Mathematical Geosciences
https://doi.org/10.1007/s11004-024-10148-3

Non-Gaussian Ensemble Optimization

Mathias M. Nilsen[1,2] · **Andreas S. Stordal**[1,2] · **Patrick N. Raanes**[1] · **Rolf J. Lorentzen**[1] · **Kjersti S. Eikrem**[1]

Received: 14 June 2023 / Accepted: 3 June 2024

Abstract
Ensemble-based optimization (EnOpt), commonly used in reservoir management, can be seen as a special case of a natural evolution algorithm. Stein's lemma gives a new interpretation of EnOpt. This interpretation enables us to study EnOpt in the context of general mutation distributions. In this paper, a non-Gaussian generalization of EnOpt (GenOpt) is proposed, where the control gradient is estimated using Stein's lemma, and the mutation distribution is updated separately via natural evolution. For the multivariate case, a Gaussian copula is used to represent dependencies between the marginals. The correlation matrix is also iteratively optimized. It is shown that using beta distributions as marginals in the GenOpt algorithm addresses the truncation problem that sometimes arises when applying EnOpt on bounded optimization problems. The performance of the proposed optimization algorithm is evaluated on several test cases. The experiments indicate that GenOpt is less dependent on the chosen hyperparameters, and it is able to converge more quickly than EnOpt on a reservoir management test case.

Keywords Black-box optimization · Natural evolution · Ensemble optimization · Gaussian copula · Reservoir management

1 Introduction

Ensemble optimization (EnOpt) is a widely studied topic in the reservoir management community since its introduction by Lorentzen et al. (2006) and Chen et al. (2009). EnOpt is a gradient estimation method that uses random draws from a Gaussian distribution to approximate the gradient based on the function values of the sample. The performance of EnOpt depends strongly on the choice of covariance matrix used to sample the control vector. Several methods for iteratively adapting the covariance

✉ Mathias M. Nilsen
mani@norceresearch.no

[1] Energy & Technology, NORCE Norwegian Research Centre, P.O.B 22, Nygårdstangen, 5838 Bergen, Vestland, Norway

[2] Department of Mathematics, University of Bergen, P.O.B 7803, 5020 Bergen, Vestland, Norway

Published online: 28 June 2024

matrix have been developed and tested. Fonseca et al. (2013) employed the covariance matrix adaptation evolution strategy (CMA-ES) developed by Hansen and Ostermeier (2001), where a subset of sampled controls is used to update the covariance matrix at each iteration during the optimization. Another approach was developed by Stordal et al. (2016), where they showed that EnOpt is a special case of the natural evolution strategy (NES) algorithm presented by Wierstra et al. (2008), and that the gradient approximation used in EnOpt converges to the natural gradient (Amari 1998) of the expected objective function. Based on that, a natural gradient expression for the covariance was derived and used to update the matrix.

The EnOpt gradient may be interpreted as the derivative of the expected objective function with respect to the mean of the Gaussian distribution from which the random sample is drawn (Stordal et al. 2016). Additionally, Stein's lemma (Stein 1981) provides a second interpretation, showing that the EnOpt gradient is the expected gradient of the objective function (as opposed to the gradient of the expected objective function). The two interpretations coincide because the gradient of the expected function is taken with respect to a location parameter.

This paper proposes a new method where the gradient estimate from Stein's lemma is used to update the control vector, while the mutation gradient (gradient of NES) is used to update the sampling distribution. Furthermore, by introducing a Gaussian copula, it is shown that it is possible to decouple the update of the marginal distributions and the joint distribution while still imposing a dependence structure between the variables in the control vector. The correlation matrix in the copula is adapted either by the CMA strategy or the mutation gradient. In this manner, a novel simultaneous update for the multivariate distribution and the control vector is presented. It is also shown that using bounded distributions (such as the beta distribution) can be beneficial for optimization problems where the control vector is constrained within upper and lower limits.

Non-Gaussian sampling strategies of the control vector are not a new concept within EnOpt-related methods. Wierstra et al. (2014) suggest that using a Cauchy distribution, which exhibits heavy tails, might be beneficial in the case of multi-modal objectives. Also, Fonseca et al. (2017) points out that it is not necessary that the ensemble be drawn from a Gaussian distribution, while Sarma and Chen (2014) utilized a Sobol sequence to approximate the gradient rather than standard Monte Carlo sampling. Ramaswamy et al. (2020) proposed sampling based on optimal supersaturated designs, and they also compared several different sampling strategies. Lastly, Zhou and Hu (2014) formulated an evolution strategy based on the exponential family of distributions. However, their experimental results were restricted to Gaussian sampling. None of the references above suggest changing how the samples are used to estimate the average gradient, which is the approach taken here.

The outline of this paper is as follows. In Sect. 2, the connection between EnOpt and Stein's lemma is presented without assuming Gaussianity. In Sect. 3 the new framework is applied to the exponential family of mutation distributions, where the update equations of both the control vector and the distribution parameters are derived. In Sect. 4, explicit equations are derived for the Gaussian and the beta distribution. It is also shown how the beta distribution will be used to address the sample truncation

problem for bounded optimization. In Sect. 5 a multivariate extension using Gaussian copulas is introduced. This eventually leads to the GenOpt (Generalized EnOpt) algorithm. In Sect. 6 the new algorithm is applied to several test cases, using beta distributions as marginals in combination with a Gaussian copula. The paper ends with a summary and discussion in Sect. 7.

2 Interpretation of EnOpt Using Stein's Lemma

Consider the objective function $J(x) : \mathbb{R}^d \to \mathbb{R}$ and a Gaussian random variable $X \sim \mathcal{N}(\mu, \Sigma)$. In Stordal et al. (2016) it was shown that the EnOpt gradient, given by

$$\frac{1}{N}\sum_{n=1}^{N} J(X^n)(X^n - \mu), \quad X^n \sim \mathcal{N}(\mu, \Sigma), \tag{1}$$

is a special case of NES (Wierstra et al. 2008; Sun et al. 2009) and that the gradient approximation in Eq. 1 converges almost surely, as $N \to \infty$, to the natural gradient (Amari 1998) of the expected objective function with respect to location parameter μ. That is,

$$\frac{1}{N}\sum_{n=1}^{N} J(X^n)(X^n - \mu) \xrightarrow[N\to\infty]{} \mathrm{Cov}(J(X), X) = \Sigma \nabla_\mu \mathbf{E}[J(X)]. \tag{2}$$

It is possible to reduce the variance of the gradient estimate of EnOpt by subtracting a constant b from $J(X^n)$. In particular, the choice $b = \bar{J} = N^{-1}\sum_{n=1}^{N} J(X^n)$ often reduces the variance of the estimate (Stordal et al. 2016). However, this introduces a bias that can be corrected by replacing N with $N - 1$ in Eq. 1.

Equation 2 shows that the EnOpt gradient can be interpreted as a gradient of an expected, or smoothed, objective function, where the gradient is taken with respect to the mean of the Gaussian distribution. Stein's lemma offers a second interpretation of the EnOpt gradient. It states that if X is a Gaussian random variable with mean μ and covariance matrix Σ and both $\mathbf{E}[J(X)(X - \mu)]$ and $\mathbf{E}[\nabla_x J(X)]$ exist, then

$$\mathrm{Cov}(J(X), X) = \Sigma \mathbf{E}[\nabla_x J(X)]. \tag{3}$$

The comparison of Eqs. 2 and 3 shows that the EnOpt gradient also converges to the expected gradient $\mathbf{E}[\nabla_x J(X)]$, pre-multiplied with Σ. The matrix Σ plays the role of the inverse of the Fisher information matrix for the natural gradient (see e.g. Stordal et al. 2016), whereas, in the EnOpt community, it is viewed as a preconditioning matrix that may be removed by multiplying the left-hand side of Eq. 2 with Σ^{-1} (or a sampling counterpart).

The identity

$$\nabla_\mu \mathbf{E}[J(X)] = \mathbf{E}[\nabla_x J(X)], \tag{4}$$

is due to the Gaussian assumption on X, because μ is a location parameter. For a general random variable $X \in \Omega$ with density $f(x|\theta)$, where θ are parameters of the pdf, Stein's lemma can be expressed as

$$\mathbf{E}[\nabla_x J(X)] = -\mathbf{E}[J(X)\nabla_x \log f(X|\theta)], \tag{5}$$

assuming $\mathbf{E}[|\nabla_x J(x)|] < \infty$ and $f(x|\theta)J(x) \to 0$ as $x \to S$, where S is the boundary of Ω. The validity of Eq. 5 can be shown by utilizing the identity $\nabla f(x|\theta) = f(x|\theta)\nabla \log f(x|\theta)$ and integration by parts

$$\begin{aligned}
\mathbf{E}[\nabla \boldsymbol{J}(\boldsymbol{X})] &= \int_\Omega \nabla J(x) \cdot f(x|\theta) \mathrm{d}\Omega, \\
&= \int_\Omega \nabla\big(\boldsymbol{J}(x) f(x|\theta)\big)\mathrm{d}\Omega \; - \int_\Omega J(x)\nabla f(x|\theta)\mathrm{d}\Omega, \\
&= \int_S \underbrace{J(x) f(x|\theta)}_{=0 \text{ at } S} \cdot \hat{n}\mathrm{d}S \; - \int_\Omega J(x)\nabla \log f(x|\theta) \cdot f(x|\theta)\mathrm{d}\Omega, \\
&= -\mathbf{E}[\boldsymbol{J}(\boldsymbol{X})\nabla \log f(\boldsymbol{X}|\theta)],
\end{aligned} \tag{6}$$

where $\hat{n}$ is a unit normal vector to S. In addition, the gradient of $E[J(X)]$[1] with regard to θ, referred to as the mutation gradient, is given by

$$\nabla_\theta \mathbf{E}[J(X)] = \mathbf{E}[J(X)\nabla_\theta \log f(X|\theta)], \tag{7}$$

and the natural gradient is given by

$$\bar{\nabla}_\theta \mathbf{E}[J(X)] = I(\theta)^{-1}\nabla_\theta \mathbf{E}[J(X)], \tag{8}$$

where $I(\theta) = \mathbf{E}[\nabla_\theta \log f(X|\theta)\nabla_\theta^\top \log f(X|\theta)]$ is the Fisher information matrix. Natural gradients have been shown to improve optimization compared to vanilla gradients (Amari 1998). The reason for this is that the θ-space is not Euclidean, but Riemannian. In that manner, $I(\theta)$ is the metric tensor, making the natural gradient invariant of the parameterization (see Amari and Douglas 1998 for an illustrative example). The equivalence given in Eq. 4 arises from the identity

$$\nabla_x \log f(x|\mu) = -\Sigma^{-1}(x-\mu) = -\nabla_\mu \log f(x|\mu); \tag{9}$$

hence, the expected gradient coincides with the mutation gradient if $\nabla_x \log f(x|\mu) = -\nabla_\mu \log f(x|\mu)$ in general.

To summarize, it is possible to formulate two generalizations of EnOpt for a given density $f(x|\theta)$, one using Stein's lemma and one using mutation. In the Gaussian case, they both coincide with the original EnOpt. However, if $f(x|\theta)$ does not have a location parameter, the control, x, must be connected to the distribution parameters for

[1] The term "mutation gradient" is used to refer to the gradient of $\mathbf{E}[J(X)]$ with respect to the parameters θ of the mutation distribution $f(x|\theta)$.

mutation. A good choice could be the mean or the mode of the distribution. However, this paper suggests a different approach where the two interpretations of EnOpt are combined into a single optimization algorithm. The expected gradient from Stein's lemma is used to update the current control x_k (where k is the iteration index), while the mutation gradient is used to update the parameters of the sampling distribution (θ_k). The next section focuses on the exponential family of distributions, which includes the Gaussian distribution as a special case. The algorithms are presented in univariate form, with a trivial extension to the multivariate case if the variables are independent.

3 Exponential Family of Mutation Distributions

Consider the univariate exponential family of probability densities of the form

$$f(x|\theta) = h(x) \exp\left(\eta(\theta)^\top T(x) - A(\theta)\right), \tag{10}$$

where $\theta \in \mathbb{R}^m$ is a vector of parameters. In general, $T : \mathbb{R} \to \mathbb{R}^m$ and $\eta : \mathbb{R}^m \to \mathbb{R}^m$ are both vector functions, while $A(\theta)$ and $h(x)$ are scalar functions. The density is defined on the interval (a, b), where a and b could be $\pm\infty$. The objective is to minimize the function $J(x)$, using samples from the density in Eq. 10 to approximate the gradient via Stein's lemma. Inserting Eq. 10 into Eq. 5 yields

$$\mathbf{E}[\partial_x J(X)] = -\mathbf{E}\left[J(X)\left(\frac{\partial_x h(X)}{h(X)} + \eta(\theta)^\top \partial_x T(X)\right)\right], \tag{11}$$

where $h(x)\exp\left(\eta(\theta)^\top T(x)\right) J(x) \to 0$ as $x \to a$ or b and $\mathbf{E}[|\partial_x J(x)|] < \infty$ or $h(x)\exp\left(\eta(\theta)^\top T(x)\right) \to 0$ if a and b are finite (Landsman and Nešlehová 2008). A Monte Carlo approximation of the expectation in Eq. 11 yields a generalized version of EnOpt for exponential distributions. The update equation for x is given by

$$x_{k+1} = x_k + \alpha_x \frac{1}{N-1} \sum_{n=1}^{N} \left(J(X^n) - \bar{J}\right)\left(\frac{\partial_x h(X_k^n)}{h(X_k^n)} + \eta(\theta)^\top \partial_x T(X_k^n)\right), \tag{12}$$

where α_x is the step size. Note that J is centered with $\bar{J}$ in Eq. 12, which is possible without changing the expectation in Eq. 11, since it only requires that $f(x|\theta)$ evaluated at $x = a$ and b is the same. However, then N must be replaced with $N - 1$ as before.

It is also possible to minimize $L(\theta) \equiv \mathbf{E}[J(X)] = \int J(x) f(x|\theta) \mathrm{d}x$ with respect to θ using mutation. The mutation gradient is

$$\nabla_\theta L(\theta) = \mathbf{E}\left[\boldsymbol{J}(x)\left(\nabla_\theta \eta(\theta)^\top \boldsymbol{T}(\boldsymbol{X}) - \nabla_\theta \boldsymbol{A}(\theta)\right)\right], \tag{13}$$

with corresponding natural gradient

$$\bar{\nabla}_\theta L(\theta) = I(\theta)^{-1}\mathbf{E}\left[\boldsymbol{J}(x)\left(\nabla_\theta \eta(\theta)^\top \boldsymbol{T}(\boldsymbol{X}) - \nabla_\theta \boldsymbol{A}(\theta)\right)\right], \tag{14}$$

where

$$I(\theta) = \mathbf{E}\left[\left(\nabla_\theta \eta(\theta)^\top \boldsymbol{T}(\boldsymbol{X}) - \nabla_\theta \boldsymbol{A}(\theta)\right)\left(\nabla_\theta \eta(\theta)^\top \boldsymbol{T}(\boldsymbol{X}) - \nabla_\theta \boldsymbol{A}(\theta)\right)^\top\right]. \quad (15)$$

At iteration k, a sample $\{X_k^n\}_{n=1}^N$ is drawn from $f(x|\theta_k)$ and used in a Monte Carlo approximation of Eq. 14 to update θ. The update equation for θ then reads

$$\theta_{k+1} = \theta_k - \alpha_\theta \frac{1}{N-1} I(\theta)^{-1} \sum_{n=1}^{N} \left(\boldsymbol{J}(\boldsymbol{X}_k^n) - \bar{\boldsymbol{J}}\right)\left(\nabla_\theta \eta(\theta_k)^\top \boldsymbol{T}(\boldsymbol{X}_k^n) - \nabla_\theta \boldsymbol{A}(\theta_k)\right). \quad (16)$$

In other words, it is possible to update the control via Eq. 12 and update the mutation distribution via Eq. 16. In the following section, a few examples of different distributions are presented.

4 Example of Distributions

4.1 Gaussian Distribution

As a special case, consider the univariate Gaussian density with known σ. The density can be written on the form of Eq. 10 with

$$\begin{aligned} \eta(\theta) &= \frac{\mu}{\sigma}, \quad h(x) = \left(\sqrt{2\pi\sigma^2}\right)^{-1} \exp(-x^2/(2\sigma^2)), \\ T(x) &= \frac{x}{\sigma}, \quad A(\theta) = \frac{\mu^2}{2\sigma^2}, \end{aligned} \quad (17)$$

so that

$$\partial_x h(x) h(x)^{-1} = -x/\sigma^2, \quad \eta(\theta)\partial_x T(x) = \tfrac{\mu}{\sigma^2}, \quad (18)$$

and Eq. 12 (with centering of J) becomes

$$x_{k+1} = x_k - \alpha \frac{1}{N-1} \sigma^{-2} \sum_{n=1}^{N} \left(J(X_k^n) - \bar{J}\right)\left(X_k^n - x_k\right). \quad (19)$$

In the same manner, Eq. 16 can be used for the parameter μ with $\partial_\theta A(\theta) = \frac{\mu}{\sigma^2}$ and $\partial_\theta \eta(\theta) = \sigma^{-1}$ together with $I(\theta) = \sigma^2$ to get (with centering of J)

$$\mu_{k+1} = \mu_k - \alpha \frac{1}{N-1} \sigma^{-2} \sum_{n=1}^{N} \left(J(X_k^n) - \bar{J}\right)\left(X_k^n - \mu_k\right). \quad (20)$$

Equations 19 and 20 show that the update for x and μ coincide. However, this is not the case for a general distribution, as shown in the next section. Furthermore, σ^2 may

also be updated at each iteration using the natural gradient in order to accelerate the convergence significantly (see e.g. Stordal et al. 2016)

$$\sigma_{k+1}^2 = \sigma_k^2 - \alpha \frac{1}{N-1} \sum_{n=1}^{N} \left(J(X_k^n) - \bar{J} \right) ((X_k^n - \mu_k)^2 - \sigma_k^2). \tag{21}$$

4.2 Beta Distribution

The beta distribution is a well-known distribution defined on a bounded interval with two parameters $\theta = [\alpha,\ \beta]^\top$. The density is given by

$$f(x|\theta) = \frac{\Gamma(\alpha+\beta)}{\Gamma(\alpha)\Gamma(\beta)} x^{\alpha-1}(1-x)^{\beta-1}, \quad 0 \le x \le 1, \tag{22}$$

where Γ is the gamma function. The exponential parameterization reads

$$\begin{aligned}
&\eta(\theta) = [\alpha - 1,\ \beta - 1]^\top, \quad h(x) = 1, \quad T(x) = [\log(x),\ \log(1-x)]^\top, \\
&A(\theta) = \log\Gamma(\alpha) + \log\Gamma(\beta) - \log\Gamma(\alpha+\beta), \\
&I(\theta) = \begin{bmatrix} \partial_\alpha\psi(\alpha) - \partial_\alpha\psi(\alpha+\beta) & -\partial_\beta\psi(\alpha+\beta) \\ -\partial_\beta\psi(\alpha+\beta) & \partial_\beta\psi(\beta) - \partial_\beta\psi(\alpha+\beta) \end{bmatrix},
\end{aligned}$$

where ψ is the digamma function. The update Eqs. 12 and 16 for the beta distribution are then

$$x_{k+1} = x_k + \alpha_x \frac{1}{N-1} \sum_{n=1}^{N} \left(J(X_k^n) - \bar{J} \right) \left(\frac{\alpha_k - 1}{X_k^n} - \frac{\beta_k - 1}{1 - X_k^n} \right), \tag{23}$$

$$\begin{aligned}
\theta_{k+1} = \begin{bmatrix} \alpha_k \\ \beta_k \end{bmatrix} &- \alpha_\theta \frac{1}{N-1} I(\theta)^{-1} \sum_{n=1}^{N} \left(J(X_k^n) - \bar{J} \right) \\
&\left(\begin{bmatrix} \log(X_k^n) \\ \log(1 - X_k^n) \end{bmatrix} - \begin{bmatrix} \partial_\alpha A(\theta_k) \\ \partial_\beta A(\theta_k) \end{bmatrix} \right),
\end{aligned} \tag{24}$$

where

$$\begin{aligned}
\partial_\alpha A(\theta_k) &= \psi(\alpha_k) - \psi(\alpha_k + \beta_k), \\
\partial_\beta A(\theta_k) &= \psi(\beta_k) - \psi(\alpha_k + \beta_k).
\end{aligned}$$

Often it is of interest to compute the gradient in a region around the current control, $[x_k - \varepsilon,\ x_k + \varepsilon]$. It is possible to define a beta distribution on an arbitrary interval. However, this is equivalent to sampling $X_k \sim \text{Beta}(\alpha_k, \beta_k) \in [0,\ 1]$ and then applying the linear transformation $Y_k = x_k + 2\varepsilon(X_k - \frac{1}{2})$. The domain of the new random variable Y_k is then $[x_k - \varepsilon,\ x_k + \varepsilon]$, and the gradient in Eq. 23 is re-scaled by a factor

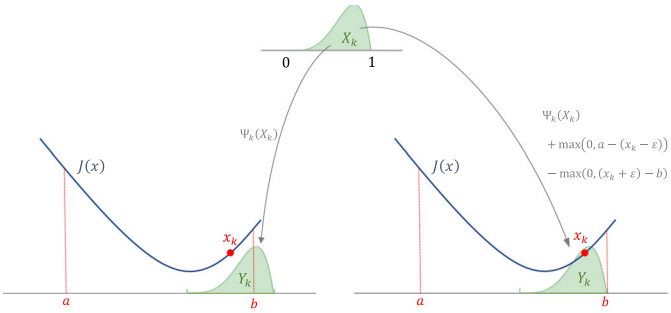

Fig. 1 The differences of the transformation defined in Eq. 26 (to the left) and Eq. 27 (to the right) when the current control x^i (red dot) lies close to the upper border, b. An arbitrary objective function $J(x)$ is illustrated as the dark-blue curve. The lower and upper control bounds are shown on red

(2ε), giving

$$\mathbf{E}[\partial_{y_k} J(Y_k)] = -(2\varepsilon)^{-1}\mathbf{E}\left[\boldsymbol{J}\left(x_k + 2\varepsilon(\boldsymbol{X}_k - \frac{1}{2})\right)\left(\partial_\theta \eta(\theta)^\top \boldsymbol{T}(\boldsymbol{X}_k) - \partial_\theta \boldsymbol{A}(\theta)\right)\right]. \tag{25}$$

4.3 Bounded Optimization with the Beta Distribution

Optimization problems are often subject to constraints in the form of bounds, such that $x \in [a,\ b]$. In the original EnOpt formulation, where the ensemble is drawn from a Gaussian distribution, some of the ensemble members might lie outside of the allowed region. The common way to deal with this problem is to truncate those members at the border, leading to a sampling bias in the gradient estimate. This sampling bias can be large when x is located close to one of the bounds. However, as the beta distribution is bounded on an interval $[0,\ 1]$, this problem can be eliminated. Consider the following transformation of the random variable X

$$\boldsymbol{\Psi}_k(\boldsymbol{X}) = x_k + \left(\boldsymbol{X} - \frac{1}{2}\right) \cdot \min\left(2\varepsilon,\ b - a\right). \tag{26}$$

$\Psi_k(X)$ is the same transformation as in Eq. 25, except that if the sampling region 2ε is larger than the allowed interval $b - a$, the sampling region is set to be $b - a$. If the lower or upper bounds of the transformed ensemble $\Psi_k(X)$ are located outside of a or b, it should be translated to lie within the allowed region. This can be achieved by adding the following two terms to the transformation in Eq. 26

$$\boldsymbol{Y}_k = \boldsymbol{\Psi}_k(\boldsymbol{X}_k) + \max\left(0,\ a - (x_k - \varepsilon)\right) - \max\left(0, (x_k + \varepsilon) - b\right). \tag{27}$$

The differences in the transformations in Eqs. 26 and 27 are illustrated in Fig. 1, where the current control x_k (red dot) lies close to the upper bound.

5 Multivariate Extension and the GenOpt Algorithm

The algorithms from the previous section were defined for univariate distributions. Many optimization problems of interest are multivariate, and EnOpt is defined for a multivariate Gaussian distribution. Multivariate EnOpt is discussed at length in Stordal et al. (2016), where the mean and covariance matrix are updated as

$$\mu_{k+1} = \mu_k - \alpha_x \frac{1}{N-1} \sum_{n=1}^{N} \left(J(X_k^n) - \bar{J}\right)(X_k^n - \mu_k), \tag{28}$$

$$\mathbf{\Sigma}_{k+1} = \mathbf{\Sigma}_k + \alpha_\theta \frac{1}{N-1} \sum_{n=1}^{N} \left(J(X_k^n) - \bar{J}\right)((X_k^n - \mu_k)(X_k^n - \mu_k)^\top - \mathbf{\Sigma}_k). \tag{29}$$

Alternative multivariate distributions exist, such as the multivariate extension of the beta distribution, the d-dimensional Dirichlet distribution, with density

$$f(x|\theta) = \frac{\Gamma\left(\sum_{j=1}^{d} \alpha_j\right)}{\prod_{j=1}^{d} \Gamma(\alpha_j)} \prod_{j=1}^{d} x_j^{\alpha_j - 1}, \quad \sum_{j=1}^{d} x_j = 1. \tag{30}$$

However, the constraint $\sum_{j=1}^{d} x_j = 1$ makes the distribution complicated to use. Also, the covariance between x_i and x_j is given by

$$\mathrm{Cov}(x_i, x_j) = -\frac{\alpha_i \alpha_j}{(\sum_{k=1}^{d} \alpha_k)^2 (\sum_{k=1}^{d} \alpha_k + 1)}, \tag{31}$$

which is necessarily negative since each $\alpha_i > 0$. Hence, the Dirichlet distribution seems unfit as a mutation distribution for optimization. An alternative is the zero-mean multivariate Laplace distribution of dimension d with density given by

$$f(x|\theta) = \frac{2}{(2\pi)^{d/2} |\mathbf{\Sigma}|^{1/2}} \left(\frac{x^\top \mathbf{\Sigma}^{-1} x}{2}\right) K_d(\sqrt{2x^\top \mathbf{\Sigma}^{-1} x}), \tag{32}$$

where K_d is the modified Bessel function of the second kind, which makes differentiation with regard to x or $\mathbf{\Sigma}$ intractable.

Here, a more general approach is taken where one uses univariate distributions for each component of the control vector and a Gaussian copula with correlation matrix $\mathbf{R}$ to define the dependence structure implicitly. The next subsection will cover the basics of Gaussian copulas and how to sample from them.

5.1 The Gaussian Copula

Given a random vector, $X = (X_1, \ldots, X_d)$, with continuous marginal cdf's $F_i(x_i|\theta_i)$, the copula of X is defined as the joint distribution of the uniform variables

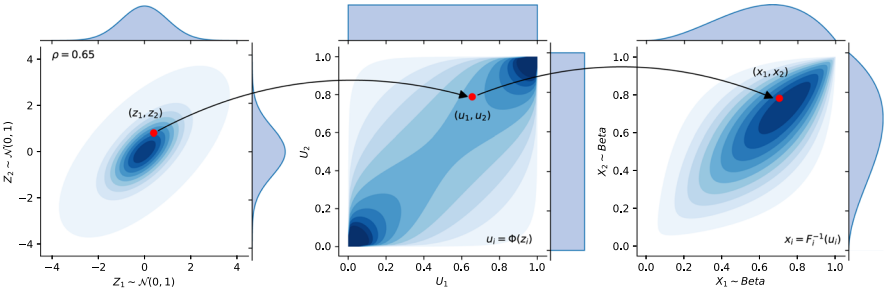

Fig. 2 The sampling process of one single draw, x_i (red dot), of the multivariate distribution (blue contour map) in Eq. 34 with two beta distributions as marginals. Left panel: Shows the sample drawn from $\mathcal{N}(0, \mathbf{R})$. Middle panel: Shows the distribution with uniform marginals after applying the standard normal cdf on each component of z, $(u_1,\ u_2) = (\Phi(z_1)\,,\ \Phi(z_2))$. Right panel: Shows the final multivariate distribution after applying the inverse cdf of the beta distributions, $x_i = F_i^{-1}(u_i\,, \theta_i)$. This illustration is strongly motivated by an example shown by Hazarika et al. (2019)

$(U_1, \ldots, U_d) = (\boldsymbol{F}_1(\boldsymbol{X}_1|\theta_1), \ldots, \boldsymbol{F}_d(x_d|\theta_d))$. The copula contains all the information of the dependence between the components in X. The Gaussian copula is defined over the hypercube $[0, 1]^d$ as

$$C_{\mathbf{R}}(u) = \Phi_{\mathbf{R}}\left(\Phi^{-1}(u_1), \ldots, \Phi^{-1}(u_d)\right),$$

where Φ is the cdf of the standard normal distribution and $\Phi_{\mathbf{R}}$ is the joint cdf of a zero mean d-variate Gaussian distribution with covariance matrix equal to the correlation matrix $\mathbf{R}$. The pdf of the Gaussian copula is given by

$$c_{\mathbf{R}}(u) = \frac{1}{\sqrt{|\mathbf{R}|}} \exp\left(-\frac{1}{2}\begin{pmatrix}\Phi^{-1}(u_1)\\ \vdots \\ \Phi^{-1}(u_d)\end{pmatrix}^T (\mathbf{R}^{-1} - \mathbf{I}) \begin{pmatrix}\Phi^{-1}(u_1)\\ \vdots \\ \Phi^{-1}(u_d)\end{pmatrix}\right), \tag{33}$$

where $|\mathbf{R}|$ is the determinant of $\mathbf{R}$, and $u_i = F_i(x_i|\theta_i)$. The multivariate density of X is given by

$$f(x|\theta) = c_{\mathbf{R}}(u) \prod_{i=1}^{d} f_i(x_i|\theta_i), \tag{34}$$

where $f_i(x_i|\theta_i)$ is the marginal pdf of component x_i.

The density in Eq. 34 cannot be evaluated analytically. However, it is possible to sample from it. To draw one realization X from the distribution in Eq. 34, first draw one realization from a multivariate normal distribution, $Z \sim \mathcal{N}(0, \mathbf{R})$. Then transform each component Z_i (where $i = 1, \ldots, d$) of Z by applying the cdf of a standard normal distribution to obtain $U_i = \Phi(Z_i)$. Finally, for each U_i apply the inverse cdf of the marginal distribution, $X_i = F^{-1}(U_i|\theta_i)$. Figure 2 visualizes this process with a two-

dimensional Gaussian copula with correlation $\rho = 0.65$, and beta(3, 2) as marginal distributions.

5.2 Stein's Lemma and the Copula

To obtain the expected gradient in the multivariate case, Stein's lemma, Eq. 5, is applied to the density in Eq. 34. Consider component i of the expected gradient

$$\mathbf{E}[\partial_{x_i} J(X)] = -\mathbf{E}[J(X)\partial_{x_i} \log f(X|\theta)], \tag{35}$$

where

$$\log f(x|\theta) = \log c_{\mathbf{R}}(u) + \sum_{i=1}^{d} \log f_i(x_i|\theta_i). \tag{36}$$

Hence,

$$\partial_{x_i} \log f(x|\theta) = \partial_{x_i} \log c_{\mathbf{R}}(u) + \partial_{x_i} \log f_i(x_i|\theta_i). \tag{37}$$

The term $\partial_{x_i} \log f_i(x_i|\theta_i)$ was studied in great detail in the previous sections. However, there is also an extra term contributing to the gradient arising from the copula: $\partial_{x_i} \log c_{\mathbf{R}}(u)$. Recall that $u = [u_1(x_1), \cdots, u_d(x_d)]^\top$. Now, defining $z \equiv \left[\Phi^{-1}(u_1), \cdots, \Phi^{-1}(u_d)\right]^\top$ and $\mathbf{H} \equiv \left(\mathbf{R}^{-1} - \mathbf{I}\right)$,

$$\partial_{x_i} \log c_{\mathbf{R}}(u) = \partial_{x_i} \left(-\frac{1}{2} z^T \mathbf{H} z\right) = -\sum_j H_{ij} z_j \cdot \partial_{u_i} \Phi^{-1}(u_i) \cdot \partial_{x_i} F_i(x_i|\theta_i). \tag{38}$$

By definition $\partial_{x_i} F_i(x_i|\theta_i) = f_i(x_i|\theta_i)$, and the inverse function theorem gives

$$\partial_{u_i} \Phi^{-1}(u_i) = \frac{1}{\phi(z_i)} = \frac{1}{\phi\left(\Phi^{-1}(F_i(x_i|\theta_i))\right)}, \tag{39}$$

where ϕ is the pdf of a standard normal distribution so that

$$\partial_{x_i} \log c_{\mathbf{R}}(u) = -\frac{f_i(x_i|\theta_i)}{\phi(z_i)} \sum_{j=1}^{d} H_{ij} z_j = -\frac{f_i(x_i|\theta_i)}{\phi(z_i)} \sum_{j=1}^{d} \left(\mathbf{R}^{-1} - \mathbf{I}\right)_{ij} z_j, \tag{40}$$

where z_j is the jth component of the random variable $Z \sim \mathcal{N}(0, \mathbf{R})$. In general, the ith component of the gradient gets contributions from all the other components due to the correlation matrix $\mathbf{R}$. In the case of no correlation, $\mathbf{R} = \mathbf{I}$, the term in Eq. 40 will vanish, and each gradient component will only have a contribution from the respective marginal distribution. The Monte Carlo estimate of the expectation in Eq. 35 is then

given by

$$\mathbf{E}[\partial_{x_i} J(X)] \approx \frac{-1}{N-1} \sum_{n=1}^{N} \left(J(X^n) - \bar{J}\right) \left(\partial_{x_i} \log f_i(X_i^n|\theta_i) - \frac{f_i(X_i^n|\theta_i)}{\phi(Z_i^n)} \sum_{j=1}^{d} \left(\mathbf{R}^{-1} - \mathbf{I}\right)_{ij} Z_j^n \right). \tag{41}$$

5.3 The Mutation Gradient and the Natural Gradient

The expected gradient of the multivariate distribution in Eq. 34 was derived via Stein's lemma. In this subsection, the mutation gradient is derived and discussed. The parameters of the distribution in Eq. 34 are

$$\theta^T = \left[\theta_1^T, \cdots, \theta_d^T\right], \tag{42}$$

and $\mathbf{R}$, where each $\theta_i \in \mathbb{R}^{m_i}$. The mutation gradient of θ can be split into d individual gradients

$$\nabla_\theta \mathbf{E}[J(X)] = \begin{pmatrix} \nabla_{\theta_1} \mathbf{E}[J(X)] \\ \vdots \\ \nabla_{\theta_d} \mathbf{E}[J(X)] \end{pmatrix}, \tag{43}$$

where each $\nabla_{\theta_i} \mathbf{E}[J(X)] = \mathbf{E}[J(X) \nabla_{\theta_i} \log f(x|\theta)] \in \mathbb{R}^{m_i}$ will only have a contribution from the ith marginal since

$$\nabla_{\theta_i} \left(\log f(x|\theta)\right) = \nabla_{\theta_i} \left(\log c_R + \sum_j \log f_j(x_j|\theta_j) \right) = \nabla_{\theta_i} \log f_i(x_i|\theta_i). \tag{44}$$

Recall that the natural gradient is used to optimize the mutation distribution, Eq. 8. Therefore the Fisher information matrix is needed. However, it can be shown that multiplication of the inverse Fisher matrix with the mutation gradient in Eq. 43 can be done marginally. More specifically, the Fisher matrix is block-diagonal

$$I(\theta) = \begin{pmatrix} \boldsymbol{I_1(\theta_1)} & & \\ & \ddots & \\ & & \boldsymbol{I_d(\theta_d)} \end{pmatrix}, \tag{45}$$

such that each subgradient $\bar{\nabla}_{\theta_i}$ of the Natural gradient $\bar{\nabla}_\theta$ is given by

$$\bar{\nabla}_{\theta_i} = I_i^{-1}(\theta_i) \nabla_{\theta_i}. \tag{46}$$

To show this, consider the element $I_{kl} = \mathbf{E}\left[\partial_{\theta_k} \log f(X|\theta)\, \partial_{\theta_l} \log f(X|\theta)\right]$ of the Fisher information matrix, where θ_k and θ_l are components of the parameter vector to different marginal distributions

$$\begin{aligned} I_{kl} &= \mathbf{E}\left[\partial_{\theta_k} \log f(X|\theta)\, \partial_{\theta_l} \log f(X|\theta)\right] \\ &= \int \partial_{\theta_k} \log f_k(x_k|\theta_k)\, \partial_{\theta_l} \log f_l(x_l|\theta_l)\, c_{\mathbf{R}}(u(x)) \prod_{j=1}^{d} f_j(x_j|\theta_j)\, \mathrm{d}\Omega \\ &= \int \left(\partial_{\theta_k} \log f_k(x_k|\theta_k) f_k(x_k|\theta_k)\right) \\ &\quad \left(\partial_{\theta_l} \log f_l(x_l|\theta_l) f_l(x_l|\theta_l)\right)\, c_{\mathbf{R}}(u(x)) \prod_{j\neq k,l} f_j(x_j|\theta_j)\, \mathrm{d}\Omega. \end{aligned} \tag{47}$$

Using the definition $f(x)\partial_x \log f(x) = \partial_x f(x)$, gives

$$\begin{aligned} I_{kl} &= \int \left(\partial_{\theta_k} f_k(x_k|\theta_k)\right)\left(\partial_{\theta_l} f_l(x_l|\theta_l)\right)\, c_{\mathbf{R}}(u(x)) \prod_{j\neq k,l} f_j(x_j|\theta_j)\, \mathrm{d}\Omega \\ &= \partial_{\theta_k}\partial_{\theta_l} \int f_k(x_k|\theta_k) f_l(x_l|\theta_l)\, c_{\mathbf{R}}(u(x)) \prod_{j\neq k,l} f_j(x_j|\theta_j)\, \mathrm{d}\Omega \\ &= \partial_{\theta_k}\partial_{\theta_l} \int c_{\mathbf{R}}(u(x)) \prod_{j=1}^{d} f_j(x_j|\theta_j)\, \mathrm{d}\Omega = \partial_{\theta_k}\partial_{\theta_l} \underbrace{\int f(x|\theta)\, \mathrm{d}\Omega}_{=1} = 0. \end{aligned} \tag{48}$$

This means that all elements of the off-diagonal blocks of the Fisher matrix are zero, and each marginal mutation gradient, ∇_{θ_i}, can be multiplied by its respective inverse Fisher matrix $I_i^{-1}(\theta_i)$. Finally, the Monte-Carlo estimate of the natural gradient of each marginal mutation gradient can be approximated by

$$\bar{\nabla}_{\theta_i} \mathbf{E}[J(X)] \approx \frac{1}{N-1} I^{-1}(\theta_i) \sum_{n=1}^{N} \left(J(X^n) - \bar{J}\right)\left(\nabla_{\theta_i} \log f_i(X_i^n|\theta_i)\right). \tag{49}$$

In summary, each component of the control vector x and each marginal density $f_i(x_i|\theta_i)$ can be updated individually while their dependency (correlation) is represented by $\mathbf{R}$ in the Gaussian copula. The final piece missing before the full Generalized EnOpt (GenOpt) can be presented, is an adaption scheme for the correlation matrix. The following subsections focus on two different approaches. The first is the mutation approach, and the second is the covariance matrix adaption (CMA) approach developed by Hansen and Ostermeier (2001).

5.3.1 The Mutation Approach for the Correlation Matrix

The mutation approach for the adaption of the correlation matrix is very similar to the mutation approach for the covariance matrix in standard EnOpt (Stordal et al.

2016), shown in Eq. 29. As the density $c_{\mathbf{R}}(u(x))$ in Eq. 33 has a Gaussian structure, the natural gradient $\bar{\nabla}_{\mathbf{R}}\mathbf{E}[J(X)]$ is the same as $\bar{\nabla}_{\mathbf{\Sigma}}\mathbf{E}[J(X)]$ in Stordal et al. (2016) with $X^n - \mu \to Z^n$ and $\Sigma \to \mathbf{R}$. The only difference is that the diagonal elements of $\bar{\nabla}_{\mathbf{R}}\mathbf{E}[J(X)]$ should be set to 0, such that the diagonal elements of the correlation matrix is always 1. The mutation gradient of the correlation matrix is therefore approximated by

$$\bar{\nabla}_{\mathbf{R}} L(\theta) = \bar{\nabla}_{\mathbf{R}}\mathbf{E}[J(X)] \approx \frac{1}{N-1}\sum_{n=1}^{N}\left(J(X^n) - \bar{J}\right)\left(Z^n Z^{nT} - \mathbf{R}\right), \tag{50}$$

where the diagonal elements of the gradient are set to 0. This can then be used to update the correlation iteratively,

$$\mathbf{R}_{k+1} = \mathbf{R}_k - \alpha_{\mathbf{R}}\bar{\nabla}_{\mathbf{R}} L(\theta_k), \tag{51}$$

where $\alpha_{\mathbf{R}}$ is a learning rate. A check to ensure that $\mathbf{R}_{k+1}$ satisfies the conditions of a correlation matrix is employed after the update.

5.3.2 The CMA Approach for the Correlation Matrix

Fonseca et al. (2013) introduced the covariance matrix adaption evolution strategy (CMA-ES) developed by Hansen and Ostermeier (2001) to EnOpt as a way of iteratively adapting the covariance used to perturb the control vector. The goal of this subsection is to briefly describe the CMA update and how it is applied to $\mathbf{R}$.

The CMA update consists of a rank-λ and a rank-1 update, given by

$$\begin{aligned}\mathbf{\Sigma}_{k+1} = (1-\alpha_1-\alpha_\lambda)\mathbf{\Sigma}_k + \underbrace{\alpha_1 \boldsymbol{p}_{k+1}\boldsymbol{p}_{k+1}^{\top}}_{\text{rank-1 update}} \\ + \underbrace{\alpha_\lambda \sum_{n=1}^{\lambda} w_n (\boldsymbol{x}_k^n - \boldsymbol{x}_k)(\boldsymbol{x}_k^n - \boldsymbol{x}_k)^{\top}}_{\text{rank-}\lambda\text{update}},\end{aligned} \tag{52}$$

where α_1 and α_λ are learning rates. The rank-λ update selects the λ best ensemble members and uses them to estimate the covariance. The ensemble members are sorted such that $J(x_1) < J(x_2) < \cdots < J(x_\lambda)$. Each member n is given a weight w_n, such that $\sum_{n=1}^{\lambda} w_n = 1$. In Fonseca et al. (2013) they were chosen as $w_n = 1/\lambda$, meaning all λ samples were given equal weighting, while Hansen (2006) suggests

$$w_n = \frac{\log(\lambda+1) - \log n}{\sum_i^{\lambda}\log(\lambda+1) - \log\lambda}. \tag{53}$$

The quantity $\boldsymbol{p}^{k+1} \in \mathbb{R}^d$ in the rank-1 update is the so-called evolution path. It is the cumulation of earlier steps, and it is given by

$$\boldsymbol{p}_{k+1} = (1-\alpha_c)\boldsymbol{p}_k + \sqrt{\alpha_c(2-\alpha_c)\mu_{\text{eff}}}\frac{\boldsymbol{x}_{k+1}-\boldsymbol{x}_k}{\alpha_x}, \tag{54}$$

where α_c is yet another learning rate, and the parameter $\lambda_{\text{eff}} = (\sum_{n=1}^{\lambda} w_i^2)^{-1}$. For further details, the reader is referred to Hansen (2006).

Since CMA updates a covariance matrix, some changes must be made for the algorithm to be used on the correlation matrix in GenOpt. In the rank-λ update, the sample correlation is used instead of sample covariance,

$$\boldsymbol{\Sigma}_k^{\lambda} \equiv \sum_{n=1}^{\lambda} w_n (z_k^n - \bar{z}_k)(z_k^n - \bar{z}_k)^{\top}, \tag{55}$$

$$\mathbf{R}_k^{\lambda} \equiv \operatorname{diag}\left(\boldsymbol{\Sigma}_k^{\lambda}\right)^{-\frac{1}{2}} \boldsymbol{\Sigma}_k^{\lambda} \operatorname{diag}\left(\boldsymbol{\Sigma}_k^{\lambda}\right)^{-\frac{1}{2}}. \tag{56}$$

One should use the Gaussian ensemble Z from the copula sampling to estimate the sample correlation. A similar transformation is applied to the rank-1 term,

$$\boldsymbol{\Sigma}_{k+1}^{\text{one}} \equiv \boldsymbol{p}_{k+1}(\boldsymbol{p}_{k+1})^{\top}, \tag{57}$$

$$\mathbf{R}_{k+1}^{\text{one}} \equiv \operatorname{diag}\left(\boldsymbol{\Sigma}_{k+1}^{\text{one}}\right)^{-\frac{1}{2}} \boldsymbol{\Sigma}_{k+1}^{\text{one}} \operatorname{diag}\left(\boldsymbol{\Sigma}_{k+1}^{\text{one}}\right)^{-\frac{1}{2}}. \tag{58}$$

The CMA-GenOpt update is then given by

$$\mathbf{R}_{k+1} = (1-\alpha_1-\alpha_\lambda)\mathbf{R}^k + \alpha_1 \mathbf{R}_{k+1}^{\text{one}} + \alpha_\lambda \mathbf{R}_{k+1}^{\lambda}. \tag{59}$$

5.4 The GenOpt Algorithm

Everything that has been developed so far in the multivariate case can be merged in a single algorithm denoted Generalized EnOpt (GenOpt). In summary, GenOpt uses Stein's lemma to update the control vector x using Eq. 41, the natural gradient in Eq. 49 to update the parameters θ, and CMA or mutation to adapt the correlation $\mathbf{R}$. However, there is one final detail that has to be addressed. Recall that the EnOpt gradient in Eq. 2 is pre-multiplied with the covariance matrix Σ. This is not the case for the GenOpt gradient. In most cases, where EnOpt is typically applied, the preconditioning of the covariance is desired to obtain a smooth solution of the controls (Chen et al. 2009). Therefore, the GenOpt gradient is pre-multiplied with the covariance matrix in the following algorithm and examples. The covariance matrix can easily be constructed from the correlation matrix and the individual variances of the marginal distributions.

The GenOpt algorithm is presented in Algorithm 1, where the function name "CorrAdapt" indicates the process of adapting the correlation $\mathbf{R}$ either by CMA or mutation. In principle, GenOpt can be applied with any marginal distribution (assuming that analytical expressions exist). However, from this point on, beta distributions will be

used as marginals. The transformation in Eq. 27 is applied to the ensemble X in the construction of the $\boldsymbol{J}$ vector in Algorithm 1.

Algorithm 1 GenOpt

Require: Initial control vector, x, and objective function J ▷ $x \in \mathbb{R}^d$
Require: Initial marginals distribution and parameters $\{F_i, \theta_i\}_{i=1}^N$ ▷ $\theta_i \in \mathbb{R}^{m_i}$
Require: Initial correlation matrix $\mathbf{R} \in \mathbb{R}^{d \times d}$
while not Converged **do**
1. Sample Gaussian copula and multivariate distribution
$\{z^n\}_{n=1}^N \sim \mathcal{N}(0, \mathbf{R})$ ▷ $z^n \in \mathbb{R}^d$
$\{u^n\}_{n=1}^N$, where each component $u_i^n = \Phi(z_i^n)$ ▷ $u^n \in \mathbb{R}^d$
$\{x^n\}_{n=1}^N$, where each component $x_i^n = F_i^{-1}(u_i^n|\theta_i)$ ▷ $x^n \in \mathbb{R}^d$

2. Estimate gradients
for $i = 1$ to d **do**

$$g_i = -\frac{1}{N-1}\sum_{n=1}^{N}\left(J(x^n)-\overline{J}\right)\left(\partial_{x_i}\log f_i(x_i^n|\theta_i) - \frac{f_i(x_i^n|\theta_i)}{\phi(z_i^n)}\sum_{j=1}^{d}\left(\mathbf{R}^{-1}-\mathbf{I}\right)_{ij} z_j^n\right)$$

$$p_i = \frac{1}{N-1}\sum_{n=1}^{N}\left(J(x^n)-\overline{J}\right)\left(I^{-1}(\theta_i)\nabla_{\theta_i}\log f_i(x_i^n|\theta_i)\right)$$

end for
$\Sigma_{ij} = \frac{\mathbf{R}_{ij}}{\sigma_i\sigma_j}$ for $i, j = 1, \ldots, d$ ▷ Construct covariance matrix

3. Update
$x \leftarrow x - \alpha_x \Sigma g$ ▷ $g = [g_1, \cdots, g_d]^T \in \mathbb{R}^d$
$\theta_i \leftarrow \theta_i - \alpha_\theta p_i$ for $i = 1, \ldots, d$ ▷ Update each marginal distribution, $p_i \in \mathbb{R}^{m_i}$
$\mathbf{R} \leftarrow$ CorrAdapt() ▷ Update correlation matrix
end while

6 Numerical Test Cases

In this section, three applications of GenOpt are presented. First, a proof of concept illustrating how non-Gaussian distributions are modified using GenOpt is shown. Next, GenOpt and EnOpt are compared on two test cases. The first is the well-known Rosenbrock function, where many optimization runs are included for comparison since the objective function is cheap to evaluate. Then, a synthetic reservoir management (Jansen et al. 2014) test case is presented. In both the following examples, beta distributions are used as marginals for the Gaussian copula. This is of particular interest for the reservoir case where the controls are bounded such that the transformation proposed in Eq. 27 is tested. In addition, the initial correlation is set to be the identity matrix $\mathbf{R} = \mathbf{I}$, and the CMA scheme is used to update $\mathbf{R}$.

6.1 Visualization

Adaptive mutation of a distribution improves the relevance of samples used to estimate the expected gradient. This is illustrated in Fig. 3, where five mutation steps are shown.

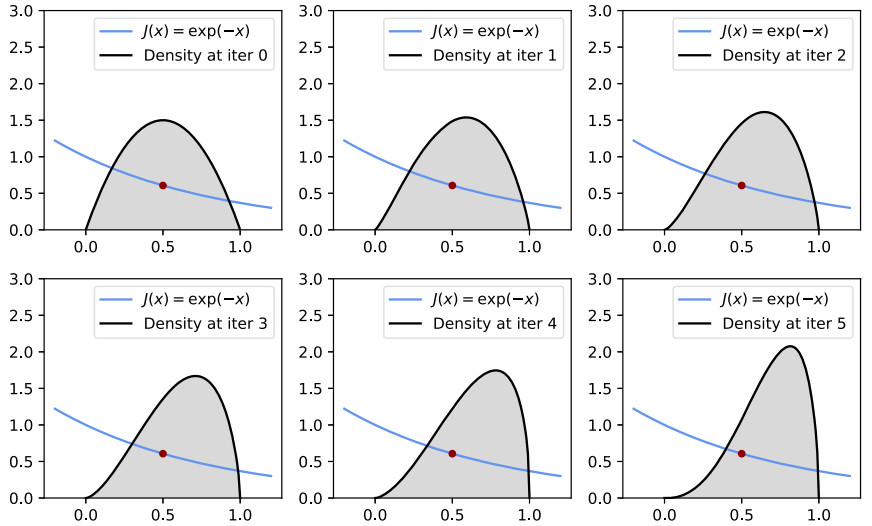

Fig. 3 A one-dimensional example of how five mutation steps changes the beta distribution (black curve with shaded gray area). The blue curve is the objective function $J(x) = \exp(-x)$, and the red dot is the current control (fixed in this example). The initial distribution with $\alpha = \beta = 2.0$ is shown in the upper left panel, while the final distribution is shown in the lower right panel, where now $\alpha = 3.8$ and $\beta = 1.5$

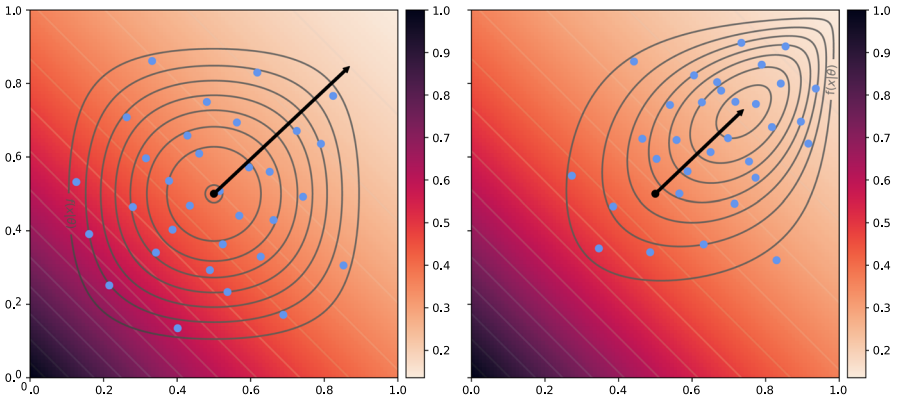

Fig. 4 The effect of 12 mutation steps on the mutation distribution. The mutation distribution comprises a Gaussian copula and two beta distributions as marginals. CMA is used on each step to adapt the correlation matrix in the copula. The heat map represents the objective function $J(x) = \exp[-(x+y)]$. The contour lines represent the mutation distribution $f(x|\theta)$, and the blue dots show an ensemble drawn from the distribution. The black dot is the current control x (fixed during mutation), and the arrow shows the negative GenOpt gradient calculated from the ensemble. The left panel is the initial state, and the right panel is the final state

After five iterations, the probability mass is shifted to the right of x, a region of smaller objective function values.

Figure 4 shows an example that includes adaptation of the correlation matrix in a copula, where 12 mutation steps are presented for the objective function $J(x, y) =$

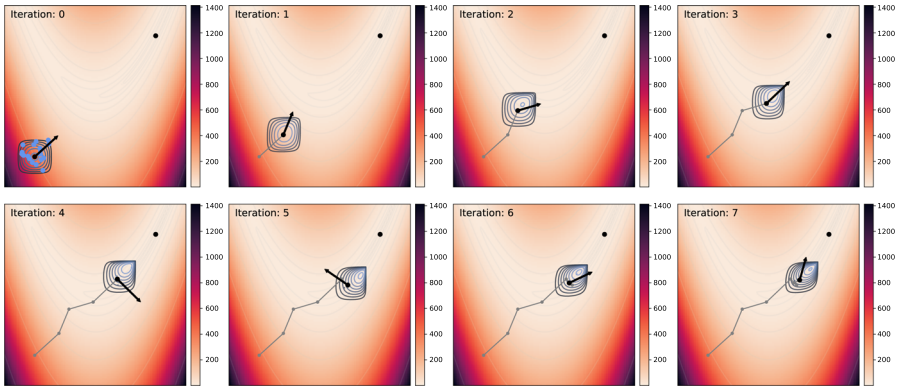

Fig. 5 Seven iterations of GenOpt on the Rosenbrock function (shown as a heat-map). The contours in each panel indicates the mutation distribution at the respective iteration

$\exp(-(x+y))$. Again, beta distributions are used as marginals with initial parameters $\alpha = \beta = 2.0$, and the initial correlation between x and y is set to zero. The bi-variate distribution $f(x|\theta)$ is represented by the black contour lines. An ensemble is drawn (shown as blue dots) and used to estimate a gradient (shown as an arrow). The left plot shows the initial distribution, and the right plot shows the updated distribution. The CMA strategy is utilized in this example as well. As for the one-dimensional case, the probability mass moves to regions of lower objective function values.

6.2 The Rosenbrock Function

The examples in the previous section only illustrated the adaptive mutation. Here, a demonstration of optimization on both x and $f(x|\theta)$ is shown. The objective function under consideration is the Rosenbrock function given by

$$J(x) = \sum_{i=1}^{d-1} 100(x_{i+1} - x_i^2)^2 + (1 - x_i^2)^2, \tag{60}$$

defined on $\mathbb{R}^d$. The global minimum of the Rosenbrock function is located at $x = (1, \ldots, 1)$, inside a long and almost flat parabolic valley. For $d \geq 4$, there is also a local minimum (Shang and Qiu 2006). Figure 5 shows seven optimization steps using GenOpt with $d = 2$. Each panel of Fig. 5 shows one iteration. The starting point of the optimization is set to $(-1, -1)$. The Rosenbrock function is plotted as a heat map, and the minimum of the function is shown as a black dot. Similar to the previous section, the mutation distribution is represented by contour lines, and the gradient at the current control is shown as an arrow (scaled by $(2|\nabla J(x)|)^{-1}$). The optimization path is sketched with a gray line. Figure 5 shows that after seven iterations, the current control is within the valley, and the mutation distribution has been adapted such that the mode is shifted towards smaller $J(x)$ values. It is also worth noticing that the CMA algorithm has introduced a correlation between components of x at this point. In this

example, a backtracking scheme is implemented where the step size is cut in half if the objective function is not decreased.

Next, 100 starting points are randomly selected over the domain $[-2, 2]^d$ for $d = 10,\ 20,\ 50,\ 100$. Both GenOpt and EnOpt run until backtracking is unable to find an improvement (with a tolerance of zero), with 10 allowed step size cuts at each iteration. The maximum number of iterations is set to 3,000. The random seed is fixed for each pair of runs, and the step-size α_x is set to 1 for both methods. The mutation step-size is varied for the different runs with $\alpha_\theta = 1.0,\ 0.5,\ 0.1,\ 0.01$ and 0.001. To make the comparison as fair as possible, the mutation in Eq. 51 is used for the correlation adaption. The initial α and β were fixed at 20 for GenOpt (chosen by some trial and error). The initial variance is also varied with values $\sigma^2 = 1.0, 0.1, 0.01$ and 0.001. For GenOpt, this means that ε is selected from the equality

$$\sigma^2 = 4\epsilon^2 \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}, \tag{61}$$

where the term involving α and β comes from the variance of the beta distribution itself.

For all runs, the ensemble size is fixed at $N_\mathrm{e} = 10$, and the gradients are normalized with the L_2 norm. The results are summarized in Fig. 6 where the average for the objective function and number of iterations (over 100 runs) are reported. The solid lines show the result for GenOpt, while the EnOpt results are dashed.

The experiment indicates that the GenOpt algorithm is less dependent on the choice of α_θ and initial variances. The two methods seem to be fairly equivalent for large initial variance or small mutation step sizes. However, for smaller variances and larger mutation step sizes, GenOpt is able to reach lower values of the objective function, while EnOpt seems to be converging prematurely. This explains why the number of iterations is on average higher for GenOpt in these cases. For smaller α_θ and initial variance, GenOpt reaches a slightly lower objective value with fewer iterations than EnOpt on average. The code used for these experiments is available at GitHub.

6.3 The Egg Model

The last example is a reservoir management test case that employs the Egg model introduced by Jansen et al. (2014). The Egg model is a synthetic reservoir consisting of $60 \times 60 \times 7 = 25,200$ grid cells with high-permeability channels embedded within a low-permeability background. The reservoir contains 8 injection wells and 4 production wells, which are distributed throughout the domain. The permeability field and wells are shown in Fig. 7. Optimization on the rates for the injectors and the bottom hole pressures (BHP) for the producers is performed over a discretized time period of 3,600 days, where the controls can change every 360 days, giving a total of 10 control steps. As a result, the control vector has dimension $(8+4) \cdot 10 = 120$. The objective is to maximize the Net Present Value (NPV) defined as

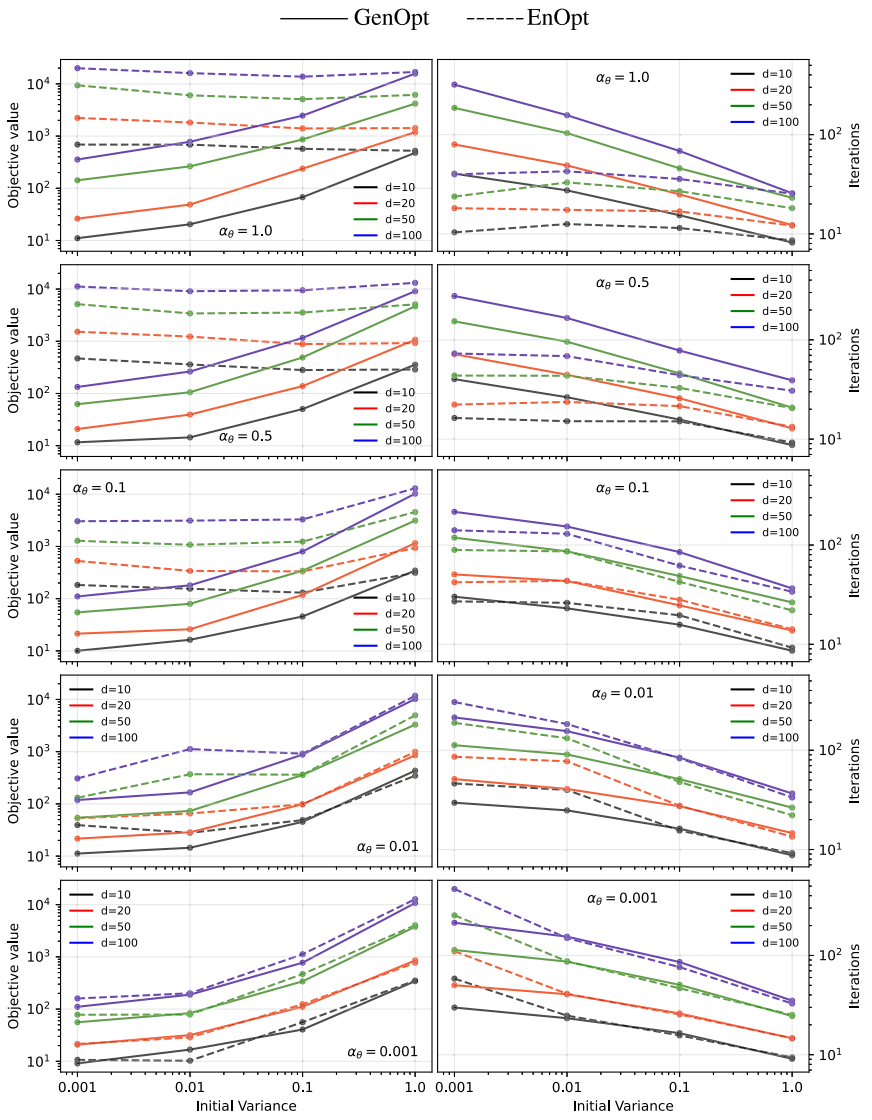

Fig. 6 The average objective value obtained (left panels) and the average iteration count (right panels) until termination of 100 optimization runs. Each panel displays the outcomes corresponding to a specific mutation step-size α_θ, and the x-axis represents four distinct initial variances. Solid lines show the results achieved by GenOpt, whereas dashed lines correspond to EnOpt. The colors of each line signify the dimensionality of the Rosenbrock function. The ensemble size was set to 10 throughout all experiments

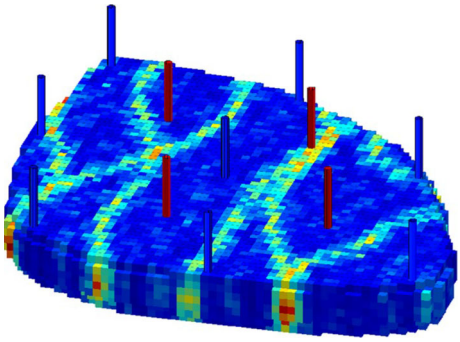

Fig. 7 Shows the permeability field of the Egg model and the position of the injectors (blue) and producers (red). Figure taken from Jansen et al. (2014)

Table 1 Economic data

	Values	Unit
Initial BHP	300	bar
Initial injection rate	80	Sm^3/day
Price of oil, ω_o	300	USD/Sm^3
Cost of water disposal, ω_{wp}	40	USD/Sm^3
Cost of water injection, ω_{wi}	10	USD/Sm^3

Table 2 GenOpt parameters

	Values
Initial α_0	20.0
Initial β_0	20.0
Step-size, α_x	$0.05/\|\nabla J\|_\infty$
Step-size, α_θ	0.01, 0.1, 0.5
Sampling region, ε	0.01, 0.05, 0.1, 0.5

$$J^{\mathrm{NPV}}(x) = \sum_{i=1}^{10} \frac{\omega_o Q_{op}^i - \omega_{wp} Q_{wp}^i - \omega_{wi} Q_{wi}^i}{(1+\tau)^{\Delta t_i}}, \tag{62}$$

where ω_o denotes the price of oil, ω_{wp} the cost of water disposal, and ω_{wi} the cost of water injection. The quantities Q_{op}^i and Q_{wp}^i represent the amounts of oil and water produced, during the ith time window, while Q_{wi}^i denotes the amount of water injected during the same time interval. Here τ represents a discount rate. The values for ω_o, ω_{wp} and ω_{wi}, as well as the initial controls are shown in Table 1. The injection rates and BHP have upper and lower bounds chosen as [150 Sm^3/day, 380 Sm^3/day] and [0 bar, 150 bar], respectively, and the discount rate is set to $\tau = 0.1$. For practicality, the domain is transformed to [0, 1] for the optimization.

The selected parameters for GenOpt are presented in Table 2, and every possible combination in the table is tested three times (with a different random seed each time). For every random seed, EnOpt is also tested with four different initial variances, computed using the relation given in Eq. 61. The initial parameters α_0 and β_0 are

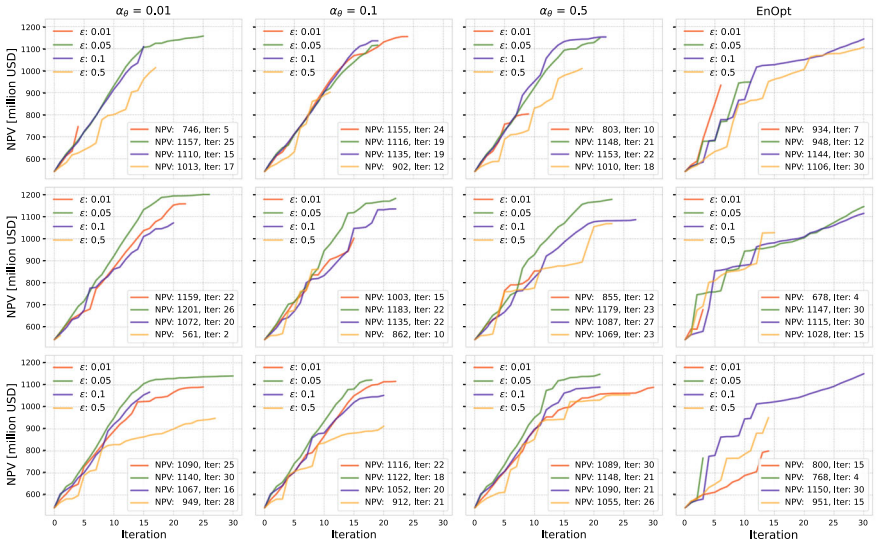

Fig. 8 Optimization results from all the Egg model runs. Each row corresponds to a different random seed. The first three panels of each row show four GenOpt optimization runs (each with a different ε) with the same mutation step size. The last panel of each row shows the respective EnOpt runs, with initial variance $\propto \varepsilon^2$. The y-axis shows the NPV increase as a function of the iteration (shown on the x-axis)

both set to 20.0 (for each marginal distribution) since these values renders the beta distribution symmetric and gives it a Gaussian-like shape that closely aligns with that of EnOpt's starting point. The ensemble size is set to $N_\mathrm{e} = 50$, and the CMA parameters are $\alpha_\lambda = \alpha_1 = 0.05$, $\lambda = 20$, and $\alpha_c = 0.4$. An initial time correlation for the controls in each well is also imposed. This ensures smooth temporal variations for the controls

$$\mathrm{Corr}(t, t+n) = \rho^n, \tag{63}$$

where $\rho = 0.5$ and n is the number of time steps (in terms of control changes) between the two times. Correlation between controls in different wells is disregarded during the optimization. The optimization is done using PET (developed by NORCE Energy Tehnology 2023), and the simulation uses the OPM simulator (Rasmussen et al. 2021). Figure 8 shows the increase of the NPV per iteration, where each row shows the results for a different random seed. The three first columns show results using GenOpt with different α_θ, and the right column shows EnOpt results. Each panel shows results using four different values for ε. For EnOpt, the initial variances are given by Eq. 61. The step-size for the covariance mutation in EnOpt is set to 0.2, this choice is based in previous experience with EnOpt on the Egg model. The final NPV and the number of iterations used to achieve that NPV for each line are shown in the legend at the bottom right corner of each panel. The maximum number of iterations is 30.

The results indicate that $\varepsilon = 0.05$ (green line) is a good choice for GenOpt in this particular case. Recall that the optimization space is transformed to $[0, 1]^d$. For EnOpt, however, the variance with $\varepsilon = 0.1$ (blue line) seems to be more stable. As expected, $\varepsilon = 0.5$ (yellow line) performs the worst for GenOpt as it allows for sampling of the entire region at each iteration.

The highest NPV reached by any of the optimization runs is achieved by GenOpt with $\alpha_\theta = 0.01$ and $\varepsilon = 0.05$. For this case the NPV was 1.201 billion USD after 26 iterations. The highest NPV reached by EnOpt was 1.150 billion USD in 30 iterations. It is possible that further improvements in the NPV could be achieved by increasing the maximum number of allowed iterations. The experiments indicate that GenOpt can reach a higher NPV quicker than EnOpt.

7 Summary

In this paper, non-Gaussian Ensemble Optimization is studied. The main motivation for this is the possible adaptive ability of a non-Gaussian distribution via mutation optimization, in addition to eliminating the problem of sample truncation in the case of bounded optimization. Stein's lemma offers an interpretation of EnOpt which makes the introduction of a general mutation distribution natural. An extension of EnOpt, denoted GenOpt, is proposed and tested, where Stein's lemma is used to estimate the gradient of the control vector and mutation optimization is applied to the sampling distribution. The multivariate extension is introduced using Gaussian copulas. The parameters of each marginal distribution are updated independently, and the correlation matrix in the copula is also adapted. Beta distributions are used as marginal distributions as they are bounded on a domain, and a transformation is proposed to ensure that all ensemble members lie within the allowed control bounds. This transformation introduces one extra parameter which relates to the initial variance of the distribution.

GenOpt is tested and compared to EnOpt on several test cases. First on the Rosenbrock function with increasing dimensionality, and later on a benchmark reservoir optimization test case. The experiments show that GenOpt is less sensitive to the choice of mutation step and suffers less from premature convergence. On the well control test case with the Egg model reservoir, GenOpt converges quicker than EnOpt.

Because GenOpt is a black-box optimization method, its potential applications are not restricted to reservoir management. In the future, it would be interesting to compare GenOpt and EnOpt in other areas. A potential example could be optimizing offshore wind farm layout, where EnOpt has already been used (Eikrem et al. 2023). In addition, testing GenOpt with a different marginal distribution can also be interesting. A short illustration of this is shown in "Appendix A" using a logistic marginal distribution. Another future test of great interest is to employ GenOpt on a robust optimization problem (as described in Chen et al. 2009), where the objective function is stochastic.

However, similar results are expected for a robust optimization problem since this would only affect the $J(X) - \bar{J}$ term in the gradient computation. Investigation into an adaptive scheme for ε could improve the efficiency of the algorithm.

Appendix A: Illustration with the Logistic Distribution

Although the numerical examples in this paper use beta marginals, the theory of GenOpt is formulated such that any distribution (bounded or not) can be used. Here, we present a short illustration of how one can easily use other marginal distributions with GenOpt, as shown with the logistic distribution. The pdf of the logistic distribution is given as

$$f(x|\mu, s) = \frac{e^{-(x-\mu)/s}}{s\left(1 + e^{-(x-\mu)/s}\right)^2}, \tag{64}$$

where μ is the mean of the distribution, and s is a scale parameter related to the variance by $\mathrm{Var}[X] = s^2\pi^2/3$. All that is needed to calculate the marginal contribution of the control gradient of GenOpt is to calculate $\partial_x \log f(x|\mu, s)$, which is

$$\partial_x \log f(x|\mu, s) = -\tanh\left(\frac{x-\mu}{2s}\right)/s. \tag{65}$$

This results in the following update equation for the ith component

$$x_{k+1}^i = x_k^i - \alpha_x \sum_{n=1}^{N} \left(J(X^n) - \bar{J}\right)\left(\tanh\left(\frac{X_i^n - x_k^i}{2s_k^i}\right)/s_k^i - \partial_{x_i} \log c_{\mathbf{R}}\right), \tag{66}$$

where $\partial_{x_i} \log c_{\mathbf{R}}$ is the contribution form the copula. Similarly, a mutation gradient for the scale parameter can be derived. However, this is quite lengthy and is therefore omitted.

Figure 9 shows the negative gradient (normalized by L_2 norm) of GenOpt with the logistic marginal distribution (from Eq. 66), the EnOpt gradient and the analytical gradient of the Rosenbrock function for points randomly chosen in the domain. For GenOpt and EnOpt, the ensemble size is set to 20, and the variance is set to 0.1. No correlation was used in the sampling, so that the copula term in Eq. 66 vanishes.

Acknowledgements The authors acknowledge funding from the Centre of Sustainable Subsurface Resources (Grant Number 331841) supported by the Research Council of Norway, research partners NORCE Norwegian Research Centre and the University of Bergen, and user partners Equinor ASA, Wintershall Dea Norge AS, Sumitomo Corporation, Earth Science Analytics, GCE Ocean Technology, and Schlumberger Norge AS.

Funding Open access funding provided by NORCE Norwegian Research Centre AS

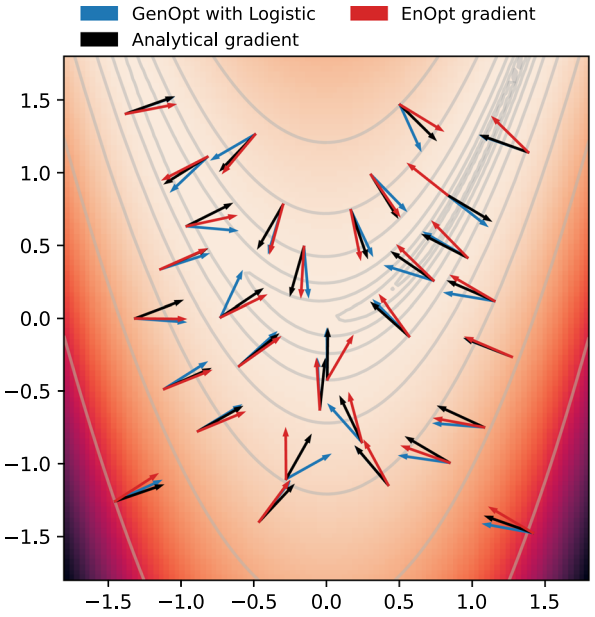

Fig. 9 The gradient of GenOpt (with logistic marginals) and EnOpt (red), together with the analytical gradient (black) of the Rosenbrock function for 32 randomly chosen points

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

Amari S (1998) Natural gradient works efficiently in learning. Neural Comput 10(2):251–276. https://doi.org/10.1162/089976698300017746

Amari S, Douglas S (1998) Why natural gradient? In: Proceedings of the 1998 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP '98 (Cat. No.98CH36181), pp 1213–1216, https://doi.org/10.1109/ICASSP.1998.675489

Chen Y, Oliver DS, Zhang D (2009) Efficient ensemble-based closed-loop production optimization. SPE J 14(04):634–645. https://doi.org/10.2118/112873-PA

Eikrem KS, Lorentzen RJ, Faria R, Stordal AS, Godard A (2023) Offshore wind farm layout optimization using ensemble methods. Renew Energy 216:119061. https://doi.org/10.1016/j.renene.2023.119061

Fonseca RM, Leeuwenburgh O, Van den Hof PM, Jansen JD (2013) Improving the ensemble optimization method through covariance matrix adaptation (CMA-EnOpt). In: SPE Reservoir Simulation Conference, The Woodlands, Texas, USA, https://doi.org/10.2118/163657-MS

Fonseca RRM, Chen B, Jansen JD, Reynolds A (2017) A stochastic simplex approximate gradient (StoSAG) for optimization under uncertainty. Int J Numer Methods Eng 109(13):1756–1776. https://doi.org/10.1002/nme.5342

Hansen N (2006) The CMA evolution strategy: a comparing review. Towards a new evolutionary computation: advances on estimation of distribution algorithms, vol 192. Springer, Berlin, pp 75–102. https://doi.org/10.1007/3-540-32494-1_

Hansen N, Ostermeier A (2001) Completely derandomized self-adaptation in evolution strategies. Evol Comput 9(2):159–195. https://doi.org/10.1162/106365601750190398

Hazarika S, Dutta S, Shen HW, Chen JP (2019) Codda: a flexible copula-based distribution driven analysis framework for large-scale multivariate data. IEEE Trans Vis Comput Graph 25(1):1214–1224. https://doi.org/10.1109/TVCG.2018.2864801

Jansen JD, Fonseca RM, Kahrobaei S, Siraj MM, Van Essen GM, Van den Hof PM (2014) The egg model-a geological ensemble for reservoir simulation. Geosci Data J 1(2):192–195. https://doi.org/10.1002/gdj3.21

Landsman Z, Nešlehová J (2008) Stein's Lemma for elliptical random vectors. J Multivar Anal 99(5):912–927. https://doi.org/10.1016/j.jmva.2007.05.006

Lorentzen RJ, Berg AM, Naevdal G, Vefring EH (2006) A new approach for dynamic optimization of waterflooding problems. In: SPE Intelligent Energy International Conference and Exhibition, https://doi.org/10.2118/99690-MS

NORCE energy & technology, data assimilation and optimization group (2023) Python Ensemble Toolbox (PET). https://github.com/Python-Ensemble-Toolbox/PET

Ramaswamy KR, Fonseca RM, Leeuwenburgh O, Siraj MM, Van den Hof PM (2020) Improved sampling strategies for ensemble-based optimization. Comput Geosci 24(3):1057–1069. https://doi.org/10.1007/s10596-019-09914-8

Rasmussen AF, Sandve TH, Bao K, Lauser A, Hove J, Skaflestad B, Klöfkorn R, Blatt M, Rustad AB, Sævareid O, Lie KA (2021) The open porous media flow reservoir simulator. Comput Math Appl 81:159–185. https://doi.org/10.1016/j.camwa.2020.05.014

Sarma P, Chen W (2014) Improved estimation of the stochastic gradient with Quasi-Monte Carlo methods. In: ECMOR XIV-14th European Conference on the Mathematics of Oil Recovery, vol 2014. European Association of Geoscientists and Engineers, pp 1–25, https://doi.org/10.3997/2214-4609.20141779

Shang YW, Qiu YH (2006) A note on the extended Rosenbrock function. Evol Comput 14(1):119–126. https://doi.org/10.1162/evco.2006.14.1.119

Stein CM (1981) Estimation of the mean of a multivariate normal distribution. Ann Stat 9(6):1135–1151. https://doi.org/10.1214/aos/1176345632

Stordal AS, Szklarz SP, Leeuwenburgh O (2016) A theoretical look at ensemble-based optimization in reservoir management. Math Geosci 48:399–417. https://doi.org/10.1007/s11004-015-9598-6

Sun Y, Wierstra D, Schaul T, Schmidhuber J (2009) Efficient natural evolution strategies. In: Proceedings of GECCO, pp 539–545

Wierstra D, Schaul T, Peters J, Schmidhuber J (2008) Natural evolution strategies. In: 2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence), pp 3381–3387, https://doi.org/10.1109/CEC.2008.4631255

Wierstra D, Schaul T, Glasmachers T, Sun Y, Peters J, Schmidhuber J (2014) Natural evolution strategies. J Mach Learn Res 15(27):949–980 (http://jmlr.org/papers/v15/wierstra14a.html)

Zhou E, Hu J (2014) Gradient-based adaptive stochastic search for non-differentiable optimization. IEEE Trans Autom Control 59(7):1818–1832. https://doi.org/10.1109/TAC.2014.2310052