Using semi-independent variables to enhance optimization search

Gandomi, Amir H.; Deb, Kalyanmoy; Averill, Ronald C.; Rahnamayan, Shahryar; Omidvar, Mohammad Nabi

DOI:

[10.1016/j.eswa.2018.11.031](https://doi.org/10.1016/j.eswa.2018.11.031)

License:

Creative Commons: Attribution-NonCommercial-NoDerivs (CC BY-NC-ND)

Document Version

Peer reviewed version

Citation for published version (Harvard):

Gandomi, AH, Deb, K, Averill, RC, Rahnamayan, S & Omidvar, MN 2019, 'Using semi-independent variables to enhance optimization search', *Expert Systems with Applications*, vol. 120, pp. 279-297.

<https://doi.org/10.1016/j.eswa.2018.11.031>

[Link to publication on Research at Birmingham portal](#)

General rights

Unless a licence is specified above, all rights (including copyright and moral rights) in this document are retained by the authors and/or the copyright holders. The express permission of the copyright holder must be obtained for any use of this material other than for purposes permitted by law.

- Users may freely distribute the URL that is used to identify this publication.
- Users may download and/or print one copy of the publication from the University of Birmingham research portal for the purpose of private study or non-commercial research.
- User may use extracts from the document in line with the concept of 'fair dealing' under the Copyright, Designs and Patents Act 1988 (?)
- Users may not further distribute the material nor use it for the purposes of commercial gain.

Where a licence is displayed above, please note the terms and conditions of the licence govern your use of this document.

When citing, please reference the published version.

Take down policy

While the University of Birmingham exercises care and attention in making items available there are rare occasions when an item has been uploaded in error or has been deemed to be commercially or otherwise sensitive.

If you believe that this is the case for this document, please contact UBIRA@lists.bham.ac.uk providing details and we will remove access to the work immediately and investigate.

Accepted Manuscript

Using Semi-independent Variables to Enhance Optimization Search

Amir H Gandomi , Kalyanmoy Deb , Ronald C Averill ,
Shahryar Rahnamayan , Mohammad Nabi Omidvar

PII: S0957-4174(18)30751-6
DOI: <https://doi.org/10.1016/j.eswa.2018.11.031>
Reference: ESWA 12330

To appear in: *Expert Systems With Applications*

Received date: 3 May 2018
Revised date: 23 November 2018
Accepted date: 24 November 2018

Please cite this article as: Amir H Gandomi , Kalyanmoy Deb , Ronald C Averill , Shahryar Rahnamayan , Mohammad Nabi Omidvar , Using Semi-independent Variables to Enhance Optimization Search, *Expert Systems With Applications* (2018), doi: <https://doi.org/10.1016/j.eswa.2018.11.031>

This is a PDF file of an unedited manuscript that has been accepted for publication. As a service to our customers we are providing this early version of the manuscript. The manuscript will undergo copyediting, typesetting, and review of the resulting proof before it is published in its final form. Please note that during the production process errors may be discovered which could affect the content, and all legal disclaimers that apply to the journal pertain.

Highlights

- A heuristic, semi-independent variable formulation, is introduced to enhance optimization.
- This formulation embodies a set of expected or desired relationships among the original variables.
- Variable interaction analysis is used to measure the complexity of various formulations.
- The efficacy of the heuristic is demonstrated by testing it on various optimization algorithms.
- Population-based algorithms benefited the most from the proposed method.

* Corresponding Author.

Tel.: 201-216-5029

Fax: 201-216-5385

e-mail: a.h.gandomi@stevens.edu

URL: <http://gandomi.beacon-center.org/>

Using Semi-independent Variables to Enhance Optimization Search

Amir H Gandomi^{*1}, Kalyanmoy Deb², Ronald C Averill³, Shahryar Rahnamayan⁴,
Mohammad Nabi Omidvar⁵

¹*School of Business, Stevens Institute of Technology, Hoboken, NJ 07030, USA*

e-mail: a.h.gandomi@stevens.edu

²*Department of Electrical and Computer Engineering, Michigan State University, East Lansing, MI 48824, USA*

e-mail: kdeb@egr.msu.edu

³*Department of Mechanical Engineering, Michigan State University, East Lansing, MI 48824, USA*

e-mail: averillr@egr.msu.edu

⁴*Department of Electrical, Computer and Software Engineering, University of Ontario Institute of Technology, Oshawa, ON, Canada*

e-mail: Shahryar.Rahnamayan@uoit.ca

⁵*School of Computer Science, University of Birmingham, UK*

e-mail: m.omidvar@cs.bham.ac.uk

Table of Contents

Abstract	4
Keywords.....	4
1. Introduction.....	5
2. Background.....	7
2.1 Variable Interaction Analysis	7
2.2 Optimization Algorithms.....	8
2.2.1 Implementation.....	10
3. Semi-Independent Variables.....	10
3.1 Illustrative examples	12
3.1.1 Stepped Beam Design	12
3.1.2 Townsend function.....	19
4. Numerical Case Studies.....	20
4.1 Case 1: Stepped cantilever beam.....	20
4.2 Case 2: Retaining wall design optimization	26
4.3 Case 3: 1 bay-8 story frame design optimization	31
4.4 Case of a 3-bay 15-story frame.....	38
SIV-I Scheme.....	40
SIV-II Scheme.....	40
4.5 Inclusive Discussion of Results	44
5. Summary, Conclusion, and Future Studies	44
Acknowledgment	46
References	46

Abstract

In this study, the concept of a semi-independent variable (SIV) problem representation is investigated that embodies a set of expected or desired relationships among the original variables, with the goal of increasing search effectiveness and efficiency. The proposed approach intends to eliminate the generation of infeasible solutions associated with the known relationships among the variables and cutting the search space, thereby potentially improving a search algorithm's convergence rate and narrowing down the search space. However, this advantage does not come for free. The issue is the multiplicity of SIV formulations and their varying degree of complexity, especially with respect to variable interaction. In this paper, we propose the use of automatic variable interaction analysis methods to compare and contrast different SIV formulations. The performance of the proposed approach is demonstrated by implementing it within a number of classical and evolutionary optimization algorithms (namely, interior-point algorithm, simulated annealing, particle swarm optimization, genetic algorithm and differential evolution) in the application to several practical engineering problems. The case study results clearly show that the population-based algorithms can significantly benefit from the proposed SIV formulation resulting in better solutions with fewer function evaluations than in the original approach. The results also indicate that an automatic variable interaction analysis is capable of estimating the difficulty of the resultant SIV formulations prior to any optimization attempt.

Keywords

Semi-independent variable; Heuristics; Evolutionary computation; Swarm Intelligence.

1. Introduction

Real-world optimization problems are complex, involving various intricacies such as non-differentiable objective and constraint functions, discrete search spaces, non-linear constraints, uncertainties in decision variables and parameters, computationally expensive function evaluations, and high dimensionality. These features create challenges for existing optimization algorithms. In some cases, we can explore and take advantage of available knowledge or desired solution properties (heuristics) to make optimization of such problems more tractable. This is the focus of the current study.

It is generally expected that the incorporation of available knowledge about a problem into the optimization process should enhance its performance. As an example, suppose we want to solve the 8-queen problem (i.e., finding the positions of eight queens on a chessboard such that no two queens can attack each other):

- a) If we do not consider any knowledge about the possible solution, our search space has $64^8 (= 2.8147 \times 10^{14})$ possibilities.
- b) If we consider the simple heuristic "each queen can be placed in any unfilled cell," the number of possibilities would be reduced to $64 \times 63 \times 62 \times 61 \times 60 \times 59 \times 58 \times 57 (= 1.7846 \times 10^{14})$.
- c) If we include the heuristic "no two queens in the same row," the number of possibilities would be $64 \times 56 \times 48 \times 40 \times 32 \times 24 \times 16 \times 8 (= 6.7646 \times 10^{11})$.
- d) If we consider queen1 in row 1, queen2 in row 2, ..., queen8 in row 8, the number of possibilities would be reduced to $8^8 (= 16,777,216)$.
- e) If we impose the constraint that "no two queens can be in the same row or column," the size of the search space would only be $8! (= 40,320)$.

The search space for cases b), c), d), and e) are 1.57, 416, 16.7 million, and 6.98 billion times smaller than the original search space (given in the case a), respectively. Motivated by this example, one would expect that the incorporation of information about the solution of an engineering problem into its formulation or during the search process should have a significant impact on the overall optimization performance. It should be noted that although heuristics are generally beneficial, their overuse can lead to design landscapes that are difficult to search.

In this study, we investigate the effect of incorporating domain knowledge into the optimization process of various structural engineering problems using a range of classic and metaheuristic

algorithms. What we are specifically interested in is to impose a set of relationships among the input variables with the aim of shrinking the search space and eliminating the generation of solutions that do not satisfy the expected solution properties, thereby focusing the search on the parts of the design space that satisfy the desirable characteristics of a potential solution. This approach, which we call semi-independent variable (SIV) formulation, works by identifying a set of base variables (a.k.a independent variables IV) and redefine the remaining ones as functions of the base variables in order to impose any arbitrary relationship we know from experience a solution should have among its components.

As mentioned before, the new formulation results in a smaller search space; however, the imposed interaction among the variables (hence making them semi-independent) can potentially add to the complexity of the new formulation. Therefore, the balance between the performance boost obtained due to shrinkage of the search space and the overall interaction complexity of the semi-independent variables that determine the final efficacy of the new formulation. It should be noted that the semi-independent formulation is not necessarily unique and each may have a different level of variable interaction complexity. It is clear that the formulation with the least level of complexity is preferred. In this paper, we employ state-of-the-art variable interaction analysis algorithms (Omidvar *et al.* 2014, 2017) to assess each SIV formulation prior to any optimization attempt. For the purposes of this paper, we use the differential grouping algorithm (Omidvar *et al.* 2017) (DG2) which can reveal the underlying structure of a given function by quantifying the dependence level between its input variables. DG2 works based on the principle of perturbing the inputs for every pair of variables and measuring the second-order finite differences as an indication of the degree of variable dependence. The output which can be visualized using heat-maps can help in analyzing the underlying interaction structure of the input variables, which we use in this paper to compare SIV formulations.

The organization of the rest of this paper is as follows. First, the concept is explained using an illustrative example – a well-known stepped beam problem. Then, the potential of the use of SIVs is assessed using four other numerical optimization problems. A general case of the given illustrative example is solved, as the first example which is one of the benchmark problems. Then three other complex engineering problems which have been already solved by other researchers in the literature are solved to evaluate the semi-independent variables effects on

performance of variant algorithms. The results confirm the improvements in both convergence rate and final solutions after using them in most problems.

2. Background

This section covers some preliminary concepts before going into the details of the proposed semi-independent variable formulation. Since variable interaction analysis plays a key role in assessing the difficulty of a formulation, the next section is devoted to defining variable interaction, and the differential grouping algorithm which we adopted in this paper for interaction analysis. Finally, the section ends with a brief overview of optimization methods and some implementation details.

2.1 Variable Interaction Analysis

A decision variable $\mathbf{x}_i$ is known as *separable* if:

$$\arg \min_{\mathbf{x}} f(\mathbf{x}) = \left(\arg \min_{x_i} f(\mathbf{x}), \arg \min_{\forall x_j, j \neq i} f(\mathbf{x}) \right), \quad (1)$$

which means that there does not exist any other decision variable interacting with $\mathbf{x}_i$.

Based on the definition of separability, a problem $f(\mathbf{x})$ is known as *fully separable* if:

$$\arg \min_{\mathbf{x}} f(\mathbf{x}) = \left(\arg \min_{x_1} f(x_1, \dots), \dots, \arg \min_{x_D} f(\dots, x_D) \right), \quad (2)$$

where there does not exist any interaction between any pair of decision variables in $\mathbf{x}$. By contrast, a problem $f(\mathbf{x})$ is known as *fully nonseparable* if every pair of decision variables interact with each other.

However, if only part of the decision variables is separable while the others are nonseparable, the problem is known as *partially nonseparable*:

$$\arg \min_{\mathbf{x}} f(\mathbf{x}) = \left(\arg \min_{x_i} f(x_1, \dots), \dots, \arg \min_{x_m} f(\dots, x_m) \right), \quad (3)$$

where $\mathbf{x}_1, \dots, \mathbf{x}_m$ are disjoint sub-vectors of $\mathbf{x}$, and $2 \leq m \leq D$. A problem is *partially additively separable* if:

$$f(\mathbf{x}) = \sum_{i=1}^m f_i(\mathbf{x}_i), \quad (4)$$

where $\mathbf{x}_i$ are mutually exclusive decision vectors of f_i , and m is the number of independent components.

In order to determine the nonseparable groups, Omidvar *et al.* proposed the differential grouping (DG) method on the basis of the following theorem (Omidvar *et al.* 2014):

Theorem 1: Let $f(\mathbf{x})$ be an additively separable function. $\forall a, b_1 \neq b_2, \delta \in \mathbb{R}, \delta \neq 0$. If the following condition holds:

$$\Delta_{\delta, x_p}[f](\mathbf{X})|_{x_p=a, x_q=b_1} \neq \Delta_{\delta, x_p}[f](\mathbf{X})|_{x_p=a, x_q=b_2}, \quad (5)$$

then x_p and x_q are nonseparable, where

$$\Delta_{\delta, x_p}[f](\mathbf{x}) = f(\dots, x_p + \delta, \dots) - f(\dots, x_p, \dots), \quad (6)$$

refers to the forward difference of f with respect to the variable x_p with interval δ .

The quantities in (6) are real-valued numbers; therefore, the equality check cannot be evaluated exactly over the floating-point number field on computer systems. Consequently, the equality check needs to be converted to an inequality check by introducing a sensitivity parameter: $|\Delta^{(1)} - \Delta^{(2)}| > \epsilon$. Here, $\Delta^{(1)}$ and $\Delta^{(2)}$ denote the left and right-hand side of (6) respectively. In the absence of representation and roundoff errors, ϵ can be theoretically set to zero; however, this is not usually the case, and the optimal value of ϵ is often a nonzero positive number. This parameterization makes DG sensitive to choices of ϵ whose optimal value may vary from function to function and difficult to tune by practitioners. To alleviate this problem, Omidvar *et al.* proposed DG2, a parameter-free version of version of DG, which automatically sets ϵ by estimating the bounds on the computational roundoff errors to maximize the accuracy of variable interaction detection (Omidvar *et al.* 2017).

2.2 Optimization Algorithms

Optimization algorithms can be classified based on many features. For example, they can be classified as either population-based algorithms or point-based algorithms. Population-based algorithms are usually gradient-free methods; in general, they use the population information such as natural evolution or natural swarm behavior, instead of a single point information. The mathematical optimization algorithms are usually point-based algorithms which are mostly

gradient based. However, there also exist gradient-free algorithms. Figure 1 shows the different classification based on the above descriptions. As it is shown in this figure, among all the optimization algorithms, the gradient-based algorithms are greedy algorithms as they can be easily trapped in the local minima; the other algorithms called global optimization algorithms.

Figure 1. Classification of optimization algorithms

As it is shown in Figure 1, evolutionary computation, swarm intelligence, and gradient-free trajectory based algorithms are classified as global optimization algorithms. Also, evolutionary computation can be classified to the ones using binary or real-valued variables which have different nature and performance. One mathematical algorithm has been used for benchmarking, and one well-known algorithm is also selected for each of these global algorithms, and their performances are evaluated in the current study. A well-known mathematical optimization algorithm for nonlinear problems is interior-point algorithm (IP) which is used here as the gradient-based algorithm. More information about the mathematical procedure of IP can be found in several texts including Byrd *et al.* (1999). The global optimization algorithms used in this study are simulated annealing (SA) (Kirkpatrick *et al.* 1983), particle swarm optimization (PSO) (Kennedy and Eberhart 1995), genetic algorithm (GA) (Holland 1975), and differential evolution (DE) (Storn and Price 1997). SA algorithm is one of the traditional trajectory based global optimization algorithm. PSO is a typical swarm intelligence algorithm. GA and DE are used here as well-known binary and real-valued evolutionary algorithms, respectively, although

real-valued GAs are also proposed (Janikow and Michalewicz 1991). It should be noted that comparing different algorithms is not the purpose of this study and these algorithms are only selected to evaluate the performance of SIV approach on different classes of optimization algorithms.

2.2.1 Implementation

Standard MATLAB® toolboxes, optimization, and global optimization toolboxes, have been used in order to evaluate the standard and well-tuned form of IP, SA, PSO and GA. For these algorithms, the default and suggested values by MATLAB® have been used for the algorithms' parameters. For the DE algorithm, a developed standard code has been used. The main DE parameters are set based on the suggested values in the literature (Gandomi *et al.* 2012) as differential weight (F) is equal to 0.7 and crossover probability (C_r) is equal to 0.9. The same maximum number of evaluations has been used for all algorithms, and all population-based algorithms, PSO, GA, and DE were stopped if the relative change in best objective function value over the last 20 iterations is less than the tolerance which is set to 1×10^{-6} . Because the stochastic nature of the algorithms, each algorithm is run for 51 times with random initial solution(s) and the mean the convergence histories of the best runs and the medians are reported.

3. Semi-Independent Variables

Sometimes there is a trend among all/some the variables which could be considered as dependence among these variables. In such a case, this set of variables is called semi-independent. This relation can be used and integrated into the optimization process using a heuristic way of redefining variables. The variables in a semi-independent set can be redefined using a base variable, which has the same form as an independent variable (IV), and variable related to the base variable called semi-independent variable (SIV). For instance, if we know that the optimum solution satisfies the conditions $x_1 \geq x_2 \geq \dots x_{i-1} \geq x_i$ and $x_i \leq x_{i+1} \leq \dots \leq x_n$ (shown in Figure 2), we can then define a vector $\mathbf{P}$ such that $0 < p_j \leq 1$ for $j \in \{1, 2, \dots, n-1\}$ and then use it to satisfy the above inequalities. For this purpose, the components of $\mathbf{p}$ can be formulated as follows:

$$x_2 = x_1 \times p_1$$

$$\begin{aligned}
 &\vdots \\
 &x_i = x_{i-1} \times p_{i-1} \\
 &\text{and} \\
 &x_{i+1} = x_i \div p_i \\
 &\vdots \\
 &x_n = x_{n-1} \div p_{n-1}
 \end{aligned} \tag{7}$$

Using these definitions for x_2 to x_n we make sure that all the inequalities have been satisfied. Here, the first variable can be considered as the base variable, and the variables transformation can be defined as $[x_1, \dots, x_n] \rightarrow [x_1, p_1, \dots, p_{n-1}]$ where x represents a regular independent variable, and a p represents an SIV set.

Figure 2. Relation of the variables in the optimum solution known as a heuristic about the problem

In general, the variables can be divided into any number of semi-independent sets. Since the original objective function and the constraints have to be evaluated using the original variables, the SIVs should be mapped back to the original space in each iteration. The simplified framework for optimization by using SIV is shown in Figure 3.

Identification of independent and semi-independent variables and problem formulation.

Initialization of population in the new search space

while stopping criteria is not satisfied

```

Apply the search operators
Update the variables
Map the SIV back to the original space
Evaluate the fitness and update the population

```

```

end

```

```

Map the variables back to the original space for post-processing

```

Figure 3. A simplified framework for integrating SIV in an optimization process.

Redefining the variables by means of SIVs intend to reduce the size of the search space by using the information about the relationship among variables. This mapping can help the optimizer to move more effectively on feasible regions. These points and implementations of SIV are addressed in the next section using an illustrative example. The idea is simple and can be applied to a wide range of optimization problems such as drill shafts, buildings, flywheels, airplane wings, and nose cone vehicles designs.

3.1 Illustrative examples

3.1.1 Stepped Beam Design

This illustrative example is about optimization of stepped cantilever beams with circular sections. The stepped cantilever beam with the circular section is shown in Figure 4, which is scalable. This illustrative example has been investigated along with the detailed implementation of SIV for this problem. The cantilever beam is comprised of different segments of the variable cross-section, and radius (r) of each cross-section is considered as the design variables. Therefore, the problem dimension is equal to a number of its segments.

Figure 4. Schematic of the Stepped beam problem with circular sections

In this case study, the objective function is the volume of the beam, which shall be minimize and is formulated as follows:

$$\min_{r \in \Omega} V = \sum_{i=1}^N l_i \pi r_i^2, \quad (8)$$

where N is the number of segments, l_i is the length of the i th segment, and r_i is the radius of the i th cross-section. The general form of this problem has different kinds of constraints which we solved as our first numerical example. The current illustrative example is only subject to the stress constraints which means bending stresses in each beam segment must be less than the allowable stress (σ_a). Therefore, there are N nonlinear constraints which should be satisfied. For the i^{th} section, the stress constraint can be defined as:

$$\frac{\sigma_i}{\sigma_a} - 1 \leq 0, \quad (9)$$

where

$$\sigma_i = \frac{M_i r_i}{I_i}, \quad (10)$$

$$M_i = F(L + (1 - i) \times l_i), \quad (11)$$

$$I_i = \pi \frac{r_i^4}{4}. \quad (12)$$

The problem bound constraints are $0 \leq r_i \leq 30$, and F is the force is applied to the end of the beam.

Descent Semi-Independent Variables. For this problem, the closer a section is to the restrained side, the larger is its radius. We can use this knowledge to define SIVs, such that the cross-section of each segment is forced to be smaller than all other segments closer to the restrained

side. It should be noted that this section reduction is not a constraint, it is rather a solution we know is viable in practice. This formulation which we call Descending Semi-Independent Variables (DSIV) is formulated as follows. The base variable is set to be the radius of the first section (r_1) which is in the same range as an independent variable ($0 \leq r_1 \leq 30$). All other variables are redefined to be SIVs denoted as $(p_1, \dots, p_{n-1})$. Here, each SIV (p_j) is between 0 and 1 ($0 < p_j \leq 1$) and the relationship between p_j and the independent variable r_j is defined to be $r_j = r_{j-1} \times p_j$, where r_{j-1} is the radius of the adjacent segment closer to the restrained side. Therefore, the radius of a section with the exception of the first one can be obtained using the following equation:

$$r_i = r_1 \times \prod_{j=1}^{i-1} p_j \quad \text{for } i \in \{2, \dots, N\}. \quad (13)$$

It should be noted that in this formulation, the radius of the SIVs should be updated from a larger radius to a smaller radius. In other words, the closer a section radius is to the base radius, the sooner it gets updated.

Ascent Semi-Independent Variables. Another way of formulating the SIVs for this problem is to take the radius of the segment at the end of the beam as the base variable and force the radius of all other segments to be larger as they get closer to the restrained side. This formulation which we call Descending Semi-Independent Variables (DSIV) is formulated as follows. The base variable is set to be the radius of the first section (r_1) which is in the same range as an independent variable ($0 \leq r_1 \leq 30$). All other variables are redefined to be SIVs denoted as $(p_1, \dots, p_{n-1})$. Here, each SIV (p_j) is between 0 and 1 ($0 < p_j \leq 1$) and the relationship between p_j and the independent variable r_j is defined to be $r_j = r_{j-1} \times p_j$, where r_{j-1} is the radius of the adjacent segment closer to the restrained side. Therefore, the radius of a section with the exception of the first one can be obtained using the following equation: This formulation which we call Ascending Semi-Independent Variables (ASIV) is formulated as follows. The base variable is set to be the radius of the last section (r_N) which is in the same range as an independent variable ($0 \leq r_N \leq 30$). All other variables are redefined to be SIVs denoted as $(q_1, \dots, q_{n-1})$. Here, each SIV (q_j) is between 0 and 1 ($0 < q_j \leq 1$) and the relationship between q_j and the independent variable r_j is defined to be $r_j = r_{j+1} \div q_j$, where r_{j+1} is the radius of the adjacent

segment closer to the free end. Therefore, the radius of a section with the exception of the first one can be obtained using the following equation:

$$r_i = r_N \times \prod_{j=N-1:-1}^i \frac{1}{q_j} \quad \text{for } i \in \{1, \dots, N-1\}. \quad (14)$$

It should be noted that in this formulation, the radius of the SIVs should be updated from a smaller radius to a larger radius. In other words, the closer a section radius is to the free end radius (r_N), the sooner it gets updated.

Using the two formulations mentioned above (DSIV and ASIV), our knowledge of the problem ($0 \leq r_N \leq r_i \leq r_1 \leq 30$) is utilized in the modeling of the problem and is automatically satisfied during the search process. Both formulations also reduce the search space, which for a two-dimensional version of the problem is effectively half of the original search space (Figure 5). The rectangle shown in Figure 5 represents the search space for a 2D version of the problem before an SIV formulation. The triangular gray area shows the search space for both SIV formulations mentioned above, which force $r_2 \leq r_1$. This clearly shows how the incorporation of expert knowledge about the characteristics of a viable solution can significantly reduce the size of the search space. For a general N-dimensional stepped beam problem, the search space of the transformed SIV formulation is reduced to $1/2^{N-1}$ of the original search space.

Figure 5. Search space before and after using SIV for 2D scenario

Analysis of SIV Formulations

As was previously mentioned, different SIV formulations can have various effects on the variable interaction structure of the resultant objective functions. Our intuitive understanding of the SIV concept may suggest that introducing explicit linkages between the input variables for the sake of shrinking the search space will inevitably result in stronger nonlinearity and a more complex interaction structure among the input variables. Contrary to this commonsensical intuition, some SIV formulations can indeed result in objective functions with simpler variable interaction structure and reduced nonlinearity. In what follows, we compare and analyze the original formulation of the illustrative example with the two SIV formulations presented above. We also show how a variable interaction analysis algorithm such as Differential Grouping (DG2) (Omidvar *et al.* 2017) can give us insight into the relative complexity of various formulations.

Figures 6(a) shows the landscape of the search spaces and constraints for the original formulation of the stepped cantilever beam with circular sections, and Figures 6(b) and (c) show those of the transformed search spaces for 2D cases using descent and ascent SIVs formulations respectively. In other words, the triangular area under the diagonal dashed line in Figure 6(a) is mapped to Figures 6(b) and 6(c) using different mapping transformations. The boundaries of (the constraints) feasible and infeasible solutions are shown with the marked lines and in the global best, highlighted point, both constraints are active. From the landscape contours, it is clear that the search space is twisted after defining the SIV sets. By comparing these two plots, it is clear that one of the linear constraints in Figure 6(a) is turned to a nonlinear constraint in the twisted search space (Figure 6(b)) because of the dependency of the second variable to the first variable. However, it is not the case in Figure 6(c) as the constraint is a linear constraint. From Figures 6(b) and 6(c) it is clear that the feasible search area near to global optimum is wider in Figure 6(b) compare to Figure 6(c).

Figure 6. The landscape of the objective function and constraints (a) original search space, (b) mapped search space by using descending SIV, and (c) mapped search space by using ascending SIV

Another point is that, from Figure 6(a), it seems the hit ratio (ratio of feasible space to whole search space) is increased after using SIV. However, after plotting the new search spaces, it is clear that the hit ratios are decreased. This difference comes from mapping the triangular space to the rectangular space, which should be considered for analyzing the twisted search space. It should be noted that the slopes of stress constraints and also the landscape of the objective function in the original space are directed toward the global optimum. In other words, the physics of the problem lead us to toward the best solution.

The above analysis shows that each mapping may have different features and has its own advantages and disadvantages, which should be considered and evaluated. In general, finding a tradeoff between cutting the search space and adding to the complexity of the problem is crucial. Here we employ the Differential Grouping (DG2) algorithm (Omidvar *et al.* 2017) to perform such a tradeoff analysis prior to any optimization effort.

Figures 7(a)-(c) show the output of DG2 on different formulations of a 10-dimensional stepped cantilever beam with circular sections. The color intensity in each cell signifies the interaction level between each pair of the input variables (darker colors mean stronger interactions). Among the three, the original formulation (Figure 7(a)) has the largest search space and the lowest interaction level among its variables. We know that both SIV formulations significantly reduce the size of the search; however, this comes at the expense of introducing some degree of variable interaction between the decision variables. Figures 7(b) and 7(c) clearly show that the descending SIV formulation (DSIV) is less complex than the ascending SIV formulation (ASIV). In DSIV only the first two variables interact while the other variables are separable. In ASIV however, the last variable interacts with every other variable making the problem harder to solve. The scale of the numbers in Figures 7(b) and 7(c) also shows that the intensity of interactions in the ASIV formulation is much higher than those of DSIV. The variable interaction analysis of the three formulations suggests that the DSIV formulation should have the best optimization performance due to its smaller search space while maintaining a simple variable interaction structure. In the next section, we use four engineering problems to test the efficacy of various SIV formulations. As shall see later, the prediction of variable interaction analysis about the effectiveness of various formulations is consistent with the optimization results.

Figure 7. DG values for (a) IV, (b) descending SIV, and (c) ascending SIV.

3.1.2 Townsend function

The second illustrative example is the Townsend function, shown in Figures 8(a) and 8(d). As it is shown, this problem involves multiple local minima and maxima within its search space and has the following formula:

$$f(x_1, x_2) = -[\cos((x_1 - 0.1)x_2)]^2 - x_1 \sin(3x_1 + x_2),$$

where $0 \leq x_1, x_2 \leq 5$.

By imposing $x_2 \leq x_1$ using an SIV formulation, we have $x_2 = p x_1$, where $0 \leq p \leq 1$. The contour and 3D plots of the problem before and after applying SIV are shown in Figures 8(a) to 8(f). As shown in Figures 8(a) and 8(d), the original formulation has multiple local minima and maxima. However, after applying SIV, we actually cut half of the original search space (Figures 8(b) and 8(e)). In the case of maximization, Figures 8(c) and 8(f) show that not only some of the local maxima are eliminated, but also the search space is smoothened after the application of SIV. The same can be said in the case of minimization of the Townsend function. Therefore, this illustrative example shows how SIV formulation can help eliminating some local minima and smooth out the landscape in such a case.

Figure 8. Search space before and after using SIV the Townsend function. (a) original contour plot, (b) cut contour plot, (c) contour plot with SIV, (d) original 3D plot, (e) cut 3D plot, (f) 3D plot with SIV.

4. Numerical Case Studies

In this section, we investigate the efficacy of the proposed SIV formulations using four well-known engineering problems. The first problem is a stepped cantilever beam design problem with rectangular sections is a continuous problem which is a generalized form of the illustrative example with three different classes of constraints. The second problem is cost minimization of retaining wall structures which is a mixed variable problem. The third problem is weight optimization of a 1-bay 8-story frame with predefined beam and column sections as variables which makes it a discrete optimization problem. The fourth problem is weight minimization of a 3-bay 15-story frame. All these problems have been solved using the optimization algorithms, IP, SA, PSO, GA, and DE. To minimize the stochastic nature of algorithms on the reported results, summary statistics are reported based on 51 independent runs.

4.1 Case 1: Stepped cantilever beam

This problem is the general case of the illustrative example originally presented by Thanedar and Vanderplaats (1995) which has been used by many researchers as a benchmark for structural optimization problems (Gandomi and Yang 2011). The schematic presentation of the problem is given in Figure 9. The cantilever beam is comprised of different segments of variable cross-section. The since the height (h) and the width (b) of the cross-sectional area of each segment are chosen as design variables.

Figure 9. Schematic of the Stepped beam problem with rectangular section

In this case study, the objective function is the minimization of the volume which can be simply formulated as follow:

$$\min_{b, h \in \Omega} V = \sum_{i=1}^N l b_i h_i \quad (15)$$

where the number of segments is equal to 5, b_i and h_i are respectively the i^{th} section width and height. This problem is subjected to the following eleven constraints:

- Stress constraints: The bending stresses in each beam segment (σ_i) must be less than the allowable stress (σ_a):

$$\frac{\sigma_i(b_i, h_i)}{\sigma_a} - 1 \leq 0, \quad (16)$$

and $\sigma_i(b_i, h_i)$ indicates the mean bending stresses in each beam segment, which is related to the cross-section variables.

- Deflection constraint: The deflection of the cantilever beam (δ) must be smaller than the allowable deflection (δ_a):

$$\frac{\delta(b_1, \dots, b_5, h_1, \dots, h_5)}{\delta_a} - 1 \leq 0, \quad (17)$$

where δ is a function of all variables.

- Aspect ratio constraints: The aspect ratio between the height and the width of the cross-section of each segment must be less than 20, which can be formulated as:

$$\frac{h_i}{20b_i} - 1 \leq 0, \quad (18)$$

where $1 \leq b_i \leq 5$ and $30 \leq h_i \leq 65$. From this formulation, the ratio of feasible space to infeasible space (hit ratio) is very low and it is approaching zero for dimensions above 10.

SIV Case 1: Descent SIVs. For this problem, two sets of descent SIVs are considered, one for section widths ($\mathbf{b}$) and one for section heights ($\mathbf{h}$). Like the illustrative example, from left to right the sections always reduce in this problem. The base variable here is considered as the first section width (b_1) and the first section height (h_1) which has the same bounds as the independent variable ($1 \leq b_1 \leq 5$ and $30 \leq h_1 \leq 65$). The other variables are formulated as SIVs such that the cross-section of each segment (both height and width) is forced to be smaller than all other segments closer to the restrained side. In this formulation, which we call Descending SIV (DSIV), $p_1, \dots, p_4$ are replaced with $b_2, \dots, b_5$ and $q_1, \dots, q_4$ are substituted with $h_2, \dots, h_5$ in the optimization process where the elements of vectors $\mathbf{p}$ and $\mathbf{q}$ values are between 0 and 1. Therefore, width and height in other sections except first section ($i \in \{2, \dots, 5\}$) can be obtained as follows:

$$b_i = 1 + p_{i-1}(b_{i-1} - 1) \quad (19)$$

$$h_i = 30 + q_{i-1}(h_{i-1} - 30) \quad (20)$$

SIV Case 2: Ascent SIVs. As mentioned in the illustrative example, another way of formulating SIVs is to take the variables associated with the end segment as base variables and relatively define all the other variables such that they are forced to grow larger as they get closer to the restrained side. In this formulation, which we call Ascending SIV (ASIV), b_5 and h_5 are considered as the base variables and the remaining variables are defined as follows:

$$b_i = 1 + \frac{b_{i+1} - 1}{p_i} \quad (21)$$

$$h_i = 30 + \frac{h_{i+1} - 30}{q_i}, \quad i \in \{1, \dots, 4\}. \quad (22)$$

Figure 10 shows the result of variable interaction analysis of the three formulations using DG2. Similar to the illustrative example, the color intensity signifies the strength of interactions with lighter colors denoting weaker interactions. As can be seen, the original non-SIV formulation has a complicated interaction structure making it a nonseparable problem (Figure 10(a)). It is interesting to see that all the width variables fully interact with uniform strength, while the interaction strength of the height variables with their respective width variable fades out as the sections get farther away from the restraining side. In the DSIV formulation, however, the interaction structure is considerably simpler. Figure 10(b) shows that the two base variables (h_1

and b_1) have a strong dependence, but the remaining variables either interact weakly or are completely separable. This formulation not only makes the search space significantly smaller, but also makes the problem simpler to optimize. Therefore, the DG2 analysis suggests that the overall optimization performance of DSIV should outperform the original formulation.

With respect to the ASIV formulation, the base height variable (h_5) have a very strong interaction with its respective base variable (b_5) as well as all other semi-independent height variables (q_1 to q_4). The scale in Figure 10(c) shows that the strength of interactions in this formulation is significantly higher than all other formulations. This indicates that although the search space in the ASIV formulation is smaller than the original formulation, its complex interaction structure is detrimental to the optimization performance. Overall, the DG2 analysis suggests that the DSIV formulation should outperform all other formulations due to its smaller search space as well as a simpler variable interaction structure. As shall be seen later, this prediction is completely consistent with the optimization results using four different optimizers (shown in Figure 11). It should also be noted that the optimizers do not make explicit use of the variable interaction information discovered by DG2. Other studies have shown that incorporating these information into the optimization process can improve the convergence behavior (Yu *et al.* 2009; Mei *et al.* 2016; and Yang *et al.* 2016); however, this study is beyond the scope of this paper.

Figure 10. DG values for (a) IV, (b) DSIV, and (c) ASIV.

The Wilcoxon rank-sum test is performed at a significance level of 5% to compare the results obtained by IV, DVIS, and ASIV for different methods. The medians of the solutions for the SA, PSO, GA, and DE global algorithms are presented in Figure 11. The results of the IP algorithm are excluded from Figure 11 due to its poor performance and low success rate. The experimental

results clearly show that all DSIV formulations consistently outperform the non-SIV formulations. This is in contrast with ASIV formulations where SA achieves the only successful case. The Wilcoxon rank-sum test with Holm p-value correction is performed at a significance level of 5% to compare the results obtained by IV and SIV approaches. The results for different methods with the null hypothesis of equal medians are presented in Table 1. Comparing IV with SIV, the results indicate the rejection of the null hypothesis for all algorithms except GA and significantly better performance for SA, PSO, and DE algorithms. Comparing IV and ASIV, we can see the significantly worse performance by ASIV. From DSIV and ASIV results, it can be seen that DSIV is significantly better than ASIV in all cases.

Table 1. p-values based on a 95% confidence interval for different algorithms.

Optimization Method	IV vs DSIV	IV vs ASIV	DSIV vs ASIV
SA	0.0006 ↑	N/A*	N/A
PSO	0.0034 ↑	0.0004 ↓	0.0000 ↓
GA	0.8777 ≈	0.0024 ↓	0.0037 ↓
DE	0.0000 ↑	0.2059 ≈	0.0000 ↓

↑, ≈, and ↓: Second method performs statistically significantly better, equal and worse. * it is N/A since some of SA-ASIV runs cannot find the feasible region.

Figure 11. Median results of various optimization algorithms for the stepped beam design problem using different SIV formulations.

The convergence histories of the SA, PSO, GA and DE algorithms are presented in Figures 12(a) to 11(d), respectively. In each plot, three different strategies are compared, including IV (non-SIV), DSIV, and ASIV versions of algorithms. Figure 12(a) shows that both DSIV and ASIV approaches can improve the convergence of SA. Although SA-DSIV had a slower convergence into the feasible region, it could ultimately find better solutions. It should be noted that in 7 out of 51 runs, SA could not even find the feasible region. Therefore, these 7 runs are eliminated from the mean. In the case of PSO, the DSIV scheme improves the regular IV strategy while the ASIV scheme performs similarly to the non-SIV case. In GA, DSIV cannot improve the algorithm's performance, while ASIV performing worse than the baseline non-SIV case (Figure 12(c)). Figure 12(d) shows that ASIV cannot improve the performance of DE, while the DSIV strategy can significantly outperform the other two strategies.

Figure 12. Mean convergence histories of the stepped beam design problem using (a) SA, (b) PSO, (c) GA, and (d) DE algorithms with IV, DSIV and ASIV.

From the Figure 13 (c) GA, it seems using ASIV twisted landscape in a way that was not helpful. As it is mentioned, one of the reasons is that slopes of stress constraints and also the landscape of the objective function in the original space are directed toward the global optimum, but they may not as before, after using the SIV strategies. For this case, although ASIV cuts and limits the search, twisting the landscapes may make the problem more difficult.

From the mean of the section widths found by PSO and shown in Figure 13, it can be seen that SIV formulations keep the desired trend in the final section width values. However, the trend cannot be found when the regular IV considered for solving this problem using PSO algorithm. It should be noted that all the other global optimization algorithms have found the trends in their final solutions.

Figure 13. Mean section widths using the PSO algorithm with IV, DSIV, and ASIV.

4.2 Case 2: Retaining wall design optimization

To design a retaining wall, the objective function is defined based on the cost of the structure. Cost is determined by the estimated combined cost of steel reinforcement and concrete

(including the price of materials and labor) per meter (\$/m). The objective function can be generally formulated as follow:

$$\min_{R,X \in \Omega} Cost = C_c V_c + C_s W_{st}, \quad (23)$$

where C_c is unit cost of concrete, V_c is the volume of concrete, C_s is unit cost of steel, and W_{st} is the weight of steel.

Design variables for the retaining wall, shown in Figure 14, include: width of base slab ($X1$), width of the slab ($X2$), stem thickness at the bottom of the wall ($X3$), stem thickness at the top of the wall ($X4$), base slab thickness ($X5$), distance from the front of the shear key to the front of the toe of the wall ($X6$), width of the shear key ($X7$), height of the shear key ($X8$), the vertical steel area in the stem per unit length of wall ($R1$), the horizontal steel area of the toe slab ($R2$), the horizontal steel area of the heel slab ($R3$), and the vertical steel area of the shear key per unit length of wall ($R4$). For the steel reinforcement areas, discrete variables are considered including 223 values for each area. There are several effective loads on retaining wall including weight of all the sections of the reinforced concrete wall, the weight of backfill acting on the heel of the wall, the weight of soil on the toe of the wall, the distributed surcharge load, the force resulting from the active earth pressure, the forces resulting from passive earth pressure on the base shear key and the front part of the toe section of the wall, and the force resulting from the bearing stress of the base soil. The more modeling detail and problem formulation (such as objective function and constraints) can be found in Gandomi *et al.* (2015).

Figure 14. Design variables for the general retaining wall.

The final retaining wall design has to satisfy three factors of safety including overturning factor, sliding factor, and bearing factor. Additionally, each section of the wall (stem, heel, toe, and shear key) must be checked for shear and moment capacity. These geotechnical and structural requirements are respectively formulated based on ACI 318-05 (2005) and Das (1994) and totally results in 24 nonlinear constraints. There are some inequality geometrical constraints also defined for this problem to prevent infeasible retaining wall dimensions (Camp and Akin 2012). One of them is simply formulated as $X_6 + X_7 \leq X_1$. It is also known that $X_4 \leq X_3$ even though they have the same range in the problem definition. These two conditions can be satisfied by using SIV scheme. For the $X_4 \leq X_3$, the inequality condition can be easily satisfied by defining one SIV for X_4 called p as follows:

$$X_4 = LB_4 + p(X_3 - LB_4). \quad (24)$$

For the other inequality condition, however, two SIVs are needed. Using two SIVs for, called q_1 and q_2 , to satisfy the relationship among, X_1 , X_6 and X_7 by the following formulation:

$$X_6 = LB_6 + q_1(X_1 - LB_6 - LB_7), \quad (25)$$

$$X_7 = LB_7 + q_2(X_1 - X_6 - LB_7). \quad (26)$$

It should be noted that X_6 should be updated before X_7 because its value is needed for updating X_7 (Eq. 22). Using Eq. 21, the X_6 is a value between LB_6 and $X_1 - LB_7$, and using Eq. 22 is a value from its lower bound (LB_7) to the $X_1 - X_6$ and, therefore, $X_6 + X_7 \leq X_1$ is satisfied.

Although these two constraints are not active in the global optimum solution, handling them forces the algorithm to abandon searching in the infeasible parts of the search created by these constraints. Therefore, the least expectation is that they improve the convergence of the algorithm to get near to the optimum region. This problem is optimized using the mathematical method (IP) and four global optimization algorithms, SA, PSO, GA, and DE algorithms.

Figure 15. DG values for (a) IV, (b) SIV.

Figure 15 shows the variable interaction analysis of DG2 on the original and the SIV formulations of the retaining wall problem. As can be seen, both formulations have an identical interaction structure. Therefore, the SIV formulation has the advantage of shrinking the search space without having a detrimental effect on the complexity of interactions. As a result, the algorithms capable of exploiting the shrinkage in the size of the search space should perform better than others.

The median of the solutions for the namely SA, PSO, GA, and DE algorithms are shown in Figure 16. From this figure, it seems even handling two out of 26 constraints by proposed approach can slightly improve the final results in all the global algorithms. IP algorithm with IV cannot find a feasible solution in most runs, but after using SIV in the IP algorithm, it could find the feasible solution in most cases. However, the median solution of IP with SIV is still far from

solutions found by global optimization algorithms presented in Figure 16. This also shows the problem is highly constrained and leading to a feasible solution is important for this problem. After Wilcoxon rank-sum test for Medians of final results, it is observed that (p -value = 0.012) only the DE with SIV significantly outperforms the algorithm without SIV.

Figure 16. Median solution of SA, PSO, GA and DE algorithms for retaining wall design problem with and without SIV approaches.

The mean convergence histories of PSO, GA, and DE algorithms are respectively presented in Figure 17 (a) to (c). It should be noted that the SA algorithm does not work properly for this problem, and about 40% of runs it did not find feasible solutions in both cases. From this figure, it is clear that using the SIVs can clearly improve the convergence rate of the population-based algorithms and the evolutionary algorithms in particular. Although the geometry constraints are not active for the final solutions, it seems it helps the search process by leading the solutions to the feasible region and guides them toward the optimum. From the presented results in Figure 16 and 17, are not only the convergence histories show better convergence behavior, but also the algorithms reach to better final solutions because of applying the SIV scheme.

Figure 17. Average convergence histories of best fitness values for the population-based algorithms. (a) PSO, (b) GA and (c) DE

4.3 Case 3: 1 bay-8 story frame design optimization

In this problem, the design of a one-bay eight-story framed story frame is going to be optimized which is one of the benchmark structural engineering problems (Gandomi and Yang 2011). The configuration of the framed structure and its applied wind loads are shown in Figure 18. This case is a challenging structural design optimization problem, and it has been already solved by some other researchers in the field.

Figure 18. Topological structure of the one-bay eight-story frame.

Fabrication conditions are imposed on the construction of the frame as requiring the same beam section to be used for two following stories, and that the same column section is used for every two consecutive stories. Considering the conditions and also symmetry for this 24-member frame, the elements are classified into eight different groups as it is illustrated in Figure 18. The lateral drift at the top of the structure is the only performance constraint. The structural analysis part has been done by using matrix stiffness method implementation of finite element method coded in MATLAB®. The objective function here is the minimization of the frame weight (W) which can be formulated as:

$$\min_{B, C \in \Omega} W = \sum_{i=1}^{NB} \rho \left(\sum_{j=1}^{NEB_i} l_{i,j}^b \right) B_i + \sum_{i=1}^{NC} \rho \left(\sum_{j=1}^{NEC_i} l_{i,j}^c \right) C_i, \quad (27)$$

where NB and NC are respectively the numbers of beam and column groups and both are equal to 4, NEB_i and NEC_i are respectively the number of elements in i^{th} beam and column groups, ρ is density, $l_{i,j}^b$ and $l_{i,j}^c$ are respectively the length of j^{th} element in the i^{th} beam group, and B_i and C_i are respectively the section area for the i^{th} beam and column groups. This problem is a discrete optimization problem and the values of the element groups are chosen from 267 W-shape sections, in other words $B_i, C_i \in \{W_1, \dots, W_{267}\}$.

In this problem, we also know that it is not practical to install a column with a larger section area on a column with smaller section area. Also, physics of the problem tell us the lower column carry more load, and therefore the section area should be more or equal for a lower story. The same also happens for the beams and physics of the problem tells that a higher level beam should have less or equal section area. Based on these conditions, two sets of SIVs are defined for beams (p_1, p_2 , and p_3) and columns (q_1, q_2 , and q_3) which can be formulated as:

$$B_1, C_1 \in \{W_1, \dots, W_{267}\},$$

for beams group sections $b \in \{2, 3, 4\}$:

$$B_b = LB_b + p_{b-1} \times (B_{b-1} - LB_b), \quad (28)$$

for columns group sections, $c \in \{2, 3, 4\}$:

$$C_c = LB_c + q_{c-1} \times (C_{c-1} - LB_c). \quad (29)$$

Figure 19. DG values for (a) IV, (b) SIV.

The variable interaction analysis of DG2 (shown in Figure 19) reveals that the SIV formulation simplifies the interaction pattern of the variables to a great extent. In the original formulation, all variables interact with varying dependence levels, while the SIV formulation results in an almost separable problem containing only three interacting variables. This means that the SIV formulation not only makes the search space smaller but also reduces the complexity of the problem. The experimental results, which will be presented next, confirm that the SIV formulation results in a performance boost in most optimizers.

Figure 20 shows the convergence plots for the SA, PSO, GA, and DE algorithms with and without SIVs. It can be seen that using SIV significantly improves the performance of all four global algorithms, and the convergence history of the algorithms have a huge gap when they use the SIV approach. This Figure clearly shows the effectiveness of the SIV for this knotty engineering problem.

Figure 20. Convergence histories for the 1-bay 8-story frame problem using (a) SA, (b) PSO, (c) GA and (d) DE algorithms with and without SIV.

The results of the median fitness for the used algorithms are presented in Figure 21. First, it should be noted that IP did not converge to the optimum, as it is expected. However, it gets more benefit from using SIV than any other method. From the figure, it is also clear that the final PSO results are almost the same as it is converged to the optimum in both cases. However, as it is shown in Figure 20(b), SIVs significantly improve the convergence of PSO as the PSO-SIV converge to an optimum value after about 500 function evaluations (FE) which was not found by PSO-IV even after 1,500 FE. This is very important for such tedious problems, and expensive optimization in general, which involve finite element method (FEM), computational fluid dynamics (CFD) or complex computation and each simulation may take several hours/days. The final results are significantly improved in the other global optimization, SA, GA, and DE algorithms when SIV are used. From the Wilcoxon rank-sum test results, it is observed that SIV significantly improves all the algorithms performances except the PSO algorithm.

Figure 21. Median results of different algorithms for the 1-bay 8-story frame problem using different optimization algorithms with IV and SIV.

As it is mentioned, two SIV sets have been used in this problem, one for columns and one for beams. The mean of final solutions which are section area of columns and beams are respectively presented in Figures 22 (a) and (b). From Figure 22 (a) it can be seen that all the final columns' cross-section areas are monotonically reducing by increasing the number of the story except the ones obtained using the evolutionary algorithms, GA and DE, without using the SIVs for columns section areas. For the beams, the results are fascinating and show that none of the algorithms can find the descending behavior of the beams by increasing the story heights when they do not use SIVs. These results clearly show that how the SIVs can be highly necessary for such a problem.

Figure 22. Mean of final solutions (cross-section areas) using SA, PSO and GA algorithms and with and without SIVs. a) columns, b) beams

Figures 23(a) to (f) show the convergence histories of the mean standard deviation of columns and beams in both independent and semi-independent forms, in PSO, GA, and DE algorithms, respectively. At first, it is clear that using SIVs in all the algorithms decrease the standard deviation of the variables (columns and beams section areas).

Based on the defined formulation for the SIVs in this problem reflected in Eqs. (24) and (25), the dependency of the variables is increased with the number of the story. In other words, C2 depends on C1, C3 to C1 and C3, and C4 depends on all other columns' cross-sections below it. What is consistent in Figures 23(a) to (f) is that the more dependency for a variable, the less variation (or standard deviation) during the search process. Therefore, the variables in a higher

story have less variation history no matter which optimization method or which SIV set (columns or beam) is considered. This history behavior also confirms that more dependencies can be regarded as more limitations for an SIV to guide it toward the global optimum.

Figure 23. Variation of average value of standard deviations of variable values with generations and using SIV and not using SIV for (a) columns in PSO, (b) beams in PSO, (c) columns in GA, (d) beams in GA, (e) columns in DE, and (f) beams in DE.

4.4 Case of a 3-bay 15-story frame

The 3-bay 15-story frame design problem has been solved by researchers as a benchmark frame structure problem. The topology and the service loading conditions of the structure are shown in Figure 24. This design optimization problem has three beams and four columns in each story. Fabrication conditions are imposed on the construction of the frame as requiring the same beam section to be used for all stories, and that the same column section is used for every three following stories. The problem has one constraint on the roof drift, and all 105 members should satisfy the AISC (2001) combined strength constraints. More details about the case study such as material and mechanical properties and can be found in Talatahari *et al.* (2015).

Figure 24. Topology of the 3-bay 15-story frame.

The structural analysis part has been done by using matrix stiffness method implementation of finite element method coded in MATLAB®. The objective function here is the minimization of the frame weight (W) which can be formulated as:

$$\min_{A \in \Omega} W = \sum_{i=1}^{NG} \rho \left(\sum_{j=1}^{NE_i} l_{i,j} \right) A_i, \quad (30)$$

where NG is the number of groups, NE_i is the number of elements in i^{th} group, ρ is density, $l_{i,j}$ is the length of j^{th} element in the i^{th} group, and A_i is the section area for the i^{th} group of the element. All members have been grouped into 11 member groups (or 11 design variables). Columns are grouped in 10 column groups, and all beams are in one beam group. In this case study, all beam and column element groups should be chosen from all 267 W-shapes, in other words $A_i \in \{W_1, \dots, W_{267}\}$.

SIV-I Scheme. In this problem, we know that it is not practical to install a column with a larger section area on a column with smaller section area. Also, physics of the problem tell us the lower column carry more load, and therefore the section area should be more or equal for a lower story. Based on these conditions, two sets of SIVs are defined for inner columns (p_1, p_2, p_3 and p_4) and outer columns (q_1, q_2, q_3 and q_4). All the SIVs here are between 0 and 1, and they are related to two base variables, the first inner and outer column groups (A_1 and A_6). Therefore, the variables can be redefined as:

$$A_1, A_6 \in \{W_1, \dots, W_{267}\},$$

for inner columns group sections, the base variable is A_1 and its related SIVs are column groups 2 to 5 which are defined as follow:

$$A_{ic} = LB_{ic} + p_{ic-1} \times (A_{ic-1} - LB_{ic}), \quad (31)$$

where $ic \in \{2, 3, 4, 5\}$ for outer column group sections, the base variable is A_6 and its related SIVs are column groups 7 to 10 which are defined as follow:

$$A_{oc} = LB_{oc} + q_{oc-6} \times (A_{oc-1} - LB_{oc}), \quad (32)$$

where $oc \in \{7, 8, 9, 10\}$

SIV-II Scheme. In addition to the physical information mentioned in SIV-I, we know that, in each floor, an inner column has a bigger section in comparison with that of the outer column; hence this can be used as a heuristic. It is because an inner column has more tributary area. Using this information, we can say that $A_1 < A_6$, and, therefore, they can be related using a SIV.

Considering this relation between inner and outer columns, now we can redefine only one SIV set for both inner and outer columns, which can be formulated as:

$$A_1 \in \{W_1, \dots, W_{267}\}.$$

There are nine SIVs as $0 < p_1, p_2, \dots, \text{ and } p_9 \leq 1$ and the SIVs are defined as follow:

$$A_c = \begin{cases} LB_c + p_{c-1} \times (A_{c-1} - LB_c) & c \in \{2, 3, 4, 5\} \\ LB_6 + p_5 \times (A_1 - LB_6) & c = 6 \\ LB_c + p_{c-5} \times (A_{c-1} - LB_c) & c \in \{7, 8, 9, 10\} \end{cases} \quad (33)$$

Figure 25. DG values for (a) IV, (b) SIV-I, (c) SIV-II.

The variable interaction analysis results presented in Figure 25 show that both SIV formulations simplify the interaction pattern to a great extent. Among the two SIV formulations, SIV-I has a slightly higher interaction level between several input variables as compared to SIV-II. The overall conclusion is that both SIV formulations should have a positive impact on the optimization performance due to the reduction in the size of the search space as well as the simplified interaction structure.

Figure 26 shows the best convergence plots for the SA, PSO, GA and DE algorithms with IV, SIV-I and SIV-II strategies. For the SA algorithm, it seems none of the SIV strategies work, and with using SIVs the results are getting worse. However, SA cannot reach the global optimum area which is less than 500, even after 10,000 function evaluations. For PSO, also the SIV strategies only improve the convergence rate at the beginning of the search process, and they could improve the results while the number of function evaluations is less than 5,000. It should be noted that 5,000 function evaluations is a very high number for such an expensive optimization problem and usually far less number of evaluation is used in practice. For the two evolutionary algorithms, GA and DE, both SIV strategies work very well and improve the

convergence rate and final results. Another point from Figures 26 (a) to (d) is that using SIV-II always outperform the SIV-I, which interestingly says that more dependency and merging two SIV sets can improve the performance. From the results, it can be concluded that if SIV can be used in different ways, they all should be tried, merging different SIV sets in particular.

Figure 26. Convergence history for the 3-bay 15-story frame problem using different optimization algorithms with and without SIV. (a) SA, (b) PSO, (c) GA and (d) DE algorithm.

The results of median fitness for the used global algorithms are presented in Figure 27. First, it should be noted that IP did not converge to the optimum and order of the values obtained for the frame weight is far from the ones achieved by the global algorithms. As seen from the convergence plots, none of the SIV-I and SIV-II can improve the SA algorithm, and they both lose the accuracy. The SIV strategies, from the final PSO results, it can be seen that using SIV

strategies slightly underperform the PSO with IV. However, as it is shown in Figure 26 (b), SIV strategies improve the convergence rate of PSO which is vital for tedious problems involve FEM. The final results are significantly improved in GA and DE algorithms when either of SIV-I or SIV-II is used. Comparing the median results of SIV-I and SIV-II schemes, again, it can be seen that SIV-II is always outperformed SIV-I which shows merging SIV sets and increase the dependency improves the final solutions, no matter which optimization algorithm has been used. The Wilcoxon rank-sum test with Holm p-value correction is performed with null hypothesis of equal medians to compare the results obtained by IV, SIV-I and SIV-II approaches for different methods. From Table 2, the results indicate the significantly better performances of SIV approaches only for evolutionary algorithms, GA and DE. Comparing SIV approaches, we can see SIV-II can significantly outperform the SIV-I approach.

Figure 27. Median results of different algorithms for the 3-bay 15-story frame problem using different optimization algorithms with original variables (IV), SIV-I and SIV-II.

Table 2. p-values based on a 95% confidence interval for different algorithms with and without SIV.

Optimization Method	IV vs SIV-I	IV vs SIV-II	SIV-I vs SIV-II
SA	0.0000 ↓	0.0067 ↓	0.2965 ≈
PSO	0.0000 ↓	0.0067 ↓	0.0110 ↑
GA	0.0000 ↑	0.0000 ↑	0.3154 ≈
DE	0.0000 ↑	0.0000 ↑	0.0010 ↑

↑, ≈, and ↓: Second method performs statistically significantly better, equal, and worse.

4.5 Inclusive Discussion of Results

As it is discussed in section 3, imposing a set of relationships among the input variables using SIV formulation results in shrinking the search space and eliminating the generation of solutions that do not satisfy the expected solution properties, thereby focusing the search on the parts of the design space that satisfy the desirable characteristics of a potential solution. However, this advantage does not come for free and the imposed interaction among the variables (hence making them semi-independent) can potentially add to the complexity of the new formulation. Four case studies are investigated here to investigate the tradeoff between cutting the search space and adding to the complexity of the problem.

The case studies' results clearly show that the population-based algorithms can significantly benefit from the proposed SIV formulation resulting in better solutions with fewer function evaluations than in the original approach. The first case-study compared IV, DSIV, and ASIV formulations. In this case-study, variable interaction analysis indicated that the ASIV formulation is more complex than the DSIV formulation and the experimental results confirmed this assessment. For all algorithms with the exception of GA, the DSIV formulation performed significantly better than the original formulation. In case-studies 2 to 4, SIV improved the convergence rates and the final solution quality of both GA and DE algorithms. For the PSO algorithm, although the convergence rates are boosted, the final results did not improve. In all cases, the results also indicate that an automatic variable interaction analysis is capable of estimating the difficulty of the resultant SIV formulations prior to any optimization attempt. Overall, the results illustrate that the use of SIV in customizing the representation of an optimization problem is highly effective in reducing the overall computational cost.

5. Summary, Conclusion, and Future Studies

Incorporating knowledge within an optimization algorithm can significantly enhance its performance. In this study, a new heuristic has been proposed to preserve the variables' relations during the optimization process. We call this encoding of information by means of enforcing functional relationships among input variables of an optimization problem the semi-independent variable (SIV) formulation. The proposed approach is simple and can significantly reduce the search space, thereby focusing the search effort on parts of the space containing the solutions with desired properties. We have shown that SIV formulations are not unique and each may have

different variable interaction properties. Although all SIVs are designed to reduce the size of the search space, not all SIV formulations result in more straightforward optimization problems. To address this issue, we have shown how variable interaction analysis algorithms such as Differential Grouping can be used to shed light on the relative complexity level of various formulations. The variable interaction analysis allows practitioners to predict the relative performance of different formulations prior to any optimization effort.

The empirical results on four engineering problems confirm that the conclusions derived from the variable interaction analysis are consistent with the actual optimization results. The empirical evidence suggests that SIV formulations that result in a simpler variable interaction structure have a positive impact on the final optimization performance. This is due to the shrinkage of the search space as well as the reduced complexity of the resulting optimization problem. In some cases, however, the SIV formulation results in a more complex variable interaction structure. In such cases, the final result may vary depending on whether the shrinkage in size or the added complexity due to the SIV formulation is dominant. In our experiments, the SIV formulations with a more complex problem interaction pattern often have a detrimental effect on the overall optimization performance.

Besides the reduction in the size of the search space and the complexity of the resultant SIV formulations, the search behavior of various optimizers may result in various responses to different SIV formulations. For example, the rotational invariance property of optimization algorithms has a direct impact on their performance on problems with various degrees of variable interaction. In this study, we used four well-known global optimization algorithms, SA, PSO, GA, DE, and a classic mathematical algorithm for nonlinear optimization to evaluate the performance of different SIV formulations. Among these algorithms, DE and GA benefited the most from the SIV formulations. These two algorithms are generally more robust to the transformations of the search space as compared to the other algorithms. In the case of PSO, the convergence speed is improved in most cases due to the shrinkage in the size of the search space. However, PSO does not seem to benefit from the simplified variable interaction structure. We attribute this behavior to PSO's lack of rotational invariance. Unlike most algorithms which have a consistent response to SIV formulations, the performance of SA varies from problem to problem. Overall, the results have demonstrated that the use of an SIV approach improves the

convergence rate and the final solution quality. Population-based optimization algorithms have shown to better exploit the SIV formulations as compared to the point-based methods.

As the SIV approach has different effects on different algorithms, further studies can focus on designing particular algorithms to work more efficiently with SIVs. In this study, the optimization algorithms do not make explicit use of the structural information extracted by means of variable interaction analysis. Incorporation of such information through explicit problem decomposition can further improve the optimization results. For the future studies, we also focus on seeding and niching using SIV. The performance of the SIV formulation for other types of problems (such as multi-objective optimization and dynamic optimization) could also be a topic of future research.

Acknowledgment

This material is based in part upon work supported by the National Science Foundation under Cooperative Agreement No. DBI-0939454. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

References

- ACI (2005). Building code requirements for structural concrete and commentary (ACI 318-05), American Concrete Institute, Detroit.
- AISC (2001) Manual of steel construction—Load resistance factor design, 3rd Ed., American Institute of Steel Construction, Chicago.
- Byrd, R.H., Mary E. Hribar, and Jorge Nocedal, (1999) "An Interior Point Algorithm for Large-Scale Nonlinear Programming," SIAM Journal on Optimization, Vol 9, No. 4, pp. 877–900.
- Camp, C. V., Akin, C. (2012). "Design of retaining walls using big bang-big crunch optimization Optimum Design of Cantilever Retaining Walls." J. Struct. Eng. 138, 438-448.
- Das, B. M. (1994). "Principles of geotechnical engineering", PWS Publishing, Boston.

Gandomi A.H., Kashani AR, Roke DA, Mousavi M, (2015) Optimization of retaining wall design using recent swarm intelligence techniques. *Engineering Structures*, Volume 103, 15 November 2015, 72–84

Gandomi A.H., Yang X. S., (2011) Benchmark problems in structural optimization, in: *Computational Optimization and Applications*, (Eds. X. Y. Yang and S. Koziel), vol. 356/2011, Springer, 259–281

Gandomi AH, Yang XS, Talatahari S, Deb S. (2012) Coupled eagle strategy and differential evolution for unconstrained and constrained global optimization. *Computers & Mathematics with Applications* 63 (1), 191–200.

Holland, J.H., (1975). “Adaptation in natural and artificial systems” (Ann Arbor: The University of Michigan Press, c).

Janikow, C. Z.; Michalewicz, Z. (1991). "An Experimental Comparison of Binary and Floating Point Representations in Genetic Algorithms". *Proceedings of the Fourth International Conference on Genetic Algorithms*: 31–36.

Kennedy J, Eberhart R. (1995) Particle swarm optimization. In: *Proceeding of the IEEE on international conference on neural networks*. Perth, Australia. p. 1942–8.

Kirkpatrick, S.; Gelatt Jr, C. D.; Vecchi, M. P. (1983). *Optimization by Simulated Annealing*. *Science*. 220 (4598): 671–680.

Mei, Y., Omidvar, M. N., Li, X., & Yao, X. (2016). A competitive divide-and-conquer algorithm for unconstrained large-scale black-box optimization. *ACM Transactions on Mathematical Software (TOMS)*, 42(2), 13.

Omidvar, M. N., Li, X., Mei, Y., & Yao, X. (2014). Cooperative co-evolution with differential grouping for large scale optimization. *IEEE Transactions on Evolutionary Computation*, 18(3), 378-393.

Omidvar, M. N., Yang, M., Mei, Y., Li, X., & Yao, X. (2017). DG2: A Faster and More Accurate Differential Grouping for Large-Scale Black-Box Optimization. *IEEE Transactions on Evolutionary Computation*, 21(6), 929 – 942, 2017.

Storn, R., Price, K. (1997). Differential Evolution – A Simple and Efficient Heuristic for Global Optimization over Continuous Spaces. *Journal of Global Optimization* 11(4), pp. 341-359.

Talatahari, S., Gandomi, A.H., Yang, X.S. and Deb, S., "Optimum design of frame structures using the eagle strategy with differential evolution." *Engineering Structures* 91 (2015): 16-25.

Thanedar P.B., Vanderplaats G.N. (1995) Survey of discrete variable optimization for structural design. *J Struct Eng, ASCE*, 2 (121), pp. 301–306.

Yang, M., Omidvar, M. N., Li, C., Li, X., Cai, Z., Kazimipour, B., & Yao, X. (2017). Efficient Resource Allocation in Cooperative Co-Evolution for Large-Scale Global Optimization. *IEEE Transactions on Evolutionary Computation*, 21(4), 493-505.

Yu, T. L., Goldberg, D. E., Sastry, K., Lima, C. F., & Pelikan, M. (2009). Dependency structure matrix, genetic algorithms, and effective recombination. *Evolutionary Computation*, 17(4), 595-626.