Online Optimization with Costly and Noisy Measurements using Random Fourier Expansions

Laurens Bliek*, Hans R. G. W. Verstraete*, Michel Verhaegen, Member, IEEE and Sander Wahls, Member, IEEE

Abstract—This paper analyzes DONE, an online optimization algorithm that iteratively minimizes an unknown function with costly and noisy measurements. The algorithm maintains a surrogate of the unknown function in the form of a random Fourier expansion (RFE). The surrogate is updated whenever a new measurement is available, and then used to determine the next measurement point. The algorithm is comparable to Bayesian optimization algorithms, but its computational complexity per iteration does not depend on the number of measurements. We derive several theoretical results that provide insight on how the hyperparameters of the algorithm should be chosen. The algorithm is compared to a Bayesian optimization algorithm for a benchmark problem and two optics applications, namely, optical coherence tomography and optical beam-forming network tuning. It is found that the DONE algorithm is significantly faster than Bayesian optimization in all three discussed problems, while keeping a similar or better performance.

Index Terms—derivative-free optimization, bayesian optimization, surrogate model, learning systems, adaptive optics

I. INTRODUCTION

ANY optimization algorithms use the derivative of an objective function, but often this information is not available in practice. Often, not even a closed form expression for the objective function is known, and function evaluations are costly. Examples are objective functions that represent the performance of a simulation or an experimental set-up. Approximating derivatives with finite difference using evaluations of an expensive objective function is very costly in high-dimensional problems. More efficient algorithms for derivative-free optimization problems exist. In derivative-free optimization, a model is often built that can be optimized with traditional (derivative-based) optimization methods. An overview can be found in [1], [2]. Some examples of commonly used derivative-free optimization algorithms are the simplex method [3], NEWUOA [4], BOBYQA [5], and DIRECT [6]. Furthermore, measurements or simulations of a practical problem are usually corrupted by noise. Several techniques have been developed to cope with a higher noise level and make better use of the expensive objective functions evaluations. Filtering and pattern search optimization algorithms such as implicit filtering [7] and SID-PSM [8] handle local minima resulting from high frequency components very well. Bayesian optimization, also known as sequential Kriging optimization, deals with heteroscedastic noise and perturbations very well. One of the first and best known Bayesian optimization algorithms is EGO [9]. Bayesian optimization relies on a surrogate model of the unknown function that represents a certain prior distribution, for example Gaussian processes or Student’s-t processes [10], [11], [12], [13]. In these processes different kernels and kernel learning methods are used for the covariance function [14], [15]. The surrogate model is used to decide where the next measurement should be taken. The new measurement is used to update the surrogate model. Bayesian optimization has been successfully used in various applications, including active user modeling and reinforcement learning [16], robotics [17], hyper-parameter tuning [11], and optics [18].

Recently, the DONE algorithm was proposed [19]. It is similar to Bayesian optimization, but simpler, faster, and with new results easier to tune. The DONE algorithm uses random Fourier expansions [20] (RFEs) as a surrogate model. The nature of the DONE algorithm makes the understanding of the hyperparameters easier. Due to its simplicity, we can derive a way to choose the probability distribution of the frequencies of the RFE basis functions that could result in a lower variance, as well as an upper bound on the regularization parameter of the RFE fitting procedure.

We investigate a benchmark problem and two applications in optics. The first application is optical coherence tomography (OCT), a 3D imaging method based on interference often used to image the human retina [21], [22], [19]. In this paper we numerically compare DONE to BayesOpt [13], a Bayesian optimization library that was shown to outperform many other similar libraries in [13]. The second application we consider is the tuning of an optical beam-forming network (OBFN). OBFNs are used in wireless communication systems to steer phased array antennas in the desired direction by making use of positive interference of synchronized signals [23], [24], [25], [26], [27], [28].

This paper is organized as follows. The next section gives a short overview and provides new theoretical insights on random Fourier expansions, the surrogate model on which the DONE algorithm is based. Section III explains the DONE algorithm. Theoretically optimal as well as more practical ways to choose the hyperparameters of this algorithm are given in Section IV. In Section V the DONE algorithm and BayesOpt are compared for the two adaptive optics applications and for a benchmark problem. We conclude the paper in Section VI.

II. RANDOM FOURIER EXPANSIONS

In this section, we will describe the surrogate model that we will use for optimization. Many machine learning techniques

*Both authors contributed equally to this work. Corresponding authors: l.bliet@tudelft.nl, h.r.g.w.verstraete@tudelft.nl.
All authors are with the Delft Center for Systems and Control, Delft University of Technology, Mekelweg 2, 2628 CD, Delft, Netherlands.
utilize kernel-based models [29]. A model can be trained and evaluated by the use of a real mapping \( k(x_i, x_j) \) that stores information of the datapoints. To train the model, a linear system involving the kernel matrix with elements \( k(x_i, x_j) \) has to be solved, while evaluating the model in some point \( x \) amounts to computing

\[
g_{KM}(x) = \sum_{i=1}^{N} a_i k(x, x_i). \tag{1}
\]

The computation costs of these two operations grow cubically and linearly in the number of datapoints \( N \), respectively. This can be prohibitive for large values of \( N \). Several techniques have been developed to reduce this complexity, many of which make use of sparsity [30], [31], [32], [33]. Another recent trend is to use a low-rank approximation of the kernel matrix by using random features [20]. Assuming the kernel \( k \) has the form \( k(x_i - x_j) \) and has Fourier transform \( p \), it can be normalized such that \( p \) is a probability distribution [20]. Then, we have

\[
k(x_i - x_j) = \int_{\mathbb{R}^d} p(\omega) e^{-i\omega^T(x_i - x_j)} d\omega = \int_{\mathbb{R}^d} p(\omega) \cos(\omega^T(x_i - x_j)) d\omega = \frac{1}{2\pi} \int_{\mathbb{R}^d} \cos(\omega^T(x_i - x_j) + b)d\omega = \frac{1}{2\pi} \int_{\mathbb{R}^d} p(\omega) \cos(\omega^T x_i + b) \cdot \cos(\omega^T x_j + b) d\omega = E[2\cos(\omega^T x_i + b) \cos(\omega^T x_j + b)] \approx \frac{2}{D} \sum_{k=1}^{D} \cos(\omega_k^T x_i + b_k) \cos(\omega_k^T x_j + b_k), \tag{2}
\]

if \( \omega_k \) are independent and identically distributed (i.i.d.) samples of the random variable \( \omega \) with probability distribution \( p_\omega \), and \( b_k \in [0, 2\pi] \) are i.i.d. samples of a uniform distribution. For \( c_k = \sum_{i=1}^{N} \frac{2}{D} a_i \cos(\omega_k^T x_i + b_k) \) we have:

\[
g_{KM}(x) \approx \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k). \tag{3}
\]

Note that the number of coefficients \( D \) is independent of the number of measurements \( N \). This is especially advantageous in online applications where the number of data samples \( N \) keeps increasing. We use the following definition of a random Fourier expansion:

**Definition 1.** A Random Fourier Expansion (RFE) is a function \( g : \mathbb{R}^d \to \mathbb{R} \),

\[
g(x) = \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k), \tag{4}
\]

with \( D \in \mathbb{N} \), \( b_k \) drawn from \( p_b = \text{Uniform}(0, 2\pi) \) and with \( \omega_k \in \mathbb{R}^d \) drawn from any continuous probability distribution \( p_\omega \). The \( b_k \) and the \( \omega_k \) are mutually independent.

### A. Ideal RFE Weights

In this section, we derive ideal but in practice unknown weights \( c \). We start with the case of infinitely many samples (see also [34], [35]).

**Theorem 1.** Let \( f \in L^2(\mathbb{R}^d) \) be a real-valued function and let

\[
\bar{c}(\omega, b) = \left\{ \begin{array}{ll}
\frac{1}{\pi}|\hat{f}(\omega)| \cos(\angle \hat{f}(\omega) - b), & b \in [0, 2\pi], \\
0, & \text{otherwise}.
\end{array} \right.
\]

Then, for all \( x \in \mathbb{R}^d \)

\[
f(x) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \bar{c}(\omega, b) \cos(\omega^T x + b) d\omega d\omega. \tag{6}
\]

Here, \( |\hat{f}| \) and \( \angle \hat{f} \) denote the magnitude and phase of the Fourier transform \( \hat{f}(\omega) = \int_{\mathbb{R}^d} f(x) e^{-i\omega^T x} dx \).

**Proof.** For \( b \in [0, 2\pi] \), we have

\[
\bar{c}(\omega, b) = \frac{1}{\pi} \Re \left\{ \hat{f}(\omega) e^{-ib} \right\}. \tag{7}
\]

Using \( \int_{0}^{2\pi} e^{-2ib} db = 0 \),

\[
f(x) = \Re \{ f(x) \} = \Re \left( \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{f}(\omega) e^{i\omega^T x} \right) = \Re \left( \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \hat{f}(\omega) e^{i\omega^T x} \int_{0}^{2\pi} \bar{c}(\omega, b) \cos(\omega^T x + b) d\omega \right)
\]

\[
= \Re \left( \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \int_{0}^{2\pi} \bar{c}(\omega, b) \cos(\omega^T x + b) d\omega d\omega \right)
\]

\[
= \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \int_{0}^{2\pi} \bar{c}(\omega, b) \cos(\omega^T x + b) d\omega d\omega. \tag{8}
\]

where, for \( b \in [0, 2\pi] \),

\[
\bar{c}(\omega, b) = \frac{1}{\pi} \Re \left\{ \hat{f}(\omega) e^{-ib} \right\} = \frac{1}{\pi} \Re \left\{ \int_{\mathbb{R}^d} f(x) e^{-i(\omega^T x + b)} dx \right\}
\]

\[
= \frac{1}{\pi} \int_{\mathbb{R}^d} f(x) \cos(\omega^T x + b) dx. \tag{9}
\]

The function \( \bar{c} \) in Theorem 1 is not unique. However, of all functions \( c \) that satisfy (6), the given \( \bar{c} \) is the one with minimum norm.

**Theorem 2.** Let \( \bar{c} \) be as in Theorem 1. If \( \bar{c} : \mathbb{R}^d \times [0, 2\pi] \to \mathbb{R} \) satisfies

\[
f(x) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \int_{0}^{2\pi} \bar{c}(\omega, b) \cos(\omega^T x + b) d\omega d\omega \quad \text{a.e.} \tag{10}
\]
Proof. First, using Parseval’s theorem and \( \int_0^{2\pi} \cos(a-b) \, db = \pi \) for a constant \( a \),  
\[
\|\hat{c}\|_2^2 = \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \hat{c}(\omega,b)^2 \, db \, d\omega = \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \frac{1}{\pi} |\hat{f}(\omega)|^2 \cos(\angle \hat{f}(\omega) - b)^2 \, db \, d\omega = \int_{\mathbb{R}^d} \frac{1}{\pi} |\hat{f}(\omega)|^2 \, d\omega = \frac{(2\pi)^d}{\pi} \int_{\mathbb{R}^d} |\hat{f}(\omega)|^2 \, d\omega.
\]

Furthermore, equality only holds if \( \hat{c} = \tilde{c} \), so the minimum norm solution is unique in \( L^2 \).

For a finite number of basis functions as in Definition 1 we can prove unbiasedness of the RFE as in the following theorem. Variance properties are analyzed in Appendix B.

**Theorem 3.** For any continuous probability distribution \( p_\omega \) with \( p_\omega(\omega) > 0 \) if \( |\hat{f}(\omega)| > 0 \), the choice
\[
c_k = \frac{2}{D(2\pi)^d} \frac{|\hat{f}(\omega_k)|}{p_\omega(\omega_k)} \cos(\angle \hat{f}(\omega_k) - b_k)
\]

makes the RFE \( g(x) = \sum_{k=1}^D c_k \cos(\omega_k^T x + b_k) \) an unbiased estimator of \( f(x) \).

**Proof.** Using Theorem 1 we have:
\[
f(x) = \frac{1}{(2\pi)^d} \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \hat{c}(\omega,b) \cos(\omega^T x + b) \, db \, d\omega = \mathbb{E} \left[ \frac{2\pi}{(2\pi)^d} \hat{c}(\omega,b) \cos(\omega^T x + b) \right] = \mathbb{E} \left[ \sum_{k=1}^D \frac{2\pi}{D(2\pi)^d} \frac{\hat{c}(\omega_k,b_k)}{p_\omega(\omega_k)} \cos(\omega_k^T x + b_k) \right] = \mathbb{E} \left[ \sum_{k=1}^D \frac{2}{D(2\pi)^d} \frac{|\hat{f}(\omega_k)|}{p_\omega(\omega_k)} \cos(\angle \hat{f}(\omega_k) - b_k) \cos(\omega_k^T x + b_k) \right] = \mathbb{E} [g(x)].
\]
The following theorem shows that the RFE using the least squares solution can approximate the function \( f \) arbitrarily well. Similar results were given in [36], [37], [34], [35], but not for the weights resulting from the linear least squares problem as described in this section. Instead, the ideal weights for which we gave a closed-form expression in Theorem 3 were used.

**Theorem 4.** Suppose that \( f \in L^2 \cap L^\infty \) and that \( \sup_{b \in B} \frac{\| \bar{c}(\omega_k) \|_{p_0}}{(p_0(p_0+1))} < \infty \). Then for every \( \epsilon > 0 \) and \( \delta > 0 \) there exist \( N_0 \) and \( D_0 \) such that
\[
\int_{\mathbb{R}^d} \left( f(x) - \sum_{k=1}^{D} c_{N,k} \cos(\omega_k^T x + b_k) \right)^2 p_x(x) dx < \epsilon \quad (20)
\]
for all \( N \geq N_0 \), \( D \geq D_0 \), \( 0 < \lambda \leq \Lambda \) with probability at least \( 1 - \delta \). Here, \( \Lambda \) is the solution to
\[
\left\| \left( \frac{1}{N} A_N^T A_N + \Lambda I_{D \times D} \right)^{-1} \frac{1}{N} A_N^T y_N \right\|_2^2 = \sum_{k=1}^{D} \left( \frac{\bar{c}(\omega_k, b_k)}{(2\pi)^d D p_0(\omega_k)p_0(b_k))} \right)^2,
\]
and \( c_{N,k} \) is the \( k \)-th element of the weight vector given in (19).

The proof of this theorem is given in Appendix A. In Section IV-B we show how to obtain \( \Lambda \) in practice.

### III. ONLINE OPTIMIZATION ALGORITHM

We will derive an iterative algorithm to minimize a function \( f \) by training a RFE model. Updating this model has a constant computation time of order \( O(D^2) \) per iteration, with \( D \) the number of basis functions. This is in contrast to Bayesian optimization algorithms, where a covariance matrix of which the size depends on the number of measurements is updated. Instead of sampling the whole input space, online methods have the advantage that only points of interest will be measured, reducing the number of measurements required for an accurate model.

#### A. Recursive Least Squares Approach for the Weights

In the online scenario, a new measurement \((x_n, y_n)\) becomes available at each iteration. These are used to update the RFE. Let \( a_n = [\cos(\omega_1^T x_n + b_1) \cdots \cos(\omega_D^T x_n + b_D)] \), then the least squares approach in the previous section is equivalent to minimizing the expression
\[
J_n(c) = \frac{1}{n} \sum_{i=1}^{n} (y_i - a_i c)^2 + \lambda \| c \|_2^2.
\]

To get rid of the factor \( 1/n \) we introduce the factors \( \alpha_i = (i-1)/i \), \( i \geq 2 \), and note that
\[
\frac{1}{n} = \frac{1}{n} \frac{i}{i + 1} \frac{i + 1}{i + 2} \cdots \frac{n - 1}{n} = \prod_{j=i+1}^{n} \alpha_j \frac{1}{i}, \quad (23)
\]
for \( i = 1, \ldots, n \). For the case \( i = 1 \) we define \( \alpha_1 = 1 \). We absorb the factor \( 1/i \) into \((y_i - a_i c)^2\) by setting
\[
\tilde{y}_i = y_i/\sqrt{i}, \quad \tilde{a}_i = a_i/\sqrt{i}.
\]

Using (23), we can rewrite (22) as
\[
J_n(c) = \sum_{i=1}^{n} \prod_{j=i+1}^{n} \alpha_j \left( \tilde{y}_i - \tilde{a}_i c \right)^2 + \lambda \| c \|_2^2.
\]

Here, \( \alpha_j \) can be interpreted as a variable forgetting factor.

Let \( c_n \) be the solution of
\[
\arg\min_c J_n(c).
\]

Assuming we have found \( c_n \), we would like to use this information to find \( c_{n+1} \) without solving (26) again. The recursive least squares algorithm is a computationally efficient method that determines \( c_{n+1} \) from \( c_n \) as follows [38, Sec. 21]:
\[
\gamma_n = 1 / (1 + \alpha_n^{-1} \tilde{a}_n \tilde{P}_{n-1}^{-1} \tilde{a}_n^T), \quad (27)
\]
\[
\tilde{g}_n = \alpha_n^{-1} \gamma_n \tilde{P}_{n-1}^{-1} \tilde{a}_n, \quad (28)
\]
\[
c_n = c_{n-1} + \tilde{g}_n(\tilde{y}_n - \tilde{a}_n c_{n-1}), \quad (29)
\]
\[
\tilde{P}_n = \alpha_n^{-1} \left[ \tilde{P}_{n-1} - \tilde{g}_n \tilde{g}_n^T / \gamma_n \right], \quad (30)
\]

with initialization \( c_0 = 0, \tilde{P}_0 = \lambda^{-1} I_{D \times D} \). Note that \( \alpha_n^{-1} = \frac{n-1}{n} \gamma_n \) for \( n \geq 2 \) and \( \alpha_1^{-1} = 1 \).

We implemented a square-root version of the above algorithm, also known as the inverse QR algorithm [38, Sec. 21], which is known to be especially numerically reliable. Instead of performing the update rules (28)-(30) explicitly, we find a rotation matrix \( \Theta_n \) that lower triangularizes the upper triangular matrix in Eq. (31) below and generates a post-array with positive diagonal entries:
\[
\left[ \begin{array}{ccc} 1 & \alpha_n^{-1/2} \tilde{a}_n & \alpha_n^{-1/2} \tilde{P}_{n-1}^{1/2} \\ 0 & \alpha_n^{-1/2} & \alpha_n^{-1/2} \tilde{P}_{n-1}^{1/2} \\ \end{array} \right] \Theta_n = \left[ \begin{array}{ccc} \gamma_n^{-1/2} & 0 \\ 0 & \tilde{P}_n^{1/2} \\ \end{array} \right]. \quad (31)
\]

The rotation matrix \( \Theta_n \) can be found by performing a QR decomposition of the transpose of the matrix on the left-hand side of (31), or by the procedure explained in [38, Sec. 21]. The computation complexity of this update is \( O(D^2) \) per iteration.

#### B. DONE Algorithm

We now explain the different steps of the DONE algorithm.

The DONE algorithm is used to iteratively find a minimum of a function \( f \in L^2 \) on a compact set \( X \subseteq \mathbb{R}^d \) by updating a RFE \( g(x) = \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k) \) at each new measurement, and using this RFE as a surrogate of \( f \) for optimization. It is assumed that the function \( f \) is unknown and only measurements perturbed by noise can be obtained: \( y_n = f(x_n) + \eta_n \). The algorithm consists of four steps that are repeated for each new measurement: 1) take a new measurement, 2) update the RFE, 3) find a minimum of the RFE, 4) choose a new measurement point. We now explain each step in more detail.

**Initialization**

Before running the algorithm, an initial starting point \( x_1 \in X \) and the number of basis functions \( D \) have to be chosen. The parameters \( \omega_k \) and \( \delta_k \) are drawn from continuous probability distributions as defined in Definition 1. The probability distribution \( p_\omega \) and the regularization parameter \( \lambda \) have to be chosen.
a priori as well. See Sect. IV for practical ways of choosing the hyperparameters. These hyperparameters stay fixed over the whole duration of the algorithm. Let $P_0^{1/2} = \lambda^{-1/2}I_{D\times D}$, and $n = 1$.

**Step 1: New measurement**

Unlike in Section II-B, it is assumed that measurements are taken in a recursive fashion. At the start of iteration $i$, a new measurement $y_n = f(x_n) + \eta_n$ is taken at $x_n$.

**Step 2: Update the RFE**

As explained in Section III-A, we update the RFE model $g(x) = \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k)$ using the new measurement from Step 1 by using the inverse QR algorithm as given by Eqs. (27)-(30). Only the weights $c_k$ are updated. The $\omega_k$ and $b_k$ parameters stay fixed throughout the whole algorithm.

**Step 3: Optimization on the RFE**

After updating the RFE, an iterative optimization algorithm is used to approximate a (possibly local) minimum of the RFE, giving a point $\hat{x}_n$. All derivatives of the RFE can easily be calculated. Using an analytical expression of the Jacobian will increase the performance of the optimization method used in this step, while not requiring extra measurements of $f$ as in the finite difference method. For functions that are costly to evaluate, this is a big advantage. The method used in the proposed algorithm is an L-BFGS method [39], [40]. Other optimization methods can also be used. The starting value for this optimization procedure is the last measurement point plus a random perturbation:

$$x_{\text{init}} = P_\mathcal{X}(x_n + \zeta_n),$$

(32)

where $P_\mathcal{X}$ is the projection onto $\mathcal{X}$. The random perturbation prevents the optimization algorithm from starting exactly in the point where the model was trained. Increasing its value will increase the exploration capabilities of the DONE algorithm but might slow down convergence. In the proposed algorithm, $\zeta_n$ is chosen to be white Gaussian noise.

**Step 4: Choose a new measurement point**

The minimum found in the previous step can be used to update the RFE again. A perturbation to the current minimum is added to avoid the algorithm getting trapped unnecessarily in insignificant local minima or saddle points [41]:

$$x_{n+1} = P_\mathcal{X}(\hat{x}_n + \zeta_n).$$

(33)

The random perturbations can be seen as an exploration strategy and are again chosen to be white Gaussian noise. Increasing their variance $\sigma_\zeta$ increases the exploration capabilities of the DONE algorithm but might slow down convergence. In practice, we typically use the same distribution for $\xi$ and $\zeta$. Lastly $n = n + 1$. The algorithm proceeds at Step 1.

The full algorithm for the case $\mathcal{X} = [lb, ub]^d$ is shown in Algorithm 1.

---

**Algorithm 1 DONE Algorithm**

1: procedure DONE($f, x_1, N, \{lb, ub, D, \lambda, \sigma_\zeta, \sigma_\xi\}$)
2: \hspace{1cm} Draw $\omega_1 \ldots \omega_D$ from $p_\omega$ independently.
3: \hspace{1cm} Draw $b_1 \ldots b_D$ from Uniform$(0, 2\pi)$ independently.
4: \hspace{1cm} $P_0^{1/2} = \lambda^{-1/2}I_{D\times D}$
5: \hspace{1cm} $c_0 = [0 \ldots 0]^T$
6: \hspace{1cm} $\hat{x}_0 = x_1$
7: \hspace{1cm} for $n = 1, 2, 3, \ldots$
8: \hspace{2cm} $\hat{a}_n = [\cos(\omega_1 x_n + b_1) \ldots \cos(\omega_D x_n + b_D)] / \sqrt{n}$
9: \hspace{2cm} $\hat{y}_n = (f(x_n) + \eta_n) / \sqrt{n}$
10: \hspace{2cm} $\alpha_n^{-1/2} = n/(n-1)$
11: \hspace{2cm} $\sigma_\zeta = \alpha_1^{-1/2} = 1$
12: \hspace{2cm} $g(x) = \text{updateRFE}(c_n, P_n^{1/2}, \alpha_n^{-1/2}, \hat{a}_n, \hat{y}_n)$
13: \hspace{2cm} Draw $\zeta_n$ from $N(0, \sigma_\zeta^2 I_{D \times d})$.
14: \hspace{2cm} $x_{\text{init}} = \max(\min(x_n + \zeta_n, ub), lb)$
15: \hspace{2cm} $[\hat{x}_n, \hat{y}_n] = \text{L-BFGS}(g(x), x_{\text{init}}, lb, ub)$
16: \hspace{2cm} $\xi_n = \max(\min(\hat{x}_n, \zeta_n, ub), lb)$
17: \hspace{1cm} return $\hat{x}_n$

---

**Algorithm 2 updateRFE**

1: procedure UPDATERFE($c_{n-1}, P_{n-1}^{1/2}, \alpha_{n-1}^{-1/2}, \hat{a}_{n-1}, \hat{y}_{n-1}$)
2: Retrieve $g_n \gamma_{n-1}^{1/2}, \gamma_{n-1}^{-1/2}$ and $P_{n-2}^{1/2}$ from (31)
3: $c_n = c_{n-1} + g_n (\hat{y}_{n-1} - \hat{a}_{n-1} c_{n-1})$
4: $g(x) = [\cos(\omega_1^T x + b_1) \ldots \cos(\omega_D^T x + b_D)] c_n / \sqrt{n}$
5: return $g(x)$

---

**IV. CHOICE OF HYPER-PARAMETERS**

In this section we will provide an in-depth analysis of the hyper-parameters of the DONE algorithm, as well as practical ways of choosing them, based on theoretical results. The performance of DONE depends on the following hyperparameters:

- Number of basis functions $D$.
- Probability distribution $p_\omega$.
- Regularization parameter $\lambda$.
- Exploration parameters $\sigma_\xi$ and $\sigma_\zeta$.

The influence of $D$ is straightforward: increasing $D$ will lead to a better performance (a better RFE fit) of the DONE algorithm at the cost of more computation time. Overfitting should not be a concern for this parameter since we make use of regularization. The parameters $p_\omega$ and $\lambda$ are discussed in the remainder of this section. The exploration parameters determine the trade-off between exploration and exploitation, similar to the use of the acquisition function in Bayesian optimization [16], [15]. The parameter $\sigma_\xi$ influences the exploration of the RFE surrogate in Step 3 of the DONE algorithm, while $\sigma_\zeta$ determines exploration of the original function. Assuming both to be close to each other, $\sigma_\xi$ and $\sigma_\zeta$ are usually chosen to be equal. If information about local optima of the RFE surrogate or of the original function is available, this could be used to determine good values for these hyperparameters. Alternatively, a similar approach as in Bayesian optimization could be used, like calculating the expected improvement, but this will remain for future work.
A. Probability Distribution of Frequencies

Recall the parameters $\omega_k$ and $b_k$ from Definition 1. Since these parameters are samples from continuous probability distributions $p_\omega$ and $p_b = \text{Uniform}(0, 2\pi)$, respectively, we want to investigate the first and second order moments of the RFE to find a distribution $p_\omega$ that minimizes the variance.

Unfortunately, as shown in Theorem 7 in Appendix B, it turns out that the optimal probability distribution is

$$p_\omega^*(\omega) = \frac{|\hat{f}(\omega)|}{\int_{\mathbb{R}^2} |\hat{f}(\omega)| d\omega} \sqrt{\cos(2\angle \hat{f}(\omega) + 2\omega^T x) + 2d\omega}$$

which depends on the input $x$ and both the phase and magnitude of the Fourier transform of $f$. But, if both $|\hat{f}|$ and $\angle \hat{f}$ were known, then the function $f$ itself would be known, and standard optimization algorithms could be used directly.

Besides this, we would like to use a probability distribution for $\omega_k$ that does not depend on the input $x$, since the $\omega_k$ parameters are chosen independently from the input in the initialization step of the algorithm.

In calibration problems, the objective function $f$ suffers from an unknown offset, $f(x) = \hat{f}(x + \Delta)$, resulting in a phase offset in the frequency domain. However, the magnitude $|\hat{f}|$ can be measured in this case. Section V-B describes an example of such a problem. We will derive a way to choose $p_\omega$ for calibration problems.

In order to get a close to optimal probability distribution for $\omega_k$ that is independent of the input $x$ and of the phase $\angle \hat{f}$ of the Fourier transform of $f$, we look at a complex generalization of the RFE. In this complex problem, it turns out we can circumvent the disadvantages mentioned above by using a probability distribution that depends only on $|\hat{f}|$.

**Theorem 5.** Let $\tilde{g}(x) = \sum_{k=1}^D \tilde{c}_k e^{i\omega_k^T x + b_k}$, with $\omega_k$ drawn i.i.d. from a continuous probability distribution $\tilde{p}_\omega$ that satisfies $\tilde{p}_\omega(\omega) > 0$ if $|\hat{f}(\omega)| > 0$, and $b$ drawn uniformly from $[0, 2\pi]$. Then $\tilde{g}(x)$ is an unbiased estimator of $f(x)$ for all $x$ if

$$\hat{c}_k = \frac{\hat{f}(\omega_k)e^{-ib_k}}{D(2\pi)^d \tilde{p}_\omega(\omega_k)}$$

For this choice of $\hat{c}_k$, the variance of $\tilde{g}(x)$ is minimal if

$$\tilde{p}_\omega(\omega) = \frac{|\hat{f}(\omega)|}{\int_{\mathbb{R}^2} |\hat{f}(\omega)| d\omega},$$

giving a variance of

$$\text{Var}[\tilde{g}(x)] = \frac{1}{D(2\pi)^d} \left( \int_{\mathbb{R}^2} |\hat{f}(\omega)| d\omega \right)^2 - f(x)^2.$$  

Proof. The unbiasedness follows directly from the Fourier inversion theorem:

$$\mathbb{E}[\tilde{g}(x)] = D \int_{\mathbb{R}^d} \int_{0}^{2\pi} \frac{\hat{f}(\omega)e^{-ib}}{(2\pi)^d \tilde{p}_\omega(\omega)} \cos(2\angle \hat{f} + 2\omega^T x) d\omega = f(x)$$

The proof of minimum variance is similar to the proof of [42, Thm. 4.3.1].

Note that the coefficients $\tilde{c}_k$ can be complex in this case. The probability distribution $\tilde{p}_\omega$, which is optimal for a complex RFE, still is close-to-optimal when used in the real RFE from Definition 1.

**Theorem 6.** Let $\tilde{p}$ be as in (36) and let $g$ be as in Definition 1. Let $P$ be the set of probability distribution functions over $\omega$. Then we have

$$\text{Var}_{p_\omega \in P}[g(x)] \leq \sqrt{3} \min_{p_\omega \in P} \text{Var}_{p_\omega \in P}[g(x)].$$

The proof is given in Appendix B. We now discuss how to choose $p_\omega$ in practice.

If no information of $|\hat{f}|$ is available, the standard approach of choosing $p_\omega$ as a zero-mean normal distribution can be used. The variance $\sigma^2$ is an important hyper-parameter in this case, and any method of hyper-parameter tuning can be used to find it. However, most hyper-parameter optimization methods are computationally expensive because they require running the whole algorithm multiple times. In the case that $|\hat{f}|$ is not exactly known, but some information about it is available (because it can be estimated or measured for example), this can be circumvented. The variance $\sigma^2$ can simply be chosen in such a way that $p_\omega$ most resembles the estimate for $|\hat{f}|$, using standard optimization techniques or by doing this by hand. In this approach, it is not necessary to run the algorithm at all, which is a big advantage compared to most hyper-parameter tuning methods. All of this leads to a rule of thumb for choosing $p_\omega$ as given in Algorithm 3.

**Algorithm 3 Rule of thumb for choosing $p_\omega$**

1: if $|\hat{f}|$ is known exactly then
2: Set $p_\omega = |\hat{f}|/\int |\hat{f}(\omega)| d\omega.$
3: else
4: Measure or estimate $|\hat{f}|.$
5: Determine $\sigma^2$ for which the pdf of $\mathcal{N}(0, \sigma^2 I_{d\times d})$ is close in shape to $|\hat{f}|/\int |\hat{f}(\omega)| d\omega.$
6: Set $p_\omega = \mathcal{N}(0, \sigma^2 I_{d\times d}).$

**B. Upper Bound on the Regularization Parameter**

The regularization parameter $\lambda$ is used in the performance criterion (17) to prevent under-fitting or over-fitting of the RFE under noisy conditions or when dealing with few measurements. To guarantee convergence of the least squares solution
we need $\lambda \leq \Lambda$ as in Theorem 4. In practice, $\lambda = 0.01\Lambda$ works well. Here we will provide a method to estimate $\Lambda$ using only one run of the algorithm.

The regularization parameter $\lambda$ is related to a hard constraint on the norm $||c_n||^2_2$, with larger $\lambda$ yielding a smaller $||c_n||^2_2$. Theorem 2 shows that the norm $||c||^2_2 \approx (2\pi)^2||f||^2_2$ is a lower bound on the norm of the weights in the infinite-dimensional case. In Appendix A this is related to the finite-dimensional case, leading to the following equation for the upper bound $\Lambda$ for $\lambda$ from Theorem 4:

$$||\left(1/Na^Tc + \Lambda I_{D\times D}\right)^{-1}\frac{1}{N}A^Tc||^2_2 = \sum_{k=1}^{D} \frac{c(\omega_k, b_k)}{(2\pi)^D p_n(\omega_k) p_b(b_k)} = B^2. \quad (40)$$

Using the definitions of $\hat{a}_n$ and $\hat{y}_n$ from Section III-A, note that the left hand side of the equation above is equal to

$$\left\|\left(\sum_{n=1}^{N} \hat{a}_n^T\hat{a}_n + \Lambda I_{D\times D}\right)^{-1}\sum_{n=1}^{N} \hat{a}_n^T\hat{y}_n\right\|^2_2. \quad (41)$$

If the quantities $\hat{a}_n$ and $\hat{y}_n$ are stored while running the DONE algorithm, this expression can be computed for different $\lambda$ values. In Section IV-A, it is assumed that no information of $\angle \hat{f}$ is available, but that $||\hat{f}||$ can be measured or estimated. Since $c$ as defined in Theorem 1 depends on both these quantities, this implies that $B$ is unknown. However, if we assume that $D$ is large and that $p_{\omega}$ is a good approximation of $\tilde{\omega} = \int_{R^d} |\hat{f}(\omega)|/\int_{R^d} |\hat{f}(\omega)|d\omega$ as in Algorithm 3, we can make the following approximation of $B$ (after getting rid of the square cosine as in Eq. (11)):

$$B = \frac{2}{(2\pi)^D} \left[ \frac{1}{2\pi D} \int_{R^d} \left(\int_{R^d} |\hat{f}(\omega)|/p_{\omega}(\omega)\right) \left(\int_{R^d} |\hat{f}(\omega)|/p_{\omega}(\omega)\right) d\omega \right]^{1/2} \approx \frac{2}{(2\pi)^D} \left[ \frac{1}{2\pi D} \int_{R^d} \left(\int_{R^d} |\hat{f}(\omega)|/p_{\omega}(\omega)\right) \left(\int_{R^d} |\hat{f}(\omega)|/p_{\omega}(\omega)\right) d\omega \right]^{1/2} \approx \frac{\sqrt{2}}{(2\pi)^D} \int_{R^d} |\hat{f}(\omega)| d\omega = B_a. \quad (42)$$

Using the exact value or an estimate of $\int_{R^d} |\hat{f}(\omega)| d\omega$ as in Algorithm 3, we calculate the expression in (41) for multiple values of $\Lambda$ and take the value for which it is closest to $B_a$. This procedure for finding $\Lambda$ is shown in Algorithm 4.

**Algorithm 4 Rule of thumb for finding $\Lambda$**

1. Run Algorithm 3 to get $\int_{R^d} |\hat{f}(\omega)| d\omega$.
2. Run the DONE algorithm once with some default value for $\lambda$ while storing $\hat{a}_n$ and $\hat{y}_n$ for all $n$.
3. Determine $\Lambda$ for which the expression in (41) is close to $B_a^2 = \frac{2}{(2\pi)^D} \left(\int |\hat{f}(\omega)| d\omega\right)^2$.

V. Numerical Examples

In this section, we will compare the DONE algorithm to a Bayesian optimization library called BayesOpt [13] in a benchmark problem and in two applications.

A. Benchmark Problem: Camelback Function

The camelback function

$$f(x, y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + (-4 + 4y^2)y^2 \quad (43)$$

with $(x, y) \in [-2, 2] \times [-1, 1]$ is a standard test function for nonlinear optimization. It has two global minima and two local minima. We analyzed the hyperparameters for DONE on this test function, and the algorithm was compared to BayesOpt.

First we computed the Fourier transform of the function. Then we fitted a function $h(\omega) = \frac{C}{\sigma \sqrt{2\pi}}e^{-\frac{\omega^2}{2\sigma^2}}$ to approximate the magnitude of the Fourier transform in both directions. To validate, two RFEs were fit to the original function using a normal distribution with standard deviation $\sigma = 10$ (good fit) and $\sigma = 0.1$ (bad fit) for $\omega$: the good fit gave a training error of $8.45 \cdot 10^{-7}$, the bad fit gave a training error of $1.84 \cdot 10^{-2}$, which shows the big impact of this hyperparameter on the least squares fit.

The norm $||c_n||^2_2$ depends on the value of $\lambda$, as explained in Sec. IV-B. Fig. 1 plots these two quantities, as well as the lower bound $\frac{\sqrt{2}}{(2\pi)^D} \int |\hat{f}(\omega)| d\omega$ for the norm of the weights, giving an upper bound on $\Lambda$ $\approx 1 \cdot 10^{-5}$. Since the function $h(\omega) = \frac{C}{\sigma \sqrt{2\pi}}e^{-\frac{\omega^2}{2\sigma^2}}$ approximated $|\hat{f}|$, we used $C$ as an approximation of the normalization factor $\int |\hat{f}(\omega)| d\omega$. To choose a value for $\lambda$ below its upper bound, we chose $\lambda = \Lambda/100 \approx 1 \cdot 10^{-5}$. We validated that this choice for $\lambda$ gave a good fit.

We also looked at the difference between using the real RFE from Definition 1 and the complex RFE from Theorem 5. Using the hyperparameters $\sigma = 10$ and $\lambda = 1 \cdot 10^{-5}$, we compared a least squares fit using 500 measurements for different values of $D$. Fig. 2 shows the mean and variance of the mean square error over 100 runs. We see that the real RFE indeed performs close-to-optimal as predicted by Theorem 6.

Using the hyperparameters $\sigma = 10$ and $\lambda = 1 \cdot 10^{-5}$, we also performed 10 runs of the DONE algorithm and compared it to the results from [13, Table 1] (method “BayesOpt1”). The number of basis functions $D$ was set to 50 and the initial guess was chosen randomly. The resulting distance to the true optimum and the computation time (with their standard deviations) for 50 and 100 measurements can be found in
These aberrations can be removed by using active components such as deformable mirrors in combination with optimization algorithms [22], [19]. The arguments of the optimization can be the voltages of the deformable mirror or a mapping of these voltages to other coefficients such as the coefficients of Zernike polynomials. The intensity of the image at a certain depth is then maximized to remove as much of the aberrations as possible. In [19] it was shown experimentally that the DONE algorithm greatly outperforms other derivative-free algorithms in final root mean square (RMS) wavefront error and image quality. Here, we numerically compare the DONE algorithm to BayesOpt [13]. The numerical measurements are obtained by simulating the OCT transfer function as described in [43], [44] and optimizing towards the maximum OCT signal. Three Zernike aberrations are considered, namely the defocus and two astigmatisms. The noise of a real OCT signal is approximated by adding Gaussian white noise with a standard deviation of 0.01. The results are shown in Fig. 3. For the DONE algorithm the same parameters are used as described in [19], only $\lambda$ is chosen to be equal to 3. The DONE algorithm is compared to BayesOpt with the default parameters. Other values for the parameters of BayesOpt did not result in a significant performance increase with trial and error. To use the BayesOpt algorithm, the inputs had to be normalized between 0 and 1. For each input aberration, the region $-0.45 \mu m$ to $0.45 \mu m$ was scaled to the region 0 to 1. The results for BayesOpt and DONE are very similar. The mean error of the DONE algorithm is slightly lower than the BayesOpt algorithm. However, the total average computation time for the DONE algorithm was 106 ms, while the total average computation time of Bayesopt is 1019 ms.

\begin{table}[h]
\centering
\caption{DONE vs BayesOpt on the Camelback function}
\begin{tabular}{|l|l|l|}
\hline
 & Dist (50 samp.) & Time (50 samp.) \\
\hline
DONE & $1.6032 \cdot 10^{-4}$ (2.3535 \cdot 10^{-3}) & 0.0256 (0.0016) \\
BayesOpt & 0.0021 (0.0044) & - \\
\hline
DONE & $3.0379 \cdot 10^{-5}$ (6.9305 \cdot 10^{-3}) & 0.0308 (0.0019) \\
BayesOpt & $< 1 \cdot 10^{-5}$ ($< 1 \cdot 10^{-5}$) & 0.3049 (0.0563) \\
\hline
\end{tabular}
\end{table}

C. Tuning of an Optical Beam-forming Network

In wireless communication systems, optical beam-forming networks (OBFNs) can be used to steer phased array antennas [23] in the desired direction. The principle is to combine the signals that arrive at different antenna elements in a phased array in such a way that positive interference of the signals occurs only in a specific direction. A device based on optical ring resonators [24] (ORRs) that can perform this signal processing technique in the optical domain was proposed in [25]. This OBFN can provide accurate control of the reception angle in broadband wireless receivers.

The performance of the OBFN under study depends on various factors, such as the reception angle of the incoming signal, the number of ORRs and their properties, network structure, actuator voltages, etc. Although physical models for this OBFN and the ORRs are available, these models can become very complex for large-scale OBFNs, and phenomena like temperature instabilities, vibrations and manufacturing errors cause model errors that can be difficult to analyze due to nonlinearities. The main challenge is to find the actuator voltages that maximize the signal-to-noise ratio (SNR).
The DONE algorithm was tested on this simulation to find the optimal heater voltages. The number of basis functions was set to 6000, the probability distribution of $p(x)$ was set to a normal distribution with variance 0.5, and the regularization parameter was set to 0.1. The exploration parameters $\sigma_\epsilon$ and $\sigma_\zeta$ were set to 0.01. The total number of measurements was set to 3000.

Just like in the previous application, the DONE algorithm is compared to a Bayesian optimization library called BayesOpt [13]. The same simulation was used in both algorithms, and BayesOpt also had 3000 function evaluations available. The other parameters for BayesOpt were set to their default values, except for the noise parameter which was set to 0.1 after calculating the influence of the measurement noise on the objective function. Also, in-between hyper-parameter optimization was turned off after noticing it did not influence the results while being very time-consuming.

Results for both algorithms on this application are shown in Fig. 4. The found optimum at each iteration is shown for the two algorithms. For DONE, the mean of 10 runs is shown, while for BayesOpt only one run is shown. The dotted line represents an offline approach: it is the average of 10 runs of a similar procedure as in [28], where a RFE with the same hyperparameters as in DONE was fitted to 3000 random measurements and then optimized. The figure clearly shows the advantage of the online approach: because measurements are only taken in regions where the objective function is low, the RFE model can become very accurate in this region. The figure also shows that DONE outperforms BayesOpt for this application in terms of accuracy. However, the total computation time shows this more clearly: one run of the DONE algorithm took about 2 minutes, while one run of BayesOpt took 5800 minutes. The computation time for this example and the other two examples is shown in Table II.

The big difference in computation time for the OBFN application can be explained by looking at the total number of measurements $N$. Even though the input dimension is high compared to the other problems, $N$ is the main parameter that causes Bayesian optimization algorithms to slow down for a large number of measurements. This is because the models used in Bayesian optimization depend on the kernel matrix of all samples, which will increase in size each iteration. The low-rank properties of RFEs give fast update rules for the DONE algorithm that are independent of the number of measurements.

### Table II

| Problem | Method | Input dim. | $N$ | $D$ | Time (s) |
|---------|--------|------------|-----|-----|----------|
| Camelback | DONE | 2 | 100 | 50 | 0.0322 |
| | BayesOpt | 2 | 100 | - | 0.3049 |
| OCT | DONE | 3 | 100 | 1000 | 0.106 |
| | BayesOpt | 3 | 100 | - | 1.019 |
| OBFN | DONE | 24 | 3000 | 6000 | 137.3 |
| | BayesOpt | 24 | 3000 | - | 3.48 $\cdot 10^5$ |

In [28], first measurements were obtained at random locations and then a RFE was fit and optimized. Now we implemented the DONE algorithm on this problem, which means that samples were chosen adaptively, leading to faster convergence.

An OBFN simulation based on the same physical models as in [28] will be used in this section, with the following differences: 1) the implementation is done in C++; 2) ORR properties are equal for each ORR; 3) heater voltages with offset and crosstalk [27, Appendix B] have been implemented; 4) a small region outside the bandwidth of interest has a desired group delay of 0; 5) an $8 \times 1$ OBFN with 12 ORRs is considered; 6) the standard deviation of the measurement noise was set to $7.5 \cdot 10^{-3}$. The input of the simulation is the normalized heater voltage for each ORR, and the output is the corresponding mean square error of the difference between OBFN path group delays and desired delays. The simulation contains 24 heaters, making the problem 24-dimensional.
Fig. 4. The mean square error of DONE and BayesOpt applied to the OBFN application, plotted versus the number of iterations. For DONE, the values are averaged over 10 runs. For BayesOpt only 1 run is shown. The dotted line is the result of fitting a RFE using 3000 random measurements and optimizing that RFE, averaged over 10 runs.

VI. CONCLUSIONS

We have analyzed an online optimization algorithm called DONE that is used to find the minimum of a function using measurements that are costly and corrupted by noise. This was done by training a surrogate model, namely a random Fourier expansion, updating it when new measurements are available and minimizing this surrogate with standard derivative-based methods. This allows to measure only in regions of interest, reducing the overall number of measurements required.

The algorithm is comparable to Bayesian optimization algorithms, but the computational complexity of the operations in the DONE algorithm does not depend on the number of data samples. Therefore, the computational complexity is equal for each iteration in the algorithm. An efficient implementation of the model update step is done using the inverse QR algorithm, while the optimization of the surrogate model is implemented with an L-BFGS method using the available derivative information of the surrogate.

We applied the DONE algorithm to a benchmark problem and to two optics applications: optical coherence tomography and optical beam-forming network tuning. We compared the algorithm to BayesOpt, a Bayesian optimization library. The DONE algorithm gave accurate results on these applications while being faster than the Bayesian optimization algorithm if the number of measurements is high, due to the low-rank approximation of the kernel. Also, due to the theory presented in this paper we could derive rules of thumb for tuning the hyperparameters of DONE.

APPENDIX A

PROOF OF CONVERGENCE OF THE LEAST SQUARES SOLUTION

In this section, we show that using the least squares solution in the RFE gives a function that approximates the true unknown function $f$. To prove this, we make use of the following theorems from literature [45, Thm. 2] and [46, Key Thm.]. We are ready to prove Theorem 4.

Proof of Theorem 4. Let $C_M = \{ c \in \mathbb{R}^D : ||c||_2 \leq M \}$ with $M \geq B$ a finite constant and

$$B = \sqrt{\frac{\sum_{k=1}^{D} \left( \frac{\hat{c}(\omega_k, b_k)}{(2\pi)^d D p_\omega(\omega_k)p_b(b_k)} \right)^2}{2}}.$$  

We need $M \geq B$ to make use of the results in [36]. Note that $C_M$ is a compact subset of $\mathbb{R}^D$. Also note that $c_N$ is the solution of the constrained (but not regularized) problem

$$\min_{c \in C_M} = \frac{1}{N} ||y_N - A_N c||_2^2.$$  

if $\lambda$ is chosen equal to the optimal Lagrange multiplier of the constraint $||c||_2 \leq M$. Since $M$ is allowed to have any value greater than or equal to $B$, $\lambda$ has to be chosen smaller than or equal to the optimal Lagrange multiplier of the constraint $||c||_2 \leq B$. In other words, the upper bound $\Lambda$ on $\lambda$ satisfies

$$\left\| \left( \frac{1}{N} A_N^T A_N + \Lambda I_{D \times D} \right) \right\|_2^{-1} \left\| \frac{1}{N} A_N^T y_N \right\|_2^2 = B^2.$$  

Recall from Section II-B that the vector $y_N$ depends on the function evaluations and on measurement noise $\eta$ that is assumed to be zero-mean and of finite variance $\sigma^2_n$. We first consider the noiseless case, i.e. $y_n = f(x_n)$. Let

$$E(x, c) = f(x) - \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k).$$  

Using the Cauchy-Schwarz inequality we have the following bound for all $x \in \mathbb{R}^d$, $c \in C_M$:

$$E(x, c)^2 = f(x)^2 + \sum_{k=1}^{D} c_k^2 \cos(\omega_k^T x + b_k) \leq f(x)^2 + \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k)^2 \leq f(x)^2 + \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k)^2 \leq f(x)^2 + M^2 + 2f(x)M \leq (||f||_\infty + M)^2.$$  

(48)
Note that $E(x, c)$ is continuous in $c$ and measurable in $x$. Using Theorem [45, Thm. 2] we get the following result with probability 1:

\[
\lim_{N \to \infty} \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} E(x_n, c)^2 - \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dx \right| = 0. \tag{49}
\]

Since almost sure convergence implies convergence in probability, we also have:

\[
\lim_{N \to \infty} P \left( \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} E(x_n, c)^2 - \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dx \right| > \epsilon \right) = 0 \quad \forall \epsilon > 0. \tag{50}
\]

For the case with noise, let

\[
\tilde{E}(x, \eta, c)^2 = \left( f(x) + \eta - \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k) \right)^2 = E(x, c)^2 + 2\eta E(x, c) + \eta^2. \tag{51}
\]

Using the properties of the noise $\eta$ with probability distribution $p_\eta$, this gives

\[
\int \left( y - \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k) \right)^2 dP(x, y) = \int \left( f(x) + \eta - \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k) \right)^2 p_x(x)p_\eta(\eta) dxd\eta
\]

\[
= \int_{\mathbb{R}^d} \tilde{E}(x, \eta, c)^2 p_x(x)p_\eta(\eta) dxd\eta
\]

\[
= \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dP(x, y) + 2 \int_{\mathbb{R}^d} E(x, c) \eta p_\eta(\eta) dP(x) + \int_{\mathbb{R}^d} \eta^2 p_\eta(\eta) dP(x)
\]

\[
= \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dxd\eta + \int_{\mathbb{R}^d} E(x, c)E[\eta] p_x(x) dxd\eta + E[\eta^2]
\]

\[
= \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dxd\eta + \sigma_\eta^2. \tag{52}
\]

Let $\epsilon_0, \epsilon_1, \epsilon_2, \epsilon_3 > 0$, with $\epsilon_1 + \epsilon_2 + \epsilon_3 = \epsilon_0$. Then we have, following a similar proof as in [47]:

\[
P \left( \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} \tilde{E}(x_n, \eta_n, c)^2 - \int_{\mathbb{R}^d} \tilde{E}(x, \eta, c)^2 p_x(x)p_\eta(\eta) dxd\eta \right| > \epsilon_0 \right)
\]

\[
= P \left( \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} E(x_n, c)^2 - \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dx \right| > \epsilon_1 \right) + \frac{1}{N} \sum_{n=1}^{N} \eta_n E(x_n, c) \right| + \frac{1}{N} \sum_{n=1}^{N} \eta_n^2 - \sigma_\eta^2 > \epsilon_0 \right) \geq P \left( \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} E(x_n, c)^2 - \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dx \right| > \epsilon_1 \right)
\]

\[
\leq P \left( \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} E(x_n, c)^2 - \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dx \right| > \epsilon_1 \right) + \frac{1}{N} \sum_{n=1}^{N} \eta_n E(x_n, c) \right| + \frac{1}{N} \sum_{n=1}^{N} \eta_n^2 - \sigma_\eta^2 \right) \geq P \left( \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} E(x_n, c)^2 - \int_{\mathbb{R}^d} E(x, c)^2 p_x(x) dx \right| > \epsilon_1 \right)
\]

Of these last three probabilities, the first one is proven to converge to zero in (50), while the last one converges to zero by the weak law of large numbers. For the second probability, we can make use of Theorem [45, Thm. 2] again, noting that $\eta_n E(x_n, c)$ is continuous in $c$. For this, we need to use (48) to get

\[
|\eta E(x, c)| \leq \eta (\|f\|_\infty + M) \quad \forall x, \eta, c. \tag{53}
\]

Again, since uniform convergence implies convergence in probability, and since $E[\eta E(x_n, c)] = E[\eta] E(x_n, c) = 0$, using Theorem [45, Thm. 2] gives the desired convergence in probability

\[
\lim_{N \to \infty} P \left( \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} \eta_n E(x_n, c) \right| > \epsilon_2 \right) = 0 \quad \forall \epsilon_2. \tag{54}
\]
Together with the other two convergences and (53) we get:

$$\lim_{N \to \infty} P \left( \sup_{c \in C_M} \left| \frac{1}{N} \sum_{n=1}^{N} \tilde{E}(x_n, \eta_n, c)^2 \right. \right.$$

$$\left. - \int_{\mathbb{R}^d} \int_{\mathbb{R}} \tilde{E}(x, \eta, c)^2 p_x(x)p_\eta(\eta)d\eta dx > \epsilon \right) = 0.$$  

(55)

To make use of Theorem [46, Key Thm.], we need the following bound, which follows from (48) and (52):

$$0 \leq \int_{\mathbb{R}^d} \int_{\mathbb{R}} \tilde{E}(x, \eta, c)^2 p_x(x)p_\eta(\eta)d\eta dx \leq (\|f\|_\infty + M)^2 + \sigma_\eta^2.$$  

(56)

Now we can use Theorem [46, Key Thm.] to get: for any \(\epsilon_1 > 0, \delta_1 > 0\) there exists an \(N_0\) such that for all \(N > N_0\) we have

$$\int_{\mathbb{R}^d} \int_{\mathbb{R}} \tilde{E}(x, \eta, c_N)^2 p_x(x)p_\eta(\eta)d\eta dx \leq \epsilon_1$$  

(57)\

with probability at least \(1 - \delta_1\). Here, \(c^0 \in C_M\) minimizes

$$\int_{\mathbb{R}^d} \int_{\mathbb{R}} \tilde{E}(x, \eta, c_0)^2 p_x(x)p_\eta(\eta)d\eta dx.$$

According to [36, Thm 3.1], for any \(\delta_2 > 0\), with probability at least \(1 - \delta_2\) there exists a \(c \in C_M\) with the following bound:

$$\int_{\mathbb{R}^d} \left( f(x) - \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k) \right)^2 p_x(x)dx < \gamma(\delta_2) \frac{\sqrt{D}}{\gamma(\epsilon_1)}.$$

$$\gamma(\delta_2) = \sup_{\omega, b} \left( \frac{1}{(2\pi)^d} \frac{\hat{c}(\omega, b)}{p_\omega(\omega)p_0(b)} \right) \left( 1 + \sqrt{2 \log \frac{1}{\delta_2}} \right).$$  

(58)

For this particular \(c\), using (52) we also have

$$\int_{\mathbb{R}^d} \int_{\mathbb{R}} \tilde{E}(x, \eta, c)^2 p_x(x)p_\eta(\eta)d\eta dx < \gamma(\delta_2) \frac{\sqrt{D}}{\gamma(\epsilon_1)} + \sigma_\eta^2.$$  

(59)

Since \(c^0 \in C_M\) minimizes

$$\int_{\mathbb{R}^d} \int_{\mathbb{R}} \tilde{E}(x, \eta, c_0)^2 p_x(x)p_\eta(\eta)d\eta dx$$

we have:

$$\int_{\mathbb{R}^d} \int_{\mathbb{R}} \tilde{E}(x, \eta, c_0)^2 p_x(x)p_\eta(\eta)d\eta dx < \gamma(\delta_2) \frac{\sqrt{D}}{\gamma(\epsilon_1)} + \sigma_\eta^2$$  

(60)

with probability at least \(1 - \delta_2\).

Note that the probability \(1 - \delta_2\) refers to the randomly drawn \((\omega, b)\), while the probability \(1 - \delta_1\) referred to the randomly drawn samples \((x, y)\). Since these are independent of each other, we can multiply the two probabilities and use the triangle inequality on (57) and (60) to get the following result:

For any \(\epsilon_1 > 0, \delta_1 > 0, \delta_2 > 0\) there exists an \(N_0\) such that for all \(N > N_0\) we have

$$\int_{\mathbb{R}^d} \int_{\mathbb{R}} \tilde{E}(x, \eta, c_N)^2 p_x(x)p_\eta(\eta)d\eta dx < \epsilon_1 + \frac{\gamma(\delta_2)^2}{D} + \sigma_\eta^2.$$  

(61)

*In the same reference, a similar bound exists for the infinity norm instead of the 2-norm (Theorem 3.2). The weights found in the proof of that result satisfy \(c \in C_M\). Here we also made use of the result from Theorem 1 (this paper).

with probability at least \((1 - \delta_1)(1 - \delta_2)\).

Using (52) gives the following result:

For any \(\epsilon_1 > 0, \delta_1 > 0, \delta_2 > 0\) there exists an \(N_0\) such that for all \(N > N_0\) we have

$$\int_{\mathbb{R}^d} E(x, c_N)^2 p_x(x)dx < \epsilon_1 + \frac{\gamma(\delta_2)^2}{D}.$$  

(62)

with probability at least \((1 - \delta_1)(1 - \delta_2)\).

Choosing \(\Lambda\) as in the start of this proof and choosing \(D_0, \epsilon_1, \delta_1\) and \(\delta_2\) such that \(D_0 > \gamma(\delta_2)^2/(\epsilon - \epsilon_1)\) and \((1 - \delta_1)(1 - \delta_2) = \delta\) concludes the proof.

\(\square\)

**APPENDIX B
MINIMUM-VARIANCE PROPERTIES**

The following theorem presents a minimum-variance distribution for \(\omega\) for a RFE.

**Theorem 7.** Given \(x\), the probability distribution \(p^*_\omega\) that minimizes the variance of the unbiased estimator \(g(x) = \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k)\) as defined in Definition 1, with \(c_k\) as defined in Theorem 3, is equal to

$$p^*_\omega(\omega) = \frac{\lfloor f(\omega) \lfloor \cos(2\int \hat{f}(\omega) + 2\omega^T x) \right) + 2}{\int_{\mathbb{R}^d} f(\omega) \right) \cos(2\int \hat{f}(\omega) + 2\omega^T x) + 2d\omega}.$$  

(63)

For this choice of \(p_\omega\), the variance is equal to

$$\frac{1}{2D(2\pi)^{2d}} \left( \int_{\mathbb{R}^d} |f(\omega)| \sqrt{\cos(2\int \hat{f}(\omega) + 2\omega^T x) + 2d\omega} \right)^2 - f(x)^2.$$  

(64)

**Proof.** The proof is similar to the proof of [42, Thm. 4.3.1]. Let \(q_\omega\) be any probability distribution of \(\omega\) that satisfies \(q_\omega(\omega) > 0\) if \(|f(\omega)| > 0\). Let \(\text{Var}_{q_\omega,p_\omega}\) be the variance of \(g\) under the assumption that \(p_\omega = q_\omega, p_n = \text{Uniform}(0,2\pi)\), and \(c_k = \frac{1}{D(2\pi)^d} \int_{\mathbb{R}^d} f(\omega) \cos(2\int \hat{f}(\omega) + b_k).\) According to Theorem 3, this choice for \(c_k\) makes sure that \(g\) is an unbiased estimator. Now the variance of \(g\) can be computed as:

$$\text{Var}_{q_\omega,p_\omega}[g(x)]$$

$$= \text{Var}_{q_\omega,p_\omega} \left[ \frac{1}{D} \sum_{k=1}^{D} c_k \cos(\omega_k^T x + b_k) \right]$$

$$= D \left( \text{Var}_{q_\omega,p_\omega} \left[ c_1 \cos(\omega_1^T x + b_1) \right] \right)$$

$$= \frac{D}{2\pi} \int_{\mathbb{R}^d} \int_{\mathbb{R}^d} \left( \frac{2}{D(2\pi)^d} q_\omega(\omega) \cos(2\int \hat{f}(\omega) - b) \right)^2 \cos(\omega^T x + b)^2 q_\omega(\omega) d\omega d\omega - f(x)^2.$$  

(65)
For the stated choice of $p^*_\omega$, using
\begin{align*}
\int_0^{2\pi} \cos(\omega T x + b)^2 db &= \frac{2\pi}{4} + \frac{1}{8} \int_0^{2\pi} \cos(\omega T x + 2b) db \\
&= \frac{\pi}{4} \left[ \cos(\omega T x + 2\pi) \right] + f(x)^2
\end{align*}
we get:
\begin{align*}
\text{Var}_{p^*_\omega, p_\omega}[g(x)] + f(x)^2 &= \frac{D}{2\pi} \int_{\mathbb{R}^d} \int_0^{2\pi} \left( \frac{2}{D(2\pi)^d} \right)^2 \left( \frac{1}{p^*_\omega(\omega)} \right) \frac{1}{p^*_\omega(\omega)} db d\omega \\
&= \frac{D}{2\pi} \int_{\mathbb{R}^d} \int_0^{2\pi} \cos(\omega T x + b)^2 p^*_\omega(\omega) db d\omega \\
&= \frac{D}{2\pi} \int_{\mathbb{R}^d} \int_0^{2\pi} \cos(\omega T x + b)^2 p^*_\omega(\omega) db d\omega \\
&= \frac{D}{2\pi} \left( \frac{2}{D(2\pi)^d} \right)^2 \left( \frac{1}{\sqrt{\pi}} \right) \left( \frac{\pi}{4} \right) \left[ \cos(\omega T x + 2\pi) \right] + f(x)^2
\end{align*}
(66)

This shows that the chosen probability distribution $p^*_\omega$ gives the minimum variance.

The following theorem compares the variances in real and complex RFEs for different probability distributions.

**Theorem 8.** Let $\tilde{p}_\omega$, $p^*_\omega$, $\tilde{g}$ and $g$ be as in Theorems 5 and 7. Then
\begin{align*}
\frac{1}{\sqrt{3}} \text{Var}_{p^*_\omega, \tilde{p}_\omega}[g(x)] &\leq \frac{1}{\sqrt{3}} \text{Var}_{p^*_\omega, p_\omega}[g(x)] \\
\frac{1}{\sqrt{3}} \text{Var}_{p^*_\omega, \tilde{p}_\omega}[g(x)] &\leq \frac{1}{\sqrt{3}} \text{Var}_{\tilde{p}_\omega, \tilde{p}_\omega}[g(x)]
\end{align*}
(70)

**Proof.** From
\begin{align*}
1 \leq \sqrt{\cos(2\omega T x + 2\pi) + 2} \leq \sqrt{3},
\end{align*}
(72)
and from (63) and (36) it follows that
\begin{align*}
\frac{1}{\sqrt{3}} \text{Var}_{p^*_\omega, p_\omega}[g(x)] &\leq \frac{1}{\sqrt{3}} \text{Var}_{\tilde{p}_\omega, \tilde{p}_\omega}[g(x)]
\end{align*}
(73)
Combining the above with (67) yields:
\begin{align*}
\frac{1}{\sqrt{3}} \text{Var}_{p^*_\omega, \tilde{p}_\omega}[g(x)] + f(x)^2 &= \frac{1}{\sqrt{3}} \text{Var}_{p^*_\omega, \tilde{p}_\omega}[g(x)] + f(x)^2 \\
&\leq \frac{1}{\sqrt{3}} \text{Var}_{\tilde{p}_\omega, \tilde{p}_\omega}[g(x)] + f(x)^2
\end{align*}
(74)
Combining (72) with (37) yields:
\begin{align*}
\frac{1}{\sqrt{3}} \text{Var}_{p^*_\omega, \tilde{p}_\omega}[g(x)] + f(x)^2 &= \frac{1}{\sqrt{3}} \text{Var}_{p^*_\omega, \tilde{p}_\omega}[g(x)] + f(x)^2 \\
&\leq \frac{1}{\sqrt{3}} \text{Var}_{\tilde{p}_\omega, \tilde{p}_\omega}[g(x)] + f(x)^2
\end{align*}
(75)
ACKNOWLEDGMENT

This research is supported by the Dutch Technology Foundation STW (project 13336), which is part of the Netherlands Organisation for Scientific Research (NWO), and which is partly funded by the Ministry of Economic Affairs, and by the Netherlands Enterprise Agency (RVO), Research Program for Innovation in Photonic Devices (IPD12020).

REFERENCES

[1] A. R. Conn, K. Scheinberg, and L. N. Vicente, Introduction to derivative-free optimization. SIAM, 2009, vol. 8.
[2] L. M. Rios and N. V. Sahinidis, “Derivative-free optimization: a review of algorithms and comparison of software implementations,” Journal of Global Optimization, vol. 56, no. 3, pp. 1247–1293, 2013.
[3] J. A. Nelder and R. Mead, “A simplex method for function minimization,” The computer journal, vol. 7, no. 4, pp. 308–313, 1965.
[4] M. J. Powell, “The newuoa software for unconstrained optimization without derivatives,” in Large-scale nonlinear optimization. Springer, 2007, pp. 257–297.
[5] ——, “The bohbya algorithm for bound constrained optimization without derivatives,” 2009.
[6] D. R. Jones, C. D. Perttunen, and B. E. Stuckman, “Lipschitzian optimization without the lipschitz constant,” Journal of Optimization Theory and Applications, vol. 79, no. 1, pp. 157–181, 1993.
[7] P. Gilmore and C. T. Kelley, “An implicit filtering algorithm for optimization of functions with many local minima,” SIAM Journal on Optimization, vol. 5, no. 2, pp. 269–285, 1995.
[8] A. L. Custódio and L. N. Vicente, “Using sampling and simplex derivatives in pattern search methods,” SIAM Journal on Optimization, vol. 18, no. 2, pp. 537–555, 2007.
[9] M. Pogu and J. S. De Cursi, “Global optimization by random perturbation,” in Proceedings of the Conference on Optimization 2015. CAO2015Garmisch-Partenkirchen, Germany, 69 October 2015. [Online]. Available: http://www.scientificdirect.com/science/article/pii/S2405896315023368
[10] T. Hofmann, B. Schölkopf, and A. J. Smola, “Kernel methods in machine learning,” The annuals of statistics, pp. 1171–1220, 2008.
[11] C. J. Burges et al., “Simplified support vector decision rules,” in ICML, vol. 96. Citeseer, 1996, pp. 71–77.
[12] D. Khiob, “A statistical approach to some basic mine valuation problems on the witwatersrand,” Journal of Chemical, Metallurgical, and Mining Society of South Africa, 1951.
[13] J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, “Algorithms for Innovation in Photonic Devices (IPD12020).”
[14] J. Snoek, H. Larochelle, and R. P. Adams, “Practical bayesian optimization of expensive black-box functions,” in Advances in Neural Information Processing Systems 25: Proceedings of the 2012 Conference, vol. 1. MIT Press, 2012, p. 335.
[15] J. Quinonero-Candela and C. E. Rasmussen, “A unifying view of sparse approximate gaussian process regression,” The Journal of Machine Learning Research, vol. 6, pp. 1939–1959, 2005.
[16] M. Pogu and J. S. De Cursi, “Global optimization by random perturbation algorithms for waveform aberration correction in adaptive optics: optical coherence tomography,” Optics letters, vol. 38, no. 22, pp. 4801–4804, 2013.
[17] R. Y. Rubinstein and D. P. Kroese, “The cross-entropy method: a review,” Annals of Operations Research, vol. 100, no. 1, pp. 25–58, 2000.
[18] M. R. Nasiri-Avanaki, S. Hojjatoleslami, H. Paun, S. Tuohy, A. Meadway, G. Dobre, and A. Podeleau, “Optical coherence tomography system optimization using simulated annealing algorithm,” Proceedings of Mathematical Methods and Applied Computing,(WSEAS, 2009), pp. 669–674, 2009.
[19] S. Bonora and R. Zawadzki, “Wavefront sensorless modal deformable mirror correction in adaptive optics: optical coherence tomography,” Optics letters, vol. 38, no. 22, pp. 4801–4804, 2013.
[20] M.-R. Nasiri-Avanaki, S. Hojjatoleslami, H. Paun, S. Tuohy, A. Meadway, G. Dobre, and A. Podeleau, “Optical coherence tomography system optimization using simulated annealing algorithm,” Proceedings of Mathematical Methods and Applied Computing,(WSEAS, 2009), pp. 669–674, 2009.
[21] C. Roelfzen, L. Zhuang, R. Heideman, A. Borremans, and V. W. Etten, “Ring-resonator-based tunable optical delay line in InP/InP waveguide technology,” 2005.
[22] A. Rahimi and B. Recht, “Random features for large-scale kernel machines,” in Advances in neural information processing systems, 2007, pp. 5722–5725.
[23] A. Rahimi and B. Recht, “Random features for large-scale kernel machines,” Advances in neural information processing systems, vol. 20, pp. 1177–1184.
[45] R. I. Jennrich, “Asymptotic properties of non-linear least squares estimators,” *The Annals of Mathematical Statistics*, pp. 633–643, 1969.

[46] V. N. Vapnik, “An overview of statistical learning theory,” *Neural Networks, IEEE Transactions on*, vol. 10, no. 5, pp. 988–999, 1999.

[47] A. Beitollahi and P. Azhdari, “Convergence in probability and almost surely convergence in probabilistic normed spaces,” *Mathematical Sciences*, vol. 6, no. 1, pp. 1–5, 2012.