Toward Large-Scale Continuous EDA: A Random Matrix Theory Perspective

A. Kabán A.Kaban@cs.bham.ac.uk
School of Computer Science, University of Birmingham, Edgbaston, B15 2TT, Birmingham, UK

J. Bootkrajang Jakramate.B@cmu.ac.th
Department of Computer Science, Chiang Mai University, Muang, Chiang Mai 50200, Thailand

R. J. Durrant BobD@waikato.ac.nz
Department of Statistics, University of Waikato, Private Bag 3105, Hamilton 3240, New Zealand

doi:10.1162/EVCO_a_00150

Abstract

Estimations of distribution algorithms (EDAs) are a major branch of evolutionary algorithms (EA) with some unique advantages in principle. They are able to take advantage of correlation structure to drive the search more efficiently, and they are able to provide insights about the structure of the search space. However, model building in high dimensions is extremely challenging, and as a result existing EDAs may become less attractive in large-scale problems because of the associated large computational requirements. Large-scale continuous global optimisation is key to many modern-day real-world problems. Scaling up EAs to large-scale problems has become one of the biggest challenges of the field. This paper pins down some fundamental roots of the problem and makes a start at developing a new and generic framework to yield effective and efficient EDA-type algorithms for large-scale continuous global optimisation problems. Our concept is to introduce an ensemble of *random projections* to low dimensions of the set of fittest search points as a basis for developing a new and generic divide-and-conquer methodology. Our ideas are rooted in the theory of random projections developed in theoretical computer science, and in developing and analysing our framework we exploit some recent results in nonasymptotic random matrix theory.

Keywords

Large-scale optimisation, estimation of distribution algorithms, random projections, random matrix theory

1 Introduction

Estimations of distribution algorithms (EDAs) are population-based stochastic blackbox optimisation methods that are recognised as a major paradigm of evolutionary computation (EC) (Larrañaga and Lozano, 2002). Unlike most traditional EC approaches, which have no explicit mechanism to take advantage of any correlation structure in the sample of high-fitness individuals, EDAs guide the search for the global optimum by estimating the distribution of the fittest sample and drawing new candidates from this distribution. One of the unique advantages stemming from this approach is that the

Manuscript received: October 28, 2013; revised: August 20, 2014; accepted: March 30, 2015.

parameter estimates in EDA are often interpretable and may shed light on the problem structure.

However, it has been widely observed that as the search space dimensionality increases, model building becomes more difficult and declines in effectiveness (Omidvar and Li, 2011; Dong et al., 2013). Indeed, attempts to use the full power of multivariate model building, such as the estimation of multivariate normal algorithm (EMNA), when the search space exceeds 50–100 dimensions, have been scarce. The current practice of EDA most often resorts to independence models or models with some limited dependency structure (Wang and Li, 2008; Bosman et al., 2013; Dong et al., 2013; Ros and Hansen, 2008) in exchange for feasibility when the problem is high-dimensional. Some authors employ univariate heavy tail search distributions; for example, Wang and Li (2008) propose a univariate EDA (UMDAc) with Gaussian and Lévy search distribution for large-scale EDA, and while this improves the exploration ability to some extent, a univariate model unfortunately means that nonseparable problems cannot be tackled adequately—a fact both proved theoretically (Mühlenbein and Mahnig, 1999; Larrañaga and Lozano, 2002) and shown experimentally (Echegoyen et al., 2011).

More refined univariate methods are sep-CMA-ES (Ros and Hansen, 2008) and the univariate version of AMaLGaM (Bosman, 2009); these only estimate the diagonal entries of the sample covariance matrix to reduce the search cost of model building, although in a different way than UMDAc does. By construction, these methods are aimed at dealing with dimensionwise separable problems, and this serves as an approximation of nonseparable problems with few dependencies. In practice, these independence factorisations turn out to be more effective than fully dependent models in high dimensions (Ros and Hansen, 2008; Bosman, 2009), because estimating a reliable fully dependent model requires considerably larger population sizes, which then consumes the budget of function evaluations rapidly, in addition to an increased per-generation time and space complexity. In a different vein, L-CMA-ES (Knight and Lunacek, 2007) addresses the latter issue and obtains savings in terms of the time and the space complexity over the full covariance CMA-ES by employing a limited memory version. However, this does not reduce (but slightly increases) the number of function evaluations taken to reach a target value.

Large-scale continuous optimisation problems are one of the most important concerns in evolutionary computation research because they appear in many real-world problems (Tang et al., 2009; Molina et al., 2010; Omidvar and Li, 2011) such as data mining and biocomputing (Sun et al., 2012), robotics and computational vision (Simonyan et al., 2014). Competitions are organised each year at major conferences, notably the Congress on Evolutionary Computation (CEC), to promote research in this area. Indeed, many optimisation methods suffer from the curse of dimensionality and deteriorate quickly when the search space dimension increases to the thousands. The current target at these competitions is difficult nonseparable problems on 1,000-dimensional search spaces. The state-of-the-art best performers are EC methods that use cooperative co-evolution (Yang et al., 2008a), multilevel co-evolution (Yang et al., 2008b), and hybrid methods that include local searches (Molina et al., 2010). EDA approaches did not yet feature in these competitions.

Our motivation in this work is as follows. It is often infeasible to obtain the exact solution to complicated high-dimensional nonseparable problems. Hence, it is desirable to develop alternative approaches with differing search biases that are able to obtain approximate solutions with a limited budget. By reducing the degrees of freedom in the parametrisation of the search distribution, the preceding methods were able to achieve a better scaling in terms of the search costs, and various ways of doing this

induce different search biases; for instance, UMDAc has a bias for separable problems, EDA-MCC (Dong et al., 2013) has a bias for block-diagonal dependency structures; sep-CMA-ES has a bias for separable problems. Each method is more likely to succeed on problems that are similar or not too different from their own search biases.

What these methods have in common in their way of restricting the covariance is a binary decision-making step by which they estimate certain dependencies and neglect certain others. The EDA-type approach we develop in this paper keeps with the general idea of reducing the degrees of freedom of the covariance, but without making such binary decisions on any of the individual dependencies. Instead of dependency selection, we use compression. More specifically, we simply work with a combination of compressed versions of EMNA's full covariance estimate. This avoids rank deficiency and misestimation of the covariance when the sample size is small; it is by construction invariant to rotations of the search space; and from the perspective of optimisation it creates a different kind of search bias by which separable problems are no longer necessarily the easy type but in turn some of the more sophisticated nonseparable problems may become more manageable. Our goal is to find approximate solutions to difficult problems within a limited budget of function evaluations, and we demonstrate the effectiveness of our approach on the CEC'10 competition benchmark suite (Tang et al., 2009) that was designed with this goal in mind, mimicking real-world scenarios. In addition, despite our building on EMNA, the time complexity per generation of our method is only quadratic in the problem dimension whereas EMNA's is cubic. Moreover, our approach lends itself naturally to parallel implementation, since the problem of estimating the search distribution can be split over several cores.

Section 2 discusses some fundamentals of the problem of model estimation in high-dimensional probability spaces that motivated our approach. Section 3 introduces our new approach along with an analysis of its working. In Section 4 we demonstrate that this approach is competitive with the state-of-the-art experiments on a battery of test functions. Section 5 concludes the paper. A preliminary version of this work appeared in Kabán et al. (2013). The MATLAB code for our present work is available from http://www.cs.bham.ac.uk/∼axk/rpm.zip

2 The Challenges of Model Estimation in High Dimensions

Let us examine a typical EDA optimisation scheme. Consider the multivariate Gaussian search distribution. Let $x^* \in \mathbb{R}^d$ denote the global optimum, and let $B(x^*, \epsilon)$ be the d-dimensional Euclidean ball with centre x^* and radius ϵ. By definition,

$$\Pr_{x\sim\mathcal{N}(\mu,\Sigma)}[\|x - x^*\| \leq \epsilon] = \int_{x\in B(x^*,\epsilon)} \mathcal{N}(x|\mu, \Sigma)dx \tag{1}$$

is the probability that a draw from the search distribution parametrised by μ and Σ falls in the ϵ-neighbourhood of the global optimum.

In EMNA, the parameters $\mu \in \mathbb{R}^d$ and $\Sigma \in \mathbb{R}^{d\times d}$ are maximum likelihood estimates (MLE) from N' selected search points of the population. Hence Σ is a matrix-valued random variable, that is, a random matrix. Analytic tools are available from random matrix theory (RMT) to analyse random matrices, which were also used in statistics to analyse covariance estimation problems (Vershynin, 2012b; Srivastava and Vershynin, 2013) and which previously have never been exploited in EDA optimisation.

We start by noting that the eigenvalues of the covariance estimate used in EDA to generate the new generation of individuals play the role of some learning rates for the optimisation process. This observation is certainly not new; the widely successful covariance matrix adaptation evolutionary strategy (CMA-ES) (Hansen, 2006) builds

on this observation also. Here we use this observation to highlight what goes wrong with EMNA in high dimensions, which then opens up new options to deal with the problem with the use of new tools.

To see this, note that by the mean value theorem for multivariate definite integrals (Apostol, 1957, p. 401), there exists a point in the ball of radius ϵ around x^*, such that Eq. (1) can be written as follows:

$$\exists \tilde{x} \in B(x^*, \epsilon), \text{ s.t. } \Pr_{x\sim\mathcal{N}(\mu,\Sigma)}[\|x - x^*\| \leq \epsilon] = \text{Volume}(B(x^*, \epsilon))\mathcal{N}(\tilde{x}|\mu, \Sigma), \quad (2)$$

where $B(x^*, \epsilon)$ is the ball centred at x^* with radius ϵ. Switching to the eigen-basis of Σ, this further equals

$$\text{Volume}(B(x^*, \epsilon)) \prod_{i=1}^{d} \mathcal{N}(U_i^T(\tilde{x} - \mu)|0, \lambda_i), \quad (3)$$

where U_i denotes the ith eigenvector of Σ, and λ_i is its associated eigenvalue.

Now, we want our search strategy to maximise Eq. (1), namely, the probability that the multivariate Gaussian search distribution $\mathcal{N}(\mu, \Sigma)$ reaches the global optimum. The effect of the eigenvalue λ_i can be read from the partial derivative of the right-hand side of Eq. (3) with respect to λ_i, which is

$$\frac{\delta}{\delta\lambda_i} = \text{Vol}(B(x^*, \epsilon)) \prod_{j\neq i}^{d} \mathcal{N}(U_j^T\tilde{x}|U_j^T\mu, \lambda_j) N(U_i^T\tilde{x}|U_i^T\mu, \lambda_i) \left(\frac{\|U_i^T(\tilde{x} - \mu)\|^2}{\lambda_i} - 1 \right) \frac{1}{2\lambda_i}. \quad (4)$$

From Eq. (4) we see that

- if $\lambda_i < \|U_i^T(\tilde{x} - \mu)\|^2$, then the probability in Eq. (1) is an increasing function of λ_i,
- if $\lambda_i > \|U_i^T(\tilde{x} - \mu)\|^2$, then the probability in Eq. (1) is a decreasing function of λ_i,

and so the optimal value of the ith eigenvalue of Σ is the squared length of the projection of $\tilde{x} - \mu$ onto the corresponding eigendirection, namely, $\lambda_i^{\text{opt}} = \|U_i^T(\tilde{x} - \mu)\|^2$. In other words, when $\|U_i^T(\tilde{x} - \mu)\|^2 > \lambda_i$, then the probability (1) of drawing a point in the ϵ-neighbourhood of x^* can be increased by increasing λ_i. On the other hand, when $\|U_i^T(\tilde{x} - \mu)\|^2 < \lambda_i$, then (1) can be increased by decreasing λ_i. Hence the eigenvalues of Σ play the role of learning rates in Gaussian EDA, and good estimates of these eigenvalues are essential.

Unfortunately, as is well known from RMT, in small sample conditions the smallest eigenvalue is *severely* underestimated, while the largest eigenvalue is overestimated. An example is shown in Figure 1, where we generated 100 points from a 100-dimensional Gaussian with identity covariance, yet the sample covariance of those 100 points has eigenvalues ranging all the way from close to 0 to around 4.

The extent of this misestimation is well understood in RMT, and indeed this theory has given rise to new methods of covariance estimation (Marzetta et al., 2011; Durrant and Kabán, 2015) that are able to remedy the problem effectively even when Σ is singular, using an ensemble of random projections of the covariance estimate. In this work we make extensive use of these results.

These recent RMT-based covariance estimation methods have been found to have certain advantages, for example, in comparison with the Ledoit-Wolf estimator (Marzetta et al., 2011) in terms of approximating the true covariance; they also performed better in data classification (Durrant and Kabán, 2015). Furthermore, these

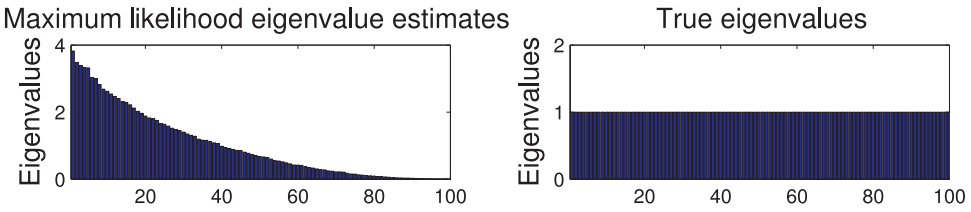

Figure 1: Eigenvalue misestimation from $N' = 100$ points in $d = 100$ dimensions. The horizontal axis runs though the indices of the ordered list of eigenvalues; the vertical axis shows the magnitude of each eigenvalue.

RMT-based methods do not impose a predefined and possibly unjustified structural constraint such as sparsity, diagonality, block-diagonal structure, or limited dependency structure. Instead the reduction of the degrees of freedom comes from exploiting randomised compressions of the maximum likelihood covariance estimate. Moreover, for EDA-type search, these new estimators also have computational advantages, since we only have to sample from a number of small-dimensional multivariate Gaussians. Hence our approach lends itself to parallel implementation that fits well with the algorithmic structure of population-based search, which could potentially be further exploited when the problem scale requires it.

3 Approach to Large-Scale Stochastic Optimisation

The main goal of this paper is to develop a new approach to large-scale stochastic optimisation in application to EDA that is effective and computationally efficient in finding approximate solutions to difficult problems with limited search costs. The term *difficult* is of course relative; here we use it to refer to problems that cannot be solved by existing specialised optimisation methods and that have not yet been solved to a satisfactory extent by other existing heuristic optimisation methods.

Building on recent results in other areas, our concept is to introduce an ensemble of random projections that reduce the dimensionality of the fittest high-dimensional search points. Random projection (RP) is often termed a nonadaptive dimensionality reduction method, since it projects the data in directions that are uniformly randomly chosen (independently of the data being projected, as opposed to, for instance, principal component analysis, where the projection directions are determined as a function of the data). Perhaps surprisingly for common intuition, RPs enjoy nice theoretical properties, most notably a high probability guarantee of low distortion of the Euclidean geometry, and the reduced space makes subsequent computations, estimation, and sampling easier. This approach provides us a basis for developing a new and generic divide-and-conquer methodology rooted in the theory of RPs and exploiting recent advances of nonasymptotic random matrix theory and related fields.

At a high level, the rationale is as follows:

- Random matrices that satisfy the Johnson-Lindenstrauss lemma (JLL) (Dasgupta, 1999) are approximate isometries. Hence, with appropriate choice of the target dimension, important structures such as Euclidean distances and dot products are approximately preserved in the reduced space. This makes it possible to capture correlations between the d-dimensional search variables in the $k \ll d$-dimensional space.

- In the low-dimensional projection space the distribution of the projected points can become "more Gaussian" as a consequence of the central limit theorem, in a sense made precise by Diaconis and Freedman (1984). Also, both parameter estimation and sampling become feasible and computationally affordable, so there is no need to overly restrict the parametric form of the search distribution and its covariance matrix.
- There is a natural smoothing effect that emerges when appropriately combining the ensemble of estimates from several random subspaces (Mahoney, 2011; Marzetta et al., 2011; Durrant and Kabán, 2015). This ensures that the exploration ability of the search distribution can be maintained even with small population sizes.

Random projections have been used in approximation theory since the 1970s (Lorentz et al., 1996). In computer science, information theory, signal processing, and machine learning, random matrices provide a mechanism for dimensionality reduction while preserving the essential information in the data (Vempala, 2004). Compared with other methods in that context, they lead to (1) faster algorithms that are (2) simpler to analyse, (3) lend themselves to parallel implementation, and (4) exhibit robustness. The reader may refer to the recent review by Mahoney (2011). We aim to exploit these characteristics for high-dimensional optimisation.

3.1 New Search Operators for EDA

Let $R \in \mathbb{R}^{k\times d}$ be a random matrix with entries drawn i.i.d. from a univariate Gaussian $\mathcal{N}(0, \sigma^2)$. When d is large, as a consequence of the measure concentration phenomenon in high dimensions, the rows of this matrix are almost orthogonal and have a Euclidean norm close to their expected value, which is $\sigma\sqrt{d}$ (Dasgupta, 1999; Vempala, 2004). So if we choose $\sigma^2 = 1/d$, then R well approximates an orthonormal matrix to project from $\mathbb{R}^d$ to $\mathbb{R}^k$, where k may be chosen much lower than d.

Further, let $x_0 \in \mathbb{R}^d$ be a point in the search space. Denote by $\mathcal{S}^R_{x_0}$ the unique affine subspace parallel to R that passes through x_0. We define new search operators as follows:

`Project` takes a (random) $R \in \mathbb{R}^{k\times d}$, an $x_0 \in \mathbb{R}^d$, and a sample $\mathcal{P}^{fit} = (x_i \in \mathbb{R}^d)_{i=1:N'}$, and projects $\mathcal{P}^{\text{fit}}$ onto $\mathcal{S}^R_{x_0}$, that is, returns $\mathcal{P}_R = (R^T R(x_i - x_0) + x_0)_{i=1:N'}$.

`sEstimate` takes a sample $\mathcal{P}_R$ that lives in a subspace $\mathcal{S}^R_{x_0}$ and computes the maximum likelihood parameter estimates $\hat{\theta}_R$ (for Gaussian search distribution, $\hat{\theta}_R = (\hat{\mu}_R, \hat{\Sigma}_R)$) of the search distribution $\mathcal{D}_R$, which is with respect to (w.r.t.) the restriction of the Lebesgue measure to the k-dimensional affine subspace $\mathcal{S}^R_{x_0}$.

`sSample` takes parameter estimates $\hat{\theta}_R$ obtained by `sEstimate` and returns a sample of N points drawn i.i.d. from $\mathcal{D}_R$ with parameters $\hat{\theta}_R$. These points will live in a k-dimensional affine subspace of the search space.

`Combine` takes populations from several k-dimensional subspaces $\mathcal{S}^{R_i}_{x_0}, i = 1, \ldots, M$ and returns a population that lives in the full search space $\mathbb{R}^d$.

Using these operators, the high-level outline of our meta-algorithm is as follows:

(1) Initialise population $\mathcal{P}$ by generating N individuals uniformly randomly.

Algorithm 1

Denote by $\mathcal{P}^{\text{fit}} = \{x_1, \ldots, x_{N'}\}$ the set of N' selected fit individuals, and let N be the population size.

1. *Inputs*: $\mathcal{P}^{\text{fit}}, M, k,$ where $M \geq \lceil d/k \rceil$
2. Estimate $\mu := mean(\mathcal{P}^{\text{fit}})$
3. For $i = 1, \ldots, M$

   Generate a random projection matrix R_i

   Project the centred points into k-dimensions:
   $\mathbf{Y}^{R_i} := [R_i(x_n - \mu); n = 1, \ldots, N']$

   Estimate the $k \times k$ sample covariance Σ^{R_i}

   Sample N new points $y_1^{R_i}, \ldots, y_N^{R_i} \overset{iid}{\sim} \mathcal{N}(0, \Sigma^{R_i})$
4. Let the new population $\mathcal{P} := \sqrt{\frac{dM}{k}}[\frac{1}{M}\sum_{i=1}^{M} R_i^T y_1^{R_i}, \ldots, \frac{1}{M}\sum_{i=1}^{M} R_i^T y_N^{R_i}] + \mu$
5. *Output*: $\mathcal{P}$

(2) Let $\mathcal{P}^{\text{fit}}$ be the fittest $N' < N$ individuals from $\mathcal{P}$.

(3) For $i = 1, \ldots, M$ $(M \geq 1)$ randomly oriented (affine) $k < d$-dimensional subspaces $\mathcal{S}_{x_0}^{R_i}$
   a. Project $\mathcal{P}^{\text{fit}}$ onto $\mathcal{S}_{x_0}^{R_i}$.
   b. Produce N new individuals on the subspace $\mathcal{S}_{x_0}^{R_i}$ using the sequence `sEstimate`; `sSample`.

(4) Create the new population $\mathcal{P}$ using `Combine`.

(5) If stopping criteria are met then Stop; else Goto 2.

We instantiate this by taking the translation vector x_0 of the consecutive set of subspaces (in consecutive generations) to be the mean of $\mathcal{P}^{\text{fit}}$ in the previous generation. Further, in this work we instantiate the `Combine` operator as a scaled average of the individuals produced on the individual subspaces. Note, this simple combination scheme makes no appeal to fitness evaluation within subspaces.

The scaling just mentioned is important. An orthogonal projection from $\mathbb{R}^d$ to $\mathbb{R}^k$ shortens the lengths of vectors by a factor of $\sqrt{k/d}$, and averaging M i.i.d. points reduces their standard deviation by a factor of $\sqrt{M}$, hence a scaling factor of $\sqrt{(dM)/k}$ is needed to recover the original scale. This is the case when the entries of R were drawn with variance $\sigma^2 = 1/d$. With generic σ^2 the appropriate scaling is $\sqrt{M/(k\sigma^2)}$.

In Algorithm 1, we now present the specific algorithm, which is an instantiation of the module for creating the new generation (steps 3 and 4 of the foregoing list).

Note that in practice the loop in step 3 of Algorithm 1 can be split over multiple cores, since each random subspace is both generated and sampled from, independently of all the others.

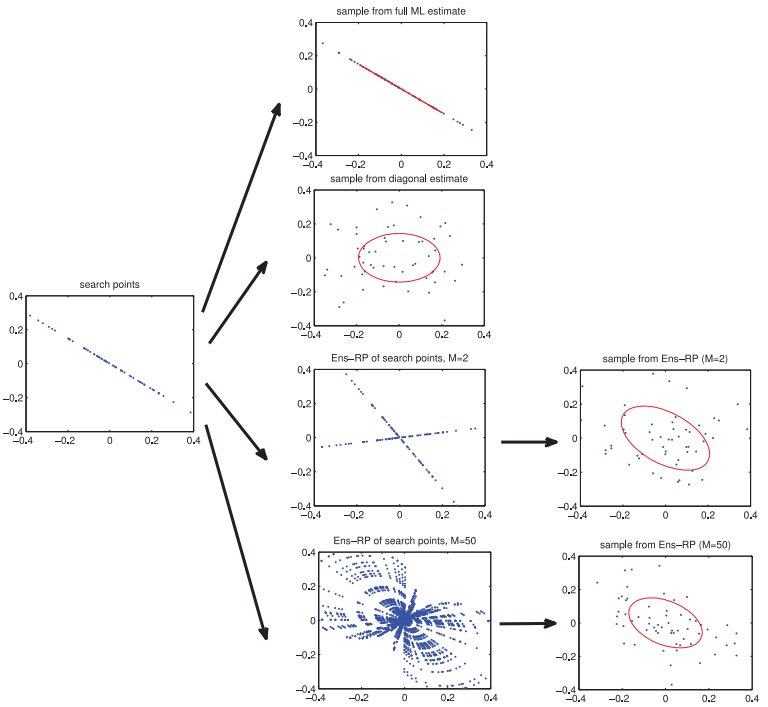

Figure 2: Illustration of the use of random projections for sampling the new population.

The working of this method is illustrated in Figure 2, with the caveat that high-dimensional geometry is hard to capture on a 2D figure and should be read as follows. In large-scale problems, in order to remain search cost effective, we would often like to work with a population size that is no larger than the problem dimensionality (Dong et al., 2013), since a large population size would consume the search budget rapidly. Then the number of fit points N' becomes smaller than the dimension of the search space d; hence the fit individuals live in the N'-dimensional subspace of the search space determined by their span. The leftmost plot in Figure 2 illustrates a situation where some N' points live in a subspace (here 1D) of the overall space (here 2D). Hence, the maximum likelihood (ML) covariance estimate of the fit points is singular. Sampling points from a distribution with a singular covariance means that the next generation is confined in the same subspace. The top plot in the central column of Figure 2 illustrates this. Now, to get around this problem, univariate methods like UMDAc impose a diagonality constraint, that is, estimate only the variances. Hence the next generation is allowed in the full search space; however, any connection between the orientation of the fitness density of the parent population and its estimate is lost as a result of neglecting the correlations. This is seen in the second plot in the central column of Figure 2. The remaining plots in the central column show what happens when we use a random projection ensemble. The upper one shows a case where the number of random subspaces is the smallest that still spans the full search space, and the lower one shows a case where a large number of random subspaces are used. In both cases, the fit points are projected onto each of the random subspaces, and a new generation is sampled within each subspace. The new individuals from these multiple

worlds are then averaged to give the new generation shown in the rightmost column of Figure 2 together with the ML covariance estimate of this new population. We see that the new ML covariance estimate tends to respect the orientation of the fitness density of the parent population while it also eliminates degeneracy. It is also easy to picture that the probability recovering the correct orientation gets higher as the number of random projections gets larger. This is because the resulting outcome from a very small number of uniformly random directions have a higher variability, whereas this variation diminishes as we add more random directions into the combination.

3.2 Analysis of the Algorithm That Creates New Generations

To understand the effect of Algorithm 1, we analyse it by examining the new full-rank search distribution in the original search space that it implicitly implements. We stress, however, that all *estimation* and *sampling* take place in the k-dimensional projected spaces $\mathcal{S}_\mu^{R_i}$, and it is this fact that enables us to finesse both the computational issues associated with sampling in the high-dimensional search space and the degeneracy of the covariance estimate in the search space.

Fix the set of selected fit individuals $\mathcal{P}^{\text{fit}}$, and denote by Σ the maximum likelihood estimate of their sample covariance. This covariance estimate is never computed explicitly throughout the algorithm, but it is useful for the theoretical analysis of this section.

Now, it is straightforward to verify that, by construction, conditionally on the matrices R_i, $i = 1, \ldots, M$, the new population, $\mathcal{P}$, obtained by Algorithm 1, is distributed i.i.d. as $\mathcal{N}(\mu, \frac{d}{k}[\frac{1}{M}\sum_{i=1}^{M} R_i^T R_i \Sigma R_i^T R_i])$. However, while Σ is singular when N' is smaller than d, the matrix $\frac{d}{k}[\frac{1}{M}\sum_{i=1}^{M} R_i^T R_i \Sigma R_i^T R_i]$ is almost surely (a.s.) positive definite provided $M \geq \lceil d/k \rceil$ and R_i are Gaussian with independent entries, or Haar random matrices (i.e., matrices with orthonormal rows, drawn with a uniformly random orientation). So with this minimum number of RPs we already avoid degeneracy and the problem of getting stuck in a bad subspace as a result. Of course, in order to also recover the correct orientation of the covariance, we need to use a large enough number of random projections so that the finite average gets close to its infinite limit. Fortunately, the law of large numbers guarantees that this is feasible, and the concentration of measure for sums of random matrices lets us quantify the gap between them. This analysis is detailed in the next sections. The resulting full d-dimensional covariance matrix provides information about the correlations between the original search variables, and may be used for learning about the problem structure in the usual way as is normally done in EDA.

3.2.1 Infinitely Many Random Projections

Recall that the random projections R_i are drawn independent and identically distributed (i.i.d.). Therefore, fixing Σ, by the law of large numbers, the ensemble may be thought of as a finite approximation of the expectation

$$\frac{1}{M}\sum_{i=1}^{M} R_i^T R_i \Sigma R_i^T R_i \underset{M\to\infty}{\longrightarrow} \mathrm{E}_R[R^T R \Sigma R^T R], \tag{5}$$

and we can understand the effect of the RP ensemble by computing this expectation.

For Haar random matrices, this expectation was computed by Marzetta et al. (2011).

LEMMA 1 (Marzetta et al., 2011): *Let R be a $k \times d$ Haar random matrix (i.e., having orthonormal rows in a uniformly random orientation), $k < d$, and Σ a $d \times d$ fixed positive semi-definite*

matrix. Then,

$$E_R[R^T R\Sigma R^T R] = \frac{k}{d}\left(\frac{dk-1}{d^2-1}\Sigma + \frac{d-k}{d^2-1}Tr(\Sigma)I_d\right), \tag{6}$$

where $Tr(\cdot)$ denotes the trace of its argument, and I_d is the d-dimensional identity matrix.

Observe that the covariance estimate in Lemma 1 is k/d times a convex combination of the maximum likelihood covariance estimate Σ and the spherical covariance estimate $\frac{Tr(\Sigma)}{d}I_d$, since the combination coefficients sum to 1: $\frac{dk-1}{d^2-1} + \frac{d(d-k)}{d^2-1} = 1$. Hence this method interpolates between the unconstrained maximum likelihood covariance as in EMNA and the spherically constrained estimate, and the balance between these two components is controlled by the size of k relative to d. Observe also that the coefficient of the first term is $\frac{dk-1}{d^2-1} = \mathcal{O}(1/d)$ (for a constant k), while that of the second term is $\frac{d(d-k)}{d^2-1} = \mathcal{O}(1)$. So for a constant k, the higher the problem dimension, the less weight is put on the full unconstrained maximum likelihood covariance estimate.

Now, from the observations at the beginning of Section 3.1, when d is large we may obtain a similar effect from using R_i with i.i.d. Gaussian entries. The following lemma computes the matrix expectation in Eq. (5) under such Gaussian R_i, which also turns out to have a closed form. In fact, this matrix expectation is available in closed form for a much larger class of random matrices, too (Kabán, 2014).

LEMMA 2: *Let R be a $k \times d$ random matrix, $k < d$, with entries drawn i.i.d. from $\mathcal{N}(0, \sigma^2)$; and Σ a $d \times d$ fixed positive semi-definite matrix. Then,*

$$E_R[R^T R\Sigma R^T R] = \sigma^4 k((k+1)\Sigma + Tr(\Sigma)I_d). \tag{7}$$

Before starting the proof, we observe that for $\sigma^2 = 1/d$, we get $\mathrm{E}_R[R^T R\Sigma R^T R] = \frac{k}{d}\left(\frac{k+1}{d}\Sigma + \frac{Tr(\Sigma)}{d}I_d\right)$, which is k/d times a linear combination of the maximum likelihood covariance estimate and the spherical covariance estimate (now the coefficients sum to $1 + (k+1)/d$), and again the coefficient of the first term is $\mathcal{O}(1/d)$, while that of the second is $\mathcal{O}(1)$. The Gaussian R_i is more convenient to use, since we do not need to orthogonalise its rows, and for large d problems with small k we indeed experienced no difference in their behaviour. However, for experiments aimed at quantifying the effect of k, the Haar matrices are more appropriate to use so that the interpolation effect is captured precisely though the convex combination.

PROOF OF LEMMA 2: Make the eigendecomposition $\Sigma = U\Lambda U^T$, where $UU^T = I_d$. Then we can rewrite:

$$\mathrm{E}[R^T R\Sigma R^T R] = \mathrm{E}[R^T RU\Lambda U^T R^T R] \tag{8}$$

$$= \mathrm{E}[UU^T R^T RU\Lambda U^T R^T RUU^T]. \tag{9}$$

Note, the Gaussian distribution is rotation-invariant, so RU has the same distribution as R. So we can absorb U into R and have the right-hand side of Eq. (9) further equal to

$$\mathrm{E}[UR^T R\Lambda R^T RU^T] = U\mathrm{E}[R^T R\Lambda^T R^T R]U^T. \tag{10}$$

Therefore it is enough to compute $\mathrm{E}[R^T R\Lambda R^T R]$ with Λ being diagonal.

We rewrite the expectation in Eq. (10). Denote by r_i the ith column of R, and by ρ the rank of Σ. Then we can rewrite:

$$\mathrm{E}_R[R^T R\Lambda R^T R] = \sum_{i=1}^{\rho}\lambda_i \begin{bmatrix} \mathrm{E}[(r_1^T r_i)^2] & \dots & \mathrm{E}[(r_1^T r_i)(r_i^T r_d)] \\ \vdots & \ddots & \vdots \\ \mathrm{E}[(r_d^T r_i)(r_i^T r_1)] & \dots & \mathrm{E}[(r_d^T r_i)^2]. \end{bmatrix}. \tag{11}$$

We first compute the diagonal elements of a generic term of this sum. These have the form $\mathrm{E}[(r_j^T r_i)^2]$. We need to take separately the cases when $j = i$ and when $j \neq i$.

Case $j = i$:

$$\begin{aligned}\mathrm{E}[(r_i^T r_i)^2] &= \mathrm{E}\left[\left(\sum_{j=1}^{k} r_{ji}^2\right)^2\right] = \sum_{j=1}^{k}\sum_{j'=1}^{k}\mathrm{E}[r_{ji}^2 r_{j'i}^2] \\ &= \sum_{j=1}^{k}\sum_{\substack{j'=1 \\ j'\neq j}}^{k}\mathrm{E}[r_{ji}^2]\mathrm{E}[r_{j'i}^2] + \sum_{j=1}^{k}\mathrm{E}[r_{ji}^4] \\ &= (k^2-k)\sigma^4 + 3k\sigma^4 \\ &= \sigma^4 k(k+2). \end{aligned} \tag{12}$$

Case $j \neq i$:

$$\begin{aligned}\mathrm{E}[(r_i^T r_j)^2] &= \mathrm{E}\left[\left(\sum_{\ell=1}^{k} r_{\ell i} r_{\ell j}\right)^2\right] = \sum_{\ell=1}^{k}\sum_{\ell'=1}^{k}\mathrm{E}[r_{\ell i} r_{\ell j} r_{\ell' i} r_{\ell' j}] \\ &= \sum_{\ell=1}^{k}\sum_{\substack{\ell'=1 \\ \ell'\neq\ell}}^{k}\mathrm{E}[r_{\ell i}]\mathrm{E}[r_{\ell j}]\mathrm{E}[r_{\ell' i}]\mathrm{E}[r_{\ell' j}] + \sum_{\ell=1}^{k}\mathrm{E}[r_{\ell i}^2 r_{\ell j}^2] \\ &= \sigma^4 k. \end{aligned} \tag{13}$$

Next, we compute the off-diagonal elements. These have the form $\mathrm{E}[(r_j^T r_i)(r_i^T r_\ell)]$ with $j \neq \ell$.

$$\begin{aligned}\mathrm{E}[(r_j^T r_i)(r_i^T r_\ell)] &= \mathrm{E}\left[\left(\sum_{m=1}^{k} r_{mi} r_{mj}\right)\left(\sum_{m'=1}^{k} r_{m'i} r_{m'\ell}\right)\right] \\ &= \sum_{m=1}^{k}\sum_{m'=1}^{k}\mathrm{E}[r_{mi} r_{mj} r_{m'i} r_{m'\ell}] \\ &= 0, \end{aligned} \tag{14}$$

by the independence of the entries of R and the fact that they have zero mean. Indeed, since $j \neq \ell$, the product inside this expectation will always have at least one independent entry of R on its own.

Hence we obtain that for diagonal Λ, $\mathrm{E}_R[R^T R \Lambda R^T R]$ is a diagonal matrix, and in particular it follows that if Σ is diagonalised as $\Sigma = U \Lambda U^T$, then U also diagonalises $\mathrm{E}_R[R^T R \Sigma R^T R]$.

Now, by putting together Equations (12), (13), and (14), after a little algebra we obtain

$$\mathrm{E}[R^T R \Lambda R^T R] = \sigma^4 k \left(\mathrm{Trace}(\Lambda) I_d + (k+1)\Lambda\right). \tag{15}$$

Finally, bringing the orthogonal matrices U and U^T back into the picture, we find that in expectation we obtain a regularised version of the sample covariance estimate,

$$U\,\mathrm{E}[R^T R \Lambda R^T R] U^T = \mathrm{E}[R^T R \Sigma R^T R] = \sigma^4 k \left(\mathrm{Trace}(\Sigma) I_d + (k+1)\Sigma\right), \tag{16}$$

which concludes the proof of Lemma 2. ■

In consequence, in the limit of $M \to \infty$ the new population $\mathcal{P}$ returned by Algorithm 1 will be distributed i.i.d. as $\mathcal{N}\left(\mu, \frac{\text{Trace}(\Sigma)}{d} I_d + \frac{k+1}{d}\Sigma\right)$. Of course, when M is finite, the covariance obtained will concentrate around its expectation; hence it will be close to the estimate just computed. This can be quantified precisely using matrix-valued tail bounds (Srivastava and Vershynin, 2013; Ahlswede and Winter, 2002).

3.2.2 Finitely Many Random Projections

Here we bound the deviation of the assembled covariance with finite M from its computed expectation. This is summarised in the following result.

THEOREM 1 (FINITE NUMBER OF RANDOM PROJECTIONS): *Let Σ be a positive semi-definite matrix of size $d \times d$ and rank ρ, and $R_i, i = 1, \ldots, M$ independent random projection matrices, each having entries drawn i.i.d. from $\mathcal{N}(0, 1/d)$, and denote by $\|\cdot\| = \lambda_{\max}(\cdot)$ the spectral norm of its argument. Then, $\forall \epsilon \in (0, 1)$,*

$$Pr\left\{\|\frac{1}{M}\sum_{i=1}^{M} R_i^T R_i \Sigma R_i^T R_i - E[R^T R \Sigma R^T R]\| \geq \epsilon \|E[R^T R \Sigma R^T R]\|\right\}$$

$$\leq 2d \exp\left\{-\epsilon^2 M^{\frac{1}{3}} \frac{\|E[R^T R \Sigma R^T R]\|}{4\tilde{K}}\right\} + 4M \exp\left\{-\frac{M^{\frac{1}{3}}}{2}\right\}, \tag{17}$$

where $\tilde{K} = \|\Sigma\| \left(\frac{1}{M^{1/6}}(1 + \sqrt{\frac{k}{d}}) + \frac{1}{\sqrt{d}}\right)^2 \left(\frac{1}{M^{1/6}}(\sqrt{\frac{\rho}{d}} + \sqrt{\frac{k}{d}}) + \frac{1}{\sqrt{d}}\right)^2$ is bounded w.r.t. M.

PROOF: We use the following Ahlswede-Winter-type result from random matrix theory about sums of independent random matrices.

THEOREM 2 (CONCENTRATION OF MATRIX SUMS) (Adapted from Ahlswede and Winter, 2002): *Let $X_i, i = 1, \ldots, M$ be $d \times d$ independent random positive semi-definite matrices satisfying $\|X_i\| \leq 1$ a.s. Let $S_M = \sum_{i=1}^{M} X_i$, and $\Omega = \sum_{i=1}^{M} \|E[X_i]\|$. Then, $\forall \epsilon \in (0, 1)$,*

$$Pr\left(\|S_M - E[S_M]\| \geq \epsilon\Omega\right) \leq 2d \exp\left(-\epsilon^2 \Omega/4\right). \tag{18}$$

The proof of Theorem 2 is given in the Appendix for completeness.

Observe, we do not have $\|R_i^T R_i \Sigma R_i^T R_i\|$ bounded a.s. when R_i have Gaussian entries, so we cannot apply this result directly. However, this condition can be satisfied by exploiting concentration, as follows.

First, we note that this random variable has the same distribution as $\|R_i^T R_i \Lambda R_i^T R_i\|$, where Λ is the diagonal matrix of eigenvalues of Σ. Here we used the rotation invariance of the Gaussian. Now, let ρ be the rank of Σ, and denote by $\underline{\Lambda}$ the $\rho \times \rho$ submatrix of Λ that contains the nonzero diagonals, and denote by $\underline{R_i}$ the corresponding $k \times \rho$ submatrix of R_i that are not wiped out by the zeros of Λ. Then we can write $\|R_i^T R_i \Lambda R_i^T R_i\| = \|R_i^T \underline{R_i \Lambda R_i^T} R_i\|$, and we can bound this with high probability (w.r.t. the random draws of $\overline{R_i}$):

$$\|R_i^T \underline{R_i \Lambda R_i^T} R_i\| \leq \|\Sigma\| \cdot \|R_i^T R_i\| \cdot \|\underline{R_i^T R_i}\|. \tag{19}$$

The following result bounds the largest singular value of a Gaussian matrix with i.i.d. entries:

LEMMA 3 (LARGEST SINGULAR VALUE OF GAUSSIAN MATRICES). (Rudelson and Vershynin, 2010, Eq. 2.3): *Let A be an $n \times N$ matrix, $n < N$, with standard normal entries, and denote by $s_{min}(A)$, $s_{max}(A)$ its least and greatest singular values. Then*

$$Pr\{s_{max}(A) \leq \sqrt{N} + \sqrt{n} + \epsilon\} \geq 1 - e^{-\epsilon^2/2}, \quad \forall \epsilon > 0. \tag{20}$$

We apply this to both R_i and $\underline{R_i}$. As these matrices have entries drawn i.i.d. from $\mathcal{N}(0, 1/d)$, we have $\forall \eta > 0$,

$$\|R_i^T \underline{R_i \Lambda R_i^T} R_i\| \leq \|\Sigma\| \cdot \left(1 + \sqrt{k/d} + \frac{\eta}{\sqrt{d}}\right)^2 \left(\sqrt{\rho/d} + \sqrt{k/d} + \frac{\eta}{\sqrt{d}}\right)^2 =: K(\eta)$$

with probability (w.p.) $1 - 2\exp(\eta^2/2)$.

Now, let $X_i(\eta) := R_i^T R_i \Sigma R_i^T R_i / K(\eta)$. Then we have

$$\|X_i(\eta)\| \leq 1 \text{ w.p. } 1 - 2\exp(-\eta^2/2). \tag{21}$$

Hence, by union bound, we have it uniformly for all $i = 1, \ldots, M$ that $\|X_i(\eta)\| \leq 1$ w.p. $1 - 2M\exp(-\eta^2/2)$. This holds for any choice of $\eta > 0$, and we will eventually choose η to override the M factor as well as to (approximately) tighten the final form of the deviation bound.

We now apply Theorem 2 conditionally on the event that $\|X_i(\eta)\| \leq 1, \forall i = 1, \ldots, M$, and use the bound on the probability that this condition fails. It is easy to see that in our case $\Omega(\eta) = \frac{M}{K(\eta)}\|\mathrm{E}[R^T R \Sigma R^T R]\|$, where $R \sim R_i$, and $\mathrm{E}[S_M(\eta)] = M \cdot \mathrm{E}[X_i(\eta)] = \frac{M}{K(\eta)}\mathrm{E}[R^T R \Sigma R^T R]$, and so we get

$$\begin{aligned}
&\Pr\left\{\|\frac{1}{K(\eta)}\sum_{i=1}^{M} R_i^T R_i \Sigma R_i^T R_i - \frac{M}{K(\eta)}\mathrm{E}[R^T R \Sigma R^T R]\| \geq \epsilon \frac{M}{K(\eta)}\|\mathrm{E}[R^T R \Sigma R^T R]\|\right\} \\
&= \Pr\left\{\|\frac{1}{M}\sum_{i=1}^{M} R_i^T R_i \Sigma R_i^T R_i - \mathrm{E}[R^T R \Sigma R^T R]\| \geq \epsilon \|\mathrm{E}[R^T R \Sigma R^T R]\|\right\} \\
&\leq 2d\exp\left\{-\epsilon^2 \frac{M}{4K(\eta)}\|\mathrm{E}[R^T R \Sigma R^T R]\|\right\} + 4M\exp\left\{-\frac{\eta^2}{2}\right\}.
\end{aligned}$$

Finally, we choose $\eta = M^{1/6}$ and denote $\tilde{K} := M^{2/3} K(M^{1/6})$, which yields the statement of Theorem 1. ■

This analysis shows that we can use a finite number of random subspaces, since we have control over the spectral distance between the resulting finite average of the d-dimensional rank-k covariances and the infinite limit of this sum. Hence, we may expect a similar behaviour from a finite ensemble, which is pleasing. The practical implication, as mentioned, is that an efficient parallel implementation can be realised where the estimation and sampling within each subspace is run on a separate core.

In closing, we should mention that although we used the truncation method here, a more direct route might exist if the a.s. boundedness condition could be relaxed in Theorem 2. In particular, we see from its proof (in the Appendix) that some suitable alternative to the Taylor expansion–based inequality in Eq. (24), along with the finiteness condition on the variances of the matrix summands, could possibly lead to a variation of Theorem 2 to eliminate the boundedness condition.

The finite sample analysis in Theorem 1 is too crude to give us the minimum order of M required for the matrix average to get close to its expectation with a given confidence. If we disregard the truncation step, then equating the right-hand side of Eq. (18) to some given $\delta \in (0, 1)$ and solving for M yields $M = \frac{4}{\epsilon^2}\frac{K(\eta)}{\|\mathrm{E}[R^T R \Sigma R^T R]\|}\log\frac{2d}{\delta}$, and noting that $\frac{K(\eta)}{\|\mathrm{E}[R^T R \Sigma R^T R]\|} = \frac{(\sqrt{d}+\sqrt{k}+\eta)^2(\sqrt{\rho}+\sqrt{k}+\eta)^2}{k^2+k(1+\mathrm{Tr}(\Sigma)/\|\Sigma\|)} = \mathcal{O}(d)$, we get $M \in \mathcal{O}(d\log(d))$. Otherwise a more careful choice for η, such as $\eta = M^{\tau/4}$, with τ a small positive number bounded away from zero, would yield nearly the same order for M (raised to a power just slightly larger than 1). However, the $\log(d)$ oversampling factor is due to the relatively general

conditions in the Ahlswede-Winter bound that we used, which holds for sums of generic positive semi-definite matrices and does not exploit the Gaussian (or sub-Gaussian) distribution of the entries of R. There are more refined analyses of analogous problems (Vershynin, 2012a) that we believe to be possible to adapt to our case to eliminate this log factor. Based on these considerations it is expected that $M = \mathcal{O}(d)$ is required.

3.3 Computational Complexity per Generation

Accounting for the cost of M different $k \times d$ matrix multiplications on N' points, our approach has time complexity (per generation) of $\mathcal{O}(M(k^3 + N'kd))$. As mentioned, the finite M implementation can be run in parallel on M separate cores, which is an advantageous structural property of our algorithm that could be exploited to gain further computational efficiency when needed. However, even on a single core, and taking $M = \mathcal{O}(d)$ as discussed, the complexity per generation becomes $\mathcal{O}(dk^3 + N'kd^2) \ll \mathcal{O}(d^3)$ when $k < N' \ll d$. This is indeed a regime where the proposed method yields good performance also (see Section 4). In contrast, other full covariance EDA-type methods such as EMNA, ED-EDA (Dong and Yao, 2008), and CMA-ES (Hansen, 2006) have a time complexity per generation of $\mathcal{O}(d^3)$.

In sum, the net effect of our RP ensemble approach is to get samples from the regularised covariance, without ever computing the maximum likelihood estimate, and without the need to explicitly sample from a $d \times d$ covariance, which would be an $\mathcal{O}(d^3)$ operation.

3.4 Alternative Random Projections Matrices

Our analysis in the previous sections focused on RP matrices with i.i.d. Gaussian entries. These have the advantage of being full row rank a.s., and they made our analysis in Section 3.2.1 more straightforward, but there are several alternatives available in the random projections literature that have a faster matrix multiplication time and still have full rank with very high probability. Of these we mention two.

The following sparse RP matrix proposed by Achlioptas (2003) is one of the earliest sparse constructions that is still in wide use. The entries R_{ij} are drawn i.i.d. as the following:

$$R_{ij} = \begin{cases} +\sqrt{3} & \text{with probability } 1/6, \\ -\sqrt{3} & \text{with probability } 1/6, \\ 0 & \text{with probability } 2/3, \end{cases}$$

and then normalised to have variance $1/d$. Using these, the matrix multiplications take roughly one third of the time that they do in the case of the Gaussian RP matrices.

Interesting to note, for this particular sparse RP, the limit at $M \to \infty$ happens to be exactly the same as that for the Gaussian RP (see Kabán, 2014, Lemma 2), and one can also obtain a similar concentration guarantee as that we gave for an ensemble of Gaussian RP matrices in Theorem 1.

Finally, the RP matrix with coin flip entries (Achlioptas, 2003) is also in use in the random projections literature for its computational efficiency. The entries R_{ij} are

$$R_{ij} = \begin{cases} +1 & \text{with probability } 1/2, \\ -1 & \text{with probability } 1/2, \end{cases}$$

and normalised to have variance $1/d$. So multiplication with this matrix is efficient, since it only involves bit flipping.

Again, these binary RP matrices are full rank with high probability. However, it may be verified using Lemma 2 of Kabán (2014) that for this binary RP ensemble the limit at $M \to \infty$ no longer coincides with that of the Gaussian and the sparse RP ensembles. Instead, for certain Σ, the regularisation effect is diminished, which when it happens can reduce the exploration ability of the search. Therefore we may expect the binary RP matrices to perform worse, and we use them mainly to see the robustness of our algorithm to such deviations from the analysis presented earlier.

4 Experiments

To test the potential of our idea and the ability of our algorithm to find a near-optimal solution in large-scale problem settings, we tested it on all multimodal functions of the suite of benchmark test functions from the CEC'10 competition on large-scale global optimisation (Tang et al., 2009). There are 12 multimodal functions in this test suite, which were created from shifted and group-rotated versions of three multimodal base functions, and in each case the search space is 1,000-dimensional. The reason we have chosen to focus on multimodal functions is that this is the category where our methodology is expected to provide most benefits.

The test bed was designed to contain a couple of separable problems and a suite of problems with a varying degree of nonseparability to give insights into the behaviour and performance of optimisation methods.

A test function is called separable if its global optimum can be found by optimising it in each of its arguments separately. Otherwise it is called nonseparable. A nonseparable function is called m-nonseparable if at most m of its arguments are not independent. Finally, a nonseparable function is called fully nonseparable if any two of its arguments are not independent.

The test functions in our study are listed here, and they are all minimisation problems having their global optimal fitness value equal to zero:

- Separable functions:
  - T1: Shifted Rastrigin's function
  - T2: Shifted Ackley's function
- Partially separable functions having a small number of variables that are dependent and all the remaining ones independent ($m = 50$):
  - T3: Single-group shifted and m-rotated Rastrigin's function
  - T4: Single-group shifted and m-rotated Ackley's function
  - T5: Single-group shifted and m-dimensional Rosenbrock's function
- Partially separable functions consisting of multiple independent subcomponents, each of which is m-nonseparable ($m = 50$). This category includes two subtypes, $d/(2m)$-group m-nonseparable and d/m-group m-nonseparable functions:
  - T5: $d/(2m)$-group shifted and m-rotated Rastrigin's function
  - T7: $d/(2m)$-group shifted and m-rotated Ackley's function
  - T8: $d/(2m)$-group shifted and m-dimensional Rosenbrock's function
  - T9: d/m-group shifted and m-rotated Rastrigin's function
  - T10: d/m-group shifted and m-rotated Ackley's function
  - T11: d/m-group shifted and m-dimensional Rosenbrock's function
- A fully nonseparable function:
  - T12: Rosenbrock's function

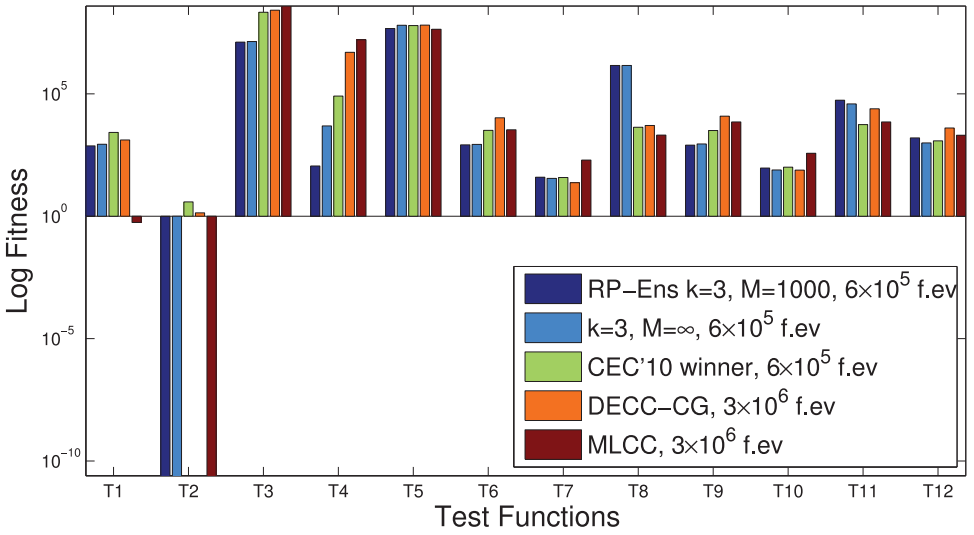

Figure 3: Comparison of our RP ensemble EDA algorithm with the CEC'10 large-scale optimisation competition winner (Molina et al., 2010) on 12 multimodal functions, after 6×10^5 function evaluations. Results of other state-of-the-art co-evolutionary-based methods, MLCC and DECC-CG, are also shown for reference; the last two are quoted from Molina et al. (2010) and use 3×10^6 function evaluations. All results represent averages of the best fitness from 25 independent repetitions.

See Tang et al. (2009) for more details on these functions.

4.1 Performance Results on the CEC'10 Large-Scale Optimisation Competition Benchmark

We use a simple averaging combination of RP-EDAs as in Algorithm 1. We allow a fixed budget of 6×10^5 function evaluations and use a population size of $N = 300$, and the number of retained individuals is set to $N' = 75$. We use truncation selection with elitism. We take the random subspace dimension to be $k = 3$, and the number of subspaces is set to $M = 1{,}000$. We also experimented with other parameter settings and observed that the results are qualitatively unchanged as long as we set k low enough to get reliable $k \times k$ covariance estimates from N' points and large enough to capture sufficient covariance structure. The number of random subspaces M must always be set above the minimum of $M_{\min} \geq \lceil d/k \rceil$ in order to cover the search space, and it is preferable to set it larger so that the finite average that appears in the analysis of covariance construction, Eq. (5), gets closer to the expectation and hence recovers the correct directions of the covariance. Note that a larger M does not incur extra function evaluations and increases the per generation time complexity only linearly. Of course, we do not claim optimality of these parameter settings, and indeed this may be problem-dependent in principle. Guidelines with more detailed discussion on how to set these parameters are given in Section 4.2.

Figure 3 gives a visual summary of the results obtained in comparison with the competition winner (Molina et al., 2010)—a fairly sophisticated memetic algorithm based on local search chains—and two other state-of-the-art co-evolutionary methods referenced on the competition's page, namely DECC-CG (Yang et al., 2008a) and MLCC

Table 1: Comparison on separable functions. The symbols in the last column indicate if the fitness value achieved by the method in that row is statistically significantly better (+) or worse (−) than each of our four RP ensemble EDA variants. These were determined using a 2-tailed t-test with 95% confidence level. The symbols at the four different positions in the last column are comparisons with the four variants of our method in the same order as listed in the first four rows for each function.

Func.	Method	max FE	Mean	Std	t-tests vs. g	∞	s	b
T1	RP-Ens (g) k=3 M=1000	6e+05	784.21	76.017	∅	+		
	RP-Ens k=3 M=∞	6e+05	868.96	49.715	−	∅	−	−
	RP-Ens (s) k=3 M=1000	6e+05	760.74	50.835		+	∅	+
	RP-Ens (b) k=3 M=1000	6e+05	818.36	40.7		+	−	∅
	CEC'10 winner	6e+05	2670	163	−	−	−	−
	DECC-CG	3e+06	1310	32.6	−	−	−	−
	MLCC	3e+06	0.557	2.21	+	+	+	+
	sep-CMA-ES	3e+06	5677.9	476.52	−	−	−	−
	EDA-MCC c=20 N=300	6e+05	1237	1237	−	−	−	−
T2	RP-Ens (g) k=3 M=1000	6e+05	2.5366e-13	4.8104e-15	∅		+	+
	RP-Ens k=3 M=∞	6e+05	2.5267e-13	3.743e-15		∅	+	+
	RP-Ens (s) k=3 M=1000	6e+05	0.00013352	2.367e-06	−	−	∅	
	RP-Ens (b) k=3 M=1000	6e+05	0.0001336	2.9135e-06	−	−		∅
	CEC'10 winner	6e+05	3.84	0.213	−	−	−	−
	DECC-CG	3e+06	1.39	0.0973	−	−	−	−
	MLCC	3e+06	9.88e-13	3.7e-12			+	+
	sep-CMA-ES	3e+06	21.062	0.038944	−	−	−	−
	EDA-MCC c=100 N=1500	6e+05	0.2025	0.012411	−	−	−	−

(Yang et al., 2008b). A statistical analysis and further comparisons are given in Section 4.1.1. The bar chart in Figure 3 shows the averages of the best fitness values (in log scale) from 25 independent runs. We also included results from 25 independent runs of the limiting version of our algorithm, that is, $k = 3$, $M = \infty$, which we implemented using the analytic expression computed in Eq. (16) (with sampling done in the full d-dimensional space). The reason we included this is to assess how our algorithm with a finite M deviates from it. For DECC-CG and MLCC we used the results produced with 3×10^6 function evaluations quoted from Molina et al. (2010); that is a considerably larger budget than we allowed for our method, as it is interesting to see that our results still compare well to these also.

Thus, we see that our simple RP ensemble EDA algorithm is highly competitive with the best state-of-the-art methods for large-scale optimisation and even slightly outperforms the CEC'10 competition winner on some of the functions on this difficult benchmark. Furthermore, it is worth noticing that the performance with $M = 1{,}000$ is nearly indistinguishable from that with infinite M.

4.1.1 Statistical Analysis and Further Comparisons with State-of-the-Art EDA-Type Methods

In Tables 1–4 we provide a detailed statistical analysis of the results shown in Figure 3, and in addition to the competition winner and the high-ranking co-evolutionary

Table 2: Comparison on single-group nonseparable functions.

Func.	Method	max FE	Mean	Std	*t*-tests vs. g	∞	s	b
T3	RP-Ens (g) k=3 M=1000	6e+05	1.2397e+07	3.1864e+06	∅			
	RP-Ens k=3 M=∞	6e+05	1.3202e+07	3.1221e+06		∅		
	RP-Ens (s) k=3 M=1000	6e+05	1.2675e+07	4.3228e+06			∅	
	RP-Ens (b) k=3 M=1000	6e+05	1.2886e+07	4.3566e+06				∅
	CEC'10 winner	6e+05	2.17e+08	8.56e+07	−	−	−	−
	DECC-CG	3e+06	2.63e+08	8.44e+07	−	−	−	−
	MLCC	3e+06	3.84e+08	6.93e+07	−	−	−	−
	sep-CMA-ES	3e+06	1.1867e+08	2.9231e+07	−	−	−	−
	EDA-MCC c=100 N=1500	6e+05	1.0161e+07	1.7962e+06	+	+	+	+
T4	RP-Ens (g) k=3 M=1000	6e+05	117.78	17.151	∅	+		−
	RP-Ens k=3 M=∞	6e+05	5049	754.24	−	∅	−	−
	RP-Ens (s) k=3 M=1000	6e+05	121.32	11.349		+	∅	−
	RP-Ens (b) k=3 M=1000	6e+05	82.249	2.1203	+	+	+	∅
	CEC'10 winner	6e+05	81400	2.84e+05				
	DECC-CG	3e+06	4.96e+06	8.02e+05	−	−	−	−
	MLCC	3e+06	1.62e+07	4.97e+06	−	−	−	−
	sep-CMA-ES	3e+06	6.5427e+06	3.762e+06	−	−	−	−
	EDA-MCC c=100 N=1500	6e+05	12.379	0.34627	+	+	+	+
T5	RP-Ens (g) k=3 M=1000	6e+05	1.3429e+08	2.5957e+08	∅			
	RP-Ens k=3 M=∞	6e+05	1.216e+08	1.986e+08		∅		
	RP-Ens (s) k=3 M=1000	6e+05	6.6642e+07	3.63e+07			∅	
	RP-Ens (b) k=3 M=1000	6e+05	7.8953e+07	3.8096e+07				∅
	CEC'10 winner	6e+05	6.13e+07	1.27e+08				
	DECC-CG	3e+06	6.44e+07	2.89e+07				
	MLCC	3e+06	4.38e+07	3.45e+07			+	+
	sep-CMA-ES	6e+05	7.8052e+06	1.6452e+06	+	+	+	+
	EDA-MCC c=100 N=1500	6e+05	1.9746e+11	4.2577e+11	−	−	−	−

See Table 1 caption for explanation of symbols.

methods we also present further comparisons with recent and state-of-the-art EDA-type methods: EDA-MCC (Dong et al., 2013), sep-CMA-ES (Ros and Hansen, 2008), and AMaLGaM-Univariate (Bosman, 2009) on function T12. For our method we included results obtained with the alternative random projection matrices (see Section 3.4).

We chose the particular EDA-type methods to compare by the following reasoning. EDA-MCC (Dong et al., 2013) was chosen because it is a recent method specifically developed to scale up EDA to high dimensions. It assumes that the covariance of the selected points has a block-diagonal structure, so depending on the block size it interpolates between UMDAc and EMNA. At first sight this seems quite similar to our approach, since the blocks define subspaces of the search space; however, the subspaces in EDA-MCC are axis-aligned and disjoint, whereas in our approach they are not. The implication of this difference is that when we decrease the subspace dimension, the covariance becomes more spherical while still avoiding the independence assumption of UMDAc (see our analysis in the earlier sections). This results in a better capability to escape unwanted early convergence. We used our own implementation of EDA-MCC, since there is no publicly available implementation. Since the guidelines on

Table 3: Comparison on the $d/(2m)$-group nonseparable functions.

Func.	Method	max FE	Mean	Std	*t*-tests vs. g	∞	s	b
T6	RP-Ens (g) k=3 M=1000	6e+05	832.18	62.543	∅	+		
	RP-Ens k=3 M=∞	6e+05	900.08	79.14	−	∅	−	−
	RP-Ens (s) k=3 M=1000	6e+05	804.09	79.982		+	∅	+
	RP-Ens (b) k=3 M=1000	6e+05	856.02	77.542		+	−	∅
	CEC'10 winner	6e+05	3220	185	−	−	−	−
	DECC-CG	3e+06	10600	295	−	−	−	−
	MLCC	3e+06	3430	872	−	−	−	−
	sep-CMA-ES	3e+06	6279.5	251.42	−	−	−	−
	EDA-MCC c=20 N=300	6e+05	1376.1	1376.1	−	−	−	−
T7	RP-Ens (g) k=3 M=1000	6e+05	41.664	8.7003	∅	−		
	RP-Ens k=3 M=∞	6e+05	30.093	7.9886	+	∅	+	+
	RP-Ens (s) k=3 M=1000	6e+05	40.36	10.973		−	∅	
	RP-Ens (b) k=3 M=1000	6e+05	43.469	8.6383		−		∅
	CEC'10 winner	6e+05	38.3	7.23		−		+
	DECC-CG	3e+06	23.4	1.78	+	+	+	+
	MLCC	3e+06	198	0.698	−	−	−	−
	sep-CMA-ES	3e+06	212.06	6.0235	−	−	−	−
	EDA-MCC c=100 N=1500	6e+05	14.658	0.45607	+	+	+	+
T8	RP-Ens (g) k=3 M=1000	6e+05	1.4442e+06	53945	∅			+
	RP-Ens k=3 M=∞	6e+05	1.4528e+06	69153		∅		
	RP-Ens (s) k=3 M=1000	6e+05	1.4281e+06	59947			∅	+
	RP-Ens (b) k=3 M=1000	6e+05	1.4842e+06	49239	−		−	∅
	CEC'10 winner	6e+05	4340	3210	+	+	+	+
	DECC-CG	3e+06	5120	3950	+	+	+	+
	MLCC	3e+06	2080	727	+	+	+	+
	sep-CMA-ES	6e+05	596.05	173.43	+	+	+	+
	EDA-MCC c=100 N=1500	6e+05	2.7149e+06	9.9543e+05	−	−	−	−

See Table 1 caption for explanation of symbols.

the parameter setting of EDA-MCC are not prescriptive and were only tested up to 500 dimensions (Dong et al., 2013), for our 1,000-dimensional benchmark set we ran experiments with two different sensible parameter settings and chose the best of the two results to report. This was to ensure that we did not inadvertently disadvantage this method. With block size of EDA-MCC denoted by c, the two versions we ran were $N = 300$, $c = 20$, and $N = 150$, $c = 100$. In both cases we set the budget of maximum function evaluations equal to our proposed RP-Ens-EDA, namely, 6×10^5.

The sep-CMA-ES method (Ros and Hansen, 2008) was included in our comparison because it is a variant of CMA-ES developed to handle high-dimensional problems, and it currently represents the gold standard for comparisons in new EDA research. We used the MATLAB implementation available from the authors[1] with the diagonal option, default parameter settings, and random initialisation. We ran sep-CMA to a maximum of function evaluations equal to that used for our RP-Ens-EDA in the first instance, namely, 6×10^5, but for functions on which it did not outperform our methods

[1]https://www.lri.fr/∼hansen/cmaes.m

Table 4: Comparison on the d/m-group nonseparable and fully nonseparable functions.

Func.	Method	max FE	Mean	Std	t-tests vs. g	∞	s	b
T9	RP-Ens (g) k=3 M=1000	6e+05	807.11	49.957	∅	+		+
	RP-Ens k=3 M=∞	6e+05	880.23	62.453	−	∅	−	
	RP-Ens (s) k=3 M=1000	6e+05	811.45	56.551		+	∅	+
	RP-Ens (b) k=3 M=1000	6e+05	852.96	59.161	−		−	∅
	CEC'10 winner	6e+05	3190	146	−	−	−	−
	DECC-CG	3e+06	12200	897	−	−	−	−
	MLCC	3e+06	7110	1340	−	−	−	−
	sep-CMA-ES	3e+06	6763.5	275.75	−	−	−	−
	EDA-MCC c=20 N=300	6e+05	1474.8	1474.8	−	−	−	−
T10	RP-Ens (g) k=3 M=1000	6e+05	90.481	17.178	∅	−		+
	RP-Ens k=3 M=∞	6e+05	72.997	20.589	+	∅	+	+
	RP-Ens (s) k=3 M=1000	6e+05	94.305	16.164		−	∅	
	RP-Ens (b) k=3 M=1000	6e+05	102.04	13.593	−	−		∅
	CEC'10 winner	6e+05	102	14.2	−	−		
	DECC-CG	3e+06	76.6	8.14	+		+	+
	MLCC	3e+06	376	47.1	−	−	−	−
	sep-CMA-ES	3e+06	413.83	10.971	−	−	−	−
	EDA-MCC c=100 N=1500	6e+05	5.0668	0.70999	+	+	+	+
T11	RP-Ens (g) k=3 M=1000	6e+05	54105	10877	∅	−		+
	RP-Ens k=3 M=∞	6e+05	40345	12018	+	∅	+	+
	RP-Ens (s) k=3 M=1000	6e+05	56828	14651		−	∅	+
	RP-Ens (b) k=3 M=1000	6e+05	1.3655e+05	52065	−	−	−	∅
	CEC'10 winner	6e+05	5530	3940	+	+	+	+
	DECC-CG	3e+06	24600	10500	+	+	+	+
	MLCC	3e+06	7090	4770	+	+	+	+
	sep-CMA-ES	6e+05	1447.1	309.65	+	+	+	+
	EDA-MCC c=100 N=1500	6e+05	4.2507e+07	5.3257e+06	−	−	−	−
T12	RP-Ens (g) k=3 M=1000	6e+05	1614.7	249.44	∅	−		+
	RP-Ens k=3 M=∞	6e+05	988.13	0.75027	+	∅	+	+
	RP-Ens (s) k=3 M=1000	6e+05	1626.2	237.59		−	∅	+
	RP-Ens (b) k=3 M=1000	6e+05	4552.7	3204	−	−	−	∅
	CEC'10 winner	6e+05	1210	142	+	−	+	+
	DECC-CG	3e+06	4060	366	−	−	−	
	MLCC	3e+06	2050	180	−	−	−	+
	sep-CMA-ES	6e+05	1046.5	51.644	+	−	+	+
	sep-CMA-ES	3e+06	903.63	39.149	+	+	+	+
	EDA-MCC c=100 N=1500	6e+05	6.1795e+07	7.7141e+06	−	−	−	−
	AMaLGaM-Univ	6e+05	8.2448e+08	2.5912e+08	−	−	−	−
	AMaLGaM-Univ	3e+06	990.73	15.174	+		+	+

See Table 1 caption for explanation of symbols.

we further ran it to a maximum of 3×10^6 function evaluations and reported that result instead. This was to avoid having the limited budget as the major obstacle for sep-CMA-ES.

Finally, the AMaLGaM-Univariate was chosen as a representative of AMaLGaM (Bosman, 2009; Bosman et al., 2013) because it was previously demonstrated to work up to 1,000-dimensional problems (Bosman, 2009), and among the 12 versions of the AMaLGaM package this version was found by the authors to work best in high-dimensional problem solving (Bosman, 2009), with results comparable to sep-CMA-ES. For this reason, and since the available software implementation of AMaLGaM that we used[2] contains a preset list of test functions of which one (Rosenbrock) is in common with our test suite (function T12), we included a comparison with AMaLGaM-Univariate on this function only. We tested several versions of AMaLGAM on this function, and AMaLGaM-Univariate was indeed the variant that produced the best results, which is in line with the authors' own finding.

Tables 1–4 give for each of the 12 test functions, and for each competing method, the mean and the standard deviation of the best fitness achieved, as computed from 25 independent repetitions. To determine the statistical significance of the differences in performance, we performed 2-tailed t-tests for each competing method against each of the four variants of our method. The symbols in the last column indicate whether the fitness achieved by a competing method is statistically significantly better (+) (that is, significantly lower, since we tackle minimisation problems) or worse (−) (i.e., higher) than that of a variant of RP ensemble EDA at the 95% confidence level. The symbol in the first position is a comparison with RP-Ens-EDA that uses $M = 1{,}000$ Gaussian RP matrices (g); the second symbol is a comparison with RP-Ens-EDA that uses infinitely many Gaussian or sparse RP matrices (∞) (their infinite ensemble limits coincide; see Kabán, 2014, Lemma 2), the third symbol is a comparison with RP-Ens-EDA that uses 1,000 sparse RP matrices (s); and the last symbol is a comparison with RP-Ens-EDA that uses 1,000 binary RP matrices (b). The symbol $\emptyset$ is placed where a comparison is not applicable (the method on a row and column coincide). The absence of any symbol in any of the four positions means that the associated comparison test detected no significant difference at the 95% confidence level.

One thing we notice from Tables 1–4 is that although some differences of statistical significance were detected in comparisons between some of the variants of our own method, these different variants behaved very similarly when looked at through comparisons with other methods. A second observation to be made is the great diversity in the performance behaviour among the competing methods versus ours; that is, on nearly every function (except T6 and T9, on which our proposed RP-Ens-EDA methods are the overall winners) there is at least one method that does better than ours and at least one that does worse than ours, but the methods that outperform RP-Ens-EDA on one function lose out on another function. This reflects a nice complementarity of the search biases of the methods, so although for the purpose of our comparison all these methods are treated as competitors, in reality in the toolbox of a practitioner they may be used to cooperate in solving difficult problems.

Let us first look at the details of the comparisons between pairs of our own methods. That is, we look at the first four rows for each test function and follow the markers in the last column (an antisymmetric matrix of markers). We notice the following. In the comparisons between the finite dimensional ensemble with Gaussian RPs versus its limit of infinite ensemble, we see that on only four out of twelve functions the infinite ensemble was significantly superior, and on another four functions the finite ensemble performed better. No statistically significant differences were detected on the remaining

[2]http://homepages.cwi.nl/∼bosman/source_code.php

Table 5: Aggregated summary of comparisons among our own methods. Number of test functions on which the methods in the rows win: lose against the methods in the columns. Only differences that are significant at the 95% confidence level are counted. All counts are out of the total number of test functions, namely, 12.

	No. Wins: No. Losses			
	RP-Ens (g)	RP-Ens (∞)	RP-Ens (sp)	RP-Ens (b)
RP-Ens (g)	N/A	4 : 4	1 : 0	6 : 1
RP-Ens (∞)	4 : 4	N/A	5 : 4	5 : 3
RP-Ens (sp)	0 : 1	4 : 5	N/A	6 : 1
RP-Ens (b)	1 : 6	3 : 5	1 : 6	N/A

four functions. So we can conclude that our practical algorithm using a finite ensemble of just 1,000 RPs is performing no worse that an infinite ensemble, and we also see the gap between them is small in practice.

In the comparisons between the Gaussian versus sparse RP ensembles, both taken with finite ensemble sizes, the 2-tailed t-test rejects the null of equal means only once out of 12 functions; in practical terms the two different RP ensembles appear equivalent. Indeed, apart from function T2, where in fact both algorithms got practically close to the global optimum, no statistically significant difference was detected between these two types of random projections. We have so far not identified any obvious reasons for the statistical difference observed on T2; it does appear consistent in all repeated runs. However, it is surprising to see the equivalent performance in all the 11 other functions, given that in stochastic search algorithms like ours there are many factors that may influence the dynamics. These results imply that taking advantage of the computational efficiency offered by the sparse RP matrices essentially comes for free.

The comparisons between the Gaussian and the binary finite ensemble turn out favourable for the former on six of the functions, whereas the binary ensemble wins only once. This was somewhat expected because of the reduced exploration capabilities of the binary RP ensemble (see Section 3.4). In answer to the question set out there, we should observe also that although these differences are statistically significant, the actual fitness values achieved are rather close on average for the two RP variants. By this we may conclude that our overall algorithm appears robust to deviations from the conditions of our analysis presented earlier, where we treated the case of Gaussian R.

It may also be interesting to comment on the comparisons between the finite binary RPs versus the infinite ensemble of (Gaussian or sparse) RPs. Despite the fact that the analytical form of the infinite ensemble covariance with the binary RPs is provably different from that with the Gaussian or the sparse RPs, for only five out of twelve functions is the latter superior with statistical significance, while the binary ensemble wins on one function. Four of those five functions coincide with those we observed in the case of the sparse RP ensemble.

Table 5 summarises the findings for the comparisons between the variants of our methods in terms of the overall number of test functions on which a variant wins / loses against another method. All counts are out of the 12 overall test functions, and only the differences that are significant with 95% confidence are counted. Based on these results we may then conclude that our new algorithmic framework appears to be quite

Table 6: Aggregated summary of comparison results from Tables 1, 3, and 4. Number of test functions on which our methods (rows) win: lose against other competing methods (columns). Only differences that are significant with 95% confidence are counted. All counts are out of the total number of test functions, namely, 12. For all our variants the number of wins obtained by our proposed methods is larger than or equal to the number of the losses, and this is so in each individual comparison.

	No. Wins: No. Losses				
	CEC10	DECC-CG	MLCC	sep-CMA-ES	EDA-MCC
RP-Ens (g)	6 : 3	7 : 4	7 : 3	8 : 4	8 : 4
RP-Ens (∞)	8 : 2	7 : 3	7 : 3	8 : 4	8 : 4
RP-Ens (sp)	5 : 3	7 : 4	7 : 5	8 : 4	8 : 4
RP-Ens (b)	5 : 4	6 : 4	6 : 6	8 : 4	8 : 4

robust and efficient, and is able to take advantage of speedups offered by alternative RP matrices.

We now look at the comparisons between the previously existing methods and the set of our own in the results given in Tables 1–4. Of the separable functions, MLCC is the only one that is better or not statistically worse than ours. Of the single-group nonseparable functions, EDA-MCC performed better than our proposed methods on T3 and T4 with all of the other competing methods being significantly worse than ours. However, on T5, the EDA-MCC turns out to be significantly worse, while sep-CMA-ES outperforms our RP-Ens-EDA instead. MLCC is also better or not significantly worse on this function.

In the $d/(2m)$-group m-nonseparable functions, our approach is the overall winner on T6, outperformed by DECC-GG and EDA-MCC on T7, and outperformed by four competitors on T8.

In the d/m-group nonseparable functions, the algorithm we proposed is the overall winner on T9, outperformed by EDA-MCC and partly outperformed by DECC-CG on T10, and outperformed by four competitors on T11. Finally, on the nonseparable function T12, sep-CMA-ES outperformed our algorithms, AMaLGaM-Univariate outperformed our finite-M variants only when given a larger budget of function evaluations, and the CEC'10 winner also marginally (but with statistical significance) outperformed our finite-M versions.

An aggregated summary of these findings in terms of the number of functions (from the total of 12) on which our method wins/loses is provided in Table 6 for each of the competing methods. Only the differences significant with 95% confidence are counted. We see that the version with binary RPs is the least effective, as expected, but still the number of wins obtained by our proposed methods is larger than or equal to the number of losses in each individual comparison.

4.1.2 Fitness Trajectories through the Generations

In order to gain more insight into the behaviour of our RP ensemble EDA algorithm it is useful to inspect its convergence behaviour by looking at the evolution of the best fitness across generations. This is plotted in Figures 4 and 5 comparatively for both the Gaussian RP and the sparse RP (Achlioptas, 2003) for four very different settings of the pair of parameters k and M: $k = 3$, $M = 1{,}000$; $k = 3$, $M = \infty$; $k = 3$, $M = \lceil d/k \rceil$;

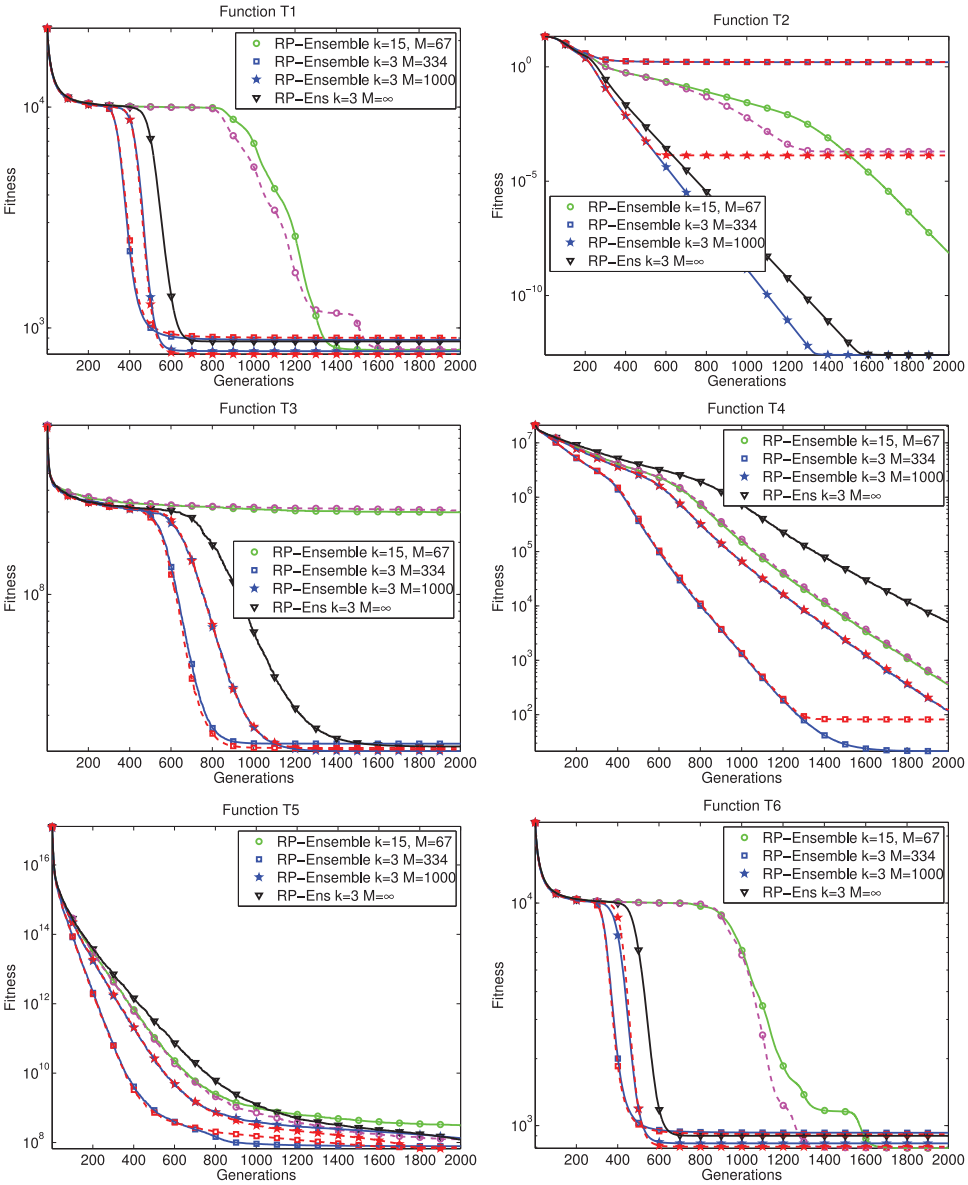

Figure 4: Convergence behaviour of our RP ensemble EDA methods on 1,000-dimensional multimodal test functions (functions T1–T6) for four different choices for the parameter pair k and N. The continuous lines are results with Gaussian RPs, and the dashed lines with the same markers are those with analogous versions that used sparse RP. In most cases the dashed lines are indistinguishable from the continuous ones.

$k = 15$, $M = \lceil d/k \rceil$. The last two use the minimum number of random subspaces that cover the full search space a.s. (in case of the Gaussian RP) or with very high probability (in case of the sparse RP). As before, we use population sizes of $N = 300$, $N' = 75$. All results represent averages of the best fitness as obtained from 25 independent runs.

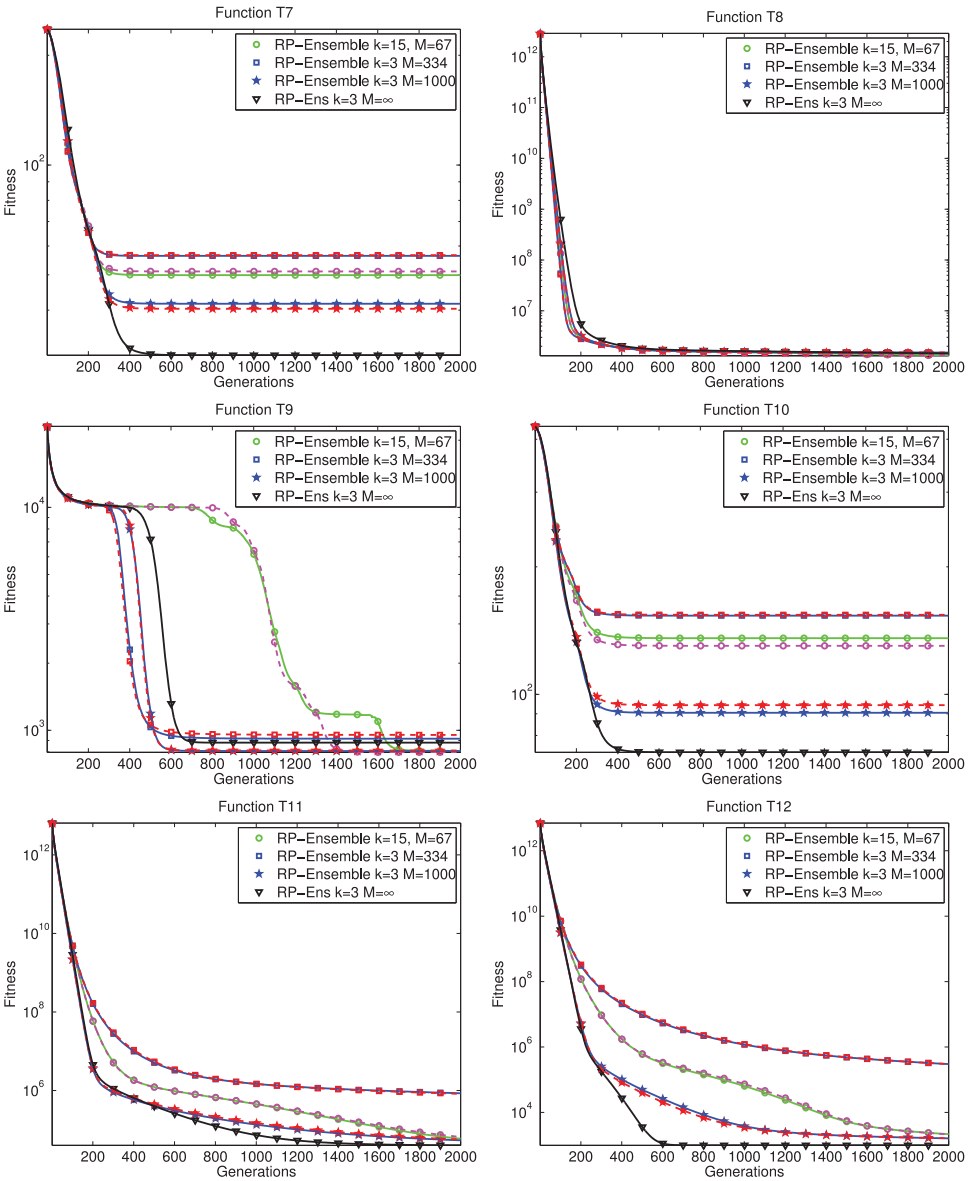

Figure 5: Convergence behaviour of our RP ensemble EDA methods on 1,000-dimensional multimodal test functions (functions T7–T12) for four different choices for the parameter pair k and N. The continuous lines are results with Gaussian RPs, and the dashed lines with the same markers are those with analogous versions that used the sparse RP. In most cases the dashed lines are indistinguishable from the continuous ones.

First, we see that the behaviour of the two different RP matrices is nearly indistinguishable even when the ensemble size is minimal. However, the minimal M that barely spans the search space tends to be a poor choice relative to a larger M, and since increasing M does not involve extra function evaluations, we recommend using a

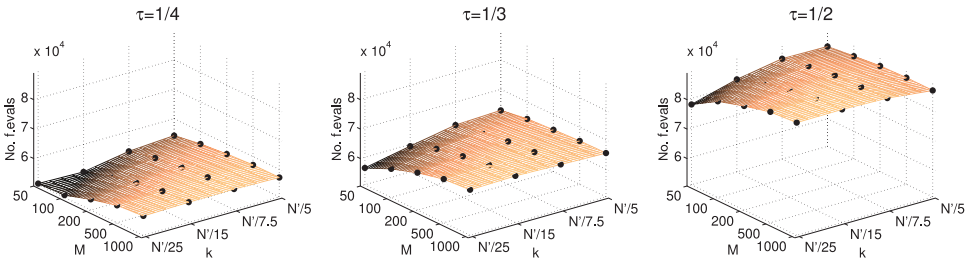

Figure 6: Number of fitness evaluations taken to reach the threshold of 10^{-10} on the sphere function.

larger value of M (of the order of d) in order to work in the regime where the ensemble covariance is understood by the analytical treatment we presented in previous sections. A more systematic empirical study of the effects of the parameter choices follows in the next section.

Furthermore, from Figures 4 and 5, we can see a clear tendency of our RP ensemble EDA algorithms to escape early convergence (which is known to be typical of both UMDAc and EMNA in high dimensions) and to explore the search space. It is particularly pleasing that the versions with finite number of random subspaces also perform well, and as expected, a larger value of M gets the performance closer to that of the idealised version with $M = \infty$.

4.2 Impact of the Parameters and Generic Guidelines for Setting the Parameters

Some observations already emerged regarding the setting of M and k, and the rule-of-thumb parameter settings we used in the large 1,000-dimensional experiments (see Section 4.1) turned out to perform well. Here we conduct a set of systematic experiments with parameter values varied on a grid and tested on four 100-dimensional functions. Two of these functions are among the few re-scalable ones of the same CEC'10 test suite that we used earlier—T2 (Ackley function, fully separable) and T12 (Rosenbrock, fully nonseparable)—and two others are toy problems that serve to demonstrate certain characteristics of the behaviour of the method: the sphere function ($f(x) = x^T x$) and the rotated ellipse ($f(x) = (Mx)^T \Lambda (Mx)$, $\Lambda = \mathrm{diag}_i(10^{\frac{i-1}{d-1}})$, where the rotation matrix M was uniformly randomly generated in each repeated run and then fixed).[3] Throughout this section we use a population size fixed to $N = 300$, and we focus to study the influence of k, M, and the selection pressure $\tau = N/N'$. We set the maximum function evaluations to 3 million—much larger than previously—in order to count the number of fitness evaluations required to reach various target values. For each combination of parameter values we performed 10 independent repetitions. We varied $\tau \in \{1/4, 1/3, 1/2\}$, $k \in \{N'/25, N'/15, N'/7.5, N'/5\}$ (making sure that we always had at least $5k$ points to estimate a $k \times k$ covariance) and $M \in \{50, 100, 500, 1{,}000\}$.

Figures 6–13 show the average of the function evaluations needed to reach two chosen target values for the sphere, Ackley, ellipse, and Rosenbrock functions, respectively.

[3]This differs from the literature standard, where rotation is typically fixed to 45 degrees to have a maximal departure from the coordinate axes. However, our proposed approach is by construction rotation-invariant; hence at this point the angle of rotation makes no difference.

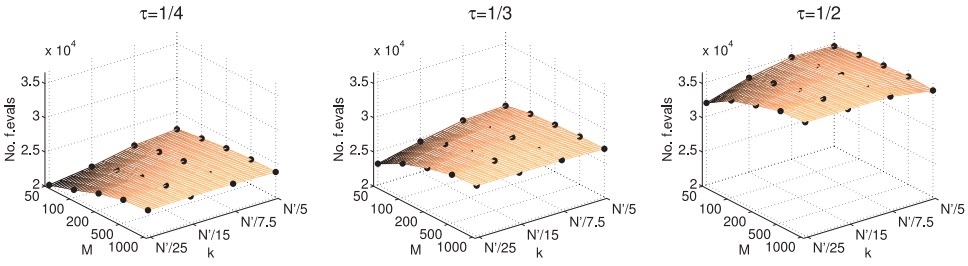

Figure 7: Number of fitness evaluations taken to reach the threshold of 10^{-1} on the sphere function.

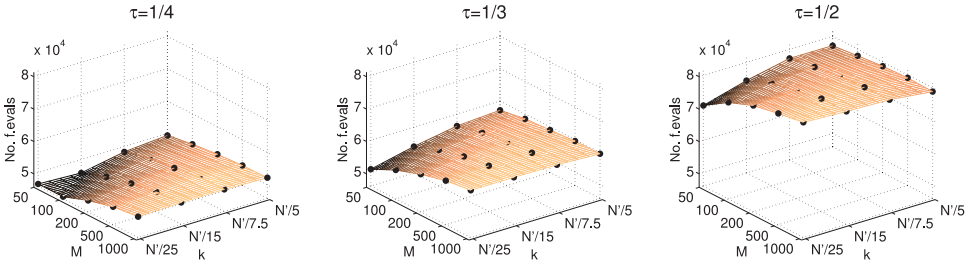

Figure 8: Number of fitness evaluations taken to reach the threshold of 10^{-5} on the Ackley function.

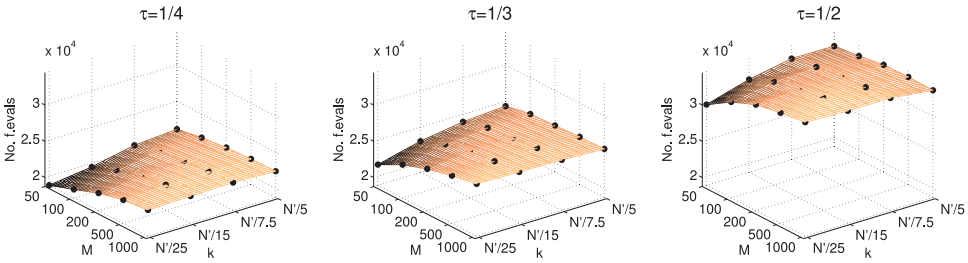

Figure 9: Number of fitness evaluations taken to reach the threshold of 10^{-1} on the Ackley function.

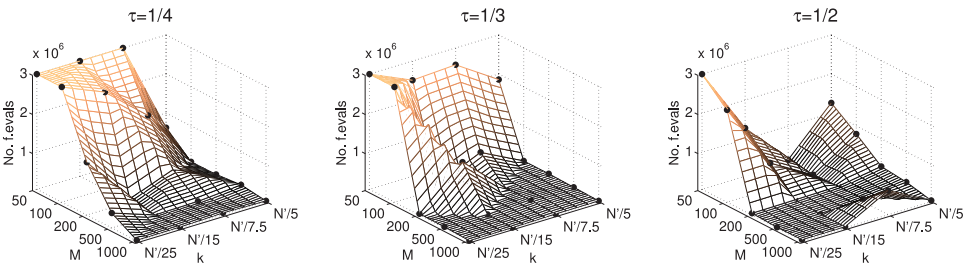

Figure 10: Number of fitness evaluations taken to reach the threshold of 10^{-1} on the ellipse function.

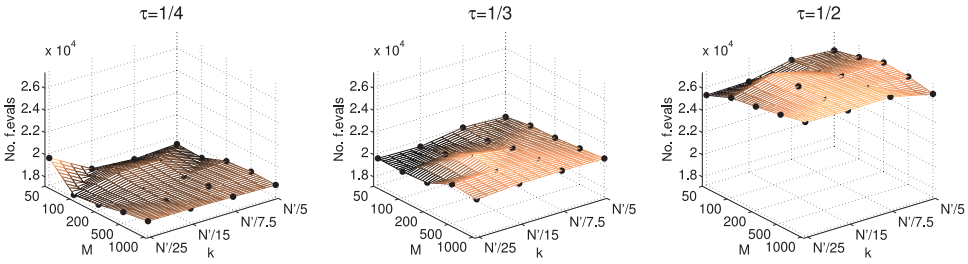

Figure 11: Number of fitness evaluations taken to reach the threshold of 10^2 on the ellipse function.

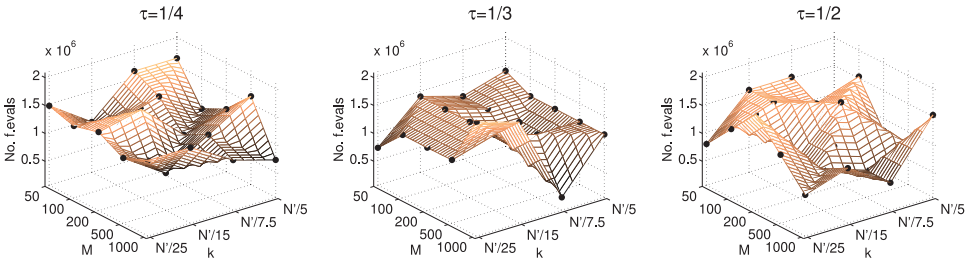

Figure 12: Number of fitness evaluations taken to reach the threshold of 10^2 on the Rosenbrock function.

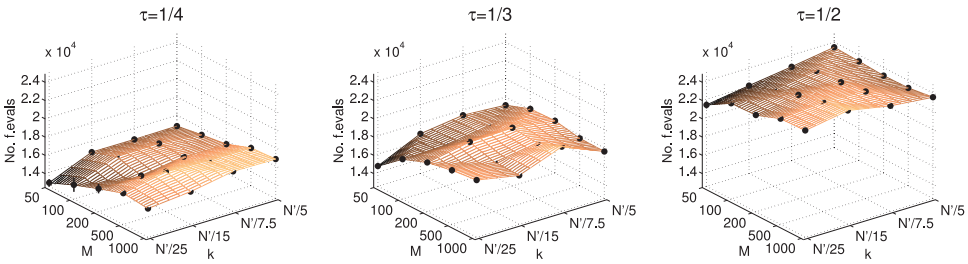

Figure 13: Number of fitness evaluations taken to reach the threshold of 10^4 on the Rosenbrock function.

If a target was not reached, then the count appears as 3×10^6 (i.e., the maximum number of function evaluations cutoff). There are also error bars on the black markers on these surface plots that represent one standard deviation computed from the 10 repeated runs; however, these are rather small and hardly visible at the overall scale. But the trend of the average search costs against the various parameter settings is clearly visible. Two target values (chosen from powers of 10) are displayed for each of these functions, of which one was chosen such that at least for half of the various combinations of values for k and M the specified target was successfully reached, and the second threshold value was a relatively larger one in order to see the influence of the parameter settings at two different stages of the optimisation. We will refer to this latter target value as the coarser target.

We found, rather unsurprisingly, that the optimal parameter setting depends on the problem in a complex way in the case of tight targets. However, this is not as much the case for the coarser target. The latter is of interest in the practical cases where a "quick and dirty" solution is sought, that is, when we seek an approximate solution to a difficult problem with limited resources. Considering that evolutionary heuristics are most often used in this latter role (He and Yao, 2003; Yu et al., 2012), it may be of interest to look at the search costs involved at both scales.

We now go through the results obtained. We see in Figures 6 and 7 that for the sphere function the surface plots have nearly identical shapes for both target values, while the search costs on the vertical axes differ. The unchanged behaviour is most likely because at both coarse and fine-grained scales the fitness landscape has the same spherical shape, which is easily modelled by our ensemble covariance with a low value of k. So the search strategy of our approach works equally well on both scales. Looking at the three surface plots that correspond to different selection pressures τ, one notices that the smaller value of $\tau = 1/4$ required lower search costs to reach the target in comparison with the larger value of $\tau = 1/2$. This observation is also consistently valid for the other functions. A larger value of τ means less selection pressure, which leads to slower convergence. Now, for each individual τ value, the associated surface plot indicates that a small k and (interestingly) a small M reaches the target quicker in the case of the sphere function. The small value of M works here, but we need to keep in mind that the analysis in Section 3.2.2 gives little guarantee for small values of M in terms of recovering the correct direction of the covariance, and indeed choosing M too small works poorly.

Next, the results on the Ackley function in Figures 8 and 9 display a striking similarity with those in the case of the sphere function. This is despite the fact that Ackley is a multimodal function with several local optima. The reason that our method has such similar behaviour on this function is most likely that the Ackley function has a spherical basin of attraction around its global optimum, and on a coarse scale the Ackley function resembles a spherical shape. So the close-to-spherical covariance induced in our approach when k is small turns out advantageous again in this case. Indeed, all the observations we have made about the results on the sphere function do carry over on Ackley. We did not reach a target of 10^{-10} as we did on the sphere function, but we did reach below 10^{-5}.

For the rotated ellipse function, the picture looks very different, as expected. Figures 10 and 11 show that small values of k and M all failed to reach the target of even 10^{-1}. For this fitness landscape we need a higher value of M and k to have the flexibility to model an elongated covariance, and to recover the correct direction of the covariance (which is only guaranteed for a large M). What is interesting to note is that the picture looks almost symmetric in k versus M (Figure 10), that is, a less elongated covariance (due to small k) can still work on this ellipse function provided that we recover its orientation (with larger enough M). Since the time complexity per generation is only linear in M but cubic in k, it seems a good heuristic to increase M first, and increase k only if the increased M did not deliver satisfactory performance. Of course, a very ill-conditioned fitness function would be a very difficult problem for our approach because we would need a large k close to d—in which case the averaging ensemble approach itself becomes no longer profitable. Possibly a weighted averaging combination might be developed to handle this case. It is important to remember the no-free-lunch theorem (Wolpert and Macready, 1997), which implies that no method is best on all problems, but each method works well on certain problems, namely, the problems that match the method's own

search bias. In our case, we have just seen this at work, where the Ackley function is an easy problem for our method, whereas the ellipse is a difficult problem. In Section 4.3 we find that for the sep-CMA-ES method the relative difficulty of these two functions is exactly the opposite of what was true for our RP ensemble EDA.

Let us now look at the coarser target value for the same rotated ellipse function. It is interesting to observe in Figure 11 that the surface plots of the search costs to reach the target value of 10^2 take the same shape as those in the case of the sphere and Ackley functions. This means that at a coarser scale our simple strategy still works with the same profitable parameter values as before. Hence apparently when an approximate solution is needed at low search costs, our method with the rule-of-thumb parameter settings is appropriate to use.

Finally, we show similar results on the Rosenbrock function in Figures 12 and 13. The search costs for the target value of 10^2 display rather complicated shapes, although the differences on the vertical axes are not particularly large. However, when we inspect the surface plots of the search costs for the coarse target of 10^4, we recognise the same shape that we have seen for all the other three functions. This reinforces the previous conclusion.

Based on these results we can set the following guidelines:

- A tight selection pressure, for instance, $\tau = 1/4$, worked best. This may be different from other approaches, such as EDA-MCC, which recommends $\tau = 1/2$, since in that approach a larger group size is important in order to avoid similarity with UMDAc. By contrast, the spherical component of our covariance favours exploration and has a better chance to avoid early convergence, which may be the reason that a higher selection pressure is more cost-efficient. However, these differences when varying τ have not been massive, and hence when the budget of function evaluations is not particularly tight, a larger τ may be justified, especially if we want to increase k.

- Regarding the setting of M, there is no substantial cost to setting it to a higher rather than a lower value. It does not incur any additional function evaluations, and it increases the per generation time complexity only linearly. A large enough M has the substantial benefit that we recover the orientation of the covariance from the ensemble, and in many cases this reduces the required function evaluations to reach tighter target values. Small values of M work well occasionally but can also lead to poor performance in some cases. We recommend setting M of the order of d.

- k is the parameter that controls the extent of regularisation. The smaller the value of k, the closer to spherical the ensemble covariance. A small k works very well on functions with a spherical basin of attraction around the optimum, and this also approximates other functions at a coarser scale. Therefore a small k can be effective when the goal is to get an approximate solution with limited resources. On the other hand, when the goal is to reach a target very close to the optimum and the fitness landscape is rather ill-conditioned, then k would need to be large. In that case we need to weigh the benefits against the much increased per generation computation time (cubic in k). The practitioner needs to weigh these trade-offs in making the appropriate choice for the problem at hand.

Finally, a comment is in order about the population size N. Since the estimation step is only required to estimate $k \times k$ covariances, N only needs to be large enough to have of the order k (e.g., a minimum of $5 \times k$) selected points in order to get sufficiently good covariance estimates in the k-dimensional space.

4.3 Scalability Experiments

Our final set of experiments measures the search costs (number of function evaluations) to reach a specified target value as the problem dimension varies, and compares these with the search costs of sep-CMA-ES.

We use the same four functions as previously, namely sphere, Ackley, rotated ellipse, and Rosenbrock. We fix the value to reach (VTR) to 10^{-5} and vary the dimensionality in $d \in [50, 1{,}000]$. We count the number of fitness evaluations needed for our proposed RP ensemble EDA to reach the VTR. We repeated the experiment for three other choices of VTR: 10^{-2}, 10^{2}, and 10^{3} in order to make sure that the conclusions would not be specific to a particular choice of VTR. In all these experiments we used the rule-of-thumb parameter settings based on the previous observations. We set the maximum fitness evaluations to 3×10^{6}, so the algorithm stops either upon reaching the VTR or when the maximum function evaluations are exhausted.

The results are displayed in the log-log plots in Figure 14. The figure shows the average number of function evaluations as computed from the successful runs out of 10 independent repetitions for each problem dimension. When none of the 10 repeated runs reached the prespecified VTR, there are missing data in these plots.

From Figure 14 we observe that a linear fit matches tightly the obtained scalability measurements on the log-log plots (dashed lines). The slopes of these lines signify the degree of the polynomial that describes the scaling of our algorithm. The dotted line that corresponds to linear scaling (slope = 1) is also superimposed on these plots for visual comparison.

From this figure we see that our proposed algorithm displays a near-linear dependence of the search costs (number of function evaluations to reach a VTR) as the problem dimension varies. More precisely, the scaling is sublinear in the majority of the cases tested (slope of the best-fitting line on the log-log plot is smaller than 1) in all experiments with sphere and Ackley as well as in the two larger VTR values for ellipse. In the remaining cases the scaling is slightly superlinear but still very close to linear (the slopes are 1.08, and 1.09 on the two smaller VTR values for ellipse, and 1.21 for Rosenbrock).

These results match the best scaling known for sep-CMA-ES (Ros and Hansen, 2008), and considering that the volume of the search space grows exponentially with the dimension; this is indeed as good as one can hope for. Note that a larger VTR can sometimes be reached with very few costs, for instance, this is what we see on the Ackley function. Hence we see that the proposed method is most profitable for quickly finding approximate solutions.

Figure 15 gives a detailed comparison with sep-CMA-ES, using a comparison protocol similar to that utilised by Bosman (2009). Our proposed RP-Ens-EDA and sep-CMA-ES present comparable scaling efficiency overall but have very different search biases and perform well in different situations. We compare the average numbers of function evaluations used by our method to reach various prespecified VTRs against those required by sep-CMA-ES to reach the same VTRs. We included a wider range of VTRs here, equally spaced in the interval $[10^{-10}, 10^{6}]$ for a better visibility of the behaviour

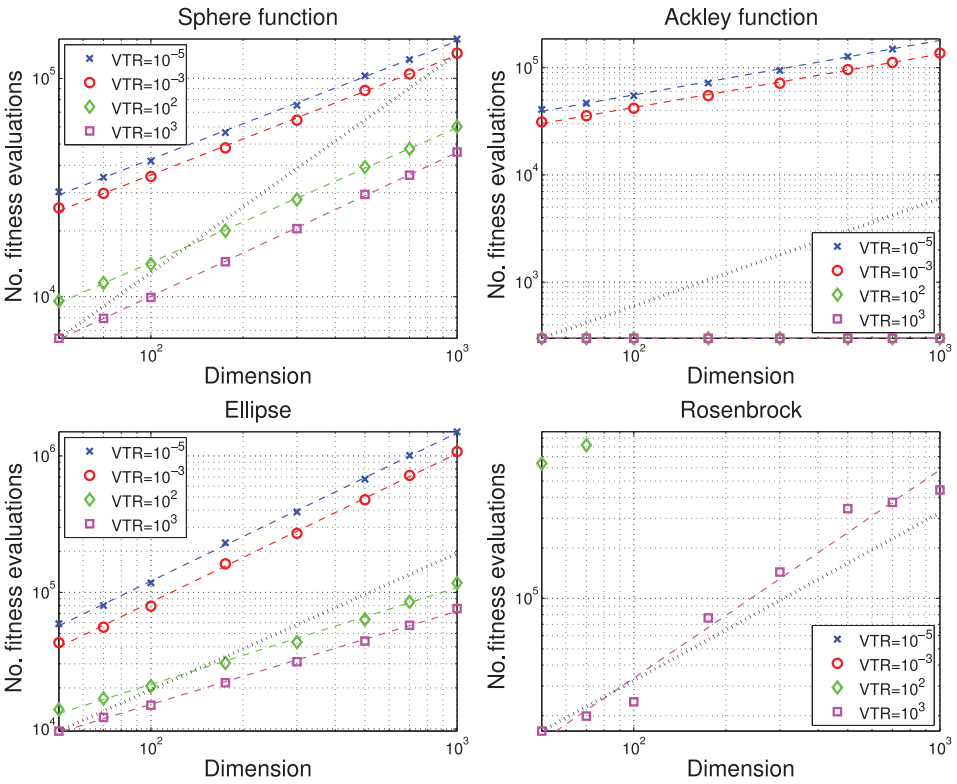

Figure 14: Scalability experiments. Number of function evaluations taken by successful runs of our RP ensemble to reach a prespecified value to reach (VTR) as the problem dimensionality is varied in $d \in [50, 1{,}000]$. The markers represent averages computed from 10 independent repetitions, the dashed lines show the best linear fit on these measurements on the log-log scale, and the dotted line corresponds to linear scaling (slope $= 1$). The parameter settings were $k = 3$, $N = 300$, $N' = N/4 = 75$, $M = 2 \times d$, with Gaussian RPs, and the maximum allowed function evaluations were set to 3×10^6.

of the two methods comparatively. We can summarise the following observations and conclusions from these results:

- Our method gains advantage in higher dimensions, whereas sep-CMA-ES scales better in lower dimensions. We see that in 1,000 dimensions RP-Ens-EDA consistently scales better on three out of the four functions tested (sphere, Ackley, and Rosenbrock) and partly on ellipse. In 500 dimensions our method scales no worse on two out of four functions (sphere and Ackley) and partly on ellipse, and it scales worse on one function (Rosenbrock). In 100 and 50 dimensions RP-Ens-EDA only scales better on one function (Ackley), whereas sep-CMA-ES scales better on the remaining three.

- Our method scales better than sep-CMA-ES for larger VTRs and loses from its efficiency in fine-grained search, whereas the efficiency of sep-CMA-ES is about the same at all scales. This is particularly visible on ellipse but also shows up on sphere.

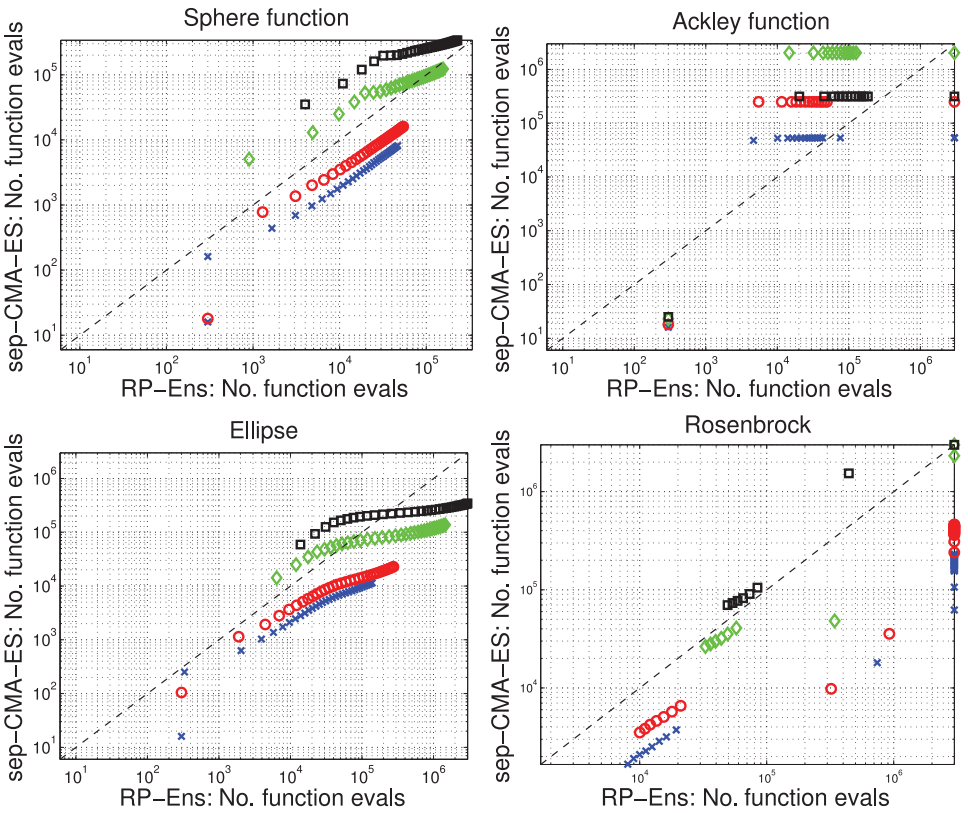

Figure 15: Average number of function evaluations of our RP ensemble (using the same parameter settings as in Figure 14) versus sep-CMA-ES (with its default parameters) for 33 target values (equally spaced on log10 scale) in the range $[10^{-10}, 10^{6}]$. The markers represent different dimensionalities: crosses, $d = 50$; circles, $d = 100$; diamonds, $d = 500$; squares, $d = 1{,}000$.

- The difficult/easy function types are different for these two methods. Clearly, Ackley is easier than ellipse for our method, whereas it is the other way around for sep-CMA-ES.

From these results we may conclude that our approach addresses the need to have simpler models for efficiently finding approximate solutions in high-dimensional problems. The model simplicity allows a more accurate means of estimation and more efficient sampling in order to be able to go to higher-dimensionality problem solving without discarding all the dependencies. Of course, there is no free lunch, and we are not able to solve all problems by our proposed method. However, our means to scalability is that we made estimating a high-dimensional general covariance matrix tractable with a small population. We achieved this by means of compression and averaging instead of inclusion/omission of individual dependencies, as other heuristics do. This difference turns out to induce a search bias that behaves quite differently from that of existing approaches, and it effectively allows us to find approximate solutions to high-dimensional complicated problems with a limited budget, whereas the finer-grained search remains in need of greater search costs.

5 Outlook and Future Work

We presented a new methodology for designing and developing EDA-type methods for large-scale optimisation. Our approach was to employ multiple random projections of the fit individuals and to carry out the estimation and sampling operations in low-dimensional spaces, where these are both efficient and reliable, rather than working in the original high-dimensional space. We carried out some theoretical analysis showing that the effect of our divide-and-conquer methodology can be reassembled and understood in the full high-dimensional search space. Finally, we presented empirical results using a simple instantiation of our proposed methodology, which demonstrated its effectiveness. On a battery of 12 multimodal test functions from the large-scale CEC'10 competition we obtained results that are competitive to the best state of the art. We believe these results may give a new perspective to research on EDA-type model-building optimisation algorithms, and future work is aimed at better understanding and exploiting its potential. In particular, the observed complementarity of the search biases induced in our approach and those of other state-of-the-art EDA-type methods suggests that combining their strengths would be a worthwhile avenue for further work.

References

Achlioptas, D. (2003). Database-friendly random projections: Johnson-Lindenstrauss with binary coins. *Journal of Computer and System Sciences*, 66(4): 671–687.

Ahlswede, R., and Winter, A. (2002). Strong converse for identication via quantum channels. *IEEE Transactions on Information Theory*, 48:568–579.

Apostol, T. (1957). *Mathematical analysis*. Reading, MA: Addison-Wesley.

Bosman, P. (2009). On empirical memory design, faster selection of Bayesian factorizations and parameter-free Gaussian EDAs. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 389–396.

Bosman, P., Grahl, J., and Thierens, D. (2013). Benchmarking parameter-free amalgam on functions with and without noise. *Evolutionary Computation*, 21:445–469.

Dasgupta, S. (1999). Learning mixtures of Gaussians. In *Proceedings of the Annual Symposium on Foundations of Computer Science*, pp. 634–644.

Diaconis, P., and Freedman, D. (1984). Asymptotics of graphical projection pursuit. *Annals of Statistics*, 12(3): 793–815.

Dong, W., Chen, T., Tino, P., and Yao, X. (2013). Scaling up estimation of distribution algorithms for continuous optimization. *IEEE Transactions on Evolutionary Computation*, 25(6): 797–822.

Dong, W., and Yao, X. (2008). Unified eigen analysis on multivariate Gaussian based estimation of distribution algorithms. *Information Sciences*, 178:3000–3023.

Durrant, R., and Kabán, A. (2015). Random projections as regularizers: Learning a linear discriminant ensemble from fewer observations than dimensions. *Machine Learning*, 99(2): 257–286.

Echegoyen, C., Zhang, Q., Mendiburu, A., Santana, R., and Lozano, J. (2011). On the limits of effectiveness in estimation of distribution algorithms. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1573–1580.

Hansen, N. (2006). The CMA evolution strategy: A comparing review. In J. Lozano, P. Larrañaga, I. Inza, and E. Bengoetxea (Eds.), *Towards a new evolutionary computation. Advances in estimation of distribution algorithms*, pp. 75–102. New York: Springer.

He, J., and Yao, X. (2003). An analysis of evolutionary algorithms for finding approximation solutions to hard optimisation problems. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 2004–2010.

Kabán, A. (2014). New bounds on compressive linear least squares regression. In *Proceedings of the International Conference on Artificial Intelligence and Statistics*, pp. 448–456.

Kabán, A., Bootkrajang, J., and Durrant, R. (2013). Towards large scale continuous EDA: A random matrix theory perspective. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 383–390.

Knight, J. N., and Lunacek, M. (2007). Reducing the space-time complexity of the CMA-ES. In *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO)*, pp. 658–665.

Larrañaga, P., and Lozano, J. (2002). *Estimation of distribution algorithms: A new tool for evolutionary computation*. New York: Springer.

Lorentz, G., von Golitschek, M., and Makovoz, Y. (1996). *Constructive approximation: Advanced problems*. New York: Springer.

Mahoney, M. (2011). Randomized algorithms for matrices and data. *Foundations and Trends in Machine Learning*, 3(2): 123–224.

Marzetta, T., Tucci, G., and Simon, S. (2011). A random matrix-theoretic approach to handling singular covariance estimates. *IEEE Transactions on Information Theory*, 57(9): 6256–6271.

Molina, D., Lozano, M., and Herrera, F. (2010). MA-SW-chains: Memetic algorithm based on local search chains for large scale continuous global optimization. In *Proceedings of the IEEE World Congress on Computational Intelligence*, pp. 3153–3160.

Mühlenbein, H., and Mahnig, T. (1999). Convergence theory and application of the factorized distribution algorithm. *Journal of Computing and Information Technology*, 7:19–32.

Omidvar, M. N., and Li, X. (2011). A comparative study of CMA-ES on large scale global optimisation. In *AI 2010: Advances in Artificial Intelligence*, pp. 303–312.

Ros, R., and Hansen, N. (2008). A simple modification in CMA-ES achieving linear time and space complexity. In *Proceedings of the International Conference on Parallel Problem Solving From Nature*, pp. 296–305.

Rudelson, M., and Vershynin, R. (2010). Non-asymptotic theory of random matrices: Extreme singular values. In *Proceedings of the International Congress of Mathematicians*.

Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Learning local feature descriptors using convex optimisation. *Pattern Analysis and Machine Intelligence*, 36:1573–1585.

Srivastava, N., and Vershynin, R. (2013). Covariance estimation for distributions with 2+epsilon moments. *Annals of Probability*, 41:3081–3111.

Sun, J., Garibaldi, J., and Hodgman, C. (2012). Parameter estimation using metaheuristics in systems biology: A comprehensive review. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 9:185–202.

Tang, K., Li, X., Suganthan, P., Yang, Z., and Weise, T. (2009). Benchmark functions for the CEC 2010 special session and competition on large-scale global optimization. http://nical.ustc.edu.cn/cec10ss.php Technical Report, Nature Inspired Computation and Application Laboratory, USTC, China.

Vempala, S. (2004). *The random projection method*. Providence, R.I.: American Mathematical Society.

Vershynin, R. (2011). A note on sums of independent random matrices after Ahlswede-Winter. http://www-personal.umich.edu/~romanv/teaching/reading-group/ahlswede-winter.pdf

Vershynin, R. (2012a). How close is the sample covariance matrix to the actual covariance matrix? *Journal of Theoretical Probability*, 25:655–686.

Vershynin, R. (2012b). Introduction to the non-asymptotic analysis of random matrices. In Y. Eldar and G. Kutyniok (Eds.), *Compressed sensing: Theory and applications*, pp. 210–268. New York: Cambridge University Press.

Wang, Y., and Li, B. (2008). A restart univariate estimation of distribution algorithm: Sampling under mixed Gaussian and Lévy probability distribution. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 3917–3924.

Wolpert, D., and Macready, W. (1997). No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation*, 1:67–82.

Yang, Z., Tang, K., and Yao, X. (2008a). Large scale evolutionary optimization using cooperative coevolution. *Information Sciences*, 178:2985–2999.

Yang, Z., Tang, K., and Yao, X. (2008b). Multilevel cooperative coevolution for large scale optimization. In *Proceedings of the IEEE Congress on Evolutionary Computation*, pp. 1663–1670.

Yu, Y., Yao, X., and Zhou, Z.-H. (2012). On the approximation ability of evolutionary optimization with application to minimum set cover. *Artificial Intelligence*, 180–181:20–33.

Appendix

Ahlswede-Winter–type bounds (Ahlswede and Winter, 2002) are generalisations of Chernoff bounds to matrix-valued random variables. These bounds deal with random matrices whose entries are not independent, and obtain concentration results for the sum of multiple independent copies of such matrices. Just as for Chernoff bounds, there are several versions in use, and we give some details of the proof for the version that we employed (see Theorem 2). The reader is also referred to the unpublished notes of Vershynin (2011).

DEFINITION: *A symmetric matrix A is called positive semi-definite (p.s.d.) if all its eigenvalues are non-negative. The notation $\succcurlyeq$ stands for the p.s.d ordering, that is, $A \succcurlyeq B$ means that $A-B$ is p.s.d. The spectral norm of a symmetric matrix A is defined as $\|A\| = max_i\, |\lambda_i(A)|$, where $\lambda_i(A)$ is the ith eigenvalue of A.*

PROOF OF THEOREM 2: The main ingredient of the proof is the Ahlswede-Winter inequality, stated here without proof, which can be found in Ahlswede and Winter (2002, Appendix, Theorem 18). It employs the Golden-Thompson inequality from matrix algebra.

THEOREM 3 (AHLSWEDE-WINTER INEQUALITY): *Let $X_i, i = 1, \ldots, M$ be $d \times d$ independent random symmetric matrices, and let $S_M = \sum_{i=1}^{M} X_i$. Then $\forall \xi > 0, \forall t > 0$,*

$$Pr\{\|S_M\| \geq t\} \leq 2d \cdot exp(-\xi t) \prod_{i=1}^{M} \|E[\, exp(\xi X_i)]\|. \tag{22}$$

Define $Z_i = X_i - \mathrm{E}[X_i]$, and apply the Ahlswede-Winter inequality to $Z_i, i = 1, \ldots, M$. We have, $\forall t > 0$,

$$\Pr\{\|S_M - \mathrm{E}[S_M]\| \geq t\} \leq 2d \cdot \exp(-\xi t) \prod_{i=1}^{M} \|\mathrm{E}[\exp(\xi Z_i)]\|. \tag{23}$$

To bound the matrix norm in the right-hand side, note that for any $\xi \in [0, 1]$,

$$\exp(\xi Z_i) \preccurlyeq I + \xi Z_i + \xi^2 Z_i^2. \tag{24}$$

This holds because $e^y \leq 1 + \xi y + \xi^2 y^2$ holds $\forall y \in [-1, 1]$, and all the eigenvalues of Z_i are in $[-1, 1]$. The latter can be seen by noting that $\mathrm{E}[X_i] \succcurlyeq 0$, so $Z_i = X_i - \mathrm{E}[X_i] \preccurlyeq X_i$, hence $\|Z_i\| \leq \|X_i\| \leq 1$. Now, taking expectation on both sides of Eq. (24) and noting that $\mathrm{E}[Z_i] = 0$ gives

$$\mathrm{E}[\exp(\xi Z_i)] \preccurlyeq I + \xi^2 \mathrm{E}[Z_i^2] \preccurlyeq \exp(\xi^2 \mathrm{E}[Z_i^2]), \tag{25}$$

where the last inequality holds because $1 + y \leq e^y, \forall y \in \mathbb{R}$. From Eq. (25) it follows that

$$\|\mathrm{E}[\exp(\xi Z_i)]\| \leq \|\exp(\xi^2 \mathrm{E}[Z_i^2])\| = \exp(\xi^2 \|\mathrm{E}[Z_i^2]\|). \tag{26}$$

We estimate the variance $\mathrm{E}[Z_i^2]$:

$$\mathrm{E}[Z_i^2] = \mathrm{E}[(X_i - \mathrm{E}[X_i])^2] = \mathrm{E}[X_i^2] - (\mathrm{E}[X_i])^2 \tag{27}$$

$$\preccurlyeq \mathrm{E}[X_i^2] \tag{28}$$

$$\preccurlyeq \mathrm{E}[\|X_i\| \cdot X_i] \tag{29}$$

$$\preccurlyeq \mathrm{E}[X_i], \tag{30}$$

where Eq. (28) follows because $\mathrm{E}[X_i]^2 \succcurlyeq 0$, and for Eq. (30) we used that $\|X_i\| \leq 1$. Hence,

$$\|\mathrm{E}[Z_i^2]\| \leq \|\mathrm{E}[X_i]\|. \tag{31}$$

Using this and Eq. (26), the matrix norm we need for the right-hand side of Eq. (23) is bounded as

$$\|\mathrm{E}[\exp(\xi Z_i)]\| \leq \exp(\xi^2 \|\mathrm{E}[X_i]\|). \tag{32}$$

Plugging this into the Ahlswede-Winter inequality, we get

$$\Pr\{\|S_M - \mathrm{E}[S_M]\| \geq t\} \leq 2d \cdot \exp(-\xi t) \prod_{i=1}^{M} \exp(\xi^2 \|\mathrm{E}[X_i]\|) \tag{33}$$

$$= 2d \cdot \exp(-\xi t + \xi^2 \sum_{i=1}^{M} \|\mathrm{E}[X_i]\|) \tag{34}$$

$$= 2d \cdot \exp(-\xi t + \xi^2 \Omega). \tag{35}$$

Since this holds for any $\xi \in [0, 1]$, we minimise the right-hand side to tighten the bound. Minimisation yields $\xi = t/(2\Omega)$, and this value needs to be in $[0, 1]$ in order to have

$$\Pr\{\|S_M - \mathrm{E}[S_M]\| \geq t\} \leq 2d \cdot \exp\left(-\frac{t^2}{4\Omega}\right). \tag{36}$$

Putting $t := \epsilon\Omega$ corresponds to $\xi = \epsilon/2$, which is indeed in $[0, 1]$ as required, and yields the form stated in Theorem 2. ■