Reduced Variable Neighbourhood Search for the Generation of Controlled Circular Data

Sergio Consoli¹(✉) , Domenico Perrotta¹ , and Marco Turchi²

¹ European Commission, Joint Research Centre, Via E. Fermi 2749,
21027 Ispra, VA, Italy

{sergio.consoli,domenico.perrotta}@ec.europa.eu

² Fondazione Bruno Kessler-IRST, Via Sommarive 18, 38123 Povo, Trento, Italy
turchi@fbk.eu

Abstract. A number of artificial intelligence and machine learning problems need to be formulated within a directional space, where classical Euclidean geometry does not apply or needs to be readjusted into the circle. This is typical, for example, in computational linguistics and natural language processing, where language models based on Bag-of-Words, Vector Space, or Word Embedding, are largely used for tasks like document classification, information retrieval and recommendation systems, among others. In these contexts, for assessing document clustering and outliers detection applications, it is often necessary to generate data with directional properties and units that follow some model assumptions and possibly form close groups. In the following we propose a Reduced Variable Neighbourhood Search heuristic which is used to generate high-dimensional data controlled by the desired properties aimed at representing several real-world contexts. The whole problem is formulated as a non-linear continuous optimization problem, and it is shown that the proposed Reduced Variable Neighbourhood Search is able to generate high-dimensional solutions to the problem in short computational time. A comparison with the state-of-the-art local search routine used to address this problem shows the greater efficiency of the approach presented here.

Keywords: Optimization · Reduced Variable Neighbourhood Search · Synthetic directional data generation · Circular statistics

1 Introduction and Background

The analysis and interpretation of directional data requires specific data representations, descriptions and distributions. Directional data occurs in many application areas like, e.g. earth sciences, astronomy, meteorology and medicine. Note that directional data is an “interval type” data: points are typically considered on the unit radius circle, or sphere, and the focus is on the direction of the

data vectors. The position of the “zero degrees” is then arbitrary, and the angle between two unit length vectors is used as a natural measure for their distance. Although the angular distances can be used sometimes also with general data vectors defined in the Euclidean space, usual statistics, such as the arithmetic mean and the standard deviation, are not appropriate because they do not have this rotational invariance property [34], and one must rely on proper directional (or circular) statistics [26]. In this context classical Euclidean geometry laws do not apply or need to be properly reformulated into the circle.

Nowadays also a number of popular artificial intelligence and machine learning problems are modelled using directional statistics. This is typical, for example, in computational linguistics and natural language processing, where language models based on Bag-of-Words [42], Vector Space [35], or Word Embedding [25], are largely used for tasks like document classification, information retrieval and recommendation systems, among others. In these models, documents are represented by feature vectors. Specifically, in Bag-of-Words, a text (such as a sentence or a document) is represented as the bag (multiset) of its words, disregarding grammar and even word order but keeping multiplicity. A Vector Space model is a special Bag-of-Words representing a document by a feature vector, where each feature is a word (term) and the feature’s value is a term weight. The term weight might be, either, a binary value (with 1 indicating that the term occurred in the document, and 0 indicating that it did not); a term frequency value (indicating how many times the term occurred in the document); or a TF-IDF (term frequency-inverse document frequency) value [1], a numerical statistic that is intended to reflect how important a word is to a document into a collection or corpus. The entire document is thus a feature vector, and each feature vector corresponds to a point in a vector space. The model for this vector space is such that there is an axis for every term in the vocabulary, and so the vector space is v -dimensional, where v is the size of the vocabulary. For long documents, where the vocabulary can be excessively large to handle, Word Embedding models can be a preferred option [25]. In Word Embedding model, like e.g. the popular Word2Vec [27], GloVe [31], or the most recent BERT [18], words or phrases from the vocabulary are mapped to lower-dimensions vectors of real numbers. Conceptually it involves a mathematical embedding from a space with many dimensions per word to a continuous vector space with a much lower dimension [25]. Although these representation models have been largely used in computational linguistics and natural language processing, they have found also applications for generic visual categorization in computer vision [6, 16, 40], among others.

Typically, reasons for preferring a directional approach in these contexts are driven by the problem. For example, the angular distance is known to favor vector components with higher variance (see e.g. [33]), which in linguistics is desirable for giving more weight to the more informative terms. Other reasons are of computational nature. When the space dimension v and sample size n may be both very large, the computation of the ordinary distances may become time consuming. In addition, the high dimensional vectors are often very sparse and an

angular distance, namely the cosine similarity, would neglect the 0-components and save time compared to an ordinary Euclidean distance. Besides, v is often considerably larger than n , leading to singular covariance matrices and complicating the use of the Mahalanobis distance (see e.g. [20], [15] and [19]), which otherwise is in general a good choice for taking into account data correlations.

In cluster analysis, as in many statistical and data science problems, the distance measure plays a crucial role [8, 10]. Points are assigned to groups according to their distance from the closer group centroid, and points that cannot be firmly assigned to any group are considered outliers [13]. A clustering method, e.g. K-means [36], may produce equivalent results with different distance measures (and centroid definitions) but behaves very differently with others.

To study the relationship between distance measures and objectively compare the performance of clustering, outliers detection, or other data analysis frameworks, it is often necessary to conduct systematic tests on a big number of different data sets artificially generated with known properties. In fact, it is well known that there is always some bias in the use of standard collections of data from real life applications¹ which should be rather used to confirm the properties of methods previously studied under controlled settings [12].

As an example, [38] have extended the Forward Search [2] method to the automatic detection of outliers in human label documents set. These outliers are documents that are wrongly assigned to a category or weakly correlated to other documents, and that need to be removed before training a learning system on these data. The synthetic datasets produced with our method allow to calibrate the outliers detection strategy and to assess the quality of the information retrieval process [38].

While it is rather easy to generate synthetic multivariate data according to some probability distribution or covariance structure in the Euclidean space, efficient methods that at the same time impose constraints on the distribution of the pairwise angular distances of the generated data vectors have been less explored in the literature. In [14] we have proposed a simple local search approach for addressing the problem of high-dimensional directional data generation controlled by specific properties. In particular, motivated by problems in computational linguistics, it is hypothesized that data are power-law distributed [19, 34], and that their pairwise cosine similarities are distributed as a von Mises distribution [41], i.e. the analogous of the normal distribution in circular statistics [26], around a desired cosine value. This represents a common realistic setting in several computational linguistics applications [22, 32].

In this contribution we propose an efficient optimization approach based on a Reduced Variable Neighbourhood Search (RVNS) heuristic [9, 21, 30] which is characterized by a very high speed and some interesting implementation improvements with respect to the local search approach in the literature [14]. This algorithm is characterized by an ease of implementation and simplicity, and it takes

¹ see for example the UC Irvine Machine Learning Repository (UCI): <https://archive.ics.uci.edu/ml/index.php>, a popular open-source repository of annotated datasets for machine learning tasks.

inspiration from the recently proposed “less is more approach” [10,29], which supports the adoption of non-sophisticated and effective metaheuristics instead of hard-to-reproduce and complex solution approaches. We will show how the proposed RVNS heuristic is able to generate efficiently high-dimensional directional data controlled by the desired properties. The generation algorithm is computationally efficient and flexible as well in terms of adaptability to other distributions and problems.

The rest of the paper is organized as follows. Section 2 formulated the whole problem as a non-linear continuous optimization problem. Section 3 describes the implementation details of the considered optimization algorithms, namely the state-of-the-art local search routine in the literature and our proposed RVNS approach. Our computational analysis is reported in Sect. 4, where it is shown that the proposed RVNS is able to generate high-dimensional solutions to the problem in shorter computational time with respect to the local search routine, demonstrating the efficiency of the approach presented here. Finally the paper ends with conclusions and suggestions for possible future research in Sect. 5.

2 Problem Formulation

The problem has been firstly formulated by [14] distinguishing between two general settings. The first is the ideal case of a single homogeneous group of n documents represented in a vector space of size v (vocabulary size). Here, high recall and precision are achieved by minimizing the sum of the pairwise similarities between documents,

$$F = \sum_{i=1}^n \sum_{j=1, j \neq i}^n cs(Z_i, Z_j) \quad (1)$$

where cs is the cosine similarity function that for two vectors Z_i and Z_j is:

$$cs(Z_i, Z_j) = \frac{Z_i \cdot Z_j}{\|Z_i\|_2 \cdot \|Z_j\|_2} = \frac{\sum_{k=1}^v z_{(i, k)} \cdot z_{(j, k)}}{\sqrt{\sum_{k=1}^v z_{(i, k)}^2} \cdot \sqrt{\sum_{k=1}^v z_{(j, k)}^2}}. \quad (2)$$

Clearly, $0 \leq cs(Z_i, Z_j) \leq 1$. In the second setting we assume $K > 1$ homogeneous groups and a number of isolated outliers. Now, the quantity F computed within a given homogeneous group should be maximized, while it should be minimized when computed on the set of the K estimated centroids². The “density” of the documents within an homogeneous group is determined by the average value of the quantity F for that group. Therefore, to generate artificial data suitable for benchmarks in the two general settings we can use the following constrained non-linear continuous optimization problem formulation [24].

Given a cosine similarity value $\tilde{cs} \in (0, 1)$, a tolerance $\xi \in (0, 1)$, $\xi \ll \tilde{cs}$, find a set $\mathcal{Z}$ of n v -dimensional vectors with $v \gg n$ non-negative variables, such

² It can be shown that the mean vector estimated within a group is a centroid with good properties (see e.g. [37], Section 8.2.6).

that the set of $n(n-1)/2$ pairwise cosine similarity values $\mathcal{C} = \{cs(Z_i, Z_j) ; j = 2, \dots, n ; 1 < i < j\}$ satisfies:

1. $cs(Z_i, Z_j) \in [\tilde{cs} - \xi, \tilde{cs} + \xi]$,
2. $\mathcal{C}$ is a random sample drawn from a von Mises distribution

$$f(x|\mu, \kappa) = \frac{\exp^{\kappa(\cos(x-\mu))}}{2\pi I_0(\kappa)} \quad 0 < x \leq 2\pi$$

with mean $\mu(\mathcal{C}) \approx \tilde{cs}$ and concentration parameter $\kappa(\mathcal{C}, \xi)$, being $I_0(\cdot)$ the modified Bessel function of order 0.

We use the Watson goodness-of-fit test [39] to assess whether $\mathcal{C}$ is consistent with the hypothesized von Mises null distribution with known mean ($\tilde{cs}$) and concentration parameter estimated from $\mathcal{C}$ itself.

In the von Mises, the concentration parameter κ is the analogous of the reciprocal of the variance for the normal distribution and links the sample standard deviation $\sigma(\mathcal{C})$ with the chosen tolerance ξ . This is done by using the well-known empiric *three-sigma rule* [34] and monitoring the constraint $\sigma(\mathcal{C}) \approx \xi/(10/3)$ during the data generation process, which corresponds to a concentration parameter $\kappa(\mathcal{C}, \xi) \approx 9$, ensuring that the data are representative of more than 99.9% of the generating von Mises distribution.

3 Optimization Methods

This section contains the implementation details of the considered optimization algorithms, namely the state-of-the-art local search routine implemented in [14] and the RVNS approach that we propose here. For a survey on the basic concepts of approximate optimization methods, including stochastic local search and metaheuristics, the reader is referred to [3, 4, 23]. The algorithms have been implemented using the commercial software package MATLAB, R2019b 64-bit, version 9.7.0³.

3.1 Local Search Algorithm

The local search algorithm starts from an initial solution, $\mathcal{X}^0$, of n data vectors with v variables, constructed as described in [14] such that all the vectors have exactly the same pairwise cosine similarity value $\tilde{cs}$ [11]. Consider the set $\mathcal{X}$ of n data vectors with $v \gg n$ variables, where all the vector tails are equal from variable index $n+1$ onwards, i.e. $z_{(i, k)} = z_{(j, k)} \forall k \geq n+1$ and $\forall i, j \leq n$, with $i \neq j$. For simplicity, all equal tail variables will be denoted without the vector index, e.g. $z_{(k)}$ for variable index $k \geq n+1$, where $z_{(k)} = z_{(i, k)} = z_{(j, k)} \forall i, j \leq n, i \neq j$. Furthermore, the vectors share the same diagonal variable value, say z_D , i.e. $z_{(i, i)} = z_{(j, j)} = z_D \forall i, j \leq n$. The remaining variables of the vectors

³ ©TMThe MathWorks Inc., Natick, MA (US). The source code of the algorithms is available to download upon request from the authors.

of $\mathcal{Z}$ are all set equal to a same constant value, say z_C , i.e. $z_{(i, k)} = z_{(j, k)} = z_C \forall k \leq n$ and $\forall i, j \leq n$, with $i \neq j$. As proved in [14], all row vectors of such initial solution, referred to as $\mathcal{Z}^0$, have the same pairwise distance for geometric construction. This allows our local search algorithm to start from a solution which already satisfies the first constraint of the problem. Summarizing, the initial solution $\mathcal{Z}^0$ is [14]:

$$\mathcal{Z}^0 = \begin{pmatrix} Z_1 \\ Z_2 \\ \dots \\ Z_{n-1} \\ Z_n \end{pmatrix} = \begin{pmatrix} z_D & z_C & \dots & \dots & z_C & z_{(n+1)} & \dots & z_{(v)} \\ z_C & z_D & z_C & \dots & \vdots & z_{(n+1)} & \dots & z_{(v)} \\ \vdots & \ddots & \ddots & \ddots & \vdots & \vdots & \vdots & \vdots \\ \vdots & \vdots & z_C & z_D & z_C & z_{(n+1)} & \dots & z_{(v)} \\ z_C & \dots & \dots & z_C & z_D & z_{(n+1)} & \dots & z_{(v)} \end{pmatrix}. \quad (3)$$

Consider an arbitrary cosine similarity value $\tilde{cs} \in (0, 1)$, and let $z_C, z_{(n+1)}, \dots, z_{(v)}$ be an arbitrary set of non-negative reals (i.e. $\in \mathbb{R}^+$). Then, $\forall i, j \leq n$, with $i \neq j$, $cs(Z_i, Z_j) = \tilde{cs}$ i.i.f.

$$z_D = \frac{z_C + \sqrt{z_C^2 - \tilde{cs} \cdot \{z_C^2 + (\tilde{cs} - 1)[(n-1)z_C^2 + \sum_{k=n+1}^v z_{(k)}^2\]}}}{\tilde{cs}}. \quad (4)$$

Calculate $cs(Z_i, Z_j)$, $\forall Z_i, Z_j \in \mathcal{Z}^0$, directly by the definition of cosine similarity, and by replacing the component values with those specified in Eq. 3–4. By computing the resulting term, the expression is invariant $\forall i, j \leq n$, $i \neq j$, and it ends exactly in $\tilde{cs}$. For more details on these calculations the reader is referred to [14].

It is worth noting in Eq. 4 that $\Delta = z_C^2 - \tilde{cs} \cdot \{z_C^2 + (\tilde{cs} - 1)[(n-1)z_C^2 + \sum_{k=n+1}^v z_{(k)}^2]\}$ is always non-negative. By computing directly $\Delta \geq 0$, trivially it is obtained:

$$z_C^2 \geq -\frac{\tilde{cs} \cdot \sum_{k=n+1}^v z_{(k)}^2}{1 + (n-1)\tilde{cs}}, \quad (5)$$

which is always true because: $\tilde{cs} \geq 0$; $\sum_{k=n+1}^v z_{(k)}^2 > 0$; $(n-1)\tilde{cs} > 0$ as n is always > 1 ; and then, being the second member of the inequality a negative number, the first member, z_C^2 , is always bigger than the second member [14].

Such produced initial solution $\mathcal{Z}^0$ represents a good starting point for the whole optimization algorithm because it already satisfies the first constraint of the problem (i.e. $cs(Z_i, Z_j) \in [\tilde{cs} - \xi, \tilde{cs} + \xi]$, $\forall Z_i, Z_j \in \mathcal{Z}$). However $\mathcal{Z}^0$ violates the second problem constraint, i.e. that the distribution of the data in $\mathcal{C}$ is a random sample drawn from a von Mises distribution with mean $\mu(\mathcal{C}) \approx \tilde{cs}$ and and concentration parameter $\kappa(\mathcal{C}, \xi) \approx 9$, i.e. $\sigma(\mathcal{C}) \approx \xi/(10/3)$. Indeed, by construction of the initial solution $\mathcal{Z}^0$, the distribution of the data in $\mathcal{C}$ is a Dirac centered in $\tilde{cs}$.

The local search routine proceeds now by perturbing such initial solution $\mathcal{Z}^0$ in order to move gradually the distribution of the data in $\mathcal{C}$ from the Dirac to

a von Mises distribution with the desired properties. This recurring disturbance is performed by an iterative process which, at each iteration, injects a noise $\mathcal{N}$ normally distributed, with $\mu = 0$, to the incumbent solution. The perturbation is greedily accepted iff the modified solution does not violate both the first and second constraints of the problem; otherwise the perturbation is rejected. The satisfaction of the second problem constraint is checked by performing a Watson goodness-of-fit test [39] at 1% significant level to ensure that the modified intermediate solution is consistent with the Von Mises distribution with the desired properties. The first problem constraint is validated instead by making sure that all pairwise cosine similarities among the perturbed vectors fall within the desired interval $[\tilde{c}s - \xi, \tilde{c}s + \xi]$. For this reason, it is needed to inject a noise being small enough to be very likely to be accepted, but not excessively small otherwise the local search routine would become too slow in terms of convergence within a reasonable number of steps. [14] found experimentally that a random noise $\mathcal{N}$ whose components follows a normal distribution with $\mu(\mathcal{N}) = 0$ and $\sigma(\mathcal{N}) = 0.01$ is a good choice yielding a satisfactory tradeoff between convergence speed and constraints violations given by the random perturbation.

The overall local search routine stops when it is reached a perturbed solution satisfying all problem constraints and having a standard deviation of the pairwise cosine similarities equal or larger than the desired threshold value of $\xi/(10/3)$ given by the second problem constraint.

3.2 Reduced Variable Neighbourhood Search Heuristic

An important drawback of the local search routine just described in the previous section is that the normal perturbation that is applied to the incumbent solution is static and does not adapt automatically to the size of the problem to handle. Although in [14] it has been fixed experimentally a random normal noise $\mathcal{N}$ having $\mu(\mathcal{N}) = 0$ and $\sigma(\mathcal{N}) = 0.01$ as a good perturbation setting, on average, it is very likely that, either, this $\sigma(\mathcal{N})$ value will be too small for large size problems, yielding to an excessively slow convergence time for the algorithm, or too large for small size problem instances, producing a large number of constraints violations which will increase as well exponentially the computational running time of the entire procedure.

For this reason we propose here an intelligent optimization approach based on Reduced Variable Neighbourhood Search (RVNS) [9, 21, 30] aimed at achieving high-quality performance for the problem by automating the resulting optimization strategy and adapting online to the size of the problem to tackle. The aim is to lead the local search to achieve a proper balance of diversification (exploration) and intensification (exploitation) during the search process, a fundamental objective for any effective heuristic solution approach. The diversification capability of a metaheuristic refers to its aptitude of exploring thoroughly different zones of the search space in order to identify promising areas. When a promising area is detected, the metaheuristic needs to exploit it intensively to find the relative local-optimum, but at the same time without wasting excessive computational resources. This is referred as the intensification capability of the metaheuristic.

Finding a good balance between diversification and intensification is indeed an essential task for the proper effectiveness of a metaheuristic [3, 4, 23].

RVNS is a variant of the classic Variable Neighbourhood Search (VNS) algorithm [5, 10, 21], which is a popular metaheuristic based on dynamic changes of neighbourhood structures of an incumbent solution during the search process [9, 21]. The VNS methodology is based on the core concept of searching for new solutions in increasingly distant neighbourhoods of the current solution, jumping only if a better solution is found, without being limited only to a fixed trajectory [30]. RVNS is a variant that has been shown to be successful for many combinatorial problems where local optima with respect to one or several neighbourhoods are relatively close to each other [28].

The variable metric method has been suggested by [17]. The idea is to change the metric in each iteration such that the search direction (steepest descent with respect to the current metric) adapts better to the local shape of the function. The RVNS method is obtained if random points are selected from the current neighbourhood under exploration and no descent is followed. Rather, the values of these new points are compared with that of the incumbent and updating takes place in case of improvement. RVNS is a typical example of a pure stochastic heuristic, akin to a classic Monte-Carlo method, but more systematic [28]. It is useful especially for very large problem instances for which the local search within the classic VNS approach might become costly, as it is the case with our problem.

The details of our heuristic based on Reduced Variable Neighbourhood Search for the given problem are specified in Algorithm 1. The algorithm starts with an initial solution $\mathcal{Z}$ equal to $\mathcal{Z}^0$, that is the same starting solution in Eq. 3 described previously in Sect. 3.1. It is by construction a Dirac centered in $\tilde{cs}$, already satisfying the first constraint of the problem but not the second.

Then, the *shaking phase*, which represents the core idea of RVNS, is applied to $\mathcal{Z}$. The shaking phase aims to change the neighbourhood structure, $N_k(\cdot)$, of the incumbent solution to achieve a larger algorithm diversification. The new incumbent solution, $\mathcal{Z}$, is generated at random in order to avoid cycling, which might occur if a deterministic rule is used.

The simplest and most common choice for the neighbourhood structure consists of setting neighbourhoods with increasing cardinality: $|N_1(\cdot)| < |N_2(\cdot)| < \dots < |N_{k_{max}}(\cdot)|$, where k_{max} represents the maximum size of the shaking phase. Let k and k_{step} be, respectively, the current size and the step size of the shaking phase.

The algorithm starts by selecting the first neighbourhood ($k \leftarrow 1$) and, at each iteration, it increases the parameter k if the new incumbent solution violates one of the problem constraints ($k \leftarrow k + 1$), until the largest neighbourhood is reached ($k = k_{max}$). The process of changing neighbourhoods when no improvement occurs diversifies the search trajectory. In particular, the choice of neighbourhoods of increasing cardinality yields a progressive diversification of the search process. Note that the k_{step} parameter has been introduced in

order to adapt the classic RVNS schema from a combinatorial to a continuous optimization setting.

For the given problem, a shaking phase of size k consists of perturbing the incumbent solution $\mathcal{Z}$ with a random noise $\mathcal{N}$ normally distributed with $\mu = 0$ and $\sigma(\mathcal{N}) = k \cdot k_{step}$, producing a perturbed solution in the $N_k(\cdot)$ of the current solution. Note that each perturbation corresponds just to a limited local modification of the incumbent solution $\mathcal{Z}$ which hopefully will not violate both

Input: The number of vectors to generate, n , with $v \gg n$ non-negative data variables; the reference cosine similarity value $\tilde{cs} \in (0, 1)$; and the tolerance $\xi \ll \tilde{cs}$ to bound the cosine similarities in $[\tilde{cs} - \xi, \tilde{cs} + \xi]$

Output: A set $\mathcal{Z}$ of n vectors with v non-negative data variables;

Initialization:

- Let $\mathcal{C} = \bigcup_{i,j=1,j>i}^n cs(Z_i, Z_j)$ be the distribution of the pairwise cosine similarities of the vectors in $\mathcal{Z}$;
- Let k , k_{step} , and k_{max} , respectively the current size, the step size, and the maximum size, of the shaking phase;

begin

- Generate the initial solution $\mathcal{Z}^0$ of n vectors with v variables:
 $(\mathcal{Z}^0, \mathcal{C}) \leftarrow \text{Initial-solution-construction}(n, v, \tilde{cs})$;
- Set $\mathcal{Z} = \mathcal{Z}^0$;
- Set k_{step} and k_{max} arbitrarily;

repeat

- Set $k \leftarrow 1$;
- while** $k < k_{max}$ **do**
 - Set $\mathcal{Z}' = \mathcal{Z}$;
 - Select at random the noise matrix $\mathcal{N}^{n,v}$ whose components follows a normal distribution with $\mu(\mathcal{N}) = 0$ and $\sigma(\mathcal{N}) = k \cdot k_{step}$;
 - Perturb $\mathcal{Z}$: $z_{(i, j)} = z_{(i, j)} \cdot (1 + \mathcal{N}_{(i, j)}) \quad \forall i = 1, \dots, n; \forall j = 1, \dots, v$;
 - Evaluate the pairwise cosine similarities of the modified vectors in $\mathcal{Z}$: $\mathcal{C} = \bigcup_{i,j=1,j>i}^n cs(Z_i, Z_j)$;
 - foreach** $c \in \mathcal{C}$ **do**
 - if** $(c \notin [\tilde{cs} - \xi, \tilde{cs} + \xi])$ **then** //first constraint violated
 - Restore the previous solution: $\mathcal{Z} = \mathcal{Z}'$;
 - if** $k == k_{max}$ **then**
 - Halve the step size of the shaking phase $k_{step} \leftarrow k_{step}/2$;
 - else**
 - Increase the current size of the shaking phase: $k \leftarrow k + 1$;
 - end**
 - **break**;
 - end**
 - end**
 - if** $(\chi^2\text{-test}(\mathcal{C}, \tilde{cs}) == \text{False})$ **then** //second constraint violated
 - Restore the previous solution: $\mathcal{Z} = \mathcal{Z}'$;
 - if** $k == k_{max}$ **then**
 - Halve the step size of the shaking phase $k_{step} \leftarrow k_{step}/2$;
 - else**
 - Increase the current size of the shaking phase: $k \leftarrow k + 1$;
 - end**
 - end**
 - if** $\mathcal{Z} \neq \mathcal{Z}'$ **then** //new perturbed solution accepted
 - Double the step size of the shaking phase $k_{step} \leftarrow k_{step} \cdot 2$;
 - Restart with the first neighbourhood structure: $k \leftarrow 1$;
 - end**
 - until** $\sigma(\mathcal{C}) \geq \xi/(10/3)$;
 - $\Rightarrow \text{Return}(\mathcal{Z})$.

end

Algorithm 1: Reduced Variable Neighbourhood Search heuristic

problem constraints, as already described for the basic local search procedure in Sect. 3.1. Otherwise, if one of the two constraints are violated by $\mathcal{Z}$, the new solution is discarded by restoring the previous solution ($\mathcal{Z} = \mathcal{Z}'$), and this is perturbed again in a larger neighbourhood ($k \leftarrow k + 1$).

The process of increasing progressively the parameter k in case of no improvements occurs until the maximum size of the shaking phase, k_{max} , is reached. When this happens, meaning that the value of k_{step} may be too large having produced already k_{max} consecutive unsuccessful perturbations with the same k_{step} , the step size of the shaking phase is halved ($k_{step} \leftarrow k_{step}/2$) in order to produce next perturbations having standard deviation proportionally smaller ($\sigma(\mathcal{N}) = k \cdot k_{step}$). In this way when the algorithm restarts from the first neighbourhood ($k \leftarrow 1$) of $\mathcal{Z}$, a fine-grained noise will be iteratively produced such that to increase the acceptance chances of the next perturbations.

Conversely, if the perturbed solution $\mathcal{Z}$ does not violate both problem constraints (i.e. $\mathcal{Z} \neq \mathcal{Z}'$), this is accepted as the new incumbent solution and the search is restarted from its first neighbourhood ($k \leftarrow 1$). In this case, the value of k_{step} is doubled ($k_{step} \leftarrow k_{step} \cdot 2$) in order to produce next perturbations having standard deviation proportionally larger. This simple reactive schema is aimed at achieving an optimal setting of k_{step} in order to speeding up the converge speed of the algorithm. Given this reactive schema, the value of k_{max} is not relevant to the overall performance of the algorithm, as this value depends directly from k_{step} which is optimally tuned on-line. Therefore the value of k_{max} is set equal to 5 at the beginning of the algorithm and is not required to change.

The algorithm continues iteratively with the same procedure and stops when the standard deviation of the pairwise cosine similarities of the obtained solution $\mathcal{Z}$ is equal or larger than the expected standard deviation, $\xi/(10/3)$, that is the prefixed problem goal, giving $\mathcal{Z}$ as the final output.

4 Computational Analysis

This section reports our computational experiments on the comparison of the performance of the proposed RVNS approach with respect to that of the local search routine in the literature. The heuristics are identified with the following abbreviations: *LS*, for the local search routine described in Sect. 3.1; and *RVNS*, for the Reduced Variable Neighbourhood Search implementation described in Sect. 3.2.

In the experiments we generate data vectors drawn from a power-law distribution [7]⁴, which is a realistic setting in computational linguistics [35], and considering a number of components, v , ranging from $1 \cdot 10^3$ to $100 \cdot 10^3$ components.

As shown in [14], the easiest setting for our problem is to fix the desired cosine similarity value, $\tilde{cs}$ to 0.5. This setting is sufficient for comparing the performance of the two optimization algorithms by varying the number of components v only.

⁴ Although we generate data from a power-law distribution the approaches can be easily generalized to any kind of distribution of the data.

Table 1. Computational results of the compared algorithms (*LS* and *RVNS*) for $\tilde{cs} = 0.5$ and v ranging from $1 \cdot 10^3$ to $10 \cdot 10^3$ components. The reported results are the average values over 10 problem instances for each components dimension v . For each algorithm, the first column is the average computational running time in seconds (*time*); the second column is the average of the number of unsuccessful iterations (*unsucc*), that is when the perturbed solutions violated one of the problem constraints; the third column is the average of the total number of iterations required by the algorithm to stop (*tot iter*).

<i>size v</i>	<i>LS</i>			<i>RVNS</i>		
	<i>time</i>	<i>unsucc</i>	<i>tot iter</i>	<i>time</i>	<i>unsucc</i>	<i>tot iter</i>
1000	1.00	123.2	172.7	0.43	66.1	77.4
2000	2.21	114.9	162.2	1.41	80.8	94.3
3000	3.18	123.5	169.0	1.60	70.7	82.9
4000	3.86	104.8	154.7	2.40	84.3	98.8
5000	5.23	122.7	167.6	2.97	83.7	98.2
6000	6.67	125.4	176.9	4.98	118.4	137.4
7000	7.04	111.0	158.3	4.47	94.5	110.1
8000	9.29	126.2	180.4	3.40	61.5	71.7
9000	8.88	109.2	156.2	6.65	105.9	123.3
10000	10.29	106.1	149.1	6.67	88.6	103.8
Total:	57.65	1167.0	1647.1	34.98	854.5	997.9

Indeed the computational times to generate the vectors increase proportionally with the number of components v of the vectors. In particular, for high values of v , the computational times may become quite large. In addition, the desired tolerance value, ξ , is set to 0.1, yielding to a bounded interval for the pairwise cosine similarities: $[\tilde{cs} - 0.1, \tilde{cs} + 0.1] = [0.4, 0.6]$ for each considered cosine similarity value $\tilde{cs}$. The number of vectors to be generated is also an arbitrary user choice; in our study we have chosen to generate 100 vectors for each of the controlled circular data group, as in [14].

Our computational results are reported in Table 1 and Table 2, which report a comparison of *LS* and *RVNS* respectively for small a large instances of the problem. In particular, Table 1 contains the results obtained by the two algorithms by setting $\tilde{cs} = 0.5$ and letting v range from $1 \cdot 10^3$ to $10 \cdot 10^3$ number of components, with a step of $1 \cdot 10^3$. Table 2 contains instead the results obtained by the two algorithms for larger instances with $\tilde{cs} = 0.5$ and v ranging from $10 \cdot 10^3$ to $100 \cdot 10^3$, with a step of $10 \cdot 10^3$ components. In both tables, for each dataset having components dimension v we have generated 10 different problem instances, therefore the results reported in the tables are the average values among the 10 generated problem instances⁵. For each algorithm, the first column

⁵ All the computations have been made on an Intel Core i7 microprocessor at 2.6 GHz with 16.0 GB RAM.

Table 2. Computational results of the compared algorithms (*LS* and *RVNS*) for larger problem instances with $\tilde{cs} = 0.5$ and v ranging from $10 \cdot 10^3$ to $100 \cdot 10^3$ components. The reported results are the average values over 10 problem instances for each components dimension v . For each algorithm, the first column is the average computational running time in seconds (*time*); the second column is the average of the number of unsuccessful iterations (*unsucc*), that is when the perturbed solutions violated one of the problem constraints; the third column is the average of the total number of iterations required by the algorithm to stop (*tot iter*).

<i>size v</i>	<i>LS</i>			<i>RVNS</i>		
	<i>time</i>	<i>unsucc</i>	<i>tot iter</i>	<i>time</i>	<i>unsucc</i>	<i>tot iter</i>
$10 \cdot 10^3$	10.29	106.1	149.1	6.67	88.6	103.8
$20 \cdot 10^3$	20.92	103.1	149.9	12.41	85.6	101.4
$30 \cdot 10^3$	32.09	118.1	164.2	19.34	87.8	102.3
$40 \cdot 10^3$	44.15	116.1	166.7	31.54	102.4	119.5
$50 \cdot 10^3$	52.55	113.1	159.8	40.87	107.6	126.1
$60 \cdot 10^3$	63.86	114.2	160.6	40.56	88.2	103.8
$70 \cdot 10^3$	68.16	100.9	146.4	50.82	95.7	111.4
$80 \cdot 10^3$	91.59	123.1	173.7	44.55	73.4	85.9
$90 \cdot 10^3$	105.11	127.5	178.3	59.42	86.8	101.6
$100 \cdot 10^3$	111.04	121.6	170.4	71.74	89	104.3
Total:	599.76	1143.8	1619.0	377.92	905.1	1060.1

in the tables is the average computational running time in seconds (*time*); the second column is the average of the number of unsuccessful iterations (*unsucc*), that is when the perturbed solutions violated one of the problem constraints; the third column is the average of the total number of iterations required by the algorithm to stop (*tot iter*).

Looking at Table 1 containing smaller problem instances, *RVNS* performed better than *LS* in all cases. It was able to generate the required controlled datasets in almost half of the time required by *LS*. The superiority of *RVNS* with respect to *LS* is also highlighted by the smaller number of unsuccessful perturbations, *unsucc*, and the smaller number of iterations, *tot iter*, obtained by *RVNS*. Practically the heuristic based on Reduced Variable Neighbourhood Search results to be a smarter optimization routine than the classic local search in the literature, being able to produce recurrent perturbations with a higher acceptance likelihood.

As shown in Table 2, these results are also confirmed for larger instances of the problem. *RVNS* generated the desired controlled circular data in shorter computational time than *LS*, requiring an inferior number of total iterations and producing a smaller number of unsuccessful perturbations as well. As we can see from this table, for higher values of v the computational times become quite large (see e.g. the case with $v = 100 \cdot 10^3$), confirming the high complexity

of the problem. Nevertheless *RVNS* was able to generate the controlled data in considerably shorter computational time with respect to *LS*, confirming the superiority of the Reduced Variable Neighbourhood Search approach. As shown in our computational analysis *RVNS* is able to produce quickly circular data with the desired properties for vectors with very large components dimension v , which represents an important achievement.

Figure 1 shows an example of a circular dataset generated by the *RVNS* procedure. The histogram of the pairwise cosine similarities distribution in the figure has the typical shape of a normal distribution, which is in fact very similar in the circle to a von Mises of concentration parameter 9. The values fall within the expected interval $[\tilde{c}s - 0.1, \tilde{c}s + 0.1] = [0.4, 0.6]$, showing the ability of the algorithm to produce the data with the desired characteristics.

Fig. 1. An example of a circular dataset generated by the *RVNS* procedure with $\tilde{c}s = 0.5$. As expected the histogram of the pairwise cosine similarities distribution for samples of the generated power-law data vectors has the shape of a von Mises distribution, i.e. the analogous of the normal distribution in the circle.

5 Conclusions

In many artificial intelligence and machine learning environments where stored entities are compared with each other or with incoming patterns, it is often necessary to compute artificially similar data to be used for experiments, simulations, or generally for test purposes. Hypothesizing a computational linguistic setting, we have modelled this problem as a constrained non-linear continuous optimization problem where the goal is to generate controlled datasets of similar circular vectors satisfying the conjecture in Eq. 1 for n documents in a vector space of size v (vocabulary size), and leaving the user the choice of the degree of closeness between the data (i.e. the setting of the pairwise similarity value among the generated vectors). Assuming a cosine similarity metric among the

data, the problem has been conduced to the artificial generation of similar data in the circular space, further increasing the complexity of the problem. In addition, as these controlled datasets are aimed to represent real-world settings in computational linguistics, we have added the further constraint that such circular data need to be power-law distributed, while their pairwise cosine similarity values should follow a von Mises distribution within a desired bounded similarity interval.

In order to address the problem and to produce solutions within reasonable computational running time we have elaborated an optimization procedure based on Reduced Variable Neighborhood Search. In our computational experience we have shown the superiority of the proposed approach with respect to a local search routine in the literature in terms, in particular, of convergence speed. The proposed Reduced Variable Neighbourhood Search is efficient and scale well by adapting automatically to the dimension of the problem to tackle. The presented optimization strategy allows the generation of high-dimensional controlled datasets in the circular space having the desired properties within small computational running time.

References

1. Aizawa, A.: An information-theoretic perspective of TF-IDF measures. *Inf. Process. Manag.* **39**(1), 45–65 (2003)
2. Atkinson, A.C., Riani, M., Cerioli, A.: *Exploring Multivariate Data with the Forward Search*. Springer, New York (2004). <https://doi.org/10.1007/978-0-387-21840-3>
3. Borenstein, Y., Moraglio, A.: *Theory and Principled Methods for the Design of Metaheuristics*. Springer, Heidelberg (2013). <https://doi.org/10.1007/978-3-642-33206-7>
4. Borne, P., Philip, F.G., Popescu, D., Stefanoiu, D., Kamel, A.E.: *Optimization in Engineering Sciences: Approximate and Metaheuristic Methods*. Wiley, New York (2014)
5. Braysy, O.: A reactive variable neighborhood search for the vehicle routing problem with time windows. *INFORMS J. Comput.* **15**(4), 347–368 (2003). <https://doi.org/10.1287/ijoc.15.4.347.24896>
6. Chum, O., Philbin, J., Zisserman, A.: Near duplicate image detection: min-hash and TF-IDF weighting. In: *BMVC 2008 - Proceedings of the British Machine Vision Conference 2008* (2008)
7. Clauset, A., Shalizi, C.R., Newman, M.E.J.: Power-law distributions in empirical data. *SIAM Rev.* **51**(4), 661–703 (2009)
8. Consoli, S., Darby-Dowman, K., Geleijnse, G., Korst, J., Pauws, S.: Heuristic approaches for the quartet method of hierarchical clustering. *IEEE Trans. Knowl. Data Eng.* **22**(10), 1428–1443 (2010)
9. Consoli, S., Darby-Dowman, K., Mladenović, N., Moreno-Pérez, J.: Variable neighbourhood search for the minimum labelling Steiner tree problem. *Ann. Oper. Res.* **172**(1), 71–96 (2009). <https://doi.org/10.1007/s10479-008-0507-y>
10. Consoli, S., Korst, J., Pauws, S., Geleijnse, G.: Improved metaheuristics for the quartet method of hierarchical clustering. *J. Global Optimiz.* **78**(2), 241–270 (2020). <https://doi.org/10.1007/s10898-019-00871-1>

11. Consoli, S., Moreno Pérez, J.: Solving the minimum labelling spanning tree problem using hybrid local search. *Electron. Notes Discrete Math.* **39**, 75–82 (2012)
12. Consoli, S., Recupero, D.R., Petkovic, M. (eds.): *Data Science for Healthcare: Methodologies and Applications*. Springer, Cham (2019). <https://doi.org/10.1007/978-3-030-05249-2>
13. Consoli, S., Stilianakis, N.: A quartet method based on variable neighborhood search for biomedical literature extraction and clustering. *Int. Trans. Oper. Res.* **24**(3), 537–558 (2017)
14. Consoli, S., Turchi, M., Perrotta, D.: A local search approach for generating directional data. *Intell. Data Anal.* **20**(2), 439–453 (2016)
15. Croux, C., Gelper, S., Haesbroeck, G.: Robust scatter regularization. In: Saporta, G., Lechevallier, Y. (eds.) *Compstat 2010, Book of Abstracts*. Paris: Conservatoire National des Arts et Metiers (CNAM) and the French National Institute for Research in Computer Science and Control (INRIA), p. 138 (2010)
16. Csurka, G., Dance, C.R., Fan, L., Willamowski, J., Bray, C.: Visual categorization with bags of keypoints. In: *Workshop on Statistical Learning in Computer Vision, ECCV*, pp. 1–22 (2004)
17. Davidović, T.: Scheduling heuristic for dense task graphs. *Yugoslav J. Oper. Res.* **10**, 113–136 (2000)
18. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *arXiv e-prints arXiv:1810.04805* (2018)
19. Filzmoser, P., Maronna, R., Werner, M.: Outlier identification in high dimensions. *Comput. Stat. Data Anal.* **52**(3), 1694–1711 (2008)
20. Friedman, J., Hastie, T., Tibshirani, R.: Sparse inverse covariance estimation with the graphical lasso. *Biostatistics* **9**(3), 432–441 (2008)
21. Hansen, P., Mladenović, N.: Variable neighborhood search. In: Marti, R., Pardalos, P., Resende, M. (eds.) *Handbook of Heuristics*, pp. 759–787. Springer, Heidelberg (2018). https://doi.org/10.1007/978-3-319-07124-4_19
22. Hoff, P.H.: Simulation of the matrix Bingham-von Mises-fisher distribution with application to multivariate and relational data. *J. Comput. Graph. Stat.* **18**, 438–456 (2009)
23. Hoos, H.H., Stutzle, T.: *Stochastic Local Search: Foundations and Applications*. Morgan Kaufmann Publishers, San Francisco (2005)
24. Jeyakumar, V., Rubinov, A.: *Continuous Optimization: Current Trends and Modern Applications*. Springer, Boston (2005). <https://doi.org/10.1007/b137941>
25. Kusner, M., Sun, Y., Kolkin, N., Weinberger, K.: From word embeddings to document distances. In: *32nd International Conference on Machine Learning, ICML 2015*, vol. 2, pp. 957–966 (2015)
26. Mardia, K.V., Jupp, P.E.: *Directional Statistics*. Wiley, Hoboken (2008)
27. Mikolov, T., Sutskever, I., Chen, K., Corrado, G., Dean, J.: Distributed representations of words and phrases and their compositionality. In: *Advances in Neural Information Processing Systems, NIPS 2013* (2013)
28. Mladenović, N., Petrović, J., Kovačević-Vujčić, V., Čangalović, M.: Solving spread spectrum radar polyphase code design problem by tabu search and variable neighborhood search. *Eur. J. Oper. Res.* **151**(2), 389–399 (2003)
29. Mladenović, N., Todosijević, R., Urošević, D.: Less is more: basic variable neighborhood search for minimum differential dispersion problem. *Inf. Sci.* **326**, 160–171 (2016)

30. Pei, J., Darzić, Z., Drazic, M., Mladenović, N., Pardalos, P.: Continuous variable neighborhood search (C-VNS) for solving systems of nonlinear equations. *INFORMS J. Comput.* **31**, 235–250 (2019)
31. Pennington, J., Socher, R., Manning, C.: GloVe: global vectors for word representation. In: *EMNLP 2014–2014 Conference on Empirical Methods in Natural Language Processing, Proceedings of the Conference*, pp. 1532–1543 (2014)
32. Presutti, V., Nuzzolese, A., Consoli, S., Gangemi, A., Reforgiato Recupero, D.: From hyperlinks to semantic web properties using open knowledge extraction. *Semant. Web* **7**(4), 351–378 (2016)
33. Qian, G., Sural, S., Pramanik, S.: A comparative analysis of two distance measures in color image databases. In: *Proceedings of the 2002 International Conference on Image Processing*, vol. 1, pp. I-401–I-404 (2002)
34. Rice, J.: *Mathematical Statistics and Data Analysis*, 2nd edn. Duxbury Press, Pacific Grove (1995)
35. Salton, G., Wong, A., Yang, C.S.: A vector space model for automatic indexing. *Commun. ACM* **18**, 613–620 (1975)
36. Steinbach, M., Karypis, G., Kumar, V.: A comparison of document clustering techniques. In: *KDD Workshop on Text Mining*, pp. 420–428 (2000)
37. Tan, P.N., Steinbach, M., Kumar, V.: *Introduction to Data Mining*. Addison-Wesley, Boston (2006)
38. Turchi, M., Perrotta, D., Riani, M., Cerioli, A.: Robustness issues in text mining. In: Kruse, R., Berthold, M., Moewes, C., Gil, M., Grzegorzewski, P., Hryniewicz, O. (eds.) *Synergies of Soft Computing and Statistics for Intelligent Data Analysis. AISC*, vol. 190, pp. 263–272. Springer, Heidelberg (2013). https://doi.org/10.1007/978-3-642-33042-1_29
39. Watson, G.S.: Goodness-of-fit tests on a circle. *Biometrika* **48**(1/2), 109–114 (1961)
40. Weston, J., Bengio, S., Usunier, N.: Large scale image annotation: learning to rank with joint word-image embeddings. *Mach. Learn.* **81**(1), 21–35 (2010). <https://doi.org/10.1007/s10994-010-5198-3>
41. Wood, A.T.A.: Simulation of the von Mises Fisher distribution. *Commun. Stat. - Simul. Comput.* **23**(1), 157–164 (1994)
42. Zhang, Y., Jin, R., Zhou, Z.H.: Understanding bag-of-words model: a statistical framework. *Int. J. Mach. Learn. Cybern.* **1**(1–4), 43–52 (2010). <https://doi.org/10.1007/s13042-010-0001-0>

Open Access This chapter is licensed under the terms of the Creative Commons Attribution 4.0 International License (<http://creativecommons.org/licenses/by/4.0/>), which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license and indicate if changes were made.

The images or other third party material in this chapter are included in the chapter's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the chapter's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder.