Table of Contents

[1. Introduction 2](#_Toc22740054)

[2. Background 4](#_Toc22740055)

[2.1 (1+1)-ES 5](#_Toc22740056)

[2.2 (1 + λ) - ES & (1, λ) – ES 7](#_Toc22740057)

[2.3.1 (µw, λ) - CMA-ES 7](#_Toc22740058)

[2.3.2 LS-CMA-ES 13](#_Toc22740059)

[2.3.3 LR-CMA-ES 15](#_Toc22740060)

[2.3.4 IPOP-CMA-ES 16](#_Toc22740061)

[2.3.5 (µ + λ) - CMA-ES 16](#_Toc22740062)

[2.4 Common Random Numbers (CRNs) 17](#_Toc22740063)

[3. Methodology 17](#_Toc22740064)

[References 18](#_Toc22740065)

# Introduction

Evolution Strategies is a sub field of evolutionary computation that performs function optimization in a continuous domain. Interestingly, Evolution Strategies was not initially designed to find the optimum of complex functions. It was originally used for the design and analysis of experiments that require frequent variable adjustments with the goal of obtaining an optimal state [1]. There are many algorithms encompassed in the field of Evolution Strategies and they are often referred to as black-box optimization algorithms. The term “black-box” means that the algorithm is unaware of the problem domain knowledge. In other words, the algorithm does not know, and does not need to know, the optimal solution of the objective function or its characteristics. The goal of black-box optimization algorithms in the field of Evolutionary Algorithms is to minimize an objective function in a continuous domain as shown in Equation 1.

Equation 1

This paper is focused on improving one of the state-of-the-art ES algorithms, IPOP-CMA-ES. However, other ES algorithms will be discussed, namely the (1+1)-ES, (1+λ)-ES and the (1, λ)-ES, to help explain the motivation behind the proposed algorithm variant, DC-CMA-ES. The proposed algorithm variant looks to improve the evolutionary process within IPOP-CMA-ES. This modified evolutionary process provides a slightly unorthodox elitist method of generating solutions and computing the mean. Currently, the use of elitism in the ES field has the potential to negatively affect an algorithm’s convergence towards the global optimum, especially in multimodal functions and higher dimensions. This modification is a blend, influenced by some of the different properties of elitism in ES algorithms such as the (1+1)-ES, (1+λ)-ES, (1, λ)-ES and (µ + λ) – IPOP-CMA-ES. For both the (1+1)-ES and IPOP-CMA-ES, the methods for updating the mean are predefined and constant throughout the entire run of the algorithm. DC-CMA-ES allows the algorithm to indirectly determine how the mean is computed based on the current success of elitism. This modified evolutionary process is referred to as Dual-Center Sampling (DCS). As a result, DC-CMA-ES has been shown to have better performance over IPOP-CMA-ES on a diverse function test suite.

# Background

Before diving into the world of Evolutionary Strategies, knowledge of multivariate normal distributions and covariance matrices is extremely necessary and will be discussed beforehand to aid in better understanding of the topics ahead. A multivariate normal distribution is an n-dimensional normal distribution which can be denoted as *N(m,C)*. Here, *m* is the mean vector of length *n* (the “favorite” point in space from which sampling of new solutions takes place) and *C* is the *n x n* symmetric positive-definite covariance matrix [2]. A close up of a womans face

Description automatically generated

Figure 1 – Image taken from a CMA-ES Tutorial by (Auger & Hansen, 2013)

The first image on left of Figure 1 shows a multivariate normal distribution with mean of 0 and standard deviation as the identity matrix. The identity matrix is responsible for the circular shape around the mean, while σ is a scalar which can shrink or grow the search space. The middle image shows another multivariate normal distribution this time with a standard deviation of *D2*, where *D* is a diagonal matrix. Using this diagonal matrix to transform a sample from *N(0,I)* can result in “stretching” or “flattening” of the search space. The last image on the right of Fig. 1 uses the covariance matrix as it’s standard deviation. Transforming *N(0,I)* using the covariance matrix can also result in stretching or flattening, but also changes the direction of the search space.

## (1+1)-ES

The (1+1) – ES was the first Evolutionary Strategies algorithm for solving black-box optimization problems. This algorithm was developed in the 1960s by three students of the Technical University of Berlin, Hans-Paul Schwefel, Ingo Rechenberg and Peter Bienert [3]. This algorithm is the simplest in the field of ES. This algorithm generates one offspring from one parent, by utilizing both the multivariate normal distribution N(0,I) and a scalar σ known as the step-size for mutation (Shown in Equation 2)

Equation 2

If the new offspring’s fitness is better than the parent’s then itself becomes the new parent for the next generation, otherwise the parent remains the same. If the parent was replaced, then that would count as a successful mutation, otherwise it would count as a failure. The algorithm archives or keeps track of the number of successes and failures. This archive is used in the self-adaptation of the step-size σ. This self-adaptation method is known as the 1/5th success rule. This self-adaptation method is based on the observation that the fastest convergence to the global optimum is achieved when approximately 1/5th of all mutations are successful, i.e. when the parent is replaced with the better offspring 1/5th of the time [3]. If the current success-rate is 1/5 then the value of σ stays the same. However, if the current success-rate is less than 1/5 then σ is increased by a constant factor, c. On the contrary, if the current success-rate is greater than 1/5 then σ is decreased by a constant factor, c. Therefore, to obtain the new step-size:

Equation 3

The value recommended for c in Equation 3 is and was recommended by [4] using theoretical methods. The probability of success, ps, is measured based the 10n most recent mutations, where n is the problem dimension. The equation used to calculate the probability of success can be shown via the pseudocode of the (1+1) – ES algorithm in Figure 2.

Figure 2 – (1+1)- ES taken from [3]

A screenshot of a cell phone

Description automatically generated

## 2.2 (1 + λ) - ES & (1, λ) – ES

The only difference between the (1 + λ) – ES and the (1 + 1) – ES is that instead of only generating one offspring from the parent, (1 + λ) – ES will generate λ offspring and replace the parent with the best offspring provided it is more fit than the current parent. However, in the (1, λ) – ES, λ offspring are also produced every generation, but the current parent is always replaced by the best offspring.

## 2.3.1 (µw, λ) - CMA-ES

## 

Covariance Matrix Adaptation – Evolution Strategies is one of the most powerful black box optimization techniques in the field of Evolutionary Strategies. What makes CMA-ES unique is the sampling of a covariance matrix to determine the mutation distribution, and using a method called step-size cumulation to determine the scale of the mutation distribution. The covariance matrix determines the direction of the search while the step size determines “how far” the search would be in that direction [2].

However, in CMA-ES the covariance matrix *C* undergoes an eigen decomposition. This decomposes the matrix into eigenvectors (which determine the direction of the search space) and eigenvalues (which determine the scale of the eigenvectors). Therefore, the process of sampling new solutions is shown below in Equation 4 [3].

Equation 4

In Equation 4, the eigenvectors and eigenvalues BD are used to shape the spherical noise, z which is sampled from the multivariate normal distribution. ⟨x⟩ and ⟨y⟩ are updated using a method similar to an EDA, known as weighted-intermediate recombination. Weighted-intermediate recombination uses a set of µ weights w1 ≥ w2 ≥ . . . ≥ wµ with for generating the new ⟨x⟩ and new ⟨y⟩ [3]:

Equation 5

≡

Equation 6

Equation 5 shows how weighted intermediate recombination is used to determine the new point in space from which the next generation of sampling will take place. A weighted average was taken of the best µ solutions (xi’s) to create the new center. It’s important to note later that, the weighted average of the shaped noise, . In Equation 6, the best µ shaped-noise vectors (yi’s) associated with the best µ xi solutions from Equation 5, also under go weighted intermediate recombination to find the average of the best mutation steps. Having covered the concepts such as multivariate normal distributions, covariance matrices and weighted intermediate recombination, the algorithm will be discussed in more detail starting with the cumulation methods for step-size adaptation (updating σ) and updating the covariance matrix. These cumulation paths can be thought of as history vectors. They accumulate parameter information over several generations [3]. In Equation 7, the first term is the decay factor where the cumulation time parameter cσ<1 and approximately 1/ cσ [5]. This means as new parameter information accumulates, the older information is gradually weened out of the cumulation path.

Equation 7

The expression under the square root of Equation 7, is a normalization constant associated with the first term, . The variance effective selection mass µ*eff* is necessary for adapting the strategy parameters [3]. It is important to note that BD-1BT = C1/2 as this inverse matrix reverses the “shaping” of which is done in Equation 6 because ||pσ|| is then compared with the expected length of a normally distributed random vector, E(||N(0,I)||) in the step-size update in Equation 6.

Equation 8

The idea behind Equation 6 is that if the length of pσ is greater than the expected length of a normally distributed random vector then the value of σ will increase. For better understanding, consider Figure 3 below which illustrates the length of the center (mean vector) in the generation sequence. The black arrow heads represent the position of the center at a specific generation, the arrows with white arrow heads represent the distance between the original center and the best center so far, and the arrow tails represent how much the center has moved.

A screenshot of a social media post

Description automatically generated

Figure 3 - Image taken from a CMA-ES Tutorial by [2]

In the left segment of Figure 3, you will notice that center moves around a lot less than the other segments. In this case the step-size is decreased to continue a tighter search around the current center. However, in the right segment of Figure 3, the opposite happens. The step size is increased because the generation-steps are much longer and are relatively in the same direction. Therefore, in this case a bigger step-size would be more appropriate because it would be faster to make bigger steps in that direction rather than multiple small steps in the same direction.

Now we will move on to the steps of the covariance path update and the covariance matrix update.

Equation 9

⟨y⟩

Equation 9 represents the covariance path update. Once again, the first term in this equation is the decay factor where the cumulation time parameter cc<1 and approximately 1/ cc [5]. In the second term, hσ is either 1 or 0 based on Equation 10, where t is the generation number and n is the problem dimension [3].

Equation 10

Like the previous path update, the expression with the square root is a normalization constant associated with the first term, . The purpose of hσ is to choose whether to update pc based on information about the current generation t and the value of ||pc|| [3]. It’s also important to note that the length of a multivariate normally distributed random vector can be approximated by the gamma function shown in Equation 11.

Equation 11

The variable µeff is the variance effective selection mass which is defined in Equation 12.

Equation 12

After the covariance path update, the next step of the algorithm is the covariance matrix update which is shown in Equation 13.

Equation 13

The covariance matrix update is a weighted matrix addition with 3 terms. The first term is the decay factor of the current covariance matrix. The second term is known as the rank-one update, which uses the covariance path and its transpose to form another matrix. According to [2], “the rank-one update uses the evolution path and reduces the number of function evaluations to learn straight ridges from O(n2) to O(n). The third term is known as the rank-µ update, also known as the weighted empirical covariance matrix, was added to the update equation to accommodate large population sizes. In essence, the covariance matrix is updated to increases the likelihood of successful steps to occur in the next generation [2]. The following equations show how the strategy parameters from the algorithm are calculated:

cc is the parameter for the cumulation time of pc which is approximately 1/pc.

cσ is the parameter for the cumulation time of pσ which is approximately 1/pσ.

dσ is a dampening parameter that determines the change rate of the global step-size, σ during the step-size update.

c1 is the parameter that determines the change rate of the covariance matrix with respect to the rank-one update.

cµ is the parameter that determines the change rate of the covariance matrix with respect to the rank-one update.

wi:λ are the set of logarithmic weights used for weighted intermediate recombination of the sorted population to calculate the mean point. This method is also sometimes referred to as flattening.

Figure 4 – **(µw, λ) - CMA-ES Algorithm** [3]

A screenshot of a cell phone

Description automatically generated

## 2.3.2 LS-CMA-ES

The LS-CMA-ES algorithm is a CMA-ES variant that updates the covariance matrix based on an inverse Hessian matrix that is estimated via a solution to the appropriate least-squares estimation problem. To solve this least-squares estimation problem, the algorithm collects an archive of tuples (x, f(x)) where x is a candidate solution and f(x) is the fitness value of that candidate solution [3]. As a result, the least-squares estimation problem is defined by the following minimization task:

Equation 14

The result of this minimization in Equation 14 produces two estimators, for the gradient and for the Hessian matrix. The Taylor series expansion up to the quadratic term only provides an estimation of the true fitness landscape at x0­, it is of interest to obtain an error measure of the estimate to determine if should be used for the covariance matrix update. The error measure is shown in Equation 15 [3].

Equation 15

Since solving Equation 14 and computing H-1 uses numerical methods algorithms that have a time complexity of *O*(*n*6), executing this step at each generation is not ideal for performance. As a result, LS-CMA-ES has two different modes known as LS and CMA, used for performing the covariance matrix update [3].

In LS mode, Equation 15 is executed once every *nupd* generations. Additionally, if the error measure falls below a defined threshold, *Q*t, the covariance matrix until the new update after another *nupd* generations. If Q > Qt, the algorithm switches into CMA mode. In CMA mode the covariance matrix is updated using the rank-one update (the 2nd term in Equation 13). In this algorithm, sampling offspring from the parent is shown in Equation 16 below.

Equation 16

A global step size σ is used and updated via mutative step size adaptation. For example, if b is the index of the best offspring after selection, then the new global step size is updated as Finally, the covariance matrix is updated using rank-one update (the 2nd term in Equation 13). It is important to note that the evolution path pc is still updated regardless of which mode the algorithm is in so that the covariance matrix is always up-to-date before switching into, or staying in CMA mode [3]. The recommended exogenous strategy parameters are shown below:

## 2.3.3 LR-CMA-ES

LR-CMA-ES is an extension of (µw, λ) - CMA-ES that introduces local restart conditions. The idea of restarting the algorithm is to avoid wasting function evaluations when the algorithm is facing stagnation in its optimization process. This algorithm provides five restart conditions for identifying stagnation, in which a new run of regular (µw, λ) - CMA-ES will commence [3]. There are two tolerance values, T*x*= σ10-12 and T*f* = 10-12, used within the restart conditions. The first restart condition, *equalfunvalhist* is satisfied if either the best fitness values of the last generations are the same or the difference between the best fitness value and the weakest fitness values is small than T*x* [3]. The second restart condition, *TolX*, is satisfied if for a vector v = σpc , T*x* , where *i* ϵ {1,2,…,n}. The third restart condition, *noeffectaxis*, looks at the main coordinate axes formed by C. This condition is satisfied when, 0, where γi is the ith eigenvalue and ui is the ith eigenvector of C respectively, where . The fourth restart condition, *noeffectcoord*, also looks at the coordinate axis and is satisfied when . The fifth and final condition, *conditioncov*, is satisfied when the condition number of the covariance matrix exceeds 1014 [3].

## IPOP-CMA-ES

IPOP-CMA-ES is an extension of LR-CMA-ES with one small but very effective change. Should there be a restart of the (µw, λ) - CMA-ES, then the population size of the next run of (µw, λ) - CMA-ES is multiplied by a certain factor η, commonly set as η = 2 [3].

## (µ + λ) - CMA-ES

(µ + λ) - CMA-ES is an extension of regular (µw, λ) - CMA-ES where the selection method is altered to perform elitism. Elitism is a well-known method used in the selection process of many Evolutionary Algorithms. What elitism does is it includes a certain number of the best selected offspring from the previous generation, in the selection processes of the current generation. In (µw, λ) - CMA-ES, we selection the best µ offspring from the population, λ. Therefore, with elitism we add the µ offspring from the previous generation to the λ offspring of the current generation, before performing a sort based on fitness and selecting the best µ offspring from the sorted µ + λ population. The main idea behind elitism in Evolutionary Algorithms is to ensure that there are always a set of the fittest solutions included in the population every generation.

## (1+1)-Cholesky-CMA-ES

The main property of this CMA-ES variant is that the covariance matrix is implicitly updated without using an eigen decomposition. As the name suggests, the algorithm uses a Cholesky decomposition instead. This method reduces the computational complexity in each generation from *O*(*n3*) to *O*(*n*2) [3]. [6] proves that the Cholesky factors A can be updated without explicit knowledge of the covariance matrix. The theorem will be discussed without the proof. However, consider this lemma which is required for the proof of the following theorem [3]. For any vector and , the following equation holds:

Equation 17

The theorem then states, let be a symmetric positive definite matrix with the Cholesky decomposition where . Additionally, let defined the update of C with . Finally, the update Cholesky factor is given below [3]:

Equation 18

An offspring is produced via the parent x with the following equation:

Equation 19

Using the previously discussed theorem, the Cholesky factor A is adapted with a constant exogenous strategy parameter denoted by ca.

Equation 20

The adaptation show in Equation 20 is executed if the value of (shown in Equation 21) is less than a threshold value, denoted *pt*.

Equation 21

The global step size adaptation in this algorithm possesses similarities with the 1/5th success rule of the (1+1)-ES shown earlier in Equation 3. Therefore, if the new offspring has a better fitness value than that of its parent, λs = 1, otherwise λs = 0. These measures of success are accumulated over generations by using the parameter *cp* as a learning rate as shown in Equation 21 [3]. Now, with the success rate the global step size is updated as follows:

Equation 22

where is the target value for the success rate [3].

The recommended exogenous strategy parameter settings are as follows:

## Active-CMA-ES

The idea behind Active CMA-ES is that the worst offspring should be considered for the adaptation of the covariance matrix by utilizing negative weights. As a result, Active-CMA-ES modifies the covariance matrix update of (µw, λ) - CMA-ES as follows [3]:

Equation 23

Equation 24

The parameter *cc* is also modified as . The parameter *β* has been tuned with empirical investigation, described in [7]. The parameter was chosen as a compromise between improving convergence velocity and ensuring the covariance matrix C is positive definite, in order to make the system more robust [3].

## 2.3.8 (µ, λ)-CMSA-ES

(µ, λ)-CMSA-ES aims to reduce the number of exogenous strategy parameters by reintroducing self-adaptation of the global step size σ. This simplifies the number of exogenous strategy parameters from five in (µw, λ)-CMA-ES down to two [3]. Offspring solutions each have their own individual step sizes which are calculated using the global step size σ as follows [3]:

The *µ* best offspring go through recombination with identical weights and is applied to the vectors and for to obtain the vectors and the new global step size σ. Therefore, the new parent is updated as . The vector is necessary for updating the covariance matrix C, using the learning rate as follows [3]:

Equation 25

The recommended settings of the exogenous strategy parameters are:

## sep-CMA-ES

This CMA-ES variant reduces the space and time complexity of the (µw, λ)-CMA-ES algorithm down to *O*(*n*) as compared to the original *O*(*n3*). The reason for this increase in performance is because simply uses a diagonal matrix D which only contains the square root of the main diagonal elements of the covariance matrix C, along its own diagonal. This means that no computationally expensive eigen decompositions need to be done. The drawback is that this matrix is unable to generate correlated mutations like a covariance matrix. Hence, the covariance matrix update is modified from (µw, λ)-CMA-ES as follows [3]:

Equation 26

The learning rate is increased because of the reduced complexity of the covariance matrix and is newly determined by:

Equation 27

All other parameters from (µw, λ)-CMA-ES remain unchanged.

## (1+1)-Active-CMA-ES

This algorithm is an extension of the (1+1)-Cholesky-CMA-ES that uses the Active-CMA-ES ideology of taking the worst offspring into account for the covariance matrix update. Rather than using the Cholesky decomposition for the covariance matrix , this algorithm uses the Cholesky factor A and its inverse A-1 [3]. Thus, the theorem discussed in 2.3.6 is modified. Let be a positive definite symmetric matrix with Cholesky decomposition . In addition, let be the update equation for C where . Let with and let be the Cholesky decomposition of the updated covariance matrix [3]. Then is updated as follows:

Equation 28

and is update as follows:

Equation 29

.

New offspring are generated the same way, . The success rate from (1+1)-Cholesky-CMA-ES is now modified with a condition as follows:

Equation 30

With this new success rate, a dampening parameter and the target success rate *pt*, the global step size is now:

Equation 31

## Common Random Numbers (CRNs)

CRN is one of the most popular Variance Reduction Techniques (VRT). It is used when comparing similar systems with different configurations in hopes of investigating which system is better under the given circumstances [6]. The main idea is that two different configurations are compared while sharing the same experimental conditions, so that the results observed between the two system configurations are held with more integrity than if the two systems being compared did not share the same experimental conditions. For example, “experimental conditions” for computer simulations may be generated random variates or numbers that are used to feed both systems simultaneously as they carry out their separate operations. In other words, CRN is a VRT that attempts to induce positive correlation by using the same random numbers within simulation of both system configurations [6].

However, there is no guarantee that CRNs will always be a successful VRT. The success of CRN depends heavily on the models under comparison. The reason for this is that different models may generate very different responses to the CRNs. Therefore, it is most effective when conducting CRN simulations with similar models. On the contrary, there are some classes of models where CRN is guaranteed to be successful as a VRT [6]. These classes of models and conditions are discussed in [7].

# Methodology

# References

|  |  |
| --- | --- |
| [1] | H.-G. Beyer and H.-P. Schwefel, "Evolution strategies: a comprehensive introduction," *Natural Computing,* vol. 1, no. 1, pp. 3-52, 2002. |
| [2] | A. Auger and N. Hansen, "Tutorial CMA-ES - Evolutionary Strategies and Covariance Matrix," *Proceeding of the Fifteenth Annual Conference Companion on Genetic and Evolutionary Computation Conference Companion - Gecco 13 Companion,* pp. 1-83, 2013. |
| [3] | T. Bäck, C. Foussette and P. Krause, Contemporary Evolution Strategies, Berlin: Springer Berlin, 2013. |
| [4] | H.-P. Schwefel, "Numberische Optimisrung von Computer-Modellen Mittles der Evolution-strategie," *Journal of Applied Mathematics and Mechanics,* vol. 60, no. 5, pp. 1-272, 1977. |
| [5] | N. Hansen and A. Ostermeier, "Completey Derandomized Self-Adaptation in Evolution Strategies," *Evolutionary Computation,* vol. 9, no. 2, pp. 1-195, 2001. |
| [6] | A. M. Law, Simulation Modeling & Analysis, 4th ed., New York: McGraw-Hill, 2007. |
| [7] | P. Lecuyer, "Efficiency improvement and variance reduction," in *Proceedings of Winter Simulation Conference*, Montreal, 1994. |