manifolds	Provides possibility to use non-quadratic data approximation terms and trimmed robust version
Self-Organizing maps	Same as above
Principal graphs/trees	Same as above
K-means	Can include adaptive error potentials based on estimated error distributions inside clusters
High-dimensional data mining	
Use of fractional metrics	Introducing fractional metrics in existing data-mining techniques can potentially deal with the curse of dimensionality, helping to better distinguish close from distant data points [8]
Regularized regression	
Lasso	Application of PQSQ-based potentials should lead to speeding up the algorithm in case of large datasets
Elastic net	Same as above

Table 1: List of methods which can use PQSQ-based error potentials

[5] B. Flury, Principal points, Biometrika 77 (1990) 33–41.

[6] T. Hastie, Principal curves and surfaces, PhD Thesis, Stanford University, California (1984).

[7] A. N. Gorban, N. R. Sumner, A. Y. Zinovyev, Topological grammars for data approximation, Applied Mathematics Letters 20 (2007) 382–386.

[8] C. C. Aggarwal, A. Hinneburg, D. A. Keim, On the surprising behavior of distance metrics in high dimensional space, in: Database Theory - ICDT 2001, 8th International Conference, London, UK, January 4-6, 2001, Proceedings, Springer, 2001, pp. 420–434.

[9] C. Ding, D. Zhou, X. He, H. Zha, R1-PCA: rotational invariant L1-norm principal component analysis for robust subspace factorization, ICML (2006) 281–288.

[10] S. Hauberg, A. Feragen, M. J. Black, Grassmann averages for scalable robust pca, in: Computer Vision and Pattern Recognition (CVPR), 2014 IEEE Conference on, IEEE, pp. 3810–3817.

[11] L. Xu, A. L. Yuille, Robust principal component analysis by self-organizing rules based on statistical physics approach, Neural Networks, IEEE Transactions on 6 (1995) 131–143.

[12] H. Allende, C. Rogel, S. Moreno, R. Salas, Robust neural gas for the analysis of data with outliers, in: Computer Science Society, 2004. SCCC 2004. 24th International Conference of the Chilean, IEEE, pp. 149–155.

[13] T. Kohonen, Self-organizing Maps, Springer Series in Information Sciences, Vol.30, Berlin, Springer, 2001.

[14] I. T. Jolliffe, N. T. Trendafilov, M. Uddin, A modified principal component technique based on the lasso, Journal of computational and Graphical Statistics 12 (2003) 531–547.

[15] E. J. Candès, X. Li, Y. Ma, J. Wright, Robust principal component analysis?, Journal of the ACM (JACM) 58 (2011) 11.

[16] H. Zou, T. Hastie, R. Tibshirani, Sparse principal component analysis, Journal of computational and graphical statistics 15 (2006) 265–286.

[17] J. Cuesta-Albertos, A. Gordaliza, C. Matrán, et al., Trimmed k-means: An attempt to robustify quantizers, The Annals of Statistics 25 (1997) 553–576.

[18] C. Lu, Z. Lin, S. Yan, Smoothed low rank and sparse matrix recovery by iteratively reweighted least squares minimization., IEEE Trans Image Process 24 (2015) 646–654.

[19] Q. Ke, T. Kanade, Robust l 1 norm factorization in the presence of outliers and missing data by alternative convex programming, in: Computer Vision and Pattern Recognition, 2005. CVPR 2005.

IEEE Computer Society Conference on, volume 1, IEEE, pp. 739–746.

[20] N. Kwak, Principal component analysis based on l1-norm maximization, Pattern Analysis and Machine Intelligence, IEEE Transactions on 30 (2008) 1672–1680.

[21] J. Brooks, J. Dulá, E. Boone, A pure l1-norm principal component analysis., Comput Stat Data Anal 61 (2013) 83–98.

[22] J. Brooks, S. Jot, pcal1: An implementation in r of three methods for l1-norm principal component analysis, Optimization Online preprint (2012).

[23] Y. W. Park, D. Klabjan, Algorithms for l1-norm principal component analysis (2014).

[24] A. Gorban', A. Rossiev, Neural network iterative method of principal curves for data with gaps, Journal of Computer and Systems Sciences International c/c of Tekhnicheskaia Kibernetika 38 (1999) 825–830.

[25] A. Gorban, A. Zinovyev, Visualization of data by method of elastic maps and its applications in genomics, economics and sociology, IHES Preprints (2001).

[26] A. Gorban, A. Y. Zinovyev, Method of elastic maps and its applications in data visualization and data modeling, International Journal of Computing Anticipatory Systems, CHAOS 12 (2001) 353–369.

[27] A. Gorban, A. Zinovyev, Elastic principal graphs and manifolds and their practical applications, Computing 75 (2005) 359–379.

[28] A. N. Gorban, A. Zinovyev, Principal manifolds and graphs in practice: from molecular biology to dynamical systems., Int J Neural Syst 20 (2010) 219–232.

[29] A. N. Gorban, A. Pitenko, A. Zinovyev, Vidaexpert: user-friendly tool for nonlinear visualization and analysis of multidimensional vectorial data, arXiv preprint arXiv:1406.5550 (2014).

[30] C. Croux, P. Filzmoser, M. R. Oliveira, Algorithms for projection–pursuit robust principal component analysis, Chemometrics and Intelligent Laboratory Systems 87 (2007) 218–225.

[31] L. Yang, R. Jin, Distance metric learning: A comprehensive survey, Michigan State Universiy 2 (2006).

[32] L. Wu, R. Jin, S. C. Hoi, J. Zhu, N. Yu, Learning bregman distance functions and its application for semi-supervised clustering, in: Advances in neural information processing systems, pp. 2089–2097.

[33] R. Tibshirani, Regression shrinkage and selection via the lasso, Journal of the Royal Statistical Society. Series B (Methodological) (1996) 267–288.

[34] H. Zou, T. Hastie, Regularization and variable selection via the elastic net, Journal of the Royal Statistical Society: Series B (Statistical Methodology) 67 (2005) 301–320.