DATADVANCE

## pSeven Core

# GTDR

Generic Tool for Dimension Reduction

(Feature Extraction mode)

Contact information

| | | |
|---|---|---|
| Phone | +7 (495) 669 68 15 | |
| Web | `https://www.pseven.io/` | |
| Email | `support@datadvance.net` | Technical support, questions, bug reports |
| | `info@datadvance.net` | Everything else |
| Mail | DATADVANCE, llc<br>Nauchny proezd, 17, 15th floor, office XXXI<br>117246 Moscow<br>Russia | |

User manual prepared by Pavel Erofeev, Pavel Prikhodko, Evgeny Burnaev

# Contents

# List of Figures

# List of Tables

# Chapter 1

# Introduction

## 1.1 What is GTDR (FE mode)

**GTDR (FE mode)** is a software package which implements solution of **Dimension Reduction (DR)** problem for different types of the problem statement using user-provided training data. In the current manual only **Supervised (Effective) Dimension Reduction (EDR)** or **Feature Extraction (FE)** problem statement is considered (FE mode for DR). **GTDR (FE mode)** tool extracts linear combinations of features that mostly influence the output variables thereby answering this question directly.

## 1.2 Documentation structure

Documentation for **GTDR (FE mode)** includes:

- User manual (this document) which contains:
  - A general overview of the tool's functionality;
  - Short descriptions of the problem and representative algorithms;
  - Recommendations on the tool's usage;
  - Examples of applications to model problems.

The present document has the following structure:

- Chapter 2 is an informal introduction to the tool's functionality.
- Chapter 3 gives a more formal mathematical explanation. It contains an overview of relevant effective dimension reduction concepts and state of the art methods.
- Chapter 4 provides the **GTDR (FE mode)** algorithm details.
- Chapter 5 describes the internal workflow of the tool.
- Chapter 8 gives some examples of **GTDR (FE mode)** tool usage for some toy and real-world problems.

# Chapter 2

# Overview

The main goal of Generic Tool for Dimension Reduction (Feature Extraction mode) (**GT DR (FE mode)**) is to extract the most important linear combination of features[1] for the user-provided dependency which is represented as a *data sample* [2].

**GTDR (FE mode)** tool attempts to answer the following questions:

1. *What linear combination of initial features do influence the dependency and thus should be included in the further study?*

2. *If some kind of redundancy in inputs is assumed, how the number of features considered in the problem could be effectively reduced?*

Examples of **GTDR (FE mode)** application that address the questions above are presented in the Chapter 8.

In this chapter the general problem of dimension reduction[3] is informally discussed and some motivating examples are given. The more formal mathematical definitions and statements are given in Chapter 3 along with short review of the most representative state of the art methods.

Dimension reduction is the mapping of data to a lower dimensional space such that uninformative variance in the data is discarded, or such that a subspace in which the data belongs to is detected. Dimension reduction is generally used for data visualization, and for extracting key low dimensional features (e.g., the 2-dimensional orientation of an object, from its high dimensional image representation). In some cases the desired low dimensional features depend on the task at hand. Apart from teaching us about the data, dimension reduction can lead us to better models for inference. The need for dimension reduction also arises for other pressing reasons when building approximation or design of experiment (see user manuals for GTApprox and for GTDoE resp.).

The problem of dimension reduction can be stated in two fundamentally different ways:

- The first one considers **internal structure of data** and assumes that data sample belongs to some manifold in original space of lower dimension. This type of problem statement is further referred to as *Unsupervised Dimension Reduction.*

[1] Hereinafter the term “feature” refers to a coordinate of the digital vector representation of some objects in the specified space.

[2] also known as *training data* (or *samples*)

[3] We follow the lead of statistics community to reduce “dimensionality reduction” and “dimensional reduction” to “dimension reduction”.

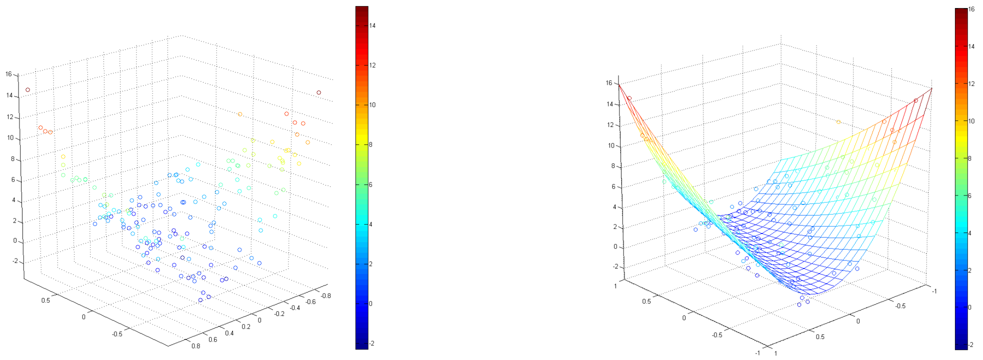
- Another approach explores the **functional dependence** in the data sample and extracts some subspace of the original feature space that preserves information about the dependence. This type of problem statement is further referred to as *Supervised Dimension Reduction* or *Effective (sufficient) Dimension Reduction* or, in some literature, *Feature Extraction*. The last term is rather confusing so we avoid it throughout the manual.

Current version of GT DR is intended to solve Unsupervised Dimension Reduction kind of problem in assumption of almost linear subspaces, i.e. subspaces that slightly deviate from hyperplanes.

To illustrate Supervised Dimension Reduction problem consider the following data generation model (see Figure 2.1):

$$y = f(x_1, x_2) + \varepsilon = (x_1 + 3x_2)^2 + \varepsilon, \tag{2.1}$$

where $x_i \in [-1, 1], i = 1, 2$ and $\varepsilon$ is a random variable with standard normal distribution $\mathcal{N}(0, 1)$. The original dimensionality of regressors is 2. But it is quite easy to see, that the effective dimensionality is 1 as $y$ really depend on the linear combination of $x_i$, namely on $\xi = x_1 + 3x_2$, i.e. $y = g(\xi) + \varepsilon = g([x_1, x_2]B^T) + \varepsilon$, where $B = [1, 3]$ is a projection matrix. Obviously, further data analysis is much easier in 1-dimensional space rather than in 2-dimensional one. **GTDR (FE mode)** is addressed to solve this particular kind of problems. One more example of application is described in Other illustrating artificial and real-world examples see in Chapter 8.



(a) Dataset $y = (x_1 + 3x_2)^2 + \varepsilon$

(b) Dataset and Surface $y = (x_1 + 3x_2)^2$

Figure 2.1: Example for Effective Dimension Reduction problem

General Effective (Supervised) Dimension Reduction problem is divided into two subproblems that differ in purpose:

1. The final goal is to build up a regression on data or similar task, i.e. we need to study the expectation of outputs with respect to inputs.

2. We need to study the whole conditional distribution of outputs with respect to inputs, which may be helpful e.g. in further accuracy estimation of approximation models build upon the data.

This distinction forms two fundamental concepts[4]:

[4] The more accurate mathematical definitions are given in Chapter 3.

- **Central Mean Subspace (CMS)** - the subspace which is a *linear span of expectation of outputs with respect to inputs.*
- **Central Subspace (CS)** - the subspace which is a *linear span of the whole conditional distribution of outputs with respect to inputs.*

To illustrate the difference consider a little bit more complicated example of data generation function:

$$y = f(x_1, x_2) + \tilde{\varepsilon}(x_1, x_2) = (x_1 + 3x_2)^2 + x_1\varepsilon,$$

where again $x_i \in [-1, 1], i = 1, 2$ and $\varepsilon$ is a random variable with standard normal distribution $\mathcal{N}(0, 1)$. The CMS is the same as in case 2.1 as $E[Y|x_1, x_2] = (x_1 + 3x_2)^2$ and defined by the projection matrix $B = [1, 3]$. But the CS is different as the error term depends on the values of $x_1$. And if we want to study the whole conditional distribution of $y$ with respect to $x_1$ and $x_2$ *we can not reduce the dimensionality of inputs* in this particular example! Important to notice that current version of **GTDR (FE mode)** tool *explores only CMS.*

## 2.1 Illustrative Example

To demonstrate the effectiveness of the tool we will use real-world data set from UCI Machine Learning Repository [5].These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents (Alcohol, Malic acid, Magnesium, Total phenols, etc.) found in each of the three types of wines. The task is to perform wine classification (i.e. to assign each wine to one of the three cultivars).

**GTDR (FE mode)** is applied in this case as a visualization tool i.e. in order to find the best low-dimensional representation of data where the classification is straightforward. The original feature space for classification task consist of 13 dimensions. **GTDR (FE mode)** allows to reduce dimensionality down to 2 (this value is automatically selected inside the tool). The resulting low-dimensional representation of the problem is depicted in figure 2.2. Different classes are marked with different colors. The classification task in this representation is obvious (e.g. see figure 2.3 with simple 1-nearest neighbor space classification).

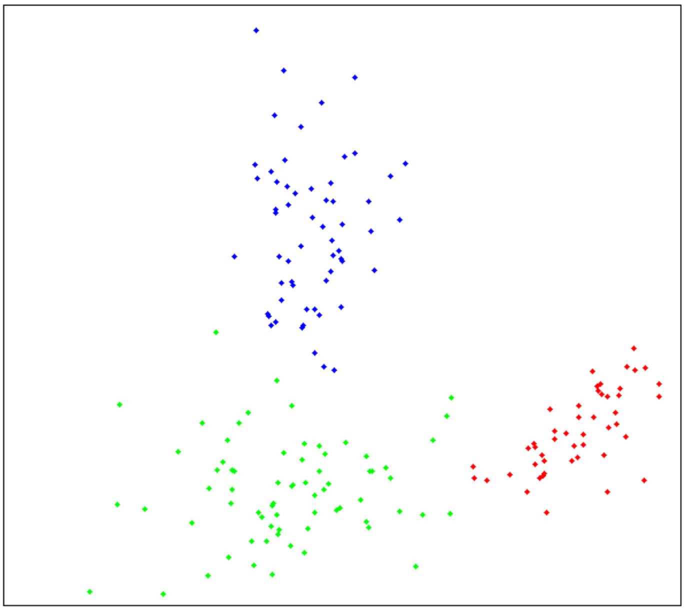


Figure 2.2: Example of classification visualization with **GTDR (FE mode)**: *blue* - class 1, *green* - class 2, *red* - class 3.

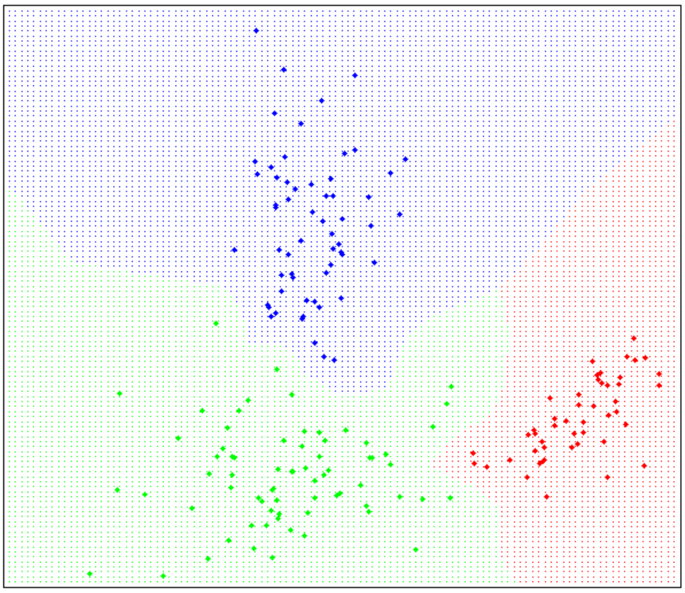


Figure 2.3: Classification of wine in low-dimensional space: *blue* - class 1, *green* - class 2, *red* - class 3.

# Chapter 3

# Problem Statements

The **GT DR** module implements the solution of a mathematical task that can be formulated as follows: based on the specified training data set (the structure of the training data set is described below), create a DR-model $\Sigma = \{p, d, \mathcal{C}, \mathcal{D}\}$ defined by the following parameters:

- $p$ - dimension of the initial vector $X$;
- $d$ - dimension of the compressed vector $\Lambda$;
- $\mathcal{C}$ - the compression procedure that reduces the vector $X$ of dimension $p$ to the $d$-dimensional compressed vector $\Lambda$;
- $\mathcal{D}$ - the decompression procedure that recovers the $d$-dimensional compressed vector $\Lambda$ to the full dimensional vector $X$.

The DR-model must also comply with the specified requirements (depending on the dimension reduction problem type), which are determined either by the dimension $d$ of the compressed vector or by the accuracy of the procedure.

As noticed in Chapter 2 the Dimension Reduction problem can be stated in two fundamentally different ways.

In the Unsupervised Dimension Reduction, considered below, the specified training data set consists of $N$ $p$-dimensional vectors $\{X_1, X_2, ..., X_N\}$ (prototypes). Also, the accuracy of DR-model $\Sigma = \{p, d, \mathcal{C}, \mathcal{D}\}$, applied to the initial vector $X$, is determined by the error of decompression, which is understood as the distance $d(X, X^*) = \|X - X^*\|$ between the original vector $X$ and the reconstructed vector $X^* = \mathcal{D}(\mathcal{C}(X))$, obtained by first applying the compression procedure and then the decompression procedure to the original vector $X$. For a given data set $\{X_1, X_2, \ldots, X_N\}$ the accuracy of DR-model can be measured by the root-mean-square error of decompression $\varepsilon$:

$$e_1(\Sigma) = \sqrt{\frac{1}{N}\sum_{i=1}^{N} \|X_i - X_i^*\|^2}.$$

Now let us formulate two possible statements for Supervised and Unsupervised Dimension Reduction.

**Unsupervised Dimension Reduction Task 1.** For a given data set $\{X_1, X_2, \ldots, X_N\}$, construct a DR-model $\Sigma = \{p, d, \mathcal{C}, \mathcal{D}\}$ belonging to the special class of models $\mathcal{M}$ such that the root-mean-square error of decompression $e_1(\Sigma)$ does not exceed the specified value $\tilde{e}_1$ and the compressed vector has the smallest possible dimension $d$.

**Unsupervised Dimension Reduction Task 2.** For a given data set $\{X_1, X_2, \ldots, X_N\}$, and the specified reduced dimension $d$, construct a DR-model $\Sigma = \{p, d, \mathcal{C}, \mathcal{D}\}$ belonging to the special class of models $\mathcal{M}$ and having the lowest possible root-mean-square error of decompression $e_1(\Sigma)$ for the value of reduced dimension $d$, specified by the User.

**Supervised Dimension Reduction.** Now let us consider Supervised (Effective) Dimension Reduction problem statement. First of all we describe the training data set and the way to estimate the accuracy of corresponding DR-model. Let $Y = f(X), X \in \mathbb{R}^d, Y \in \mathbb{R}^q$ be some dependency. The specified training data set consists of $N$ $d$-dimensional input vectors and corresponding $q$-dimensional output vectors, namely, $\big(\mathbf{X}, \mathbf{Y}\big) = \{(X_1, Y_1), (X_2, Y_2), \ldots, (X_N, Y_N)\}$, where $Y_i = f(X_i), i = 1, ..., N$. The accuracy of DR-model $\Sigma = \{p, d, \mathcal{C}, \mathcal{D}\}$, applied to the initial vector $X$, is determined by the discrepancy between function value $f(X)$ for the initial vector $X$ and function value $f(X^*)$ for the reconstructed vector $X^* = \mathcal{D}(\mathcal{C}(X))$, obtained by applying first the compression procedure and then the decompression procedure to the original vector $X$. For a given data set $\big(\mathbf{X}, \mathbf{Y}\big)$ the accuracy of DR-model is governed by the root-mean-square error $\varepsilon$:

$$e_2(\Sigma) = \sqrt{\frac{1}{N} \sum_{i=1}^{N} \|Y_i - f(X_i^*)\|^2}.$$

Finally, Effective Dimension Reduction problem can be formulated as follows. Based on a given data set $\big(\mathbf{X}, \mathbf{Y}\big)$, construct a DR-model $\Sigma = \{p, d, \mathcal{C}, \mathcal{D}\}$ having the lowest possible root-mean-square error $e_2(\Sigma)$ for the value of reduced dimension $d$, specified by the User.

In the current version of **GT DR** it is assumed that

$$\mathcal{C} : X \in \mathbb{R}^p \to \Lambda = \Lambda(X) = XB^T \in \mathbb{R}^d$$
$$D : \Lambda \in \mathbb{R}^d \to X^* = X^*(\Lambda) = \Lambda B \in \mathbb{R}^p$$

for some unknown matrix $B \in \mathbb{R}^{d \times p}$ and that the elements of the given data set were generated according to the following model:

$$Y_i = f(X_i) \approx g(\Lambda(X_i)) = g(XB^T), i = 1, \ldots, N,$$

for some unknown function $g(\Lambda), \Lambda \in \mathbb{R}^d$. Effective dimension reduction algorithm of **GT DR**, implementing solution of Supervised Dimension Reduction, estimates the matrix $B \in \mathbb{R}^{d \times p}$ using the sample $\big(\mathbf{X}, \mathbf{Y}\big)$. The rest of this document is devoted to this type of the Dimension Reduction problem.

## 3.1 Mathematical Problem Statement for Supervised (Effective) Dimension Reduction

In this section general problem statement for Supervised (Effective) Dimension Reduction problem is described in details.

Let $Y = f(X), X \in R^p, Y \in R^q$ be some considered dependency[1]. It may be some physical experiment or solver code. Let $\Sigma = \{p, d, \mathcal{C}, \mathcal{D}\}$ be some dimension reduction procedure from certain class $\mathcal{P}$, where

[1] also known as function

- $\mathcal{C} : X \in R^p \to \Lambda = \Lambda(X) \in R^d$
- $\mathcal{D} : \Lambda \in R^d \leftarrow X^* = X^*(\Lambda) \in R^p$

**GTDR (FE mode)** procedure is a dimension reduction procedure that studies conditional distribution $F_{Y|X}$ of $Y$ given $X$ and extracts feature subspace of vectors $\Lambda(X) \in R^d$ such that $Y$ independent of $X|\Lambda(X)$ and $d < p$. In the current version of **GTDR (FE mode)** only linear compression/decompression transformations are considered, i.e. $\Lambda = XB_{\mathcal{S}}^T$ and $X^* = \Lambda B_{\mathcal{S}}$, where $B_{\mathcal{S}} = [\beta_1^T, \dots, \beta_d^T]^T$ is a matrix of size $d \times p$, comprised of orthogonal vectors $\beta_i \in R^p, i = \overline{1, d}$, which forms orthogonal basis in reduced dimension space $\mathcal{S}$.

The underlying model for effective dimension reduction can be represented as follows. Consider the general data generation model $Y = f_1(X) + f_2(X) \cdot \varepsilon = f_1(X) + \varepsilon(X)$, where $f_1(\cdot)$ and $f_2(\cdot)$ are some generally nonlinear functions, $\varepsilon$ is a white noise with standard normal distribution: zero mean and some constant variance, and $E[\varepsilon(X)|X] = 0$. It is assumed that the model can be rewritten in the following way: $Y = g_1(XB_1^T) + \tilde{\varepsilon}(g_2(XB_2^T))$, where $g_1(\cdot)$ and $g_2(\cdot)$ are some functions either linear or not and $\tilde{\varepsilon}(\cdot)$ is a some random functional with zero expectation given $X$, i.e. $E[\tilde{\varepsilon}(g_2(XB_2^T))|X)] = 0$.

Two cases should be distinguished:

1. Our goal is to construct subspace containing information about regression of $Y$ with respect to $X$, i.e. $E[Y|X] = E[g_1(XB_1^T) + \tilde{\varepsilon}(g_2(XB_2^T))|X] = E[g_1(XB_1^T)|X]$. Subspace $\mathcal{S}_{CMS} = Span\{B_1\}$ is called **Central Mean Subspace (CMS)**. Extraction of such a subspace can improve regression or optimization.
2. We study the conditional distribution $F_{Y|X}$ including possible dependence of error terms on the data (case when distributions for $\varepsilon$ and $X$ are not independent). The subspace of interest is called **Central Subspace (CS)** and is more general than CMS as $\mathcal{S}_{CMS} \subset \mathcal{S}_{CS} = Span\{B_1, B_2\}$. Extraction of such a subspace can improve, e.g. regression model accuracy estimation.

### 3.1.1 Effective Dimension Reduction Procedure Performance Measures

There are four general criteria for evaluation of effective dimension reduction model performance:

1. The accuracy of reduced dimension subspace reconstruction (can be estimated only if we know exact dimension reduction subspace). This criterion is measured by the following formula[2]
$$I_1 = \left\| (I - B_{\mathcal{S}} B_{\mathcal{S}}^T)\hat{B} \right\|_{fro}{}^3,$$
where $B_{\mathcal{S}}$ is the real matrix that defines the reduced dimension subspace, $\hat{B}$ is an estimate, produced by effective dimension reduction algorithm, and $I$ is an identity matrix of appropriate size. This measure is applicable only to artificial problems as the knowledge of reduced dimension subspace is necessary.

[2] Straightforward comparison of real and estimated by **GTDR (FE mode)** tool matrices is not consistent as one can define infinitely many orthogonal bases in reduced dimension subspace (if reduced dimension is higher than 1).

[3] The notation $\|A\|_{fro} = \sqrt{tr(A^T A)}$ means Frobenius norm for matrix A.

2. The performance of the approximation model $f_{surr}(\Lambda(X))$, build using extracted features as inputs instead of full-dimensional input vectors and some approximation technique, for example, GT Approx

$$I_2 = \Big\|f(X) - f_{surr}(\Lambda(X))\Big\|.$$

3. The quantity of persistent information about dependency preserved in reduced dimension input vectors. This value can be measured as follows

$$I_3 = \Big\|f(X) - f(X^*(\Lambda(X)))\Big\|.$$

4. The performance of optimization algorithm constrained by some time budget when optimizing with respect to the coordinates of vectors from the extracted feature space. The quantitative measure of this criteria is the value of objective function after the optimization budget is spent. Obtained results should be compared with the value of objective function, obtained using the same budget but when optimizing with respect to the coordinates of initial input space.

## 3.2 State of the Art Methods

There are two types of Effective Dimension Reduction methods. Algorithms of the first type are intended to estimate Central Subspace while the rest are used for Central Mean Subspace estimation. **GTDR (FE mode)** tool belongs to the latter subset of methods. The most important and most frequently cited algorithms are named in the following subsection along with references to original articles.

All the algorithms for CMS estimation share the same structure, which can be described in two steps:

1. On the first step estimate some functional $\Pi(X)$ that belongs to $\mathcal{S}_{CMS}$ using some initial sample.

2. On the second step estimate effective dimension reduction projection matrix containing desired number of first principal components of $\Pi(X)$, that are constructed using Principal Component Analysis technique [7].

Also important to notice that most of the methods impose rather strict probabilistic assumptions on data in order to produce intended results. While **GTDR (FE mode)** tool has rather mild limitations which are described further in section 6.3.

**Central Subspace Estimation Methods.** The following two methods and their variations are generally used for CS estimation: Sliced Inverse Regression (SIR) [10] and Sliced Average Variance Estimation (SAVE) [2]. Both methods are based on the inverse regression construction implying the fact that it is also contained in CS. The main drawback of these two methods is that they require special rather strict probabilistic assumptions on the data to be consistent.

**Central Mean Subspace Estimation Methods.** The following methods are used for CMS estimation.

In case when function $g_1(\cdot)$ is linear Partial Least Squares (PLS) [6] approach is proved to give the best estimate.

The Principal Hessian Directions (PHD) [11] and Iterative Hessian Transformation (IHT) [3] methods use some estimates of mean Hessian matrix for considered dependence. But the estimates require strict probabilistic assumptions on data to be consistent.

Rather large class of methods including Minimum Average Variance Estimation (MAVE) [15, 1], Outer Product Gradient (OPG) [14] and Structural Adaptation via Maximum Minimization (SAMM) [4] significantly differ from others as they employ nonparametric regression to estimate some functional $\Pi(X)$ used for CMS construction. The weak point in this approach is that nonparametric regression is exposed to the curse of dimensionality which makes it not reasonable tool in high dimensions.

Sliced Regression (SR) [13] method discretizes model output and uses one of the other methods for further estimation which allows achieving more robustness.

**GTDR (FE mode)** tool is free from all these methods' significant drawbacks due to the special way of gradient approximation in the high-dimensional input space based on sparse expansion in parametric functions of different types. This method is much less affected by the curse of dimensionality which gives the opportunity to reduce dimension significantly and effectively.

# Chapter 4

# Algorithm Description

In this chapter details on **GTDR (FE mode)** algorithm are given.

## 4.1 Projection Matrix Estimation

Let the initial dataset $\big(\mathbf{X}, \mathbf{Y}\big)$ contain $N$ pairs of input and output vectors. Two important steps lie in the core of **GTDR (FE mode)**. In short they can be described as follows.

1. On the first step for each point from the sample retrieve gradient estimate of the dependence in the original dimension space using some of **GTApprox** capabilities:

$$\widehat{\nabla f(X)}\Big|_{X=X_i} = \left(\frac{\partial \widehat{f}(X)}{\partial x_1}, \ldots, \frac{\partial \widehat{f}(X)}{\partial x_p}\right)\Big|_{X=X_i}, \; i = \overline{1, N}.$$

2. On the second step calculate principal component directions for outer product (covariance matrix) of gradient matrix

$$\hat{\Gamma} = \left(\widehat{\nabla f(X)}\Big|_{X=X_1}, \ldots, \widehat{\nabla f(X)}\Big|_{X=X_N}\right),$$

   finding all nontrivial solutions to the problem: $\beta_j \hat{\Gamma}\hat{\Gamma}^T = \beta_j$. Take first $d$ directions corresponding to the highest eigenvalues. Finally, effective dimension reduction procedure (actually central mean subspace representation) is defined by the projection matrix $B = \big(\beta_1^T, \ldots, \beta_d^T\big)^T$.

In case we have **black box function** instead of sample the first step could be replaced with the following procedure (for time saving purpose). The gradients $\widehat{\nabla f(X)}|_{X=X_i}, i = \overline{1, N}$ are numerically estimated using robust difference scheme in the points of specific space filling design: Faure sequence (see Design of Experiment (DoE) user manual for details). For this purpose the **black box interface** of **GTDR (FE mode)** is provided which takes user specified black box function as input. This approach is preferable in noise-free problem settings.

**GTDR (FE mode)** shares basic ideas with most advanced state of the art CMS estimation methods while our comparative study showed that **GTDR (FE mode)** demonstrates better performance in terms of accuracy on toy and real world data and has less restrictive limitations (see section 6.3 for details).

## 4.2 Reduced Dimension Determination

**GTDR (FE mode)** tool also provides a functionality for automatical determination of the intrinsic data dimension – *the recommended dimension*. The basic idea lying under the dimension determination algorithm is as follows. If the projection matrix $B$ was correctly estimated by the main algorithm (see section 4.1) than we can assume that the central mean subspaces spanned by bootstrapped gradients is quite stable. The same goes for the complementary subspace. So we should seek for the dimension that provides the most stable subspace among all the bootstrapped CMSs.

It is also important to notice that this procedure is not computationally difficult compared to the main algorithm.

This functionality also provides a way of determining whether *the tool worked out or not.* If the recommended dimension is full dimension, then either there is no intrinsic dimension (in terms of linear combination of features) or there is not enough points for the accurate estimation. This case accompanied Other cases (recommended dimension is lower than full dimension) indicate that the algorithm detected some intrinsic data structure.

## 4.3 Projection Gradients and Cumulative Loadings

Projection gradients in case of linear model considered are exactly projection matrix $B$. These coefficients may be further interpreted to determine the relative importance of the original input.

Another illustrative output of the algorithm is cumulative loadings matrix $L$ which represents the relative cumulative impact of the original space variables on the reduced-dimension space. Let projection matrix $B$ consist of elements $b_{ij}$, then for given reduced dimension $\tilde{d}$:

$$L_{\tilde{d}j} = \sqrt{\sum_{i=1}^{\tilde{d}} (b_{ij}^2)}.$$

It means that in case the reduced dimension equals to $\tilde{d}$, the relative impacts of the original variables in the reduced space are stored in the $\tilde{d}$'th row of matrix of cumulative loadings $L$.

Note, that these both outputs describe only *relative behavior of the original inputs without accounting for the scales* (all original input variables are normalized by variance before the estimation).

# Chapter 5

# Internal Workflow

## 5.1 The Workflow

The internal workflow of the **GTDR (FE mode)** tool consists of the following steps.

1. **Preprocessing.** On the first step data preprocessing is performed in the following way: redundant data is removed from the training set and data is normalized. See Section 5.2 for details. This step is omitted in *black-box* implementation.

2. **Construction of GTDR (FE mode) procedure.** On the second step actual effective dimension reduction is performed. See Section 4 for details.

## 5.2 Preprocessing

In case of *sample-based* interface some preprocessing is performed. As we work with initial training dataset some reasonable preprocessing must be applied to it in order to remove possible degeneracies in the data. Let $\big(\mathbf{X}, \mathbf{Y}\big)$ be the $\tilde{N} \times (\tilde{p} + q)$ matrix of the training data, where the rows are $(\tilde{p}+q)$-dimensional training points, and the columns are individual scalar components of the input or output. The matrix $\big(\mathbf{X}, \mathbf{Y}\big)$ consists of the sub-matrices $\mathbf{X}$ and $\mathbf{Y}$. We perform the following operation with the matrix $\big(\mathbf{X}, \mathbf{Y}\big)$:

1. We remove all constant columns in the sub-matrix $\mathbf{X}$. A constant column in $\mathbf{X}$ means that all the training vectors have the same value of one of the input components. In particular, this means that the training DoE is degenerate and lies in a proper subspace of the design space.

2. We remove repeating rows in the matrix $\big(\mathbf{X}, \mathbf{Y}\big)$. A repeating row means that the same training vector is included more than once in the training matrix. Repetitions bring no additional information and are therefore ignored; a repeating row is counted only once.

If the above operations are nontrivial, e.g., if the matrix $\mathbf{X}$ does contain constant columns or the matrix $\big(\mathbf{X}, \mathbf{Y}\big)$ does contain repeating rows, then the removals are accompanied by warnings.

As a result of these operations, we obtain a reduced matrix $\big(\widetilde{\mathbf{X}}, \widetilde{\mathbf{Y}}\big)$ consisting of the submatrices $\widetilde{\mathbf{X}}$ and $\widetilde{\mathbf{Y}}$. Accordingly, we define *effective input dimension* $(p)$ and the *effective sample size* $(N)$ as the corresponding dimensions of the sub-matrix $\widetilde{\mathbf{X}}$.

Note that after removing repeating rows in $\big(\mathbf{X},\mathbf{Y}\big)$ the reduced matrix may still contain rows which have the same $X$ components but different $Y$ components. This means that the training data is so noisy that, in general, several different outputs correspond to the same input. If the training data does contain rows with equal $X$ but different $Y$ components, the tool produces a warning.

The effective dimension reduction space is estimated by **GTDR (FE mode)** using the reduced matrix $\big(\widetilde{\mathbf{X}},\widetilde{\mathbf{Y}}\big)$ rather than the original matrix $\big(\mathbf{X},\mathbf{Y}\big)$.

# Chapter 6

# Procedure Usage

In this section the usage of the procedure is described. The process of tool application is presented in the figure 6.1.

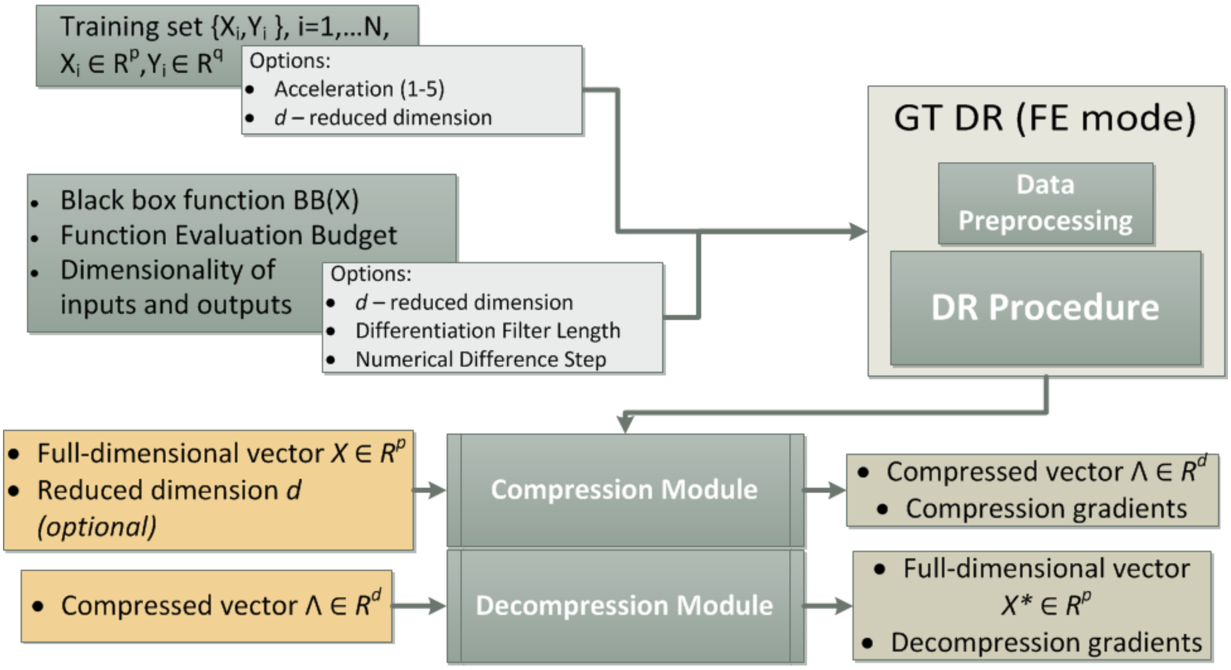

Training set {X_i,Y_i }, i=1,...N,
X_i ∈ R^p,Y_i ∈ R^q
Options:
• Acceleration (1-5)
• d – reduced dimension
• Black box function BB(X)
• Function Evaluation Budget
• Dimensionality of inputs and outputs
Options:
• d – reduced dimension
• Differentiation Filter Length
• Numerical Difference Step
GT DR (FE mode)
Data Preprocessing
DR Procedure
• Full-dimensional vector X ∈ R^p
• Reduced dimension d (optional)
Compression Module
• Compressed vector Λ ∈ R^d
• Compression gradients
• Compressed vector Λ ∈ R^d
Decompression Module
• Full-dimensional vector X* ∈ R^p
• Decompression gradients

Figure 6.1: **GTDR (FE mode)**. Scheme of Application

As shown in the figure the workflow of **GTDR (FE mode)** tool application consists of three consecutive steps:

1. Prepare and clean up the data. The **GTDR (FE mode)** does some automatic data preprocessing. See section 5.2 for details. This step is omitted in *black-box* implementation.

2. Run the **GTDR (FE mode)** tool to construct data-based effective dimension reduction module. This is the most time-consuming part.

3. Use module built on previous step to compress or decompress the data. At this stage for compression user also may specify the desired value of reduced dimension, otherwise tool estimated dimension used for compression.

## 6.1 Required Parameters

Some parameters are required for the tool. They are listed below.

- In *sample-based interface*:

  **X** : train points;

  **Y** : train values.

- In *black-box-based interface*:

  **BlackBoxFunctionInterface BB(X)** : interface to black box function;

  **FunctionEvaluationBudget** : number of times black-box function is allowed to be called;

  **InputDimensionality** $p$ : the dimensionality of input variables ($X$);

  **OutputDimensionality** $q$ : the dimensionality of output variables ($Y$).

## 6.2 Compression/Decompression Procedures Output

Both *black-box* and *sample* based implementations have unified compression/decompression procedures. The outputs are described below.

- The **compression procedure** performs orthogonal projection of the full-dimensional vectors specified into the Central Mean Subspace estimated by the tool. User may specify the dimensionality of reduced space if he/she has some prior knowledge or rely on the tool dimensionality estimation block. The gradients of this projection provided by the tool is exactly projection matrix $B$ (see Section 4 for details).

- The **decompression procedure** performs orthogonal projection of the vectors specified from the Central Mean Subspace estimated by the tool into original full-dimensional space. The data can have any dimensionality (less than full). The gradients of this projection provided by the tool is exactly transposed projection matrix $B'$ (see Section 4 for details).

## 6.3 Limitations

All limitations of **GTDR (FE mode)** are just reasonable assumptions on initial training dataset. The general restriction on the minimum size $N$ of the training set is

$$N > 2p + 2,$$

where $p$ is the input dimension of the data. As explained in Section 5.2, this condition refers to the *effective* values, obtained after preprocessing of the training data. An error with the corresponding error code will be returned if this condition is violated.

The maximum size of the training sample which can be processed by the tool is primarily determined by the user's hardware. In practice **GTDR (FE mode)** can handle samples of up to 200k points and dimension up to 500 on conventional PC with 4 Gb RAM.

# Chapter 7

# User Configurable Options

This chapter summarizes user options. Note that only a minor amount of options is available in the interface and may be configured by user. These options are mainly intended to control tradeoff between learning time and tool accuracy.

## 7.1 Common options for sample-based techniques

Options in this section are available for all sample-based techniques (sample-based FE, dimension-based and error-based DR).

- **GTDR/Normalize**
  In some cases components of input vector should be normalized, i.e. centered and then standardized by the corresponding standard deviation. Such transformation is useful when components of input vector have different physical meaning (represented in different physical units). By default the need to normalize input is determined automatically, while the option allows user to explicitly enable or disable normalization.
  - true: enable normalization
  - false: disable normalization
  - Auto: automatic normalization

| | |
|---|---|
| Values | boolean, or Auto |
| Default | Auto |

## 7.2 Dimension-based and error-based DR options

Options in this section configure the behaviour of dimension-based and error-based algorithms. These techniques are sample-based.

- **GTDR/MinImprove**
  Dimension-based dimension reduction procedure allows increasing accuracy of reconstruction by approximating nonlinear deviation of reconstructed manifold from the linear hyperplane given by principal components. This approximation is done iteratively and the approximation process is stopped on such iteration, during which the decrease of reconstruction error is less than *MinImprove* times the reconstruction error on the previous iteration.

| Values | double in range $(0, 1]$ |
|---|---|
| Default | 0.01 |

- **GTDR/Technique**
  Specifies the dimension reduction technique.

| Values | enumeration: NLPCA, PCA |
|---|---|
| Default | NLPCA |

## 7.3 Feature extraction options

Options in this section are available in sample-based FE mode.

- **GTDR/Accelerator**
  This option is a five-position switch that controls tradeoff between speed and accuracy. It affects training time by changing default values of other parameters of the algorithm. Possible values are from 1 (low speed, highest quality) to 5 (high speed, lower quality). Default value is 2.

| Values | integer in range $[1, 5]$ |
|---|---|
| Default | 2 |

## 7.4 Feature extraction options (blackbox)

Options in this section are available in FE mode with blackbox.

- **GTDR/DiffFilterSize**
  Sets the length of filter used for numerical differentiation (in general, greater length yields more robust numerical gradients, but also requires more points for each gradient estimating).

| Values | integer in range $[1, 10]$ |
|---|---|
| Default | 1 |

- **GTDR/NumDiffStep**
  Sets relative numerical differentiation step.

| Values | double in range $(0, 0.1]$ |
|---|---|
| Default | $10^{-7}$ |

# Chapter 8

# Usage Examples

In this section we present application of **GTDR (FE mode)** to some artificial toy functions and some real- world data sets to demonstrate method properties. All the tests were performed using **GTDR (FE mode)** tool with **Acceleration** = 3 and default other parameters in sample-based interface.

## 8.1 Artifical Examples

In this section we demonstrate the performance of **GTDR (FE mode)** on some artificial functions.

### 8.1.1 Example 1: Simple Function

At first we will consider the function:

$$f(x_1, x_2, x_3, x_4, x_5) = \left(\sum_{i=1}^{5} x_i\right)(x_1 + x_2),\ x_i \in [0,\ 1], i = 1, \ldots, 5. \tag{8.1}$$

According to **GTDR (FE mode)** problem statement in this case $X$ can be compressed from dimension 5 to dimension 2 without loss of it's descriptive power. Of course one doesn't know true data dimensionality beforehand, so in this case we will demonstrate **GTDR (FE mode)** performance on this data depending on sample size and dimension we decided to compress data to.

As a measure of method's quality we will use the following variability index measure (see Section 3.1.1 for more details on the performance measures):

$$I(\mathbf{X}, \mathbf{Y}, \Sigma) = \left\langle \left\| Y - f(X^* (\Lambda (X)) \right\| \right\rangle \tag{8.2}$$

where $\mathbf{X}, \mathbf{Y}$ - test data sample, $\Sigma = \{p, d, \mathcal{C} : X \to \Lambda(X),\ \mathcal{D} : \Lambda \to X^*(\Lambda)\}$ is the constructed **GTDR (FE mode)** procedure, $< .. >$ is the sample mean.

Results of these experiments are presented in the table 8.1.

Looking at the result one may notice that difference between function value computed for original $X$ and compressed and then reconstructed $X$ becomes quite small when we reach true dimension of the data and in this simple example doesn't grow if we select dimension greater than needed value. This example highlights one of the important method property: if the assumption on the data structure approximately holds, i.e. $f(X) \approx g(XB^T) + \varepsilon$, $E(\varepsilon) = 0$,

| Sample size | Compressed dimension | | | | | |
|---|---|---|---|---|---|---|
| | 0 | 1 | 2 | 3 | 4 | 5 |
| 30 | 1,2457e-00 | 7,3653e-01 | 2,4325e-04 | 2,4251e-04 | 3,6581e-05 | 4,4408e-15 |
| 100 | 1,2457e-00 | 6,9524e-01 | 3,4734e-05 | 3,2909e-05 | 2,9697e-05 | 4,4408e-15 |
| 200 | 1,2457e-00 | 7,7082e-01 | 1,1968e-04 | 7,6521e-05 | 3,9005e-05 | 4,4408e-15 |

Table 8.1: **GTDR (FE mode)** variability index measure on artificial function 1

then in order to keep all the information in the inputs it's enough to compress to any dimension not smaller than the true one, and generally the more dimensions are kept the smaller the error is.

### 8.1.2 Example 2a: More Complex Function

Another example is a little bit more complex function:

$$f(x_1, x_2, x_3, x_4, x_5) = \left(\sum_{i=1}^{5} x_i\right) \left((2x_1 + x_3)^2 + 0.1x_2\right) + x_4^3,\ x_i \in [0,\ 1], i = 1, \ldots, 5. \quad (8.3)$$

One may see that this function can be approximately compressed to dimension 3 and with good precision to dimension 4. Results of the experiments with different reduced dimensions and sample size are presented in Table 8.2. The same index of variability as in the previous section is used for **GTDR (FE mode)** performance measure.

| Sample | Compressed dimension | | | | | |
|---|---|---|---|---|---|---|
| size | 0 | 1 | 2 | 3 | 4 | 5 |
| 30 | 5,2687 | 1,7321 | 1,4648 | 0,7108 | 0,5409 | 2,1316e-14 |
| 100 | 5,2687 | 1,5188 | 1,5410 | 0,1657 | 0,0007 | 3,9079e-14 |
| 200 | 5,2687 | 1,3644 | 0,8532 | 0,1882 | 0,0002 | 5,6843e-14 |

Table 8.2: **GTDR (FE mode)** variability index measure on artificial function 2

As we increase compressed dimension similar behavior as before is observed. Also notice that although 30 points seem to be not enough to accurately catch data structure, the error still decreases monotonically even in this case.

### 8.1.3 Example 2b: More Complex Function

Consider in previous example we have the black-box function instead of the initial dataset. We can apply *black-box* implementation of **GTDR (FE mode)** to estimate CMS. The corresponding variability indices (the same as in Section 8.1.2) are represented in Table 8.3. Instead of initial sample size we used the same function evaluation budget.

| Budget | Compressed dimension | | | | | |
|---|---|---|---|---|---|---|
| size | 0 | 1 | 2 | 3 | 4 | 5 |
| 30 | 5,2687 | 0,2244 | 0,2172 | 0,1196 | 0,1083 | 6,6407e-16 |
| 100 | 5,2687 | 0,0599 | 0,0535 | 0,0036 | 6,2794e-10 | 2,38778e-15 |
| 200 | 5,2687 | 0,0991 | 0,0709 | 0,0022 | 4,4866e-11 | 2,6151e-15 |

Table 8.3: **GTDR (FE mode)** variability index measure on artificial function 2 for Black-Box implementation

This example compared to the example in Section 8.1.2 shows that in case user has black-box function for data generation it may be preferable to plug it directly into the **GTDR (FE mode)** (instead of train data sample generation).

### 8.1.4 Example 3: Simple Function with Noise

And the last example would be:

$$f(x_1, x_2, x_3, x_4, x_5) = \left(\sum_{i=1}^{5} x_i\right)(x_1 + x_2) + \varepsilon,\ x_i \in [0,\ 1], i = 1, \dots 5, \tag{8.4}$$

$\varepsilon$ is the normally distributed value with zero mean, $E[\varepsilon] = 0$, and variance $\sigma^2 = 0.1$.

In this example we will check how noise in data affects the results. Computed errors are presented in the table 8.4.

| Sample | Compressed dimension | | | | | |
|---|---|---|---|---|---|---|
| size | 0 | 1 | 2 | 3 | 4 | 5 |
| 30 | 1,2457 | 0,6795 | 0,0142 | 0,0295 | 0,0207 | 0,0517 |
| 100 | 1,2457 | 0,6678 | 0,0013 | 0,0822 | 0,0110 | 0,0248 |
| 200 | 1,2457 | 0,7685 | 0,0617 | 0,0602 | 0,0399 | 0,0811 |

Table 8.4: **GTDR (FE mode)** variability index measure on artificial function 3

One may see that noise affects the results of the procedure, but estimated errors are actually comparable to the noise value, meaning that procedure results are quite robust to the noise.

## 8.2 Real world data examples

In this section we will show application of **GTDR (FE mode)** to some real world data problems.

### 8.2.1 T-AXI problem

- **Problem description:**
  In this problem we consider The T-C_DES (Turbomachinery Compressor DESign) code (meanline axial flow compressor design tool), which is the first step of T-AXI (an axisymmetric method for a complete turbomachinery geometry design [12]).

  Program tcdes.e3c-des.exe is used for calculation of outputs $f(X)$ for new generated inputs $X$. Program can be downloaded from the link:
  *http://gtsl.ase.uc.edu/T-AXI/*.

  Program uses a 163 dimensional feature vector describing geometry and the working condition as an input.

  The task is to compress features the way that Compressor Pressure Ratio (With IGV) output can be reconstructed with sufficient precision. The dependency is considered only for $X \in V(X^0) = \{X : x^i \in [(1-\alpha)x_i^0, (1+\alpha)x_i^0]\}$, $i = 1, .., 163$ where $\alpha = 0.1$, $X^0 = (x_1^0, .., x_{163}^0)$ is given in Tables 8.5 – 8.7.

| | **Stage** | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| **Parameter** | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 |
| Stage rotor inlet angle [deg] | 10,3 | 13,5 | 15,8 | 18 | 19,2 | 19,3 | 16,3 | 15 | 13,6 | 13,4 |
| Stage rotor inlet Mach no. | 0,59 | 0,51 | 0,475 | 0,46 | 0,443 | 0,418 | 0,402 | 0,383 | 0,35 | 0,313 |
| Total Temperature Rise [K] | 52,696 | 52,301 | 51,117 | 49,736 | 49,144 | 43,617 | 45,69 | 47,269 | 48,255 | 47,565 |
| Rotor loss coef. | 0,053 | 0,0684 | 0,0684 | 0,0689 | 0,069 | 0,069 | 0,069 | 0,069 | 0,069 | 0,07 |
| Stator loss coef. | 0,07 | 0,065 | 0,065 | 0,06 | 0,06 | 0,065 | 0,065 | 0,065 | 0,065 | 0,1 |
| Rotor Solidity | 1,666 | 1,486 | 1,447 | 1,38 | 1,274 | 1,257 | 1,31 | 1,317 | 1,326 | 1,391 |
| Stator Solidity | 1,353 | 1,277 | 1,308 | 1,281 | 1,374 | 1,474 | 1,379 | 1,276 | 1,346 | 1,453 |
| Stage Exit Blockage | 0,963 | 0,956 | 0,949 | 0,942 | 0,935 | 0,928 | 0,921 | 0,914 | 0,907 | 0,9 |
| Stage bleed [%] | 0 | 0 | 0 | 0 | 1,3 | 0 | 2,3 | 0 | 0 | 0 |
| Rotor Aspect Ratio | 2,354 | 2,517 | 2,33 | 2,145 | 2,061 | 2,028 | 1,62 | 1,417 | 1,338 | 1,361 |
| Stator Aspect Ratio | 3,024 | 2,98 | 2,53 | 2,21 | 2,005 | 1,638 | 1,355 | 1,16 | 1,142 | 1,106 |
| Rotor Axial Velocity Ratio | 0,863 | 0,876 | 0,909 | 0,917 | 0,932 | 0,947 | 0,971 | 0,967 | 0,98 | 0,99 |
| Rotor Row Space Coef. | 0,296 | 0,4 | 0,41 | 0,476 | 0,39 | 0,482 | 0,515 | 0,58 | 0,64 | 0,72 |
| Stator Row Space Coef. | 0,3 | 0,336 | 0,438 | 0,441 | 0,892 | 0,455 | 0,886 | 0,512 | 0,583 | 0,549 |
| Stage Tip radius [m] | 0,3507 | 0,3358 | 0,3283 | 0,3212 | 0,3151 | 0,3084 | 0,3042 | 0,2995 | 0,297 | 0,2946 |

Table 8.5: Stage data for 10 stage design (stage.e3c-des)

| | |
|---|---|
| Mass Flow Rate [kg/s] | 54,4 |
| Rotor Angular Velocity [rpm] | 12299,5 |
| Inlet Total Pressure [Pa] | 101325 |
| Inlet Total Temperature [K] | 288,15 |
| Mach 3 - Last Stage | 0,272 |
| Clearance Ratio | 0,0015 |

Table 8.6: Initial data for 10 stage design (init.e3c-des)

- **Solution workflow:**
  We perform the following steps to perform the analysis:

  1. We generate data sample of 500 uniformly distributed random points within the region $V(X^0)$.
  2. The **GTDR (FE mode)** procedure is constructed using the generated sample.

| | |
|---|---|
| Soldity | 0,6776 |
| Aspect ratio | 5,133 |
| Phi Loss Coef. | 0,039 |
| Inlet Mach | 0,47 |
| Lambda | 0,97 |
| IGV Row Space Coef. | 0,4 |

Table 8.7: IGV data for 10 stage design (igv.e3c-des)

3. To check the performance of **GTDR (FE mode)** Index of Variability is calculated for different compression dimensions. Index of variability is computed as follows (see Section 3.1.1):

$$I(d) = \frac{\sqrt{\frac{1}{N}\sum_{i=1}^{N}(Y_i - f(X^*(\Lambda_d(X_i))))^2}}{\max_{i=1,..,N}(Y_i) - \min_{i=1,..,N}(Y_i)} 100\%, \tag{8.5}$$

where $M$ is the size of the test sample ($N = 10000$), $\Lambda_d$ is $d$-dimensional representation of $X \in R^{163}$. The plot of the index against the reduced dimension is represented on the Figure 8.1

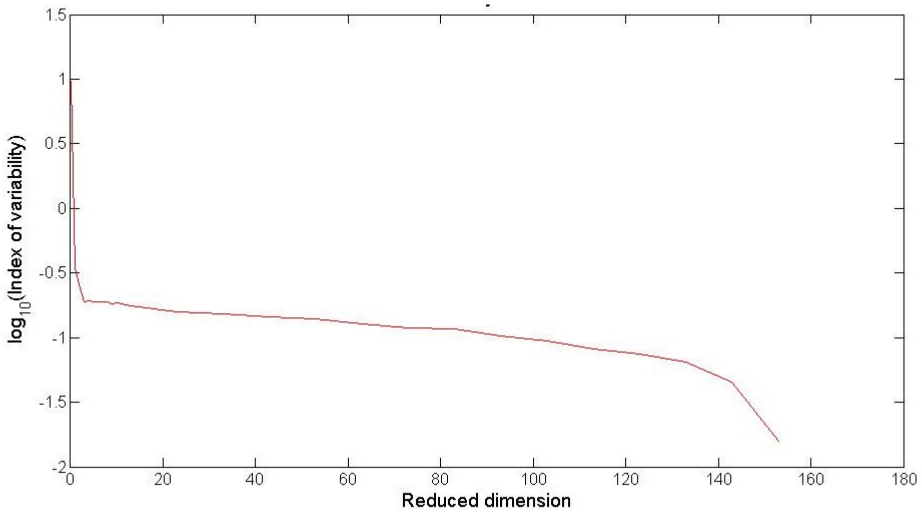


Figure 8.1: T-AXI. Index of variability for different compression dimensions $d$

- **Results:**
Plot shows that reducing dimension up to 2 keeps index of variability at 0.3%. Therefore the input dimension could be reduced from very high number of 163 variables down to 2 most representative features without significant loss of information.

### 8.2.2 Airfoil optimization problem

- **Problem description:**
This problem is devoted to optimization of a wing section airfoil. An airfoil is given by 59-dimensional vector consisting of coordinates of special points on airfoil surface. The

goal of optimization is to find minimum drag coefficient in a transonic flight regime subject to given lift coefficient and constrained spar width:

$$\min_{X \in R^{59}} CD(X, \theta(X)) \; s.t. \begin{cases} |CL(X,\theta) - CL_0| < \varepsilon, \\ t_{fs}(X) \geq t^0_{fs}, \\ t_{rs}(X) \geq t^0_{rs}, \\ \theta \in [\theta_{min}, \theta_{max}], \end{cases}$$

where

- $X$ is the description of an airfoil;
- $CD(X,\theta)$ is the drag coefficient;
- $CL(X,\theta)$ and $CL_0$ is the lift coefficient and it's target value respectively;
- $t_{fs}$, $t^0_{fs}$, $t_{rs}$ and $t^0_{rs}$ are the widths of the front and rear spar and their lower bounds respectively;
- $\theta$, $\theta_{min}$and $\theta_{max}$ are the trailing edge angle and its minimum and maximum values respectively.

To compute $CL(X,\theta)$ and $CD(X,\theta)$ an aerodynamic full-potential $2D$ solver is used. Solver is based on $2D$ full-potential model of invised compressible fluid (see [8] for details).

- **Solution workflow:**
  We perform the following steps to perform the analysis:

  1. Before actual optimization we perform some preliminary data transformation using two approaches:
     - Principal Component Analysis (PCA) of initial dataset in order to compress data to some low-dimensional subspace (as optimization in high-dimensional space is very ineffective) preserving structural features of an airfoil.
     - PCA that transforms original data into full-dimensional basis of principal components followed by **GTDR (FE mode)** procedure (PCA+FE) compressing data to some low-dimensional feature subspace.
  2. Then the optimization procedure is performed in the low-dimensional space extracted by one of the described approaches starting from some random initial point.

  Additional to the optimization experiment some approximation experiment was performed. The dataset of 1500 airfoils was compressed to some low-dimensional space and an approximator that uses the compressed representation as inputs and $CD$ as output was constructed by **GT Approx**. The errors of approximation were measured on an independent dataset. The results are represented in the table 8.8.

  Optimization was performed by **GT Opt** starting from 20 random (but the same for the both approaches) points in the space of dimension 10. The results are represented in the table 8.9.

- **Results:**
  Approximation results show that **GTDR (FE mode)** tool allows to reduce dimension of feature space from 10 down to 3 almost without loss of regression information.

| Method | Reduced dimension | Mean Absolute Error | Mean Squared Error |
|---|---|---|---|
| PCA | 2 | 0.0033 | 2.0568e-5 |
| PCA + GT (FE) | 2 | 9.6621e-4 | 2.0655e-6 |
| PCA + GT (FE) | 3 | 6.5416e-4 | 1.0901e-6 |
| PCA | 10 | 5.3667e-4 | 9.0450e-7 |

Table 8.8: Airfoil optimization problem. Approximation performance in the reduced-dimension spaces

| Method | Mean(CD) | Std(CD) | Min(CD) | Max(CD) |
|---|---|---|---|---|
| PCA | 0.0132 | 0.0030 | 0.0065 | 0.0189 |
| PCA + GT (FE) | 0.0067 | 0.0034 | 0.0037 | 0.0168 |

Table 8.9: Airfoil optimization problem. Optimization performance in the reduced-dimension spaces

Optimization results provide more evidence in favor of much better information preservation of **GTDR (FE mode)** than original PCA. This gives an opportunity to achieve better optimization results given the same time budget (150 computations of objective function in the considered case).

### 8.2.3 Fuel System Analysis problem

- **Problem description:**
  The objective of the Research into Fuel Systems project is to deliver application that can predict pressures and mass flows for gravity feed aircraft fuel systems [9]. The desktop application comprises a two phase flow (air and fuel) analysis engine that is derived from experimental observations.

  One of the task the pSeven Core models are used for in this project is to approximate pressure loss coefficient and volume flow quality of the fuel flow on the diaphragm section of the pipe using experimental data.

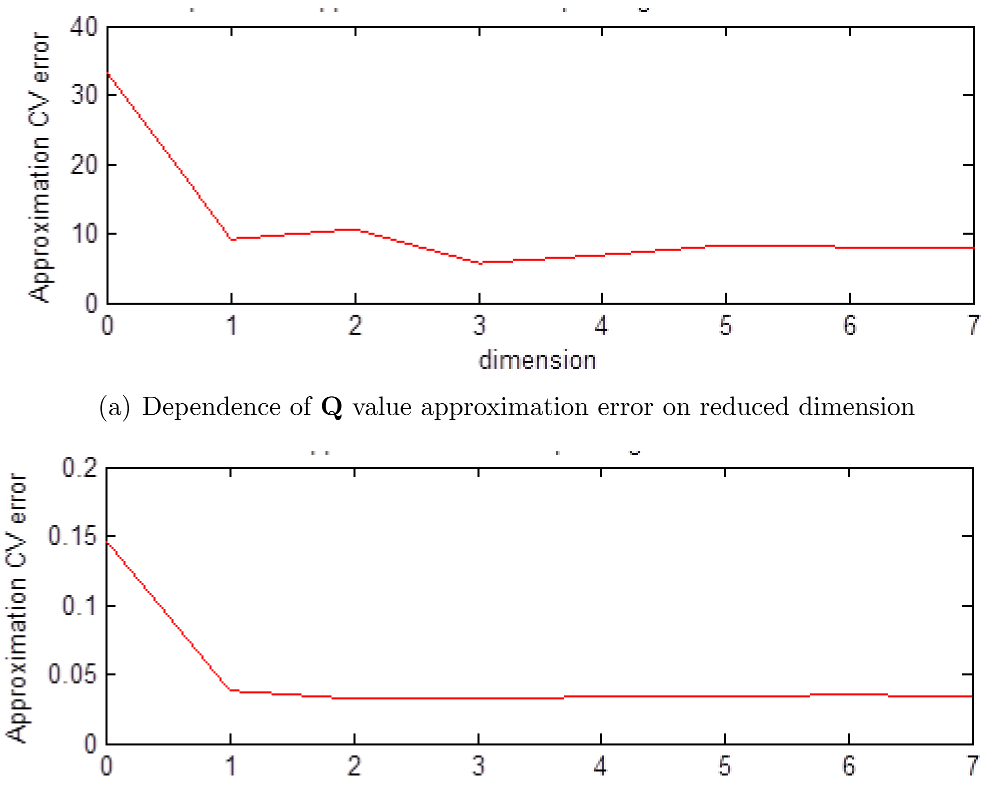
  Experimental data is a 244 points sample with 6 features describing fuel flow (flow velocity ($\mathbf{V}$), pressure after the diaphragm ($\mathbf{P}$), temperature ($\mathbf{T}$), densities of fuel ($\rho_{\mathbf{fuel}}$) and air ($\rho_{\mathbf{air}}$), ratio of diaphragm diameters ($\mathbf{r}_i$)) and two outputs pressure loss coefficient ($\mathbf{C_p}$) and volume flow quality ($\mathbf{Q}$).

  We will use **GTDR (FE mode)** to determine which features should be measured with the most accuracy. This is very important for experimental design: if the feature is unimportant then we shouldn't do additional expensive experiments in order to explore the dependence of the outputs ($\mathbf{C_p}$ and $\mathbf{Q}$) on this feature, and we can measure this feature with less precision in the experiments.

- **Solution workflow:**
  We perform the following steps to perform the analysis:

1. We use given sample of $M = 244$ points with experimental data.
2. The **GTDR (FE mode)** procedure is constructed using the generated sample.
3. To check the performance of **GTDR (FE mode)** surrogate models were build on compressed data, see Section 3.1.1.

The results are shown on the figure 8.2.



(a) Dependence of **Q** value approximation error on reduced dimension

(b) Dependence of $\mathbf{C_p}$ value approximation error on reduced dimension

Figure 8.2: Fuel System Analysis problem results. **Note:** This image was obtained using an older pSeven Core version. Actual results in the current version may differ.

- **Results:**
  Plot shows that to approximate $\mathbf{C_p}$ with precision better than when using all input variables its enough to take only 3 features constructed **GTDR (FE mode)** and to approximate **Q** with precision better than when using all variables its enough to take only 1 feature constructed by **GTDR (FE mode)**.

# Bibliography

[1] E. Burnaev, M. Belyaev, and P. Prihodko. Estimation of effective dimension reduction space for function approximation. In *Proceedings of the 33th Conference for Young Scientists and Specialists "ITaS-2010"*, pages 189–194, 2010.

[2] R. Cook. SAVE: A method for dimension reduction and graphics in regression. *Communications in statistics. Theory and methods*, 29(9-10):2109–2121, 2000.

[3] R. D. Cook and B. Li. Determining the dimension of iterative Hessian transformation. *The Annals of Statistics*, 32(6):2501–2531, Dec. 2004.

[4] A. Dalalyan, A. Juditsky, and V. Spokoiny. A new algorithm for estimating the effective dimension-reduction subspace. *The Journal of Machine Learning Research*, 9:1647–1678, 2008.

[5] M. Forina. UCI machine learning repository, 1991.

[6] I. Guyon. *Feature Extraction*, volume 207 of *Studies in Fuzziness and Soft Computing.* Springer Berlin Heidelberg, 2006.

[7] T. Hastie, R. Tibshirani, and J. Friedman. *The elements of statistical learning: data mining, inference, and prediction.* Springer, 2008.

[8] A. Jameson. Iterative solution of transonic flows over airfoils and wings, including flows at mach 1. *Communications on Pure and Applied Mathematics*, 27:283–309, 1974.

[9] E. Kitanin. Air Evolution Research in Fuel Systems 4. Technical report, IRIAS, 2010.

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

[11] H.-H. Lue. On principal Hessian directions for multivariate response regressions. *Computational Statistics*, 25(4):619–632, Mar. 2010.

[12] M. Turner. A turbomachinery design tool for teaching design concepts for axial-flow fans compressors and turbines. *Proceedings of GT2006*, 2006.

[13] H. Wang and Y. Xia. Sliced Regression for Dimension Reduction. June 2008.

[14] Q. Wu and S. Mukherjee. Variable Selection and Dimension Reduction by Learning Gradients. *Journal of Machine Learning Research*, (2003):1–27, 2008.

[15] Y. Xia, H. Tong, W. K. 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, Aug. 2002.

# Index

# Index: Options