A Method for Handling Uncertainty in Evolutionary Optimization with an Application to Feedback Control of Combustion

Nikolaus Hansen, Andre S.P. Niederberger, Lino Guzzella, Petros Koumoutsakos

▶ **To cite this version:**

Nikolaus Hansen, Andre S.P. Niederberger, Lino Guzzella, Petros Koumoutsakos. A Method for Handling Uncertainty in Evolutionary Optimization with an Application to Feedback Control of Combustion. IEEE Transactions on Evolutionary Computation, 2009. inria-00276216

HAL Id: inria-00276216
https://inria.hal.science/inria-00276216

Submitted on 21 Jun 2008

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

A Method for Handling Uncertainty in Evolutionary Optimization with an Application to Feedback Control of Combustion

Nikolaus Hansen, André S.P. Niederberger, Lino Guzzella, and Petros Koumoutsakos

***Abstract*— We present a novel method for handling uncertainty in evolutionary optimization. The method entails quantification and treatment of uncertainty and relies on the rank based selection operator of evolutionary algorithms. The proposed uncertainty handling is implemented in the context of the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) and verified on test functions. The present method is independent of the uncertainty distribution, prevents premature convergence of the evolution strategy and is well suited for online optimization as it requires only a small number of additional function evaluations. The algorithm is applied in an experimental set-up to the online optimization of feedback controllers of thermoacoustic instabilities of gas turbine combustors. In order to mitigate these instabilities, gain-delay or model-based $\mathcal{H}_\infty$ controllers sense the pressure and command secondary fuel injectors. The parameters of these controllers are usually specified via a trial and error procedure. We demonstrate that their online optimization with the proposed methodology enhances, in an automated fashion, the online performance of the controllers, even under highly unsteady operating conditions, and it also compensates for uncertainties in the model-building and design process.**

I. Introduction

Environmental considerations impose stringent emission regulations for modern gas turbines. These requirements dictate the development of lean premixed combustion systems operating with excess air to lower the combustion temperature and decrease the NO_x emission levels [37]. In turn, the operation of the combustor in the lean regime makes it prone to thermoacoustic instabilities that may cause mechanical damage, energy losses by heat transfer to walls and increased noise and pollutant emissions. Thermoacoustic instabilities arise due to a feedback loop between pressure fluctuations, flow velocity and heat release. Active control is a prevalent method to reduce thermoacoustic instabilities [38], [21]. In active control of gas turbine combustors a feedback controller receives input from pressure sensors and commands a secondary fuel injection. The adjustment of the controller parameters into a feasible working regime can be formulated as an optimization problem distinguished by two important factors: The stochastic nature of the combustion process introduces uncertainty in the computation of the objective function value while the unsteady operating conditions require the online tuning of the controller parameters.

A.S.P. Niederberger and L. Guzzella are with the Measurement and Control Laboratory

N. Hansen and P. Koumoutsakos are with the Institute of Computational Science

Evolutionary Algorithms (EAs) are intrinsically robust to uncertainties present in the evaluation of the objective function due to the implementation of a population [5], [11]. In order to improve their robustness to uncertainty two common methods are available. First, the implementation of larger population size most often increases the robustness to uncertainty [6], [32]. Second, multiple objective function evaluations can be conducted for each population member and the objective function is usually represented by the mean value. Both approaches however increase the number of function evaluations per generation typically by a factor between three and 100. Hence the large number of required function evaluations makes the methods prohibitively expensive for applications requiring an online optimization.

In this paper we propose an alternative approach to enhance the capabilities of EAs for online optimization under uncertainties. We develop a novel uncertainty handling algorithm and, motivated by the combustion problem, we demonstrate its effectiveness in the online optimization of a Gain-Delay and an $\mathcal{H}_\infty$ controller of an experimental combustor test-rig using the CMA evolution strategy. The uncertainty handling method distinguishes uncertainty measurement and uncertainty treatment. The uncertainty is measured by rank changes among members of a population. This quantification of uncertainty is well suited for any ranking-based search algorithm. It requires only a few additional function evaluations per generation, and does not rely on an underlying uncertainty distribution. The uncertainty measurement is combined with two treatments for high uncertainty levels to prevent the failure of the algorithm. The uncertainty treatments aim to ensure that the signal-to-noise ratio remains large enough to maintain the effectiveness of the evolutionary optimization algorithm.

The paper is organized as follows: In Section II the test rig, built at ETH Zurich, is presented. We cast the optimization of the controller parameters as an optimization problem under uncertainties and we discuss previous work. We address the problem of thermoacoustic instabilities for gas combustors and introduce their handling by active control strategies. Section III addresses evolutionary optimization under uncertainties. In Section IV the uncertainty handling method is introduced and combined with the CMA evolution strategy. Section V presents the verification of the algorithm on test functions. Section VI reports experiments on the test rig with the different controller structures for two operating conditions. The paper concludes with a Summary and Outlook in Section VII.

II. Active Control of Combustion Instabilities in an Experimental Test Rig

In the following we describe the ETHZ combustor test rig where online optimization of controller parameters is performed and review the common controller techniques.

A. Experimental set-up of the ETHZ combustor test rig

A schematic illustration of the test rig built at ETH Zurich is shown in Fig. 1. Preheated air premixed with natural gas flows through mixers and flow straighteners into an upstream plenum chamber duct. A downscaled, lab scale model for the ALSTOM environmental (EV) swirl burner stabilizes the flame in recirculation regions near the burner outlet plane, the combustion gases are guided through a downstream duct and they are subsequently discharged. A MOOG magnetostrictive fuel injector installed close to the flame is used as control actuator. The pressure signal is detected by water-cooled microphones distributed along the ducts. Microphone 2, placed 123mm downstream of the burner, is used to deliver the sensor signal for the controller.

The operating conditions of the combustor are characterized by the mass flow, the preheat temperature, and the ratio of the actual to the stoichiometric air/fuel ratio λ. For the present study, a mass flow of 36 g/s, a preheat temperature of 700 K, and λ values of 2.1 and 1.875 are considered. The resulting pressure spectra are shown in Figures 17 and 14. The case of $\lambda = 2.1$ exhibits a single large pressure peak at 220 Hz, whereas $\lambda = 1.875$ is characterized by one peak around 250 Hz and two smaller ones in the 330 Hz range.

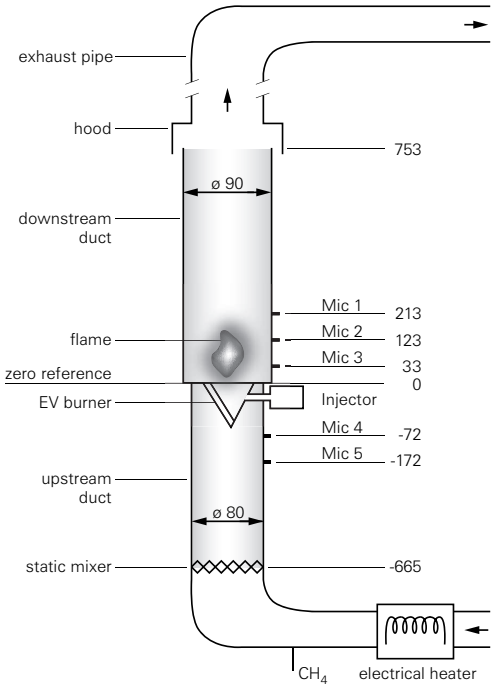

Fig. 1. An illustration of the ETH combustor. Preheated air premixed with methane enters the upstream duct, the flame is stabilized by the EV burner. All dimensions in mm.

B. Actuators

Loudspeakers, often used as actuators in a laboratory settings, are not feasible for industrial applications due to limited actuation power. In contrast, secondary fuel injection of about 10% of the total methane flow of 1 g/s yields roughly 6000 W, versus 30 W for a loudspeaker. The tradeoff between injection time delays and increased NO_x emissions due to diffusion flames has to be carefully negotiated and a suitable position for the injector must be found.

C. Controllers

The simplest controller is known as phase-shift or Gain-Delay, where the measured pressure signal is amplified and delayed by a certain amount and then fed to the actuator [46]. This simple strategy has found widespread use, but it often generates secondary peaks as the gain and phase are tuned to the dominant frequency and they are not optimal in other frequency bands. The model-based robust $\mathcal{H}_\infty$ controller design lets the engineer specify regions where the disturbance should be reduced, and the $\mathcal{H}_\infty$-optimization routine calculates the corresponding controller [56].

Gain-Delay control is convenient as there are only two parameters to adjust. This is often done by trial-and-error with satisfactory results if the spectrum of the instability features only one dominant peak. Model-based $\mathcal{H}_\infty$ controllers on the other hand offer larger design freedom and are generally associated with better performance. They involve however 10 to 20 parameters thus exacerbating their online optimization. In addition, thermal transients during start-up change the location and height of the pressure peaks, and the (steady-state) model of the process is not always accurate.

A combination of a model-based controller and an online optimization using EAs has been used to address these difficulties. More specifically, an $\mathcal{H}_\infty$ controller [44], [56] is shifted in the frequency domain while the gain and (optionally) an additional delay are adjusted, resulting in two (three) parameters to be optimized. Note that in a Gain-Delay controller only the gain and the delay are optimized by the algorithm. The cost function to be minimized is selected as the equivalent continuous level of the sound pressure

$$L_{eq} = 10 \log_{10} \frac{(p_s^2)_{av}}{p_{ref}^2} \tag{1}$$

where $(p_s^2)_{av}$ is the mean squared pressure and $p_{ref} = 20\mu$Pa the reference pressure.

The sound pressure level L_{eq} is acquired from a measurement of a few seconds for a given parameter setting. The measurements are subject to a considerable uncertainty and a tradeoff between uncertainty and speed in data acquisition can be identified. The accuracy of L_{eq} is improved with longer evaluation times. At the same time longer evaluation times decrease the number of feasible completed measurements in a given time span, and slow down the adaptation of the controller parameters. This problem will be resolved by an adaptive evaluation time for the acquisition of L_{eq}.

D. Adaptive Controllers

Unsteady combustion introduces pressure waves in the combustor. Their reflection from the boundaries modifies in turn the combustion process resulting in a potentially unstable feedback cycle. Rayleigh [50] first noted that if the heat release is in phase with the pressure fluctuations, the instability grows, while combustion is stable if they are out of phase. In a study of a swirl-stabilized premix burner [47], it was found that large coherent structures are associated with this instability. The interaction between flow instabilities and acoustic resonant modes excites unstable modes, leading to periodic combustion in large-scale structures. Fuel feedline dynamics are another mechanism causing equivalence ratio fluctuations [3].

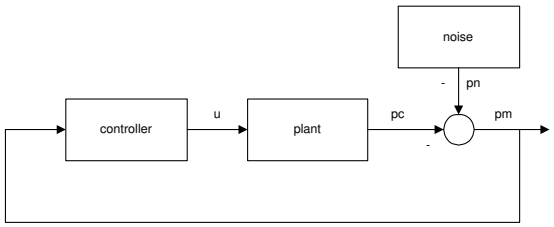

Fig. 2. A schematic diagram of the control set-up

The adaptive reduction of the pressure oscillations can be achieved by measuring the pressure with microphones, and employing a controller to command an actuating device, such as a loudspeaker or a fuel injector. Fig. 2 shows the control set-up used for this study. The uncertainty input block models the uncontrolled combustor generating the pressure signal p_n. The plant P is the block that relates the control signal input u to the pressure p_c generated by altered combustion in case of fuel injection. The sum of these two is the measured pressure signal p_m that needs to be minimized. This signal is used by the controller C to generate the control signal u for the fuel injector.

Controllers can be built based on a model of the controlled process. A so-called Self-Tuning Regulator (STR) [24], [22], [53], [52], [23] requires knowledge only of the total time delay between actuation and sensing. STRs have shown some robustness to changing operating conditions but for a combustor which is already stable, the STR does not offer any advantages over a model-based controller and may encounter numerical problems.

Adaptive controllers [8], [35] encounter problems in noisy environments and a number of open questions remain regarding algorithmic instabilities. A simple Rijke tube with very distinct pressure peaks and low uncertainty level is considered in [13]. Loudspeakers are used as actuators for a neural network controller, which requires an identification procedure beforehand. In [48] a multiobjective modified strength Pareto evolutionary algorithm has been used to optimize the fuel flow through different injection locations in an EV burner.

A range of lead/lag controllers [45] are optimized with a (1+1)-ES in [49], [55]. The influence of uncertainty and the problem with long evaluation times are identified, and a two-step evaluation procedure is proposed. The potential problems with noisy evaluations arising from elitism and the problem of premature convergence have been neglected and the method employs a pre-specified maximum number of iterations.

E. Evolutionary Algorithms for Control

An in-depth overview of evolutionary algorithms applied to controller optimization is given in [25]. One can distinguish between online and offline optimization. Online applications are rare and due to safety and time-constraints only very few online applications have been conducted in a real system [1], [43].

In order to evolve the controller either the controller parameters are directly optimized [19], [1], or the design parameters of control algorithms such as Linear Quadratic Gaussian (LQG) or $\mathcal{H}_\infty$ are manipulated [20], and the controller is calculated automatically. In order to improve the feasibility of the online application of evolutionary algorithms, tuning of an existing controller can be performed [39], [40]. Our method is based on this latter approach.

III. Optimization under Uncertainties

The identification of effective parameters for adaptive controllers can be formulated as an optimization problem, where a combustor performance related objective function, for example the time integral of the sound pressure in the combustor, is to be minimized. A general formulation of such a time dependent stochastic objective function L (also *l*oss or cost function) reads

$$L : \mathcal{S} \times \mathbb{R}_+ \to \mathbb{R}, \quad (\boldsymbol{x}, t) \mapsto f(\boldsymbol{x}, t) + N_f(\boldsymbol{x}, t) \ , \qquad (2)$$

where $\boldsymbol{x} \in \mathcal{S} \subset \mathbb{R}^n$ is a (solution) vector of controller parameters and t is time. The objective function is defined by a deterministic part f and a stochastic part $N_f \in \mathbb{R}$. The objective is to find an (approximate) minimizer of the "true" function value f. The distribution of N_f is unknown and depends on the function f, as well as on $\boldsymbol{x}$ and t. The time dependency is relevant, for example, in online control of a combustor as the operating condition may be modified manually or may change during the heating up of the rig. In general however the changes in time are often negligible when compared to the variations in N_f for each point in time. We assume that $\mathrm{E}\big[L(\boldsymbol{x}, t)\big] = f(\boldsymbol{x}, t)$, *i.e.* $\mathrm{E}\big[N_f(\boldsymbol{x}, t)\big] = 0$ for all $\boldsymbol{x} \in \mathbb{R}^n$ and all $t \geq 0$, without loss of generality. If the expectation value does not exist, we assume the median of $L(\boldsymbol{x}, t)$ equals to $f(\boldsymbol{x}, t)$, for all $\boldsymbol{x}, t$. This assumption makes the definitions of f and N_f consistent with the objective to find a minimizer of f. Furthermore, if, instead of the median, we postulate a larger quantile (for example the 95%-tile) of L equals to f, this would imply trying to find a more "robust" solution as the minimizer of f.

Equation (2) describes a generic uncertainty model. The equation includes uncertainties that may appear at any stage of obtaining the measurement L. Examples of such uncertainties include the adjustment of the variable vector $\boldsymbol{x}$ where the dependency between N_f and f becomes evident, sometimes called actuator noise [11].

From Equation (2) we can immediately imply that in a ranking-based algorithm uncertainties are problematic if and

only if, for two candidate solutions $\boldsymbol{x}_1$ and $\boldsymbol{x}_2$, the variation due to $N_f(\boldsymbol{x}_1)$ and $N_f(\boldsymbol{x}_2)$ exceeds the difference $|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)|$ such that their ordering reverses. If the uncertainties tend to exceed the difference $|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)|$ we cannot anymore conclude from two single measurements, $L(\boldsymbol{x}_1)$ and $L(\boldsymbol{x}_2)$, whether $f(\boldsymbol{x}_1) > f(\boldsymbol{x}_2)$ or $f(\boldsymbol{x}_1) < f(\boldsymbol{x}_2)$ holds with a small enough error probability (in a ranking-based search algorithm this is the only decision based on the L-values). In other words, referring to $|f(\boldsymbol{x}_1) - f(\boldsymbol{x}_2)|$ as signal and to the variations from N_f as noise, the uncertainty is problematic if and only if the signal-to-noise ratio is too small. This observation readily implies that there can only be two ways to cope with uncertainties for a ranking-based search algorithm.

1) Increasing the signal, or
2) reducing the uncertainty.

Efficient optimization techniques for problems with uncertainties must address successfully at least one of these two issues. A broad literature overview of uncertainties addressed in evolutionary optimization is given in [33].

The most common technique to approach uncertainties in the objective function value is resampling, that is the repeated evaluation of $L(\boldsymbol{x})$ for a given solution $\boldsymbol{x}$ [2], [15], [18], [58]. A statistics $\widehat{L}(\boldsymbol{x})$ of the repeated samples of $L(\boldsymbol{x})$ replaces the single measurement. Usually the mean value is taken as statistics $\widehat{L}$ and the question whether it is an appropriate statistics for uncertainty reduction is ignored. If the second moment $\mathrm{E}\big[N_f^2\big]$ exists, the variance of the mean statistics equals to the variance of L divided by the number of independent samples and hence the mean leads to a reduction of uncertainty. Taking the median as statistics reduces the uncertainty under much milder assumptions than the mean. If $\mathrm{E}\big[N_f^\alpha\big]$ exists for some $\alpha > 0$ then $\mathrm{E}\big[\widehat{L}^2\big]$ exists for any sufficiently large sample size (note that $\mathrm{E}\big[N_f^0\big] = 1$). If in addition, N_f has a continuous positive density in some neighborhood of the true median, then $\widehat{L}$ is asymptotically normally distributed with variance inversely proportional to the sample size [36].

The main drawback of repeated evaluations is the increased cost of the algorithm (in our case the evaluation of L is by far the most time consuming part of the optimization). Given the sphere function $f(\boldsymbol{x}) = \|\boldsymbol{x}\|^2$ and N_f normally distributed with standard deviation σ_ε an evolution strategy can reach a final distance to the optimum of $R^\infty \propto \sqrt{\sigma_\varepsilon}$ [9] [10]. Consequently, to reduce the final distance to the optimum R^∞ by a factor of $\alpha < 1$ the number of necessary L-samples grows with α^{-2}.

Usually the number of necessary samples varies in time and cannot be predicted in advance. In particular in the early stages of optimization the signal-to-noise ratio is expected to be large for two reasons. First, the distance between population members is large producing more likely a large difference in objective function values. Second, the difference between L-values of "bad" solutions is usually larger than for "good" solutions. In order to reduce the associated cost, adaptive reevaluation methods have been proposed [2], [15] [18]. The number of reevaluations is determined by the outcome of a statistical test, for example the t-test [18], [58]. The choice of the solutions to be reevaluated can depend on their ranking in the population [2], [58] or on the empirical variances of the measurements [18]. The number of reevaluations is limited by an upper bound to avoid divergence and to maintain adaptivity in online applications. Despite these efforts, methods that reduce the uncertainty based on reevaluations typically increase the number of function evaluations per generation by a factor between three and a 100.

A slightly different approach to reduce the uncertainty uses the already evaluated solutions. Instead of resampling L and taking statistics of the samples a surrogate function (or metamodel) is built from the already evaluated solutions [54], [14], [17]. In case of "benign" uncertainty distributions the surrogate smooths the noisy landscape. For the CMA evolution strategy a local quadratic surrogate model could speed-up the convergence on a noisy sphere function by a factor of two in small dimensions [34]. In general, the surrogate approach will be less effective with increasing search space dimension or when a large population spread is already realized by the underlying search algorithm.

A third approach addresses uncertainties in the objective function by using a large population, also referred to as implicit averaging [33]. The effect of a large population size in an evolution strategy (ES) is twofold. First, the population spread can be larger. For example, on the sphere function the optimal step-size of the $(\mu/\mu_I, \lambda)$-ES is proportional to the parent number μ, given intermediate multi-recombination and $\mu \propto \lambda \not\gg n$ [51, p.148]. Second, recombination smooths the effect of erroneous selection in search space. Consequently increasing only λ is inferior to resampling [9], [26], but increasing μ and λ is preferable to resampling [5]. A prerequisite for this advantage is that step-sizes are adapted properly, because the population spread is decisive. Otherwise increasing the population size can even be counterproductive [26].

Modifications of the selection scheme have been proposed to compensate for uncertainties. In a $(1+1)$-ES a non-zero threshold for accepting the offspring is advantageous [42]. In the $(\mu/\mu_I, \lambda)$-ES the optimal ratio between μ and λ is 0.5 [10] corresponding to a maximal step-size for a given λ. The stochastic tournament selection can be modified to make up for the stochastics introduced by the uncertain selection [16], while in evolution strategies the selection scheme is already deterministic.

Overall, the handling of uncertainties in the objective function has been mainly addressed by uncertainty reduction rather than signal improvement. In this paper we will use both approaches. First, a resampling approach is taken and adopted to the specific application to reduce the uncertainty. Second, and more importantly, the signal is improved explicitly by increasing the population spread. Both approaches are controlled by a uncertainty measurement and hence implemented in an adaptive way.

IV. An Uncertainty-Resistant Evolutionary Algorithm

In this section we describe an evolutionary algorithm that serves to minimize an objective function as defined in Equation

(2). The algorithm consists of two parts: A ranking-based evolutionary algorithm, the CMA-ES, and the uncertainty handling method. We first describe the CMA-ES and then introduce the proposed novel uncertainty handling technique.

A. The CMA Evolution Strategy

We employ the evolution strategy (ES) with Covariance Matrix Adaptation (CMA) [30], [31], [29], [28]. This choice is motivated by several reasons:

- CMA-ES is a non-elitist continuous domain evolutionary algorithm. Non-elitism avoids systematic fitness overvaluation on noisy objective functions [5], because even solutions with (erroneously) exceptionally good fitness values survive only one generation.
- The selection in CMA-ES is solely based on the ranking of solutions. This provides additional robustness in a noisy environment. Ranking-based selection is in particular invariant to strictly monotonic (order-preserving) transformations of the value L.
- The CMA-ES provides an effective adaptation of the search distribution to the landscape of the objective function.
- The CMA-ES can be reliably used with small population sizes allowing for a fast adaptation in an online application.

The CMA-ES adapts the covariance matrix of a normal search distribution to the given objective function topography. On convex-quadratic objective functions, nearly optimal covariance matrices are thus achieved. The adaptation procedure operates efficiently and independently of the given population size, which is small by default. Particularly on non-separable, badly scaled problems often a speed-up by several orders of magnitude can be achieved in terms of number of function evaluations to reach a given function value and in terms of CPU-time. The CMA-ES was evaluated on a variety of test functions [31], [29], [28], [7] and was successfully applied to a variety of real-world problems.[1]

The CMA-ES follows two fundamental **design principles** employed in the development of the algorithm. The first design principle is **invariance**. We distinguish between invariance to transformations $\mathbb{R} \to \mathbb{R}$ of the function value L, as considered in the beginning of this section, and invariance to transformations $\mathcal{S} \to \mathcal{S}$ of the solution vector $\boldsymbol{x}$ in (2). The CMA-ES reveals invariance to rigid (angle-preserving) transformations of the solution vector, like translation, rotation and reflection, given that the initial solution is transformed respectively. The CMA-ES reveals invariance to overall scaling of the search space, given that the initial scale σ is chosen accordingly. Finally, The CMA-ES reveals even invariance to any full rank linear transformation, given that the initial covariance matrix is chosen respectively. Invariance properties induce equivalence classes of objective functions and therefore allow for generalization of empirical results.

Second, the variation of object and strategy parameters is **unbiased** [12], [31]. Given random selection, that is an objective function $L(\boldsymbol{x}) = \texttt{rand}$ that is independent of $\boldsymbol{x}$, the first moment of the object parameters $\boldsymbol{x}$ is unbiased. The expectation of newly generated solutions is equal to the weighted mean of the previously selected solutions. The second moment is described by covariance matrix and step-size. The covariance matrix in the CMA-ES is unbiased, because under random selection the updated covariance matrix is equal to the previous covariance matrix in expectation. Analogously, the step-size σ is unbiased on the log scale. For the second moment, the population variance, a bias towards increase or decrease will entail the danger of divergence or premature convergence, respectively, whenever the selection pressure is low. Next we describe the algorithm in detail.

Given an initial mean value $\boldsymbol{m} \in \mathbb{R}^n$, the initial covariance matrix $\boldsymbol{C} = \mathbf{I}$ and the initial step-size $\sigma \in \mathbb{R}_+$, the λ candidate solutions $\boldsymbol{x}_k$ of one generation step obey

$$\boldsymbol{x}_k = \boldsymbol{m} + \sigma \boldsymbol{y}_k, \quad k = 1, \ldots, \lambda, \tag{3}$$

where $\boldsymbol{y}_k \sim \mathcal{N}(\mathbf{0}, \boldsymbol{C})$ denotes a realization of a normally distributed random vector with zero mean and covariance matrix $\boldsymbol{C}$. Equation (3) implements mutation in the EA by adding a random vector. The solutions $\boldsymbol{x}_k$ are evaluated on L and ranked such that $\boldsymbol{x}_{i:\lambda}$ becomes the i-th best solution vector and $\boldsymbol{y}_{i:\lambda}$ the corresponding random vector realization.

In the remainder we describe the updates of $\boldsymbol{m}$, σ, and $\boldsymbol{C}$ for the next generation step. For $\mu < \lambda$ let

$$\langle \boldsymbol{y} \rangle = \sum_{i=1}^{\mu} w_i \boldsymbol{y}_{i:\lambda}, \quad w_1 \geq \cdots \geq w_\mu > 0, \quad \sum_{i=1}^{\mu} w_i = 1 \tag{4}$$

be the weighted mean of the μ best ranked $\boldsymbol{y}_k$ vectors. The recombination weights sum to one. The so-called variance effective selection mass

$$\mu_{\text{eff}} = \frac{1}{\sum_{i=1}^{\mu} w_i^2} \geq 1 \tag{5}$$

will be used in the following. Given μ_{eff}, the particular setting of the recombination weights is, in our experience, secondary. From the definitions follows $1 \leq \mu_{\text{eff}} \leq \mu$ and $\mu_{\text{eff}} = \mu$ for equal recombination weights. The role of μ_{eff} is analogous to the role of the parent number μ with equal recombination weights and usually $\mu_{\text{eff}} \approx \lambda/4$ is appropriate. Weighted recombination is discussed in more detail in [4].

The mean of the new distribution becomes

$$\boldsymbol{m} \leftarrow \boldsymbol{m} + \sigma \langle \boldsymbol{y} \rangle = \sum_{i=1}^{\mu} w_i \boldsymbol{x}_{i:\lambda}. \tag{6}$$

Equation (6) determines the center of the next population. The equation implements selection by using $\mu < \lambda$. Using different recombination weights must also be interpreted as selection mechanism. The equation implements recombination by taking a (weighted) mean of parental solutions.

For step-size control the "conjugate" evolution path $\boldsymbol{p}_\sigma \in \mathbb{R}^n$ is introduced. The evolution path cumulates an exponentially fading pathway of the population mean in the generation sequence. Assuming that the optimal step-size leads to conjugate steps, the length of the conjugate evolution path can be used as adaptation criterion for σ. If the evolution path is long,

[1] See http://www.inf.ethz.ch/personal/hansenn/cec2005.html and http://www.inf.ethz.ch/personal/hansenn/cmaapplications.pdf

σ must be increased, whereas if the evolution path is short, σ must be decreased. Initialized with $\boldsymbol{p}_\sigma = \mathbf{0}$ the update of $\boldsymbol{p}_\sigma$ (so-called cumulation) and σ reads :

$$\boldsymbol{p}_\sigma \leftarrow (1-c_\sigma)\,\boldsymbol{p}_\sigma + \sqrt{c_\sigma(2-c_\sigma)\mu_{\mathrm{eff}}}\,\boldsymbol{C}^{-\frac{1}{2}}\langle \boldsymbol{y}\rangle \quad (7)$$

$$\sigma \leftarrow \sigma \times \exp\left(\frac{c_\sigma}{d_\sigma}\left(\frac{\|\boldsymbol{p}_\sigma\|}{\widehat{\chi}_n} - 1\right)\right) \quad (8)$$

where $1/c_\sigma > 1$ determines the backward time horizon of the evolution path $\boldsymbol{p}_\sigma$, damping $d_\sigma \approx 1$ controls the change magnitude of σ, and $\widehat{\chi}_n$ is the expected length of a random variable distributed according to $\mathcal{N}(\mathbf{0}, \mathbf{I})$. The evolution path is appropriately normalized. We have $\boldsymbol{C}^{-\frac{1}{2}} \stackrel{\mathrm{def}}{=} \boldsymbol{B}\boldsymbol{D}^{-1}\boldsymbol{B}^{\mathrm{T}}$, where $\boldsymbol{C} = \boldsymbol{B}\boldsymbol{D}^2\boldsymbol{B}^{\mathrm{T}}$ is an eigendecomposition of the symmetric, positive definite covariance matrix $\boldsymbol{C}$.[2] The transformation $\boldsymbol{C}^{-\frac{1}{2}}$ rescales $\langle \boldsymbol{y}\rangle$ into an isotropic reference system. Given $\boldsymbol{y}_{i:\lambda}$ distributed according to $\mathcal{N}(\mathbf{0}, \boldsymbol{C})$, as under random selection, we can derive that $\sqrt{\mu_{\mathrm{eff}}}\boldsymbol{C}^{-\frac{1}{2}}\langle \boldsymbol{y}\rangle$ is distributed according to $\mathcal{N}(\mathbf{0}, \mathbf{I})$. Therefore, if $\boldsymbol{p}_\sigma \sim \mathcal{N}(\mathbf{0}, \mathbf{I})$ holds before applying (7), the same holds after applying (7). The transformations make the expected length of $\boldsymbol{p}_\sigma$ independent of its orientation and allow the comparison of the length of $\boldsymbol{p}_\sigma$ with its expected length $\widehat{\chi}_n$ in (8). Step-size σ is increased if and only if $\|\boldsymbol{p}_\sigma\| > \widehat{\chi}_n$, and decreased if and only if $\|\boldsymbol{p}_\sigma\| < \widehat{\chi}_n$. In practice we use the approximation $\widehat{\chi}_n = \sqrt{2}\,\Gamma(\frac{n+1}{2})/\Gamma(\frac{n}{2}) \approx \sqrt{n}\,(1 - \frac{1}{4n} + \frac{1}{21n^2})$.

Similar to (7) an evolution path $\boldsymbol{p}_{\boldsymbol{c}}$ is constructed to update the covariance matrix. The covariance matrix admits a rank-one and a rank-μ update.

$$\boldsymbol{p}_{\boldsymbol{c}} \leftarrow (1-c_{\boldsymbol{c}})\,\boldsymbol{p}_{\boldsymbol{c}} + h_\sigma\sqrt{c_{\boldsymbol{c}}(2-c_{\boldsymbol{c}})\mu_{\mathrm{eff}}}\,\langle \boldsymbol{y}\rangle \quad (9)$$

$$\boldsymbol{C} \leftarrow (1-c_{\mathrm{cov}})\boldsymbol{C} + \frac{c_{\mathrm{cov}}}{\mu_{\mathrm{cov}}}\underbrace{\boldsymbol{p}_{\boldsymbol{c}}\boldsymbol{p}_{\boldsymbol{c}}^{\mathrm{T}}}_{\text{rank-one update}} + c_{\mathrm{cov}}\left(1-\frac{1}{\mu_{\mathrm{cov}}}\right)\underbrace{\sum_{i=1}^{\mu} w_i \boldsymbol{y}_{i:\lambda}\boldsymbol{y}_{i:\lambda}^{\mathrm{T}}}_{\text{rank-}\mu\text{ update}} \quad (10)$$

where $c_{\mathrm{cov}} \leq 1$ is a learning rate, $\mu_{\mathrm{cov}} \geq 1$ determines the portion between rank-one and rank-μ updates, and $h_\sigma = 0$ if $\|\boldsymbol{p}_\sigma\| > \left(1.5 + \frac{1}{n-0.5}\right)\widehat{\chi}_n\sqrt{1-(1-c_\sigma)^{2(g+1)}}$, and 1 otherwise, where g is the generation counter. Consequently, the update of $\boldsymbol{p}_{\boldsymbol{c}}$ is stalled whenever $\boldsymbol{p}_\sigma$ is considerably longer than expected. This mechanism is decisive after a change in the environment which demands a significant increase of the step-size whereas fast changes of the distribution shape are postponed until after the step-size is increased to a reasonable value.

For the covariance matrix update the cumulation in (9) serves to capture dependencies between consecutive steps. Dependency information would be lost for $c_{\boldsymbol{c}} = 1$, as a change in sign of $\boldsymbol{p}_{\boldsymbol{c}}$ or $\boldsymbol{y}_{i:\lambda}$ does not matter in (10). The rank-one update is particularly efficient with small offspring population sizes λ. Given $c_{\boldsymbol{c}} \propto 1/n$ the rank-one update can reduce the number of function evaluations needed to adapt to a straight ridge topography roughly from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ [29]. The rank-μ update exploits the information prevalent in a large population. Given a sufficiently large population, say $\lambda \approx n+3$, it reduces the number of generations needed to adapt a complex but globally constant topography roughly from $\mathcal{O}(n^2)$ to $\mathcal{O}(n)$ [29].

The default parameter values for all parameters, namely offspring population size λ, parent number μ, recombination weights w_i, cumulation parameter c_σ, step-size damping d_σ, cumulation parameter $c_{\boldsymbol{c}}$, mixing number μ_{cov}, and learning rate c_{cov} are [28] :

Selection and recombination:

$$\lambda = 4 + \lfloor 3\,\ln n\rfloor, \quad \mu = \lfloor \lambda/2\rfloor,$$

$$w_i = \frac{\ln(\mu+1) - \ln i}{\mu\ln(\mu+1) - \sum_{j=1}^{\mu}\ln j} \quad \text{for} \quad i = 1,\ldots,\mu,$$

Step-size control:

$$c_\sigma = \frac{\mu_{\mathrm{eff}} + 2}{n + \mu_{\mathrm{eff}} + 3},$$

$$d_\sigma = 1 + 2\times\max\left(0,\ \sqrt{\frac{\mu_{\mathrm{eff}}-1}{n+1}} - 1\right) + c_\sigma$$

Covariance Matrix Adaptation:

$$c_{\boldsymbol{c}} = \frac{4}{n+4}, \quad \mu_{\mathrm{cov}} = \mu_{\mathrm{eff}}$$

$$c_{\mathrm{cov}} = \frac{1}{\mu_{\mathrm{cov}}}\frac{2}{(n+\sqrt{2})^2} + \left(1-\frac{1}{\mu_{\mathrm{cov}}}\right)\min\left(1, \frac{2\mu_{\mathrm{eff}}-1}{(n+2)^2+\mu_{\mathrm{eff}}}\right)$$

A detailed discussion of the strategy parameters can be found in [31]. The identification procedure for c_{cov} with rank-μ update is described in [29]. Parameters for step-size adaptation, c_σ and d_σ, were accommodated for use with a large population size in [28]. With increasing μ_{eff} the backward time horizon and the change rate are reduced, such that the impact of step-size control diminishes in particular for $\mu_{\mathrm{eff}} \gg n$. All experiments in this paper are conducted with the default parameter settings.

Finally, we note that the structure of CMA-ES bears similarities with other stochastic optimization procedures, see *e.g.* [57], as well as with recursive estimation procedures, in particular of the "Gauss-Newton" type [41, pp. 366, 371, 375]. The analysis of these similarities and differences of these algorithms are far beyond the scope of this paper.

B. Box Constraint Handling

In the present algorithm parameter constraints are accounted by introducing a penalty term in the cost function. This penalty term quantifies the distance of the parameters from the feasible parameter space. The feasible space is a hypercube defined by lower and upper boundary values for each parameter. We implement a box boundary handling algorithm such that each

[2]Columns of $\boldsymbol{B}$ are an orthonormal basis of eigenvectors, $\boldsymbol{B}^{\mathrm{T}}\boldsymbol{B} = \boldsymbol{B}\boldsymbol{B}^{\mathrm{T}} = \mathbf{I}$. Diagonal elements of the diagonal matrix $\boldsymbol{D}$ are square roots of the corresponding positive eigenvalues. The matrix $\boldsymbol{D}$ can be inverted by inverting its diagonal elements. From these definitions it follows that $\boldsymbol{y}_k \sim \sigma\boldsymbol{B}\boldsymbol{D}\mathcal{N}(\mathbf{0}, \mathbf{I})$ which allows the generation of the random vector realizations on the computer.

evaluated solution is guaranteed to lie within the feasible space. This algorithm affects solely the evaluation of the solutions and entails the following steps.

- The cost of a solution $\boldsymbol{x}$ is obtained by evaluating the function L at $\boldsymbol{x}^{\text{feas}}$, where $\boldsymbol{x}^{\text{feas}}$ is the feasible solution closest to $\boldsymbol{x}$ (with minimal $\|\boldsymbol{x}^{\text{feas}} - \boldsymbol{x}\|$). Hence, a feasible solution is evaluated itself and an infeasible solution is evaluated on the boundary of the feasible space. The new feasible solution $\boldsymbol{x}^{\text{feas}}$ is used for the evaluation on L and for computing a penalty term, and it is discarded afterwards.
- A penalty term is added to the function value L penalizing infeasible solutions. The penalty depends on the distance to the feasible space and is weighted and scaled in each coordinate. The weights are set depending on observed function value differences in L and are increased if necessary, depending on the distance of the distribution mean $\boldsymbol{m}$ to the feasible space. The scaling is based on the covariance matrix diagonal elements.

The complete boundary procedure, applied after the candidate solutions of one generation are generated, reads as follows.

0) Initialization: the boundary weights γ_i are initialized once in the first generation step as $\gamma_i = 0$, for $i = 1, \ldots, n$.
1) Set weights: if the distribution mean $\boldsymbol{m}$ is out-of-bounds and either the weights were not set yet or the second generation step is conducted, set for all $i = 1, \ldots, n$,

$$\gamma_i = \frac{2\,\delta_{\text{fit}}}{\sigma^2 \times \frac{1}{n}\sum_{j=1}^{n} C_{jj}} \tag{11}$$

where δ_{fit} is the median from the last $20 + 3n/\lambda$ generations of the interquartile range of the unpenalized objective function values and C_{jj} is the j-th diagonal element of covariance matrix $\boldsymbol{C}$. The setting in Equation (11) is explained in conjunction with Equation (12) below.
2) Increase weights: for each component i, if the distribution mean m_i is out-of-bounds and the distance of m_i to the bound is larger than $3 \times \sigma\sqrt{C_{ii}} \times \max\left(1, \frac{\sqrt{n}}{\mu_{\text{eff}}}\right)$ (the typical distance to the optimum on the sphere function is coordinate wise proportional to $\sigma\sqrt{n}/\mu_{\text{eff}}$), the weight γ_i is increased according to

$$\gamma_i \leftarrow \gamma_i \times 1.1^{\max\left(1, \frac{\mu_{\text{eff}}}{10n}\right)} \quad .$$

This adjustment prevents the mean value of the distribution from moving too far away from the feasible domain (where far is naturally defined in terms of the given search distribution).
3) Compute the penalized function value for each candidate solution $\boldsymbol{x}$ as

$$L(\boldsymbol{x}) \stackrel{\text{def}}{=} L(\boldsymbol{x}^{\text{feas}}) + \frac{1}{n}\sum_{i=1}^{n} \gamma_i \frac{(x_i^{\text{feas}} - x_i)^2}{\xi_i} \tag{12}$$

where $\boldsymbol{x}^{\text{feas}}$ is the feasible solution closest to $\boldsymbol{x}$ with out-of-bounds components set to the respective boundary values. Only $\boldsymbol{x}^{\text{feas}}$ is actually evaluated on L. Finally $\xi_i = \exp\left(0.9\left(\log(C_{ii}) - \frac{1}{n}\sum_{j=1}^{n}\log(C_{jj})\right)\right)$ scales the distance coordinate wise with respect to the covariance matrix of the distribution, disregarding its overall size. The number 0.9 serves as regularizer to an isotropic shape (choosing zero would make all ξ_i isotropically equal to one).
Given that $x_i^{\text{feas}} - x_i = \sigma\sqrt{C_{ii}}$ is as large as typically sampled by the given search distribution (*i.e.* a one-σ sample) then the i-th summand in Equation (12) equals $\gamma_i \times \sigma^2 C_{ii}/\xi_i \approx \gamma_i \times \sigma^2$. With Equation (11) we have $\gamma_i \times \sigma^2 \approx 2\,\delta_{\text{fit}}$ which is a desired contribution. In particular the contribution of each component (summand) becomes identical and therefore the perturbation from the penalization on the covariance matrix adaptation procedure is minimized.

The additive penalization in Equation (12) is a quadratic function with its minimum located on the boundary [27, p.76]. Equation (12) has two important properties. First, it guarantees that the minimum of the resulting function L cannot be outside the feasible domain. Second, the construction results in a comparatively unproblematic function topography, because the partial derivative $\partial L(\boldsymbol{x})/\partial x_i$ approaches zero if the distance $x_i^{\text{feas}} - x_i$ approaches zero from the infeasible domain. For $\partial L(\boldsymbol{x})/\partial x_i \not\to 0$ a sharp ridge along the boundary can result which is quite undesirable.

C. A Method for Handling Uncertainty

We introduce a novel uncertainty-handling (UH) method, suitable for evolutionary optimization algorithms that employ rank based selection operators. The rank based selection operation allows for a robust quantification and handling of uncertainties in the cost function as shown in the following sections. We emphasize that the development of the proposed uncertainty-handling method is independent of the other operators employed in the evolutionary optimization algorithm. In the present work the UH is discussed, without loss of generality, in its implementation within the CMA-ES and the overall algorithm is referred as UH-CMA-ES. The proposed uncertainty handling preserves all invariance properties of the CMA-ES mentioned above. The method however biases the population variance when too large an uncertainty level is detected.

The uncertainty handling consists of two separate components.

- Quantification of the uncertainty effect on the ranking of the members of the population
- Treatment of the uncertainty, if necessary, to prevent the search algorithm from premature convergence

a) Uncertainty Quantification: We propose a reevaluation technique that provides a quantification of the uncertainty for any ranking-based search algorithm. The uncertainty in the objective function can affect a ranking-based search algorithm only if changes of the ordering of solutions occur. Hence, the uncertainty quantification is based on rank changes induced by reevaluations of solutions. A small perturbation can be applied, before the reevaluation is done, to cover "frozen noise", *i.e.* when the objective function is not a random variable itself

but a single realization of a noisy function L. The uncertainty quantification algorithm reevaluates each solution at most once but an extension to more reevaluations is straightforward.

First the solutions to be reevaluated are selected at random. Alternatively the best solutions might be selected for reevaluation, while our preliminary tests did not indicate major differences. More importantly, we conjecture that more scenarios exist where selecting the best solutions fails. One such scenario is the case of (seldom found) good outliers. Using the best solution then invariably selects the outlier and therefore the uncertainty treatment might be applied too often. In order to circumvent this difficulty solutions are chosen randomly.

Second, after reevaluation, the number of rank changes, Δ_i, that occur with the reevaluation of solution i, is computed. Third, the measured rank changes are compared to a threshold, or, in a sense, normalized leading to the uncertainty measurement s. The algorithm reads

1) Set $L_i^{\text{new}} = L_i^{\text{old}} = L(\boldsymbol{x}_i)$, for $i = 1, \ldots, \lambda$, and let $\mathcal{L} = \{L_k^{\text{old}}, L_k^{\text{new}} | k = 1, \ldots, \lambda\}$.
2) Compute λ_{reev}, the number of solutions to be reevaluated using parameter $r_\lambda \leq 1$; $\lambda_{\text{reev}} = f_{\text{pr}}(r_\lambda \times \lambda)$ where the function $f_{\text{pr}} : \mathbb{R} \to \mathbb{Z}$, $x \mapsto \begin{cases} \lfloor x \rfloor + 1 & \text{with probability } x - \lfloor x \rfloor \\ \lfloor x \rfloor & \text{otherwise} \end{cases}$. To avoid too long sequences without reevaluation set $\lambda_{\text{reev}} = 1$ if $\lambda_{\text{reev}} = 0$ for more than $2/(r_\lambda \times \lambda)$ generations.
3) Reevalute solutions. For each solution $i = 1, \ldots, \lambda_{\text{reev}}$ (because the solutions of the population are i.i.d., we can, w.l.o.g., choose the first λ_{reev} solutions for reevaluation)
   a) Apply a small perturbation: $\boldsymbol{x}_i^{\text{new}} = \texttt{mutate}(\boldsymbol{x}_i, \varepsilon)$ where $\boldsymbol{x}_i^{\text{new}} \neq \boldsymbol{x}_i \iff \varepsilon \neq 0$. According to (3) for the CMA-ES we apply $\texttt{mutate}(\boldsymbol{x}_i, \varepsilon) = \boldsymbol{x}_i + \varepsilon\sigma\mathcal{N}(\boldsymbol{0}, \boldsymbol{C})$.
   b) Reevaluate the solution: $L_i^{\text{new}} = L(\boldsymbol{x}_i^{\text{new}})$
4) Compute the rank change Δ_i. For each chosen solution $i = 1, \ldots, \lambda_{\text{reev}}$ the rank change value, $|\Delta_i| \in \{0, 1, \ldots, 2\lambda - 2\}$, counts the number of values from the set $\mathcal{L} \setminus \{L_i^{\text{old}}, L_i^{\text{new}}\}$ that lie between L_i^{new} and L_i^{old}. Formally we have

$$\begin{aligned}\Delta_i = {} & \texttt{rank}(L_i^{\text{new}}) - \texttt{rank}(L_i^{\text{old}}) \\ & - \texttt{sign}\left(\texttt{rank}(L_i^{\text{new}}) - \texttt{rank}(L_i^{\text{old}})\right)\end{aligned}$$

   where $\texttt{rank}(L_i)$ is the rank of the respective function value in the set $\mathcal{L} = \{L_k^{\text{old}}, L_k^{\text{new}} | k = 1, \ldots, \lambda\}$.
5) Compute the uncertainty level, s. The rank change value, Δ_i, is compared with a given limit $\Delta_\theta^{\lim}$. The limit is based on the distribution of the rank changes on a random function L and the parameter θ. Formally we have

$$\begin{aligned}s = {} & \frac{1}{\lambda_{\text{reev}}} \sum_{i=1}^{\lambda_{\text{reev}}} \Big(2\,|\Delta_i| \\ & - \Delta_\theta^{\lim}\left(\texttt{rank}(L_i^{\text{new}}) - \mathbb{1}_{L_i^{\text{new}} > L_i^{\text{old}}}\right) \\ & - \Delta_\theta^{\lim}\left(\texttt{rank}(L_i^{\text{old}}) - \mathbb{1}_{L_i^{\text{old}} > L_i^{\text{new}}}\right)\Big)\ , \end{aligned} \quad (13)$$

   where $\Delta_\theta^{\lim}(R)$ equals the $\theta \times 50\%$ile of the set $\{|1 - R|, |2 - R|, \ldots, |2\lambda - 1 - R|\}$, that is, for a given rank R, the set of absolute values of all equally probable rank changes on a random function L (where f and N_f are independent of $\boldsymbol{x}$). The summation for s in Equation (13) computes two values for $\Delta_\theta^{\lim}$ and therefore respects the symmetry between L_i^{old} and L_i^{new}.
6) Re-rank the solutions according to their rank sum, *i.e.* $\texttt{rank}(L_i^{\text{old}}) + \texttt{rank}(L_i^{\text{new}})$. Ties are resolved first using the absolute rank change $|\Delta_i|$, where the mean $\Delta_i = \frac{1}{\lambda_{\text{reev}}} \sum_{j=1}^{\lambda_{\text{reev}}} |\Delta_j|$ is used for solutions $i > \lambda_{\text{reev}}$ not being reevaluated, and second using the (mean) function value.

The parameters are set to $r_\lambda = \max(0.1, \frac{2}{\lambda})$, $\varepsilon = 10^{-7}$, and $\theta = 0.2$. A Matlab implementation for the computation of the uncertainty measurement s from the set of function values $\mathcal{L}$ (steps 4 and 5) is given in the appendix.

In Equation (13) differences between the rank change Δ_i and the limit rank change $\Delta_\theta^{\lim}$ are summed. Alternatively, only the sign of the difference could be used thus placing less emphasis on single large deviations that are typically observed in the presence of outliers. When only the sign is used it will also be appropriate to average s in the generation sequence by choosing $c_s > 0$ below.

b) Treatment of Uncertainty: The quantification of uncertainty as described above is independent of algorithms developed for the treatment of this uncertainty. In this paper we propose two methods for the treatment of uncertainty.

1) Increase of the evaluation (measuring) time of the controller's performance. Increasing the evaluation time aims to reduce the uncertainty in the evaluation. In particular for the feedback controller of the combustion set-up increasing the evaluation time is more natural than taking the mean value from multiple evaluations, as it avoids repeated ramping up and down of the controller. Otherwise, doubling the evaluation time is equivalent to taking the mean of two evaluations.
2) Increase of the population variance. This treatment can have three beneficial effects.
   - The signal-to-noise ratio is most likely improved, because the solutions in the population become more diverse.
   - The population escapes search-space regions with too low a signal-to-noise ratio, because in these regions the movement of the population is amplified.
   - Premature convergence is prevented.

The following uncertainty treatment algorithm is applied after each generation step employing uncertainty measurement s.

$\overline{s} \leftarrow (1 - c_s)\,\overline{s} + c_s s$
if $\overline{s} > 0$ % apply uncertainty treatment
  if $t_{\mathrm{eval}} < t_{\max}$
    $t_{\mathrm{eval}} \leftarrow \min(\alpha_t t_{\mathrm{eval}}, t_{\max})$
  else
    $\sigma \leftarrow \alpha_\sigma \sigma$
else if $\overline{s} < 0$ % decrease evaluation time
  $t_{\mathrm{eval}} \leftarrow \max(t_{\mathrm{eval}}/\alpha_t, t_{\min})$

Initialization is $t_{\mathrm{eval}} = t_{\min}$ and $\overline{s} = 0$ and the parameters are chosen to $c_s = 1$, $\alpha_\sigma = 1 + 2/(n+10)$, $\alpha_t = 1.5$, $t_{\min} = 1\mathrm{s}$, $t_{\max} = 10\mathrm{s}$. If the uncertainty measurement value $\overline{s}$ exceeds zero, the evaluation time t_{eval} is increased. If t_{eval} has already reached its upper bound $t_{\max}$, step-size σ is increased. Otherwise, if $\overline{s}$ is below zero, time t_{eval} is decreased. An adaptive evaluation time t_{eval} proves to be particularly useful in the early stages of an optimization run or when the operating condition is changed. In the later stages, the evaluation time will usually reach the upper bound and the adaptation will become ineffective.

c) Role of Parameters: We discuss the role of the parameters of the uncertainty measurement and the uncertainty treatment algorithm.

- $r_\lambda \in]0, 1]$, typically < 0.5, determines the fraction of solutions to be reevaluated. For $r_\lambda = 0.3$ a fraction of 30% of the solutions in the population is reevaluated. For $r_\lambda = 1$ each solution is evaluated twice. To establish a sufficiently reliable uncertainty measurement r_λ has to be chosen large enough. To minimize the additional costs (in terms of number of function evaluations), r_λ should be chosen as small as possible.
- $\varepsilon \geq 0$ and $\varepsilon \ll 1$: Mutation strength for the reevaluation, given relative to the recent mutation strength. To be able to treat "frozen" noise similar as stochastic noise, ε must be set greater than zero, such that a slightly different solution is used for the reevaluation. For the CMA-ES, according to Equation (3), we replace σ by $\varepsilon\sigma$ and $\boldsymbol{m}$ by $\boldsymbol{x}_i$ for generating a new solution to re-evaluate $\boldsymbol{x}_i$. Note that for too small ε the mutation can most likely be influenced by numerical precision.
- $\theta \in [0, 1]$: Control parameter for the acceptance threshold for the measured rank-change value. The threshold $\theta = 1$ corresponds to the median rank change that occurs under purely random selection. This is clearly an upper bound for a reasonable setting of θ.
- c_s: Learning rate for averaging the uncertainty measurement s in the generation sequence. Decreasing c_s will decrease the variance in the measurement $\overline{s}$. Using $c_s = 0.5$ instead of $c_s = 1.0$ will have a similar effect to increasing r_λ by a factor of two. Note that decreasing c_s is inexpensive when compared to increasing r_λ. On the other hand decreasing c_s introduces a time delay.
- $\alpha_\sigma > 1$: Factor for increasing the population spread (step-size) when the measured uncertainty value is above the threshold. Values larger than 2 are rarely reasonable. To make divergence most unlikely, α_σ should be as small as possible. This is particularly relevant when increasing the population spread has no significant influence on the uncertainty level, as it is the case with outliers.
- α_t: Factor for increasing the evaluation time when the measured uncertainty value is above the threshold. To achieve fast enough changes, α_t should be chosen large enough, typically not smaller than 1.2.
- $t_{\min}, t_{\max}$ are chosen based on pressure measurement data and requirements of the technical facilities of the test rig.

The final parameter settings, given above, were specified based on simulations of the uncertainty handling with the CMA-ES on the sphere function. Different parameter settings may be necessary when combining the uncertainty handling with different evolutionary algorithms.

d) Applications of Uncertainty Handling for Feedback Controllers: The two techniques, presented above, provide different treatments of uncertainty during the optimization of the combustion feedback controllers. The increase of the evaluation time is the most straightforward way to implement resampling during the operation of the controllers (and can be replaced by resampling in another application using $\lceil t_{\mathrm{eval}} \rceil$ as the number of samples). Different evaluation times were successfully applied to combustion control in [49], [55]. Longer evaluation times reduce the amount of uncertainty and the controller parameters can get closer to their desired values. For an unbounded evaluation time, UH-CMA-ES has the capability to approach the optimum with arbitrary accuracy. In order to retain however adaptability the evaluation time needs to have an upper bound.

The increase of σ ensures that the evolution strategy remains in a working regime where sufficient selection information is available. This is important, as changing operating conditions can affect the desired controller parameters and the algorithm has to track these changes even in a late stage of the optimization. The increase of σ can only be useful if the introduced increase in the population spread leads to an improved signal-to-noise ratio. Our empirical observations show that this truly holds for our application. In particular we never observed divergence of step-size σ.

The upper bound for the evaluation time limits the possible accuracy of the control parameters. The optimum cannot be approximated with arbitrary accuracy if in its vicinity the signal-to-noise ratio is too low. This failure is a property of the evolution strategy in general and is actually not caused by the uncertainty handling. The uncertainty handling only prevents the step-size to become arbitrarily small. We believe that in a noisy online application, where the optimum can change in time, in the end a trade-off exists between the objectives to retain adaptability versus getting arbitrarily close to the optimum.

V. Results on Test Functions

The effect of increasing the evaluation time on the performance of the algorithm is predictable. If the increment is fast enough the algorithm will remain operating reliably and converge to the optimum while the time per function evaluation will increase unboundedly. Hence, we are mainly interested in the effect of the step-size increment. On certain multi-modal

functions the increase of the step-size might occasionally help to locate the domain of a better local optimum, but we believe that this effect is of minor relevance. Overall, we do not expect that the uncertainty-handling would impair the performance of the algorithm on multi-modal functions. Hence we do not include experiments in multi-modal functions and we present experiments on several unimodal functions with uncertainty. These functions can also be interpreted as rugged, highly multi-modal (non-noisy) functions, because any single solution is (virtually) never evaluated twice. The "re-evaluations" are conducted with a slightly mutated solution (compare point 3 in the uncertainty measurement algorithm). Therefore no difference between "stochastic" and "frozen noise" can be observed.

The test functions obey

$$L(\boldsymbol{x}) = f(\boldsymbol{x}) + N_f(\boldsymbol{x}) = \sum_{i=1}^{n} a_i (x_i - b)^2 + N/t_{\text{eval}}^{\beta}, \quad (14)$$

where $a_i, b \in \mathbb{R}$ are chosen function-dependent and the uncertainty term is independent of $\boldsymbol{x}$ but scaled with t_{eval}^{β}, and $\beta > 0$. While this test function is additively decomposable and hence unrealistic simple, all our simulation results also hold for non-decomposable (rotated) versions. Three functions are derived, where $\beta = 0.5$.

- L_{sphere}^{C} the isotropic sphere function, where $a_i = 1$, for $i = 1, \ldots, n$, $b = 0$, and N is standard Cauchy distributed.
- L_{elli} the ellipsoid function, where $a_i = 10^{6 \times \frac{i-1}{n-1}}$, $b = 0$, and N is standard normally distributed. The condition number is 10^6. The principal axis lengths are equidistant on the log scale.
- L_{elli}^{C} the ellipsoid function L_{elli}, where $b = 5$, and N is standard Cauchy distributed.

Figure 3 shows five independent runs on L_{elli} in 10D. Premature convergence is observed without uncertainty handling (CMA-ES, left). The smallest standard deviation $\sigma\sqrt{\lambda_{\min}}$, where $\lambda_{\min}$ is the smallest eigenvalue of $\boldsymbol{C}$, exhibits a drift with all values clearly below 10^{-4}. With uncertainty handling (UH-CMA-ES, right) we choose $t_{\min} = t_{\max} = 1$ here for simplicity, implying constant $t_{\text{eval}} = 1$. Therefore only σ is changed by the uncertainty treatment. The smallest standard deviation $\sigma\sqrt{\lambda_{\min}}$ reaches a stationary value and does not drop below 10^{-4}. If t_{eval} is chosen much larger such that no rank changes occur (non-noisy case), about 20% fewer function evaluations are required to reach a function value of 1 (not shown). Note that even though the smallest standard deviation is larger than in cases without uncertainty handling, the function values are clearly better and give first evidence that the uncertainty handling works effectively.

Figure 4 shows a single run of UH-CMA-ES on L_{elli} in 8D. The course of σ (upper left) reveals that the uncertainty treatment enters after about 2000 function evaluations. After about 5000 function evaluations the adaptation of the covariance matrix is completed. The eigenvalues of the covariance matrix correspond to the inverse coefficients a_i^{-1} of L_{elli} and indicate an almost perfect adaptation to the function topography, despite the noisy environment. In view of the fact that only σ is changed to treat the noisy environment, this is a remarkable result. The mean vector fluctuates around the optimum zero (upper right), while the size of the fluctuations differs for different variables (coordinates), according to their sensitivities a_i.

Figure 5 shows a single run of UH-CMA-ES, switching from L_{sphere}^{C} to L_{elli}^{C} after 3000 function evaluations, and back again after 6000 function evaluations. Here, $t_{\min} = 1$, $t_{\max} = 10$ and t_{eval} is shown in the upper left. The initial σ is chosen far too small. Consequently σ increases from 10^{-2} to 2 in the beginning. During convergence on L_{sphere}^{C} σ drops to 2×10^{-1}, while t_{eval} increases to the upper bound $t_{\max} = 10$. When the objective function is switched t_{eval} drops to the lower bound, because the uncertainty becomes negligible when compared to the differences in the function values f. In addition σ increases fast, because the optimum has been displaced. At about 4000 function evaluations t_{eval} starts to increase again, because the uncertainty term becomes significant. As expected, the covariance matrix adapts to the new topography and the x-variables move to the new optimum, arranged according to their relevance. Switching back to L_{sphere}^{C} reveals a similar picture. The step-size increases, t_{eval} decreases, and the isotropic topography has to be re-learned. Because there is no distinct coordinate system the re-learning takes as much time as learning the elliptic topography starting from a spherical distribution. This procedure can be naturally accelerated by reseting the covariance matrix. Further experiments, conducted with larger dimensions, on non-quadratic objective functions, and with $\boldsymbol{x}$-dependent uncertainty terms, give similar results (not shown).

VI. Experimental Results

A. Implementation of the Algorithm on the Test Rig

The UH-CMA-ES delivers a set of controller parameters to be evaluated together with a requested function evaluation time. The controller parameters undergo an affine transformation from the interval $[0, 1]$ onto respective intervals specified below, and the initial values are set to the middle of the interval. The controller is assembled and written to the real-time board. In order to avoid any risks stemming from inappropriate parameter settings delivered by the algorithm, the gain of the new controller is ramped up over the course of two seconds, such that the human operator can intervene in case of a developing harmful situation. After the data acquisition has been completed, the controller gain is ramped down, and an intermediate controller keeps the combustor in a stable regime. Meanwhile, pressure data is logged, a new controller is developed and transferred to the real-time board. The total cycle time thus consists of ramping the controller gain up and down (about 2 s each), pressure data acquisition (determined by the algorithm, 1-10 s), data logging (1 s) and UH-CMA-ES computation time (negligible). The maximum time that pressure can be logged is currently limited to 10 s, due to real-time board memory constraints. The controller is sampled at 10 kHz, the frequency content of the pressure signal warrants no aliasing effects during sampling. For the following experiments, the preheat temperature and the mass flow are

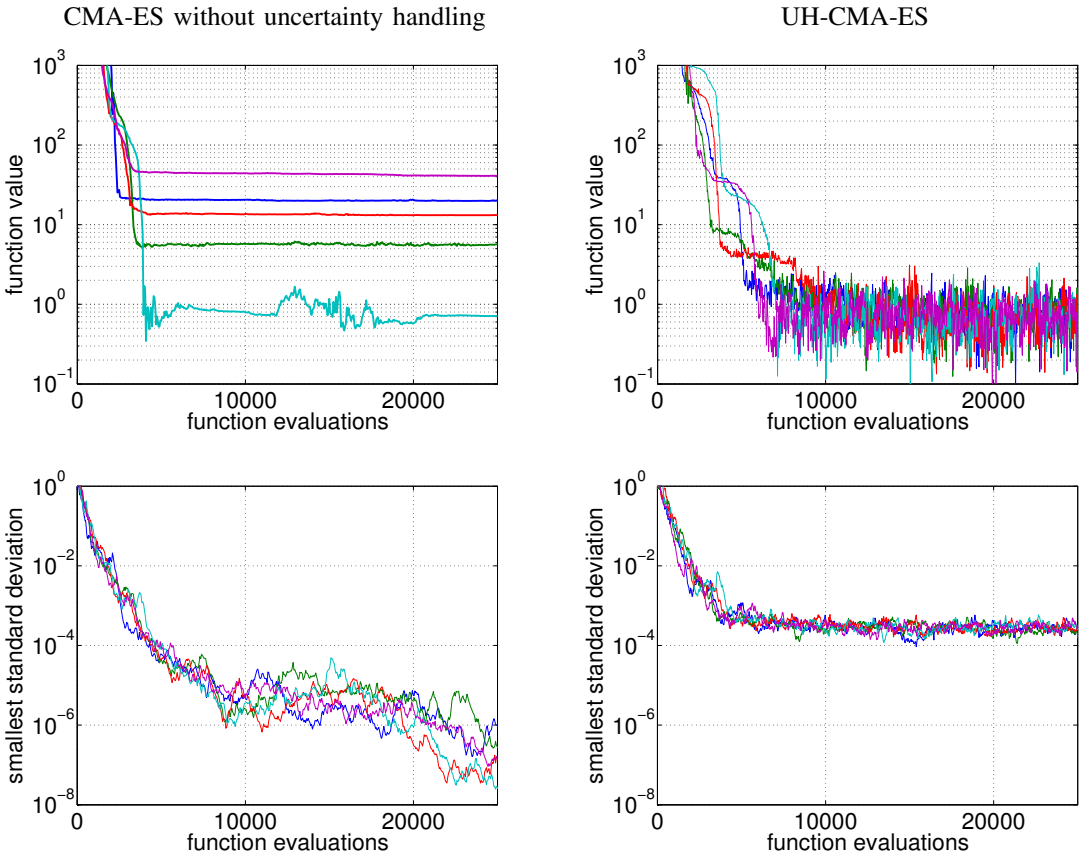

Fig. 3. Five runs on L_{elli}, where $n = 10$, and $t_{\min} = t_{\max} = 1$. Left: CMA-ES without uncertainty handling; right: UH-CMA-ES (with uncertainty handling); above: function value of the population mean, $f(\boldsymbol{m})$; below: standard deviation in the smallest principal axis. The uncertainty handling keeps the smallest standard deviation above 10^{-4} and prevents premature convergence.

kept constant at 700 K and 36 g/s, respectively. Two values for the air/fuel ratio λ are investigated, namely $\lambda = 2.1$ and $\lambda = 1.875$.

B. Experiment: Gain-Delay Controller, Cold Start, $\lambda = 2.1$ and Switch to $\lambda = 1.875$.

The combustor is fired up from ambient temperature, an operating condition is set with a mass flow of 36 g/s, a preheat temperature of 700 K, and an air/fuel ratio of $\lambda = 2.1$, and the Gain-Delay controller is turned on. As the system heats up, the sound pressure level L_{eq} from (1) rises. Previous studies have shown that the maximum absolute value of the gain for a Gain-Delay controller decreases as the combustor heats up for this operating condition. This is attributed to the fact that the low-frequency content of the pressure signal rises, and the resulting low frequency components of the fuel injection tend to alter the flame stabilization. The flame then flaps back and forth and increases the uncertainty levels.

The heat-up phase is also evident in the pressure spectra of the controlled combustor taken at 1000 s and 4700 s, shown in **Fig. 6**. The plant is uncontrolled and Gain-Delay controlled (gain -1.8×10^{-4}, delay 0.3 ms), the resulting L_{eq} are 159.87 dB, 146.90 dB, and 147.48 dB, respectively.

The UH-CMA-ES optimizes the gain and the delay of the Gain-Delay controller. The evolution of the parameters is shown in **Fig. 7**, where the gain interval $[-3 \times 10^{-4}, 0]$ and

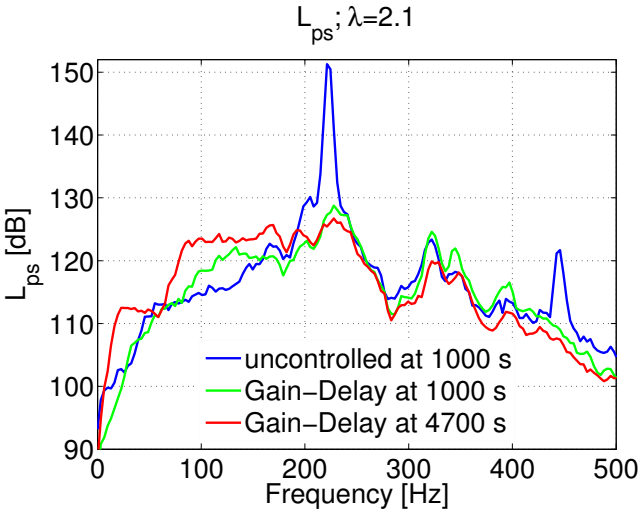

Fig. 6. A comparison of the uncontrolled and controlled spectra of the pressure signal at 1000 s and 4700 s for the same Gain-Delay controller and $\lambda = 2.1$.

the delays from $\{0.1, 0.2, \ldots, 1.5\}$ ms are mapped onto $[0, 1]$. Previous experiments with manual tuning have shown that actuator saturation and flame stabilization problems occur if the gain is chosen lower than -3×10^{-4}, or the delay is higher than 1.5 ms. The initial gain and delay passed to the UH-

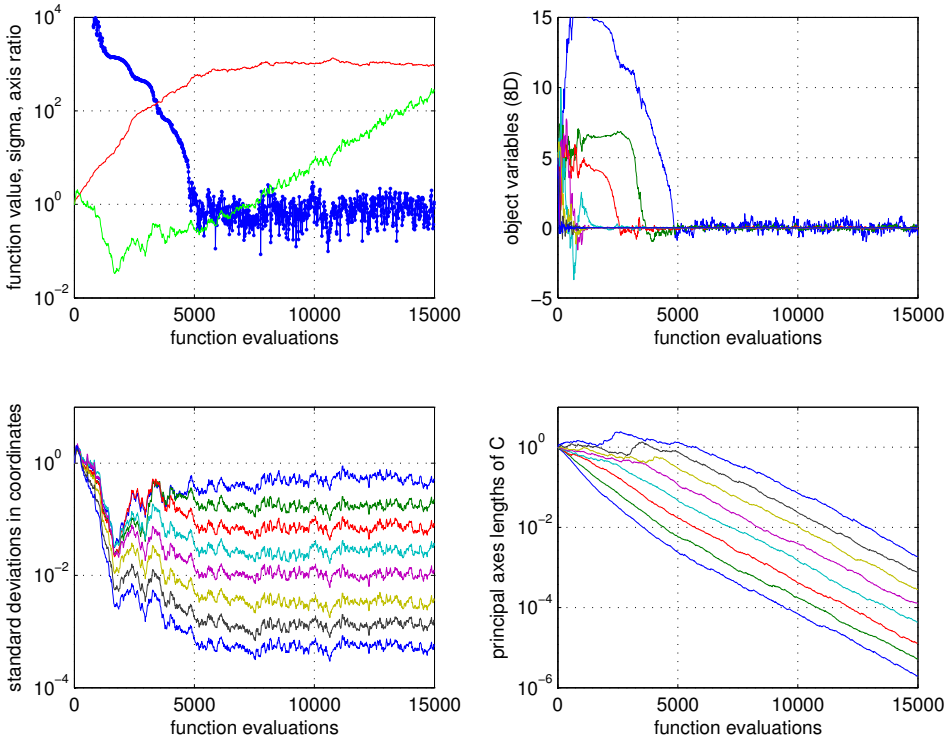

Fig. 4. A single run of the UH-CMA-ES (with uncertainty handling) on L_{elli}, where $n = 8$, and $t_{\min} = t_{\max} = 1$. Upper left: function value of the distribution mean, $f(\boldsymbol{m})$ (thick line with dots), step-size σ (mostly increasing line), ratio between largest and smallest principal axis length of $\boldsymbol{C}$ (flattening line). Upper right: components of mean vector $\boldsymbol{m}$. Lower left: coordinate-wise standard deviations $\sigma \times \sqrt{C_{ii}}$, where C_{ii} is the i-th diagonal element of $\boldsymbol{C}$. Lower right: square root of eigenvalues of $\boldsymbol{C}$

CMA-ES algorithm are -1×10^{-7} and 1.5 ms, respectively. During the first 4800 seconds L_{eq} rises as the combustor heats up, and the optimal value of the gain increases from about -2.5×10^{-4} at 1000 s to -1.8×10^{-4} at 4800 s. The rise of L_{eq} is related to the persistent change of the system conditions during heat-up and seems to have no adverse effect on the optimization. During the first 1000 s the evaluation time increases and reaches 10 s, the maximum allowed. That means uncertainty is becoming an issue. The standard deviations decrease during the first 4000 seconds and rise again as the operating condition is changed. At 4800 s, the operating condition is changed from $\lambda = 2.1$ to $\lambda = 1.875$, and the evaluation time is manually set to 1 s.

Four cost function landscapes for different time intervals are shown in **Fig. 8**. They are obtained by Delauney triangulation of a second-order polynomial fit to the L_{eq} results for the individual delay slices. Pentagrams show the best parameter set for each generation; the larger they are, the later they have been acquired for each plot. A black circle marks the last of the pentagrams. The topmost plot shows the L_{eq} for the first 150 function evaluations (up to 1300 s). The plot shows that the gain can be chosen quite negative; the overall landscape features low L_{eq} values. For the function evaluations from 150 to 250 (1300-2700 s), the evaluation time increases and yields results with less uncertainty. A trend to less negative values for the gain becomes apparent (the pentagrams indicating the best of the generations are moving to the right), and the general

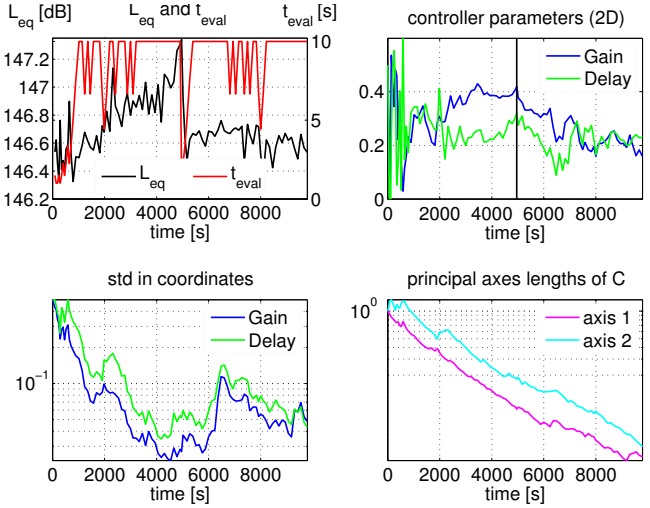

Fig. 7. Parameter evolution for the UH-CMA-ES optimization of a Gain-Delay-controller. At 4800 s, the operating condition is changed from $\lambda = 2.1$ to $\lambda = 1.875$.

background uncertainty level rises (indicated by areas getting darker).

The black polygon is the convex hull of all controller parameter values tried in the given time range. The function evaluations 250-325 (2700-3800s), shown in the third plot, indicate that the optimal value for the gain lies around -1.8×10^{-4} and a delay of 0.4-0.5 ms. The parameters evaluated

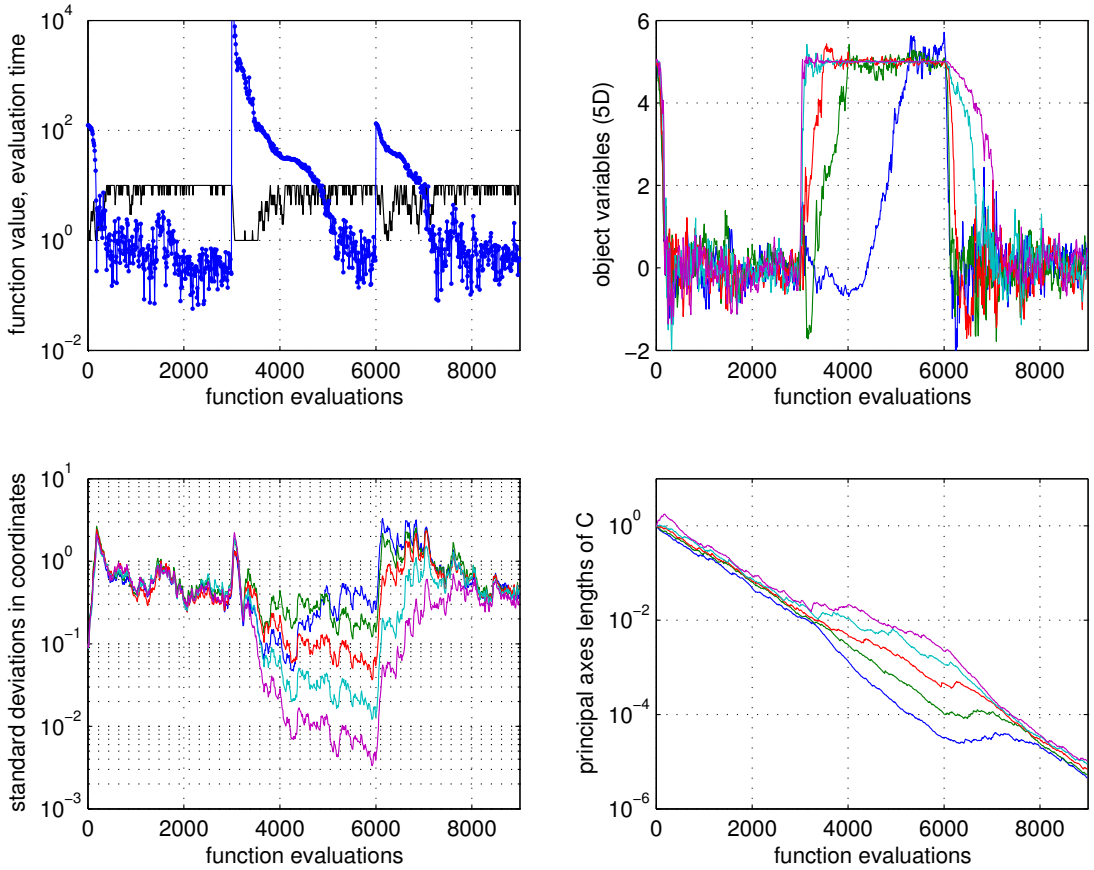

Fig. 5. A single run of the UH-CMA-ES switching between L^C_{sphere} and L^C_{elli} at 3000 and 6000 function evaluations, where $n = 5$, $\beta = 0.5$ (see Equation (14)), initial $\sigma = 10^{-2}$, initial $t_{\text{eval}} = 1$. The graphs remain identical for any $\beta > 0$ given $\alpha_t = 1.5^{\frac{0.5}{\beta}}$. Upper left: function value of the distribution mean $f(\boldsymbol{x})$ (line with dots), and t_{eval}. Upper right: components of mean vector $\boldsymbol{m}$. Lower left: coordinate-wise standard deviations $\sigma \times \sqrt{C_{ii}}$, where C_{ii} is the i-th diagonal element of $\boldsymbol{C}$. Lower right: square root of eigenvalues of $\boldsymbol{C}$

are now narrowed down to the smaller black polygon. If this result is compared to the last plot showing function evaluations 325-390 (3800-4800 s), the optimal values for the gain and the delay are confirmed, but the cost function evaluated L_{eq} rises. This is in accord with the observation that the combustor exhibits slowly rising sound pressure levels for $\lambda = 2.1$.

At run 395 (4800 s), the lambda value is changed to $\lambda = 1.875$. This operating conditions exhibits less thermal drift than the previous one. According to **Fig. 9** the changing operating conditions can clearly be discerned in the cost function L_{eq}. The algorithm finds a new minimum, where the gain can be more negative for this case.

The evaluation time increases immediately again, indicating no big improvement of the signal-to-noise ratio, even though the controller is less close to its optimal regime. This suggests that σ should be increased together with t_{eval}. The course of σ supports this conjecture. It increases by a factor of three and shows the adaptive capability of the algorithm. It takes 8 generations until the increase of σ appears, and we do not know whether this reflects a sensible adaptation or whether σ should have increased beforehand. Finally, the UH-CMA-ES successfully adjusts the controller parameters to new improved values, as shown in Fig. 7.

C. Experiment: $\mathcal{H}_\infty$ Controller, Two Parameters Optimized, $\lambda = 1.875$.

An $\mathcal{H}_\infty$ controller has been designed for the operating condition with $\lambda = 1.875$, where the goal was to simultaneously decrease the three peaks at 250 Hz and around 330 Hz. In order to keep the number of parameters small and to speed up convergence, only the gain and the frequency shift are optimized.

In the top plot of **Fig. 10** the intervals for frequency shift, $[0.95, 1.05]$, and gains $[0.4, 1.1]$, are mapped onto $[0, 1]$. The comparatively good values for L_{eq} in the beginning are related to the short evaluation time. The shorter the evaluation time is, the larger is its variation due to the uncertainty. Therefore better values occur more often. The bottom plot shows the cost function landscape. It is obtained by DACE, a Matlab toolbox for working with kriging approximations, which has been kindly provided by Hans Bruun Nielsen from the Technical University of Denmark[3]. A second-order polynomial has been used as regression model and a Gaussian correlation. For this experiment with an $\mathcal{H}_\infty$ controller, the cost function is flatter than with the Gain-Delay controller, because the controller is model-based, and thus already performs well. However, the optimization shows that in order to decrease L_{eq}, the gain has

[3]http://www2.imm.dtu.dk/∼hbn/dace/

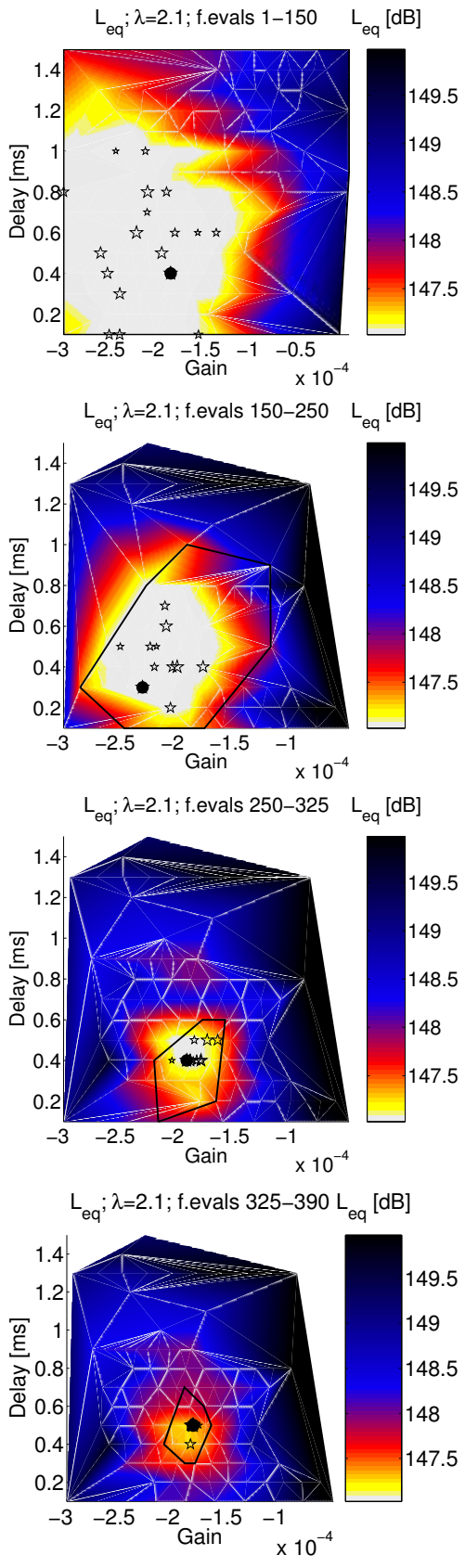

Fig. 8. UH-CMA-ES optimization of a Gain-Delay controller for $\lambda = 2.1$, heating up. Function evaluations, from top plot down: 1-150; 150-250; 250-325; 325-390. Pentagrams show the best parameter set for each generation, the larger they are, the later they have been acquired for each plot. The black polygon is the convex hull of all controller parameter values tried in the given time range.

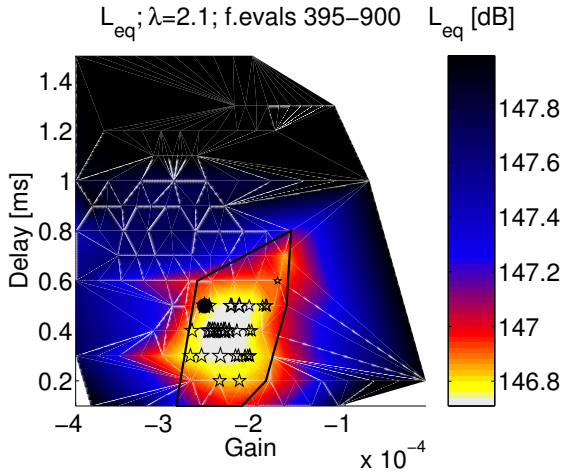

Fig. 9. UH-CMA-ES optimization of a Gain-Delay controller for $\lambda = 1.875$. Function evaluations $395 - 740$ (4900-9800 s), see Fig. 8 for explanations

to be reduced from the design value of 1 to about 0.7. This is explained by the fact that the $\mathcal{H}_\infty$-design process has been laid out primarily to decrease the three peaks in the spectrum, without special concern given to the reduction of L_{eq}.

D. Experiment: $\mathcal{H}_\infty$ Controller, Three Parameters Optimized, $\lambda = 1.875$.

For the following experiment, three parameters are optimized by the UH-CMA-ES, namely gain, frequency shift and time delay of the $\mathcal{H}_\infty$ controller.

The evolution of the parameters is shown in **Fig. 11** (frequency shift interval $[0.95, 1.05]$, gains $[0.4, 1.1]$, delays $[1, 10]$). Since the cost function takes three arguments, only four cost function landscapes with fixed delays of 0.1 ms to 0.4 ms are shown in **Fig. 12**. The topmost plot corresponds to the bottom plot of Fig. 10, where only frequency shift and gain are adjusted, but the delay is kept at 0.1 ms for all experiments. Gain and frequency shift have similar values but exhibit a larger variation. The minimum L_{eq} is lower for a delay of 0.2 ms, and even lower for a delay of 0.3 ms, while it increases again for 0.4 ms (bottom plot).

The Bode plots of the designed and optimized $\mathcal{H}_\infty$ controllers are shown in **Fig. 13**. The superior performance of the $\mathcal{H}_\infty$ controller goes hand in hand with a more complex shape. The optimized controller has nearly the same phase as the designed one, but the gain is lower. Since the delay is adjusted additionally, it is possible to move the controller in the frequency domain keeping the same phase.

As a result for the operating condition characterized by $\lambda = 1.875$, the spectra achieved with the optimized Gain-Delay and $\mathcal{H}_\infty$ controllers are compared to the uncontrolled plant in **Fig. 14**. The L_{eq} of the uncontrolled plant is 148.72 dB, the Gain-Delay controller reduces it to 146.67 dB, and finally the $\mathcal{H}_\infty$ controller reaches 146.16 dB, which is about 15% lower again. Moreover, the $\mathcal{H}_\infty$ controller is able to simultaneously push down all three peaks and to attain the flattest spectrum. This is achieved thanks to the model-based approach, conferring the most design freedom to the engineer.

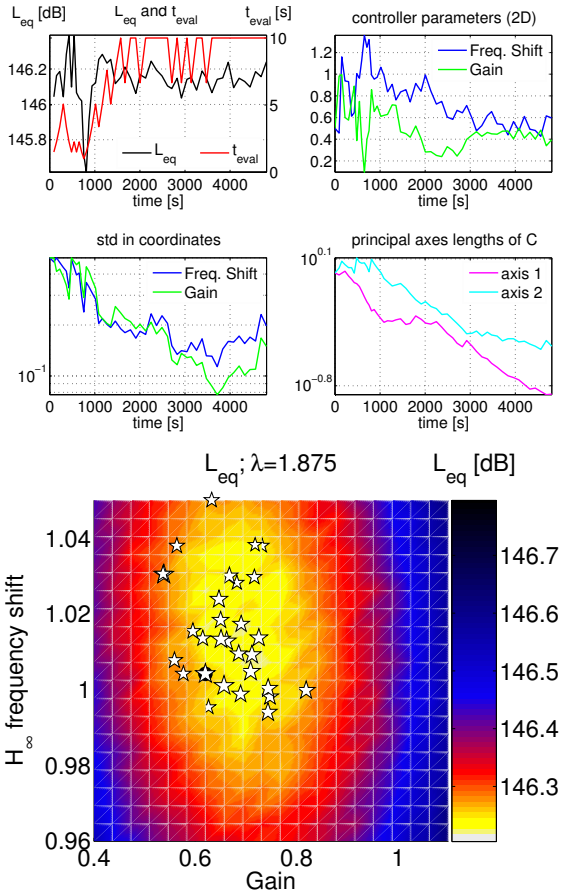

Fig. 10. Parameter evolution for the UH-CMA-ES optimization of a $\mathcal{H}_\infty$ controller for $\lambda = 1.875$ (top) and the resulting cost function landscape (bottom). Pentagrams show the best parameter set for each generation.

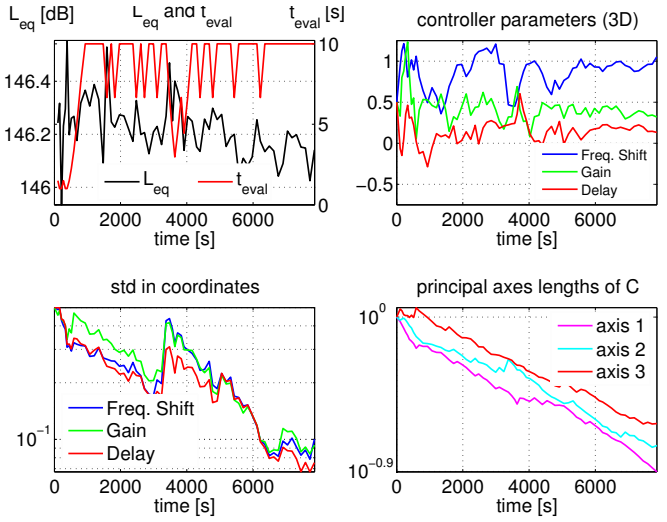

Fig. 11. Parameter evolution for the UH-CMA-ES optimization of a $\mathcal{H}_\infty$ controller for $\lambda = 1.875$.

E. Experiment: $\mathcal{H}_\infty$ Controller, Two Parameters Optimized, $\lambda = 2.1$.

Finally, the UH-CMA-ES is used to improve an $\mathcal{H}_\infty$ controller designed for $\lambda = 2.1$. The parameter evolution is shown in **Fig. 15** at the top (frequency shift interval $[0.95, 1.05]$,

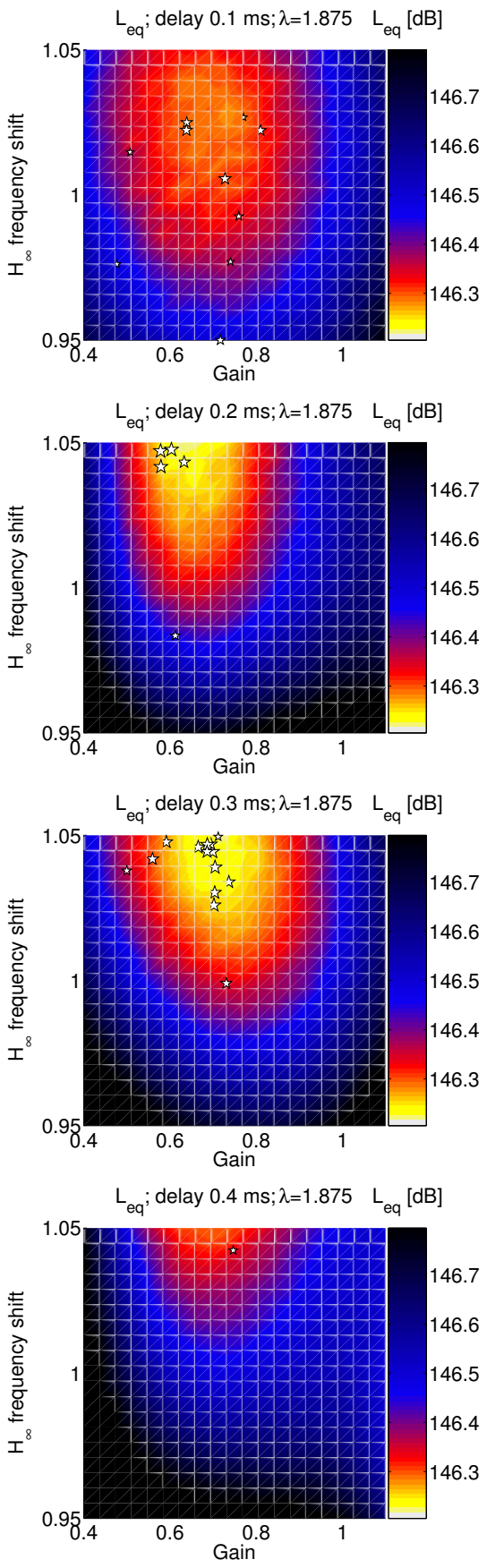

Fig. 12. UH-CMA-ES optimization of an $\mathcal{H}_\infty$ controller for $\lambda = 1.875$. Delays from top plot down: 0.1–0.4 ms. Pentagrams show the best parameter set for each generation.

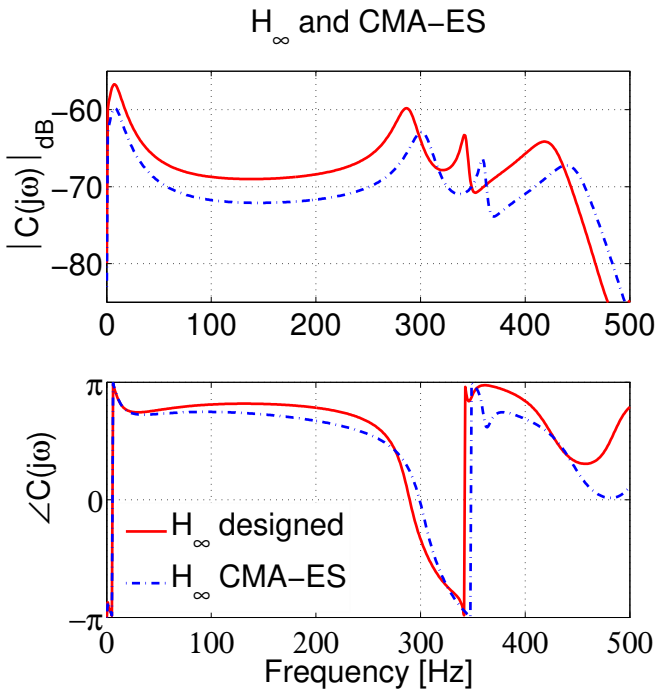

Fig. 13. Bode plots for the designed and the UH-CMA-ES-optimized $\mathcal{H}_\infty$ controllers for $\lambda = 1.875$. Amplitude (top) and phase (bottom) of the controller.

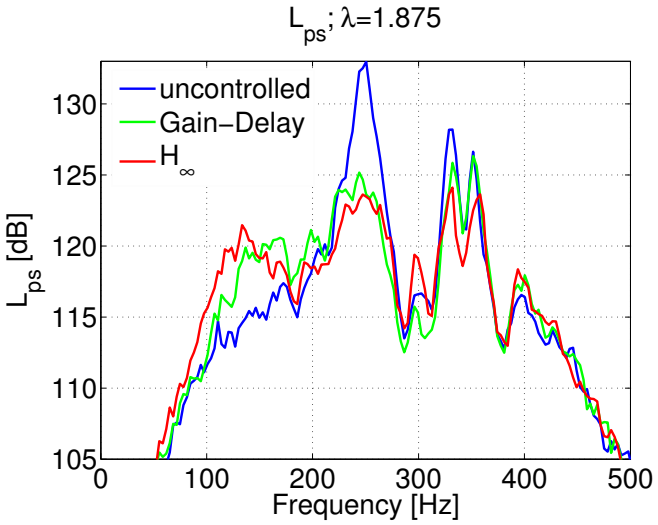

Fig. 14. Comparison of the pressure spectra when the plant is uncontrolled, Gain-Delay and $\mathcal{H}_\infty$ controlled. Both controllers are UH-CMA-ES optimized, $\lambda = 1.875$.

gain interval $[0.4, 1.1]$), and the cost function landscape at the bottom. A lower gain decreases L_{eq} compared with the designed controller with gain one, but the frequency shift does not have a decisive effect. The final standard deviations for gain and frequency shift differ by a factor of about three (lower left of the top plot of Fig. 15) reflecting the different sensitivities of the parameters. The Bode plots of the designed and the optimized controllers are shown in Fig. 16. The controller phase is quite flat and therefore tolerant against frequency shifts. The combustor pressure spectrum exhibits only one very distinct peak, and it suffices to provide the right amount of gain and phase at this frequency.

To compare the UH-CMA-ES optimized Gain-Delay and $\mathcal{H}_\infty$ controllers with the uncontrolled plant, their spectra are plotted in Fig. 17, the values of L_{eq} are 159.87 dB, 147.48 dB and 147.35 dB, respectively. They are shown for the plant which has been running for several hours and is thus heated up. In this case, the $\mathcal{H}_\infty$ controller performs only slightly better than the Gain-Delay controller, but the control signal contains about 10% less energy.

VII. Summary and Outlook

We have presented a novel evolutionary optimization algorithm (UH-CMA-ES) for problems with uncertainties. The optimization algorithm consists of the well-known CMA-ES enhanced by a novel uncertainty handling algorithm. The evolutionary algorithm is applied to the online optimization of feedback controllers in a combustor test rig. The uncertainties are associated with the stochastic nature of the cost function and, in the present application, with the online optimization of the controller parameters.

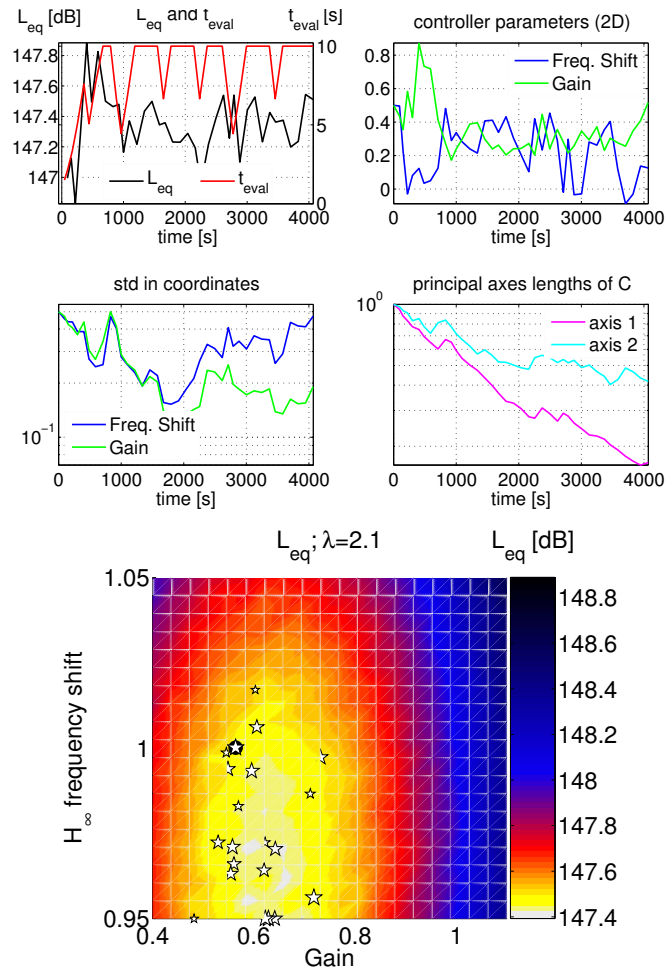

Fig. 15. Parameter evolution for the UH-CMA-ES optimization of an $\mathcal{H}_\infty$ controller for $\lambda = 2.1$ (top) and resulting cost function landscape (bottom). Pentagrams show the best parameter set for each generation.

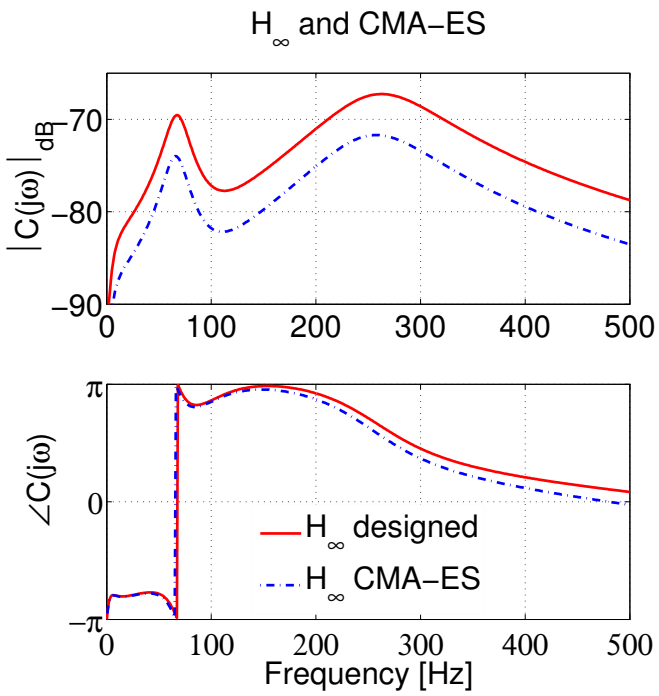

Fig. 16. The designed and the UH-CMA-ES-optimized $\mathcal{H}_\infty$ controller for $\lambda = 2.1$.

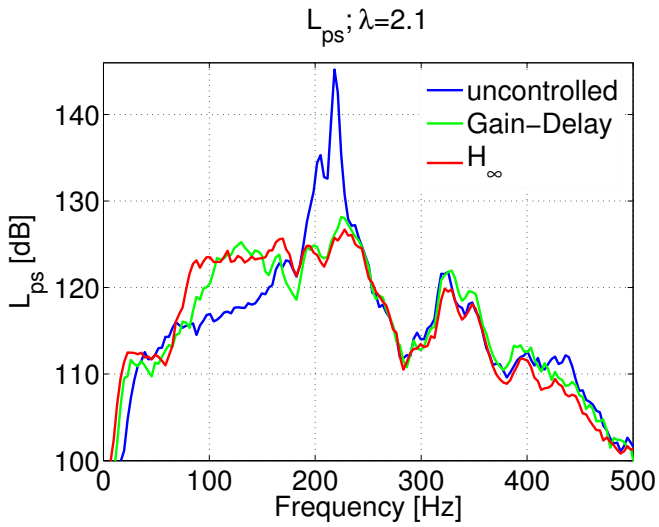

Fig. 17. Comparison of the uncontrolled with the Gain-Delay and $\mathcal{H}_\infty$ controlled plant for $\lambda = 2.1$. Both controllers are optimized by UH-CMA-ES.

The novel uncertainty-handling algorithm needs few additional cost function evaluations per generation and is therefore well suited for online applications. The algorithm distinguishes between uncertainty measurement and uncertainty treatment. The uncertainty measurement is based on rank changes induced by reevaluations of solutions. Two adaptive uncertainty treatments are proposed: increasing the time for evaluating the controller up to a prescribed bound, and increasing the population diversity. Both treatments improve the signal-to-noise ratio. The former reduces the uncertainty variance comparable to resampling of solutions, while the latter improves the signal term without additional function evaluations and with only minor computational effort.

The algorithm has been validated on test functions and it has been applied to the optimization of feedback controllers of thermoacoustic instabilities, using secondary fuel injection in a combustor test rig. The controllers employ Gain-Delay and $\mathcal{H}_\infty$ control and their parameters have been optimized online with the introduced UH-CMA-ES. The experiments show that the algorithm can optimize different controller types and can cope with changing operating conditions and high levels of uncertainty. Our results indicate that model-based $\mathcal{H}_\infty$ controllers perform best, and that they can be further improved through the use of the UH-CMA-ES. The optimized solutions deviate significantly from the originally designed solutions and can make up for uncertainties in the model-building and design process, as well as for time-varying plant characteristics.

Future work will include the acceleration of the self-tuning process for the combustion control. First, algorithm internal parameter settings can be improved and be specifically adjusted to the small dimensionality of the problem. Second, the implementation on a test rig can be improved to shorten the ramping times which are by far the most time consuming part in the initial phase of the controller tuning process. Furthermore in the context of the UH-CMA-ES a more informed way of selecting the appropriate uncertainty treatment can shorten the adaptation time considerably, in particular if the evaluation time can be reduced for a longer time interval.

VIII. Acknowledgments

Support by Daniel Fritsche during the experimental phases and assistance by Caroline Metzler for the technical illustrations is gratefully acknowledged. The authors thank David Charypar for the valuable suggestions and Stefan Kern for providing supporting data. Fruitful discussions with Bruno Schuermans and financial support from ALSTOM (Switzerland) Ltd. are gratefully acknowledged.

References

[1] M. Ahmad, L. Zhang, and J. Readle. On-line genetic algorithm tuning of a pi controller for a heating system. In Proceedings of GALESIA—genetic algorithms in engineering systems: innovations and applications, pages 510–515, 1997.

[2] A. Aizawa and B. Wah. Scheduling of genetic algorithms in noisy environment. Evolutionary Computation, pages 97–122, 1994.

[3] A. M. Annaswamy and A. F. Ghoniem. Active control of combustion instability: Theory and practice. IEEE Control Systems Magazine, 22(6):37–54, 2002.

[4] D. Arnold. Weighted multirecombination evolution strategies. Theoretical Computer Science, 1(361):18–37, 2006.

[5] D. V. Arnold. Noisy Optimization with Evolution Strategies, volume 8. Kluwer, Boston, 2002.

[6] D. V. Arnold and H.-G. Beyer. Local performance of the $(\mu/\mu_i, \lambda)$-es in a noisy environment. In W. Martin and W. Spears, editors, Foundations on Genetic Algorithms FOGA, pages 127–142. Morgan Kaufmann, 2000.

[7] A. Auger and N. Hansen. Performance evaluation of an advanced local search evolutionary algorithm. In Proceedings of the IEEE Congress on Evolutionary Computation, 2005.

[8] A. Banaszuk, K. B. Ariyur, M. Krstić, and C. A. Jacobson. An adaptive algorithm for control of combustion instability. Automatica, 40(11):1965–1972, 2004.

[9] H.-G. Beyer. Toward a Theory of Evolution Strategies: Some Asymptotical Results from the $(1,^+\lambda)$-Theory. Evolutionary Computation, 1(2):165–188, 1993.

[10] H.-G. Beyer. Evolutionary Algorithms in Noisy Environments: Theoretical Issues and Guidelines for Practice. Computer Methods in Applied Mechanics and Engineering, 186(2–4):239–267, 2000.

[11] H.-G. Beyer and D. Arnold. Qualms regarding the optimality of cumulative path length control in CSA/CMA-evolution strategies. Evolutionary Computation, 11(1):19–28, 2003.

[12] H.-G. Beyer and K. Deb. On self-adaptive features in real-parameter evolutionary algorithms. IEEE Transactions on Evolutionary Computation, 5(3):250–270, 2001.

[13] R. Blonbou, A. Laverdant, S. Zaleski, and P. Kuentzmann. Active adaptive combustion control using neural networks. Combustion Science and Technology, 156:25–47, 2000.

[14] J. Branke. Creating robust solutions by means of evolutionary algorithms. In A. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, Parallel Problem Solving from Nature—PPSN V, Proceedings, pages 119–128. Springer, Berlin, 1998.

[15] J. Branke and C. Schmidt. Selection in the presence of noise. In E. Cantu-Paz, editor, Genetic and Evolutionary Computation Conference (GECCO'03), pages 766–777. LNCS 2273, Springer, JUL 2003.

[16] J. Branke and C. Schmidt. Selection in the presence of noise. In G. Goos, J. Hartmanis, and J. van Leeuwen, editors, Genetic and Evolutionary Computation Conference, GECCO, Proceedings, pages 766–777. Springer, 2003.

[17] J. Branke, C. Schmidt, and H. Schmeck. Efficient fitness estimation in noisy environments. In Genetic and Evolutionary Computation Conference (GECCO). Morgan Kaufmann, 2001.

[18] E. Cantu-Paz. Adaptive sampling for noisy problems. In K. e. a. Deb, editor, Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2004), pages 947–958, Berlin Heidelberg, 2004. Springer.

[19] A. Chipperfield and P. Fleming. Multiobjective gas turbine engine controller design using genetic algorithms. IEEE Transactions on Industrial Electronics, 43(5):583–587, 1996.

[20] N. V. Dakev, J. F. Whidborne, A. J. Chipperfield, and P. J. Fleming. Evolutionary H-infinity design of an electromagnetic suspension control system for a maglev vehicle. Proceedings of the Institution of Mechanical Engineers Part I- Journal of Systems and Control Engineering, 211(5):345–355, 1997.

[21] A. P. Dowling and A. S. Morgans. Feedback control of combustion oscillations. Annual Review Of Fluid Mechanics, 37:151–182, 2005.

[22] S. Evesque. Adaptive Control of Combustion Oscillations. PhD thesis, University of Cambridge, 2000.

[23] S. Evesque, A. M. Annaswamy, S. Niculescu, and A. P. Dowling. Adaptive control of a class of time-delay systems. Journal of Dynamic Systems Measurement and Control-Transactions of the ASME, 125(2):186–193, 2003.

[24] S. Evesque, A. P. Dowling, and A. M. Annaswamy. Self-tuning regulators for combustion oscillations. Proceedings of the Royal Society of London Series A- Mathematical Physical and Engineering Sciences, 459(2035):1709–1749, 2003.

[25] P. J. Fleming and R. C. Purshouse. Evolutionary algorithms in control systems engineering: a survey. Control Engineering Practice, 10(11):1223–1241, 2002.

[26] U. Hammel and T. Bäck. Evolution strategies on noisy functions: How to improve convergence properties. In Y. Davidor, H.-P. Schwefel, and R. Männer, editors, Proceedings of theInternational Conference on Evolutionary Computation, theThirdConference on Parallel Problem Solving from Nature(PPSN III), volume 866, pages 159–168, Jerusalem, 9–14 1994. Springer.

[27] N. Hansen. Verallgemeinerte individuelle Schrittweitenregelung in der Evolutionsstrategie. Mensch und Buch Verlag, Berlin, 1998.

[28] N. Hansen and S. Kern. Evaluating the CMA evolution strategy on multimodal test functions. In X. Yao et al., editors, Parallel Problem Solving from Nature - PPSN VIII, LNCS 3242, pages 282–291. Springer, 2004.

[29] N. Hansen, S. D. Müller, and P. Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (CMA-ES). Evolutionary Computation, 11(1):1–18, 2003.

[30] N. Hansen and A. Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In Proceedings of the 1996 IEEE Conference on Evolutionary Computation (ICEC '96), pages 312–317, 1996.

[31] N. Hansen and A. Ostermeier. Completely derandomized self-adaptation in evolution strategies. Evolutionary Computation, 9(2):159–195, 2001.

[32] G. Harik, E. Cantu-Paz, D. E. Goldberg, and B. L. Miller. The gambler's ruin problem, genetic algorithms, and the sizing of the populations. Evolutionary Computation, 7:231–253, 1999.

[33] Y. Jin and J. Branke. Evolutionary optimization in uncertain environments—a survey. IEEE Transactions on Evolutionary Computation, 9(3):303–318, 2005.

[34] S. Kern, N. Hansen, and P. Koumoutsakos. Local meta-models for optimization using evolution stratgies. In T. P. Runarsson et al., editors, Parallel Problem Solving from Nature—PPSN IX, Proceedings, pages 939–948. Springer, 2006.

[35] M. Krstić, A. Krupadanam, and C. Jacobson. Self-tuning control of a nonlinear model of combustion instabilities. IEEE Transactions on Control Systems Technology, 7(4):424–436, 1999.

[36] T. L. Lai, H. Robbins, and K. F. Yu. Adaptive choice of mean or median in estimating the center of a symmetric distribution. Proc. Natl. Adac. Sci., 80(18):5803–5806, 1983.

[37] A. H. Lefebvre. Gas turbine combustion. Taylor and Francis, Philadelphia, 2nd edition, 1999.

[38] T. Lieuwen and V. Yang. Combustion Instabilities in Gas Turbine Engines: Operational Experience, Fundamental Mechanisms, And Modeling. Progress in Astronautics and Aeronautics, Vol. 210. AIAA, 2005.

[39] D. A. Linkens and H. O. Nyongesa. Genetic Algorithms For Fuzzy Control.1. Offline System-Development And Application. Iee Proceedings-Control Theory And Applications, 142(3):161–176, 1995.

[40] D. A. Linkens and H. O. Nyongesa. Genetic Algorithms For Fuzzy Control.2. Online System-Development And Application. Iee Proceedings-Control Theory And Applications, 142(3):177–185, 1995.

[41] L. Ljung. System Identification, Theory for the User. Prentice Hall, 2^{nd} edition, 1999.

[42] S. Markon, D. V. Arnold, T. Baeck, T. Beielstein, and H.-G. Beyer. Thresholding - a selection operator for noisy ES. In Proceedings of the 2001 Congress on Evolutionary Computation CEC2001, pages 465–472, COEX, World Trade Center, 159 Samseong-dong, Gangnam-gu, Seoul, Korea, 27-30 2001. IEEE Press.

[43] H. Moriyama and K. Shimizu. On-line optimisation of culture temperature for ethanol fermentation using a genetic algorithm. Journal of Chemical Technology and Biotechnology, 66(3):217–222, 1996.

[44] A. S. P. Niederberger, B. B. H. Schuermans, and L. Guzzella. Modeling and active control of thermoacoustic instabilities. In 16th IFAC World Congress, Prague, 2005.

[45] N. S. Nise. Control systems engineering. Bejamin/Cummings Pub. Co., Redwood City, Calif., 2nd edition, 1995. Norman S. Nise. ill.; 24 cm.

[46] C. O. Paschereit and E. Gutmark. Proportional control of combustion instabilities in a simulated gas-turbine combustor. Journal of Propulsion and Power, 18(6):1298–1304, 2002.

[47] C. O. Paschereit, E. Gutmark, and W. Weisenstein. Coherent structures in swirling flows and their role in acoustic combustion control. Physics of Fluids, 11(9):2667–2678, 1999.

[48] C. O. Paschereit, B. B. H. Schuermans, and D. Büche. Combustion process optimization using evolutionary algorithm. In 2003 ASME Turbo Expo, volume 2, pages 281–291, Atlanta, USA, 2003. ASME, International Gas Turbine Institute.

[49] C. O. Paschereit, B. B. H. Schuermans, and D. U. Campos Delgado. Active combustion control using an evolution algorithm. In 39th Aerospace Sciences Meeting and Exhibit, Reno, USA, 2001.

[50] L. Rayleigh. The explanation of certain acoustical phenomena. Nature, 18:319:321, 1878.

[51] I. Rechenberg. Evolutionsstrategie &94. Frommann-Holzboog, 1994.

[52] A. J. Riley, S. Park, A. P. Dowling, S. Evesque, and A. M. Annaswamy. Adaptive closed-loop control on an atmospheric gaseous lean-premixed combustor. In 2003 ASME Turbo Expo, volume 2, pages 347–358, Atlanta, USA, 2003. ASME, International Gas Turbine Institute.

[53] A. J. Riley, S. Park, A. P. Dowling, S. Evesque, and A. M. Annaswamy. Advanced closed-loop control on an atmospheric gaseous lean-premixed combustor. Journal Of Engineering For Gas Turbines And Power-Transactions Of The ASME, 126(4):708–716, 2004.

[54] Y. Sano and H. Kita. Optimization of noisy fitness functions by means of genetic algorithms using history of search. In M. Schoenauer, K. Deb, G. Rudolph, X. Yao, E. Lutton, J. J. M. Guervós, and H.-P. Schwefel, editors, PPSN, volume 1917 of Lecture Notes in Computer Science, pages 571–580. Springer, 2000.

[55] B. B. H. Schuermans. Modeling and Control of Thermoacoustic Instabilities. PhD thesis, Ecole Polytechnique Fédérale de Lausanne, Switzerland, 2003.

[56] S. Skogestad and I. Postlethwaite. Multivariable Feedback Control: Analysis and Design. John Wiley and Sons Ltd., Chichester, New York, 1996.

[57] J. Spall. Introduction to Stochastic Search and Optimization. John Wiley & Sons, Inc. New York, NY, USA, 2003.

[58] P. Stagge. Averaging efficiently in the presence of noise. In A. Eiben, T. Bäck, M. Schoenauer, and H.-P. Schwefel, editors, Parallel Problem Solving from Nature—PPSN V, Proceedings, pages 109–118, Amsterdam, 1998. Springer, Berlin.

APPENDIX

A. Matlab Implementation of the Uncertainty Measurement

```
function [s ranks rankDelta] = uncertaintymeasurement(arf1, arf2, lamreev, theta)
%
% Input:
%   arf1, arf2 : two 1xlambda arrays of function values, two values for
%       each individual of the population. The first lamreev values in
%       arf2 are new (re-)evaluations of the respective individual.
%    lamreev: number of reevaluated individuals in arf2
%    theta : parameter theta for the rank change limit
%
% Using: prctile function from statistics toolbox.
%
% Output:
%   s : uncertainty measurement, s>0 means the uncertainty measure is above the
%       acceptance threshold
%   ranks : 2xlambda array of ranks of arf1 and arf2 in the set
%       [arf1 arf2], values are in [1:2*lambda]
%   rankDelta: 1xlambda array of rank movements of arf2 compared to
%       arf1.  rankDelta(i) agrees with the number of values from
%       [arf1 arf2] that lie between arf1(i) and arf2(i).

%%% verify input argument sizes
if size(arf1,1) ~= size(arf2,1)
  error('arf1 and arf2 must agree in size 1');
elseif size(arf1,2) ~= size(arf2,2)
  error('arf1 and arf2 must agree in size 2');
elseif size(arf1,1) ~= 1
  error('arf1 and arf2 must be an 1xlambda array');
end
lam = size(arf1,2);

%%% compute rank changes into rankDelta
% compute ranks
[ignore idx] = sort([arf1 arf2]);
[ignore ranks] = sort(idx);
ranks = reshape(ranks, lam, 2)';

rankDelta = ranks(1,:) - ranks(2,:) - sign(ranks(1,:) - ranks(2,:));

%%% compute rank change limits using both ranks(1,...) and ranks(2,...)
for i = 1:lamreev
  sumlim(i) = ...
      prctile(abs((1:2*lam-1) - (ranks(1,i) - (ranks(1,i)>ranks(2,i)))), ...
              theta*50) ...
      + prctile(abs((1:2*lam-1) - (ranks(2,i) - (ranks(2,i)>ranks(1,i)))), ...
                theta*50);
end

%%% compute measurement
s = mean(2*abs(rankDelta(1:lamreev)) - sumlim);
```