Empirical study of the use of confidence level in RBDO with Monte Carlo simulations

D. Salazar, R. Le Riche, G. Pujol, X. Bay and O. Roustant

1 Ecole des Mines de Saint-Etienne
Département G2I/3MI
{leriche,bay,roustant}@emse.fr

2 CNRS UMR 5146

Abstract — The problem of designing in the presence of uncertainty is studied in this Chapter from the model conception to the numerical resolution using a very simple evolutionary algorithm known as (1+1)-ES embedding a Monte Carlo simulation loop. The optimization of a quality percentile is embraced as driving criterion for tackling designs processes in the face of uncertainty. The use of statistical hypothesis test is introduced as an additional safety level for ensuring good convergence. Several confidence levels are considered and tested with respect to different parameters settings for the optimizer and the Monte Carlo simulation strategy. Two academical functions and the design of a two-bars truss are used to analyze the behavior of the optimization algorithm.

Keywords — reliability based design optimization, robust optimization, percentile, Monte Carlo estimation, kriging.

1 Introduction

Engineers have a long history with accounting for uncertainties in the design process. They have first defined critical loadings and safety factors. More recently, uncertainty quantification is receiving a large attention from the engineering community because a finer characterization of the uncertainties is seen as an important performance reserve. In the presence of uncertainties, the performance of an individual system varies. Reliability based design optimization (RBDO) and robust optimization average out uncertainties by ultimately seeking solutions having the best statistical performance measure. In robust design for example, the sensitivity of the design to the uncertainties is a criterion to minimize. Reliability criteria typically aim at not falling, with a given confidence, below a minimal performance threshold. For RBDO, the percentile is a relevant statistical performance measure.

This chapter addresses reliability based optimization where the statistical performance criteria are estimated by Monte Carlo simulations. In particular, the objective function is expressed by a percentile (Section 2). The formulation is exemplified through a two-bars truss design problem in Section 3.

After bringing a short review of the statistic procedures needed for the empirical (Monte Carlo based) estimation of the percentile in Section 4, attention shall be paid to the optimization procedure in Section 5. The percentile estimator is embedded into an optimization loop and the effect of

*Now with Michelin
control the estimator accuracy and the parameters of the optimizer on the convergence speed and robustness is discussed in Section 6: a direct inclusion of the estimator in the optimizer is compared with a more careful hypothesis testing strategy. Finally, Section 7 brings the final remarks.

2 Accounting for uncertainties in optimization problem formulations

2.1 Control/noise parameterization

Let us denote by \( y \) the output of the numerical simulator of the designed system, and by \( \theta \) the parameters it depends on. It is customary to distinguish uncertainties bearing upon parameters from model uncertainties. Parametric uncertainties describes randomness or lack of knowledge on some of the parameters entering the simulation, while model uncertainties refer to the approximations made when solving the physical model. This Chapter only considers parametric uncertainties which already represent a very large and important research domain.

The numerical simulator inputs, \( \theta \), stem from a pair of parameters: the design variables, controlled and deterministic, written as the \( x \) vector, and the noise variables, random, and grouped into the \( U \) vector. The numerical model is written \( y(\theta(x,U)) \), or more simply \( y(x,U) \) (see Figure 1).

Vector \( x \) is a numerical vector that can take values in a search space \( X \), typically of the type \( \prod_{i=1}^{d}[a_i,b_i] \). Vector \( U \) is a random vector defined on a probability space \( (\Omega,\mathcal{F},P) \). Without loss of generality, we assume that the probability law followed by \( U \) does not depend on \( x \). This assumption allows a separation between control (or design) and noisy variables.

**Example 1** A uncertain parameter \( \theta \) follows a Gaussian distribution. Its nominal value and its quality class, in other terms, its average and its standard deviation, are controlled. In such a case, \( x = (m,\sigma) \), and \( \theta = m + \sigma U \) with \( U \sim \mathcal{N}(0,1) \).

In practice, the following two cases are often seen.

1. Noisy control : \( \theta = x + U \). For example, \( x \) describes the nominal dimensions of an object, and \( U \sim \mathcal{N}(0,\sigma^2I) \) represents the manufacturing tolerance. This situation is common in reliability based design optimization (e.g., [1]).

2. Noise exogenous to the control : \( \theta = (x,U) \). For example, if \( x \) is, again, the dimensions of an object, \( U \) describes uncertain boundary conditions such as the loadings. This often occurs in robust optimization (e.g., [2]).

Note: The \( y(x,U) \) notation implies that the system variability can be explicitly given to the numerical simulator through the variable \( u \) (a realization of \( U(\omega) \)). Other problems where the variability is intrinsic to the simulator can mathematically be seen as a random processes \( Y(x) \). They are less common and not considered in this Chapter.

2.2 Including uncertainties in the optimization formulation

Now that uncertainties have been defined, they can be included in the optimization problem formulation.
Deterministic optimization problems are typically formulated as,

\[
\begin{cases}
\min_{\theta} f_i(\theta), & i = 1, \ldots, p \\
\text{where } \theta \in \Theta, \\
\text{such that } g_j(\theta) \leq 0, & j = 1, \ldots, q \\
\text{and } h_k(\theta) = 0, & k = 1, \ldots, r
\end{cases}
\]

where \( \Theta \) is the search space. \( f_i, g_j \) and \( h_k \) are called optimization criteria. In this Chapter, we will only consider single-objective problems (\( p = 1, f \) is a unique function). All optimization criteria can be seen as calculations that post-process the model outputs, \( y(\theta) \). Thus, \( f(\theta) \) is a shorthand notation for \( f(\theta, y(\theta)) \), and the same stands for other criteria. This notation tends to hide the numerical simulator, \( y \), which however is the most computationally consuming component of our methods. When tackling practical optimization problems, one should keep in mind that different simulators have different computational costs. For example, if the simulation behind the constraints (e.g., a finite elements software for calculating stresses and strains) is more expensive than that behind the objective function (e.g., a volume estimation), meta-modeling of the constraints should be considered.

Uncertainties propagate to the optimization criteria, \( f, g_j \) and \( h_k \). They are typically taken into account via statistics: average, variance, . . .

### 2.2.1 Objective function and uncertainties

The most common statistics for the objective function are:

- the average,
  \[
  m(x) := E(f(x, U)) = \int_{\Omega} f(x, u) P_U(du)
  \]
- the variance,
  \[
  v(x) := \text{var}(f(x, U)) = E((f(x, U) - m(x))^2)
  \]
- the \( \alpha \)-th order percentile,
  \[
  q_\alpha(x) \text{ such that } P(f(x, U) \leq q_\alpha(x)) = \alpha
  \]

These criteria are minimized. The percentile, also known as value-at-risk in the field of quantitative finance, is a natural way to account for uncertainties: it guarantees that \( \alpha \) percent of the observed solution instances will perform at least as good as \( q_\alpha(x) \). This Chapter will therefore dwell on percentile minimization in later sections.

The optimization criteria are now completed to include the optimization constraints.

### 2.2.2 Inequality constraints

Let \( \mathcal{G}(x) \) be the random variable “all the inequality constraints are satisfied”,

\[
\mathcal{G}(x) = \bigcap_{j=1}^{q} \{ g_j(x, U) \leq 0 \}
\]

A naive approach would be to maximize the probability of satisfying the constraints,

\[
\max_x P(\mathcal{G}(x))
\]

because it is typically antinomic with performance criteria on the objective function. This can be observed on the two-bars truss of Section 3.
A common approach is to give a lower bound on the constraints satisfaction probability, the reliability $\beta$. The criterion (6) becomes an inequality constraint:

$$P(G(x)) \geq \beta$$

(7)

In the literature, there is often one such reliability level per criterion (e.g., [4], [3]):

$$P(g_j(x, U) \leq 0) \geq \beta_j$$

(8)

Note however that by doing so, the global system reliability (i.e., the probability that all the constraints are satisfied) is guaranteed only for independent $g_j(x, U)$’s.

Let us now go back to the criteria on the objective function (Equations 2, 3 and 4). These criteria include all the realizations $f(x, U(\omega))$ for $\omega \in \Omega$ irrespective of whether the constraints are satisfied or not. Yet, the optimization aims at minimizing the objective function provided that the constraints are satisfied. Conditional probabilities allow to include constraints satisfaction in the objectives definitions. The statistical objective criteria are conditioned on the event $G(x)$:

$$m_C(x) := E(f(x, U) | G(x))$$

(9)

$$v_C(x) := \text{var}(f(x, U) | G(x))$$

(10)

$$q_{\alpha}^C(x) \text{ tel que } P(f(x, U) \leq q_{\alpha}^C(x) | G(x)) = \alpha$$

(11)

Such conditioned criteria are more difficult to estimate and will not be used later on in this Chapter.

2.2.3 Equality constraints

Uncertainties are not compatible with “true” equality constraints because the event $h_k(x, U) = 0$ has zero probability. Most often, equality constraints are removed by solving them for some of the variables which are no longer considered in the optimization [3]. In optimal design, equality constraints have typically a physical meaning such as an equilibrium state. These equalities belong more to the model than to the optimization.

For example, an important class of problems is written

$$\begin{cases}
\min_{\theta} f(\theta, y(\theta)) \\
\text{where } \theta \in \Theta, \\
\text{such that } g_j(\theta, y(\theta)) \leq 0, \quad j = 1, \ldots, q
\end{cases}$$

(12)

where the model $y(\theta)$ is the solution of the equations

$$h_k(\theta, y(\theta)) = 0, \quad k = 1, \ldots, r.$$ 

(13)

This is notably the case of finite elements models. In solid mechanics, (13) is the equality of the internal and the external forces.

2.3 Robust design formulations

Robust optimization formulations add to the maximum performance objective a criterion for the stability of the solutions with respect to the uncertainties. The stability can, for example, be written using an upper bound on the performance variance. If the stability is as important as the performance, it can be treated as an objective, in a multi-objective fashion:

$$\begin{cases}
\min_{x} E(f(x, U)) \\
\text{et } \min_{x} \text{var}(f(x, U)) \\
\text{where } x \in X \\
\text{such that } P(G(x)) \geq \beta
\end{cases}$$

(14)
The average-variance trade off can be handled with a penalty factor $\lambda$. The criterion to minimize becomes (as in [5]):

$$
f_{\lambda}(x) = E(f(x, U)) + \lambda \sqrt{\text{var}(f(x, U))}
$$

If $f(x, U)$ is a Gaussian random variable, (15) is a percentile.

Non trivial robust optimization problems have opposed average and variance criteria. The two-bars truss example shown in this Chapter is not a real robust optimization problem because the improvement direction of the average and the variance is similar.

### 2.4 Reliability based design optimization formulation

RBDO ([13]) aims at guaranteeing a minimal performance level at the solution of the optimization, which implies the use of percentiles,

$$
\begin{align*}
\min_x q_\alpha(x)
\end{align*}
$$

where $x \in X$ such that $P(G(x)) \geq \beta$

Contrary to robust optimization, the performances at the optimum are not necessarily stable. Instabilities are accepted as long as they tend to improvements. This is the situation in the two-bars truss example, where the performance percentile and the reliability rate are typically antagonistic.

### 2.5 Remark: a posteriori uncertainty handling

We assume in this Chapter that the optimization problem is solved before the actual uncertainties occur. The best alternative $x^*$ is chosen without specifying which events $U(\omega_1), U(\omega_2), \ldots$ were observed (but with regards to the probability laws). This is the adequate context of optimal design.

But in other applications, such as optimum control and inverse problems, the uncertainty has been realized before the optimization problem is solved. The relevant formulation consist in solving the problem for a specific uncertainty ([7]),

$$
\begin{align*}
\min_{x} f(x, U(\omega))
\end{align*}
$$

where $x \in X$, such that $g_j(x, U(\omega)) \leq 0$, $j = 1, \ldots, q$

and $h_k(x) = 0$, $k = 1, \ldots, r$.

In this case, it is interesting to look for the law of the random vector $X^*(\omega)$, solution of the problem (17) for all $\omega \in \Omega$ (see for example Beauzamy, [6]). This is a difficult optimization problem in the general case, because it would imply solving sufficiently many optimization problems for the realizations of the uncertainty $U(\omega)$ in order to be able to estimate the law (or the moments) of $X^*(\omega)$. This problem has been approached using chaos polynomials [8].

### 3 Example: The two-bars test case

The two-bars test case [11] aims at designing a structure made of two cylindrical bars joining each other at their top as sketched in Figure 2.

A downwards force $F$ is applied at the structure’s top. The design variables are the bars diameter $d$ and length $L$. The objective is to minimize the total volume $V$. Buckling and strength failures make the formulation constraints:

$$
\begin{align*}
\min_{d,L} V \quad \text{such that} \quad s \leq s_{\text{max}} \quad \text{and} \quad s \leq s_{\text{crit}}
\end{align*}
$$
Figure 2: The two-bars test case. Left : the overall structure. Right : a bar cross-section.

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Nominal value</th>
<th>Std. Dev. (C.V.)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d$ bars diameter (mm)</td>
<td>20 – 80</td>
<td>1 (&lt; 5%)</td>
</tr>
<tr>
<td>$L$ bars length (mm)</td>
<td>800 – 1200</td>
<td>5 (&lt; 1%)</td>
</tr>
<tr>
<td>$F$ applied force (N)</td>
<td>150000</td>
<td>30000 (20%)</td>
</tr>
<tr>
<td>$E$ Young’s modulus (N/mm$^2$)</td>
<td>210000</td>
<td>21000 (10%)</td>
</tr>
<tr>
<td>$B$ half structure width (mm)</td>
<td>750</td>
<td>5 (&lt; 1%)</td>
</tr>
<tr>
<td>$T$ bar wall thickness (mm)</td>
<td>2.5</td>
<td>0.1 (4%)</td>
</tr>
<tr>
<td>$s_{\text{max}}$ Yield strength (N/mm$^2$)</td>
<td>400</td>
<td>-</td>
</tr>
</tbody>
</table>

Table 1: Two-bars truss parameters. The perturbations around the nominal values are Gaussian and all parameters are independent from each other. (C.V. = coefficient of variation = Std. Deviation / mean). Source: [10].

where $s$ is the bar’s stress, $s_{\text{max}}$ is the yield strength and $s_{\text{crit}}$ the maximum buckling load. The two-bars problem is simple enough to have analytical expressions for the objective function and the constraints:

$$V = 2\pi T d L$$

$$s = \frac{FL}{2\pi T d \sqrt{L^2 - B^2}}$$

$$s_{\text{crit}} = \frac{\pi^2 E (T^2 + d^2)}{8L^2}$$

The problem parameters are summed up in Table 1. The solution to the deterministic two-bars problem (18) is $d^* = 38 \text{ mm}$, $L^* = 964 \text{ mm}$, in which case $V^* = 0.5754 \text{ dm}^3$, $s^* = 399.9931$, $s_{\text{crit}}^* = 404.3131 \text{ N/mm}^2$.

The contour lines of the two-bars deterministic design criteria are plotted in Figure 3.

In the two-bars test case, the model parameters are $\theta = (d, L, F, E, B, T)$. The design variables are the nominal values of the $d$ and $L$ parameters, denoted $x = (\bar{d}, \bar{L})$. The uncertainties not only affect the design variables but also other parameters. Vector $U = (\varepsilon_d, \varepsilon_L, \varepsilon_F, \varepsilon_E, \varepsilon_B, \varepsilon_T)$ is in the two-bars truss a random vector. Its components are independent and Gaussian and their statistical properties are given in Table 1. The application $\theta(x, U)$ is defined by

$$d = \bar{d} + \varepsilon_d, \quad L = \bar{L} + \varepsilon_L, \quad F = \varepsilon_F, \quad E = \varepsilon_E, \quad B = \varepsilon_B, \quad T = \varepsilon_T .$$

A natural formulation of the two-bars truss design (18) with uncertainties is

$$\min_{x=(d,L)} q_\alpha(x) \quad \text{such that} \quad p(x) \geq \beta ,$$

where $q_\alpha$ is the $\alpha$-th order percentile of the volume and $p$ is the reliability, i.e., the probability of satisfying the constraints:

$$P(V(x, U) \leq q_\alpha(x)) = \alpha \quad \text{and} \quad p(x) = P\big( s(x, U) \leq s_{\text{max}} \text{ and } s(x, U) \leq s_{\text{crit}}(x, U) \big) .$$
Figure 3: Contour lines of the two-bars structure volume $V$ (in dm$^3$). The discontinuous lines correspond to the constraints critical state (dashes : $s = s_{\text{max}}$, dots : $s = s_{\text{crit}}$). Failure occurs at the left of the lines. The optimum ($d^* = 38, L^* = 964 \text{ mm}; V^* = 0.5754 \text{ dm}^3; s^* = 399.9931, s^*_{\text{crit}} = 404.3131 \text{ N/mm}^2$) is shown as a dot.

4 Monte Carlo estimation of the design criteria

The simplest approach to account for uncertainties in an optimization problem consists in performing the Monte Carlo statistical estimations inside the optimization loop, each time the objective function or the constraints are needed. Such an approach typically has a very large numerical cost since the number of Monte Carlo simulations is multiplied by the number of analyses required by the optimizer. It can only be applied when the numerical cost of the simulator is very low, e.g., an analytical formula. At present, we do not discuss the inefficiency of such an implementation, but we review the main results about Monte Carlo estimation for later use in the chapter. Monte Carlo simulations are general and can be applied without modifying the simulator $y(\theta(x, U))$.

4.1 Crude versus Latin Hypercube Sampling

Put in simple words, Monte Carlo simulation consists in generating sequence on several states or samples for an input vector, using the cumulative distribution functions (CDF) associated to each of its components. Afterwards the input samples are evaluated by the simulator and the resulting values are used to estimate some relevant facts about the output, like its probability distribution, its central moments, etc. As the use of Monte Carlo requires generating a set of many samples of the input, the simulation can be implemented in such a way that every sample in a set is drawn regardless the value of the other samples drawn before. This simple procedure is known as Crude Monte Carlo (CMC) and represents at the same time the easiest and more inefficient possible implementation.

A more sophisticated alternative is the Latin Hypercube Sampling (LHS), which divides the sampling space into a hyper-matrix of $m$ dimensions (corresponding to the number of input variables), each one divided into $n$ non-overlapping strata of probability density ($1/n$). The idea is
to draw \( n \) samples of the input vector in such a way that each strata of each input variable’s CDF is sampled only once. This way the over representation of some regions of the sampling space is avoided or at least reduced.

Both methods are useful to estimate the parameters described in the next sections. Nevertheless, the computational effort needed for having certain level of quality is in general, for the LHS, at most as high as that needed for the CMC.

4.2 Pointwise Estimators

Regarding the two-bars truss example, let us write \( v_i, s_i, s_{\text{max}}^i \) and \( s_{\text{crit}}^i \), \( i = 1, \ldots, n \), the \( i \)-th volume, stress, maximum stress and critical buckling load samples, respectively, obtained at a given \( x \). The empirical percentile and reliability estimators are [15]

\[
\hat{q}_\alpha = v^\lfloor n\alpha \rfloor \tag{21}
\]

and

\[
\hat{p} = \frac{1}{n} \sum_{i=1}^{n} I(s^i \leq s_{\text{max}}^i \text{ and } s^i \leq s_{\text{crit}}^i) \tag{22}
\]

where \( v^\lfloor n\alpha \rfloor \) is the \( \lfloor n\alpha \rfloor \)th smallest \( v^i \) value (\( \lfloor . \rfloor \) means lower rounded integer).

4.3 Batch Estimators

If the variances of the estimators introduced in the previous section are needed, an auxiliary procedure should be implemented in order to assess the variance of the empirical percentile. Indeed, the variance of \( \hat{p} \) can be estimated as \( \sqrt{\frac{\hat{p}(1-\hat{p})}{n}} \), but the variance of \( \hat{q}_\alpha \) requires having more than one estimation in order to apply the unbiased variance estimator

\[
s_q^2 = \frac{1}{n_b-1} \sum_{i=1}^{n_b} (\bar{q}_\alpha - \hat{q}_\alpha)^2 \tag{23}
\]

where \( \bar{q}_\alpha \) is the \( \lfloor n\alpha \rfloor \)th smallest \( v^i \) value (\( \lfloor . \rfloor \) means lower rounded integer).

4.4 Confidence Interval

For the percentile \( q_\alpha \), a confidence interval can be obtained either by parametric or non-parametric statistics. For the first case it is necessary to apply the batch procedure shown in the previous section. The confidence interval is then calculated as:

\[
q_\alpha = \left[ \bar{q}_\alpha - z_{\gamma/2} \frac{s_q}{\sqrt{n_b}}, \bar{q}_\alpha + z_{\gamma/2} \frac{s_q}{\sqrt{n_b}} \right] \tag{24}
\]

with probability \( 1 - \gamma \)

where the statistic \( z_{\gamma/2} \) is the \( 1 - \gamma/2 \) percentile of a \( \mathcal{N}(0, 1) \).

The parametric calculation just presented relies on the assumption that \( n_b \) is large enough to accept that \( q_\alpha \) is normally distributed. Instead if \( n_b \) is small, say < 20, the interval can be built calculating the statistic \( z_{\gamma/2} \) through a Student distribution with \( n_b - 1 \) d.o.f.

A non-parametric approach is yet possible through the Wilks formula:

\[
P(q_\alpha \in [v^{(i_1)}, v^{(i_2)}]) = \sum_{k=0}^{i_2-i_1} C_{n_b}^k \alpha^k (1 - \alpha)^{n_b-k} \tag{24}
\]
To use this formula, one looks for the closest indexes $i_1$ and $i_2$ with $i_1 \leq \lfloor n\alpha \rfloor \leq i_2$ such that the sum (24) is larger or equal to the confidence level $1 - \gamma$. In the two bars example, $\alpha$ is equal to 0.9, then e.g. for $n = 100$ the $[v(87), v(95)]$ interval has a confidence level larger than 0.8 of containing $q_\alpha$.

With regard to $p$, for $n$ large (typically $np$ and $n(1-p)$ larger than 5), a confidence interval is given by

$$ p \in \left[ \hat{p} - z_{\gamma/2} \sqrt{\frac{p(1-p)}{n}}, \hat{p} + z_{\gamma/2} \sqrt{\frac{p(1-p)}{n}} \right] \quad \text{with probability } 1 - \gamma $$

(25)

Notice that the difficulty in estimating $p$ for given precision depends on $p$ because the estimator’s variance is proportional to $p(1-p)$. For the high reliability values we are interested in ($\geq 90\%$), $p$ is easy to estimate.

4.5 Statistical Hypothesis Testing

A statistical hypothesis test (HT) is a decision-making method from inferential statistics for discerning between two mutually exclusive postulates, called null hypothesis ($H_0$) and alternative hypothesis ($H_1$), such that the selection of either of such postulates is made by statistical criteria. The underpinning idea is that the default -null- hypothesis should not be rejected unless the weight of the evidence is significantly large -in statistical terms- to do so.

The hypotheses make statement about the observed values of some statistics calculated via sampling, with regard to the expected values of such statistics, according to the characteristics of the involved populations. For instance, having the sample mean value $\bar{x}$, one asks if the population mean $\mu_x$ can be consider equal to a particular value $\mu$; or using the HT nomenclature $H_0: \mu_x = \mu; H_1: \mu_x \neq \mu$.

In the HT four situations are possible and only one is true:

<table>
<thead>
<tr>
<th>Decision</th>
<th>True case</th>
</tr>
</thead>
<tbody>
<tr>
<td>$H_0$</td>
<td>$1 - \alpha$</td>
</tr>
<tr>
<td>$H_1$</td>
<td>$\alpha$</td>
</tr>
</tbody>
</table>

Two types of wrong decisions may occur in the HT. The first type amounts to a wrong rejection of $H_0$ given that $H_0$ is true. The probability of incurring in this error is $\alpha$; this value is called significance level. Its reciprocate $1 - \alpha$ is known as confidence level and its value is chosen by the test user. Typical figures for the confidence level are 90% and 95% which correspond to significance levels of 10% and 5% respectively. On the other hand, the second type of error or the probability of failing to reject $H_0$ is given by $\tilde{\beta}$. While $\alpha$ is known, $\tilde{\beta}$ remains unknown in general; consequently the construction of the hypothesis tests aims at maximizing their power, i.e. the probability $1 - \tilde{\beta}$.

The mechanism behind the HT consists in defining the region of acceptance reducing the probability of a wrong rejection of $H_0$ to $\alpha$. Hence a test statistic $T$ is calculated for the test. If its value lies outside the acceptance region, or in other words if $T > c_\alpha$, where $c_\alpha$ is known as critical value or statistic of comparison, then $H_0$ is rejected; otherwise $H_0$ is accepted. The calculations of the statistic $T$ for the test and for the critical value depend on if some assumptions can be made about the underlying probability distribution of the data analyzed.

**Parametric hypothesis testing** In parametric hypothesis testing it is assumed the data can be modeled using a particular distribution. In consequence, both $T$ and $c_\alpha$ make reference to a known distribution. In our case we are concerned about composite mean test of the form:
Figure 4: Example of a normal distribution used for hypothesis testing: If the statistic $T$ lies in the dark area, $H_0$ is rejected with significance level $\alpha$.

$H_0 : \mu_1 < \mu_2$

$H_1 : \mu_1 \geq \mu_2$

where the interest is to determine if the mean value of a population $\mu_1$ is or not greater than that of another population $\mu_2$. The parametric test is built assuming a distribution for the standardized difference of sample means. Hence, the test takes the form [16]:

Accept $H_0 : \quad T = \frac{\bar{x}_1 - \bar{x}_2}{\sqrt{s_1^2/n_1 + s_2^2/n_2}} > c_{\alpha}$

Reject $H_0 : \quad \text{otherwise}$

where $\bar{x}_i$ is the sample mean value of the $i$-th sample, whereas $s_i^2$ and $n_i$ are their corresponding sample variance and size. The critical value $c_{\alpha}$ is substituted by $z_{\alpha}$ if the sample sizes are big enough (say $n_i > 15$ and $n_1 + n_2 > 30$) so that the Central Limit Theorem holds, allowing a normal approximation (see fig. 4). Otherwise $c_{\alpha} = t(\alpha, \nu)$, where $t(\alpha, \nu)$ represents the $1 - \alpha$ percentile of a Student distribution with $\nu$ d.o.f. In such a case $\nu$ is calculated as:

$\nu = \left(\frac{s_1^2/n_1 + s_2^2/n_2}{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2} \right)^{\frac{1}{2}}$

Additional precisions are necessary. If the population variances are known then $s_i^2$ should be replaced by $\sigma_i^2$. Besides, if the variances are assumed equal, then $\nu = n_1 + n_2 - 2$ and $T$ in eq. (24) reduces to:

$T = \frac{\bar{x}_1 - \bar{x}_2}{s_p \sqrt{1/n_1 + 1/n_2}}$

where

$s_p = \sqrt{\frac{(n_1 - 1)s_1^2 + (n_2 - 1)s_2^2}{n_1 + n_2 - 2}}$

Finally, the equality of variances can be checked by an F-test of the form $H_0 : \sigma_1^2 = \sigma_2^2$ and $H_1 : \sigma_1^2 \neq \sigma_2^2$. As the variance ratio $F = s_1^2/s_2^2$ is F-distributed, then $H_0$ is accepted if the $F \in [F(1-\alpha, n_1-1, n_2-2), F(\alpha, n_1-1, n_2-2)]$; otherwise $H_1$ is rejected in favor of $H_1$ [16].
Non-parametric hypothesis testing  The preceding test relies upon assumptions about the distribution of $T$. If such assumptions cannot be met but the weaker assumption of continuity of the involved distributions holds, it is still possible to build a hypothesis test for comparing two samples, namely $X = \{X_i\}_{i=1}^m$ and $Y = \{Y_j\}_{j=1}^n$, based on the idea that both samples come from the same distribution. Therefore, the sorted union of both ones is expected to exhibit an homogeneous distribution. With this in mind, a statistic based on the ranks of the observations inside the sorted union is defined:

$$T_U = \sum_{i=1}^m \sum_{j=1}^n \mathbb{I}(Y_j < X_i)$$

This statistic is the base of the Wilcoxon-Mann-Whitney (WMW) test. In particular, for the case of our concern, the test (26) based on mean comparisons is no longer possible, but it can substituted for one based on the medians $M_X$ and $M_Y$ of $X$ and $Y$ respectively, yielding

$$H_0 : M_X \leq M_Y$$
$$H_1 : M_X > M_Y$$

where $H_0$ is accepted if $T_U < c_\alpha$ and is rejected otherwise. The critical value $c_\alpha$ is the lower integer such that $P\{T_U \geq c_\alpha\} = \alpha$ (cf [17]). As $T_U$ is asymptotically normal, when $m, n > 5$ the normalized statistic is approximated with a normal function, and the critical point is calculated as [17]:

$$c_\alpha = \frac{mn}{2} + \frac{z_\alpha \sqrt{mn(m+n+1)}}{12}$$

5  A simple evolutionary optimizer for noisy functions: introducing the confidence level

The general procedure for solving RBDO problems as those formulated in section 2 consists in embedding a Monte Carlo simulation loop within the simulator that evaluates the objective function, conducting the optimization using mean and variance estimates, and less often percentiles or other statistics. This is particularly true for those optimizers based on ranking of alternatives, like the evolutionary algorithms or the simplex algorithm, the type of algorithms we are interested in in this Chapter.

Working with ranks in the face of uncertainty always entails the risk of misclassifying the alternatives, as the ranking based on Monte Carlo outputs is done using estimates and never the real values. In order to reduce such a risk the user can choose to increment the number of simulations used to calculate each estimation. Nevertheless, this is not always affordable, especially in the context of some industrial problems characterized by very time consuming objective functions. The use of meta-models for substituting the original function during the Monte Carlo simulation is a common turn around for this problem (this approach is illustrated in Chapter REF).

In this section we consider a different approach based on statistical hypothesis testing. As shown previously in section 4.5, by using HT’s to compare two estimations one can control the probability of rejecting an alternative with good chances to be equal or better than the other one, although their numerical values indicate the opposite. Thereby the notion of confidence level constitutes another control parameter for the optimizer.

To the best of our knowledge, only a very few studies from the literature incorporate the notion of confidence level into the optimization loop. In [19] the authors use the HT for testing statistical equality during a simulated annealing procedure. Later in a different study [21] the HT is used for favoring convergence of a genetic algorithm. Both studies concluded favorably toward the inclusion of the HT, albeit the experiments did not go in depth into the average effect of varying the parameters on their optimizers’ performances.
We show in this section how to implement the HT inside the optimization loop using a very simple evolutionary optimizer known as (1+1)-ES. The effect of having the HT as well as the other optimization parameters are studied later on in the subsequent sections.

5.1 The (1+1)-ES – an overview

Algorithm 1.10 describes the sequence of the simplest Evolution Strategy (ES) known (1+1)-ES. The initial N-dimensional point \( x^{(0)} \) as well as the step size \( \sigma \) and the maximal iteration budget \( t^{max} \) enter the algorithm as input. Then the main loop performs two basic operations until the stop criterion is reached: first, a new point \( x^{(t+1)} \) is produced generating an isotropic displacement (called mutation) around the current point \( x^{(t)} \). Finally the new point is tested against the current one, keeping the better (lower) one between them for another iteration.

**Algorithm 1**: (1+1)-ES minimization algorithm.

- **input**: Initial point \( x^{(0)} \in \mathbb{R}^N \), step size \( \sigma \in \mathbb{R}^+ \), iteration budget \( t^{max} \)
- **1.1** Initialize \( t := 0; \)
- **1.2** while \( t \leq t^{max} \) do
  - **1.3** for \( i := 1 \) to \( N \) do
  - **1.4** \( x_i^{(t+1)} := x_i^{(t)} + \sigma \mathcal{N}(0,1); \)
  - **1.5** end
  - **1.6** if not \( f(x^{(t+1)}) \leq f(x^{(t)}) \) then
    - **1.7** \( x^{(t+1)} := x^{(t)}; \)
  - **1.8** end
  - **1.9** \( t := t + 1; \)
- **1.10** end

For analyzing the performance of the (1+1)-ES, both theoretically and empirically, it is a common practice to make use of the sphere model \( G(||x-x^*||) \), a family of functions that only depend on the distance to the optimum \( x^* \). Such functions are usually denoted as \( G(R) \) where \( R = ||x-x^*|| \) is the radius to the optimum, making evident this way the isometric nature of this functions family.

In [20] the author develops analytical formulae for the progress rate of the (1+1)-ES using the sphere model, defined as \( \varphi = E \{ R^{(t)} - R^{(t+1)} \} \), the expected reduction of distance to the optimum, thereby providing theoretical means to predict the behavior of the algorithm. The progress rate in noisy environments has been studied in [20] and [18] for the (1+1)-ES and for a population-based ES denoted \((\mu/\mu, \lambda)\)-ES. In all of the cases, the analysis relies upon the assumption that the evaluation of the objective function can be modeled as \( \tilde{G}(R) = G(R) + \sigma_e(R)\xi \), where \( G(R) \) denotes the deterministic -exact- value of the objective function, \( \sigma_e(R) \) denotes the variance and \( \xi \) a random variable, thus yielding an additive noise dependent on the radius \( R \). It is generally assumed that \( \xi \) is a Gaussian noise \( \mathcal{N}(0,1) \), although [18] covers Cauchy and \( \chi^2 \) noises as well.

After assuming that \( \sigma_e \) can be considered constant after performing a new iteration, thus yielding \( \sigma_e(R^{(t+1)}) = \sigma_e(R^{(t)}) \), in [20] the author derives a progress rate expression for noisy optimization, based on the probability of accepting a new point, defined as

\[
p_a(R^{(t)}, R^{(t+1)}) = \Phi \left( \frac{G(R^{(t)}) - G(R^{(t+1)})}{\sqrt{2} \sigma_e(R)} \right)
\]

which in the particular case of normality and equality of variances yields

\[
p_a(R^{(t)}, R^{(t+1)}) = \Phi \left( \frac{G(R^{(t)}) - G(R^{(t+1)})}{\sqrt{2} \sigma_e(R)} \right)
\]
where $\Phi$ represents the CDF of a normal distribution.

Besides the fact of providing a theoretical understanding of the dynamics of the ES, having an expression for $\varphi$ has the advantage of making possible to estimate the optimal parameters for tuning the algorithm, in particular the value of the step size $\sigma$.\footnote{\cite{track}}

An expression for $\Phi$ that is better than $\Phi$ at the lowest evaluation cost. In such a case, the assumptions of variance equality and normality are hardly acceptable a priori, making the existing theoretical results not directly applicable. This last assertion is even more critical if the optimization with HT is intended to be modeled analytically, due to the existence of different versions of the parametric HT (see section 4.5). This completely justify an empirically approach to analyzing the performance of the (1+1)-ES with and without HT.

Let us consider in the next section the implementation of HT inside the (1+1)-ES using the confidence level $1 - \alpha$ as another control parameter for the optimization process.

### 5.2 Risk-taking vs. risk-avoiding (1+1)-ES formulations

The inclusion of the HT in the algorithm 1.10 is possible by replacing line 1.6 for a call to a function that executes the test and returns a boolean answer. A call to a Monte Carlo simulation is needed beforehand to estimate the statistics used in the HT. Denoting by $n$ the number of simulations taken, the expected value and the variance are estimated as $\bar{f}(x, U_n) = \frac{1}{n} \sum_{i=1}^{n} f(x, u_i)$ and $\sigma^2(f(x, U_n)) = \frac{1}{n-1} \sum_{i=1}^{n} (\bar{f}(x, U_n) - f(x, u_i))^2$ respectively, where $u_i$ is the $i$-th realization of the random vector $U$ and $U_n = \{u_1, u_2, \ldots, u_n\}$. The resulting sequence is shown in the algorithm 2.15.

#### Algorithm 2: (1+1)-ES minimization algorithm with Monte Carlo simulation and parametric HT.

<table>
<thead>
<tr>
<th>Line</th>
<th>Action</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.2</td>
<td>Initialize $t := 0$;</td>
</tr>
<tr>
<td>1.3</td>
<td>Generate $U_{n}^{(0)}$;</td>
</tr>
<tr>
<td>1.4</td>
<td>while $t \leq t_{\text{max}}$ do</td>
</tr>
<tr>
<td>1.5</td>
<td>for $i := 1$ to $N$ do</td>
</tr>
<tr>
<td>1.6</td>
<td>$x_{i}^{(t+1)} := x_{i}^{(t)} + \sigma \mathcal{N}(0, 1)$;</td>
</tr>
<tr>
<td>1.7</td>
<td>end</td>
</tr>
<tr>
<td>1.8</td>
<td>Generate $U_{n}^{(t+1)}$;</td>
</tr>
<tr>
<td>1.9</td>
<td>$Y^{(t+1)} := {\bar{f}(x^{(t+1)}, U_{n}^{(t+1)}), s^2(f(x^{(t+1)}, U_{n}^{(t+1)})), n}$;</td>
</tr>
<tr>
<td>1.10</td>
<td>$Y^{(t)} := {\bar{f}(x^{(t)}, U_{n}^{(t)}), s^2(f(x^{(t)}, U_{n}^{(t)})), n}$;</td>
</tr>
<tr>
<td>1.11</td>
<td>if not $HT(Y^{(t+1)}, Y^{(t)}, \alpha)$ then</td>
</tr>
<tr>
<td>1.12</td>
<td>$x^{(t+1)} := x^{(t)}$;</td>
</tr>
<tr>
<td>1.13</td>
<td>$U_{n}^{(t+1)} := U_{n}^{(t)}$;</td>
</tr>
<tr>
<td>1.14</td>
<td>end</td>
</tr>
<tr>
<td>1.15</td>
<td>$t := t + 1$;</td>
</tr>
</tbody>
</table>

The execution of Monte Carlo simulation is represented by a shorthand notation in lines 2.7 to 2.9 where the $U_{n}^{(t+1)}$ is first generated and used, along with $U_{n}^{(t)}$, to estimate, for $x^{(t)}$ and $x^{(t+1)}$, their corresponding sets comprising estimates of their means and variances and the sizes of the samples used for that task.
On the other hand, the HT is supposed to be implemented by means of a function \( HT(Y^{(t+1)}, Y^{(t)}, \alpha) \) which follows the logic given in (26), i.e. to test if the mean in \( Y^{(t+1)} \) is lower or equal than its counterpart in \( Y^{(t)} \), returning true if so and false otherwise. Such an implementation might consists in a parametric or a non parametric test. Both options are explained in section 4.5. It is worth to remark that the parametric test in (27) considers different situations according to the sample sizes and whether the variances can be considered statistically equal or not. In the preceding algorithm the sample size is always equal to \( n \)

The points, or on the one hand, adding samples to \( Y \); or reestimating \( Y \) from scratch after a certain number of generations, in the case of having the same current best \( x^{(t)} \) after a long while, are plausible options.

Notice that the formulation of the HT for line 2.10 in algorithm 2.15 establishes that the new mean value is better than older one as null hypothesis, in consequence the construction of the statistic \( T \) in (27) subtracts the mean value in \( Y^{(t)} \) to the mean value of \( Y^{(t+1)} \), thus yielding a negative value if \( \bar{f}(x^{(t+1)}, U_n^{(t+1)}) < \bar{f}(x^{(t)}, U_n^{(t)}) \). Observing figure 4 we realize that making the confidence level \( 1 - \alpha \) equal to 0.5 we force the test to accept the new observation \( x^{(t+1)} \) if its corresponding mean estimate \( \bar{f} \) is lower or equal than its counterpart. In other words, by fixing \( \alpha = 0.5 \) we get the same behavior of an optimizer without HT.

Now let us see the effect of setting \( \alpha \) to values different to the half. Following the preceding discussion we see that setting \( 1 - \alpha \) to any value in \((0.5, 1)\) (or conversely letting \( \alpha \in (0, 0.5) \)) amounts to accept the new point \( x^{(t+1)} \) even if sometimes the numerical value of its mean \( \bar{f} \) is is greater than its counterpart evaluated in \( x^{(t)} \). Such a procedure, that we can call risk-taking or “exploratory”, entails accepting to step back sometimes in the path toward the optimum in order to not lose eventually the good points with unfavorable Monte Carlo simulation values.

By contrast, if one wants to conduct the optimization under a risk-avoiding attitude, the current point \( x^{(t)} \) has to be preserved until the new point \( x^{(t+1)} \) was generated good enough to replace the former one. Such a “conservative” optimizer is possible by setting \( 1 - \alpha \) to values lower than 0.5. As a matter of fact, it is more adequate to reformulate the HT for that matter, setting the null hypothesis \( H_0 \) to \( E(f(x^{(t)}, U)) \leq E(f(x^{(t+1)}, U)) \) and \( H_1 \) to its complement and then performing the test for a confidence level \( 1 - \alpha \geq 0.5 \). Nevertheless, due to the symmetry of the PDF’s considered in the HT, it is easy to see that the formulation just described is equivalent to the original test proposed in (27) with confidence values lower than 0.5. Having realized that, we notice the advantage of having a unique algorithmic formulation whose “attitude” can be tuned by changing \( \alpha \) from the exploratory state \( (\alpha \in (0, 0.5)) \) to a traditional optimizer \( (\alpha = 0.5) \) and finally to a conservative state \( (\alpha \in (0.5, 1)) \).

### 6 Effects of the step size, the Monte Carlo budget and the confidence level on the ES convergence

In this section we analyze the effects of noise on the behavior of the (1+1)-ES and its treatment using confidence levels and different parameter settings for the optimizer and the Monte Carlo simulator. The experimentation comprises two academic functions with known optima. Later on, the two-bars truss example shall also be revisited and solved with this formulation. All the study turns around the RBDO formulation in (16).

All the experiments were conducted using Scilab and following the principles described in Chapter REF.
6.1 Empirical convergence study on academic test functions

6.1.1 Experimental setting

Two academic test functions are used to analyze the performance of the \( ES - (1 + 1) \). Both cases are instances of the sphere model described in section 5.1, adapted according to the percentile criterion for reliable designs introduced in section 2.4. The first instance here proposed corresponds to the uncertainty propagation case, and consists in the 90\% percentile of a quadratic function evaluated on a deterministic vector \( x \) affected by an aleatory vector \( U \). The resulting expression is

\[
F_Q(x, U) = q_{90\%} \left( ||(x + U)||^2 \right)
\]

(30)

On the other hand, the second function is based on a deterministic hyperbolic model affected by an additive noise \( U \), yielding

\[
F_H(x, U) = q_{90\%} \left( -\frac{1}{||x||^2 + 0.1} + U \right)
\]

(31)

For the sake of illustration and analytical calculus, in this section vector \( U \) is set to a Gaussian noise \( \mathcal{N}(0, 1) \) for all the experiments, so each realization of \( U \) is a vector of realizations of standard normal random variables. Notice that \( U \) has the same dimension of \( x \) for \( F_Q \) and is one dimensional for \( F_H \).

Both functions have their global optima in the null vector \( x^* = 0 \), with functional values \( F_Q^* = q_{90\%} \left( \chi^2_\nu \right) \), for \( \nu \) equal to the dimension of \( x \), and \( F_H^* = q_{90\%} \left( \mathcal{N}(0, 1) \right) = 1.2816 \). Figure 5 shows a view of the aforementioned functions for two-dimensional arguments.

For the experiments 30 runs were made for each setting. A robust 20\%-winsorized mean is used for reporting the graphic results of the academic tests unless we say otherwise. The total number of functional evaluations was set to \( 5 \times 10^5 \), which is actually seems a very large budget but is useful to reveal at what point the dynamic of the optimizer comes to a standstill. Moreover, the calculation of percentiles can be very demanding sometimes in terms of functional evaluations if the noise is to be reduced to very low levels.

6.1.2 Convergence study using deterministic functions

Let us consider now what are the effects of varying the step size \( \sigma \) and the dimensionality of the objective functions on algorithm 1.10 using deterministic versions of \( F_Q \) and \( F_H \). It is a well known phenomenon that the efficiency of the (1+1)-ES relies directly upon the selection of the step size \( \sigma \) and that such efficiency is also connected to the dimensionality. Hence, in order to illustrate the previous remark, the two functions were studied using different step sizes (\( \sigma = 0.05, 0.25, 1, 2, 4, 6 \))
for two-dimensional (2D) and ten-dimensional (10D) arguments. For the 2D case 30 runs were performed for both $F_Q$ and $F_H$, starting from $(5,5)$ and $(2,2)$ respectively. Noticing the radial character of the objective functions, the important feature is the initial distance to the optimum, which corresponds to $5\sqrt{2}$ and $2\sqrt{2}$ for the previous cases. In the 10D case, the initial radius is kept for the hyperbolic function and enlarged to $5\sqrt{10}$ for the quadratic function. The shape of the two functions for the 2D and the 10D case are represented in figure 6.

The results of the optimization using deterministic functions is shown in figure 7. We notice that larger step size exhibit a faster convergence at first, but this trend is quickly reverted due to the higher probability of making successful steps when approaching the optimum with small step sizes, thereby suggesting a clear hierarchy of performances, starting from the smallest step size ($\sigma = 0.05$) and growing incrementally until the larger one ($\sigma = 6.00$). Regarding the increment of dimensionality, we see how to convergence turns out harder. Larger step sizes exhibit even less chances to help the convergence, and in some cases, as for $\sigma = 6.00$ in figure 7(d), the optimizer simply remains stuck in the initial point. Summarizing we find that the smaller the step size and the dimensionality of the function’s argument the greater the accuracy and quality of the convergence, provided enough functional evaluations. This behavior, which is typical of the ES in deterministic cases, will serve as reference to analyze the combined effect of having uncertainty and introducing a confidence level in the ES performance. Finally, notice that, although the evaluation budget was said to be fixed to $5 \times 10^5$, the maximal number of functional evaluations (and points evaluated) in figure 7(d) was reduced to 12500, which corresponds to the maximal number of points evaluated in the next section. This way the comparisons between the deterministic and the noisy cases are easier.

### 6.1.3 Convergence study using noisy functions

Four basic effects are considered in this section, namely the effect of varying the confidence level, the effect of changing the step size, the influence of the number of Monte Carlo simulations and finally the effect of changing the simulation strategy from CMC to LHS. Regarding the confidence level, the values considered are $\alpha = 0.10, 0.50, 0.90$, which correspond to an exploratory, a traditional and a conservative optimizer respectively. The step sizes considered, on the other hand, are restricted to $\sigma = 0.05, 0.25, 0.50, 1.00, 2.00, 4.00, 6.00$. Besides the effect of the noise due to the simulations is also considered in two ways. Firstly the estimation of the percentiles is carried out via the batching procedure described in section 4.3, allowing five different values for the number of batches $n_b$, namely $n_b = 2, 5, 10, 25, 50$, although the graphical results do not present all these values for the sake of clarity. Secondly the efficiency of the simulation strategy in reducing noise is
Figure 7: Convergence of the (1+1)-ES using deterministic functions: from top to bottom the effect of changing the dimensionality from 2D to 10D on $F_Q$ (left column) and $F_H$ (right column) can be appreciated. Notice that small step sizes bear better convergence.

covered by benchmarking the CMC and the LHS for different combinations of the aforementioned parameters.

A technical remark is necessary at this point. The batching procedure described in Section 4.3 requires performing $s_b$ calculations for each batch, thus yielding $s_b \times n_b$ functional evaluations. The batch size $s_b$ is set to 20 for all the experiments shown hereafter. On the other hand, $n_b$ will be considered as the actual number of Monte Carlo simulations for estimating means and variances. With regards to algorithm 2.15, parameter $n$ is equivalent to $n_b$, whereas the assessment of each batch percentile remains hidden, although the evaluations are counted, so that each vector $x$ reduces the evaluation budget in $s_b \times n_b$ units. This way, the batch sizes $n_b = 2, 5, 10, 25, 50$ entail 40, 100, 200, 500 and 1000 functional evaluations respectively. Likewise, for a budget of $5 \times 10^5$ evaluations, the maximal number of points visited with regards to the batch sizes are 12500, 10000, 5000, 1000 and 500.

Figures 8 and 9 illustrate the combined effect of varying the step size and the confidence level when the influence of simulation noise is minimal ($n_b = 50$) and maximal ($n_b = 2$) respectively. We notice immediately the very poor convergence of the exploratory algorithm over $F_Q$ and its divergence over $F_H$ for the two levels of noise considered. An inspection of figures 8 and 9 reveals that the divergent behavior over $F_Q$ takes place at a distance $||x(t) - x_*|| \approx 1$, which corresponds to a flat region of $F_Q$ according to figure 6(a). This in clearly in contrast to the behavior exhibit by the conservative algorithm, which is able to converge with adequate step sizes. It suggests that the exploratory algorithm is prone to get lost when the probability of generating a good point with good Monte Carlo estimates is low, as is the case in flat regions. Moreover, the risk-taking logic never scored better than the conservative or the traditional algorithms, irrespective of the step size or the Monte Carlo strategy. The exploratory algorithm shall not be considered again in the
remains of this Chapter, as exploring too much in the face of uncertainty simply does not seem a good strategy.

Let us consider now the effect of changing the step size. An interesting phenomenon is observed regarding the best step size in a noisy environment. Whereas in deterministic optimization the smaller the step size the better, we notice how this hierarchy has now an inflection point and is then reverted for the smallest step sizes, whose values make the optimizer to remain trapped into the neighborhood of the initial point. It is also noted that the best step size for the conservative optimizer is generally larger than its counterpart under traditional optimization.

The influence of the number of batches is illustrated in figure 10 for chosen step sizes that bear good performances in figure 8, viz. $\sigma = 0.50$ for $F_Q$ and $\sigma = 1.00$ for $F_H$. It seems that the optimizer always converge faster during the first iterations using $n_b = 2$ for $\alpha = 0.50$ although it finishes with lower accuracy than with the other values of $n_b$. Moreover the convergence using $n_b = 2$ looks always better using $\alpha = 0.90$. Nevertheless, the supremacy of $n_b = 2$ is only relative. Figure 11 shows the convergence for the first 500 points using different values of $n_b$. We clearly see that reducing the noise level of the points by augmenting the number of batches has positive effects on the accuracy shown by the optimizer. However, if the evaluation budget is constrained, the user is confronted with the problem of how to allocate the evaluations optimally to approximate the optimum as close as possible. The results in figure 10 suggest the development of procedures to detect changes in the convergence speed that indicate when $n_b$ has to be incremented.

At this point we are able to compare the traditional optimizer against the conservative one. It is seen that in general the former one outperforms the latter. This is due to the fact that the conservative optimizer demands greater evidence to replace the old point by the new one. In practical terms it means that the average $\bar{f}$ of new point has to be not only lower than that of the current one but significantly smaller, either because the distance between them is large or because the variances are very small. The first condition explains the fact shown in the previous figures, where the conservative optimizer needs larger step sizes to converge. The last condition, on the other hand, indicates that one has to expect more differences between the two types of optimizers when $n_b$ is 2 than when is 50. Moreover, the logic of the HT with $\alpha = 0.90$ is to avoid divergences, not to help convergence. It suggest that the conservative optimizer has chances to outperforms the traditional one when $n_b$ is small and the probability of divergence is high. This is precisely the case in figure 12 where the conservative algorithm has a better performance when big step sizes are used during the first part of the optimization. Notice the slight divergence at the beginning for $\sigma = 2.4$ and $\alpha = 0.50$. Actually, it can be also confirmed that the conservative optimizer is also more stable for this case as its behavior bears less variance.

So far the basic phenomena observed are: 1) coexisting of a natural trend of the (1+1)-ES to converge better when small step sizes are used and an opposite trend that requires large step sizes to overcome the effect of noise, 2) ineffectiveness of the risk-taking logic in noisy optimization, 3) higher efficiency of traditional over conservative optimization except for environments with high probability of divergence, and 4) the beneficial effect of reducing the noise of the point-wise estimations on the accuracy of the optimizer. Additionally, the fourth phenomenon calls for an intelligent allocation of the evaluations, incrementing $n_b$ the convergence speed slow down.

Finally, let us now see if this phenomena remains when the dimensionality of the problem is changed and when the Monte Carlo simulation strategy is changed from CMC to LHS. Figure 13 shows the results of using LHS with the minimal (n_b = 50) and the maximal (n_b = 2) estimation noise level. We note with respect to the results previously shown in figures 8 and 9 that the LHS slightly improves the accuracy of optimizer on $F_Q$ and $F_H$ for both values of $\alpha$. Moreover, reproducing the experience of figure 12 one finds in figure 14 the same beneficial effect of introducing the HT for preventing divergence. Nevertheless, as the LHS reduces the noise, the difference of performances between the two optimizers for $\sigma = 2$ and $\sigma = 4$ is less pronounced.

The combined effect of dimensionality, simulation strategy and confidence level is shown in
figures 15 to 18. The same effects described earlier can be found in these results. Small step sizes are more determinant to converge in 10D, as in the deterministic case. Convergence is also harder and the optimum is approached with less accuracy (recall that \( F \) has the same optimum in 2D and 10D).

Regarding the simulation strategy, we see how the LHS improves the convergence, particularly on \( F \) (figures 17 and 18) where the efficiency is remarkably boosted making the optimizer to converge with some parameters setting that caused divergence using CMC. Also, the effect of using the HT is more clearly noted in 10D. Indeed, if the optimizer is much more prone to diverge with large step sizes using \( \alpha = 0.90 \) as we see in figure 17(b): the introduction of the HT reduced the number of values of \( \sigma \) that induce divergence, while the use of LHS improves convergence. The combined effect of these two features dramatically improves the efficacy of the algorithm (see figures 17(a) and 17(d)).

The observations reported in this section constitute elementary phenomena of that can serve as a good basis for more intelligent optimization strategies with dynamic regulation of parameters in the realm optimization with noise. The development of theory for supporting these results is also a matter of future research.

6.2 Application to the two-bars test case

In order to conclude this Chapter, this section illustrates the resolution of RDBO through the two-bars truss problem, making use once again of the notion of confidence level.

Following section 2.4, the objective function is to minimize a percentile, which according to (20) corresponds to a volume percentile. A reliability constraint is also imposed. Setting the percentile and the reliability to 90%, the original problem in (20) takes the form

\[
\min_{x = (d, L)} q_{90\%}(x) \quad \text{such that} \quad p(x) \geq 0.90,
\]

which can be solved under the penalized form

\[
\min_{x = (d, L)} f_p(x) = q_{90\%}(x) + \lambda \max \{0, p(x) - 0.90\}
\]

As the estimation of the variance of (33) poses some problems due to the second non-regular addend, the use of non-parametric HT is then embraced. Notice that the assessment of \( f_p(x) \) requires the estimation of the percentile on the one side and the estimation of the probability on the other side. If the variance of the percentile is not needed, the estimation by batching can be replaced by the use of 21. In any case, a set of samples is needed for generating a single estimate of \( f_p(x) \).

For the sake of simplicity, let us call \( f_p(x, u) \) the expression for assessing (33) and let \( u \) denotes a realization of \( U \) such that \( f_p(x, u) \) returns one estimate of (33). The resolution of (33) requires the introduction of the non-parametric test in algorithm 1.10, thus yielding the sequence shown in algorithm 3.21.

For the resolution the initial point \( x = (d = 60 \text{ mm}, L = 1200 \text{ mm}) \) were selected. The reader can check in figure 3 the feasibility of this point, so the optimization is going to take place inside the feasible zone. The reliable optimum can also be approached from the unfeasible zone starting from the deterministic optimum. Besides, the optimizer requires domain \( d \times L \) to be normalized in order to render the isotropic mutation effective. Likewise in the penalized expression \( F_p(x, u) \) the percentile of volume is expressed in \( dm^3 \) whereas the penalization factor is set to \( \lambda = 10 \) so that the volume is always lower than one and the penalty turns out dominant with small violations of the reliability constraint.

For this example we kept the batching procedure used in the preceding section. Once again the batch size is set to \( s_b = 20 \) and the percentile is estimate averaging over the \( n_b \) estimates of it.
Algorithm 3: (1+1)-ES minimization algorithm with Monte Carlo simulation and non-parametric HT.

**input** : Initial point $x^{(0)} \in \mathbb{R}^N$, step size $\sigma \in \mathbb{R}^+$, iteration budget $t^{max}$, sample size $n$, significance level $\alpha$

3.1 Initialize $t := 0$;
3.2 Generate $U_n^{(0)}$;
3.3 $Y^{(0)} := \emptyset$;
3.4 for $i := 1$ to $n$ do
3.5 \hspace{1em} $Y^{(0)} := Y^{(0)} + \{f(x_i^{(0)}, u_i^{(0)})\}$;
3.6 end
3.7 while $t \leq t^{max}$ do
3.8 \hspace{1em} for $i := 1$ to $N$ do
3.9 \hspace{2em} $x_i^{(t+1)} := x_i^{(t)} + \sigma \mathcal{N}(0, 1)$;
3.10 end
3.11 Generate $U_n^{(t+1)}$;
3.12 $Y^{(t+1)} := \emptyset$;
3.13 for $i := 1$ to $n$ do
3.14 \hspace{1em} $Y^{(t+1)} := Y^{(t+1)} + \{f_p(x_i^{(t+1)}, u_i^{(t+1)})\}$;
3.15 end
3.16 if not $HT(Y^{(t+1)}, Y^{(t)}, \alpha)$ then
3.17 \hspace{1em} $x^{(t+1)} := x^{(t)}$;
3.18 \hspace{1em} $Y^{(t+1)} := Y^{(t)}$;
3.19 end
3.20 $t := t + 1$;
3.21 end

As we just mentioned in a previous paragraph, batching is not necessary for the non-parametric test as the variance is not asked by the HT, but it can be used anyway. The sample strategy used is CMC.

Figure 19 brings comparative results of the -plain- average of $f_p$ over 30 runs for the traditional and the conservative optimizers using different step sizes ($\sigma = 0.50, 1.00, 2.00$) and two batch sizes ($n_b = 2, 10$). The evaluations budget is set to 20000 so the optimizers visit 500 or 100 points when $n_b$ is 2 or 10 respectively. The results shown a better performance for the traditional optimizer with small step sizes. This results is consonant with the behavior observed on the academical functions. Nevertheless, the traditional optimizer bears a unexpected instability with $n_b = 10$ and $\sigma = 2$. Such a fluctuation takes place when the optimizer approaches the reliability constraint, so that the value of $f_p$ can be affected by eventual violations of some of the 30 runs executed. Actually using $n_b = 2$ the optimizers obtained better values of $f_p$ with higher reliabilities, i.e. farther the constraint. This fact could explain the absence of degradations in the $f_p$ curves for the smallest $n_b$. In any case, once again we see the benefit of the HT for avoiding divergences in figure 19(b) when large step sizes are considered.

7 Concluding remarks

The problem of designing in the presence of uncertainty has been studied in this Chapter giving priority to the optimization of percentiles as a natural expression for RDBO. After covering the
different possible formulations and the parametrization of a problem of that kind, the introduction of hypothesis testing for guiding the optimization has been analyzed using the evolutionary optimizer (1+1)-ES.

The experiments covered the influence of the step size $\sigma$, noise in point-wise estimations through the number of samples $n_b$ and the sampling strategies CMC and LHS. The confidence level $1 - \alpha$ selected for the HT was also considered, associating to each significance level an attitude toward risk, namely exploratory or risk-taking ($\alpha = 0.10$), conservative or risk-avoiding ($\alpha = 0.90$) and finally risk-neutral ($\alpha = 0.50$) which corresponds to the traditional optimization schemes.

The results made evident some basic phenomena in the behavior of the (1+1)-ES: 1) optimal small step sizes in deterministic environments become inefficient due to noise, 2) the optimizer can diverge in noisy, flat regions, 3) exploratory optimization is not effective and prone to diverge, 4) the reduction of point-wise estimation noise leads to better results, but an efficient allocation of the functional evaluation budget can consider different noise levels along the optimization, and finally 5) traditional optimization is in general more efficient than the conservative one, except when the chances for diverging become important; in such cases the conservative optimization provides better results. This last remark was particularly true when the optimizer is combined with LHS.

The combination of conservative and traditional optimization schemes for enhancing efficiency along with a better use of the optimization budget through the variation of the noise level according to the convergence speed are future research directions derived from this work.

References


Figure 8: Effect of the size step and the confidence level on the quadratic (left column) and the hyperbolic (right column) functions with CMC simulation at the minimum level of noise ($n_b = 50$): each row (from top to bottom) corresponds to a different confidence optimizer, namely exploratory ($\alpha = 0.10$), traditional ($\alpha = 0.50$) and conservative ($\alpha = 0.90$) respectively.
Figure 9: Effect of the size step and the confidence level on the quadratic (left column) and the hyperbolic (right column) functions with CMC simulation at the maximal level of noise ($n_b = 2$): each row (from top to bottom) corresponds to a different confidence optimizer, namely exploratory ($\alpha = 0.10$), traditional ($\alpha = 0.50$) and conservative ($\alpha = 0.90$) respectively.
Figure 10: Effect of the sample size (number of batches) and the confidence level on the quadratic (left column) and the hyperbolic (right column) functions for chosen step sizes: the top row corresponds to the traditional optimizer ($\alpha = 0.50$) while the bottom row to the conservative optimizer ($\alpha = 0.90$).
Figure 11: Effect of the sample size (number of batches) and the confidence level on the quadratic (left column) and the hyperbolic (right column) functions for chosen step sizes with respect to the number of points evaluated: the top row corresponds to the traditional optimizer ($\alpha = 0.5$) while the bottom row to the conservative optimizer ($\alpha = 0.9$).

Figure 12: Comparison of the convergence of the traditional ($\alpha = 0.5$) and the conservative optimizer ($\alpha = 0.9$) on the 2D hyperbolic function. The use of the HT reduced the probability of divergence, hence improving the convergence during the first stage of the process when large step sizes are used.
Figure 13: Effect of the size step and the confidence level on the quadratic (left column) and the hyperbolic (right column) functions with LHS: the traditional (\( \alpha = 0.50 \)) and conservative (\( \alpha = 0.90 \)) optimizer are represented from top to bottom.

Figure 14: Comparison of the convergence of the traditional (\( \alpha = 0.50 \)) and the conservative optimizer (\( \alpha = 0.90 \)) on the 2D hyperbolic function using LHS. The use of the HT reduced the probability of divergence, hence improving the convergence during the first stage of the process when large step sizes are used.
Figure 15: Comparative effect of sampling with CMC (left) and LHS (right) on the quadratic function: the traditional (top) and conservative (bottom) optimizers for \( n_b = 2 \).

Figure 16: Comparative effect of sampling with CMC (left) and LHS (right) on the quadratic function: the traditional (top) and conservative (bottom) optimizers for \( n_b = 50 \).
Figure 17: Comparative effect of sampling with CMC (left) and LHS (right) on the hyperbolic function: the traditional (top) and conservative (bottom) optimizers for $n_b = 2$.

Figure 18: Comparative effect of sampling with CMC (left) and LHS (right) on the hyperbolic function: the traditional (top) and conservative (bottom) optimizers for $n_b = 50$. 
Figure 19: Resolution of the two-bars truss examples using non-parametric HT and CMC for $n_b = 2$ (top) and $n_b = 10$ (bottom) and a budget of 20000 evaluations.