Effective Dimension Reduction based on Gaussian Processes

Pavel Prikhodko, Pavel Erofeev

Institute for Information Transmission Problems
PreMoLab
Datadvance

June 22, 2012

DATADVANCE
AN EADS COMPANY

Dependency Recovery Problem Statement

- Let the dependency be in the following form:

$$y = f(\mathbf{x}), \tag{1}$$

where $y \in \mathbb{R}, \mathbf{x} \in \mathbb{X} \subset \mathbb{R}^m$, $f(\mathbf{x})$ — some function.

- We seek to recover the dependency (1) using the dataset $D = (X, \mathbf{y}) = \left\{(\mathbf{x}_i, y_i = f(\mathbf{x}_i))\right\}_{i=1}^{n}$:

$$y = \hat{f}(\mathbf{x}).$$

- Approximation quality is generally estimated as root mean squared error on the test (independent) dataset $D' = \left\{(\mathbf{x}'_j, y'_j = f(\mathbf{x}'_j))\right\}_{j=1}^{n'}$:

$$Q(\hat{f}) = \frac{1}{n'} \sum_{j=1}^{n'} (y'_j - \hat{f}(\mathbf{x}'_j))^2.$$

Additional Assumptions

Let a random field $f(\mathbf{x}, \omega)$ given, $\omega \in \Omega$ — a random event in Ω. It is assumed that for arbitrary $\mathbf{x} \in \mathbb{X}$ there exist first and second moments:

$$\mathsf{M}(\mathbf{x}) = \mathbb{E} f(\mathbf{x}),$$

$$\mathsf{K}(\mathbf{x}_1, \mathbf{x}_2) = \mathbb{E}(f(\mathbf{x}_1) - \mathbb{E} f(\mathbf{x}_1))(f(\mathbf{x}_2) - \mathbb{E} f(\mathbf{x}_2)),$$

and also conditional expectation

$$\mathbb{E}(f(\mathbf{x}) | f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_l)).$$

Additionally assume that the random field is Gaussian. For those field joint distribution $f(\mathbf{x}_1), f(\mathbf{x}_2), \ldots, f(\mathbf{x}_l)$ — is normal and thus can be defined by expectation and covariance matrix.

Covariance Matrix Form

Let the dependency $y(\mathbf{x})$ be described by:

$$y(\mathbf{x}) = f(\mathbf{x}) + \varepsilon(\mathbf{x}),$$

where $f(\mathbf{x})$ — some realization of Gaussian random field, and $\varepsilon(\mathbf{x})$ — uncorrelated Gaussian noise with variance σ_1^2.
Assume that covariance function $K_0(\mathbf{x}, \mathbf{x}')$ of the Gaussian field $f(\mathbf{x})$ belongs to some parametric family:

$$K_0(\mathbf{x}, \mathbf{x}') = \sigma_0^2 K_0(\mathbf{x}, \mathbf{x}'|\Theta),$$

where Θ — some parameter set, σ_0^2 - scale parameter of covariance function.
Then covariance function of the process $y(\mathbf{x})$ takes form:

$$K(\mathbf{x}, \mathbf{x}') = K_0(\mathbf{x}, \mathbf{x}') + \sigma_1^2 \delta(\mathbf{x}, \mathbf{x}'),$$

where $\delta(\mathbf{x}, \mathbf{x}')$ — Kronecker symbol.

Gaussian Processes

now the distribution of $y(\mathbf{x})$ takes form:

$$Law(y(\mathbf{x})|y(\mathbf{x}_1), y(\mathbf{x}_2), \ldots, y(\mathbf{x}_n)) \sim \mathcal{N}(\hat{f}(\mathbf{x}), \hat{\sigma}^2(\mathbf{x}))$$

where expectation $\hat{f}(\mathbf{x})$:

$$\hat{f}(\mathbf{x}) = \mathsf{E}(y(\mathbf{x})|y(\mathbf{x}_1), y(\mathbf{x}_2), \ldots, y(\mathbf{x}_n)) = \mathbf{k}(\mathbf{x})\mathrm{K}^{-1}\mathbf{y},$$

where vector $\mathbf{k}(\mathbf{x}) = \{K(\mathbf{x}, \mathbf{x}_i)\}_{i=1}^n$, matrix $\mathrm{K} = \{K(\mathbf{x}_i, \mathbf{x}_j)\}_{i,j=1}^n$.
Variance estimation takes form:

$$\hat{\sigma}^2(\mathbf{x}) = \mathsf{D}(y(\mathbf{x})|y(\mathbf{x}_1), y(\mathbf{x}_2), \ldots, y(\mathbf{x}_n)) =$$

$$= K_0(\mathbf{x}, \mathbf{x}) + \sigma_1^2 - \mathbf{k}(\mathbf{x})\mathrm{K}^{-1}\mathbf{k}(\mathbf{x})^T.$$

Note, that approximation based on Gaussian processes takes form: $\hat{f}(\mathbf{x}) = \sum_{i=1}^n \alpha_i K(\mathbf{x}, \mathbf{x}_i')$. So the distance choice in covariance function directly influences the form of approximation.

Parameter Choice

To estimate covariance function parameters employ maximum likelihood principle: $\mathbf{a} = \left\{\Theta, \sigma_0, \sigma_1\right\}$. The likelihood in this case takes form:

$$\log p(\mathbf{y}|X, \mathbf{a}) = -\frac{1}{2}\mathbf{y}^T \mathrm{K}^{-1}\mathbf{y} - \frac{1}{2}\log|\mathrm{K}| - \frac{n}{2}\log 2\pi,$$

where $|\mathrm{K}|$ — determinant of matrix K.
Parameters $\mathbf{a}$ are selected via maximizing log-likelihood:

$$\log p(\mathbf{y}|X, \mathbf{a}) \to \max_{\mathbf{a}}.$$

Exponential Covariance Function

Use exponential family to model covariance function:

$$K_0(\mathbf{x}, \mathbf{x}') = \sigma_0^2 \exp(-d(\mathbf{x}, \mathbf{x}')),$$

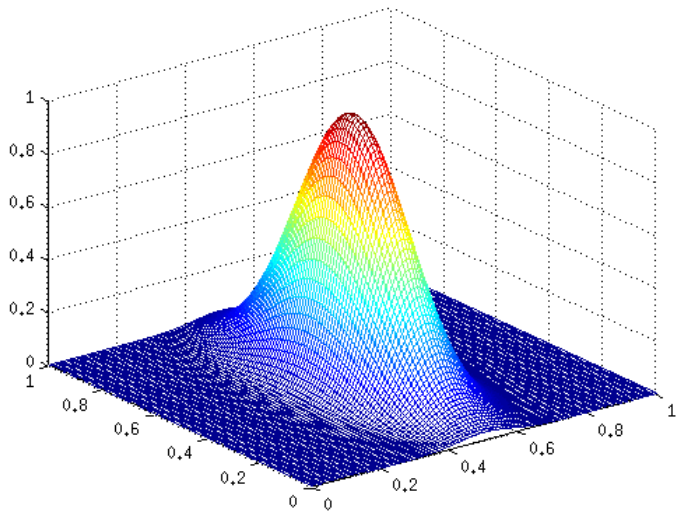
where $d(\mathbf{x}, \mathbf{x}')$ — distance between vectors $\mathbf{x}$ $\mathbf{x}'$ in some metrics specified.

Weighted Euclidean Distance

Weighted Euclidean distance:

$$d(\mathbf{x}, \mathbf{x}') = \sum_{i=1}^{n} \theta_i^2 (x_i - x_i')^2,$$

where $\theta_i \in \mathbb{R}, i = 1, \ldots, m$;
When using such metrics m hyperparameters are to be tuned.

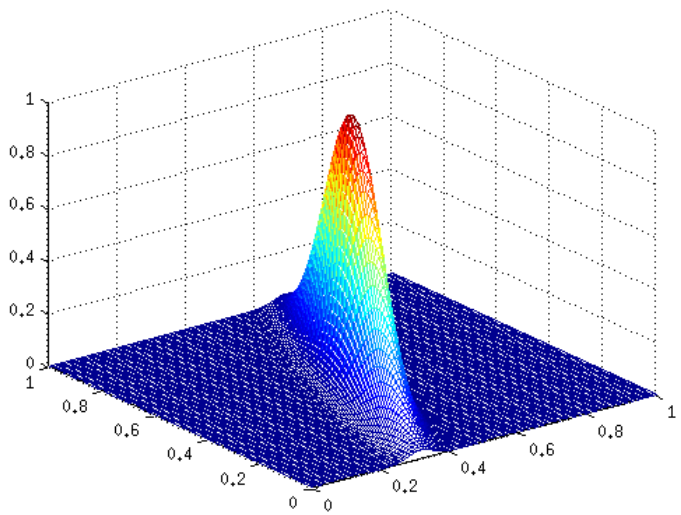


Mahalanobis Distance

Mahalanobis distance:

$$d(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T A (\mathbf{x} - \mathbf{x}'),$$

where $A \in \mathbb{R}^{m \times m}$ - some positively definite matrix.



Mahalanobis Distance

When constructing the model as A is positively definite one can use Cholesky decomposition

$$A = L^T L,$$

where L — upper triangular matrix with positive elements on the main diagonal.
In this case $\frac{m(m+1)}{2}$ hyperparameters are to be tuned.

The influence of axes rotation on the approximation quality

Consider a 2-dimensional vectors $\mathbf{x}$.
Let the function take form: $y = f_B(\mathbf{x}) = f(\mathbf{xB})$, where $\mathbf{B}$ — some orthogonal rotation matrix of size 2×2.
Explore the dependency of approximation quality for different rotation matrices B and different metrics types.

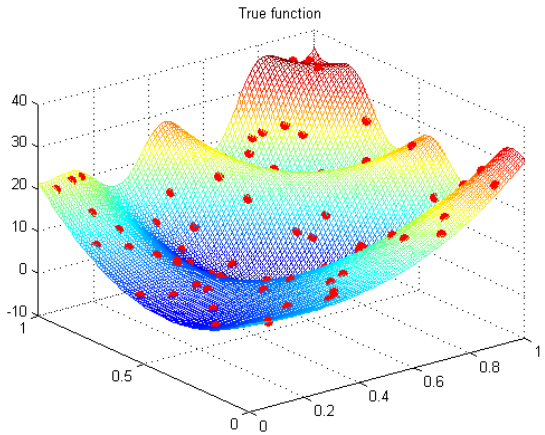
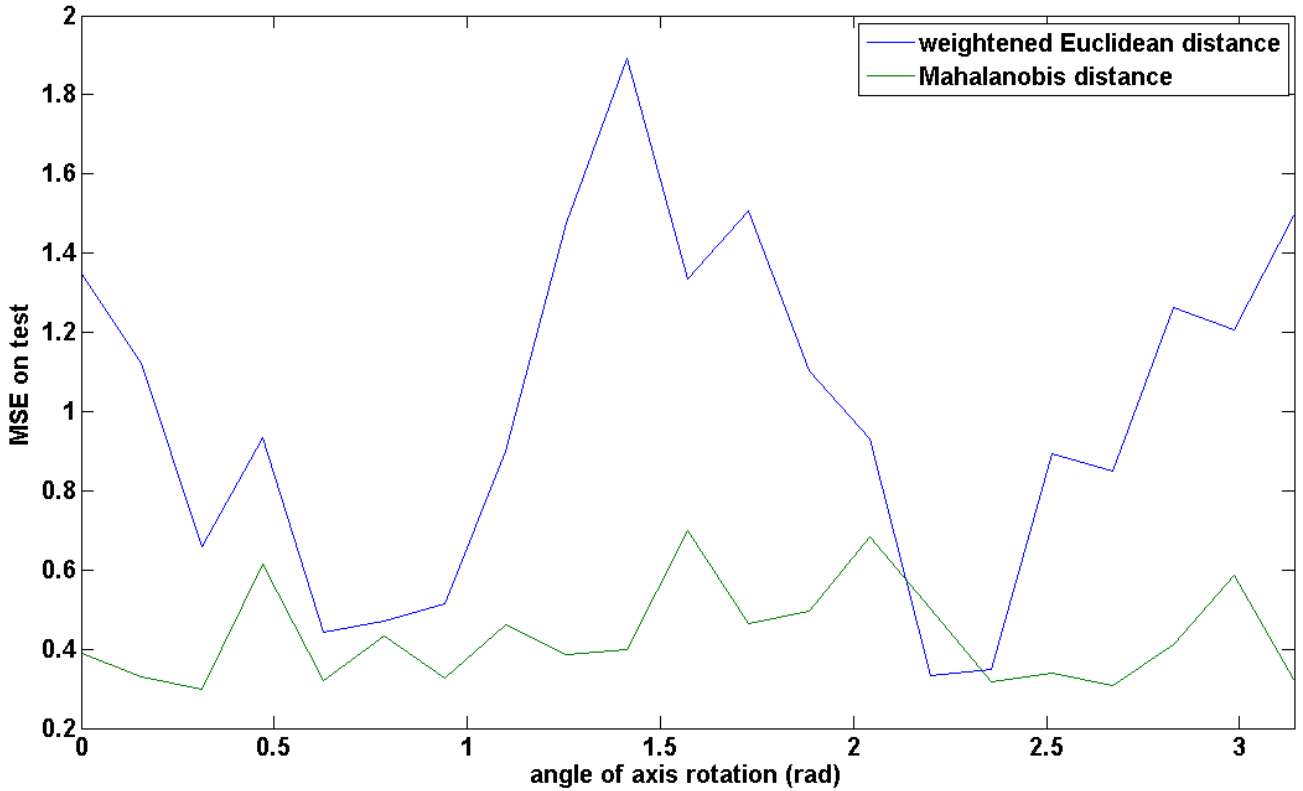


Figure: Mystery

Rotation of axes

The dependency of approximation error (MSE on test set) on the axis rotation angle.



Dolan-Moré

- T problems, A approximation algorithms,
- e_{ta} — approximation error of a'th algorithm on t'th problem.
- $\tilde{e}_t = \min_a e_{ta}$.

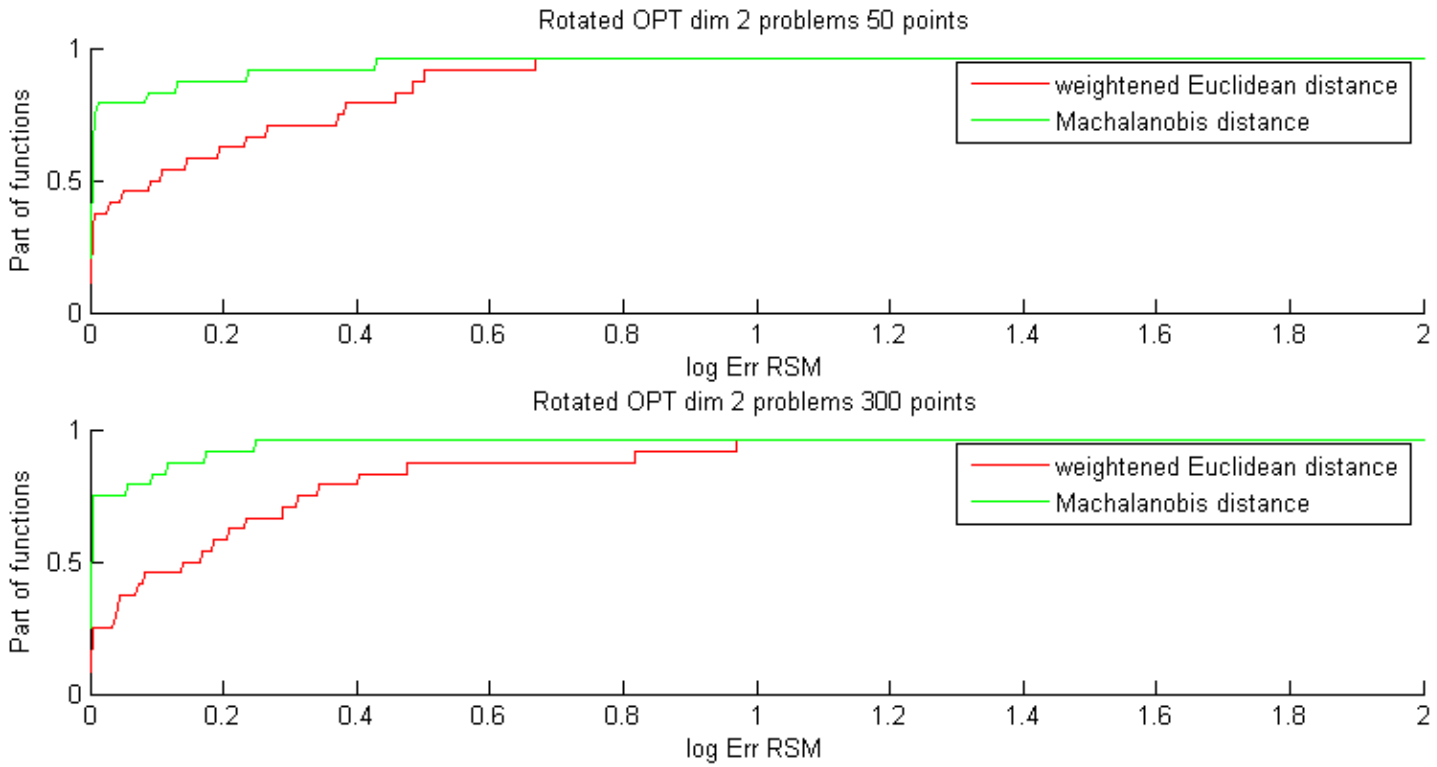
$$p_a(h) = \frac{\#\{t : e_{ta} < h\tilde{e}_t\}}{T}$$

- The higher curve lies — the better.
- $p_a(1)$ — the fraction of problems where algorithm a has the best quality.

On the figures $log(h)$ is plotted along y-axis.

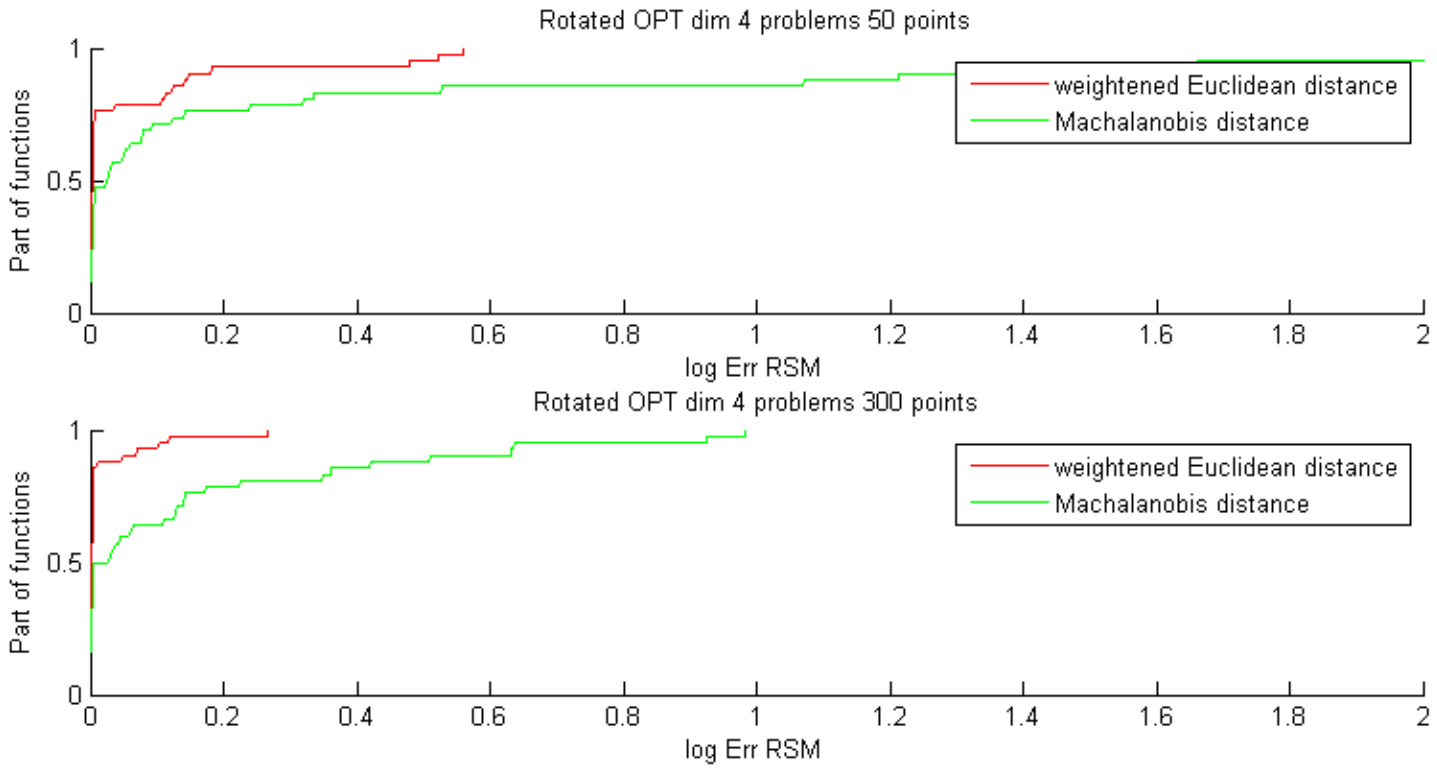
Rotation influence on large set of 2D problems

Test with axis rotation on 28 functions, general for global optimization algorithms testing (e.g., Branin, Shekel, Shubert, Rastrigin, Camelback, etc.).



Rotation influence on large set of 4D problems

Test with axis rotation on 16 functions, general for global optimization algorithms testing (e.g., Colville, Powell, Ackley, etc.).



Findings

One can propose several statements based on the tests:

- Mahalanobis distance usage instead of weighted Euclidean allows to significantly reduce axis rotation influence on the approximation quality (at least in 2D case).
- As dimensionality grows the quality of approximation constructed using Mahalanobis distance is tending to decrease with respect to those constructed using weighted Euclidean distance.

One of the possible explanations of such behaviour is that $\frac{m(m+1)}{2}$ hyperparameters need to be tuned in Mahalanobis case against m hyperparameters in weighted Euclidean distance case.

VEGA (Variable Extraction via Gradient Approximation)

Let us define the distance in covariance function in the following way:

$$d(\mathbf{x}, \mathbf{x}') = (\mathbf{x} - \mathbf{x}')^T B^T \Lambda B (\mathbf{x} - \mathbf{x}'),$$

where B — orthogonal matrix of size $m \times m$, Λ — diagonal positive definite matrix of size $m \times m$.

Note: Matrix A from Mahalanobis distance can be transformed to such form via Singular Value Decomposition.

VEGA (Variable Extraction via Gradient Approximation)

Given the parametrization:

B — performs axis rotation;

Λ — defines kernel width along new axis

We seek for approximation $\hat{f}$ in the form

$$\hat{f}(\mathbf{x}) = \hat{g}(\mathbf{x}B),$$

where $\hat{g}$ is an approximation based on Gaussian processes with covariance function defined by weighted Euclidean distance.

VEGA (Variable Extraction via Gradient Approximation)

Only hyperparameters of $\hat{g}$ can be estimated via maximum likelihood (diagonal matrix Λ).
Matrix B can be estimated from the following considerations.
Let some approximation $\hat{f}$ be already constructed, thus we can estimate gradients of f as

$$\hat{\Gamma} = \left\{ \frac{\partial \hat{f}(\mathbf{x})}{\partial \mathbf{x}^1}\Big|_{\mathbf{x}=\mathbf{x}^i}, \cdots \frac{\partial \hat{f}(\mathbf{x})}{\partial \mathbf{x}^p}\Big|_{\mathbf{x}=\mathbf{x}^i} \right\}_{i=1}^{N}$$

.
We can rotate axis in such way that gradients along axis are linearly uncorrelated. That is define B as eigenvectors of correlation matrix $\hat{\Gamma}^T\hat{\Gamma}$

Basic Algorithm

- Construction of approximation $\hat{f}$ based on Gaussian processes

$$\left\{\mathbf{x}_i, y_i\right\}_{i=1}^{N}, \mathbf{x}_i \in \mathbb{R}^m, y_i \in \mathbb{R}$$

- Gradient estimation

$$\hat{\Gamma} = \left\{ \frac{\partial \hat{f}(\mathbf{x})}{\partial \mathbf{x}^1}\Big|_{\mathbf{x}=\mathbf{x}^i}, \dots \frac{\partial \hat{f}(\mathbf{x})}{\partial \mathbf{x}^p}\Big|_{\mathbf{x}=\mathbf{x}^i} \right\}_{i=1}^{N}$$

  in train set points
- Gradient covariance matrix estimation $\Sigma_{\hat{\Gamma}} = \hat{\Gamma}^T\hat{\Gamma}$
- calculation of rotation matrix B compound by eigenvectors of $\Sigma_{\hat{\Gamma}}$

Iterative Scheme

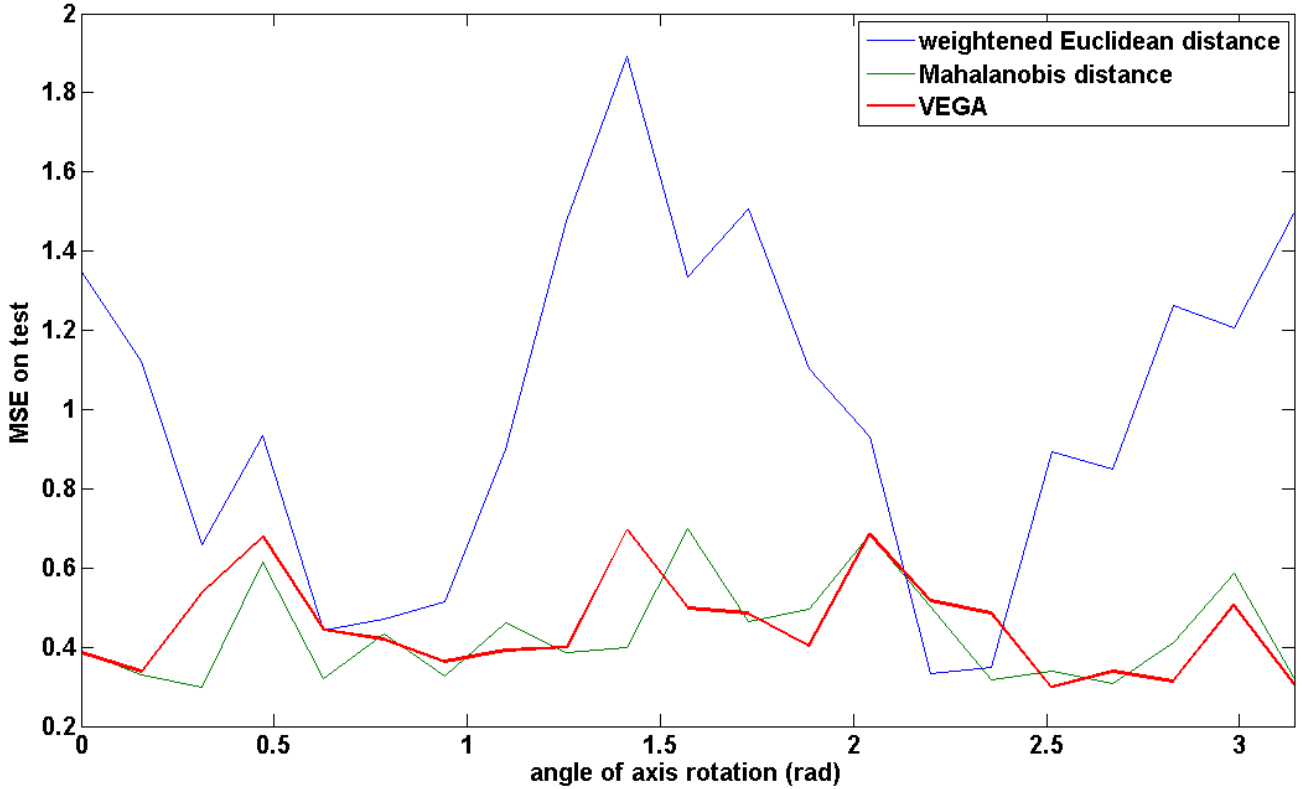
One can additionally increase quality of this approach by repeating.
On the first step we apply basic algorithm to obtain B_1 and approximation $\hat{g_1}(\mathbf{x}^{(1)})$, where $\mathbf{x}^{(1)} = \mathbf{x}B_1$.
On the second step matrix B_2 is estimated in new axis $\mathbf{x}^{(1)}$ and so on.
The resulting rotation matrix equals to product

$$B = \prod_i B_i$$

.

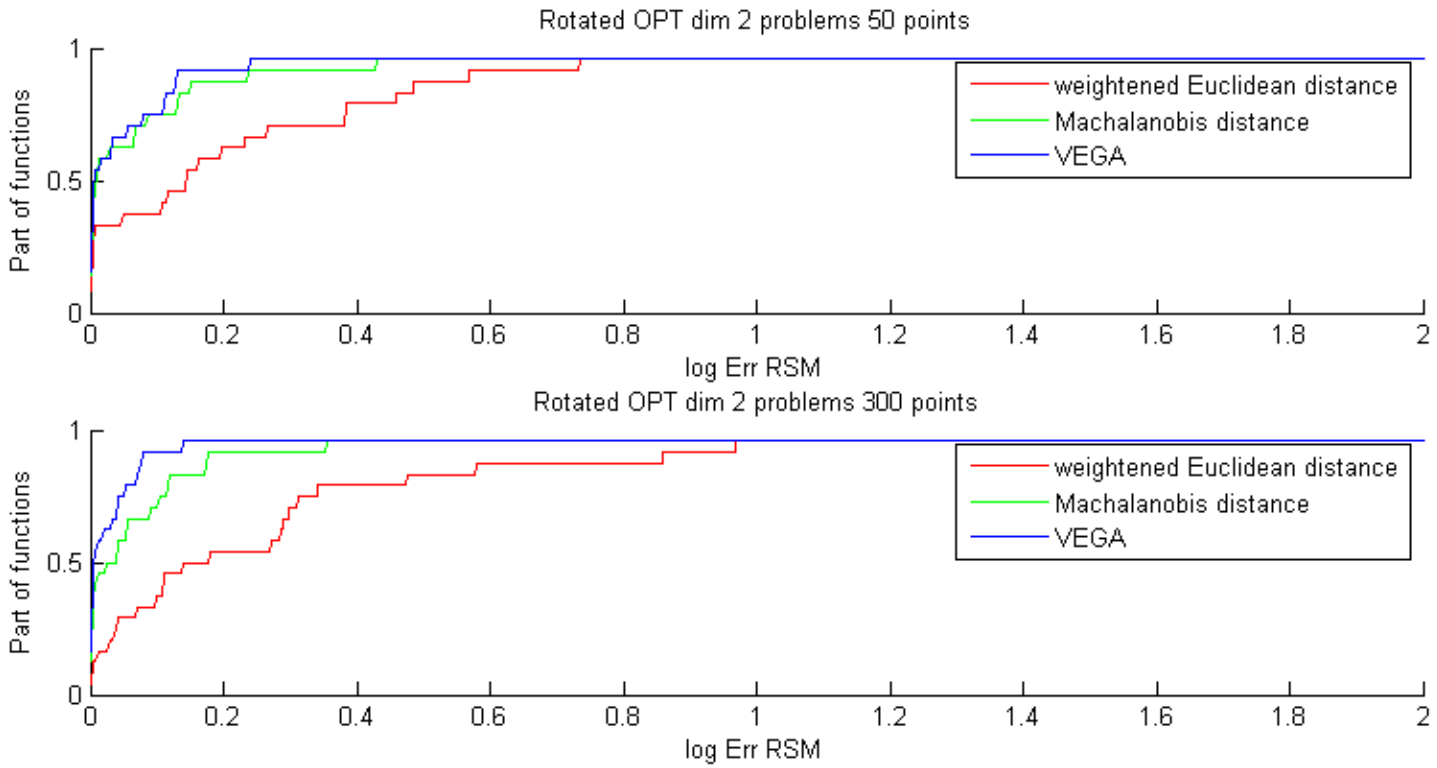
Axis Rotation in 2D Case

VEGA performance in this case is similar to the approximation based on Mahalanobis distance.



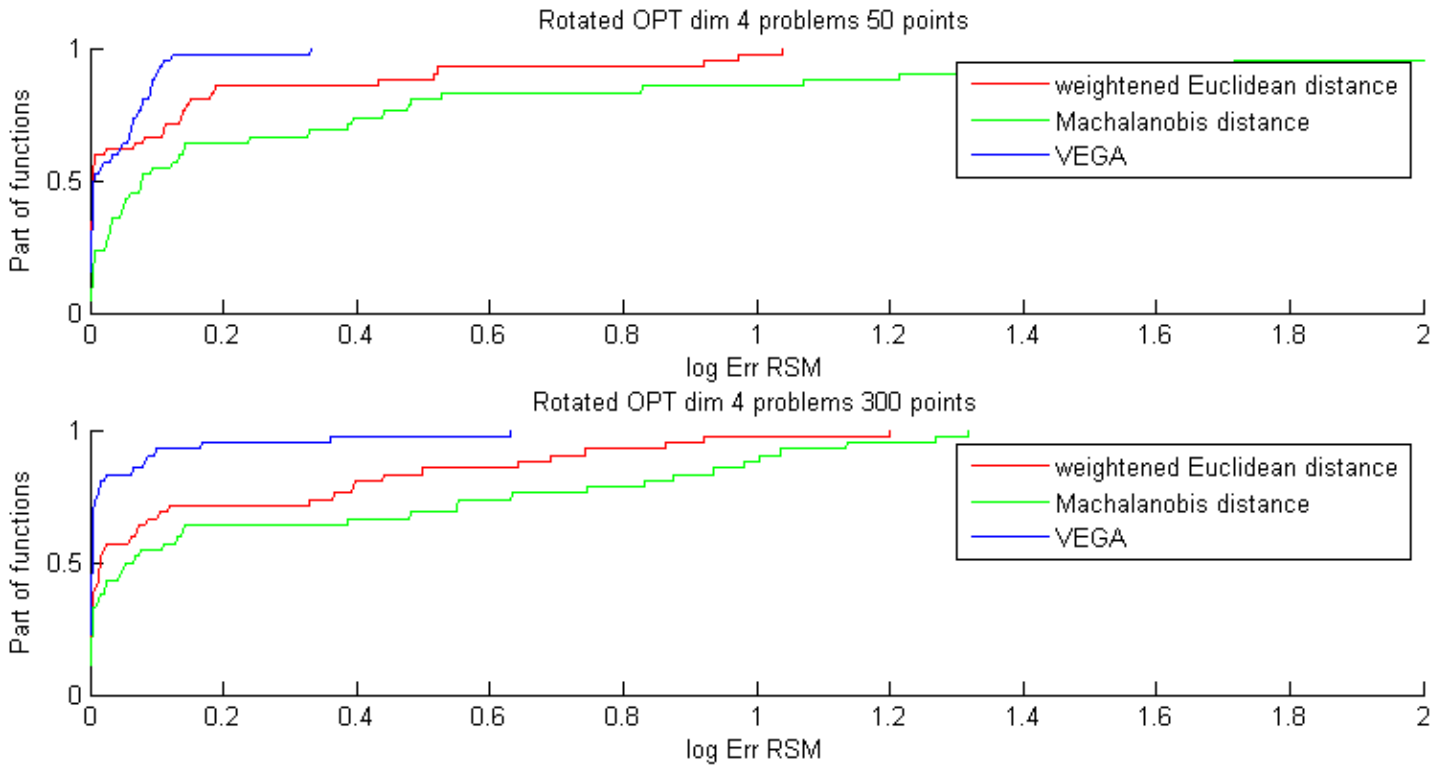
Axis Rotation in 2D Case

VEGA quality is equivalent to approximation based on Mahalanobis distance.



Axis Rotation in 4D Case

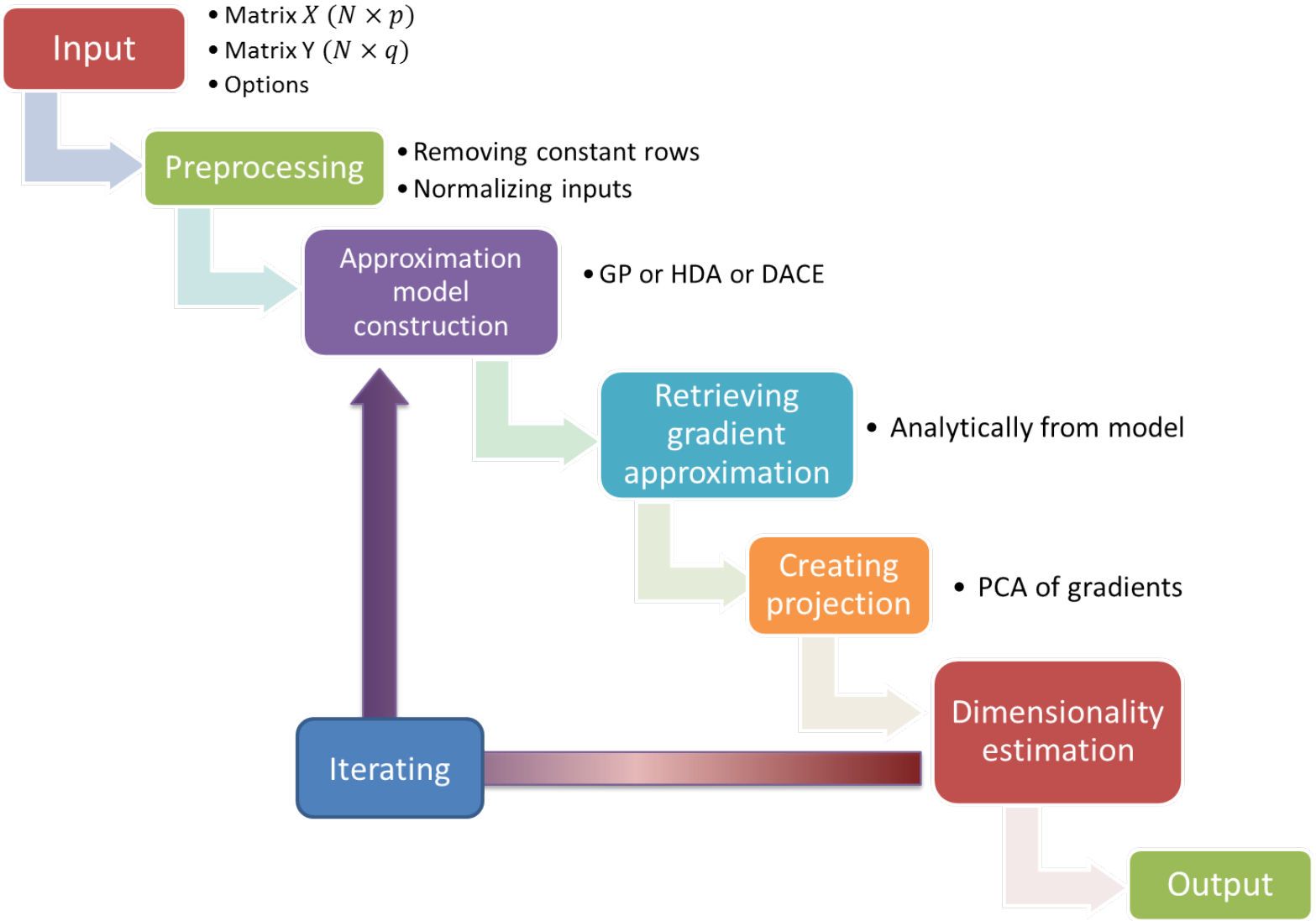
VEGA significantly outperforms approximation based on Mahalanobis distance.



Dimension Reduction

Let $\{\beta_i\}$ are eigenvectors and $\{\lambda_i\}$ are eigenvalues for $\Sigma_{\hat{\Gamma}} = \hat{\Gamma}^T\hat{\Gamma}$.
If $\lambda_i = 0$, the approximation considered has zero variability along β_i, that is $\hat{f}(\mathbf{x})$ does not depend on $\mathbf{x}\beta_i$.
One can discard "small" λs to reduced input space dimensionality (project on the linear subspace defined by matrix $\hat{B}$ compound of first eigenvectors).

VEGA Workflow



Effective Dimension Reduction Problem Statement

Let the data be generated by:

$$Y = f(\mathbf{x}) + \varepsilon(\mathbf{x}),$$

where $f(\mathbf{x}) = g(\mathbf{x}B^T)$, ε — Gaussian noise with $E\varepsilon = 0$ and $B \in \mathbb{R}^{d\times m}$, $d < m$, $BB^T = I_{d\times d}$
Effective dimension reduction problem is to find $S = span\{B\}$ (Central Mean Subspace CMS).
Let $\hat{B}$ — is an estimate B, containing CMS, then

$$f(\mathbf{x}\hat{B}^T\hat{B}) \approx f(x)$$

Comparison to other EDR methods

Several state-of-the-art methods were compared on artificial problems:

- SIR
- SAMM
- MAVE
- OPG
- PLS

Th comparison was held in terms of vicinity of $f(\mathbf{x})$ $f(\mathbf{x}BB^T)$ (reconstruction error).
The reduced dimension was given.

Comparison to other EDR methods

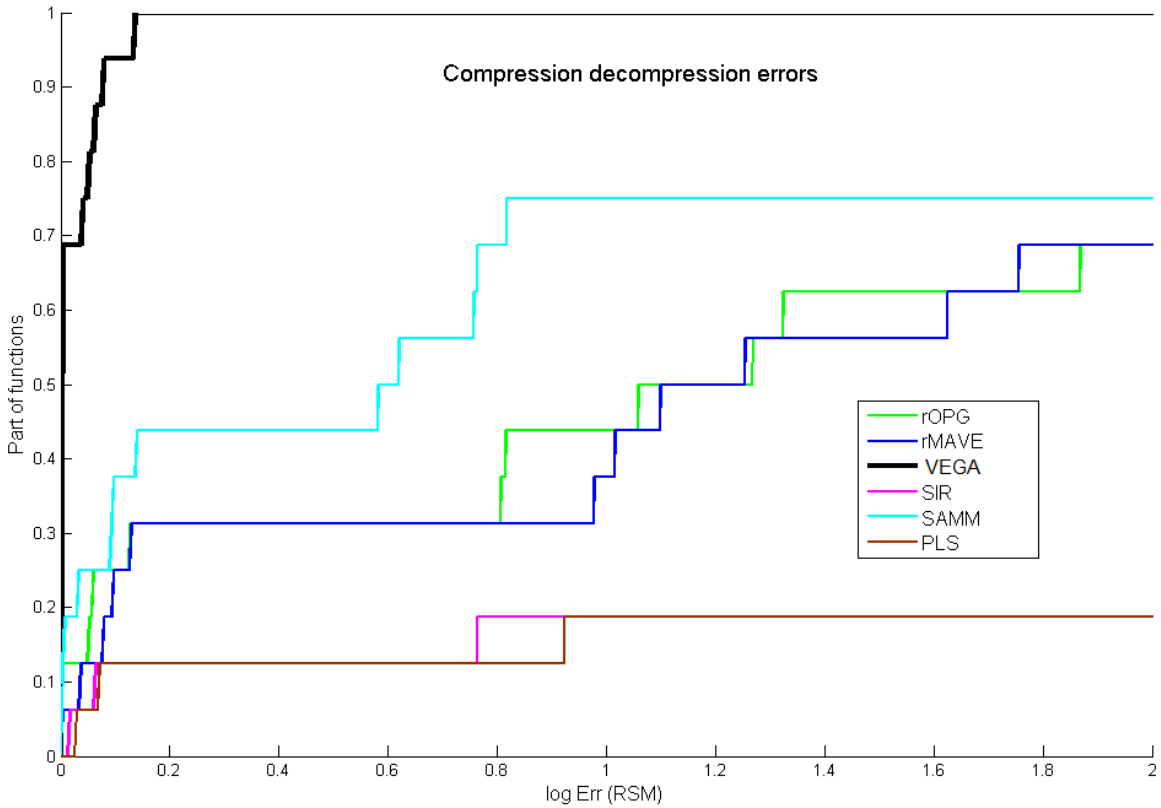


Figure: Dolan-Morè curves for reconstruction error

problem statement from Eurocopter

Problem: predict the energy consumption of helicopter maneuver, having a sample of experimental data.

- Dynamic System Rotor Analysis department.
- sample size: 705;
  - : mass, centering, altitude, moments, angles, etc (dimensionality – 51);
  - : efforts.

Eurocopter: Results

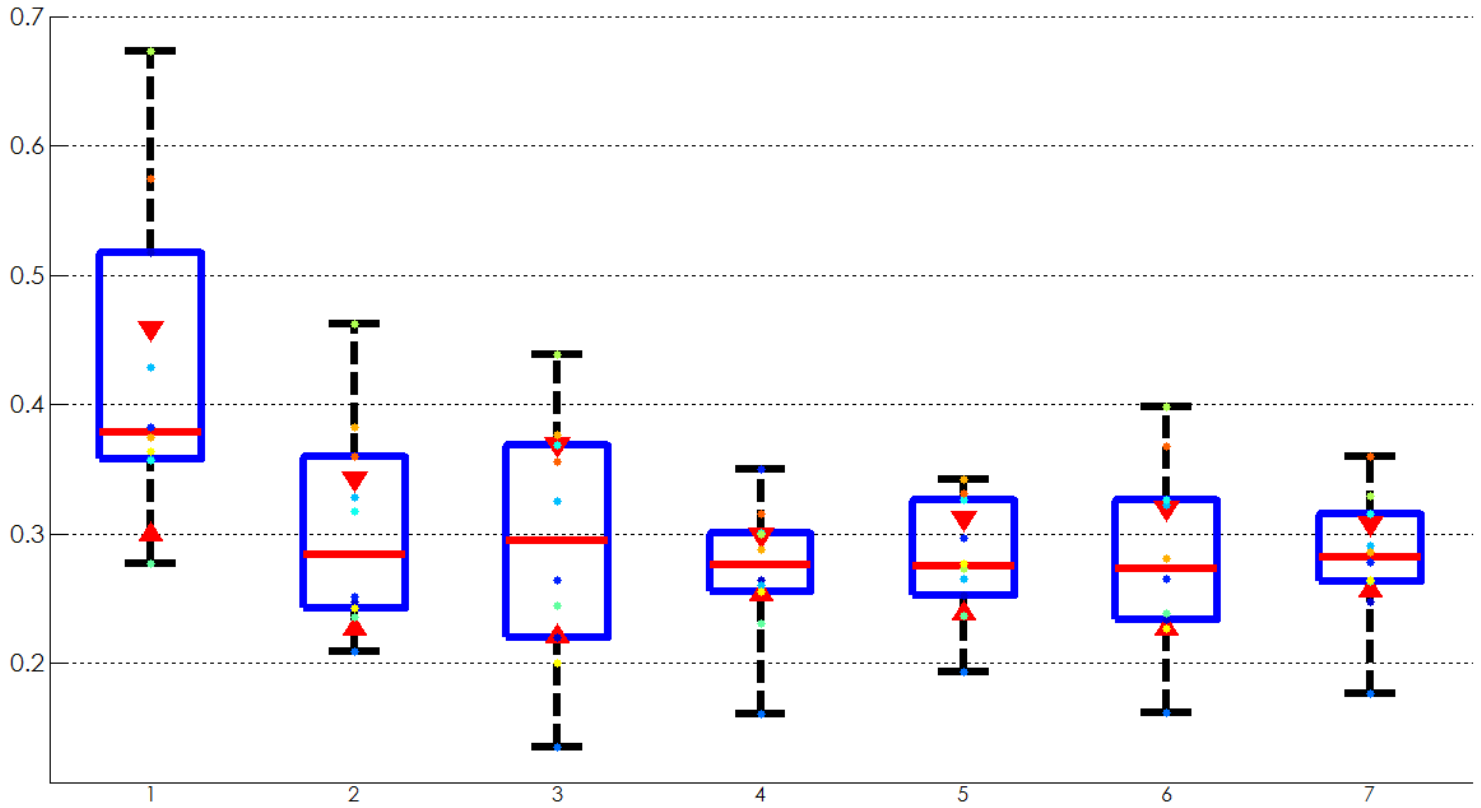


Figure: MSE for full-dimensional representation using VEGA by iteration

Eurocopter: Results

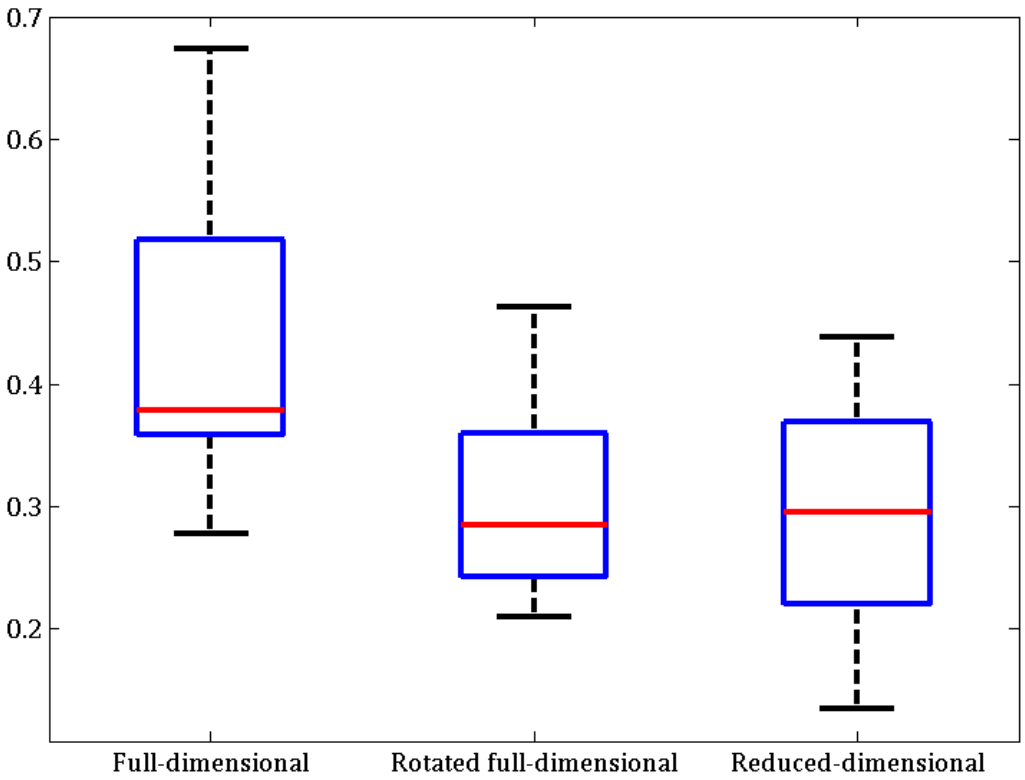


Figure: MSE for full-dimensional representation and for reduced-dimensional representation (14 parameters) on final iteration of VEGA

Discussion

- The proposed method allows to solve effective dimension reduction problem even better than state-of-the-art methods
- No additional parameters needed
- Approximation model is persistent to the axis rotation
- Much less parameters in likelihood maximization
- Improved quality of approximation based on Gaussian processes

Thank you for your attention.