Efficient Global Optimization with Indefinite Kernels

Martin Zaefferer and Thomas Bartz-Beielstein

Cologne University of Applied Sciences (TH Köln)
Faculty of Computer Science and Engineering Science,
Steinmüllerallee 1, 51643 Gummersbach, Germany
firstname.lastname@th-koeln.de

Abstract. Kernel based surrogate models like Kriging are a popular remedy for costly objective function evaluations in optimization. Often, kernels are required to be definite. Highly customized kernels, or kernels for combinatorial representations, may be indefinite. This study investigates this issue in the context of Kriging. It is shown that approaches from the field of Support Vector Machines are useful starting points, but require further modifications to work with Kriging. This study compares a broad selection of methods for dealing with indefinite kernels in Kriging and Kriging-based Efficient Global Optimization, including spectrum transformation, feature embedding and computation of the nearest definite matrix. Model quality and optimization performance are tested. The standard, without explicitly correcting indefinite matrices, yields functional results, which are further improved by spectrum transformations.

1 Introduction

When optimization requires time-consuming experiments, surrogate models are a well established approach to reduce the load of objective function evaluations [8]. Kernel-based models are a popular choice, e.g., Support Vector Machines (SVM) and especially Kriging. Often, kernels are required to be positive semi-definite (PSD), e.g., to allow for the existence of a map to a higher dimensional feature space (kernel trick) or to allow for interpretation of kernel matrices as a correlation matrices [20, 5]. While ordinary kernels are PSD, users may have to apply uncommon kernels [19]. One example are distance-based kernels for combinatorial optimization problems, that may not be definite [17, 26, 25]. Even in real-valued search spaces, prior knowledge can be used to design promising, custom, indefinite kernels. While research on indefinite kernels with Kriging is sparse, the SVM field provides an useful starting point [19].

This study outlines existing techniques for dealing with indefinite distances and kernels. The issues of their application to Kriging are elaborated and possible solutions are explained. A comparative test-study with transparent, artificial test-functions is presented, with the goal of determining the benefit of different indefiniteness correction methods.

2 Terms and Definitions

This study makes use of the following concepts.

Input space: The input space is a non-empty set $\mathcal{X}$.
Sample: A sample $x \in \mathcal{X}$ can be a vector (continuous or discrete), string, tree or some other object.
Kernel function: A symmetric function $\mathrm{k}(x, x')$ with $\mathrm{k} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$.
Distance function: A symmetric function $\mathrm{d}(x, x')$ with $\mathrm{d} : \mathcal{X} \times \mathcal{X} \rightarrow \mathbb{R}$, $\mathrm{d}(x, x') \geq 0$ and $\mathrm{d}(x, x') = 0$ if $x = x'$.
Distance metric: A distance function $\mathrm{d}(x, x')$ which a) is zero *iff* $x = x'$ and b) fulfills the triangle inequality $\mathrm{d}(x, x') + \mathrm{d}(x', x'') \geq \mathrm{d}(x, x'')$.
Kernel matrix: A matrix $\boldsymbol{K}$ with element $k_{ij} = \mathrm{k}(x_i, x_j)$.
Distance matrix: A matrix $\boldsymbol{D}$ with element $d_{ij} = \mathrm{d}(x_i, x_j)$.
Ill-conditioning: A symmetric matrix is ill-conditioned if $|\lambda_n|/|\lambda_1|$ is large. λ_n is the largest and λ_1 the smallest eigenvalue. Ill-conditioning is not in the focus of this paper, but may require related methods.
Definiteness: A symmetric $n \times n$ matrix $\boldsymbol{A}$ is positive definite (PD) iff $\boldsymbol{c}\boldsymbol{A}\boldsymbol{c}^T > 0$ for all $\boldsymbol{c} \in \mathbb{R}^n$. This is equivalent to all eigenvalues $\lambda_1 \leq \lambda_2 \leq ... \leq \lambda_n$ of $\boldsymbol{A}$ being positive. Respectively, a matrix is negative definite (ND) iff all eigenvalues are negative. The matrix is Positive or Negative Semi-Definite (PSD, NSD), iff all eigenvalues are non-negative (i.e., some are zero) or non-positive. A kernel matrix is usually required to be PSD. A broader class are Conditionally PSD or NSD (CPSD, CNSD) matrices, with the condition $\sum_{i=1}^{n} c_i = 0$. If a matrix matches none of these criteria, it is indefinite.

A function k is PSD (NSD) iff $\sum_{i=1}^{n} \sum_{j=1}^{n} c_i c_j \mathrm{k}(x_i, x_j) \geq (\leq)\ 0$, for all $n \in \mathbb{N}$ and $x \in \mathcal{X}$. It is conditionally definite if $\sum_{i=1}^{n} c_i = 0$. A distance measure $\mathrm{d}(x, x')$ is CNSD iff the Gaussian kernel $\mathrm{k}(x, x') = \exp(-\theta \mathrm{d}(x, x'))$ is PSD for all $\theta > 0$ [20, Proposition 2.28]. In case of SVM, PSD kernels guarantee that the mapping into some higher dimensional feature space exists (kernel trick) [20].
Correlation function: A special case of PSD kernels are correlation functions. Their values should be $-1 \leq k(x, x') \leq 1$, and $k(x, x') = 1$ if $x = x'$. Correlation matrices are required for statistical models like Kriging. The PSD requirement becomes clear when considering a linear combination of random variables. Indefinite matrices would imply negative variances of such combinations.
Kriging: This definition is based on [5] and some adaptations in [26]. Given a set of n samples $\boldsymbol{X} = \{x_i\}$, observations $\mathbf{y} = \{y_i\}$ and $i = 1 \ldots n$, Kriging interprets the observed responses $\mathbf{y}$ as realizations of a stochastic process. The set of random vectors $\boldsymbol{Y} = \{Y(x_i)\}$ is used to define this stochastic process. Correlations can, e.g., be modeled by the kernel

$$\mathrm{cor}\,[Y(x), Y(x')] = \mathrm{k}(x, x') = \exp(-\theta \mathrm{d}(x, x')). \quad (1)$$

with $\theta \in \mathbb{R}_+$. Both $\mathrm{k}(x, x')$ and $\mathrm{d}(x, x')$ can be chosen depending on the problem. For example, in case of combinatorial optimization $\mathrm{d}(x, x')$ can be a distance measures for binary strings, permutations, or trees [16, 26].
Kriging predictor: The correlation matrix $\boldsymbol{K}$ is used in the predictor function

$$\hat{y}(x) = \hat{\mu} + \boldsymbol{k}^T \boldsymbol{K}^{-1}(\mathbf{y} - \mathbf{1}\hat{\mu}), \quad (2)$$

where $\hat{y}(x)$ is the predicted function value of a new sample x, $\hat{\mu}$ is the Maximum Likelihood Estimate (MLE) of the process mean, $\mathbf{1}$ is a vector of ones and $\boldsymbol{k}$ is the column vector of correlations between training samples $\boldsymbol{X}$ and the new

sample x. Kernel parameters (e.g., θ) are determined by MLE. The MLE based on uncorrected, indefinite correlation matrices can be very misleading, producing unusable models. As an indefinite matrix can not be a correlation matrix, a basic assumption of the model is violated. Hence, indefiniteness requires correction.
Uncertainty estimate: The uncertainty of the prediction is estimated with

$$\hat{s}^2(x) = \hat{\sigma}^2(1 - \boldsymbol{k}^T \boldsymbol{K}^{-1} \boldsymbol{k}), \tag{3}$$

where $\hat{\sigma}^2$ is an estimate of the process variance, also determined by MLE.
Efficient Global Optimization: The uncertainty estimate $\hat{s}(x)$ is used in the Efficient Global Optimization (EGO) algorithm [10]. In EGO, a Kriging model is first built based on an initial set of observations $\mathbf{y}$ with elements $y_i = \mathrm{f}(x_i)$. Here, $\mathrm{f} : \mathcal{X} \to \mathbb{R}$ is an objective function to be minimized. It is assumed to be very expensive to evaluate (due to consumption of time or other resources). If $\hat{s}(x) > 0$, the Expected Improvement (EI) [14] of a sample is

$$\mathrm{EI}(x) = (\min(\mathbf{y}) - \hat{y}(x))\Phi\left(\frac{\min(\mathbf{y}) - \hat{y}(x)}{\hat{s}(x)}\right) + \hat{s}(x)\phi\left(\frac{\min(\mathbf{y}) - \hat{y}(x)}{\hat{s}(x)}\right),$$

else $\mathrm{EI}(x) = 0$, where Φ is the normal cumulative distribution function, and ϕ the normal probability density function. The sample x that maximizes $\mathrm{EI}(x)$ is evaluated with $\mathrm{f}(x)$. The resulting data is used to update the model. This repeats until a termination criterion is fulfilled (e.g., function evaluation budget).

3 Handling Indefinite Kernels

Several recent studies on SVMs (and related methods) dealt with indefinite kernels, cf. the survey in [19]. This topic has seen less attention in connection to Kriging [12, 3, 2]. Four types of methods can be identified. *Spectrum transformations* attempt to transform the matrix such that all eigenvalues have the desired sign. They have been used for SVMs [19] and, to some extend, for Gaussian Processes [2]. They are outlined and extended by repair methods in Sec. 3.1 to 3.3. *Nearest matrix* algorithms (Sec. 3.4) try to find matrices that are definite as well as close to the original matrices. *Feature embedding* (Sec. 3.5) understands the indefinite similarities (or distances) as features, and uses a standard, definite kernel to compute a surrogate similarity based on these features. *Method modifications* have been introduced to remove the necessity of definiteness in SVMs, e.g., by converting the quadratic programming problem to a linear one (LP-SVM or 1-norm SVM [13, 27, 11]). Method modifications are usually not transferable to Kriging and hence not considered here.

In the following, $\tilde{\boldsymbol{K}}$ denotes the definiteness-corrected variant of $\boldsymbol{K}$. Respectively, $\tilde{\boldsymbol{k}}$ will be the modified variant of $\boldsymbol{k}$ (cf. Eq. (2)). For distances, $\tilde{\boldsymbol{D}}$ and $\tilde{\boldsymbol{d}}$ are employed equivalently.

3.1 Spectrum Transformation: Kernel

The basis for the spectrum transformation is the decomposition of the kernel matrix $\boldsymbol{K} = \boldsymbol{U}\boldsymbol{\Lambda}\boldsymbol{U}^T$, where $\boldsymbol{U}$ is the matrix of eigenvectors of $\boldsymbol{K}$, $\boldsymbol{\Lambda} = \mathrm{diag}(\boldsymbol{\lambda})$ the diagonal matrix containing the eigenvalues of $\boldsymbol{K}$. Following Chen et al. [4], the spectrum transformation can be written as a linear transformation based on some vector $\boldsymbol{a} \in \mathbb{R}^n$:

$$\tilde{\boldsymbol{K}} = \boldsymbol{A}\boldsymbol{K} \quad \text{with} \quad \boldsymbol{A} = \boldsymbol{U}\text{diag}(\boldsymbol{a})\boldsymbol{U}^T. \tag{4}$$

Several choices for $\boldsymbol{a}$ are available [23, 19].

I) Spectrum *flip* transforms the eigenvalues to their absolute values, with $\tilde{\lambda}_i = |\lambda_i|$ and $\boldsymbol{a}_{flip} = \text{sign}(\boldsymbol{\lambda})$. With Eq. (6) and using $\boldsymbol{a}_{flip}$, the resulting approach is very similar to the one described by Loosli et al. [11] for SVMs in Krein spaces.

II) Spectrum *clip* removes negative eigenvalues by setting them to zero, with $\tilde{\lambda}_i = \max(\lambda_i, 0)$ and $\boldsymbol{a}_{clip} = \{\mathbb{I}(\lambda_1), ..., \mathbb{I}(\lambda_n)\}$, where $\mathbb{I}(\lambda_i) = 1$ if $\lambda_i \geq 0$ else $\mathbb{I}(\lambda_i) = 0$. Spectrum clip relates to the Moore–Penrose pseudoinverse [15], which is sometimes used in case of ill-conditioned $\boldsymbol{K}$.

III) Spectrum *shift* uses $\tilde{\lambda}_i = \lambda_i + \eta$ with $\eta \in \mathbb{R}_+$ and $\tilde{\boldsymbol{K}} = \boldsymbol{K} + \eta\boldsymbol{I}_n$. Shifting is the same as the nugget effect that may be used in the Kriging model, where η is an additional parameter determined by MLE. It may be reasonable to combine it with some of the other transformations, e.g., to deal with numerical issues or noise. The nugget effect is often used to regularize ill-conditioned $\boldsymbol{K}$ [15].

IV) Spectrum *square* uses $\tilde{\lambda}_i = (\lambda_i)^2$ and $\boldsymbol{a}_{sqr} = \boldsymbol{\lambda}$. Also: $\tilde{\boldsymbol{K}} = \boldsymbol{K}\boldsymbol{K}$.

V) Spectrum *diffusion* uses $\tilde{\lambda}_i = \exp(\lambda_i)$ and $\boldsymbol{a}_{diff} = \exp(\boldsymbol{\lambda})/\boldsymbol{\lambda}$. This leads to the diffusion Kernel, with $\tilde{\boldsymbol{K}} = \exp(\boldsymbol{K})$ [23].

Of all these transformations, only shift (cf. nugget effect [5, 15]) and clip (cf. pseudo-inverse [15] or multi dimensional scaling [3]) have been used with Kriging, although mostly for the purpose of dealing with noise or ill-conditioning.

The same transformation $\boldsymbol{A}$ has to be applied to $\boldsymbol{k}$ for prediction (see Eq. (2)):

$$\tilde{\boldsymbol{k}} = \boldsymbol{A}\boldsymbol{k}. \tag{5}$$

In case of spectrum shift, Eq. (5) is not required since the spectrum shift only affects self-similarities $\text{k}(x, x)$. While computing $\tilde{\boldsymbol{k}}$ is a consistent way to treat new test samples [4], it has been noted as a drawback due to the effort of (5) for each single prediction [11]. This issue can be remedied as follows. In the Kriging predictor given in Eq. (2), $\boldsymbol{k}^T\boldsymbol{K}^{-1}$ is computed. With the respective transformations we can prove that:

$$\tilde{\boldsymbol{k}}^T\tilde{\boldsymbol{K}}^{-1} = (\boldsymbol{A}\boldsymbol{k})^T\tilde{\boldsymbol{K}}^{-1} = \boldsymbol{k}^T\boldsymbol{A}^T\tilde{\boldsymbol{K}}^{-1}. \tag{6}$$

The computation of $\boldsymbol{A}^T\tilde{\boldsymbol{K}}^{-1}$ has to be performed only once after training, since it does not depend on the new sample. Afterwards, prediction requires only the usual computational effort of the Kriging predictor. Using Eq. (4) and $\boldsymbol{U}^T = \boldsymbol{U}^{-1}$ we can also prove that

$$\tilde{\boldsymbol{k}}^T\tilde{\boldsymbol{K}}^{-1} = \boldsymbol{k}^T\boldsymbol{K}^{-1}. \tag{7}$$

Similarly, the uncertainty estimate in Eq. (3) uses $\boldsymbol{k}^T\boldsymbol{K}^{-1}\boldsymbol{k}$. With Eq. (4), (5) and (6) this becomes:

$$\tilde{\boldsymbol{k}}^T\tilde{\boldsymbol{K}}^{-1}\tilde{\boldsymbol{k}} = \boldsymbol{k}^T\boldsymbol{A}^T\tilde{\boldsymbol{K}}^{-1}\boldsymbol{A}\boldsymbol{k}. \tag{8}$$

Hence, $\boldsymbol{A}^T\tilde{\boldsymbol{K}}^{-1}\boldsymbol{A}$ needs to be computed only once. In the following, *PSD-correction* refers to all methods that transform the spectrum of the kernel matrix, with $\tilde{\boldsymbol{K}} = \text{SPEC}_{\text{PSD}}(\boldsymbol{K})$.

3.2 Spectrum Transformation: Distance

Spectrum transformations can also be applied to distances. PSD-correction can be applied directly via $\tilde{\boldsymbol{D}} = \mathrm{SPEC}_{\mathrm{NSD}}(\boldsymbol{D}) = -\mathrm{SPEC}_{\mathrm{PSD}}(-\boldsymbol{D})$. New data is handled accordingly, i.e., $\tilde{\boldsymbol{d}} = \boldsymbol{A}\boldsymbol{d}$. Equations (6), (7) and (8) are not useful in this case. Thus, effort increases for prediction but decreases for MLE.

Alternatively, spectrum transformations can be used to generate CNSD matrices as described by Glunt et al. [6]. First, $\boldsymbol{Q} = \boldsymbol{I} - (2\mathbf{v}\mathbf{v}^T)/(\mathbf{v}^T\mathbf{v})$ with $\mathbf{v} = 1, 1, ..., 1, \sqrt{n}$ is computed and used to yield $\widehat{\boldsymbol{D}} = \boldsymbol{Q}(-\boldsymbol{D})\boldsymbol{Q}$. Then, $\widehat{\boldsymbol{D}}_{(-n,-n)}$ is extracted, which is $\widehat{\boldsymbol{D}}$ without last row and column. The matrix $\check{\mathbf{D}}$ is then constructed using $\mathrm{SPEC}_{\mathrm{PSD}}(\widehat{\boldsymbol{D}}_{-n,-n})$ and the unchanged last row and column of $\widehat{\boldsymbol{D}}$. Finally, the matrix in the original form is $\tilde{\boldsymbol{D}} = -\boldsymbol{Q}\check{\mathbf{D}}\boldsymbol{Q}$. With spectrum clip, this approach is similar to Multi Dimensional Scaling, as used by Boisvert et al. [3] to correct indefiniteness in a Kriging model.

Due to the more complex transformations in the CNSD case, $\tilde{\boldsymbol{d}} = \boldsymbol{A}\boldsymbol{d}$ is no longer valid. Instead, the augmented distance matrix $\boldsymbol{D}_{aug}$ is computed, which includes distances between all training and new data. Then,

$$\boldsymbol{D}_{aug} = \begin{bmatrix} \boldsymbol{D} & \boldsymbol{d} \\ \boldsymbol{d}^T & 0 \end{bmatrix} \quad \text{and (after transformation)} \quad \begin{bmatrix} \tilde{\boldsymbol{D}} & \tilde{\boldsymbol{d}} \\ \tilde{\boldsymbol{d}}^T & \tilde{\delta} \end{bmatrix} = \tilde{\boldsymbol{D}}_{aug}, \tag{9}$$

where $\tilde{\delta}$ is the potentially non-zero self-distance of the transformed new data. The resulting $\tilde{\boldsymbol{d}}$ can be used in Eq. (1) and Eq. (2). In the following, spectrum transformations of the distance matrix are denoted with *NSD-* or *CNSD-correction.*

3.3 Spectrum Transformations: Condition-Repair

The spectrum transformations may yield definite matrices that do not fulfill the additional conditions required for distance and correlation functions (cf. Sec. 2). One consequence is, that uncertainty estimates for observed samples (training data) become non-zero. This may stall the optimization progress (cf. a similar issue with the nugget effect described in [5]). Methods that mend this issue are referred to as *condition-repair.*

A correlation matrix can be repaired with $\tilde{k}^*_{ij} = \tilde{k}_{ij}/\mathrm{sqrt}(\tilde{k}_{ii}\tilde{k}_{jj})$ [18]. A CNSD distance matrix $\tilde{\boldsymbol{D}}$ can be repaired with $\tilde{d}^*_{ij} = 2\tilde{d}_{ij} - \tilde{d}_{ii} - \tilde{d}_{jj}$. The result is CNSD, non-negative and has zero diagonal [21]. In case of condition-repair, correlations $\tilde{\boldsymbol{k}}$ and distances $\tilde{\boldsymbol{d}}$ between training data and new samples have to be derived as outlined in Eq. (9). Spectrum shift only changes the diagonal of $\boldsymbol{K}$. Its influence on the uncertainty estimate can be remedied by re-interpolation [5].

3.4 Nearest Matrix Approach

Finding the nearest correlation matrix [7] or nearest euclidean distance matrix [6] is closely related to spectrum transformation. An alternating projections approach can be used to compute the nearest matrices. The first projection employs the spectrum clip. The second projection sets diagonals to one (correlation) or zero (distance). Thus, further condition-repair is not required. Unfortunately, these methods lack an efficient way of handling new data. Similarly to the condition-repair procedures, Eq. (9) can be used to derive $\tilde{\boldsymbol{d}}$ (or $\tilde{\boldsymbol{k}}$ analogously).

3.5 Feature Embedding

In feature embedding [11], non-CNSD distances can be used as input features for a CNSD distance function: $\tilde{d}_{ij} = \mathrm{d}_{\mathrm{def}}(\boldsymbol{d}_{i\cdot}, \boldsymbol{d}_{j\cdot})$, where $\boldsymbol{d}_{i\cdot}$ and $\boldsymbol{d}_{j\cdot}$ are the ith and jth rows of $\boldsymbol{D}$, and $\mathrm{d}_{def}(x, x')$ is a CNSD distance function (here: Euclidean). Distances $\boldsymbol{d}$ between training and new data have to be subject to $\tilde{d}_i = \mathrm{d}_{def}(\boldsymbol{d}, \boldsymbol{d}_{i\cdot})$.

4 Experimental Setup

Test-problems: The samples x were restricted to be permutations, to enable a well understandable and controllable test case. Other object types are possible but were omitted for the sake of brevity. Different numbers of permutation elements were tested: $m = 5, 7, 10$. The experiments were performed with simple test-functions $\mathrm{f}(x) = \min_i \mathrm{d}(x, \gamma_i)$, where x is a sample (permutation), and the respective function value $\mathrm{f}(x)$ is the minimum distance to randomly chosen centers $\gamma_i \in \mathcal{X}$, with $i = 1, ..., w$. For the sake of this test, the function $\mathrm{f}(x)$ was assumed to be expensive. The number of centers w control the multi-modality of the function. In case of $w = 1$, $\mathrm{f}(x)$ is unimodal (as used in [17]). For the experiments, $w = 1, 3$ and 5 was tested. Two distance measures for permutations were used: The Interchange Distance is the minimal number of transpositions of arbitrary elements required to transform one permutation into another. It is metric, but not CNSD. As a more pathological (yet admittedly quite artificial) test-case, we chose the non-metric, non-CNSD distance $\mathrm{d}_{Lp}(x, x') = (\sum_{i=1}^{n} |x_i - x'_i|^p)^{1/p}$ with $p = 1/2$. Here, the permutations are interpreted as a vector of integers.
Performance measures: Two sets of experiments were performed, 1) testing for modeling performance (including the quality of the uncertainty estimate) and 2) for optimization performance. The Root Mean Squared Error (RMSE) was used to estimate prediction accuracy. To assess the uncertainty estimate, standardized residuals $r = (y - \hat{y})/\hat{s}$ were computed, cf. [10, 22]. These are used to calculate the Cramèr-von Mises (CVM) test statistic [1] (comparing against a normal distribution with zero mean and unit variance). 10-fold cross validation is used to receive statistically sound results. For the modeling experiments, the number of samples is $n = 20, 40$ and 60. For the optimization performance, best values found after 20 and 100 objective function evaluations are reported.
Model settings: The Dividing Rectangles algorithm [9] was chosen to optimize the model parameters (θ, η) during MLE. For each parameter, 200 likelihood evaluations were allowed. A relative tolerance of 1e−6 was used to detect earlier convergence. For (uncorrected) indefinite matrices, the logarithmic likelihood evaluation was set to return a penalty of $-1e4 + \lambda_1$, to drive the search into the direction of PSD matrices. In all cases, PSD matrices could be established. However, the resulting matrix was sometimes numerically intractable in case of spectrum diffusion, which was hence excluded from further analysis (see Sec. 5). Re-interpolation [5] was employed to correct the uncertainty estimates in case of spectrum shift. Note, that η was always added to the diagonal of $\tilde{\boldsymbol{K}}$, i.e., *after* applying other correction methods. The models always used the same distance functions that were employed in the test function, combined with the kernel in

Eq. (1). This simulates the case where an adequate distance is chosen by prior knowledge. All experiments were repeated 20 times.
Optimization settings: For optimization, most settings remain unchanged. The budget of evaluations of $\mathrm{f}(x)$ was set to 100. Ten initial samples were chosen at random and evaluated with $\mathrm{f}(x)$. In each following step, the candidate that maximized EI (cf. Sec. 2) was determined by a Genetic Algorithm (GA). The GA had a budget of 2000 model evaluations for each step, except for $m = 5$, where brute force was used ($m! = 120$ model evaluations). The GA used interchange mutation (transposition of arbitrary elements) and cycle crossover. The population size was 20, the mutation rate $1/m$ and the recombination rate 0.5. As a baseline-comparison, a simple and model-free random search with 100 objective function evaluations was performed. All experiments were repeated 20 times.

5 Observations and Discussion

To summarize overall performance, statistical multiple-comparison tests were used. Since the data were non-normal and the variances inhomogeneous, a rank transformation was performed for each combination of n, w, m and distance function. Then, Tukey's Honest Significant Differences (HSD) test [24] was used with a significance level $\alpha = 0.05$. Results were largely confirmed by a non-parametric test, which disagreed in about 2 % of the cases. With the resulting pair-wise comparison, a ranking was computed. All methods that were not significantly worse than any other received rank 1 and were removed. From the remainder, every method that was not significantly worse than any other received rank 2, and so on. Results from the spectrum diffusion approach were excluded as it performed poorly and failed several times, due to numerical issues with excessively large numbers. Table 1 reports the respective ranks. Interestingly, the ranks for model accuracy and optimization performance disagree often. One reason may be, that optimization only requires a locally accurate model.

It could be observed, that usable models were achieved by the standard approach, as it outperformed the random search. That is because even a non-CNSD distance matrix may yield a PSD kernel matrix if θ is chosen large enough, but not too large. This becomes obvious with $\lim_{\theta\to\infty} \boldsymbol{K} = \boldsymbol{I}$, which is of course PD. However, if $\theta \to \infty$, Eq. (2) will just yield the mean of observations $\mathbf{y}$.

Enhancing the standard approach by spectrum shift improved optimization performance, but received the worst RMSE ranks. In general, a clear benefit of shift could not be observed. In combination with other indefiniteness-correcting methods, it either improved or deteriorated results. Due to the additional cost of fitting η, it may be undesirable for non-noisy data.

The simple feature embedding performed robustly, but not for smaller data sets. The performance after 20 evaluations (F_a in Tab. 1) was suboptimal. Feature embedding seemed to require larger data-sets to learn the embedding.

Spectrum transformations were among the best performers. Their main drawback is the difficulty of deciding on a) usage of condition-repair b) type of transformation and c) whether NSD-, CNSD- or PSD-correction should be used. For

Table 1. Ranks for RMSE (R), CVM values (C), best value after 20 evaluations (F_a) and 100 evaluations (F_b). Ranks are based on Tukey's HSD test, small values are better. P indicates percentage of cases where the optimum was found within 100 evaluations, large values are better. Table is sorted by $F_a + F_b$, with tie-breaker P. Color indicates a rank of 1, or $P \geq 0.9$. In the *names* columns, the leading boolean denotes whether condition-repair was used (T) or not (F). *CNSD/NSD/PSD*: the correction type, *feature*: feature embedding, *near*: nearest matrix approach, *standard*: no specific correction and *random*: random search. Other terms refer to the spectrum transformations.

names	R	C	F_a	F_b	P	names	R	C	F_a	F_b	P
T.flip.PSD	6	4	1	1	1	T.square.PSD.shift	6	5	2	2	0.91
F.clip.CNSD	1	6	1	1	0.99	standard.shift	7	4	2	2	0.91
F.clip.CNSD.shift	1	8	1	1	0.98	T.square.NSD	3	2	2	2	0.89
T.clip.NSD	1	2	1	1	0.96	T.square.CNSD	3	2	2	2	0.89
T.clip.CNSD	1	2	1	1	0.96	T.square.CNSD.shift	3	4	3	2	0.92
F.flip.CNSD.shift	1	10	1	1	0.95	F.square.PSD	4	4	2	3	0.86
near.CNSD	3	4	1	1	0.94	T.square.PSD	5	3	2	3	0.85
F.flip.CNSD	1	7	1	1	0.94	T.clip.PSD	4	3	2	3	0.84
T.flip.PSD.shift	3	5	2	1	0.98	F.clip.PSD	4	3	2	3	0.84
T.clip.CNSD.shift	2	3	2	1	0.98	standard	4	3	2	3	0.84
near.CNSD.shift	3	5	2	1	0.98	near.PSD	5	3	2	3	0.83
F.clip.PSD.shift	3	5	2	1	0.97	F.clip.NSD.shift	2	9	3	3	0.84
T.clip.PSD.shift	3	7	2	1	0.97	F.clip.NSD	2	9	3	3	0.84
T.clip.NSD.shift	2	3	2	1	0.97	F.flip.NSD.shift	4	8	3	3	0.83
F.flip.PSD	7	4	2	1	0.96	F.flip.NSD	4	6	3	3	0.82
F.flip.PSD.shift	6	4	2	1	0.95	near.PSD.shift	3	8	3	4	0.8
T.flip.NSD.shift	2	2	2	1	0.94	F.square.PSD.shift	3	6	3	4	0.76
feature	3	1	2	1	0.94	F.square.CNSD	4	8	3	5	0.7
T.flip.NSD	1	1	1	2	0.92	F.square.CNSD.shift	3	9	4	5	0.73
feature.shift	3	2	3	1	0.94	F.square.NSD.shift	2	9	4	5	0.69
T.flip.CNSD.shift	2	2	2	2	0.93	F.square.NSD	4	10	3	6	0.67
T.square.NSD.shift	3	4	2	2	0.92	random			5	7	0.35
T.flip.CNSD	1	1	2	2	0.92						

a), the results are not quite conclusive, but a large block of the worse performing methods ($F_b > 2$ in Tab. 1) does not employ condition-repair. CVM statistic values are often better if condition-repair is used. For b), spectrum square is clearly worse than clip or flip, yet it may provide good results in combination with spectrum shift. Spectrum flip was not significantly different from spectrum clip. For c), the results were mixed, but RMSE ranks seemed to better with NSD- and CNSD-correction compared to PSD-correction. Intuitively, this makes sense: NSD- and CNSD-correction correct the distance matrix, which was the source of the indefiniteness. If the kernel function is the source, only PSD-correction is applicable. Despite very similar performance, NSD- may be preferred to CNSD-correction due to higher computational complexity of the latter.

The nearest matrix approaches required the most computational effort, with tenfold run-times or more. This is due to the necessity of solving an optimization problem for each correction. Since they performed no better than the related spectrum clip methods, the nearest matrix approaches can be disregarded.

6 Conclusions and Outlook

This study dealt with indefinite kernels in the Kriging-based EGO algorithm. Working Kriging models could be derived, even when indefiniteness was not explicitly corrected (besides the penalty described in Sec. 4). Methods based on spectrum transformations improved the performance. The spectrum transformations were compared to feature embedding and computations of the nearest definite matrix. As some of the resulting matrices were no proper correlation matrices, further condition-repair mechanisms were included. In some cases, this additional condition-repair was beneficial. From the set of spectrum transformations, spectrum flip and clip performed best, while square and diffusion performed poorly, in the latter case producing numerically intractable results.

Overall, the results indicate that choosing an adequate method automatically may be problematic. Cross-validation is an option, but not ideal, due to the lack of agreement between model accuracy and optimization performance. Also, some of the worst performing models reported large likelihoods, hence disqualifying a selection based on likelihood. More extensive experiments or a theoretical analysis of the various approaches could help dealing with this issue. For theoretical considerations, it is promising to see that spectrum flip works so well, since it is theoretically well-founded for SVMs [11]. Furthermore, these results may also be of interest in the context of regularization or ill-conditioning, especially with respect to condition-repairing procedures and handling of new data samples.

References

1. T. W. Anderson. On the distribution of the two-sample cramer-von mises criterion. *Ann. Math. Statist.*, 33(3):1148–1159, sep 1962.
2. M. S. Ayhan and C.-H. H. Chu. Towards indefinite gaussian processes. Technical report, University of Louisiana at Lafayette, 2012.
3. J. B. Boisvert and C. V. Deutsch. Programs for kriging and sequential gaussian simulation with locally varying anisotropy using non-euclidean distances. *Computers & Geosciences*, 37(4):495–510, 2011.
4. Y. Chen, M. R. Gupta, and B. Recht. Learning kernels from indefinite similarities. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 145–152, New York, NY, USA, 2009. ACM.
5. A. Forrester, A. Sobester, and A. Keane. *Engineering Design via Surrogate Modelling.* Wiley, 2008.
6. W. Glunt, T. L. Hayden, S. Hong, and J. Wells. An alternating projection algorithm for computing the nearest euclidean distance matrix. *SIAM Journal on Matrix Analysis and Applications*, 11(4):589–600, 1990.
7. N. J. Higham. Computing the nearest correlation matrix—a problem from finance. *IMA Journal of Numerical Analysis*, 22(3):329–343, 2002.
8. Y. Jin. A comprehensive survey of fitness approximation in evolutionary computation. *Soft Computing*, 9(1):3–12, 2005.
9. D. R. Jones, C. D. Perttunen, and B. E. Stuckman. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181, October 1993.

10. D. R. Jones, M. Schonlau, and W. J. Welch. Efficient global optimization of expensive black-box functions. *Journal of Global Optimization*, 13(4):455–492, 1998.
11. G. Loosli, S. Canu, and C. Ong. Learning svm in krein spaces. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 38(6):1204–1216, 2015.
12. J. G. Manchuk and C. V. Deutsch. Robust solution of normal (kriging) equations. Technical report, CCG Alberta, 2007.
13. O. L. Mangasarian. Generalized support vector machines. In A. J. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146. MIT Press, 2000.
14. J. Mockus, V. Tiesis, and A. Zilinskas. *Towards Global Optimization 2*, chapter The application of Bayesian methods for seeking the extremum, pages 117–129. North-Holland, 1978.
15. H. Mohammadi, R. Le Riche, N. Durrande, E. Touboul, and X. Bay. An analytic comparison of regularization methods for Gaussian Processes. Research report, Ecole Nationale Supérieure des Mines de Saint-Etienne ; LIMOS, Jan. 2016.
16. A. Moraglio and A. Kattan. Geometric generalisation of surrogate model based optimisation to combinatorial spaces. In *Proceedings of the 11th European Conference on Evolutionary Computation in Combinatorial Optimization*, EvoCOP'11, pages 142–154, Berlin, Heidelberg, Germany, 2011. Springer.
17. A. Moraglio, Y.-H. Kim, and Y. Yoon. Geometric surrogate-based optimisation for permutation-based problems. In *Proceedings of the 13th Annual Conference Companion on Genetic and Evolutionary Computation*, GECCO '11, pages 133–134, New York, NY, USA, 2011. ACM.
18. R. Rebonato and P. Jäckel. The most general methodology to create a valid correlation matrix for risk management and option pricing purposes. *Journal of Risk*, 2(2), 1999.
19. F.-M. Schleif and P. Tino. Indefinite proximity learning: A review. *Neural Computation*, 27(10):2039–2096, oct 2015.
20. B. Schölkopf and A. J. Smola. *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond.* MIT Press, Cambridge, MA, USA, 2001.
21. B. Schölkopf. The kernel trick for distances. In T. K. Leen, T. G. Dietterich, and V. Tresp, editors, *Advances in Neural Information Processing Systems 13*, pages 301–307. MIT Press, 2001.
22. T. Wagner. *Planning and Multi-Objective Optimization of Manufacturing Processes by Means of Empirical Surrogate Models.* PhD thesis, Technische Universität Dortmund, 2013. Vulkan Verlag, Essen.
23. G. Wu, E. Y. Chang, and Z. Zhang. An analysis of transformation on non-positive semidefinite similarity matrix for kernel machines. In *Proceedings of the 22nd International Conference on Machine Learning*, 2005.
24. B. S. Yandell. *Practical Data Analysis for Designed Experiments.* Chapman and Hall/CRC, 1997.
25. M. Zaefferer, J. Stork, and T. Bartz-Beielstein. Distance measures for permutations in combinatorial efficient global optimization. In T. Bartz-Beielstein, J. Branke, B. Filipic, and J. Smith, editors, *Parallel Problem Solving from Nature–PPSN XIII*, pages 373–383, Cham, Switzerland, 2014. Springer.
26. M. Zaefferer, J. Stork, M. Friese, A. Fischbach, B. Naujoks, and T. Bartz-Beielstein. Efficient global optimization for combinatorial problems. In *Proceedings of the 2014 Conference on Genetic and Evolutionary Computation*, GECCO '14, pages 871–878, New York, NY, USA, 2014. ACM.

27. J. Zhu, S. Rosset, T. Hastie, and R. Tibshirani. 1-norm support vector machines. *Advances in neural information processing systems*, 16(1):49–56, 2004.