Combining Gaussian Processes with Neural Networks for Active Learning in Optimization

Jiří Růžička[1], Jan Koza[1], Jiří Tumpach[2],
Zbyněk Pitra[1], and Martin Holeňa[1,2,3]

[1] Czech Technical University, Prague, Czech Republic
[2] Charles University, Prague, Czech Republic
[3] Institute of Computer Science, Czech Academy of Sciences, Prague, Czech Republic

Abstract. One area where active learning plays an important role is black-box optimization of objective functions with expensive evaluations. To deal with such evaluations, continuous black-box optimization has adopted an approach called surrogate modelling or metamodelling, which consists in replacing the true black-box objective in some of its evaluations with a suitable regression model, the selection of evaluations for replacement being an active learning task. This paper concerns surrogate modelling in the context of a surrogate-assisted variant of the continuous black-box optimizer Covariance Matrix Adaptation Evolution Strategy. It reports the experimental investigation of surrogate models combining artificial neural networks with Gaussian processes, for which it considers six different covariance functions. The experiments were performed on the set of 24 noiseless benchmark functions of the platform Comparing Continuous Optimizers COCO with 5 different dimensionalities. Their results revealed that the most suitable covariance function for this combined kind of surrogate models is the rational quadratic followed by the Matérn $\frac{5}{2}$ and squared exponential. Moreover, the rational quadratic and squared exponential covariances were found interchangeable in the sense that for no function, no group of functions, no dimension and combination of them, the performance of the respective surrogate models was significantly different.

Keywords: active learning, black-box optimization, artificial neural networks, Gaussian processes, covariance functions

1 Introduction

One area where active learning plays a very important role is *black-box optimization*, in particular optimization of black-box objective functions with expensive evaluations. It is immaterial whether that expensiveness is due to time-consuming computation like in long simulations [15], or due to evaluation in costly experiments like in some areas of science [3]. To deal with such expensive evaluations, continuous black-box optimization has in the late 1990s and early 2000s adopted an approach called *surrogate modelling* or *metamodelling* [6, 12, 14, 32, 40, 43, 46]. In this case, the goal of the surrogate model is to

decrease the total number of evaluations of the true objective function. Basically, a surrogate model is any regression model that with a sufficient fidelity, approximates the true black-box objective function and replaces it in some of its evaluations. And the decision in which points to evaluate the expensive black-box objective function, and in which to use its surrogate approximation is an active learning task.

This work-in-progress paper concerns surrogate modelling in the context of a state-of-the-art method for continuous black-box optimization, the Covariance Matrix Adaptation Evolution Strategy (*CMA-ES*) [20, 23]. It reports the first results of our investigation of surrogate models based on combining artificial neural networks (ANNs) with Gaussian processes (GPs). This investigation has been motivated by the importance of surrogate models based on ANNs alone [19,27–29,40,50] and especially on GPs alone [5,6,12–14,31,32,35,47,48], as well as by the high popularity of ANNs in the last 10–15 years. To our knowledge, this is the first time that ANN+GP combinations have been investigated for possible application in surrogate modelling. On the other hand, research into combining parametric ANN models with nonparametric GP models has been around for nearly a decade, at first due to the increasing popularity of neural networks, later also due to recent theoretical results concerning relationships of asymptotic properties of important kinds of ANNs to properties of GPs [33, 37, 39]. The integration of GP with neural learning has been proposed on two different levels:

(i) *Proper integration* of an ANN with a GP, in which the GP forms the final, output layer of the ANN [8, 49].

(ii) Only a *transfer of the layered structure*, which is a crucial feature of ANNs, to the GP context, leading to the concept of deep GPs (DGPs) [7,11,24,25].

In the reported investigation, we employed proper integration, using a GP as the final layer of an ANN.

The rest of the paper is organized as follows. In the next section, the theoretical fundamentals of GP regression and integration with ANNs are recalled. Section 3 describes active learning in a surrogate-assisted variant of CMA-ES. Replacing GPs in that variant with several ANN+GP combinations is then experimentally tested in Section 4. Finally, the concluding Section 5 also outlines our future research plans.

2 Gaussian Processes and Their Integration with Neural Networks

2.1 Gaussian Processes

A *Gaussian process* on a set $\mathcal{X} \subset \mathbb{R}^d, d \in \mathbb{N}$ is a collection of random variables $(f(x))_{x\in\mathcal{X}}$, any finite number of which has a joint Gaussian distribution [45]. It is completely specified by a *mean function* $m_{\mathrm{GP}} : \mathcal{X} \to \mathbb{R}$, typically assumed constant, and by a *covariance function* $\kappa : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that for $x, x' \in \mathcal{X}$,

$$\mathbb{E}f(x) = m_{\mathrm{GP}} \tag{1}$$

$$\mathrm{cov}(f(x), f(x')) = \kappa(x, x'). \tag{2}$$

Therefore, a GP is usually denoted $\mathcal{GP}(m_{\mathrm{GP}}, \kappa)$ or $\mathcal{GP}(m_{\mathrm{GP}}, \kappa(x, x'))$.

The value of $f(x)$ is typically accessible only as a *noisy observation* $y = f(x) + \varepsilon$, where ε is a zero-mean Gausssian noise with a variance $\sigma_n > 0$. Then

$$\mathrm{cov}(y, y') = \kappa(x, x') + \sigma_n^2 \mathbb{I}(x = x'), \tag{3}$$

where $\mathbb{I}$(proposition) equals for a true proposition 1, for a false proposition 0.

Consider now the prediction of the random variable $f(x_\star)$ in a point $x_\star \in \mathcal{X}$ if we already know the observations $y_1, \ldots, y_n$ in points $x_1, \ldots, x_n$. Introduce the vectors $x = (x_1, \ldots, x_n)^\top$, $y = (y_1, \ldots, y_n)^\top = (f(x_1) + \varepsilon, \ldots f(x_n) + \varepsilon)^\top$, $k_\star = (\kappa(x_1, x_\star), \ldots, \kappa(x_n, x_\star))^\top$ and the matrix $K \in \mathbb{R}^{n \times n}$ such that $(K)_{i,j} = \kappa(x_i, x_j) + \sigma_n^2 \mathbb{I}(i = j)$. Then the probability density of the vector y of observations is

$$p(y; m_{\mathrm{GP}}, \kappa, \sigma_n^2) = \frac{\exp\left(-\frac{1}{2}(y - m_{\mathrm{GP}})^\top K^{-1}(y - m_{\mathrm{GP}})\right)}{\sqrt{2\pi \det(K)}}, \tag{4}$$

where $\det(A)$ denotes the determinant of a matrix A. Further, as a consequence of the assumption of Gaussian joint distribution, also the conditional distribution of $f(x_\star)$ conditioned on y is Gaussian, namely

$$\mathcal{N}\left(m_{\mathrm{GP}}(x_\star) + k_\star K^{-1}(y - m_{\mathrm{GP}}), \kappa(x_\star, x_\star) - k_\star^\top K^{-1} k_\star\right). \tag{5}$$

According to (3), the relationship between the observations y and y' is determined by the covariance function κ. In the reported research, we have considered 6 kinds of covariance functions, listed below. In their definitions, the notation $r = \|x' - x\|$ is used, and among the parameters of κ, aka hyperparameters of the GP, frequently encountered are $\sigma_f^2, \ell > 0$, called *signal variance* and *characteristic length scale*, respectively. Other parameters will be introduced for each covariance function separately.

(i) *Linear*: $\kappa_{\mathrm{LIN}}(x, x') = \sigma_0^2 + x^\top x'$, with a bias σ_0^2.

(ii) *Quadratic* is the square of the linear covariance: $\kappa_{\mathrm{QUAD}}(x, x') = (\sigma_0^2 + x^\top x')^2$.

(iii) *Rational quadratic*: $\kappa_{\mathrm{RQ}}(x, x') = \sigma_f^2 \left(1 + \frac{r^2}{2\alpha\ell^2}\right)^{-\alpha}$, with $\alpha > 0$.

(iv) *Squared exponential*: $\kappa_{\mathrm{SE}}(x, x') = \sigma_f^2 \exp\left(-\frac{r^2}{2\ell^2}\right)$.

(v) *Matérn* $\frac{5}{2}$: $\kappa_{\mathrm{MA5}}(x, x') = \sigma_f^2 \left(1 + \frac{\sqrt{5}r}{\ell} + \frac{5r^2}{3\ell^2}\right) \exp\left(-\frac{\sqrt{5}r}{\ell}\right)$.

(vi) One *composite covariance function*, namely the sum of κ_{SE} and κ_{QUAD}: $\kappa_{\mathrm{SE+Q}}(x, x') = \kappa_{\mathrm{SE}}(x, x') + \kappa_{\mathrm{QUAD}}(x, x')$.

2.2 GP as the Output Layer of a Neural Network

An approach integrating a GP into an ANN as its output layer has been independently proposed in [8] and [49]. It relies on the following two assumptions:

1. If n_I denotes the number of the ANN input neurons, then the ANN computes a *mapping* net *of n_I-dimensional input values into the set* $\mathcal{X}$ on

which is the GP defined. Consequently, the number n_O of neurons in the last hidden layer fulfills $\mathcal{X} \subset \mathbb{R}^{n_O}$, and the ANN maps an input v into a point $x = \mathrm{net}(v) \in \mathcal{X}$, corresponding to an observation $f(x) + \varepsilon$ governed by the GP (Figure 1). From the point of view of the ANN inputs, the GP is now $\mathcal{GP}(m_{\mathrm{GP}}(\mathrm{net}(v)), \kappa(\mathrm{net}(v), \mathrm{net}(v')))$.

2. The GP mean m_{GP} *is assumed to be a known constant*, thus not contributing to the GP hyperparameters and independent of net.

Due to the assumption 2., the GP depends only on the parameters θ^κ of the covariance function. As to the ANN, it depends on the one hand on the vector θ^W of its weights and biases, on the other hand on the network architecture, which we will treat as fixed before network training.

Consider now n inputs to the neural network, $v_1, \ldots, v_n$, mapped to the inputs $x_1 = \mathrm{net}(v_1), \ldots, x_n = \mathrm{net}(v_n)$ of the GP, corresponding to the observations $y = (y_1, \ldots, y_n)^\top$. Then the log-likelihood of $\theta = (\theta^\kappa, \theta^W)$ is

$$\begin{aligned}\mathcal{L}(\theta) &= \ln p(y; m_{\mathrm{GP}}, \kappa, \sigma_n^2) = \\ &= -\frac{1}{2}(y - m_{\mathrm{GP}})^\top K^{-1}(y - m_{\mathrm{GP}}) \\ &- \ln(2\pi) - \frac{1}{2}\ln\det(K + \sigma_n^2 I_n), \quad (6)\end{aligned}$$

where m_{GP} is the constant assumed in 2., and

$$(K)_{i,j} = \kappa(\mathrm{net}(v_i), \mathrm{net}(v_j)). \quad (7)$$

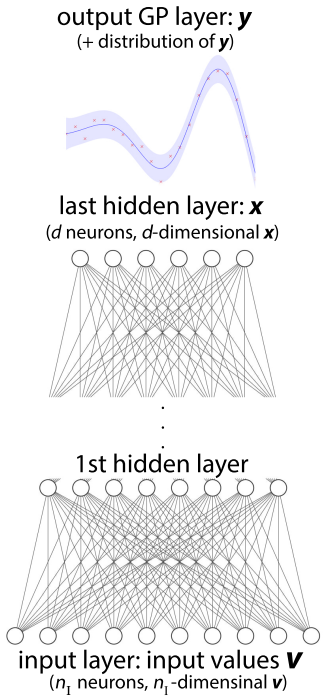

Fig. 1. Schema of the integration of a GP into an ANN as its output layer.

Let model training, searching for the vector $(\theta^\kappa, \theta^W)$, be performed in the most simple but, in the context of neural networks, also the most frequent way – as gradient descent. The partial derivatives forming $\nabla_{(\theta^\kappa, \theta^W)}\mathcal{L}$ can be computed as:

$$\frac{\partial \mathcal{L}}{\partial \theta_\ell^\kappa} = \sum_{i,j=1}^{n} \frac{\partial \mathcal{L}}{\partial K_{i,j}} \frac{\partial K_{i,j}}{\partial \theta_\ell^\kappa}, \quad (8)$$

$$\frac{\partial \mathcal{L}}{\partial \theta_\ell^W} = \sum_{i,j,k=1}^{n} \frac{\partial \mathcal{L}}{\partial K_{i,j}} \frac{\partial K_{i,j}}{\partial x_k} \frac{\partial\, \mathrm{net}(v_k)}{\partial \theta_\ell^W}. \quad (9)$$

In (8), the partial derivatives $\frac{\partial \mathcal{L}}{\partial K_{i,j}}$, $i, j = 1, \ldots, n$, are components of the matrix derivative $\frac{\partial \mathcal{L}}{\partial K}$, for which the calculations of matrix differential calculus [36] together with (4) and (6) yield

$$\frac{\partial \mathcal{L}}{\partial K} = \frac{1}{2}\left(K^{-1}yy^{\top}K^{-1} - K^{-1}\right). \tag{10}$$

3 Surrogate Modelling in the CMA-ES Context

3.1 Surrogate Models for Continuous Black-Box Optimization

Basically, the purpose of surrogate modelling – to approximate an unknown functional dependence – coincides with the purpose of *response surface modelling* in the design of experiments [26, 38]. Therefore, it is not surprising that typical response surface models, i.e., *low order polynomials*, belong also to the most traditional and most successful surrogate models [1, 2, 21, 30, 43]. Other frequently used kinds of surrogate models are *artificial neural networks* of the kind multilayer perceptron (MLP) or radial basis function network [19, 27–29, 40, 50], and the models to which the previous section was devoted – GPs, in surrogate modelling also known as *kriging* [5,6,12–14,31,32,35,47,48]. Occasinally encountered are *support vector regression* [9, 34] and *random forests* [4, 41].

From the point of view of active learning, the most attractive kind of surrogate models are GPs, due to the fact that a GP estimate $f(x)$ of the value of a true objective function for an input x is not a point, but a random variable. Its Gaussian distribution allows to define alternative criteria according to which individuals for evaluation by the true objective function can be selected, most importantly:

- *Probability of improvement* of the estimate $f(x)$ with respect to a reference value V (typically the minimal so far found value of the true objective function),

$$\mathrm{PoI}(f(x); V) = P(f(x) \leq V), \tag{11}$$

  which can be estimated using the Gaussian distribution of the GP.
- *Expected improvement* with respect to V,

$$\mathrm{EI}(f(x), V) = \mathbb{E}(V - f(x))\mathbb{I}(f(x) < V)., \tag{12}$$

3.2 Covariance Matrix Adaptation Evolution Strategy and Its Surrogate-Assisted Variant DTS-CMA-ES

The CMA-ES algorithm performs unconstrained optimization on $\mathbb{R}^d$, by means of iterative sampling of populations sized λ from a d-dimensional Gaussian distribution $\mathcal{N}(m, \sigma^2 C)$, and uses a given parent number μ among the sampled points corresponding to the lowest objective function values, to update the parameters of that distribution. Hence, it updates the expected value m, which is

used as the current point estimate of the function optimum, the matrix C and the step-size σ. The CMA-ES is invariant with respect to monotonous transformations of the objective function. Hence, to make use of the evaluations of the objective function in a set of points, it needs to know only the ordering of those evaluations. Details of the algorithm can be found in [20, 23].

During the more than 20 years of CMA-ES existence, a number of surrogate-assisted variants of this algorithm have been proposed, a survey can be found in [5,42]. Here, we will pay attention only to the most recent GP-based among them, the Doubly Trained Surrogate CMA-ES (DTS-CMA-ES) [5], a surrogate-assisted variant of CMA-ES. It employs two GPs f_1 and f_2, trained consecutively, to find an evaluation of the population $x_1, \ldots, x_\lambda$, with f_1 used for active learning of training data for f_2. Due to the CMA-ES invariance with respect to monotonous transformations, evaluates the difference between predictions only according to the difference in the ordering of those predictions, more precisely, according to the ranking difference error (RDE). The RDE of $y \in \mathbb{R}^\lambda$ with respect to $y' \in \mathbb{R}^\lambda$ considering k best components is defined:

$$\underset{\leq k}{\mathrm{RDE}}(y, y') = \frac{\sum_{i,(\rho(y'))_i \leq k} |(\rho(y'))_i - (\rho(y))_i|}{\max_{\pi \in \Pi(\lambda)} \sum_{i=1}^{k} |i - \pi^{-1}(i)|}, \tag{13}$$

where $\Pi(\lambda)$ denotes the set of permutaions of $\{1, \ldots, \lambda\}$ and $\rho(y)$ denotes the ordering of the components of y, i.e., $(\forall y \in \mathbb{R}^\lambda)\ \rho(y) \in \Pi(\lambda)\ \&\ (\rho(y))_i < (\rho(y))_j \Rightarrow y_i \leq y_j$.

The algorithm DTS-CMA-ES is described in Algorithm 1, using the following notation:

- $\mathcal{A}$ for an *archive* – a set of points that have already been evaluated by the true black-box objective function BB;
- $d_{\sigma^2 C}$ for the Mahalanobis distance given by $\sigma^2 C$:

$$d_{\sigma^2 C}(x, x') = \sqrt{(x - x')^\top \sigma^{-2} C^{-1} (x - x')}; \tag{14}$$

- $N_k(x; \mathcal{A})$ for the set of a given number k of $d_{\sigma^2 C}$-nearest neighbours of $x \in \mathbb{R}^d$ with respect to the archive $\mathcal{A}$;
- $f_i(x_1, \ldots, x_\lambda) = (f_i(x_1), \ldots, f_i(x_\lambda))$, for $i = 1, 2$;
- $\mathcal{T}_h = \bigcup_{j=1}^{\lambda} \{x \in N_h(x_j; \mathcal{A}) | d_{\sigma^2 C}(x, x_j) < r_{\max}\}$ with $r_{\max} > 0$ for $h = 1, \ldots, |\mathcal{A}|$;
- $k(\mathcal{A}) = \max\{h | |\mathcal{T}_h| \leq N_{\max}\}$, with $N_{\max} \in \mathbb{N}$;
- ρ_{PoI} for decreasing ordering of $f_1(x_1), \ldots, f_1(x_\lambda)$ according to the probability of improvement with respect to the lowest BB value found so far,

$$i < j \Rightarrow \mathrm{PoI}(\rho_{\mathrm{PoI}}(f_1(x_1, \ldots, x_\lambda)))_i; V) \geq \mathrm{PoI}(\rho_{\mathrm{PoI}}(f_1(x_1, \ldots, x_\lambda)))_j; V), \tag{15}$$

where $V = \min_{x \in \mathcal{A}} BB(x)$.

Algorithm 1 Algorithm DTS-CMA-ES

Require: $x_1, \ldots, x_\lambda \in \mathbb{R}^d$, μ, $\mathcal{A}$, σ and C – step size and matrix from the CMA-ES distribution, $N_{\max} \in \mathbb{N}$ such that $N_{\max} \geq \lambda, r_{\max} > 0, \beta, \epsilon_{\min}, \epsilon_{\max}, \alpha_{\min}, \alpha_{\max} \in (0, 1)$

1: **if** this is the 1st call of the algorithm in the current CMA-ES run **then**
2: set $\alpha = \epsilon = 0.05$
3: **else**
4: take over the returned values of α, ϵ from its previous call in the run
5: **end if**
6: Train a Gaussian process f_1 on $\mathcal{T}_{k(\mathcal{A})}$, estimating $m_{\mathrm{GP}}, \sigma_n, \sigma_f, \ell$ through maximization of the likelihood (4)
7: Evaluate $BB(x_j)$ for x_j such that $(\rho_{\mathrm{PoI}}(f_1(x_1, \ldots, x_\lambda)))_j \leq \lceil \alpha\lambda \rceil$ and not yet BB-evaluated
8: Update $\mathcal{A}$ to $\mathcal{A} \cup \{(x_j | (\rho_{\mathrm{PoI}}(f_1(x_1), \ldots, f_1(x_\lambda)))_j \leq \lceil \alpha\lambda \rceil\}$
9: Train a Gaussian process f_2 on $\mathcal{T}_{k(\mathcal{A})}$, estimating m_{GP}, σ_n, σ_f, ℓ through maximization of the likelihood (4)
10: For x_j such that $(\rho_{\mathrm{PoI}}(f_1(x_1, \ldots, x_\lambda)))_j \leq \lceil \alpha\lambda \rceil$, update $f_2(x_j) = BB(x_j)$
11: Update ϵ to $(1 - \beta)\epsilon + \beta\, \mathrm{RDE}_\mu(f_1(x_1, \ldots, x_\lambda), (f_2(x_1, \ldots, x_\lambda))$ and α to $\alpha_{\min} + \max(0, \min(1, \frac{\epsilon - \epsilon_{\min}}{\epsilon_{\max} - \epsilon_{\min}}))$
12: Update the value $f_2(x_j)$ to $f_2(x_j) - \min\{f_2(x_{j'}) | (\rho_{\mathrm{PoI}}(f_1(x_1, \ldots, x_\lambda))_{j'} > \lceil \alpha\lambda \rceil\} + \min\{f_2(x_{j'}) | (\rho_{\mathrm{PoI}}(f_1(x_1, \ldots, x_\lambda))_{j'} \leq \lceil \alpha\lambda \rceil\}$ for j fulfilling $(\rho_{\mathrm{PoI}}(f_1(x_1, \ldots, x_\lambda))_j > \lceil \alpha\lambda \rceil$
13: Return $f_2(x_1), \ldots, f_2(x_\lambda), \epsilon, \alpha$

4 Experiments with ANN+GP Integration in the DTS-CMA-ES

4.1 Experimental Setup

For the experiments, we have used the 24 noiseless benchmark functions available on a platform *Comparing Continuous Optimizers (COCO)* [10, 22]. Those benchmarks form five groups with different properties:

1. separable functions: f_1 sphere, f_2 separable ellipsoid, f_3 separable Rastrigin, f_4 Büche-Rastrigin, f_5 linear slope;
2. moderately ill-conditioned functions: f_6 attractive sector, f_7 step ellipsoid, f_8 Rosenbrock, f_9 rotated Rosenbrock;
3. highly ill-conditioned functions: f_{10} ellipsoid with high conditioning, f_{11} discus, f_{12} bent cigar, f_{13} sharp ridge, f_{14} different powers;
4. multi-modal functions with global structure: f_{15} non-separable Rastrigin, f_{16} Weierstrass, f_{17} Schaffers F7, f_{18} ill-conditioned Schaffers F7, f_{19} composite Griewank-Rosenbrock;
5. multi-modal weakly structured functions: f_{20} Schwefel, f_{21} Gallagher's Gaussian 101-me points, f_{22} Gallagher's Gaussian 21-hi points, f_{23} Katsuura, f_{24} Lunacek bi-Rastrigin.

All benchmark functions were optimized on the closed cube $[-5, 5]^d$, where d is the dimension of the input space, and the initial CMA-ES population was

sampled uniformly on $[-4, 4]^d$. For each noiseless function, 25 different variants were used, obtained as follows:

- each of the functions is scalable for any dimension $d \geq 2$, we have used the five dimensions 2, 3, 5, 10, 20;
- for each function in each dimension, 5 different instances were used, mutually differing through translations and/or rotations.

Each variant f of each benchmark function was optimized for $250d$ evaluations unless it was terminated earlier due to indicated convergence. To evaluate the success of optimization at its end, we used the approach used in [22], to calculate the proportion of achieved optimization target in the interval $[10^{-8}, 10^2]$. However, instead of calculating such a score from a given number of discrete targets log-uniformly distributed in that interval as in [22], we calculated its continuous counterpart as the ratio of the logarithmic length $\lambda_{\log}$ between the subinterval of $[10^{-8}, 10^2]$ corresponding to the achieved distance f^* to the minimum of f at the end of the optimization and the whole interval $[10^{-8}, 10^2]$,

$$r = \frac{\lambda_{\log}([10^{-8}, 10^2] \cap [f^*, +\infty))}{\lambda_{\log}([10^{-8}, 10^2])} = \frac{\max(0, \min(10, 2 - \log_{10} f^*))}{10}. \tag{16}$$

For all experiments, we used the existing implementation of DTS-CMA-ES at https://github.com/bajeluk/surrogate-cmaes, into which we implemented the ANN+GP surrogate models using the system GPyTorch [17], our implementaion is available at https://github.com/c0zzy/surrogate-networks. As to the tunable parameters of DTS-CMA-ES, we used the same values as [5]. As to the tunable parameters of GPyTorch, we fixed the five listed in Table 1 and utilized the default values of the remaining.

Finally, for the neural networks in the ANN+GP combinations, we used fully connected multilayer perceptrons with one hidden layer of a sufficiently low number of neurons to assure their trainability with comparatively small archives typically available in the DTS-CMA-ES. More precisely, all the ANNs used the fully connected topologies $(d - n_O - n_O)$ with

$$n_O = \begin{cases} 2 & \text{if } d = 2, \\ 3 & \text{if } d = 3, 5, \\ 5 & \text{if } d = 10, 20. \end{cases} \tag{17}$$

4.2 First Results and Their Discussion

The first work-in-progress results presented in this paper, compare ANN+GP combinations with different covariance functions of the GP. Table 2 reports the scores of each such combination for each noiseless benchmark function, averaged over the 25 combinations of 5 instances and 5 dimensions. In Tables 3 and 4, the scores are reported for the above defined groups of benchmark functions, and for the considered dimension, respectively. Hence, the score averaging in Table 3

Table 1. Settings for important GPyTorch parameters. For all its remaining tunable parameters, the default values were used

Optimizer	Adam
Learning rate	0.0005
Iterations	1000
Noise	0.0001
Length scale bounds	0.01, 100

includes in addition all functions of the respective group, whereas in Table 4 the scores are averaged over the 120 instances of the 24 benchmark functions. In all three tables, the ANN+GP combination with the highest average score is in bold.

Table 2. Score of the 5 compared ANN+GP combinations for each of the noiseless COCO benchmark functions, averaged over the 25 combinations of the 5 instances of that function and the 5 considered dimensions. For each function, the ANN+GP combination with the highest average score is in bold

	κ_{LIN}	κ_{QUAD}	κ_{SE}	κ_{MA5}	κ_{RQ}	$\kappa_{\text{SE+Q}}$		κ_{LIN}	κ_{QUAD}	κ_{SE}	κ_{MA5}	κ_{RQ}	$\kappa_{\text{SE+Q}}$
f_1	0.38	0.45	0.54	0.54	**0.56**	0.47	f_{13}	0.13	0.15	0.22	0.25	**0.27**	0.23
f_2	0.02	0.11	0.22	0.20	**0.26**	0.19	f_{14}	0.41	0.45	**0.55**	0.54	0.54	0.47
f_3	0.09	0.09	0.10	0.10	**0.11**	0.09	f_{15}	0.09	0.09	0.09	0.10	**0.10**	0.09
f_4	0.07	0.07	0.08	0.08	0.08	**0.09**	f_{16}	0.14	0.14	0.20	0.12	**0.15**	0.12
f_5	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	**1.00**	f_{17}	0.26	0.30	0.33	**0.34**	0.33	0.32
f_6	0.09	0.10	0.13	**0.17**	0.15	0.16	f_{18}	0.21	0.25	0.27	0.25	**0.28**	0.27
f_7	0.28	0.38	0.51	0.43	**0.54**	0.45	f_{19}	0.19	0.20	0.20	0.20	0.20	**0.21**
f_8	0.15	0.17	0.29	**0.29**	0.28	0.27	f_{20}	0.17	0.17	0.18	0.17	0.17	**0.18**
f_9	0.15	0.15	0.18	0.24	**0.30**	0.26	f_{21}	**0.19**	0.18	0.16	0.18	0.16	0.17
f_{10}	0.02	0.07	0.10	0.07	**0.10**	0.07	f_{22}	0.15	0.11	**0.16**	0.13	**0.16**	0.14
f_{11}	0.05	0.09	0.12	0.10	**0.13**	0.12	f_{23}	0.17	0.16	0.17	**0.17**	0.17	0.16
f_{12}	0.03	0.07	0.08	0.10	**0.13**	0.09	f_{24}	0.07	0.07	0.07	**0.07**	0.07	0.07

From Tables 2–4, we can see that the highest average score has been by far the most frequently achieved by ANN+GP combinations with rational quadratic covariance functions κ_{RQ}: for 14 out of the 24 functions, 3 out of the 5 groups of functions, and 4 out of the 5 considered dimensions. The rational quadratic covariance function shares the first place with the squared exponential covariance κ_{SE} for the function f_{22} Gallagher's Gaussian 21-hi points, as well as for the dimension 10. In addition, for the function f_5 linear slope, ANN+GP combinations achieve with all considered covariance function the highest possible average score 1. Apart from these shared first places, other covariance functions lead to ANN+GP combinations with the highest average score in the following cases:

Table 3. Score of the 5 compared ANN+GP combinations for each group of the noiseless COCO benchmark functions, averaged over the 100 or 125 (dependent on the group) combinations of all instances of the functions belonging to that group and the 5 considered dimensions. For each function, the ANN+GP combination with the highest average score is in bold

	κ_{LIN}	κ_{QUAD}	κ_{SE}	κ_{MA5}	κ_{RQ}	$\kappa_{\mathrm{SE+Q}}$
Separable	0.317	0.348	0.390	0.388	**0.403**	0.370
Moderately ill-conditioned	0.172	0.204	0.281	0.290	**0.323**	0.288
Highly ill-conditioned	0.134	0.170	0.218	0.218	**0.241**	0.197
Multi-modal functions globally structured	0.182	0.200	**0.221**	0.204	0.218	0.206
Multi-modal weakly structured	**0.153**	0.143	0.151	0.149	0.149	0.147

Table 4. Score of the 5 compared ANN+GP combinations for each considered dimension, averaged over the 120 considered instances of the 24 noiseless COCO benchmark functions. For each dimension, the ANN+GP combination with the highest average score is in bold

	κ_{LIN}	κ_{QUAD}	κ_{SE}	κ_{MA5}	κ_{RQ}	$\kappa_{\mathrm{SE+Q}}$
2	0.268	0.322	0.386	0.389	**0.398**	0.365
3	0.23	0.254	0.326	0.307	**0.365**	0.302
5	0.178	0.194	0.225	0.223	**0.244**	0.224
10	0.162	0.166	**0.185**	0.184	**0.185**	0.174
20	0.124	0.131	0.133	**0.138**	0.131	0.135

- the covariance κ_{SE} for the function f_{14} different powers as well as for the group of multi-modal globally structured functions;
- the Matérn $\frac{5}{2}$ covariance κ_{MA5} for the functions f_6 attractive sector, f_8 Rosenbrock, f_{17} Schaffers F7, f_{23} Katsuura, and f_{24} Lunacek bi-Rastrigin, as well as for the dimension 20;
- the composite covariance $\kappa_{\mathrm{SE+Q}}$ for the functions f_4 Büche-Rastrigin, f_{19} composite Griewank-Rosenbrock, and f_{20} Schwefel;
- the linear covariance κ_{LIN} for the function f_{21} Gallagher's Gaussian 101-me points, as well as for the group of multi-modal weakly structured functions, to which also f_{21} belongs.

The results in Tables 2–4 reflect both systematic differences due to different covariance functions and random differences due to noise. To assess the influence of different covariance functions without the interference of noise, the obtained differences were tested for statistical significance. To this end, the null hypotheses that the means of the random variables that produced the scores for the individual ANN+GP combinations are all identical were tested, using the Friedman's test with post-hoc identification of the pairs that lead to the rejection of the null hypothesis if it is rejected. Both the Friedman test and its post-hoc tests were performed on the family-wise significance level for multiple-hypotheses testing 5%, and the family-wise significances were assessed from the achieved significances of the individual tests (p-values) by means of the Holm method [16].

For individual functions, the Friedman test was always based on the 25 combinations of their 5 instances and the 5 considered dimensions. The test rejected the null hypothesis for all functions except the f_5 linear slope. The result of the post-hoc tests are summarized in Table 5. The value in each row r and column $c \neq r$ tells for how many among the remaining 23 functions the covariance function in the row was as part of an ANN+GP combination significantly better than the one in the column, or equivalently the one column was significantly worse than the one in the row. This means that the ANN+GP combination with the covariance function in the row yielded a higher average score than with the one in the column, and the post-hoc test rejected on the family-wise significance level 5% the hypothesis of equal mean values of the random variables that produced both scores. For individual function groups, the Friedman tests were based on the 100 or 125 combinations of the functions belonging to the group, their 5 instances and the 5 considered dimensions. For individual dimensions, they were based on the 120 combinations of the 24 functions and their dimensions. Both the 5 tests for the function groups and the 5 tests for the dimensions rejected their null hypotheses. The results of their post-hoc tests are summarized in Table 6. In Tables 5 and 6, the value in the cell is in bold if the covariance function in the row was more often significantly better than significantly worse than the one in the column.

Table 5. The number of functions among those 23 for which the null hypothesis of the Friedman test was rejected, for which the covariance function in the row was as part of an ANN+GP combination significantly better than the one in the column. That value is in bold if it is in addition higher than the number of functions for which the covariance function in the row was significantly worse than the one in the column

Covariance function	κ_{LIN}	κ_{QUAD}	κ_{SE}	κ_{MA5}	κ_{RQ}	$\kappa_{\mathrm{SE+Q}}$
κ_{LIN}	–	0	0	0	0	0
κ_{QUAD}	0	–	0	0	0	0
κ_{SE}	**2**	0	–	0	0	**14**
κ_{MA5}	**1**	0	0	–	9	**14**
κ_{RQ}	**6**	**1**	0	**14**	–	**18**
$\kappa_{\mathrm{SE+Q}}$	0	0	9	9	5	–

From Tables 5 and 6, we can observe that the covariance function κ_{LIN} was never either significantly better or significantly worse than κ_{QUAD}. This can be interpreted as interchangeability of both functions from the point of view of being used as covariances in the ANN+GP surrogate models for DTS-CMA-ES. The same relationship holds also for the pairs of covariance functions $(\kappa_{\mathrm{SE}}, \kappa_{\mathrm{MA5}})$ and $(\kappa_{\mathrm{SE}}, \kappa_{\mathrm{RQ}})$. It is particularly important in connection with the last mentioned pair, in view of the fact that κ_{RQ} was the covariance most often yielding the highest score, and κ_{SE} was in this respect the 3rd among the individual bench-

Table 6. The number of function groups (left), respectively considered dimensions (right) for which the covariance function in the row was as part of an ANN+GP combination significantly better than the one in the column. That value is in bold if it is in addition higher than the number of function groups, respectively dimensions for which the covariance function in the row was significantly worse than the one in the column

Covariance function	for function groups						for dimensions					
	κ_{LIN}	κ_{QUAD}	κ_{SE}	κ_{MA5}	κ_{RQ}	$\kappa_{\text{SE+Q}}$	κ_{LIN}	κ_{QUAD}	κ_{SE}	κ_{MA5}	κ_{RQ}	$\kappa_{\text{SE+Q}}$
κ_{LIN}	–	0	0	0	0	0	–	0	0	0	0	0
κ_{QUAD}	0	–	0	0	0	0	0	–	0	0	0	0
κ_{SE}	**4**	**2**	–	0	0	**4**	**4**	**2**	–	0	0	**4**
κ_{MA5}	**4**	**2**	0	–	0	**4**	**4**	**2**	0	–	1	**4**
κ_{RQ}	**4**	**2**	0	**5**	–	**5**	**4**	**2**	0	**4**	–	**4**
$\kappa_{\text{SE+Q}}$	**4**	**1**	1	1	0	–	**4**	**1**	1	1	1	–

mark functions and the 2nd among groups of functions and among the considered dimensions. Altogether, these two interchangeable covariance functions yielded the highest score for 15 from the 24 considered benchmarks, for 4 from the 5 benchmark function groups and for 4 from the 5 considered dimensions.

5 Conclusion ad Future Work

This work-in-progress paper presented an experimental investigation of surrogate models combining artificial neural networks with Gaussian processes in the context of a sophisticated surrogate-assisted variant of the black-box optimizer CMA-ES, the DTS-CMA-ES, which consecutively trains two surrogate models, using the first for active learning of training data for the second. In the experiments, a comprehensive comparison of ANN+GP surrogate models with six different covariance functions was performed on the 24 noiseless benchmark functions of the COCO platform [10, 22] in 5 dimensions. The results revealed that the most suitable covariance function for this combined kind of surrogate models is the rational quadratic, followed by the Matérn $\frac{5}{2}$, and squared exponential. Moreover, the rational quadratic and squared exponential were found interchangeable in the sense that for no function, no group of functions, no dimension, and no function-dimension combination, none of these covariance functions was significantly better than the other.

As usually with work in progress, still very much is left for the future. Most importantly, the ANN+GP surrogate models need to be compared with pure GP surrogates. Superficially, that should be no problem because the original implementation of DTS-CMA-ES uses GPs alone. However, the DTS-CMA-ES implementation relying on the system GPML [44], and our implementation of the ANN+GP surrogates relying on the system GPyTorch [17] do not allow a comparison in which the difference between pure GP and ANN+GP surrogates would reflect only the added combination of both kinds of models. According

to our experience, that difference is much more due to the incompatibility between GPML and GPyTorch. The GPML does not include any ANN extension, whereas the GPyTorch includes also pure GPs, however, their predictive acuracy is substantially lower than that of their counterparts with the same covariance function that are implemented in the GPML. Hence, to arrive to an unbiased comparison of ANN+GP and pure GP surrogate models will still need a lot of implementation effort.

Further directions of our future research include on the one hand deep GPs, mentioned already in the Introducion, on the other hand the reconsideration of the approach employed in GPyTorch – training the ANN and the GP forming the combined surrogate model together by means of likelihood maximization. Whereas maximum likelihood is indeed the commonly used objective function for GP learning [45], successful and efficient ANN learning algorithms typically rely on other objectives [18]. Therefore, we would also like to investigate a cyclic interleaving of a GP-learning phase with an ANN-learning phase, where the length of the latter will depend on the relationship of its success to the success of the former.

Finally, we intend to perform research into transfer learning of such combined surrogate models: an ANN-GP model with a deep neural network will be trained on data from many optimization runs, and then the model used in a new run of the same optimizer will be obtained through additional learning restricted only to the GP and the last 1-2 layers of the ANN.

Acknowledgment

The research reported in this paper has been supported by the Czech Science Foundation (GAČR) grant 18-18080S. For J. Tumpach, it has been also partially supported by the Charles University student grant 260575. Computational resources were supplied by the project e-INFRA LM2018140 provided within the program Projects of Large Research, Development and Innovations Infrastructures.

References

1. Auger, A., Brockhoff, D., Hansen, N.: Benchmarking the local metamodel cma-es on the noiseless BBOB'2013 test bed. In: GECCO'13. pp. 1225–1232 (2013)
2. Auger, A., Schoenauer, M., Vanhaecke, N.: LS-CMA-ES: A second-order algorithm for covariance matrix adaptation. In: Parallel Problem Solving from Nature - PPSN VIII. pp. 182–191 (2004)
3. Baerns, M., Holeňa, M.: Combinatorial Development of Solid Catalytic Materials. Design of High-Throughput Experiments, Data Analysis, Data Mining. Imperial College Press / World Scientific, London (2009)
4. Bajer, L., Pitra, Z., Holeňa, M.: Benchmarking Gaussian processes and random forests surrogate models on the BBOB noiseless testbed. In: GECCO'15 Companion. pp. 1143–1150 (2015)
5. Bajer, L., Pitra, Z., Repický, J., Holeňa, M.: Gaussian process surrogate models for the CMA evolution strategy. Evolutionary Computation **27**, 665–697 (2019)

6. Booker, A., Dennis, J., Frank, P., Serafini, D., V., T., Trosset, M.: A rigorous framework for optimization by surrogates. Structural and Multidisciplinary Optimization **17**, 1–13 (1999)
7. Bui, T., Hernandez-Lobato, D., Hernandez-Lobato, J., Li, Y., Turner, R.: Deep gaussian processes for regression using approximate expectation propagation. In: ICML. pp. 1472–1481 (2016)
8. Calandra, R., Peters, J., Rasmussen, C., Deisenroth, M.: Manifold gaussian processes for regression. In: IJCNN. pp. 3338–3345 (2016)
9. Clarke, S., Griebsch, J., Simpson, T.: Analysis of support vector regression for approximation of complex engineering analyses. Journal of Mechanical Design **127**, 1077–1087 (2005)
10. The COCO platform (2016), http://coco.gforge.inria.fr
11. Cutajar, K., Bonilla, E., Michiardi, P., Filippone, M.: Random feature expansions for deep gaussian processes. In: ICML. pp. 884–893 (2017)
12. El-Beltagy, M., Nair, P., Keane, A.: Metamodeling techniques for evolutionary optimization of computaitonally expensive problems: Promises and limitations. In: Proceedings of the Genetic and Evolutionary Computation Conference. pp. 196–203. Morgan Kaufmann Publishers (1999)
13. Emmerich, M., Giannakoglou, K., Naujoks, B.: Single- and multi-objective evolutionary optimization assisted by Gaussian random field metamodels,. IEEE Transactions on Evolutionary Computation **10**, 421–439 (2006)
14. Emmerich, M., Giotis, A., Özdemir, M., Bäck, T., Giannakoglou, K.: Metamodel-assisted evolution strategies. In: PPSN VII. pp. 361–370. ACM (2002)
15. Forrester, A., Sobester, A., Keane, A.: Engineering Design via Surrogate Modelling: A Practical Guide. John Wiley and Sons, New York (2008)
16. Garcia, S., Herrera, F.: An extension on "Statistical Comparisons of Classifiers over Multiple Data Sets" for all pairwise comparisons. Journal of Machine Learning Research **9**, 2677–2694 (2008)
17. Gardner, J.R., Pleiss, G., Bindel, D., Weinberger, K.Q., Wilson, A.G.: Gpytorch: Blackbox matrix-matrix gaussian process inference with gpu acceleration (2019)
18. Goodfellow, I., Bengio, Y., Courville, A.: Deep Learning. MIT Press, Cambridge (2016)
19. Gutmann, H.: A radial basis function method for global optimization. Journal of Global Optimization **19**, 201–227 (2001)
20. Hansen, N.: The CMA evolution strategy: A comparing review. In: Towards a New Evolutionary Computation. pp. 75–102. Springer (2006)
21. Hansen, N.: A global surrogate assisted CMA-ES. In: GECCO'19. pp. 664–672 (2019)
22. Hansen, N., Auger, A., Ros, R., Merseman, O., Tušar, T., Brockhoff, D.: COCO: a platform for comparing continuous optimizers in a black box setting. Optimization Methods and Software **35**, doi:10.1080/10556788.2020.1808977 (2020)
23. Hansen, N., Ostermaier, A.: Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation **9**, 159–195 (2001)
24. Hebbal, A., Brevault, L., Balesdent, M., Talbi, E., Melab, N.: Efficient global optimization using deep gaussian processes. In: IEEE CEC. pp. 1–12, doi 10.1109/CEC40672.2018 (2018)
25. Hernández-Muñoz, G., Villacampa-Calvo, C., Hernández-Lobato, D.: Deep gaussian processes using expectation propagation and monte carlo methods. In: ECML PKDD. pp. 1–17, paper no. 128 (2020)

26. Hosder, S., Watson, L., Grossman, B.: Polynomial response surface approximations for the multidisciplinary design optimization of a high speed civil transport. Optimization and Engineering **2**, 431–452 (2001)
27. Jin, Y., Hüsken, M., Olhofer, M., B., S.: Neural networks for fitness approximation in evolutionary optimization. In: Jin, Y. (ed.) Knowledge Incorporation in Evolutionary Computation, pp. 281–306. Springer (2005)
28. Jin, Y., Olhofer, M., Sendhoff, B.: Managing approximate models in evolutionary aerodynamic design optimization. In: CEC 2001. pp. 592–599 (2001)
29. Jin, Y., Olhofer, M., Sendhoff, B.: A framework for evolutionary optimization with approximate fitness functions. IEEE Transactions on Evolutionary Computation **6**, 481–494 (2002)
30. Kern, S., Hansen, N., Koumoutsakos, P.: Local metamodels for optimization using evolution strategies. In: PPSN IX. pp. 939–948 (2006)
31. Kruisselbrink, J., Emmerich, M., Deutz, A., Bäck, T.: A robust optimization approach using kriging metamodels for robustness approximation in the CMA-ES. In: IEEE CEC. pp. 1–8 (2010)
32. Leary, S., Bhaskar, A., Keane, A.: A derivative based surrogate model for approximating and optimizing the output of an expensive computer simulation. Journal of Global Optimization **30**, 39–58 (2004)
33. Lee, J., Bahri, Y., Novak, R., Schoenholz, S., Pennington, J., et al.: Deep neural networks as Gaussian processes. In: ICLR. pp. 1–17 (2018)
34. Loshchilov, I., Schoenauer, M., Sebag, M.: Intensive surrogate model exploitation in self-adaptive surrogate-assisted CMA-ES (saACM-ES). In: GECCO'13. pp. 439–446 (2013)
35. Lu, J., Li, B., Jin, Y.: An evolution strategy assisted by an ensemble of local gaussian process models. In: GECCO'13. pp. 447–454 (2013)
36. Magnus, J., Neudecker, H.: Matrix Differential Calculus with Applications in Statistics and Econometrics. John Wiley and Sons, Chichester (2007)
37. Matthews, A., Hron, J., Rowland, M., Turner, R.: Gaussian process behaviour in wide deep neural networks. In: ICLR. pp. 1–15 (2019)
38. Myers, R., Montgomery, D., Anderson-Cook, C.: Response Surface Methodology: Proces and Product Optimization Using Designed Experiments. John Wiley and Sons, Hoboken (2009)
39. Novak, R., Xiao, L., Lee, J., Bahri, Y., Yang, G., et al.: Bayesian deep convolutional networks with many channels are Gaussian processes. In: ICLR. pp. 1–35 (2019)
40. Ong, Y., Nair, P., Keane, A., Wong, K.: Surrogate-assisted evolutionary optimization frameworks for high-fidelity engineering design problems. In: Jin, Y. (ed.) Knowledge Incorporation in Evolutionary Computation. pp. 307–331. Springer (2005)
41. Pitra, Z., Repický, J., Holeňa, M.: Boosted regression forest for the doubly trained surrogate covariance matrix adaptation evolution strategy. In: ITAT 2018. pp. 72–79 (2018)
42. Pitra, Z., Hanuš, M., Koza, J., Tumpach, J., Holena, M.: Interaction between model and its evolution control in surrogate-assisted cma evolution strategy. In: GECCO '21: Genetic and Evolutionary Computation Conference. pp. 528–536 (2021)
43. Rasheed, K., Ni, X., Vattam, S.: Methods for using surrogate modesl to speed up genetic algorithm oprimization: Informed operators and genetic engineering. In: Jin, Y. (ed.) Knowledge Incorporation in Evolutionary Computation. pp. 103–123. Springer (2005)
44. Rasmussen, C., Williams, C.: Gpml 4.0, matlab toolbox for gaussian processes for machine learning, http://www.gaussianprocess.org/gpml/code/matlab/doc/

45. Rasmussen, C., Williams, C.: Gaussian Processes for Machine Learning. MIT Press, Cambridge (2006)
46. Ratle, A.: Kriging as a surrogate fitness landscape in evolutionary optimization. Artificial Intelligence for Engineering Design, Analysis and Manufacturing **15**, 37–49 (2001)
47. Ulmer, H., Streichert, F., Zell, A.: Evolution strategies assisted by Gaussian processes with improved pre-selection criterion. In: IEEE CEC. pp. 692–699 (2003)
48. Volz, V., Rudolph, G., Naujoks, B.: Investigating uncertainty propagation in surrogate-assisted evolutionary algorithms. In: GECCO'17. pp. 881–888 (2017)
49. Wilson, A., Hu, Z., Salakhutdinov, R., Xing, E.: Deep kernel learning. In: ICAIS. pp. 370–378 (2016)
50. Zhou, Z., Ong, Y., Nair, P., Keane, A., Lum, K.: Combining global and local surrogate models to accellerate evolutionary optimization. IEEE Transactions on Systems, Man and Cybernetics. Part C: Applications and Reviews **37**, 66–76 (2007)