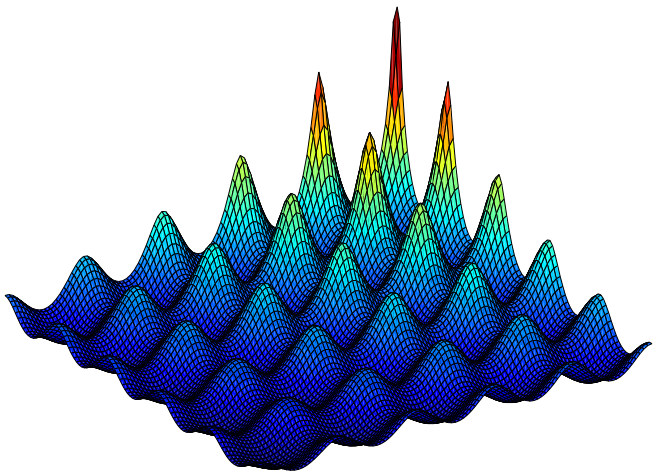
Semester Thesis

Heavy Tail Distributions in Evolutionary Algorithms

Summer Term 2005



Fabian Gemperle
Institute of Computational Science
Department of Computer Science
ETH Zurich

Advisors: Dr. Anne Auger, Dr. Nikolaus Hansen
Professor: Prof. Dr. Petros Koumoutsakos

Contents

1 Objectives and Hypothesis

In stochastic optimization procedures like *Evolutionary Algorithms* several probability distributions are known for mutation – or in other words for sampling points in the search space. Some of them are introduced in Section 3. An alternative to the frequently used Gaussian distribution is the so-called *Cauchy distribution*. *Evolution Strategies* using Cauchy mutation are known as "Fast Evolution Strategies" (FES); for more details, refer to [1]. A brief overview of Evolution Strategies in general is provided in Section 2.

Due to its heavy tails, the Cauchy distribution has different properties than the Gaussian, such as a significantly larger norm, i.e. "jump length". In [1], it is argued that this property enables Evolution Strategies employing a Cauchy mutation operator to perform more efficiently and to escape from local minima. But empirically, increased efficiency cannot be observed for all test functions considered. However, it is hinted in [2, sect. 4.5.] that FES in some cases benefit from another characteristic, namely its asymmetric shape preferring mutation directions along the main axes of the search space or so-called *reference frame*.

In [1], better performance with Cauchy mutation is apparent for a set of test functions belonging to the class of *separable functions* which are described in Section 4. Reasons for this behaviour remain unclear. For instance, it is not obvious what exactly is the influence of the heavy tails and asymmetry of a multivariate Cauchy mutation distribution on the optimization process.

The goal of this semester thesis is to identify and illustrate specific properties of Cauchy mutation operators and their relationship to separable functions and in addition, to confirm the statements made by empirical tests (Section 5) comparing the performance of miscellaneous Evolution Strategies on separable and non-separable[1] functions respectively. As a canonical example serves Rastrigin's function.

Hypotheses:

1) Evolution Strategies using a Cauchy mutation operator in the fashion of FES described in [1] have significantly better performance on *separable* functions than such utilizing other types of mutation.

2) The increase in performance of Evolution Strategies employing Cauchy mutation is caused by the heavy tails of the Cauchy distribution.

[1] including rotated separable functions

2 Evolution Strategies - An Introduction

Evolutionary Algorithms are robust optimization methods inspired by Biology. They proceed according to the principle of mutation, recombination and selection and are inherently based on randomness. Optimization is done only by comparing values of the so-called *objective* or *fitness function* and thus can be applied to black box functions. In particular, this fact may be advantageous when dealing with complicated multi-modal objective functions which pose the characteristic problem of getting trapped in a non-global local optimum (a well-known problem of deterministic optimization methods working with derivatives).
In this work, only minimization in search spaces $\mathcal{S} \subseteq \mathbb{R}^n$ incorporating the real-valued representation of *individuals*[2] is considered. So one particular problem can be stated as:

$$\vec{x}_{min} = \arg\min_{\vec{x}\in\mathcal{S}} f(\vec{x}) \quad \text{and} \quad f_{min} = f(\vec{x}_{min}) = \min_{\vec{x}\in\mathcal{S}} f(\vec{x})$$

where $f(\cdot)$ is the objective or fitness function to be minimized with n-dimensional input $\vec{x}$ out of the search space $\mathcal{S}$.

Evolution Strategies belong to one specific type of the wide variety of Evolutionary Algorithms. Evolution Strategies usually maintain a population of μ individuals during a certain number g_{max} of *generations*[3], and in every generation g produce offspring of quantity λ by mutating each individual out of the population. Frequently, also *recombination* is applied, but is discarded here because our focus lies on mutation operators. Then, selection takes place according to a certain *selection scheme*: A (μ+λ)-Evolution Strategy selects μ individuals among these λ children and μ parents whereas a (μ,λ)-Evolution Strategy selects only from the λ children and thus is rather progressive. While the first keeps the best solutions found so far in the current population, the latter has to store the best solution(s) separately. Another distinction is hard versus soft selection. With hard selection the μ best individuals among the children (and parents) are taken to survive to the next generation whilst soft selection allows worse individuals to survive with a certain, usually low probability.

For the sake of simplicity the population size is set to $\mu = 1$ and the number of offspring per generation to $\lambda = 1$ as well, viewing the (μ+λ)-Evolution Strategies just as a corresponding parallel version of this (1+1)-Evolution Strategy. So in every generation, one parent x_p is compared with its only child x_c obtained by mutation and the one with lower fitness value survives and will be the parent of the next generation, i.e. hard selection is employed. In 1-dimensional spaces, mutation is done like the following:

$$x_c \quad \sim \quad \mathcal{D}(x_p, \theta) \quad = \quad x_p + \theta\ \mathcal{D}(0,1) \quad = \quad x_p + \theta\ \Delta x$$

where $\mathcal{D}(\mu, \theta)$ is a random variable of a particular mutation distribution $\mathcal{D}$ with mean μ and scale parameter θ. In n-dimensional spaces, scalars are replaced by vectors $\vec{x}_c$, $\vec{x}_p$ and $\Delta\vec{x}$. $\mathcal{D}(\vec{\mu}, \mathbf{\Theta})$ or $\mathcal{D}(\vec{\mu}, \vec{\tau})$ then is an n-dimensional random vector with mean $\vec{\mu}$ and

[2] synonym for solution vectors $\vec{x} \in \mathcal{S}$
[3] i.e. iterations of a while loop usually

covariance matrix $\mathbf{\Theta}$ or vector $\vec{\tau}$ of scale parameters for every component (see Section 3.2).

The scale parameter θ is referred to as the *step size* of the algorithm and mainly controls the length of jumps when mutating. The optimal step size in each generation heavily depends on the current distance to the global minimum which of course cannot be taken into account during optimization. In general, early generations require a larger step size to enable exploration of the search space and in addition, large step sizes are helpful to escape local minima when being trapped. Nevertheless, towards the end of the optimization process small stepsizes are necessary for proper local convergence, i.e. to reach the desired accuracy. In [1] a self-adaptive step size is made use of, i.e. the step size is adjusted automatically depending on the Evolution Strategy's progress, whereas in the "ESSS" described in [2], the step size remains constant throughout the whole optimization process. In our empirical studies, another approach is taken: In order to prevent the choice of step sizes from additional randomness, but nonetheless to cover the most likely range of step sizes, a deterministic *cooling scheme* with a constant factor $\alpha \in (0, 1]$ is utilized, setting the initial step size θ_{init} to a rather too high value than would be needed for reasonable space exploration and the final step size θ_{final} to a rather too low value than would be needed to reach a certain accuracy more or less efficiently. These parameter settings vary among Evolution Strategies employing different mutation operators.

If one wishes to compare "fairly" the performance of Evolution Strategies using miscellaneous mutation distributions, each of the strategies should be provided with their individual optimal parameters. Because this is difficult to realize, it might be helpful to at least reduce the number of free variables of the problem like mentioned above for θ_{init} and θ_{final}. Still, some common base for comparisons is needed and in this work is contributed by equal cooling factors α for all distributions' step size adaptation, a fixed maximum number g_{max} of generations and an identical starting position $\vec{x}_{init}$. Detailed information about determining those parameters is provided in Section 5.1.

As a summary serves the following algorithm in pseudo-code, implementing a (1+1)-Evolution Strategy as used in our empirical tests. Vector $\vec{x}$ finally contains the position of a local and hopefully also of the global minimum up to a tolerance *tol* in fitness values.

INITIALIZE g_{max}, θ_{init}, θ_{final}, $\vec{x}_{init}$
$\alpha := \sqrt[g_{max}]{(\theta_{final}/\theta_{init})}$
$\vec{x} := \vec{x}_{init}$, $\theta := \theta_{init}$
$g := 1$

WHILE $g \leq g_{max}$ AND $f(\vec{x}) - f(\vec{x}_{globalmin}) > tol$ DO
    $\vec{x}' := \vec{x} + \theta \cdot \mathcal{D}(\vec{0}, \mathbb{I}_n)$[4].
    IF $f(\vec{x}') < f(\vec{x})$ THEN $\vec{x} := \vec{x}'$ END
    $\theta := \alpha \cdot \theta$
    $g := g + 1$
END

[4] or $\mathcal{D}(\vec{0}, \vec{1})$. In this work, random vector components are always uncorrelated and equal (Section 3.2)

3 Various Distributions for Mutation

3.1 Overview and Notation

In literature, a variety of different distributions is proposed as mutation operators. In the following, some of them are presented, and in addition, a distribution is introduced that will be useful to investigate Cauchy based Evolution Strategies. In Section 3.2 densities, sampling methods, some properties like variance and shape are provided. Distributions are compared by examining cuts through the joint density in Section 3.3, and a closer look at the distribution of the resulting random vector norms is taken in Section 3.4. In Table 1, abbreviations for frequently used terms and their mathematical representations are given as an overview.

Abbr.	Distribution	RV Norm	RV	Section
GO	Gaussian Distribution Original	n-dimensional	$\mathcal{N}(\vec{\mu}, \sigma^2 \mathbb{I}_n)$	3.2.1
GM1	Gaussian Distribution Modified	1-dimensional	$\mathcal{N}_{M1}(\vec{\mu}, \sigma)$	3.2.5
GMn	Gaussian Distribution "Modified"	n-dimensional	$\mathcal{N}_{Mn}(\vec{\mu}, \sigma)$	3.2.5
CO	Cauchy Distribution Original	n-dimensional	$\mathcal{C}(\vec{\mu}, \vec{\tau})$	3.2.2
CM1	Cauchy Distribution Modified	1-dimensional	$\mathcal{C}_{M1}(\vec{\mu}, \tau)$	3.2.3
CMn	Cauchy Distribution Modified	n-dimensional	$\mathcal{C}_{Mn}(\vec{\mu}, \tau)$	3.2.4

Table 1: Notation for the six mutation distributions introduced in Section 3.2. The 1^{st} column contains the abbreviations for the combined terms of the 2^{nd} and 3^{rd} column which represent the kind of distribution in general and the original dimensionality of the norm of a particular random vector (RV) with mathematical representation given in the 4^{th} column. These distributions are introduced in the sections listed in the last column.

3.2 Distribution Characteristics and Sampling

3.2.1 n-dimensional Gaussian Distribution

Scientists payed a lot of attention to the Gaussian distribution as mutator in Evolutionary Algorithms, it has been analyzed in detail. In the following, some of the facts are summarized. *GO* is used as abbreviation for *original n-dimensional Gaussian distribution.*

Sampling Method: In order to generate offspring $\vec{x}_c$ from parents $\vec{x}_p$ by mutation, the following random sampling process takes place (assuming there is a function which can return one realization $\mathcal{N}(0,1)$ of a normally distributed random variable with mean 0 and variance 1, e.g. in Matlab `randn`):

$$\vec{x}_c \quad \sim \quad \mathcal{N}(\vec{x}_p, \Sigma)$$

So $\vec{x}_p$ is the mean vector and Σ the covariance matrix of the normally distributed random vector $\vec{x}_c$. The simplest and usual version of Gaussian mutation operators emerges

in the uncorrelated case $\Sigma = \sigma^2 \mathbb{I}_n$ with n-dimensional identity matrix $\mathbb{I}_n$. Following transformations preserve equality:

$$\mathcal{N}(\vec{x}_p, \sigma^2 \mathbb{I}_n) \quad = \quad \vec{x}_p + \sigma \underbrace{\mathcal{N}(\vec{0}, \mathbb{I}_n)}_{=:\ \Delta\vec{x}} \quad = \quad \vec{x}_p + \sigma \begin{pmatrix} \mathcal{N}(0,1) \\ \mathcal{N}(0,1) \\ \vdots \\ \mathcal{N}(0,1) \end{pmatrix}$$

Characteristics: A random vector with covariance matrix $\Sigma = \sigma^2 \mathbb{I}_n$ has n independent components, each having the same distribution $\mathcal{N}(\mu, \sigma^2)$, and a spherically symmetric shape of the joint distribution with the mean vector as its center (see Figure 2). For later comparisons, only the density of the deviation $\Delta\vec{x}$ with distribution $\mathcal{N}(\vec{0}, \mathbb{I}_n)$ will be of interest, the factor σ is the standard deviation of every component and refers to the step size of the Evolution Strategy algorithm, which has to be adapted during the optimization process (In this work, a deterministic adaptation scheme for the step size will be used. Self-adapting step sizes would be another variant.) Therefore, it can be excluded from the investigation of the different random processes.

Density: The general joint density of a random vector $\vec{x} \sim \mathcal{N}(\vec{\mu}, \Sigma)$ can be written as follows:

$$p_{GO}(\vec{x} \mid \vec{\mu}, \Sigma) \quad = \quad \frac{1}{\sqrt{2\pi}^n \sqrt{\det(\Sigma)}} \ \exp(-\frac{1}{2}(\vec{x} - \vec{\mu})^t \Sigma^{-1} (\vec{x} - \vec{\mu})) \tag{1}$$

The joint density of $\Delta\vec{x} \sim \mathcal{N}(\vec{0}, \mathbb{I}_n)$ is a special case of Equation 1 and simplifies to a product of n independent 1-dimensional standard Gaussians:

$$p_{GO}(\vec{x}) \quad = \quad p_{GO}(\vec{x} \mid \vec{0}, \mathbb{I}_n) \quad = \quad \frac{1}{\sqrt{2\pi}^n} \ \exp(-\|\vec{x}\|^2/2) \tag{2}$$

This density is illustrated in Figure 1 in 1-dimensional and in Figure 2 in 2-dimensional space.

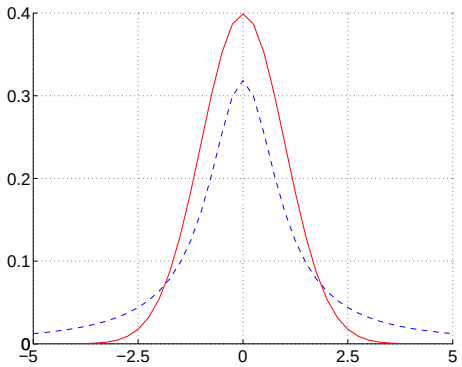


Figure 1: 1-dimensional Cauchy (blue/dashed) and Gaussian (red/solid) density for $\tau = \sigma = 1$.

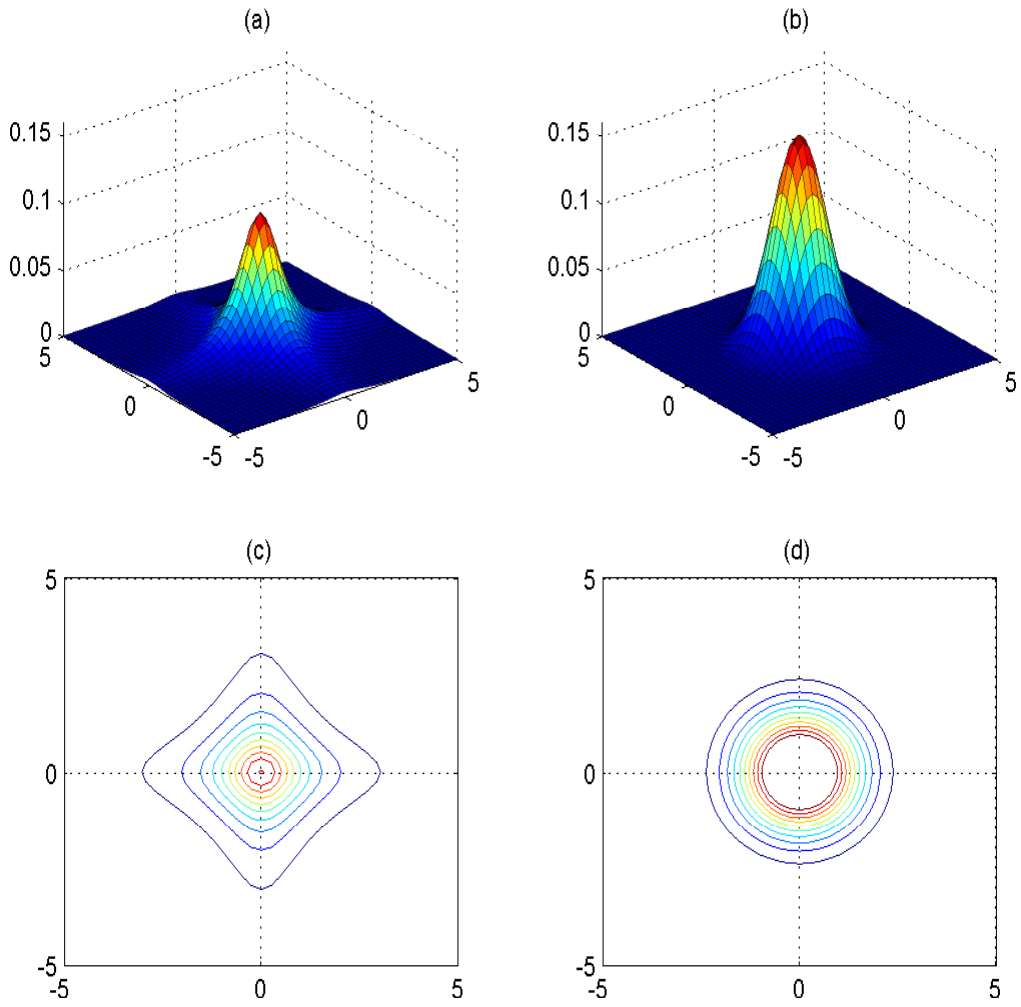


Figure 2: Top: 2-dimensional Cauchy (a) and Gaussian (b) density for $\tau = \sigma = 1$. Bottom: 10 equidistant contourlines between density values of 0.01 and 0.1 for 2-dimensional Cauchy (c) and Gaussian (d) distribution for $\tau = \sigma = 1$.

3.2.2 Non-spherical n-dimensional Cauchy Distribution

In [1], a heavytailed distribution as mutation operator is examined: the Cauchy distribution. In its original form, it is the equivalent of the uncorrelated GO but with Cauchy random vectors instead. *Non-spherical (original) n-dimensional Cauchy distribution* is abbreviated as *CO*.

Sampling Method: If a standard Gaussian random variable is divided by another independent one, the resulting ratio obeys a Cauchy distribution with location parameter or mean $\mu = 0$ and scale parameter $\tau = 1$. For Evolution Strategies, the scale parameter τ plays the same role as the standard deviation σ in Gaussian mutation, it acts as the step size of the algorithm for controlling the "jump length" i.e. the width of the density graph. However, τ^2 is *not* the same as the variance of a Cauchy distributed random

variable. CO creates random vectors having n independent Cauchy random variables as components. Adopting the notation from the last section and using the symbols $\mathcal{C}(\vec{\mu}, \vec{\tau})$ for an n-dimensional Cauchy distributed random vector with mean μ_i and scale parameter τ_i for each of the n independent components ($\vec{\mu} = (\mu_1, \mu_2, ..., \mu_n)^t$, $\vec{\tau} = (\tau_1, \tau_2, ..., \tau_n)^t$), the sampling process can be stated as:

$$\vec{x}_c \quad \sim \quad \mathcal{C}(\vec{x}_p, \vec{\tau}) \quad = \quad \vec{x}_p + \tau\mathcal{C}(\vec{0}, \vec{1}) \quad = \quad \vec{x}_p + \tau\Delta\vec{x}$$

with the same scale parameter $\tau_i = \tau$ in every component. Implementation in Matlab is done by (whereas all of the stated random variables are independent):

$$\vec{x}_c \quad \sim \quad \vec{x}_p + \tau \begin{pmatrix} \mathcal{C}(0,1) \\ \mathcal{C}(0,1) \\ \vdots \\ \mathcal{C}(0,1) \end{pmatrix} \quad = \quad \vec{x}_p + \tau \begin{pmatrix} \mathcal{N}(0,1)/\mathcal{N}(0,1) \\ \mathcal{N}(0,1)/\mathcal{N}(0,1) \\ \vdots \\ \mathcal{N}(0,1)/\mathcal{N}(0,1) \end{pmatrix}$$

Characteristics: The Cauchy density has a similar shape as the Gaussian, a peak at mean μ and decreasing probability with increasing distance from μ. Because the Cauchy density approaches zero so slowly as opposed to the exponentially decaying Gaussian density, its variance is infinite as stated in [3], see Figure 1. Figure 3 in Section 3.3 nicely illustrates the difference between Cauchy and Gaussian asymptotic behaviour on a log scale. In [1], it is argued that this property enables the Fast Evolution Strategies with CO to perform "larger jumps" than such with GO, which is helpful when having to escape from local minima of multimodal fitness functions. This is certainly true, but not the whole gain on success or speed-up is due to this special feature as experiments will confirm further on.
Another important property of this type of Cauchy distribution is its spherically asymmetric shape. That is why we call it the n-dimensional *non-spherical* Cauchy distribution (CO). Figure 2 depicts the density of the 2-dimensional $\Delta\vec{x} \sim \mathcal{C}(\vec{0}, \vec{1})$, a concentration of probability mass along the main axes is apparent. Figures 3, 4 and 5 in Section 3.3 show the discrepancy of a cut along the first main axis and a cut diagonally through the joint density in several dimensionalities. With regard to Evolution Strategies using the CO mutation operator, this means that it is most likely to generate offspring $\vec{x}_c$ from $\vec{x}_p$ in direction of the main axes, assuming a fixed norm $\|\tau\Delta\vec{x}\|$ of the shift. Conversely, the spherically symmetric GO in Section 3.2.1 chooses each direction with same probability.

Density: In general, the joint density of a random vector $\vec{x} \sim \mathcal{C}(\vec{\mu}, \vec{\tau})$ is the product of n independent 1-dimensional Cauchy densities:

$$p_{CO}(\vec{x} \mid \vec{\mu}, \vec{\tau}) \quad = \quad \prod_{i=1}^{n} \frac{1}{\pi} \frac{\tau_i}{\tau_i^2 + (x_i - \mu_i)^2} \quad = \quad \frac{1}{\pi^n} \prod_{i=1}^{n} \frac{1}{\tau_i} \frac{1}{1 + (\frac{x_i - \mu_i}{\tau_i})^2} \tag{3}$$

The joint density of $\Delta\vec{x} \sim \mathcal{C}(\vec{0}, \vec{1})$ is the "standard" case of Cauchy density in Equation 3 and looks like:

$$p_{CO}(\vec{x}) \quad = \quad p_{CO}(\vec{x} \mid \vec{0}, \vec{1}) \quad = \quad \frac{1}{\pi^n} \prod_{i=1}^{n} \frac{1}{1 + x_i^2} \tag{4}$$

Graphs of this density in one and two dimensions are provided in Figure 1 and 2 respectively.

3.2.3 Spherical n-dimensional Cauchy Distribution with 1-dimensional Norm

In order to analyze the effect of the dimension of the search space on the distribution of the samples' norms, in [2, sect.3], n-dimensional spherically symmetric distributions are defined as mutation operators having a resulting random vector norm independent of the search space dimensionality. The main idea is to sample a random direction $\vec{v} \sim \mathcal{U}(S_n(R))$ with $\|\vec{v}\| = 1$ distributed uniformly on the n-dimensional hypersphere surface $S_n(R)$ with radian R – formally defined in Equation 5. The norm R is obtained by $R \sim \mathcal{C}(0, \tau)$, that is 1-dimensional Cauchy distributed. Both terms multiplied, $R\vec{v}$, constitute the scaled deviation $\tau\Delta\vec{x}$ to get $\vec{x}_c$ from $\vec{x}_p$.

$$S_n(R) = \left\{(r, \alpha_1, \alpha_2, ..., \alpha_{n-2}, \alpha_{n-1}) \in [0, \infty) \times [0, \pi) \times [0, \pi) \times ... \times [0, \pi) \times [0, 2\pi) \mid r = R\right\} \tag{5}$$

$(r, \alpha_1, \alpha_2, ..., \alpha_{n-1})$ are (hyper[5])spherical coordinates – with one norm-variable and n-1 angle-variables – as utilized for sampling in [2, sect.3]. In this work, another approach is taken to get rid of angles and to apply proper norm sampling. *CM1* is the abbreviation for *spherical (modified) n-dimensional Cauchy distribution with 1-dimensional norm.*

Sampling Method: Knowing that GO is spherically symmetric in the uncorrelated case, its realizations of random vectors can be used as uniform random directions $\vec{v}$ after having normalized them. The random vector norm r which actually just covers the range $[0, \infty]$ is sampled according to $r \sim |\mathcal{C}(0, \tau)|$. So the overall sampled random vector can be written as:

$$\vec{x}_c \quad \sim \quad \vec{x}_p + r\vec{v} \quad = \quad \vec{x}_p + |\mathcal{C}(0, \tau)|\, \mathcal{U}(S_n(1))$$

Implementation can be done as mentioned in the paragraph above (τ will again be the step size):

$$\vec{x}_c \quad \sim \quad \vec{x}_p + \tau|\mathcal{C}(0, 1)|\, \mathcal{U}(S_n(1)) \quad = \quad \vec{x}_p + \tau\, |\frac{\mathcal{N}_1(0, 1)}{\mathcal{N}_2(0, 1)}|\, \frac{\mathcal{N}_3(\vec{0}, \mathbb{I}_n)}{\|\mathcal{N}_3(\vec{0}, \mathbb{I}_n)\|}$$

New Notation: Different numerical indices at calligraphic distribution symbols indicate that these random variables or vectors are independent whereas identical indices have the meaning of identical realization of the random variable or vector. This is needed for $\mathcal{N}_3(\vec{0}, \mathbb{I}_n)$ which occurs twice in the expression above. However, the individual components in random vectors like $\mathcal{N}_3(\vec{0}, \mathbb{I}_n)$ are again independent within the vector itself.
Let a sample vector of CM1 be labelled:

$$\mathcal{C}_{M1}(\vec{\mu}, \tau) \quad = \quad \vec{\mu} + \tau|\mathcal{C}(0, 1)|\, \mathcal{U}(S_n(1)) \quad = \quad \vec{\mu} + \tau\Delta\vec{x}$$

The n-dimensional vector $\vec{\mu}$ is the location parameter and τ the scalar scale parameter of the 1-dimensional Cauchy norm distribution. In the Evolution Strategies' sampling process of $\mathcal{C}_{M1}(\vec{\mu}, \tau)$ above, we have $\vec{\mu} = \vec{x}_p$ and τ will be the step size.

[5] i.e. 3-dimensional spherical coordinates generalized to n dimensions

Characteristics: As its name suggests, this "modified" distribution CM1 has attributes of the Cauchy distribution but concurrently, it is spherically symmetric around $\vec{\mu}$ in whatever dimensionality. The asymmetry due to multivariate Cauchy influence is not apparent anymore as opposed to the CO variant in Section 3.2.2, because the direction distribution is spherical itself and only one scalar Cauchy distributed random variable is involved for norm sampling. n-dimensional probability distributions modified in this particular way have throughout the same distribution $|\mathcal{C}(0,\tau)|$ on their norms independent of the present number of dimensions n. Nevertheless, the Cauchy distributed norm maintains the property of heavy tails.

Density: For later comparisons, we again just focus on the shift $\Delta\vec{x} \sim \mathcal{C}_{M1}(\vec{0},1)$. Two parts are needed to define a density for $\mathcal{C}_{M1}(\vec{0},1)$:

- Density for $|\mathcal{C}(0,1)|$:
  This can be derived from the density of the random variable $\mathcal{C}(0,1)$. Sampling r directly from $|\mathcal{C}(0,1)|$ is equivalent to sampling x from $\mathcal{C}(0,1)$ and then taking the absolute value of x to get r. If one does the latter, each absolute value r is twice as probable as the corresponding value x which can have either positive or negative sign. So the appropriate density function of $|\mathcal{C}(0,1)|$ is the doubled density of $\mathcal{C}(0,1)$ but restricted to the range $r \in [0,\infty)$:

$$p(r) \quad = \quad \frac{2}{\pi}\frac{1}{1+r^2} \tag{6}$$

- Density[6] for $\mathcal{U}(S_n(R))\ \forall R \in [0,\infty)$:
  A uniform density function given on a certain domain is constant on this domain and has to sum up to 1. In our case, the domain is $S_n(R)$, the surface of the n-dimensional hypersphere with fixed radian R as defined in Equation 5. In [4], the volume of $S_n(R)$ is given by $2R^{n-1}\pi^{n/2}/\Gamma(n/2)$, where $\Gamma(\cdot)$ is the Gamma function. Therefore, the density $q_n(r,\alpha_1,\alpha_2,...,\alpha_{n-1})$ for $\mathcal{U}(S_n(R))$ can be stated as:

$$q_n(r,\alpha_1,\alpha_2,...,\alpha_{n-1}) \quad = \quad \frac{\Gamma(n/2)}{2r^{n-1}\pi^{n/2}}\ \mathbb{1}_{S_n(R)}(r,\alpha_1,\alpha_2,...,\alpha_{n-1}) \tag{7}$$

where $\mathbb{1}_{\cdot}(\cdot)$ is an indicator function with spherical coordinates as arguments. More precisely:

$$\mathbb{1}_{S_n(R)}(r,\alpha_1,\alpha_2,...,\alpha_{n-1}) = \begin{cases} 1 & \text{if } (r,\alpha_1,\alpha_2,...,\alpha_{n-1}) \in S_n(R) \\ 0 & \text{otherwise} \end{cases}$$

The n-dimensional overall density follows by multiplying $p(r)$ and $q_n(r,\alpha_1,\alpha_2,...,\alpha_{n-1})$:

$$p^{(n)}_{CM1}(r,\alpha_1,\alpha_2,...,\alpha_{n-1}) \quad = \quad \frac{2}{\pi}\ \frac{1}{1+r^2}\ \frac{\Gamma(n/2)}{2r^{n-1}\pi^{n/2}}\ \mathbb{1}_{S_n(r)}(r,\alpha_1,\alpha_2,...,\alpha_{n-1})$$

[6] This is *not* a density for a random vector in n dimensions, but is a density for an $(n-1)$-dimensional manifold if restricted to $S_n(R)$ with fixed radian R.

Since the indicator function $\mathbb{1}_{S_n(r)}(r, \alpha_1, \alpha_2, ..., \alpha_{n-1})$ has always value 1, the density term $p^{(n)}_{CM1}(r, \alpha_1, \alpha_2, ..., \alpha_{n-1})$ does not depend on the angles $\alpha_1, \alpha_2, ..., \alpha_{n-1}$ anymore and so, the norm r fully determines the probability independent of the direction chosen. In fact, this holds for all spherically symmetric densities. Omitting angles and simplifying eventually yields:

$$p^{(n)}_{CM1}(r) \quad = \quad \frac{2}{\pi} \frac{1}{1+r^2} \frac{\Gamma(n/2)}{2r^{n-1}\pi^{n/2}} \quad = \quad \frac{\Gamma(n/2)}{\sqrt{\pi}^{n+2} r^{n-1}} \frac{1}{1+r^2} \tag{8}$$

The term r^{n-1} in $p^{(n)}_{CM1}(r)$'s denominator causes a singularity at $r = 0$.

3.2.4 Spherical n-dimensional Cauchy Distribution with n-dimensional Norm

In order to analyze the behaviour of n-dimensional Cauchy mutation operators (especially on separable functions) and to support the hypothesis stated in Section 1, one important tool is needed: an n-dimensional spherically symmetric distribution which has the same distribution on its norm as the n-dimensional non-spherical Cauchy distribution (CO). Such two mutation distributions can then be compared under the condition of equal norm distributions, i.e. they have the same probability to perform some beneficial "large jump" when stuck in a local minimum. What is left as a difference between these two mutation operators, is their individual shape: Spherically symmetric versus asymmetric, i.e. whether certain directions within the search space are preferred or not. *Spherical (modified) n-dimensional Cauchy distribution with n-dimensional norm* is abbreviated as *CMn*.

Sampling Method: Sampling can be done in the fashion of the last section. The only difference is that r now is distributed like the norm of an n-dimensional non-spherical Cauchy random vector, so $r \sim \|\mathcal{C}(\vec{0}, \vec{\tau})\|$.

$$\vec{x}_c \quad \sim \quad \vec{x}_p + r\vec{v} \quad = \quad \vec{x}_p + \|\mathcal{C}(\vec{0}, \vec{\tau})\| \, \mathcal{U}(S_n(1))$$

Following recast excludes the step size τ from random sampling:

$$\vec{x}_c \quad \sim \quad \vec{x}_p + \|\tau\mathcal{C}(\vec{0}, \vec{1})\| \, \mathcal{U}(S_n(1)) \quad = \quad \vec{x}_p + \tau\|\mathcal{C}(\vec{0}, \vec{1})\| \, \mathcal{U}(S_n(1))$$

To reduce the whole sampling process to sampling normals, it is rewritten to the following form:

$$\vec{x}_c \quad \sim \quad \vec{x}_p + \tau \, \| \begin{pmatrix} \mathcal{N}_1(0,1)/\mathcal{N}_2(0,1) \\ \mathcal{N}_3(0,1)/\mathcal{N}_4(0,1) \\ \vdots \\ \mathcal{N}_{2n-1}(0,1)/\mathcal{N}_{2n}(0,1) \end{pmatrix} \| \quad \frac{\mathcal{N}_{2n+1}(\vec{0}, \mathbb{I}_n)}{\|\mathcal{N}_{2n+1}(\vec{0}, \mathbb{I}_n)\|}$$

Again, the indices at the calligraphic distribution symbols are identifiers of different realizations of random variables ($\mathcal{N}_1(0,1)$ ... $\mathcal{N}_{2n}(0,1)$) or n-dimensional vectors with independent components ($\mathcal{N}_{2n+1}(\vec{0}, \mathbb{I}_n)$).
A sample vector of CMn shall be labelled:

$$\mathcal{C}_{Mn}(\vec{\mu}, \tau) \quad = \quad \vec{\mu} + \tau\|\mathcal{C}(\vec{0}, \vec{1})\| \, \mathcal{U}(S_n(1)) \quad = \quad \vec{\mu} + \tau\Delta\vec{x}$$

Characteristics: $\mathcal{C}_{Mn}(\vec{\mu}, \tau)$ essentially adopts the properties of $\mathcal{C}_{M1}(\vec{\mu}, \tau)$, except for the norm $r \sim \|\mathcal{C}(\vec{0}, \vec{\tau})\|$ whose individual distribution is equivalent to the norm distribution of a random vector sampled from CO. Discrepancies that arise when utilizing the norm of an n-dimensional Cauchy vector instead of a 1-dimensional are difficult to illustrate in the Cauchy case because the variance and also the expected norm are infinite and hence, cannot be computed to scale the densities by a finite factor in order to be able to compare both of them properly. However, in the next section the same modifications are applied to GO where a variance exists and the norms can be analyzed in detail.

Density: Densities for the uniform direction and the Cauchy norm were needed to state a density function for $\mathcal{C}_{Mn}(\vec{\mu}, \tau)$. Some effort was made to derive a closed expression for the density of an n-dimensional Cauchy random vector's norm distribution. Density function transformations like convolutions to add the squares of vector components, cartesian-to-spherical-coordinates to marginalize the norm by angles or just even computing the marginal of an n-dimensional Cauchy distribution seems to be rather complicated – computations using symbolic maths software did not produce any results. So the density function for $\mathcal{C}_{Mn}(\vec{\mu}, \tau)$ remains unknown.

3.2.5 Modified n-dimensional Gaussian Distributions

As previously mentioned, the modifications applied to the Cauchy distributions in Sections 3.2.3 and 3.2.4 can also be deployed to the Gaussian distribution. Main ideas are imparted in the following, but these mutation operators will not be used in our empirical tests. For instance, the modified n-dimensional Gaussian distribution with 1-dimensional norm is described and was empirically tested in [2].

Sampling Method: Accordingly, with notation adopted from the two sections before, alternative ways of sampling n-dimensional Gaussian random vectors $\mathcal{N}_{M1}(\vec{\mu}, \sigma)$ and $\mathcal{N}_{Mn}(\vec{\mu}, \sigma)$ respectively would be ($\vec{\mu} = \vec{x}_p$):

- GM1 – modified n-dimensional Gaussian distribution with 1-dimensional norm:

$$\vec{x}_c \quad \sim \quad \vec{x}_p + \sigma |\mathcal{N}(0,1)| \, \mathcal{U}(S_n(1)) \quad = \quad \vec{x}_p + \sigma \underbrace{\mathcal{N}_{M1}(\vec{0}, 1)}_{=:\, \Delta\vec{x}}$$

  In detail:

$$\vec{x}_c \quad \sim \quad \vec{x}_p + \sigma \; |\mathcal{N}_1(0,1)| \quad \frac{\mathcal{N}_2(\vec{0}, \mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0}, \mathbb{I}_n)\|}$$

- GMn – "modified' n-dimensional Gaussian distribution with n-dimensional norm:

$$\vec{x}_c \quad \sim \quad \vec{x}_p + \sigma \|\mathcal{N}(\vec{0}, \mathbb{I}_n)\| \, \mathcal{U}(S_n(1)) \quad = \quad \vec{x}_p + \sigma \underbrace{\mathcal{N}_{Mn}(\vec{0}, 1)}_{=:\, \Delta\vec{x}}$$

  In detail:

$$\vec{x}_c \quad \sim \quad \vec{x}_p + \sigma \; \|\mathcal{N}_1(\vec{0}, \mathbb{I}_n)\| \quad \frac{\mathcal{N}_2(\vec{0}, \mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0}, \mathbb{I}_n)\|}$$

Though a bit more complicated, the second approach will be shown to sample as well the usual GO described in Section 3.2.1, so $\mathcal{N}_{Mn}(\vec{0},1) = \mathcal{N}(\vec{0},\mathbb{I}_n)$. For this reason, sampling $\mathcal{N}_{Mn}(\vec{0},1)$ does not make sense at all in practice but for analytical purpose in the next paragraph, it will be helpful indeed.

Characteristics: In order to explore an important difference between distributions with 1-dimensional norm and those with n-dimensional norm, the Gaussian distribution is a feasible example to compare their corresponding covariance matrices focussed on the shifts $\Delta\vec{x} \sim \mathcal{N}_{M1}(\vec{0},1)$ and $\Delta\vec{x} \sim \mathcal{N}_{Mn}(\vec{0},1) = \mathcal{N}(\vec{0},\mathbb{I}_n)$ respectively. Due to spherical symmetry in both cases, the covariance matrix is diagonal and furthermore has identical entries within the diagonal. Thus, it is sufficient to compute just the first entry of the covariance matrix, i.e. the variance in the first random vector component, for each of the shift's distributions:

- $\Delta\vec{x} \sim \mathcal{N}_{Mn}(\vec{0},1) = \mathcal{N}(\vec{0},\mathbb{I}_n)$:
  The variance in the first component $\hat{\mathcal{N}}_{Mn}(\vec{0},1)$ or $\hat{\mathcal{N}}(\vec{0},\mathbb{I}_n)$ just equals the first entry of the covariance matrix $\Sigma = \mathbb{I}_n$, which is 1. Referring to the first component of a $\mathcal{D}$-distributed random vector $\mathcal{D}(\vec{\mu},\Theta)$ with parameters $\vec{\mu}$ and Θ is done by a hat: $\hat{\mathcal{D}}(\vec{\mu},\Theta)$. The particular kind of parameter Θ depends on the distribution $\mathcal{D}$ which can be any of the so far introduced distributions.
  Another way to approach variance would be the following:

$$
\begin{aligned}
VAR\left[\hat{\mathcal{N}}_{Mn}(\vec{0},1)\right] &\overset{\vec{\mu}=\vec{0}}{=} E\left[\left(\hat{\mathcal{N}}_{Mn}(\vec{0},1)\right)^2\right] \\
&\overset{def.}{=} E\left[\left(\|\mathcal{N}_1(\vec{0},\mathbb{I}_n)\| \ \frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right] \\
&= E\left[\|\mathcal{N}_1(\vec{0},\mathbb{I}_n)\|^2 \ \left(\frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right] \\
&\overset{independence}{=} E\left[\|\mathcal{N}_1(\vec{0},\mathbb{I}_n)\|^2\right] \ E\left[\left(\frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right] \\
&\overset{\text{Reference [5]}}{=} E\left[\chi_n^2\right] \ E\left[\left(\frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right] \\
&\overset{\text{Reference [5]}}{=} n \ \underbrace{E\left[\left(\frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right]}_{=:\ q \text{ for later use}} \\
&\overset{*}{\equiv} VAR\left[\hat{\mathcal{N}}(\vec{0},\mathbb{I}_n)\right] \\
&\overset{def.}{=} 1 \\
\Rightarrow \quad q &= 1/n
\end{aligned}
$$

where "*" signals the equivalence of $\mathcal{N}_{Mn}(\vec{0},1)$ and $\mathcal{N}(\vec{0},\mathbb{I}_n)$ and different numerical indices at calligraphic distribution symbols indicate different realizations of random variables or vectors as supplied before.

- $\Delta\vec{x} \sim \mathcal{N}_{M1}(\vec{0},1)$:

$$
\begin{aligned}
VAR\left[\hat{\mathcal{N}}_{M1}(\vec{0},1)\right] &\overset{\vec{\mu}=\vec{0}}{=} E\left[\left(\hat{\mathcal{N}}_{M1}(\vec{0},1)\right)^2\right] \\
&\overset{def.}{=} E\left[\left(|\mathcal{N}_1(0,1)|\ \frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right] \\
&= E\left[|\mathcal{N}_1(0,1)|^2\ \left(\frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right] \\
&\overset{independence}{=} E\left[|\mathcal{N}_1(0,1)|^2\right]\ E\left[\left(\frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right] \\
&\overset{\text{Reference [5]}}{=} E\left[\chi_1^2\right]\ E\left[\left(\frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right] \\
&\overset{\text{Reference [5]}}{=} 1\ \underbrace{E\left[\left(\frac{\hat{\mathcal{N}}_2(\vec{0},\mathbb{I}_n)}{\|\mathcal{N}_2(\vec{0},\mathbb{I}_n)\|}\right)^2\right]}_{=:\ q} \\
&= q \\
&\overset{\text{replace q}}{=} 1/n
\end{aligned}
$$

Concluding, the shifts $\Delta\vec{x}$ distributed by $\mathcal{N}_{Mn}(\vec{0},1) = \mathcal{N}(\vec{0},\mathbb{I}_n)$ and $\mathcal{N}_{M1}(\vec{0},1)$ differ in their variance in each component by a factor dependent on the dimensionality n of the search space:

$$VAR\left[\hat{\mathcal{N}}_{Mn}(\vec{0},1)\right] = VAR\left[\hat{\mathcal{N}}(\vec{0},\mathbb{I}_n)\right] \quad = \quad n\ VAR\left[\hat{\mathcal{N}}_{M1}(\vec{0},1)\right]$$

If the previously mentioned step size σ (renamed to σ_{ndim} for random vectors with n-dimensional norm and σ_{1dim} respectively for random vectors with 1-dimensional norm) is included into the last equation, this leads to the variance of the distribution of $\sigma\Delta\vec{x}$:

$$VAR\left[\sigma_{ndim}\hat{\mathcal{N}}_{Mn}(\vec{0},1)\right] = VAR\left[\sigma_{ndim}\hat{\mathcal{N}}(\vec{0},\mathbb{I}_n)\right] \quad = \quad n\ VAR\left[\sigma_{1dim}\hat{\mathcal{N}}_{M1}(\vec{0},1)\right]$$

$$\sigma_{ndim}^2 VAR\left[\hat{\mathcal{N}}_{Mn}(\vec{0},1)\right] = \sigma_{ndim}^2 VAR\left[\hat{\mathcal{N}}(\vec{0},\mathbb{I}_n)\right] \quad = \quad n\ \sigma_{1dim}^2 VAR\left[\hat{\mathcal{N}}_{M1}(\vec{0},1)\right]$$

In terms of standard deviations or step sizes, this yields for each component:

$$\sigma_{ndim} \quad = \quad \sqrt{n}\ \sigma_{1dim} \tag{9}$$

Density: To state a density formula for $\Delta\vec{x} \sim \mathcal{N}_{M1}(\vec{0}, 1)$ and $\Delta\vec{x} \sim \mathcal{N}_{Mn}(\vec{0}, 1)$ respectively, three essentials are needed:

- Density for $r \sim \|\mathcal{N}(\vec{0}, \mathbb{I}_n)\|$:
  Here, one actually asks for the distribution of an n-dimensional Gaussian random vector's norm. In general, a sum of squares of n standard-normally distributed random variables, i.e. n vector components of $\mathcal{N}(\vec{0}, \mathbb{I}_n)$, follows a χ_n^2 distribution with n degrees of freedom (see [5]). Applying the square root to this χ_n^2-distributed random variable leads then to the random variable representing the norm of the vector having accordingly a χ_n distribution with n degrees of freedom and density (see [6]):
  $$p_{\chi_n}(r) \quad = \quad \frac{2^{1-n/2} r^{n-1} \exp(-r^2/2)}{\Gamma(n/2)} \tag{10}$$

- Density for $r \sim |\mathcal{N}(0, 1)|$:
  This is the special case of having a norm of a scalar standard Gaussian, which is again χ-distributed, but only with $n = 1$ degree of freedom. According to [7], such a distribution is of type *half-normal* with scale parameter $\theta = \sqrt{\pi/2}$ and therefore:
  $$p_{\chi_1}(r) \quad = \quad \frac{\sqrt{2}}{\sqrt{\pi}} \exp(-r^2/2) \quad = \quad \frac{2}{\sqrt{2\pi}} \exp(-r^2/2) \tag{11}$$

- Density[7] for $\mathcal{U}(S_n(R))\ \forall R \in [0, \infty)$:
  In analogous manner as CM1 and using the same notation as in Section 3.2.3, the uniform density of the direction for a fixed radian R in n-dimensional space is:
  $$q_n(r, \alpha_1, \alpha_2, ..., \alpha_{n-1}) \quad = \quad \frac{\Gamma(n/2)}{2r^{n-1}\pi^{n/2}}\ \mathbb{1}_{S_n(R)}(r, \alpha_1, \alpha_2, ..., \alpha_{n-1})$$

The n-dimensional overall density for general r follows by multiplying $q_n(r, \alpha_1, \alpha_2, ..., \alpha_{n-1})$ and $p_{\chi_n}(r)$, and, $q_n(r, \alpha_1, \alpha_2, ..., \alpha_{n-1})$ and $p_{\chi_1}(r)$ respectively (simplified and angles omitted as in Equation 8 in Section 3.2.3):

$$p_{GMn}^{(n)}(r) \quad = \quad \frac{2^{1-n/2} r^{n-1} \exp(-r^2/2)}{\Gamma(n/2)} \frac{\Gamma(n/2)}{2r^{n-1}\pi^{n/2}} \quad = \quad \frac{1}{\sqrt{2\pi}^n} \exp(-r^2/2) \tag{12}$$

$$p_{GM1}^{(n)}(r) \quad = \quad \frac{2}{\sqrt{2\pi}} \exp(-r^2/2) \frac{\Gamma(n/2)}{2r^{n-1}\pi^{n/2}} \quad = \quad \frac{\Gamma(n/2)}{\sqrt{2}\sqrt{\pi}^{n+1} r^{n-1}} \exp(-r^2/2) \tag{13}$$

With these definitions, explicit coordinates $\vec{x}$ of a particular point in space are not needed to compute the corresponding probability. By reason of the spherical shape, everything can be expressed based on the norm r. $p_{GM1}^{(n)}(r)$ has a singularity at $r = 0$ because of division by r^{n-1}. However, in $p_{GMn}^{(n)}(r)$ the term r^{n-1} cancels out, and concluding by setting $r = \|\vec{x}\|$, the density of GMn given in Equation 12 is equal to the density of GO (Equation 2) and hence, these distributions are identical.

[7] This is *not* a density for a random vector in n dimensions, but is a density for an $(n-1)$-dimensional manifold if restricted to $S_n(R)$ with fixed radian R.

3.3 Comparison of Distributions

In the following, various plots depicting cuts through the joint densities of shifts $\Delta\vec{x}$ given in the last sections shall illustrate the diverse behaviour of different mutation operators, rather focussing on directions within the search space than on norms.

In a univariate search space, this question is not further interesting because GO, GM1 and GMn, and, CO, CM1 and CMn respectively, collapse into the usual scalar densities as shown in Figure 1 in the previous section.

For the purpose of plotting cuts through the joint multivariate densities of GO, CO, CM1 and GM1 defined by Equations 2, 4, 8 and 13, suitable straight directions (always passing the origin) are needed.
Concerning GM1 and CM1, this problem is already resolved: The variable r in $p^{(n)}_{GM1}(r)$ and $p^{(n)}_{CM1}(r)$ represents the distance from the origin along every possible direction, no matter which one. Each cut covers two of those directions: positive and negative. Because the "cut density" is symmetric around the origin, it is sufficient to consider just the positive part for computations. GO's density $p_{GO}(\vec{x})$ (Equation 2) can be replaced by $p^{(n)}_{GMn}(r)$ (Equation 12) which is based on the variable r for the norm, too.
For CO, two directions will be compared: along the first main axis $\vec{e}_1$ and along the first diagonal $\vec{d}_1$ without loss of generality by reason of symmetry. Corresponding parametrization can be achieved by:

$$\vec{x} = r\vec{e}_1 = r(1,0,0,...,0)^t \qquad \text{and} \qquad \vec{x} = r\vec{d}_1 = r(1,1,1,...,1)^t/\sqrt{n} \qquad \|\vec{e}_1\| = \|\vec{d}_1\| = 1.$$

Remark: All plots belonging to this section depict graphs of density cuts, not marginals. In addition, a $\log_{10}$ scale is used to point out differences in asymptotic behaviour.

Due to the logarithmic scale in Figure 3(a) showing cuts through 3-dimensional densities, the exponential, steep decline of the Gaussian opposed to the polynomial, slow decay of the Cauchy distributions becomes apparent, whether with or without modifications as described in Sections 3.2.3 for Cauchy and 3.2.5 for Gaussian. A closer view of the same situation is provided in Figure 3(b), where the gap between trajectories of the CO density along main axis and diagonal gets obvious. The singularity at the origin associated with the modified/spherical distribution versions is not graphed fully.

From Figure 3(a), 4(a) and (b), depicting the same graphs for 3-, 10- and 50-dimensional spaces, it can be seen that this gap between $p_{CO}(r\vec{e}_1)$ and $p_{CO}(r\vec{d}_1)$ grows with the spaces' dimensionality. That fact is diagrammed compactly in Figure 5, charting the ratio $\frac{p_{CO}(r\ \vec{e}_1)}{p_{CO}(r\ \vec{d}_1)}$ for several dimensionalities n. It also turns out, that the gap gets wider with increasing distance from the origin, so the "longer the jump", the higher is the probability to mutate rather along main axes direction. Thus for CO holds: the more distant from zero or the higher the dimensionality, the higher is the likeliness of a certain point in space with fixed norm r to lie near a main axis rather than near a diagonal (as opposed to spherical distributions with equally probable directions). This clearly is advantageous when dealing with highly multimodal separable functions (Section 4) where it is beneficial

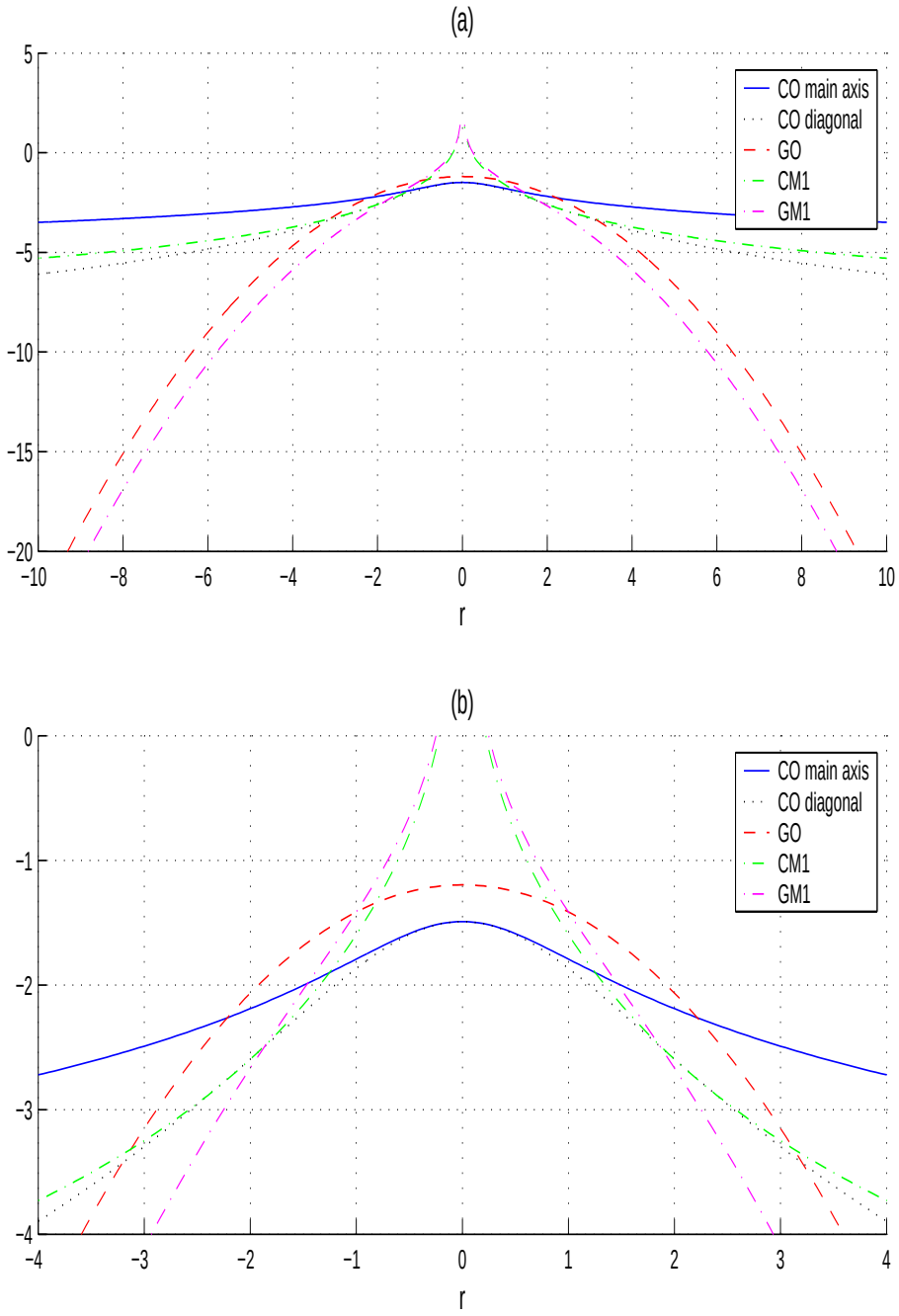


Figure 3: Cuts through 3-dimensional densities on a log scale, (a) and (b) show same graphs on different scales:
CO along first main axis $\vec{e}_1$ (blue/solid): $\log_{10}(p_{CO}(r\vec{e}_1))$ from Equation 4
CO along first diagonal $\vec{d}_1$ (black/dotted): $\log_{10}(p_{CO}(r\vec{d}_1))$ from Equation 4
GO along any direction $\vec{v}$ with $\|\vec{v}\| = 1$ (red/dashed): $\log_{10}(p_{GO}(r\vec{v}))$ from Equation 2
CM1 (green/dash-dotted): $\log_{10}(p^{(n)}_{CM1}(r))$ from Equation 8
GM1 (magenta/dash-dotted): $\log_{10}(p^{(n)}_{GM1}(r))$ from Equation 13

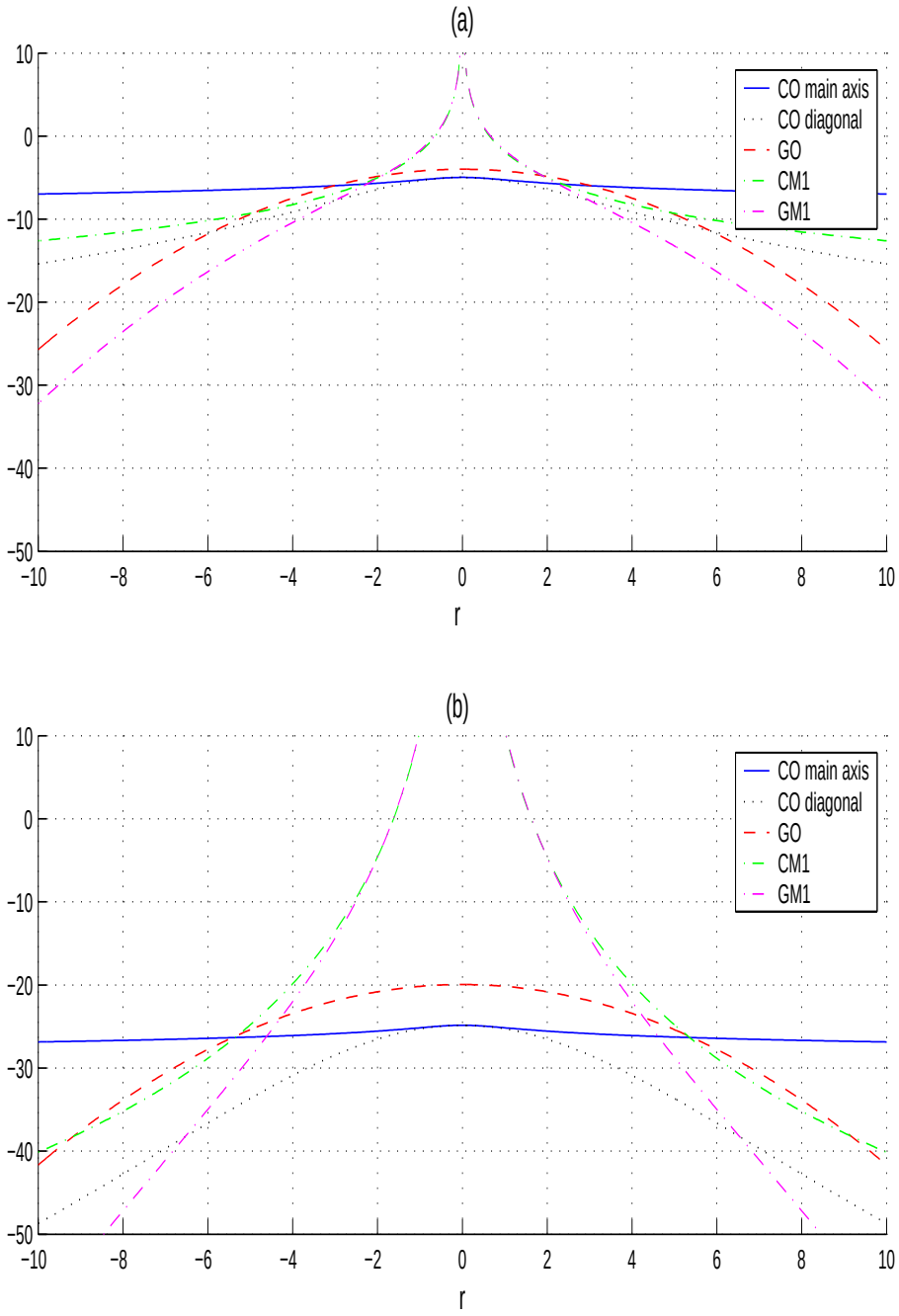


Figure 4: Cuts through n-dimensional densities on a log scale, for (a) $n = 10$ and (b) $n = 50$ dimensions:
CO along first main axis $\vec{e}_1$ (blue/solid): $\log_{10}(p_{CO}(r\vec{e}_1))$ from Equation 4
CO along first diagonal $\vec{d}_1$ (black/dotted): $\log_{10}(p_{CO}(r\vec{d}_1))$ from Equation 4
GO along any direction $\vec{v}$ with $\|\vec{v}\| = 1$ (red/dashed): $\log_{10}(p_{GO}(r\vec{v}))$ from Equation 2
CM1 (green/dash-dotted): $\log_{10}(p^{(n)}_{CM1}(r))$ from Equation 8
GM1 (magenta/dash-dotted): $\log_{10}(p^{(n)}_{GM1}(r))$ from Equation 13

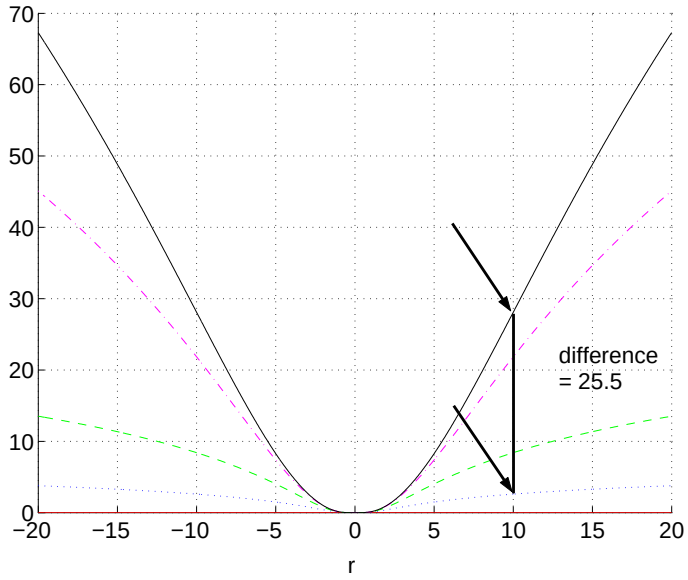


Figure 5: Log ratio $\log_{10}\left(\frac{p_{CO}(r\ \vec{e}_1)}{p_{CO}(r\ \vec{d}_1)}\right)$ of CO density cuts (Equation 4) along directions of the first main axis $\vec{e}_1$ and first diagonal $\vec{d}_1$ for several dimensionalities: $n = 1$ (red/solid), $n = 3$ (blue/dotted), $n = 10$ (green/dashed), $n = 50$ (magenta/dash-dotted), $n = 100$ (black/solid).

to produce offspring preferably in main axes direction. The situation marked by arrows in Figure 5 serves as a quantitative example: In a 3-dimensional space, we have following density values at $r = 10$ resulting in their corresponding ratio:

$$p_{CO}(10\ \vec{e}_1) \approx 3.193 \cdot 10^{-4} \quad \text{and} \quad p_{CO}(10\ \vec{d}_1) \approx 8 \cdot 10^{-7}$$

$$q_{3dim} \quad := \quad \frac{p_{CO}(10\ \vec{e}_1)}{p_{CO}(10\ \vec{d}_1)} \approx 10^{2.6} \approx 400.7 \qquad \text{in 3 dimensions.}$$

So at a distance of $r = 10$ from the origin, it is 400 times more probable to be located on the first main axis than on the first diagonal, which is indicated with the lower arrow. However, this ratio grows drastically when considering a 100-dimensional space (upper arrow in Figure 5):

$$q_{100dim} \quad := \quad \frac{p_{CO}(10\ \vec{e}_1)}{p_{CO}(10\ \vec{d}_1)} \approx 10^{28.1} \approx 1.26 \cdot 10^{28} \qquad \text{in 100 dimensions.}$$

q_{3dim} and q_{100dim} themselves differ by a huge factor of $\approx 10^{25.5} \approx 3.13 \cdot 10^{25}$.

The curve tails of the spherical Cauchy density always lie in between those of the non-spherical version along the main axis and diagonal. Like this, the mass of probability is balanced over all directions.

An important difference between random vectors with n-dimensional norm and such with 1-dimensional norm is illustrated in Figure 6 by means of the Cauchy distributions: CO

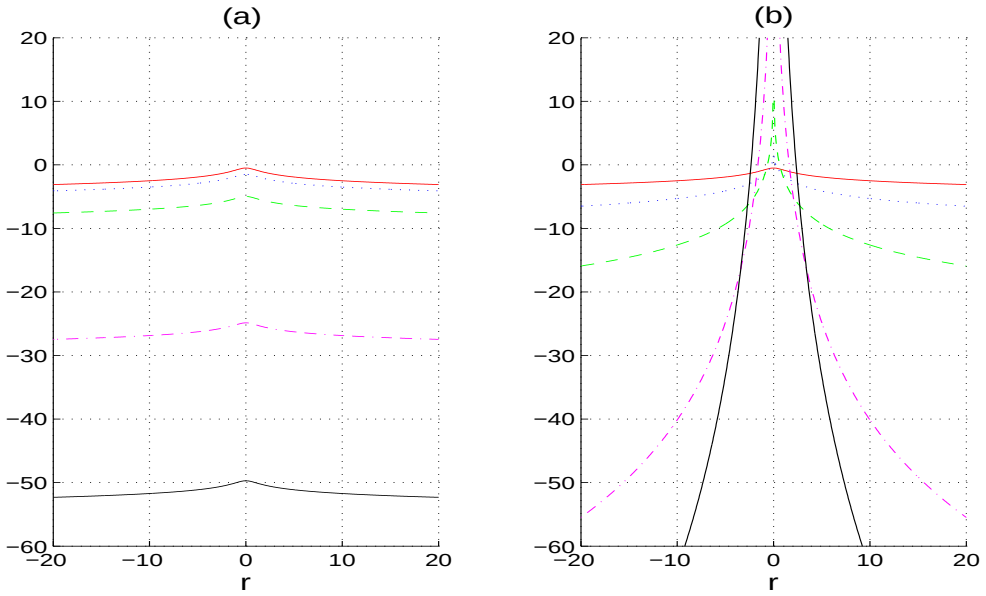


Figure 6: Log cuts through n-dimensional densities for several dimensionalities $n = 1$ (red/solid), $n = 3$ (blue/dotted), $n = 10$ (green/dashed), $n = 50$ (magenta/dash-dotted), $n = 100$ (black/solid):
(a) CO density $\log_{10}(p_{CO}(r\vec{e}_1))$ from Equation 4 along the first main axis $\vec{e}_1$
(b) CM1 density $\log_{10}(p^{(n)}_{CM1}(r))$ from Equation 8, the singularity is not graphed fully.

cut along the first main axis and CM1. Instead, a comparison of Gaussian distributions GO and GM1 could have been drawn yielding similar conclusions. Graphs of $p_{CO}(r\vec{e}_1)$ in different dimensionalities n mainly differ by certain factors[8] which become summands on a log scale – illustrated in Figure 6(a). $p^{(n)}_{CM1}(r)$'s trajectories in n-dimensional spaces but having fixed 1-dimensional norm exhibit different behaviour: the higher the dimensionality n, the more concentrated is the mass of probability around the origin, i.e. the steeper the curves get, which can be observed in Figure 6(b). When dealing with highly multivariate, multimodal fitness functions, where larger steps in the advanced optimization process are expedient to escape local minima, this property may be a disadvantage of 1-dimensional-norm mutation operators. In our empirical tests, this assertion is confirmed for $n = 5$ and especially for $n = 10$. Conversely, local convergence performance is greatly improved, which in empirical tests manifests itself in some runs with a comparably low number of generations needed to reach the global minimum within a chosen tolerance if and only if its attraction area is reached early enough during the optimization process, i.e. when the deterministically controlled step size is large enough to enforce exploration.

Up to here, all densities have been introduced as they happen to appear due to their particular sampling methods. As mentioned earlier, random vectors with a 1-dimensional norm have smaller variance than random vectors with an n-dimensional norm. This coherence was resumed in Equation 9 for Gaussians' standard deviation. In order to

[8] namely π^{-1} per additional dimension. In diagonal cuts, these factors depend on the distance from the origin.

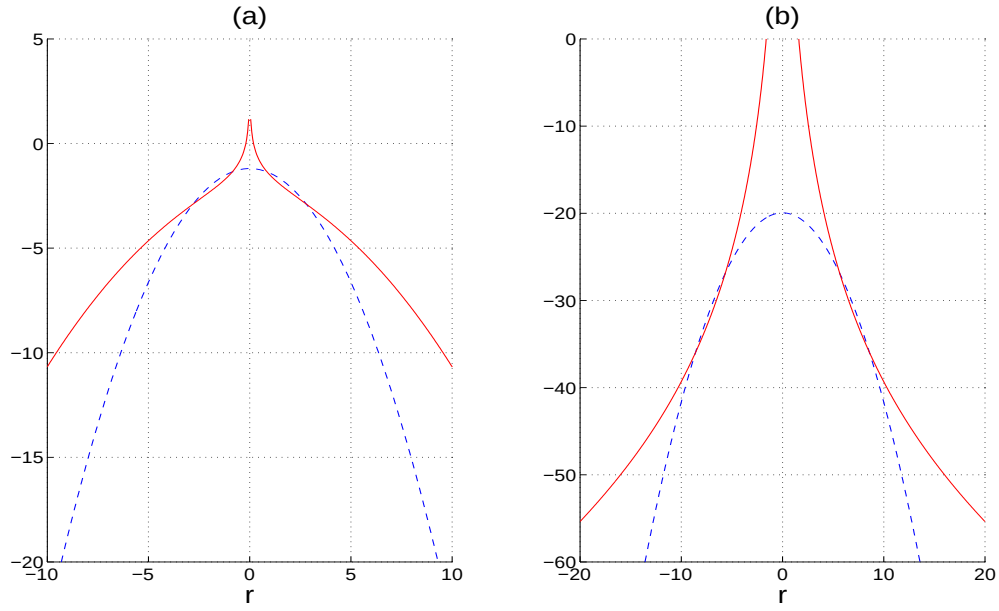


Figure 7: Comparison of log cuts in any direction $\vec{v}$ with $\|\vec{v}\| = 1$ through the n-dimensional Gaussian densities *standardized* to variance 1. In (a) for $n = 3$ and in (b) for $n = 50$ dimensions.
GO (blue/dashed): $\log_{10}(p_{GO}(r\vec{v}))$ from Equation 2
GM1 *scaled* (red/solid): $\log_{10}(p^{(n)}_{GM1}(r/\sqrt{n})/\sqrt{n}^n)$ from Equation 14, the singularity is not graphed fully.

compare the shape of GO and GM1 densities under condition of equal variance, in Figure 7 scaling is applied to $p^{(n)}_{GM1}(r)$, such that the variance of both distributions is standardized to 1. So instead, following density is plotted:

$$p^{(n)}_{GM1}(r/\sqrt{n})/\sqrt{n}^n \tag{14}$$

To get an idea about the effect of this scaling for equal variance, it is worthwile to confer GO and GM1 in Figures 3(a) and 7(a). In the latter, it can be seen that for the scaled GM1 density the mass of probability is highly concentrated around the origin, whereas for GO, its density values are larger than those of GM1 only in an intervall $|r| \approx [1, 3]$. Figure 7(b) shows the same graphs for $n = 50$, yielding similar conclusions. So, the difference in densities of GO and GM1 near the origin also is apparent when considering equal variances. This fact has to be accommodated when using mutation distributions with 1-dimensional instead of usual n-dimensional norms: Because of the shifts' smaller variance (Equation 9) and the high concentration of probability mass at the origin, the step size σ_{1dim} in general has to be chosen larger to realize jumps of increased length.

Unfortunately, it is not possible to adopt quantitative findings of the Gaussian distributions for the Cauchy ones because of Cauchy distributions' undefined variance. To get around the problem of finding individual, appropriate step sizes for CO, CMn and CM1 in the empirical tests, the step size ranges are ascertained likewise empirically instead of inferring them properly.

3.4 Comparing the Distributions of Random Vector Norms

Concerning norms, this report shall illustrate differences when employing various mutation operators – for more detail about this topic, refer to [2]. In the following, a summary will be given containing the most important facts about the distributions of norms of shift random vectors $\Delta\vec{x}$ generated by miscellaneous mutation operators introduced previously.

- $\Delta\vec{x} \sim \mathcal{N}(\vec{0}, \mathbb{I}_n) \quad \Rightarrow \quad \|\Delta\vec{x}\| \sim \chi_n$ with density from Equation 10.
  $E[\chi_n] = \frac{\sqrt{2}\Gamma((n+1)/2}{\Gamma(n/2)}$, according to [6]. For different dimensionalities $n = 1, 3, 5, 10, 50, 100$ the corresponding expectations are approximately 0.798, 1.596, 2.128, 3.084, 7.036 and 9.975.
- $\Delta\vec{x} \sim \mathcal{N}_{Mn}(\vec{0}, 1) \quad \Rightarrow \quad \|\Delta\vec{x}\| \sim \chi_n$ as above.
- $\Delta\vec{x} \sim \mathcal{N}_{M1}(\vec{0}, 1) \quad \Rightarrow \quad \|\Delta\vec{x}\| \sim \chi_1$
  $E[\chi_1] = \frac{\sqrt{2}}{\sqrt{\pi}} \approx 0.798$, independent of dimensionality n.
- $\Delta\vec{x} \sim \mathcal{C}(\vec{0}, \vec{1}) \quad \Rightarrow \quad \|\Delta\vec{x}\| \sim \|\mathcal{C}(\vec{0}, \vec{1})\|$ with unknown density.
  $E\left[\|\mathcal{C}(\vec{0}, \vec{1})\|\right] = \infty$, similar to $VAR\left[\mathcal{C}(\vec{0}, \vec{1})\right] = \infty$ as mentioned in [3].
- $\Delta\vec{x} \sim \mathcal{C}_{Mn}(\vec{0}, 1) \quad \Rightarrow \quad \|\Delta\vec{x}\| \sim \|\mathcal{C}(\vec{0}, \vec{1})\|$ as above.
- $\Delta\vec{x} \sim \mathcal{C}_{M1}(\vec{0}, 1) \quad \Rightarrow \quad \|\Delta\vec{x}\| \sim |\mathcal{C}(0, 1)|$ with density from Equation 6.
  $E[|\mathcal{C}(0, 1)|] = \infty$, similar to $VAR[\mathcal{C}(0, 1)] = \infty$ as mentioned in [3].

The infinite expectations of the norm $\|\Delta\vec{x}\|$ of Cauchy random vectors clearly indicate that on average "much" longer shifts $\Delta\vec{x}$ are generated than by other mutation operators like Gaussians. However, this might not be the only advantage when trying to escape local minima. The magnitude of jump length $\|\tau\Delta\vec{x}\|$ or $\|\sigma\Delta\vec{x}\|$ mainly is determined by the externally controlled step size τ or σ of the Evolution Strategy which could be set arbitrarily large during various phases of the optimization process. Algorithms using self-adaptive step sizes are constrained from this point of view and are yet another reason for the choice of a simple deterministic cooling scheme in our empirical tests.

Figure 8 illustrates the densities of the norm of a 10-dimensional random vector on a log-scale: in (a), densities are scaled such that all of them have equal medians[9], while in (b) densities corresponding to the sampling methods in Section 3.2 are graphed[10]. Comparing the norms' distributions in Figure 8(b), the discrepancy of the exponentially decreasing tails of Gaussian norm distributions and the heavy tails of Cauchy norm distributions becomes obvious, no matter whether with or without modifications. The concentration of probability mass at the origin in the case of mutation operators having a 1-dimensional norm is apparent as well, this time considering the norm distribution instead of a cut

[9] suitable for comparing the shape of densities
[10] suitable for comparing sampling methods

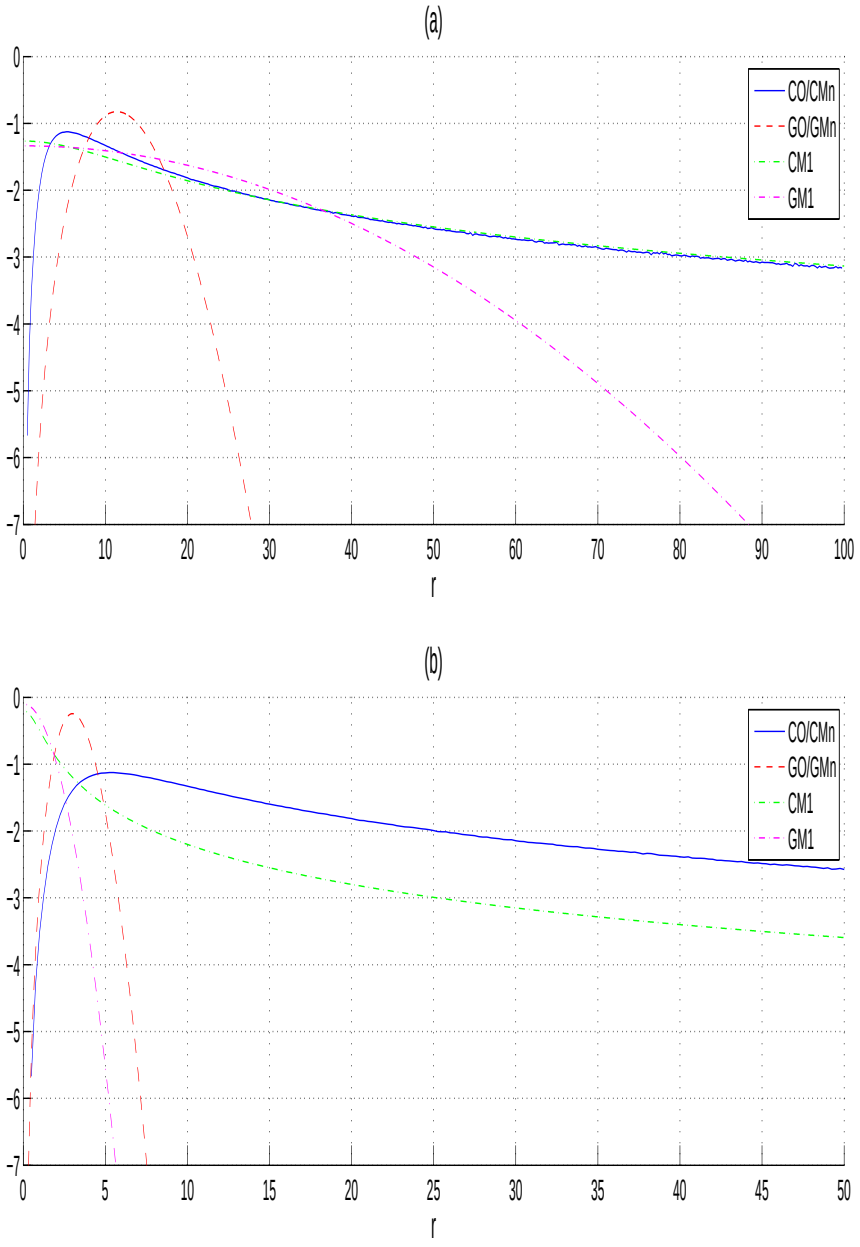


Figure 8: Illustration of the norm's distributions on a log scale, each of a 10-dimensional random vector. (a) shows densities scaled for equal medians (≈ 11.66) and (b) the raw densities of sampling methods introduced in Section 3.2.
GO (red/dashed) resulting in density $\log_{10}(p_{\chi_{10}}(r))$ from Equation 10
GM1 (magenta/dash-dotted) resulting in density $\log_{10}(p_{\chi_1}(r))$ from Equation 11
CM1 (green/dash-dotted) resulting in density $\log_{10}(p(r))$ from Equation 6
CO and CMn respectively (blue/solid) visualized by empirical density/histogram

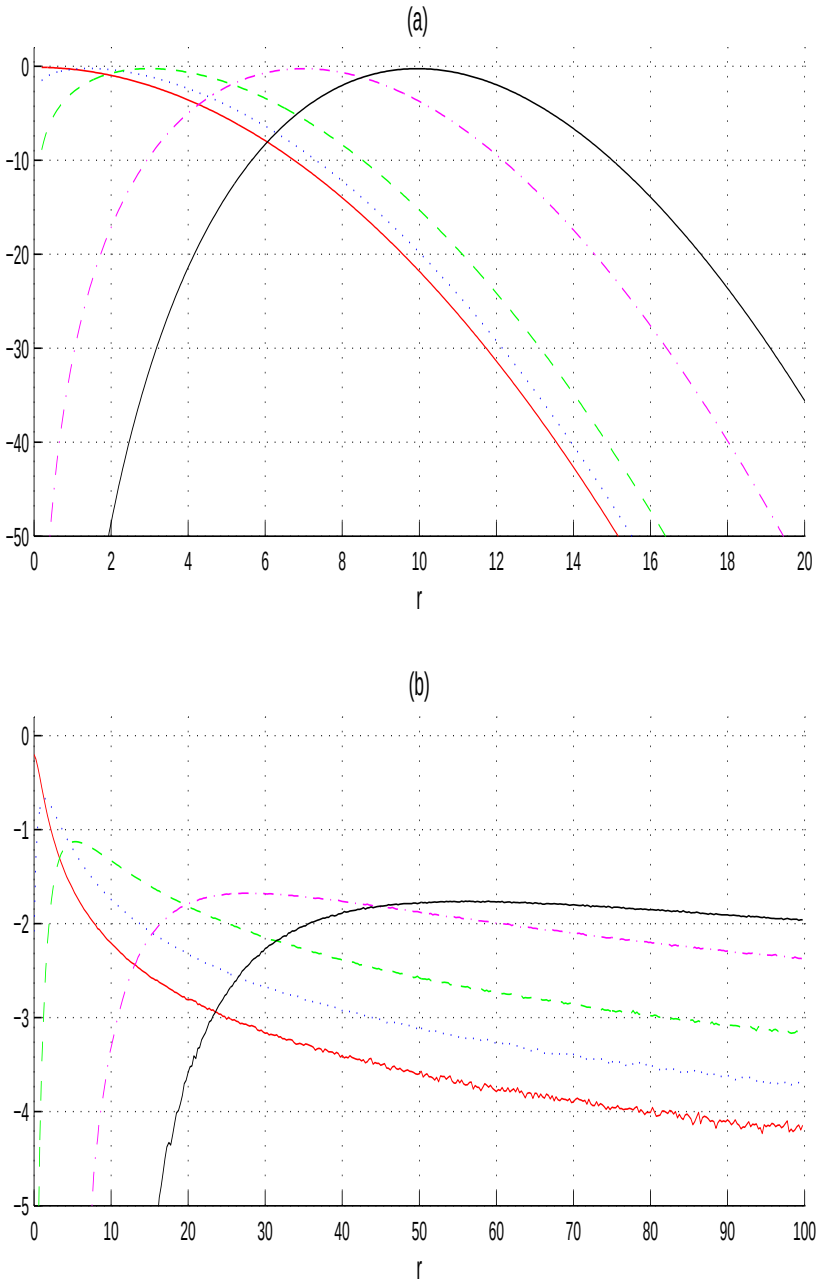


Figure 9: Random vector norms' distribution on a log scale in different dimensionalities $n = 1$ (red/solid), $n = 3$ (blue/dotted), $n = 10$ (green/dashed), $n = 50$ (magenta/dash-dotted), $n = 100$ (black/solid):
(a) GO and GMn respectively having χ-density $\log_{10}(p_{\chi_n}(r))$ from Equation 10
(b) CO and CMn respectively visualized by empirical densities/histograms
The $n = 1$ (red/solid) graphs also incorporate GM1 (density $\log_{10}(p_{\chi_1}(r))$ from Equation 11 plotted in (a)) and CM1 (density $\log_{10}(p(r))$ from Equation 6 plotted in (b)) which are invariant for all distinct dimensionalities n.

through the joint distribution.

Figure 9 depicts these norm densities on a log-scale for several dimensionalities n for Gaussian (a) and Cauchy norms (b). The so-called *curse of dimensionality* manifests itself for the mutation distributions employing n-dimensional norms: The higher the number of dimensions, the more is the mode[11] shifted away from zero and the more unprobable are small distances which implies better exploration of the search space. According to [2], the mode's position of χ_n-densities is proportional to $\sigma \cdot \sqrt{n-1}$. For CO, this effect is even stronger, for instance the norm distribution's mode of a 100-dimensional random vector lies at about $r \approx 55$ whereas for the χ_{100}-density at $r \approx 10$. However, CM1 and GM1 remain constant for whatever dimensionality and have the mode at $r = 0$. So on the one hand, mutation operators with n-dimensional norms are in need of small step sizes τ or σ for local convergence while mutators using only 1-dimensional norms frequently generate shifts near zero. But on the other hand, mutation operators with n-dimensional norms automatically generate larger step sizes, while mutators with 1-dimensional norm need artificially raised jump lengths $\|\tau\Delta\vec{x}\|$ or $\|\sigma\Delta\vec{x}\|$ to enable exploration of the search space. These facts can be accomodated by proper scaling, i.e. choosing suitable step size ranges $[\tau_{init}, \tau_{final}]$ or $[\sigma_{init}, \sigma_{final}]$.

[11] i.e. the most probable distance r

4 Separable Functions

4.1 General Remarks

Definition of separable functions: *A function* $f : \mathbb{R}^n \mapsto \mathbb{R}$ *is called separable if* $\forall i \in \{1, 2, ..., n\}$, $\forall x = (x_1, ..., x_{i-1}, x_{i+1}, ..., x_n) \in \mathbb{R}^{n-1}$ *and* $\forall y = (y_1, ..., y_{i-1}, y_{i+1}, ..., y_n) \in \mathbb{R}^{n-1}$*:*

$$\arg\min_{\alpha} f((x_1, ..., x_{i-1}, \alpha, x_{i+1}, ..., x_n) = \arg\min_{\alpha} f((y_1, ..., y_{i-1}, \alpha, y_{i+1}, ..., y_n)$$

In the case where this arg min *is not unique, the equality is in the sense of "equal sets".*

From this definition it can be seen, that each of the n coordinates of the function can be minimized independently. Considering a graphed cut along one particular coordinate axis x_i, $i \in \{1, 2, ..., n\}$, its minimum lies at the same position $\hat{x}_i$, wherever the cut is drawn exactly, i.e. independent of the fixed values x_j, $j \in \{1, 2, ..., n\} - \{i\}$. Topologically, for a bivariate separable function this implies a long (possibly very narrow) valley along the axis of x_1 passing through the minimum $\hat{x}_2$ of the second coordinate x_2 and vice versa. The global minimum then lies in the cooresponding intersection. Having once reached one of those valleys, it generally is more efficient to move along main axis directions in order to proceed to the minimum in a straight forward way.

Among separable functions, various types exist such as additively decomposable or productively decomposable functions or combinations of those like sums of separable functions for very special cases. An n-dimensional additively decomposable function is a sum of n univariate functions which do not depend on each other. A corresponding example will be given in the next section. Similarly, a productively decomposable function is a product of n univariate functions but will not be considered in this work.

4.2 Rastrigin's Function

Rastrigin's function is an example for an additively decomposable function having many local minima, depicted for the 2-dimensional case in Figure 10.

$$f_{rastrigin}(\vec{x}) \quad = \quad f_{rastrigin}(x_1, x_2, ..., x_n) \quad = \quad \sum_{i=1}^{n} (x_i^2 - 10\cos(2\pi x_i) + 10 \qquad (15)$$

The cosine term rather determines local shaping, i.e. the local minima's shape, whereas the quadratic term mainly determines the global shape, i.e. the arrangement of the various local minima in space. Local minima are found at almost-integer values 0, 0.9949, 1.9899, 2.9848, 3.9798, 4.9747... in each coordinate having function values 0, 0.9950, 3.9798, 8.9546, 15.9192, 24.8737... Therefore the set of local minima of the n-dimensional Rastrigin's function is obtained by the cartesian product of these coordinate-wise positions and their corresponding function values by adding up individual coordinates' function values. Local minima with equal function values are adjusted spherically around the origin which is the global minimum, the more distant from the origin, the higher the function value of

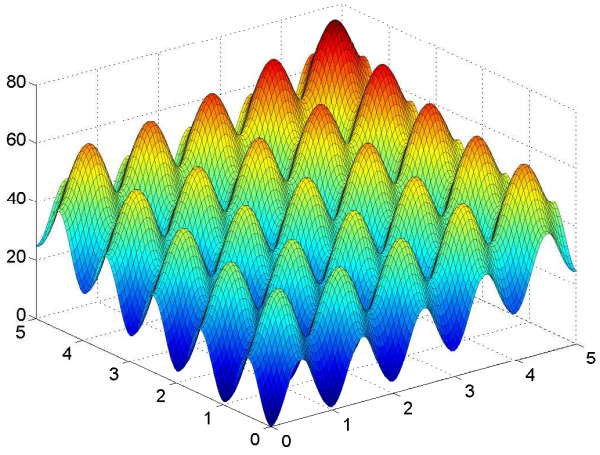


Figure 10: 2-dimensional Rastrigin's Function with the function values along z-axis for the positive quadrant. The four quadrants are symmetric around the origin, so every quadrant looks the same.

particular minima.

In order to assess the difficulty of the minimization problem at hand, usually the volume $V_{globMin}$ of the area sufficiently near[12] to the global minimum is compared with the volume $V_{searchspace}$ of the whole search space considered. For the n-dimensional Rastrigin's function, which the (1+1)-Evolution Strategy of our empirical tests in Section 5 is applied to, $V_{globMin}$ is the volume of the domain having function values all smaller than the function value of the (globally second smallest) local minimum being ≈ 0.99. This comes clear when taking into account the strictly hard selection of a (1+1)-Evolution Strategy. The initial point for the Evolution Strategies is always set to $\vec{x}_{init} = \vec{5} = (5, 5, ..., 5)^t \in \mathbb{R}^n$, which is very close to the local minimum $f_{rastrigin}(4.97, 4.97, ..., 4.97) \approx 24.87 \cdot n$. During minimization, a (1+1)-Evolution Strategy only accepts offspring $\vec{x}_c$ having:

$$f_{rastrigin}(\vec{x}_c) \leq f_{rastrigin}(\vec{x}_{init}) = f_{rastrigin}(5, 5, ..., 5) = 25 \cdot n$$

This fact together with the spherical, steplike arrangement of local minima imply a potential search space of approximatively a sphere of radian 5 around the origin.

To obtain a rough estimate of such a measure of difficulty, a Monte Carlo procedure was carried out involving 10^8 samples. For every generated sample $\vec{x} \in [-1, 1]^n$, uniformly distributed in the n-dimensional unit cube, it was checked if its function value is smaller than 0.9: $f_{rastrigin}(\vec{x}) \leq 0.9 \Rightarrow \vec{x}$ sufficiently near to global minimum, such $\vec{x}$ then were counted. These counts divided by the total number of samples represent a measure for the ratio $V_{globMin}/V_{searchspace}$ which turned out to be $1.6 \cdot 10^{-4}$ for $n = 3$, $1.8 \cdot 10^{-7}$ for $n = 5$ and finally 0 for $n = 10$ dimensions. Consequently, for such a primitive minimization algorithm it is not very likely to attain the global minimum just by chance, especially in

[12] i.e. such that local minimization leads to the global minimum

higher dimensionality.

During the optimization process Evolution Strategies utilize exploration of neighbourhood to approach the global minimum step by step. When mutating according to spherically symmetric distributions all directions within the current neighbourhood are treated with equal weight. However, when minimizing separable functions with mutation according to the non-spherical Cauchy distribution which prefers main axes' directions, certain function properties are exploited systematically. For instance, these properties and consequences can be illustrated at the 2-dimensional Rastrigin's function (used will be notation from Section 3):

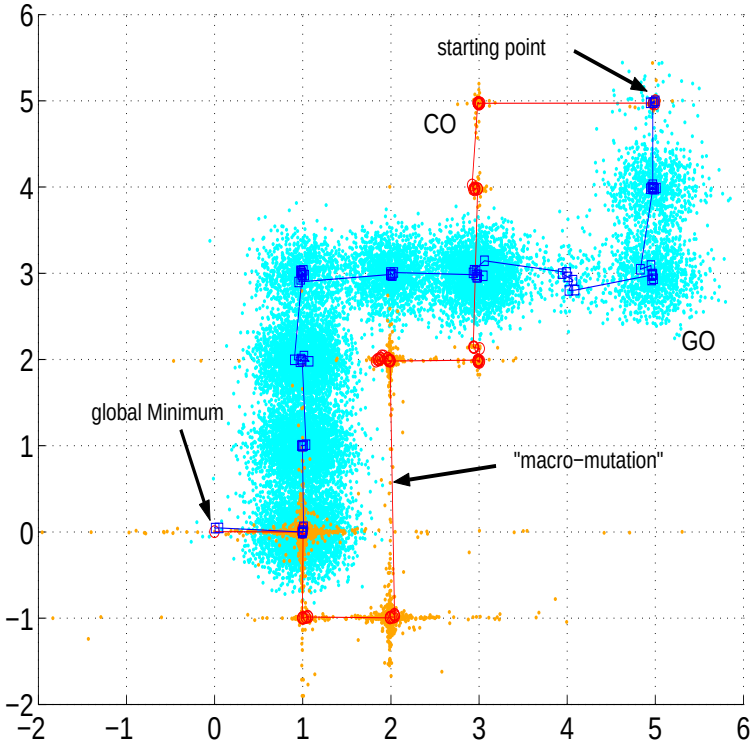


Figure 11: Track of a (1+1)-Evolution Strategy employing GO (blue line with square markers, cyan point clouds, constant step size $\sigma = 0.25$) and CO (red line with sphere markers, orange point clouds, constant step size $\tau = 0.01$) mutation when minimizing the 2-dimensional Rastrigin's Function with $\vec{x}_{init} = (5, 5)^t$. The point clouds display the overall generated offspring, but selected are only those indicated by larger sphere and square markers.

Problematic is the case of being stuck and having just small step sizes τ or σ at hand. Assuming being trapped in a particular local minimum, one strategy would be to hop local minimum by local minimum towards the global minimum. The least distant adjacent local minima always lie in main axis' direction, separated by saddles as opposed to the more distant local minima in diagonal direction being separated by comparably huge peaks. So an Evolution Strategy being able to perform steps of length ≈ 1 and having unlimited number of generations at its disposal eventually is successful in global

minimization of Rastrigin's function. If the steps are preferably directed along main axes, efficiency is increased which manifests itself in a lower number of generations used to reach the global minimum. These considerations are visualized in Figure 11: Two routes through the search space are tracked, one of a (1+1)-Evolution Strategy applying a usual Gaussian and one applying a non-spherical Cauchy mutation operator. The step size is kept constant and furthermore very small: $\sigma = 0.25$ for the Gaussian distribution with the consequence that steps longer than ≈ 1 occur rather seldom and thus the algorithm is forced to proceed to the least distant local minima one after the other as observable in Figure 11. A lot of offspring is generated in spherical clouds, but is selected only when situated in main axis direction. However, it is difficult to constrain a Cauchy Evolution Strategy in such a way, so its step size is chosen even smaller, namely $\tau = 0.01$. Looking at the corresponding route and the overall generated offspring, the preference of main axes' directions – especially when performing large steps - becomes obvious and so does the increase in efficiency. In addition, a successful "macro-mutation" with norm $\tau \ \|\Delta\vec{x}\| \approx \tau \cdot 300 \approx 3$ takes place which in general speeds up the optimization process. For the sake of completeness, the number of generations used shall be given but should not be compared quantitatively[13]: for the Cauchy Evolution Strategy 9189 and for the Gaussian 82000.

Hence, comparing different mutation operators' efficiency just based on the set of separable functions, enables the non-spherical Cauchy distribution to take advantage of a-priori knowledge about this set of functions, i.e. auspicious directions towards the global minimum are chosen preferably. In order to define a set of test functions which does not privilege certain mutation operators in that way, the set of separable functions can be enlarged with corresponding rotated functions having separability disturbed. In our empirical tests, averaging over several runs on randomly rotated Rastrigin's functions is done using:

$$f_{rastrigin}^{rotatedby\Phi}(\vec{x}) \quad = \quad f_{rastrigin}(\Phi\vec{x}) \quad \text{with } \Phi : n \times n \text{ orthogonal Matrix, } \Phi^t = \Phi^{-1}$$

Such random rotation matrices Φ are generated using a Gram-Schmidt-Ortho-Normalization-like procedure, provided as Matlab code in appendix A.1.5. To enable every (1+1)-Evolution-Strategy to start from the same position $\vec{5}$ with $f_{rastrigin}(5, 5, ..., 5) = 25 \cdot n$ in the unrotated Rastrigin's function, $\vec{x}_{init}$ for the rotated case is set to:

$$\vec{x}_{init} = \Phi^{-1} \ \vec{5} = \Phi^t \ \vec{5} \qquad \text{with}$$

$$f_{rastrigin}^{rotatedby\Phi}(\vec{x}_{init}) = f_{rastrigin}(\Phi\vec{x}_{init}) = f_{rastrigin}(\Phi\Phi^{-1} \ \vec{5}) = f_{rastrigin}(5, 5, ..., 5) = 25 \cdot n.$$

[13] because parameters are chosen to get nice visual results rather than optimal performance

5 Empirical Tests

5.1 Methodology

Test Function: This section shall provide empirical evidence in order to support the hypotheses stated at the beginning of this report. As separable fitness function serves the n-dimensional Rastrigin's function in different dimensionalities $n = 3$, 5 and 10. Rastrigin's functions randomly rotated by matrix Φ are the corresponding non-separable counterpart. To get an estimate for *quality* or *performance* of a certain Evolution Strategy on those test functions, averaging over 50 subsequent runs is done, always starting from the same position $\vec{x}_{init} = \Phi^t \vec{5}$ in the search space as described in the last paragraph of Section 4.

Measuring Performance: As a measure for performance one may consider the 50 final fitness values obtained by 50 runs of a (1+1)-Evolution Strategy using a particular mutation operator; either they are close to zero (global minimum attained) or to the fitness values of other local minima (when trapped in one of those). In the case of Rastrigin's function, taking into account its global shape similar to the sphere function, this means: The lower a final fitness value, the nearer is it located to the global minimum and therefore the better was the performance. Histogramming of these 50 values gives us the unnormalized empirical distribution of the random variable "Minimal Fitness Value found by (1+1)-Evolution Strategy". The goal then is to detect shifts between means or medians when comparing two of those empirical distributions and for this purpose, a *Wilcoxon Ranksum Test* is applied.

Another measure for performance might be the probability of success $p_{success}$ of a particular Evolution Strategy estimated on 50 runs, so: $f_{rastrigin}(\vec{x}_{final}) < 0.9 \Rightarrow$ count run as successful. This results in just two ratios to be compared. The significance of such comparisons can be verified quantitatively by employing a χ^2 *test of proportions*.

In order to detect an increase in performance when involving a Cauchy mutation operator based Evolution Strategy, following performances may be compared:

- The sets of 50 final fitness values found by a (1+1)-Evolution Strategy using a Cauchy mutation operator on separable Rastrigin's functions and on non-separable, randomly rotated ones.
- The sets of final fitness values found by Evolution Strategies using k different mutation operators on unrotated Rastrigin's functions, but as well on rotated ones. This results in two distinct symmetric $k \times k$ matrices with pair-wise comparisons as its entries, for instance the p-values of pair-wise statistical tests.

Mutation Operators: In this work, $k = 4$ different mutation operators are investigated: CO from Section 3.2.2, CMn from 3.2.4, GO from 3.2.1 and CM1 from 3.2.3). The (1+1)-Evolution Strategies employing these diverse mutation operators were implemented

in Matlab, the corresponding source code can be found in the appendix A.1.

Referring to the algorithm and corresponding step size adaptation scheme given in Section 2, parameters g_{max}, θ_{init} and θ_{final} are left to determine. For Gaussian mutation θ is renamed as σ, for Cauchy mutation as τ. The maximal number of generations allowed is set to $g_{max} := 50000 \cdot n$ adjusting to the raised problem complexity in higher-dimensional spaces.

Step Size Adaptation: Following plots shall illustrate various discrepancies in intervals $[\tau_{init}, \tau_{final}] = [\tau_0, \tau_{g_{max}}]$ arising when optimizing in different dimensionalities n and with miscellaneous mutation operators, and therefore suggest tuning the step size range individually. Using a log-scale for better readability, Figure 12 depicts the current fitness values $f_{rastrigin}(\vec{x}_i)$ and step sizes τ_i throughout the whole optimization process $i = 0, 1, ..., g_{max}$ for four distinct Evolution Strategy runs. Figure 13 provides a more detailed view on the selected step sizes of these four runs – put into a histogram are the sets:

$$\{\log_{10}(\tau_i) \mid \tau_i = \alpha^i \tau_{init} \ \wedge \ i = 1, 2, ..., g_{max} \ \wedge \ f_{rastrigin}(\vec{x}_i) < f_{rastrigin}(\vec{x}_{i-1})\}$$

The logarithm on the x-axis is needed because of the step sizes' uniformity on the log-scale. Similarly, the trajectory of the sequence of step sizes in Figure 12 results in a straight line whose constant slope can be characterized by the corresponding cooling factor α.

In Figure 12, arrows indicate the step sizes at the first improvement and at the last step which led into the global minimum's attraction area, or at the point where the attraction area of a particular local minimum is entered. Because the Evolution Strategy with CMn gets stuck, local convergence takes place until the maximal number of generations is attained, which gets obvious in Figure 13(d).

Due to its distribution concentrated around the origin, CM1 in general requires larger step sizes to accomodate for that fact, which also comes clear when comparing Figures 13(a) and (b). As well, this is the reason for the shifted lines of step sizes in Figure 12(a) and why CM1 experiences special handling in our empirical tests.

From Figures 13(a) and (c) with the CO based Evolution Strategy, it can be seen that lower step sizes are used in higher dimensionality, which can be explained with the increased shift length $\|\Delta\vec{x}\|$ as a consequence of higher dimensionality as mentioned in Section 3.4. Illustrated are runs on unrotated Rastrigin's functions only, though it turned out that intervals of selected step sizes are similar when dealing with rotated versions.

However, these single findings shall not give the impression of optimal step size intervals, which is taken care of in Section 5.4. Same analysis was done for several hundred runs for each of the four Evolution Strategies and for each of dimensionalities $n = 3$, 5 and 10. As a sensible compromise for our purposes, following parameter intervals $[\theta_{init}, \theta_{final}]$ are chosen under the condition of having the same cooling factor α[14] for various mutation

[14] i.e. $\theta_{final}/\theta_{init} = const.$ $\forall$ Evolution Strategies.

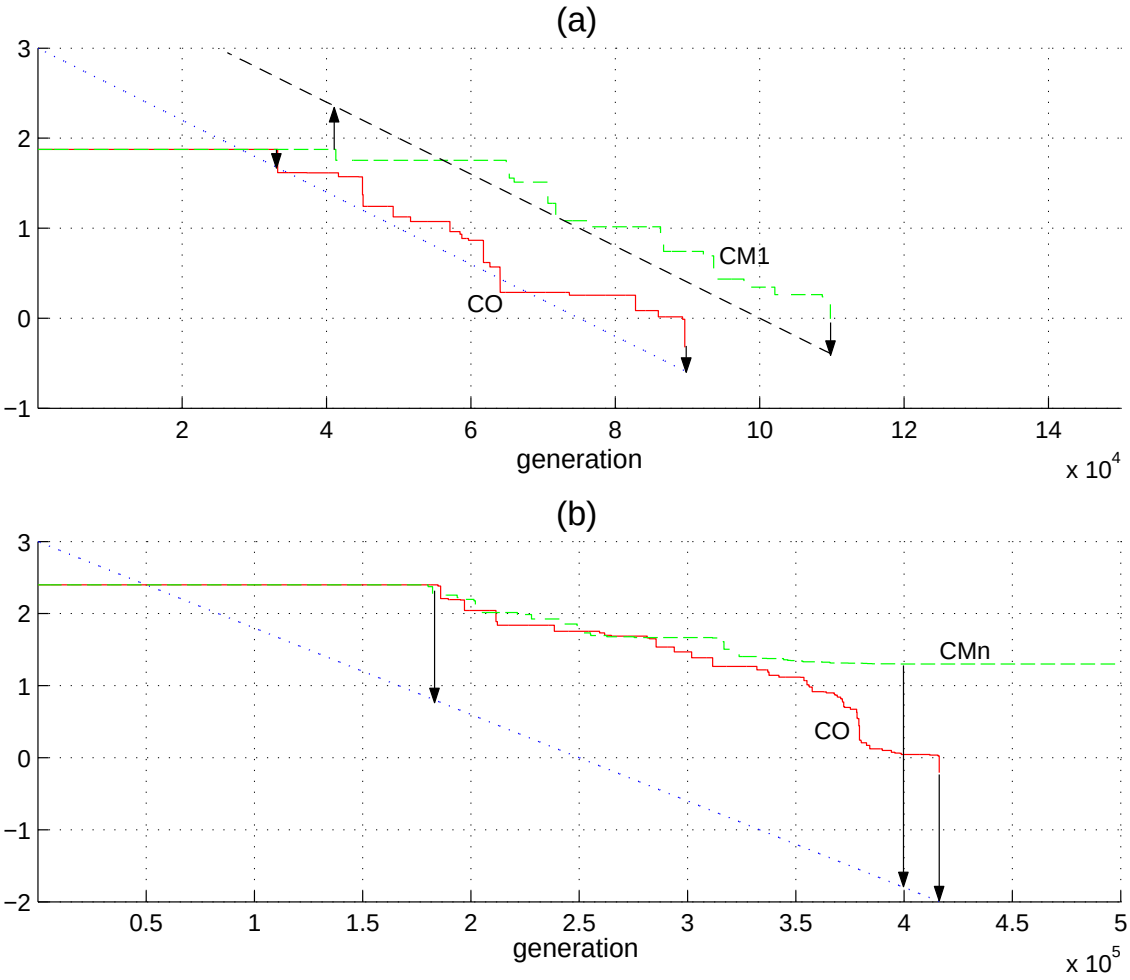


Figure 12: Optimization process of one particular (1+1)-Evolution Strategy run focusing on the fitness value in current generation i and the concurrent step size utilizing two distinct mutation operators in two different dimensionalities:
(a) Minimization of the 3-dimensional unrotated Rastrigin's function. For mutation:
CO: $\log_{10}(f_{rastrigin}(\vec{x}_i))$ red/solid and $\log_{10}(\tau_i)$ blue/dotted
CM1: $\log_{10}(f_{rastrigin}(\vec{x}_i))$ green/dashed and $\log_{10}(\tau_i)$ black/dash-dotted
Notice the different intervals $[\tau_{init}, \tau_{final}]$: for CO $[10^3, 10^{-3}]$ and for CM1 $[10^4, 10^{-2}]$.
(b) Minimization of the 10-dimensional unrotated Rastrigin's function. For mutation:
CO: $\log_{10}(f_{rastrigin}(\vec{x}_i))$ red/solid and $\log_{10}(\tau_i)$ blue/dotted
CMn: $\log_{10}(f_{rastrigin}(\vec{x}_i))$ green/dashed and $\log_{10}(\tau_i)$ blue/dotted
Both have the same step size interval $[\tau_{init}, \tau_{final}] = [10^3, 10^{-3}]$.

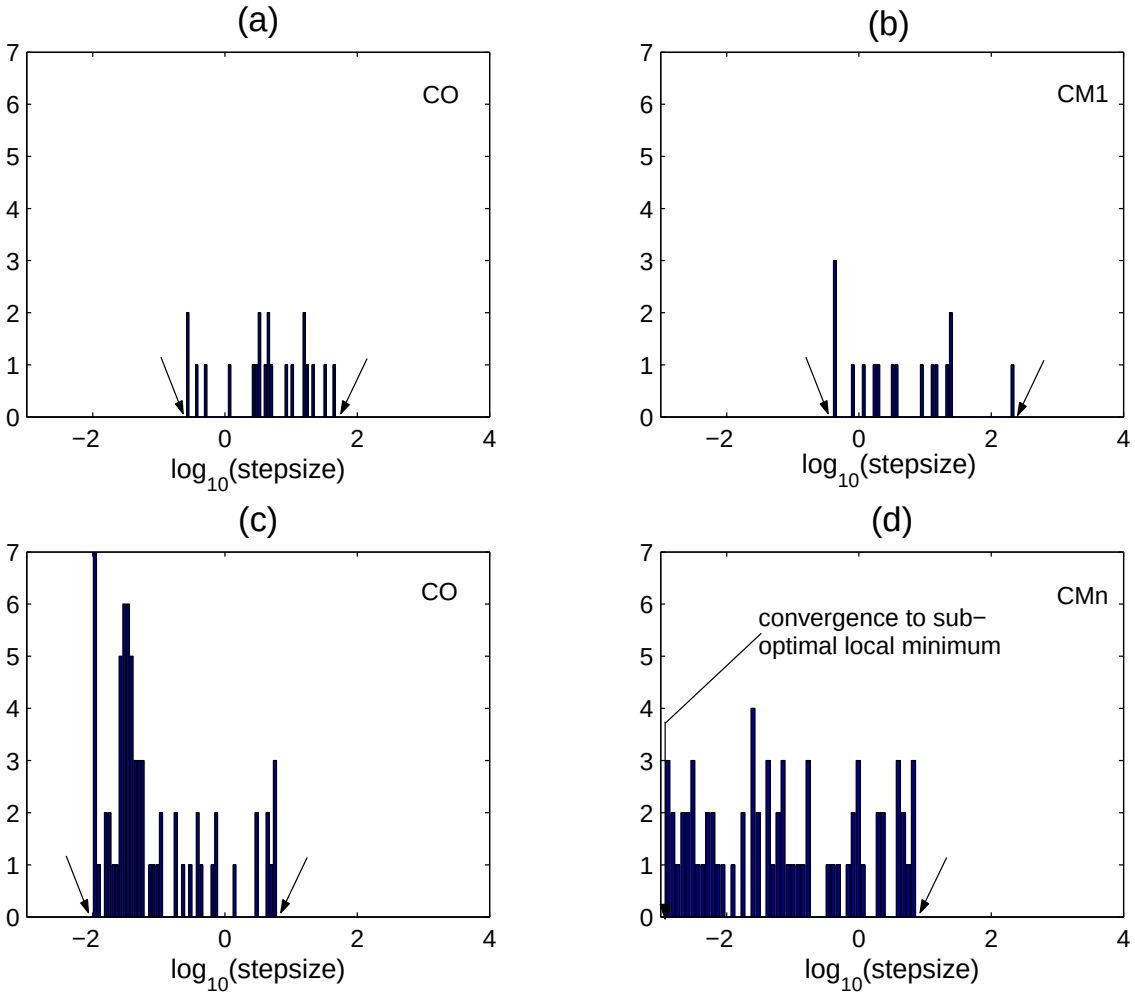


Figure 13: Recording of selected step sizes, i.e. when found a better fitness value, during the optimization processes from Figure 12. On the y-axis counts are plotted against the $\log_{10}$ of the selected step sizes on the x-axis.
(1+1)-Evolution Strategy on unrotated 3-dimensional Rastrigin's function employing
(a) CO mutation
(b) CM1 mutation
(1+1)-Evolution Strategy unrotated 10-dimensional Rastrigin's function employing
(c) CO mutation
(d) CMn mutation

operators in a certain dimensionality n, covering with high probability all potential step sizes for this special setting and holding a safeness margin of 100%, i.e. θ_{init} at least double the highest ever selected step size and θ_{final} at most half the lowest ever selected step size:

- $[10^3, 10^{-3}]$ for CO, CMn and GO mutation for all n.
- $[10^4, 10^{-2}]$ for CM1 for all n.

This results in cooling factors α having $1 - \alpha = 9 \cdot 10^{-5}$, $6 \cdot 10^{-5}$ and $3 \cdot 10^{-5}$ for dimensionalities $n = 3$, 5 and 10.

5.2 Remarks about Hypothesis Tests

Test Inputs and Assumptions: In order to compare success/failure probabilities, which are a kind of unordered, bicategorical data representable in a 2×2 contingency table, the χ^2 test of proportions is an appropriate choice, making only few rather weak assumptions[15] about the data which are fulfilled certainly. Conversely, the Wilcoxon Ranksum Test requires a continuous distribution of the data i.e. needs ordered data to be ranked uniquely. The set of 50 final fitness values obtained by our Evolution Strategy algorithms match this requirement, but introduce additional randomness due to local convergence which is not of interest here. In Statistics literature, one can find ways to get around that strong assumption, being described in the next paragraph. Hence, to get rid of influence by local convergence, final fitness values smaller than ≈ 100 can be rounded to the next larger integers except for successful optimization runs whose final fitness values are set to zero. This is possible in the case of the Rastrigin's function because of its special properties at local minima (Section 4.2). A motivating example to proceed like this can be found in [8, ch.5 sect.3, p.210 ff, ex.4].

Test Statistics: As χ^2 test statistic the usual sum of squares of relative deviations from category means is calculated and p-values are obtained via χ^2-distribution. As its name suggests, a ranksum test computes a test statistic based on the sum of ranks for each group, and if the number of data values exceeds ≈ 30, it makes use of a normal approximation with certain mean and variance instead of the exact discrete distribution to determine p-values. When testing tied data like the integer fitness values, an adjustment of this variance is necessary, explained in detail in [8, ch.1 sect.4, p.18 ff]. In addition, another adaptation needed for equally spaced data points like ours is the so-called *continuity correction*. In Matlab's *ranksum.m* procedure both of these issues are taken into account. Of course, the quality of a ranksum test suffers from ties in data because the test statistic's approximate normal distribution does not fit well anymore ([8, ch.1 sect.4, p.20]). However, results are still reasonable, but less powerful.

Test Outputs: The χ^2 test finally returns a p-value representing the significance probability of having equal proportions for each group in each category[16] whereas the ranksum

[15] advantageous: no assumptions about the data's distribution

[16] usually: contingency table with groups as rows and categories as columns.

Mutation Operator	$p_{success}$ on Rastrigin's function					
	n=3		n=5		n=10	
	unrot.	rot.	unrot.	rot.	unrot.	rot.
CO	1	0.88	1	0.04	1	0
CM*n*	0.74	0.78	0.06	0.02	0	0
GO	0.86	0.88	0.04	0.08	0	0
CM1	0.74	0.76	0	0	0	0

Table 2: Success probabilities of 50 runs of (1+1)-Evolution Strategies with four different mutation operators CO, CMn, GO and CM1, each for the unrotated and randomly rotated n-dimensional Rastrigin's function in dimensionalities $n = 3$, 5 and 10.

test returns the significance probability that there is no shift in the medians of two distinct samples. If the underlying assumptions of these statistical tests are met – which is not the case for rounded final fitness values –, it can be expected that a ranksum test takes into account more information[17] about individual samples than a χ^2 test does, and hence, p-values are more meaningful. Generally, low p-values such as 1% or less indicate a significant deviance from the assumption of having equal sources. However, in the next section results are presented as $-\log_{10}(p)$; choosing an appropriate significance level is left to the reader. For instance, a significance level of $< 1\%$ conforms to > 2 using the notation above.

5.3 Test Results and Analysis

Table 2, Figures 14, 15 and 16 hold resulting measures of performance for each of the considered mutation operators and dimensionalities. Averages of final fitness values are provided in Figures' captions.

It can be seen, that the Evolution Strategy employing the CO mutation operator always reaches the global minimum of the the unrotated Rastrigin's function, i.e. $p_{success} = 1$ and single peak at globally minimal fitness value 0 in histograms. For other variants this is never the case, rather it holds: $p_{success} < 1$ and final fitness values are distributed roughly normally having their mode shifted to a final fitness value > 0. The higher the dimensionality n, the lower is $p_{success}$ and the more shifted to right are the final fitness values in the histograms.

In the following, observed discrepancies' significance is quantitatively verified for dimensionalities $n = 3$, 5 and 10 individually. For each of the mutation operators, corresponding performances on unrotated and rotated Rastrigin's functions are compared (Table 3), and furthermore performances of various mutation operators are compared pair-wise for unrotated and rotated functions separately, which then results in two 4×4 comparison matrices for each n (Tables 4 - 9). P-values of the χ^2 test are stated in the lower triangle of these matrices, and p-values of the ranksum test in the upper triangle in the form

[17] i.e. ordering among individual samples, representing a certain distribution

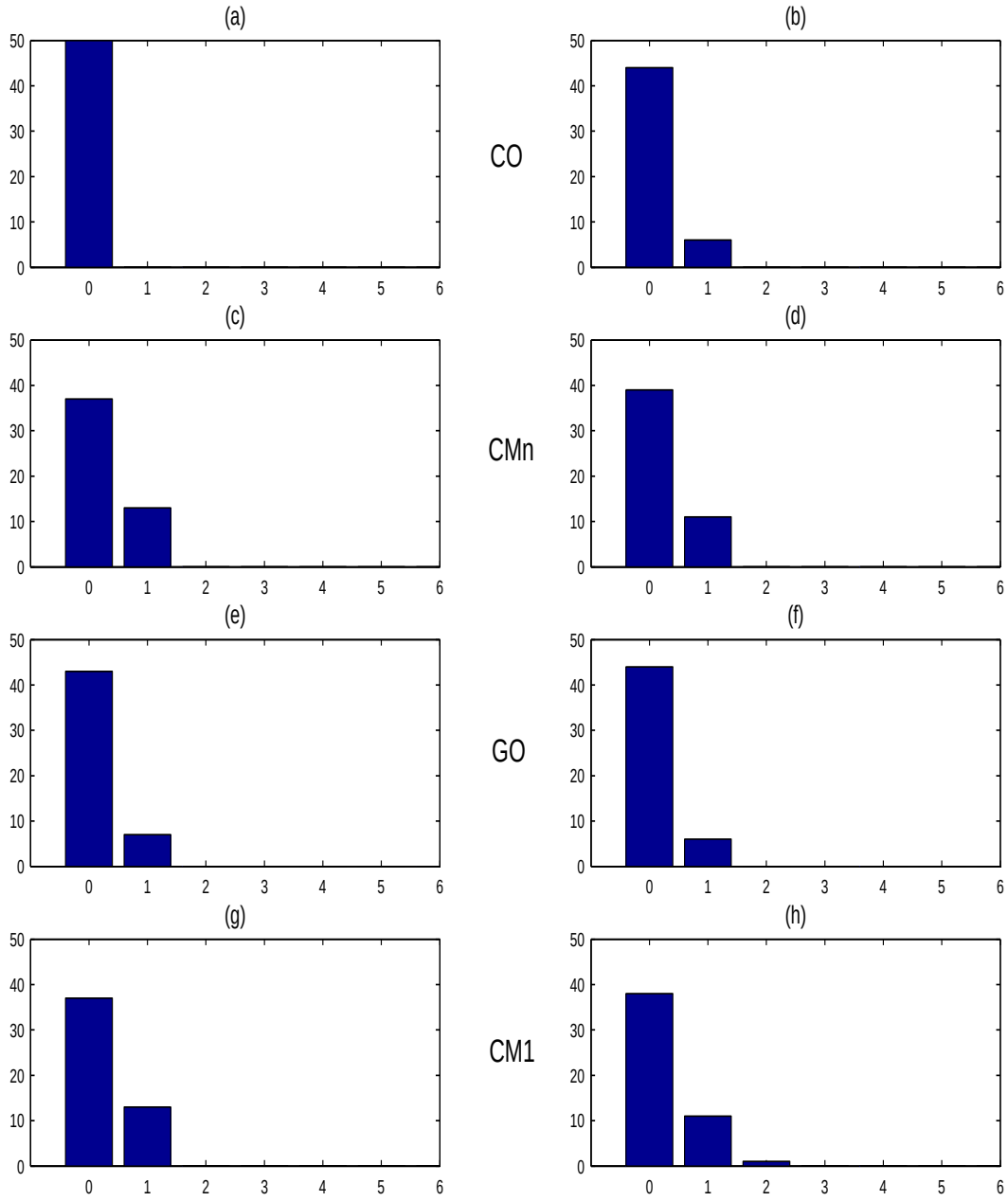


Figure 14: Histograms of final fitness values obtained by 50 runs of a (1+1)-Evolution Strategy employing four different mutation operators on 3-dimensional Rastrigin's functions (left: unrotated, right: randomly rotated Rastrigin's functions):
(a) CO on unrotated Rastrigin's functions. Final fitness value average: 0
(b) CO on randomly rotated Rastrigin's functions. Final fitness value average: 0.12
(c) CMn on unrotated Rastrigin's functions. Final fitness value average: 0.26
(d) CMn on randomly rotated Rastrigin's functions. Final fitness value average: 0.22
(e) GO on unrotated Rastrigin's functions. Final fitness value average: 0.14
(f) GO on randomly rotated Rastrigin's functions. Final fitness value average: 0.12
(g) CM1 on unrotated Rastrigin's functions. Final fitness value average: 0.26
(h) CM1 on randomly rotated Rastrigin's functions. Final fitness value average: 0.26
Stepsize range for CM1 is $[\tau_{init}, \tau_{final}] = [10^4, 10^{-2}]$ whereas for CO, CMn and GO it is $[\tau_{init}, \tau_{final}] = [10^3, 10^{-3}]$.

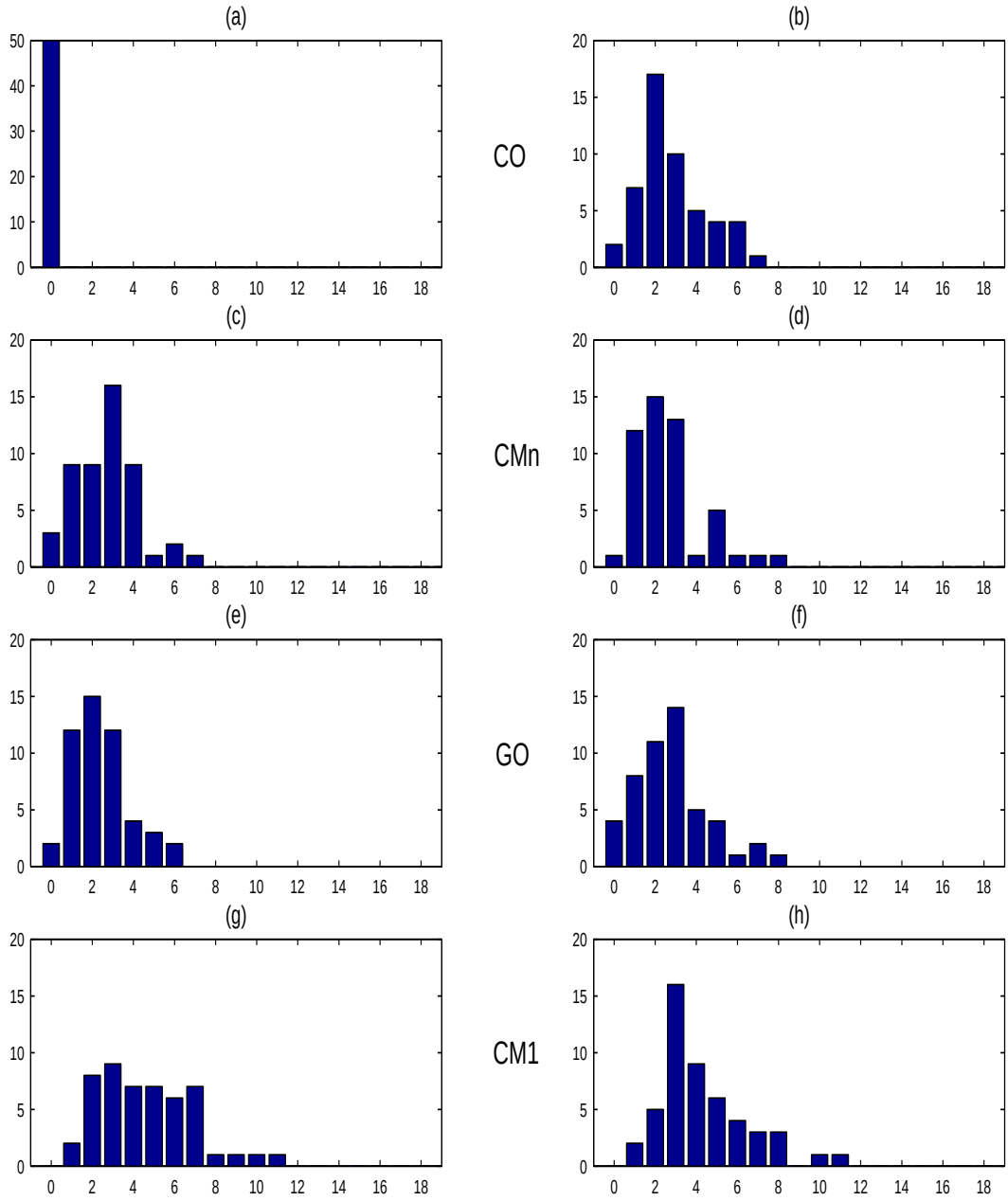


Figure 15: Histograms of final fitness values obtained by 50 runs of a (1+1)-Evolution Strategy employing four different mutation operators on 5-dimensional Rastrigin's functions (left: unrotated, right: randomly rotated Rastrigin's functions):
(a) CO on unrotated Rastrigin's functions. Final fitness value average: 0
(b) CO on randomly rotated Rastrigin's functions. Final fitness value average: 2.83
(c) CMn on unrotated Rastrigin's functions. Final fitness value average: 2.69
(d) CMn on randomly rotated Rastrigin's functions. Final fitness value average: 2.61
(e) GO on unrotated Rastrigin's functions. Final fitness value average: 2.41
(f) GO on randomly rotated Rastrigin's functions. Final fitness value average: 2.79
(g) CM1 on unrotated Rastrigin's functions. Final fitness value average: 4.60
(h) CM1 on randomly rotated Rastrigin's functions. Final fitness value average: 4.30
Stepsize range for CM1 is $[\tau_{init}, \tau_{final}] = [10^4, 10^{-2}]$ whereas for CO, CMn and GO it is $[\tau_{init}, \tau_{final}] = [10^3, 10^{-3}]$.

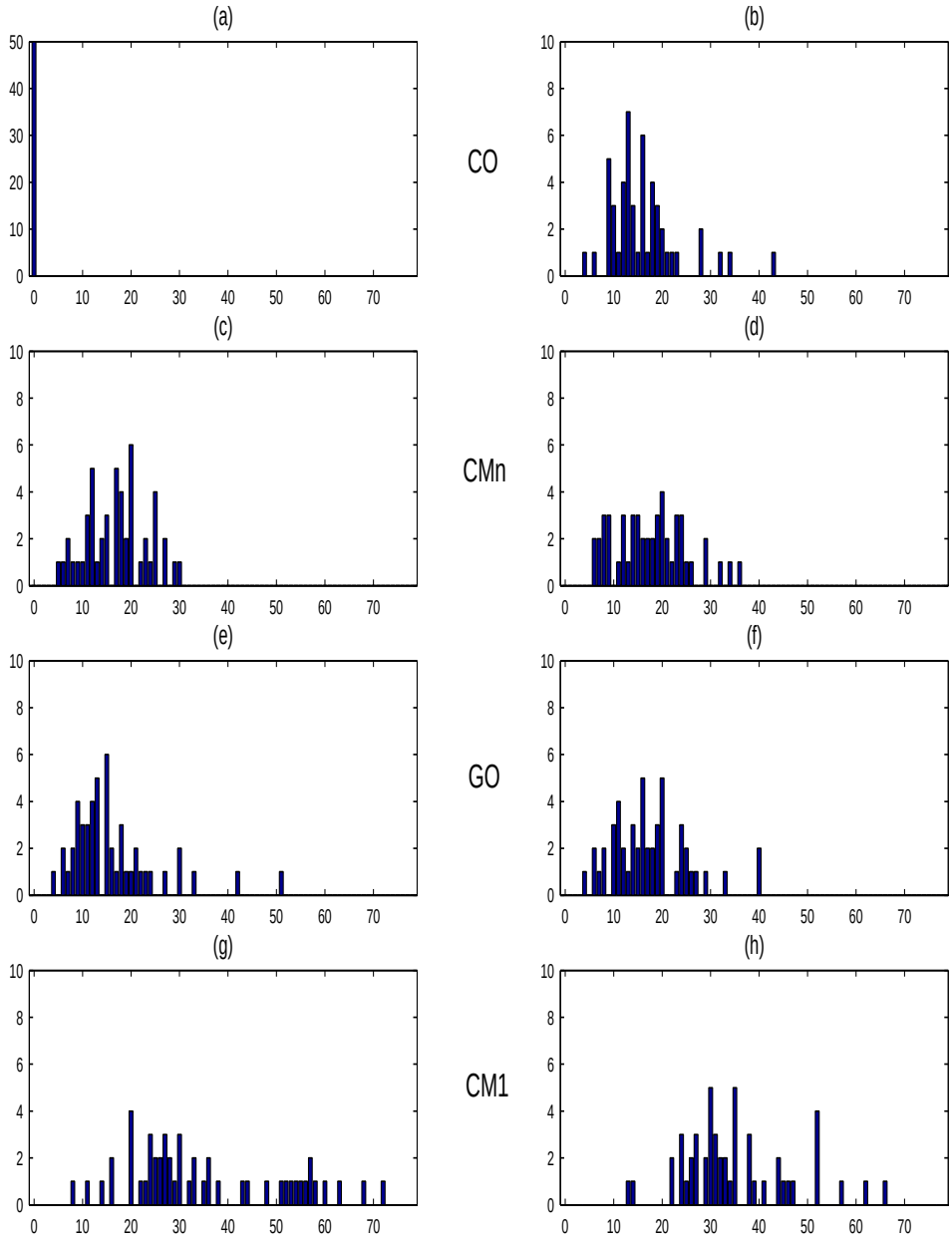


Figure 16: Histograms of final fitness values obtained by 50 runs of a (1+1)-Evolution Strategy employing four different mutation operators on 10-dimensional Rastrigin's functions (left: unrotated, right: randomly rotated Rastrigin's functions):
(a) CO on unrotated Rastrigin's functions. Final fitness value average: 0
(b) CO on randomly rotated Rastrigin's functions. Final fitness value average: 16.02
(c) CMn on unrotated Rastrigin's functions. Final fitness value average: 16.93
(d) CMn on randomly rotated Rastrigin's functions. Final fitness value average: 17.49
(e) GO on unrotated Rastrigin's functions. Final fitness value average: 16.06
(f) GO on randomly rotated Rastrigin's functions. Final fitness value average: 17.29
(g) CM1 on unrotated Rastrigin's functions. Final fitness value average: 34.90
(h) CM1 on randomly rotated Rastrigin's functions. Final fitness value average: 34.90
Stepsize range for CM1 is $[\tau_{init}, \tau_{final}] = [10^4, 10^{-2}]$ whereas for CO, CMn and GO it is $[\tau_{init}, \tau_{final}] = [10^3, 10^{-3}]$.

Mutation Operator	p-values rot./unrot. Rastrigin's function					
	n=3		n=5		n=10	
	χ^2	ranksum	χ^2	ranksum	χ^2	ranksum
CO	1.94	1.91	∞	18.46	∞	19.50
CM*n*	0.19	0.19	0.51	0.37	0.00	0.09
GO	0.12	0.11	0.40	0.51	0.00	0.72
CM1	0.09	0.06	0.00	0.30	0.00	0.35

Table 3: For each of the four mutation operators, $p_{success}$ (by χ^2 test) and the 50 final fitness values (by ranksum test) on unrotated n-dimensional Rastrigin's functions are compared with results on randomly rotated ones in dimensionalities $n = 3$, 5 and 10.

Mut.Op.	CO	CM*n*	GO	CM1
CO	0	3.91	2.19	3.91
CM*n*	3.96	0	0.86	0.00
GO	2.22	0.87	0	0.86
CM1	3.96	0.00	0.87	0

Table 4: Matrix for pair-wise comparisons of performance between the four mutation operators considered on unrotated 3-dimensional Rastrigin's functions. In the lower triangle of the matrix p-values of χ^2 tests are stated and in the upper triangle those of ranksum tests, both in the form $-\log_{10}(p)$.

$-\log_{10}(p)$. Accordingly, diagonal entries are $-\log_{10}(1) = 0$.

P-values given in these tables confirm the statements made in previous sections and enable us to decide about the hypotheses stated at the beginning of this report. Following observations support the first hypothesis in two distinct ways:

First, it can be seen from Table 3 – and taking into account averages of final fitness values and histograms in Figures 14, 15 and 16 – that CO mutation has significantly better performance on unrotated Rastrigin's functions than on rotated ones in 5 and 10 dimensions. In 3 dimensions discrepancy can be detected, too, but with a comparably rather high significance level ($\approx 2\%$). However, such differences do not occur among the three

Mut.Op.	CO	CM*n*	GO	CM1
CO	0	0.73	0.00	0.94
CM*n*	0.74	0	0.73	0.11
GO	0.00	0.74	0	0.94
CM1	0.93	0.09	0.93	0

Table 5: Matrix for pair-wise comparisons of performance between the four mutation operators considered on randomly rotated 3-dimensional Rastrigin's functions. In the lower triangle of the matrix p-values of χ^2 tests are stated and in the upper triangle those of ranksum tests, both in the form $-\log_{10}(p)$.

Mut.Op.	**CO**	**CMn**	**GO**	**CM1**
CO	0	17.90	18.47	19.53
CMn	∞	0	0.59	4.67
GO	∞	0.19	0	6.36
CM1	∞	1.10	0.81	0

Table 6: Matrix for pair-wise comparisons of performance between the four mutation operators considered on unrotated 5-dimensional Rastrigin's functions. In the lower triangle of the matrix p-values of χ^2 tests are stated and in the upper triangle those of ranksum tests, both in the form $-\log_{10}(p)$.

Mut.Op.	**CO**	**CMn**	**GO**	**CM1**
CO	0	0.39	0.04	3.75
CMn	0.25	0	0.30	5.20
GO	0.40	0.77	0	3.72
CM1	0.81	0.50	1.38	0

Table 7: Matrix for pair-wise comparisons of performance between the four mutation operators considered on randomly rotated 5-dimensional Rastrigin's functions. In the lower triangle of the matrix p-values of χ^2 tests are stated and in the upper triangle those of ranksum tests, both in the form $-\log_{10}(p)$.

spherical mutation operators.

Second, considering comparison matrices from Tables 4, 6 and 8 for unrotated Rastrigin's i.e. separable functions and again taking into account final fitness values' averages and histograms on the left-hand side in Figures 14, 15 and 16, it becomes clear that the CO mutation based Evolution Strategy performs significantly better than the three others employing spherical mutation operators. This manifests itself in high values for $-\log_{10}(p)$ in the first column (χ^2 test) and in the first row (ranksum test) of comparison matrices. The higher the dimensionality n, the more obvious is the significance of difference.

Most interesting is the comparison of CO and CMn because these two mutation operators have exactly the same distribution of norms – which therefore cannot be a source of difference in performance – and just differ in symmetry/asymmetry of their joint densities. Because of this fact, the second hypothesis can be rejected.

Another conclusion may be drawn from these results when comparing performances of the Evolution Strategy employing CM1 to the three other mutation operators' performances. Especially in higher dimensionalities, using CM1 yields larger final fitness values (Figure 16) and so performs worse than CO, CMn and GO. This observation is confirmed by rather high $-\log_{10}(p)$ values in the last column (ranksum test) of comparison matrices in Tables 8 and 9. This difference in performance cannot be a consequence of generally too small step sizes τ because they have been adjusted a priori as described in Section 5.1. One might interprete that sampling random vectors with n-dimensional norm like CO, CMn and GO is helpful to proceed faster to lower local minima during the whole

Mut.Op.	**CO**	**CMn**	**GO**	**CM1**
CO	0	19.49	19.49	19.48
CMn	∞	0	0.86	9.83
GO	∞	0.00	0	10.04
CM1	∞	0.00	0.00	0

Table 8: Matrix for pair-wise comparisons of performance between the four mutation operators considered on unrotated 10-dimensional Rastrigin's functions. In the lower triangle of the matrix (under diagonal with zeros) p-values of χ^2 tests are stated and in the upper triangle those of ranksum tests, both in the form $-\log_{10}(p)$.

Mut.Op.	**CO**	**CMn**	**GO**	**CM1**
CO	0	0.72	0.50	13.10
CMn	0.00	0	0.12	12.31
GO	0.00	0.00	0	12.06
CM1	0.00	0.00	0.00	0

Table 9: Matrix for pair-wise comparisons of performance between the four mutation operators considered on randomly rotated 10-dimensional Rastrigin's functions. In the lower triangle of the matrix (under diagonal with zeros) p-values of χ^2 tests are stated and in the upper triangle those of ranksum tests, both in the form $-\log_{10}(p)$.

optimization process whereas CM1's concentration of shifts $\Delta\vec{x}$ near the origin seems to make it difficult to escape local minima. The same fact is apparent on rotated Rastrigin's functions.

As a control, let us focus on comparison matrices in Tables 5, 7 and 9 obtained when minimizing randomly rotated Rastrigin's i.e. non-separable functions.
Histograms on the right-hand side in Figures 14, 15 and 16 look quite similar and indeed, all of the p-values, including those involving CO mutation, lie definitely above reasonable significance levels except for comparisons with the CM1 mutation operator which can be explained in the manner of the last paragraph. Thus, among performances of the three mutation operators with n-dimensional norm considered, no significant difference is apparent.

5.4 Sensitivity to Parameters

In this section, CO's advantage on separable functions is lighted from a somewhat different viewpoint. The intention is to analyze the optimality of parameters θ_{int} and θ_{final} of our chosen step size adaptation scheme and how final fitness values are affected.

5.4.1 Settings

A *grid search* approach is taken for the purpose of getting an impression about optimal step size parameters θ_{init} and θ_{final} which constitute a 2-dimensional grid space. At every

grid point, one hundred (1+1)-Evolution Strategy runs are carried out with corresponding parameters on unrotated and randomly rotated n-dimensional Rastrigin's functions respectively. Variables to be optimized are $\log_{10}(\theta_{init})$ and $\log_{10}(\theta_{final})$ with constraint $g_{max} := 50000 \cdot n$ per Evolution Strategy run fixed. As objective functions to be minimized, two possibilities are considered - the first bases primarily upon $p_{success}$ whereas the latter is based exclusively on the average of final fitness values:

- $f_{success}(\log_{10}(\theta_{init}), \log_{10}(\theta_{final})) = -p_{success} + 10^{-10} \sum_{i=1}^{100} f_{rastrigin}(\Phi \vec{x}_{final}^{\,(i)})$
- $f_{fitavg}(\log_{10}(\theta_{init}), \log_{10}(\theta_{final})) = \frac{1}{100} \sum_{i=1}^{100} f_{rastrigin}(\Phi \vec{x}_{final}^{\,(i)})$

where $\Phi \vec{x}_{final}^{\,(i)}$ is the final position (rotated by matrix Φ) found in the i-th run of the (1+1)-Evolution Strategy after having enforced full local convergence, thus points $\vec{x}_{final}^{\,(i)}$ have $f_{rastrigin}(\vec{x}_{final}^{\,(i)}) = 0$.

Because the goal is not to determine exact optimal parameters, but rather to get a visualization of the interrelationship of $[\theta_{init}, \theta_{final}]$ and performance ($f_{success}$ or f_{fitavg}) of Evolution Strategies, a quite coarse granularity for grid points is chosen covering most reasonable values, namely $\log_{10}(\theta_{init}) = -5, -4, ..., 10$ and $\log_{10}(\theta_{final}) = -10, -9, ..., 5$. Of course, intervals having $\theta_{init} < \theta_{final}$ are excluded.

Three mutation operators are tested: CO, GO and CM1. Findings for a 5-dimensional search space are presented in the next section.

5.4.2 Results

The rough shape of $f_{success}(\log_{10}(\theta_{init}), \log_{10}(\theta_{final}))$ and $f_{fitavg}(\log_{10}(\theta_{init}), \log_{10}(\theta_{final}))$, when minimizing unrotated or rotated 5-dimensional Rastrigin's functions, is plotted in Figures 17 to 21. Green arrows indicate optimal fitness values found by the grid search procedure. Results are summarized in Table 10. In general, $f_{success}$ and f_{fitavg} cannot be expected to have identical minima because they focus on slightly different measures for performance which both are convenient for global optimization purposes. However, resulting optimal parameters with respect to $f_{success}$ and f_{fitavg} should not lie too distant from each other neither due to their fitness functions' coupled base meaning[18]. Omitting the negative sign, the column "$f_{success}$" in Table 10 contains the ratio of successful runs.

Results in the first row in Table 10 for the CO-based Evolution Strategies on unrotated Rastrigin's functions are quite different from the others: $f_{success}(\log_{10}(\theta_{init}), \log_{10}(\theta_{final})) = -1$ for a whole - approximately triangular and comparably large - area of parameter points ($\log_{10}(\theta_{init})$,$\log_{10}(\theta_{final})$), all of them enabling their corresponding Evolution Strategies to reach permanently the global minimum within 100 runs (Figure 17). Also, $f_{fitavg}(\log_{10}(\theta_{init}), \log_{10}(\theta_{final})) = 0$ for more or less the same area of parameters as $f_{success}$, which results in a particular shape with a large optimal plateau at fitness value zero.

[18] $f_{rastrigin}(\vec{x}^{\,i}_{final}) \leq 0.9 \Rightarrow \vec{x}^{\,i}_{final} := \vec{0} \Rightarrow f_{rastrigin}(\vec{x}^{\,i}_{final}) = 0 \Rightarrow$ i-th run successful, i.e. increase $p_{success}$

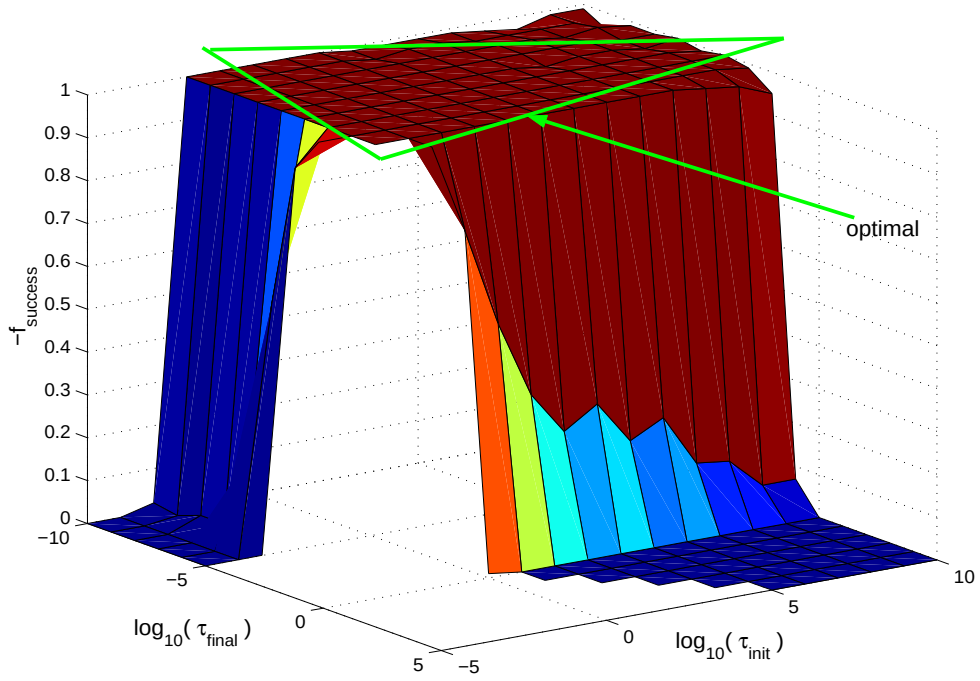


Figure 17: Surface plot of grid search's fitness function (turned negative for better visualization) $-f_{success}(\log_{10}(\tau_{init}), \log_{10}(\tau_{final}))$ when minimizing unrotated 5-dimensional Rastrigin's functions utilizing CO mutation.

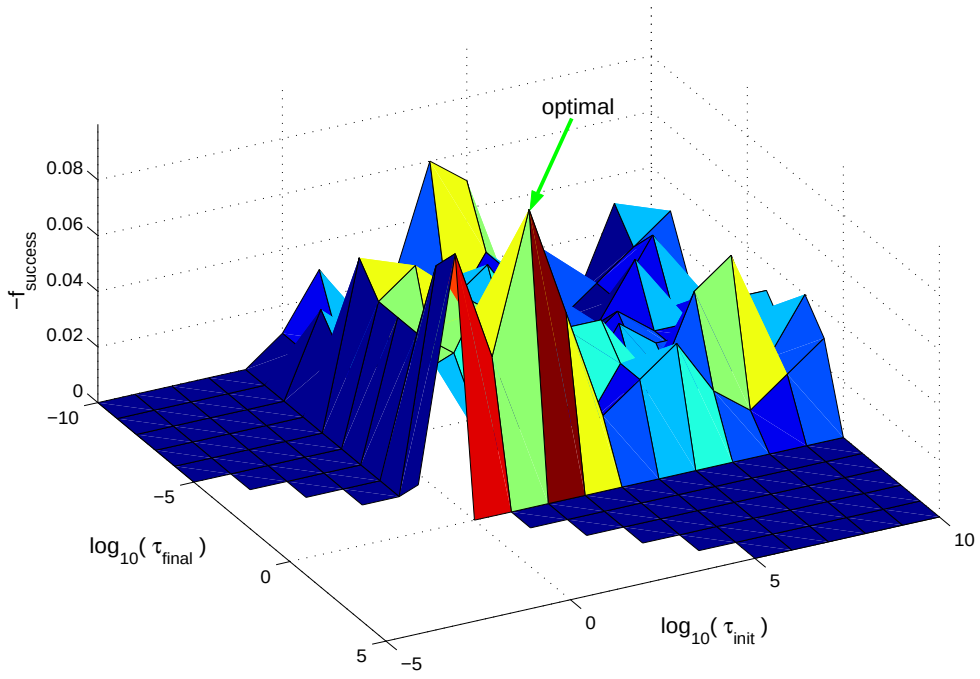


Figure 18: Surface plot of grid search's fitness function (turned negative for better visualization) $-f_{success}(\log_{10}(\tau_{init}), \log_{10}(\tau_{final}))$ when minimizing randomly rotated 5-dimensional Rastrigin's functions utilizing CO mutation.

Mut. Op.	$f_{rastrigin}$ kind	$f_{success}(\mathbf{lg}(\theta_{init}), \mathbf{lg}(\theta_{final}))$			$f_{fitavg}(\mathbf{lg}(\theta_{init}), \mathbf{lg}(\theta_{final}))$			**see Fig.**
		$\lg(\theta_{init})$	$\lg(\theta_{final})$	$f_{success}$	$\lg(\theta_{init})$	$\lg(\theta_{final})$	f_{fitavg}	
CO	**unrot.**	div.	div.	-1	div.	div.	0	17
CO	**rot.**	2	-1	-0.1	0	-1	1.67	18, 19
GO	**unrot.**	1	-1	-0.15	0	0	1.47	20
GO	**rot.**	0	-1	-0.13	0	0	1.37	-
CM1	**unrot.**	1	0	-0.04	0	0	2.91	21
CM1	**rot.**	0	0	-0.04	1	0	2.98	-

Table 10: Results of the grid search procedure: The 1^{st} and 2^{nd} column contain the mutation operator's name and the kind of Rastrigin's functions (rotated/unrotated). In the 3^{rd}-5^{th} columns, optimal initial and final parameter values (using a $\log_{10}$ abbreviated as "lg") w.r.t. $f_{success}$ and the resulting value for $f_{success}$ is given, and in the 6^{th}-8^{th} columns, the same for f_{fitavg}.

A comparison of Figures 17 and 18, illustrating $-f_{success}$[19], makes clear the following: On separable, unrotated Rastrigin's functions, a (1+1)-Evolution Strategy employing CO mutation is able to attain the global minimum in all of the 100 runs ($p_{success} = 1$, i.e. $-f_{success} = 1$) for various choices $(\tau_{init}, \tau_{final})$ as step size parameters whereas on randomly rotated Rastrigin's functions, the smallest possible fitness value $f_{success} = -1$ is never reached. Furthermore, from Figure 18 it can be seen that fitness $f_{success}$ gets noisy for rotated Rastrigin's functions, but approximately decreases with increasing distance to the optimum as opposed to Figure 17. These facts imply that CO mutation operators on separable functions like Rastrigin's are much more flexible with respect to parameters τ_{init} and τ_{final} – or in other words – on non-separable or rotated functions are much more sensitive to the choice of step size parameters. Due to the coarse granularity of the grid, no statements about the exact optimality of parameters should be made.

Fitness function $f_{success}$ turns out to be quite noisy when utilizing CO mutation on rotated Rastrigin's functions or the two other mutation operators, because only few successes can be achieved even in the optimal case. This means that larger differences in fitness for neighboured parameter points occur frequently which makes it difficult to analyze the global behaviour of $f_{success}$. The smoother f_{fitavg} helps out in this situation, applying a log-scale $\log_{10}(f_{fitavg})$ in addition to emphasize differences at small values. This is not practicable for CO mutation on unrotated Rastrigin's functions because the optimal plateau then would lie at $\log_{10}(f_{fitavg}) = -\infty$. But from Figure 19, it can be seen clearly that the smallest possible fitness value $f_{fitavg} = 0$ is not reached and f_{fitavg} changes remarkably from grid point to grid point for CO-based Evolution Strategies on rotated Rastrigin's functions.

As a control how Evolution Strategies employing other mutation operators react to the choice of step size parameters, GO and CM1 mutation are tested by this grid search procedure on unrotated Rastrigin's functions, focusing here on the less noisy f_{fitavg}. There are no large differences in performance apparent between unrotated and rotated Rastri-

[19] turned negative for better readability

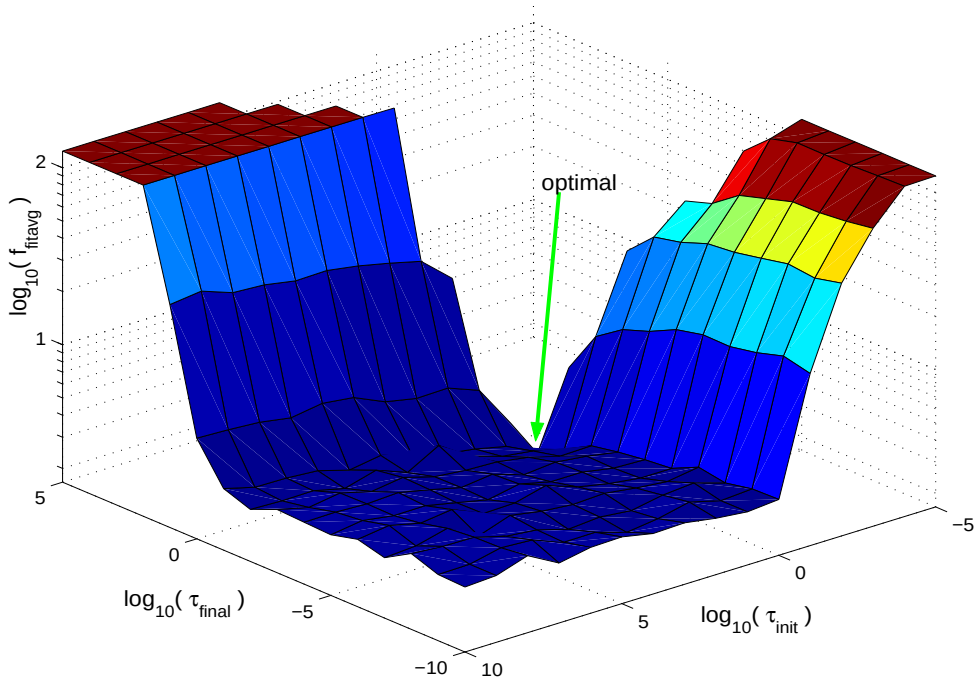


Figure 19: Log-scaled surface plot of grid search's fitness function $\log_{10}(f_{fitavg}(\log_{10}(\tau_{init}), \log_{10}(\tau_{final})))$ when minimizing randomly rotated 5-dimensional Rastrigin's functions utilizing CO mutation.

gin's functions. From Figures 20 and 21 and taking into account optimal values given in Table 10, it can be seen again that no perfect 100 runs occur and f_{fitavg} is rather sensitive to step size parameter choice compared to CO-based Evolution Strategies on separable, unrotated Rastrigin's functions.

Same investigations were done for the 3-dimensional Rastrigin's function resulting in quite similar findings, which are not provided here. However, it shall be remarked that for each of the three considered mutation operators, there exists a step size interval $[\theta_{init}, \theta_{final}]$ such that $p_{success} = 1$ for 100 (1+1)-Evolution Strategy runs on unrotated but as well on randomly rotated Rastrigin's functions. This may put p-values for 3-dimensional Rastrigin's functions in Section 5.3 into question because not the individual optimal parameters – which would make 100% success possible for all mutation operators – are used for step size adaptation.

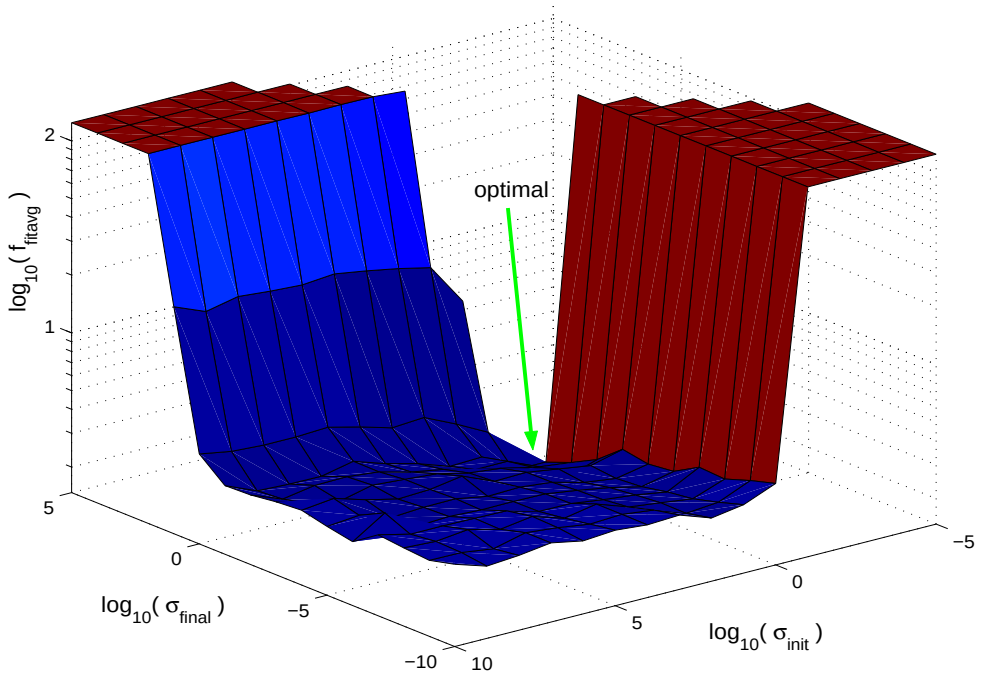


Figure 20: Log-scaled surface plot of grid search's fitness function $\log_{10}(f_{fitavg}(\log_{10}(\sigma_{init}), \log_{10}(\sigma_{final})))$ when minimizing unrotated 5-dimensional Rastrigin's functions utilizing GO mutation.

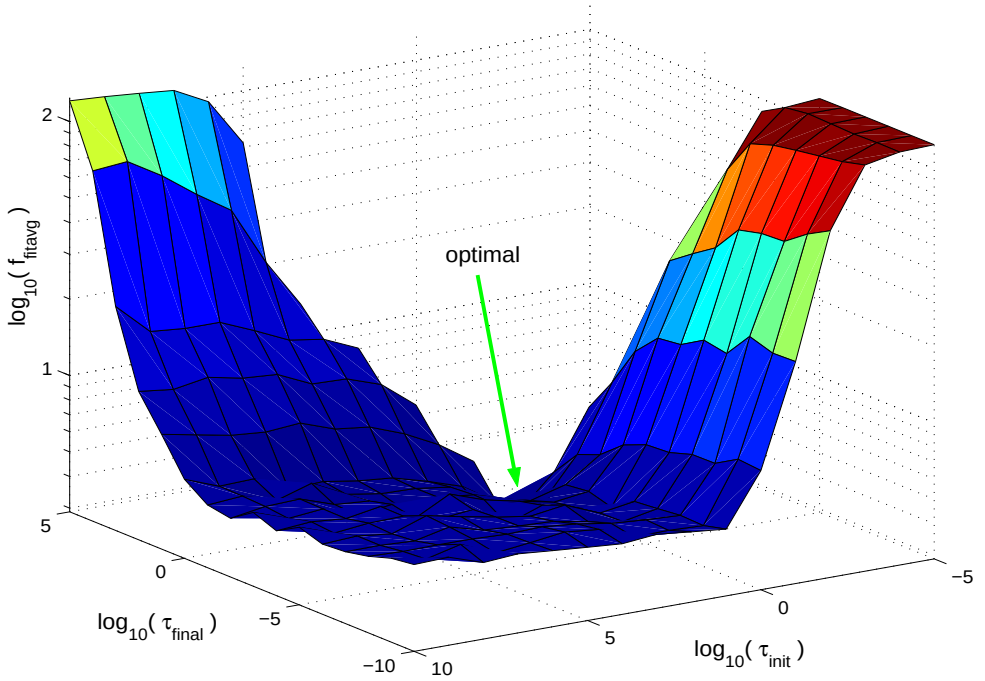


Figure 21: Log-scaled surface plot of grid search's fitness function $\log_{10}(f_{fitavg}(\log_{10}(\tau_{init}), \log_{10}(\tau_{final})))$ when minimizing unrotated 5-dimensional Rastrigin's functions utilizing CM1 mutation.

6 Conclusions

In this semester thesis, the advantage in performance of a non-spherical Cauchy mutation operator – i.e. a particular heavy-tail distribution – in Evolutionary Algorithms on certain fitness functions was investigated. The focus was on (1+1)-Evolution Strategies with a deterministic cooling scheme for step size adaptation (Section 2). Two different fitness functions were considered: Rastrigin's function as an instance of a *separable* function and randomly rotated Rastrigin's functions as corresponding non-separable counterpart.

In Section 3, various distributions for mutation were introduced and compared to each other concentrating on the shape of their joint densities and the densities of corresponding random vector norm distributions. Characteristics of separable functions in general and Rastrigin's function in more detail were provided in Section 4.

In Section 5, empirical evidence for predictions made in Sections 3 and 4 was given and furthermore, enabled us to decide about the hypotheses stated in Section 1. (1+1)-Evolution Strategies employing four different mutation operators were compared: the non-spherical Cauchy (CO), the spherical Cauchy distribution with n-dimensional norm (CMn), the original Gaussian (GO) and the spherical Cauchy distribution with fixed 1-dimensional norm (CM1). Their performance on Rastrigin's functions was measured by the final fitness values and success probability obtained by 50 runs. The significance of observed differences in performance was ascertained by ranksum and χ^2 tests respectively. From these results, the following became clear:

Evolution Strategies employing non-spherical Cauchy mutation always reached the global minimum on the separable unrotated Rastrigin's function, as opposed to the other three mutation operators which generally performed worse. Such significant difference could not be observed for non-separable rotated Rastrigin's functions. These findings clearly corroborate our first hypothesis. The higher the dimensionality of the underlying search space, the more obvious were discrepancies between the non-spherical Cauchy mutation operator and others. In particular, it can be inferred, that it is not the larger expectation of norms of non-spherical Cauchy random vectors that produce significant gain on performance, but rather the preference of certain directions through the search space. Therefore, the second hypothesis can be rejected.

Another difference in performance emerged for the spherical Cauchy mutation distribution with 1-dimensional norm: (1+1)-Evolution Strategies using this mutation operator generally performed worse than such using mutators having an n-dimensional norm, independent of the kind of fitness function, but observable especially in higher dimensionalites.

Recapitulating: Evolutionary Algorithms employing non-spherical Cauchy mutation have advantage over other mutation operators when optimizing separable functions. For arbitrary test functions however, this is not the case. The gain in performance cannot be caused by the heavy tails of the Cauchy distribution as proposed in [1], rather it is a consequence of the specific asymmetry of the non-spherical Cauchy density.

Future Work

The scope of this semester thesis covers the empirical investigation of (1+1)-Evolution Strategies employing a set of Gaussian and Cauchy mutation operators on the Rastrigin's function as an instance of an additively decomposable function, making use of a simple deterministic step size adaptation scheme.

These considerations could be extended in the following ways:

- Set of test functions: Applying Evolutionary Algorithms on further separable functions, likewise of other kind such as productively decomposable ones.
- Kinds of Evolutionary Algorithms: Comparing performances of various mutation operators employed by Evolutionary Programming or general $(\mu + \lambda)$-Evolution Strategies.
- Forms of step size adaptation: Empirical testing rather emphasizing the optimality of step size parameters of our chosen cooling scheme, but also checking self-adaptive schemes properly.

Another open issue is the exact magnitude of the impact which raised norms of a Cauchy random vector have on the performance of Evolutionary Algorithms without gaining advantage by asymmetry. Furthermore, findings about a closed density formula for Cauchy random vectors' norms are missing.

Acknowledgements

I would like to thank Prof. Dr. Petros Koumoutsakos, Dr. Anne Auger and Dr. Nikolaus Hansen for giving me the opportunity to work out a semester thesis in the area of Bio-Inspired Computing and Optimization, one of my favourite subjects during my studies of Computer Science. Dealing with this topic in the scope of a semester thesis has enabled me to widely broaden knowledge about the variety of mutation utilized in Evolutionary Algorithms and their corresponding analysis, and learn how empirical research in Scientific Computing is done properly.

Special thanks goes to Anne and Niko for advising me, Anne for providing me with mathematical tricks and Niko especially for showing me how to tune Evolution Strategies and for the consequent demonstration what a log-scale is intended for.

References

[1] X. Yao and Y. Liu. "Fast Evolution Strategies". *Control and Cybernetics*, Vol.26(No.3):pages 467–496, 1997. Polish Academy of Sciences, Systems Research Institute.

[2] A. Obuchowicz. "Multidimensional mutations in evolutionary algorithms based on real-valued representation". *International Journal of Systems Science*, Vol.34(No.7):pages 469–483, June 2003. Taylor and Francis Group.

[3] E.W. Weisstein. "Cauchy Distribution". *http://mathworld.wolfram.com/ CauchyDistribution.html*, 2005. MathWorld - a Wolfram Web Resource.

[4] G.P. Michon. "Geometry and Topology - Final Answers: What is the formula for a hyper-volume of a four-dimensional sphere?". *http://home.att.net/ numericana/answer/geometry.htm*, 2005. www.numericana.com - online companion for book Numericana by G.P. Michon.

[5] E.W. Weisstein. "Chi-squared Distribution". *http://mathworld.wolfram.com/ ChiSquaredDistribution.html*, 2005. MathWorld - a Wolfram Web Resource.

[6] E.W. Weisstein. "Chi Distribution". *http://mathworld.wolfram.com/ ChiDistribution.html*, 2005. MathWorld - a Wolfram Web Resource.

[7] E.W. Weisstein. "Half-normal Distribution". *http://mathworld.wolfram.com/ HalfNormalDistribution.html*, 2005. MathWorld - a Wolfram Web Resource.

[8] E.L. Lehmann. *"Nonparametrics - Statistical Methods Based on Ranks"*. Holden-Day inc., 1975. San Francisco.

A Appendix

A.1 Matlab Source Code for (1+1)-Evolution Strategies

A.1.1 Non-spherical Cauchy Mutation Distribution

```
function [ xbest , nGen , fxs , taus, firstSuccessfulStepsize , lastSuccessfulStepsize ,
     successfulStepsizes ] = cauchyES1plus1(stepsizeAlpha,stepsizeInit,xInit,maxGen,rotationMatrix,
     fGlobMin,tol,testFuncHandle)

% input arguments:
% testFuncHandle: function to be minimized as function handle testfuncHandle
% fGlobMin: function value of the global minimum
% tol: tolerance for stopping criterion (fx-fGlobMin > tol)
% rotationMatrix: the function will be rotated by this orthogonal nxn-matrix
% maxGen: maximal number of generations before giving up
% xInit: point(column-vector) in search space, where optimization will begin
% stepsizeInit: the initial value of the stepsize
% stepsizeAlpha: the factor for cooling down the stepsize
% output arguments:
% xbest: point in search space when this algorithm has stopped
% nGen: number of generations used when this algorithms has stopped
% fxs: large row-vector containing all the function values fx of the search
% taus: large row-vector containing all subsequent stepsizes
% firstSuccessfulStepsize: stepsize when the first fx-improvement occured
% lastSuccessfulStepsize: stepsize when the last fx-improvement occured
% successfulStepsizes: all stepsizes when fx-improvements occured

randn('seed',cputime);
[ nDim , one ] = size(xInit);
alpha = stepsizeAlpha;
tau = stepsizeInit;
fxs = ones(1,maxGen);
taus = ones(1,maxGen);
successfulStepsizes = ones(1,maxGen);
succStepCount = 1;
genCount = 1;
x = xInit;  % will be rotated in next line!!!
fx = feval(testFuncHandle,rotationMatrix*x);

while((genCount < maxGen+1) && (fx-fGlobMin > tol))
    deltax = randn(nDim,1) ./ randn(nDim,1);
    xx = x + tau * deltax;
    fxx = feval(testFuncHandle,rotationMatrix*xx);
    if( fxx < fx)
        fx = fxx;
        x = xx;
        successfulStepsizes(1,succStepCount) = tau;
        succStepCount = succStepCount + 1;
    end;
    fxs(1,genCount) = fx;
    taus(1,genCount) = tau;
    tau = alpha * tau;
    genCount = genCount + 1;
end;
nGen = genCount-1;
fxs = fxs(1,1:nGen);
taus = taus(1,1:nGen);
i=1;
while((i<nGen-1)&&(fxs(1,i) == fxs(1,i+1)))
    i = i + 1;
end;
firstSuccessfulStepsize = taus(1,i+1);
i=nGen;
while((i>1)&&(fxs(1,i) == fxs(1,i-1)))
    i = i - 1;
end;
lastSuccessfulStepsize = taus(1,i);
xbest = x;
successfulStepsizes = successfulStepsizes(1,1:succStepCount-1);
```

A.1.2 Spherical Cauchy Mutation Distribution with n-dimensional Norm

```
function [ xbest , nGen , fxs , taus , firstSuccessfulStepsize , lastSuccessfulStepsize ,
     successfulStepsizes ] = sphericalcauchyNdimNormES1plus1(stepsizeAlpha,stepsizeInit,xInit,maxGen,
     rotationMatrix,fGlobMin,tol,testFuncHandle)

% in general all computations are equivalent to cauchyES1plus1.m except sampling:

while((genCount < maxGen+1) && (fx-fGlobMin > tol))
    % sample unit vector onto unit hypersphere:
    unifRandDirection = randn(nDim,1);
    unitUnifRandDirection = unifRandDirection/norm(unifRandDirection);
    % sample jumplength:
    deltaradian = norm(randn(nDim,1)./randn(nDim,1));
    radian = tau * deltaradian;
```

```
    xx = x + radian * unitUnifRandDirection;
    fxx = feval(testFuncHandle,rotationMatrix*xx);
    ...
```

A.1.3 Spherical Cauchy Mutation Distribution with 1-dimensional Norm

```
function [ xbest , nGen , fxs , taus , firstSuccessfulStepsize , lastSuccessfulStepsize ,
     successfulStepsizes ] = sphericalcauchy1dimNormES1plus1(stepsizeAlpha,stepsizeInit,xInit,maxGen,
     rotationMatrix,fGlobMin,tol,testFuncHandle)

% in general all computations are equivalent to cauchyES1plus1.m except sampling:

while((genCount < maxGen+1) && (fx-fGlobMin > tol))
    % sample unit vector onto unit hypersphere:
    unifRandDirection = randn(nDim,1);
    unitUnifRandDirection = unifRandDirection/norm(unifRandDirection);
    % sample jumplength:
    deltaradian = abs(randn/randn);
    radian = tau * deltaradian;
    xx = x + radian * unitUnifRandDirection;
    fxx = feval(testFuncHandle,rotationMatrix*xx);
    ...
```

A.1.4 Gaussian Mutation Distribution

```
function [ xbest , nGen , fxs , sigmas , firstSuccessfulStepsize , lastSuccessfulStepsize ,
     successfulStepsizes ] = gaussES1plus1(stepsizeAlpha,stepsizeInit,xInit,maxGen,rotationMatrix,
     fGlobMin,tol,testFuncHandle)

% input arguments:
% testFuncHandle: function to be minimized as function handle testfuncHandle
% fGlobMin: function value of the global minimum
% tol: tolerance for stopping criterion (fx-fGlobMin > tol)
% rotationMatrix: the function will be rotated by this orthogonal nxn-matrix
% maxGen: maximal number of generations before giving up
% xInit: point(column-vector) in search space, where optimization will begin
% stepsizeInit: the initial value of the stepsize
% stepsizeAlpha: the factor for cooling down the stepsize
% output arguments:
% xbest: point in search space when this algorithm has stopped
% nGen: number of generations used when this algorithms has stopped
% fxs: large row-vector containing all the function values fx of the search
% sigmas: large row-vector containing all subsequent stepsizes
% firstSuccessfulStepsize: stepsize when the first fx-improvement occured
% lastSuccessfulStepsize: stepsize when the last fx-improvement occured
% successfulStepsizes: all stepsizes when fx-improvements occured

randn('seed',cputime);
[ nDim , one ] = size(xInit);
alpha = stepsizeAlpha;
sigma = stepsizeInit;
fxs = ones(1,maxGen);
sigmas = ones(1,maxGen);
successfulStepsizes = ones(1,maxGen);
succStepCount = 1;
genCount = 1;
x = xInit;  % will be rotated in next line!!!
fx = feval(testFuncHandle,rotationMatrix*x);

while((genCount < maxGen+1) && (fx-fGlobMin > tol))
    deltax = randn(nDim,1);
    xx = x + sigma * deltax;
    fxx = feval(testFuncHandle,rotationMatrix*xx);
    if( fxx < fx)
        fx = fxx;
        x = xx;
        successfulStepsizes(1,succStepCount) = sigma;
        succStepCount = succStepCount + 1;
    end;
    fxs(1,genCount) = fx;
    sigmas(1,genCount) = sigma;
    sigma = alpha * sigma;
    genCount = genCount + 1;
end;
nGen = genCount-1;
fxs = fxs(1,1:nGen);
sigmas = sigmas(1,1:nGen);
i=1;
while((i<nGen-1)&&(fxs(1,i) == fxs(1,i+1)))
    i = i + 1;
end;
firstSuccessfulStepsize = sigmas(1,i+1);
i=nGen;
while((i>1)&&(fxs(1,i) == fxs(1,i-1)))
    i = i - 1;
end;
lastSuccessfulStepsize = sigmas(1,i);
xbest = x;
successfulStepsizes = successfulStepsizes(1,1:succStepCount-1);
```

A.1.5 Additional Functions Needed

Computing Cooling Factor for Stepsize Adaptation:

```
function alpha = computeOptimalStepSize(maxGen,sigmaEnd,sigmaStart)

% solves the equation:  sigmaStart * alpha^maxGen = sigmaEnd
% with sigmaStart >> sigmaEnd and 0 << alpha < 1
% sigmaStart = sigma_0 ; sigmaEnd = sigma_maxGen

alpha = ( sigmaEnd / sigmaStart )^( 1 / maxGen);
```

Generating Random Rotation Matrices:

```
function rotmatrix = createRandomRotationMatrix(nDim)

% rotmatrix contains nDim orthonormal column vectors
% randVectors contains nDim normalized standard Gaussian random vectors

randVectors = randn(nDim,nDim);
for i=1:nDim
    randVectors(:,i) = randVectors(:,i)/norm(randVectors(:,i));
end;

rotmatrix = zeros(nDim,nDim);
rotmatrix(:,1) = randVectors(:,1);

for i=2:nDim
    curvec = randVectors(:,i);
    for j=1:(i-1)
        curvec = curvec - rotmatrix(:,j)'*randVectors(:,i)*rotmatrix(:,j)/(rotmatrix(:,j)'*rotmatrix(:,
            j));
    end;
    rotmatrix(:,i) = curvec/norm(curvec);
end;
```

Testfunction - Rastrigin's Function:

```
function rastr = rastriginsfunction(x)

% is a complicated but separable quadratic fitness function

rastr = sum(x.*x - 10*cos(2*pi*x) + 10);
```