Domer, B., Raphael, B., Shea, K. and Smith, I.F.C. "A study of two stochastic search methods for structural control", J of Computing in Civil Engineering, Vol 17, No 3, 2003, pp 132-141. http://cedb.asce.org

A study of two stochastic search techniques for structural control

Bernd Domer[1], Benny Raphael[2], Kristina Shea[3], Ian Smith, M. ASCE[4]

Keywords: optimization, stochastic search, simulated annealing, global search, structural control, tensegrity structures

Abstract

Many engineering tasks involve the search for good solutions among many possibilities. In most cases tasks are too complex to be modeled completely and their solution spaces often contain local minima. Therefore, in general classical optimization techniques cannot be applied effectively. This paper studies two stochastic search methods, one well established (simulated annealing) and one recently developed (PGSL), applied to structural shape control. Search results are applied to control the quasi-static displacement of a tensegrity structure with multiple objectives and interdependent actuator effects. The best method depends on the accuracy related to requirements defined by the objective function and the maximum number of evaluations that are allowed.

1. Introduction

Typical civil engineering tasks are complex. Nevertheless, goals may be modeled using non-monotonic objective functions to identify good solutions in very large and irregular solution spaces. In such situations, deterministic search methods, such as hill-climbing, are not reliable. The risks of terminating in a locally optimal solution are too great and differences between the local optima identified and better solutions can be too important.

Since modeling is rarely complete in civil engineering, overall optimality cannot usually be claimed. However, optimally directed algorithms are useful for finding good solutions that can subsequently be tested in real situations. Therefore, in many practical engineering contexts, stochastic optimization methods are useful for search support.

[1] Res. Assistant, Structural Engineering Institute, EPFL, 1015 Lausanne, Switzerland, bernd.domer@epfl.ch
[2] Res. Associate, Structural Engineering Institute, EPFL, 1015 Lausanne, Switzerland
[3] Lecturer, Cambridge University Engineering Department, Cambridge, UK
[4] Prof. M. ASCE, Structural Engineering Institute, EPFL, 1015 Lausanne, Switzerland

Over the past twenty years, several stochastic search methods have been proposed to increase reliability when looking for good solutions in complex search spaces. Examples include global random search methods (Masri et. al. 1980), (Raphael and Smith 2000), genetic algorithms (Goldberg 1989) and simulated annealing (Kirkpatrick et. al. 1983). Examples of applications in engineering include shape and topology optimization (Shea and Smith 1999; Deb and Gulati 2001) and building design (Grierson and Khajepour 1999). Latest developments are systems which evolve iteratively using genetic algorithms for multi-objective optimization tasks (Parmee et. al., 2000).

Algorithms have been compared on the same task using the same goals, models and constraints (El-Beltagy and Keane 1999) in order to evaluate their relative performance. For example, Youssef (Youssef et. al. 2001) tested genetic algorithms, simulated annealing and tabu search on VLSI circuit design. The main conclusion was that good results can be obtained with each one of the studied techniques when they include domain-specific knowledge. Wolpert and Mcready (Wolpert and Mcready 1997) proposed “no free lunch theorems” for optimization algorithms that do not use problem-specific tuning. Algorithms that perform well for a class of tasks do not necessarily produce good results when applied to other classes. Generally, no algorithm that is tuned only once is best for all tasks. Therefore identification of good matches between task classes and algorithms is of practical interest for effective application in engineering.

Applications of stochastic search in control involve sensor or actuator placement (Han and Lee, 1999), system identification (Kristinsson and Dumont, 1992) and state estimation of the controlled system (Gremling and Passino 2000). Genetic algorithms have been used in all cases. Chattopadyay and Seely (Chattopadyay and Seely 1994) addressed system identification in combination with actuator location by comparing simulated annealing with non-linear programming (NLP). Simulated annealing was more efficient with respect to CPU-

time than NLP in this case. In the field of structural engineering Khot (Khot 1998) added the minimization of controller movements to the same task. Salama et. al. (Salama et. al. 1993) used simulated annealing in combination with a linear finite element evaluation of control moves of a precision truss structure where the search was effective but observed non-linear behavior lead to inaccuracies.

This paper studies the use of two stochastic search methods applied to a non-linear and highly coupled task in structural control. The methods are simulated annealing and a new algorithm called probabilistic global search Lausanne (PGSL). Application of two methods to the same engineering task aims to provide insights into matching algorithms with tasks. This paper examines the use of search in structural control from a different perspective to previous work. While almost all previous studies involve minimization of the acceleration feedback gain for a control command, this structure requires multiple and closely-coupled control moves. Control moves cannot be found by direct evaluation. Therefore, generate-and-test type algorithms are applied to find control commands that are evaluated by non-linear analysis. In contrast with other studies, multiple optimal solutions exist that often involve significantly different control commands.

The size of the search space is approximately 5×10^{20} solutions (500 million trillion) and the objective function is known to be non monotonic. The time needed to compute all possible solutions on a modern PC is estimated to 4.8×10^{12} years (see section 2.1). In the next section, a description of the structural control application will be given. Section 3 then describes simulated annealing and PGSL implementations in general. In Section 4, search results are compared according to criteria such as the best overall result and progression of the search graphs from which conclusions can then be made.

2. Structural control application

Research in the field of structural control may be classified into two categories, passive and active structural control. Tuned mass dampers (TMD) are an example of passive structural control. They are added as auxiliary mass to a building and adjusted so that energy is transferred to them and then dissipated away. They are mostly used to reduce vibration amplitudes in chimneys, towers and pedestrian bridges. Since the effect of passive systems mostly depends on the existence of well-defined undamped response, their application is limited. For the reduction of the vibration of high-rise buildings, practical difficulties arise when tuned mass dampers are considered alone. Active systems, where the auxiliary mass can be moved using a motor, are used to enhance performance (Nishimura et. al. 1992). The majority of control applications are intended to reduce vibrations.

Research that is described in this paper focuses on another type of active structural control (Shea et. al. 2001). In addition to conventional structural control, this work aims to control the displacement response of a structure using interdependent actuator commands. Using the three tier control hierarchy of execution, coordination and management (Passino 1996), this task is the coordination control level. In this context, coordination means coupled control of several actuators. An important research activity involves using an actively controlled structure to improve its performance over time by integrating learning and planning methods into the control algorithm. Figure 1 shows the information flow related to the active control research that is described in this paper. Note that a structural analysis is performed for each iteration.

FIG. 1. Finding control commands through stochastic search

2.1. Control of tensegrity structures

The expression “tensegrity” was initially employed by Buckminster Fuller (Fuller 1962). It is a contraction of tensile integrity and describes a structure where compression members are

held apart by a network of tension members. The equilibrium of this structure is obtained by its self-stress state (Motro 1992, Williamson and Skelton 1998).

Such tensile structures are self-supporting and do not need heavy foundations or anchorages. They can be easily transported due to their modular construction and therefore can be used effectively for temporary events. Attaching fabric could create a tent-structure. The structural system is appealing because it combines criteria for efficient use of building materials with aesthetic goals in an original way.

The Applied Computing and Mechanics Laboratory (IMAC) has constructed a novel tensegrity-structure consisting of three modules (see Figure 2). Each module is composed of 6 bars and 24 cables. Bars and cables are linked by specially designed connections and the entire assembly is simply supported at only three positions (A, B and C, Figure 2). The center of each module consists of a central node where all bars of one module meet. A more detailed description of the structure is given in (Fest et. al. 2001).

FIG. 2. Tensegrity structure built from three modules (A, B, and C are supports)

When using this system as a temporary roof for an exhibition, large deflections can occur that are caused by actions such as snow, wind and rain induced ponding. IMAC's tensegrity structure is equipped with actuators to adjust the length of the struts such that large deflections are reduced. The actuators consist of five telescopic bars per module each of which can be adjusted in steps of 0.25mm.

For a given loading of the structure, a control movement for every telescopic bar needs to be determined to control the shape. Stochastic search is used to select good combinations. Results of Perelli (Perelli 2000) showed that structural behavior is non-linear even when control commands are on the order of one millimeter. Although the materials used (steel cables and reinforced polyester for the bars) can be assumed to behave linear-elastic (provided that no cable becomes slack), the system is geometrically non-linear. Also, each solution is

sensitive to the initial positions of the telescopic bars. These may vary when the structure is exposed to a whole cycle of different load cases. The search-space is expressed in terms of the number of possible solutions, *num*, and may be evaluated as follows:

$num = P^n$ with

n: number of telescopic bars (3 modules × 5 telescopic bars = 15)

P: number of possible bar positions (3/0.25 × 2 = 24). (bars may move ± 3mm)

Assuming that each calculation, including a non-linear analysis, takes 0.3 seconds, the time needed to evaluate all possible is 4.8×10^{12} years.

TABLE 1. Moore's law applied to the control problem

Table 1 shows the application of Moore's law, which states that processor speed doubles every 18 months, to our control problem. The size of the task increases through adding more modules. Processor speed increases cannot help here since the task is exponential in terms of computational complexity. This aspect is discussed in more detail in Section 3.

2.2. Dynamic relaxation for cable structure simulation

Approaching the shape control task using a generate-and-test search method requires an appropriate analysis method. Compared to other civil engineering structures, cable structures have special characteristics. They have geometrically non-linear behavior and as a result, the equation of the equilibrium cannot be formulated on the undeformed structure. Also, Maxwells rule cannot be applied without modification to test determinacy (Calladine 1978).

The dynamic relaxation method is popular for analyzing cable structures because it includes geometric and material non-linearities efficiently and without matrix inversion. Using the dynamic equation of a damped system with externally applied load,

$$p(t) = M\ddot{d} + C\dot{d} + Kd$$

the residual force at each node of the structure is calculated. After each time step, residual forces are summed in order to check if the equilibrium state of the structure has been reached. This is indicated when the sum is lower than a threshold value that reflects the precision required.

Although dynamic relaxation uses a dynamic description of the problem, it is used to solve static problems. The masses and damping coefficients used in this formulation are fictitious and chosen such that they lead to a rapid convergence. By the use of "kinetic damping" the algorithm implemented by Rossier (Rossier 1994), which is used in the optimization process to evaluate control solutions, avoids arbitrary selection of the viscous damping parameter.

The method accommodates geometrically non-linear systems through formulating the equilibrium of residual forces on the deformed system. Material non-linear behavior can be introduced with each new time step, because no pre-assembled stiffness matrix is used. This is especially useful for cable structures. If a compressive force is detected in a cable during the iterative process, the inner force of the cable is set to '0' for the next iteration. Since the algorithm also calculates displacements as well as the forces in the cables and bars, a supplementary form finding process is therefore not necessary.

3. Stochastic search

3.1. Stochastic search and engineering tasks

Examples of applications of stochastic optimization in engineering include:

- structural optimization (Ceranic et. al. 1999)
- spatial layout (Cagan, Degentesh and Yin 1998)
- multi-criteria optimization of building design with respect to capital cost, revenue income and life-cycle cost (Grierson and Khajepour 1999).

Optimization techniques such as linear and non-linear programming are often applicable. However, there is a certain class of problems where the application of deterministic techniques is not tractable. Intractability means that execution time increases exponentially with the number of optimization variables. One frequently used example of an intractable problem is the “traveling salesman problem”. The search for an optimal arrangement of cities to be visited by the salesman with respect to minimizing the tour length increases exponentially with the number of cities on the tour. One might argue that the application of Moore’s law will provide us with enough computer power to solve these complex problems. This is not true, as we have already demonstrated in Section 2.1.

Search techniques have been developed to help provide good solutions for intractable problems. Gradient search techniques identify a region of good solutions and a downward path is followed by accepting only better solutions. In complex solution spaces this method is likely to identify only local minima.

Stochastic search methods have been created to overcome these drawbacks. With reference to Reeves (Reeves and Beasly 1995) stochastic search may be defined as

> *“A method that makes use of random numbers and is able to find good solutions within reasonable time without guaranteeing the optimum”*

Near optimal solutions are sufficient for most engineering tasks.

3.2. Simulated annealing

3.2.1 Simulated annealing implementation

Simulated annealing (Kirkpatrick, 1983) stems from an analogy to the annealing of metals where temperature schedules are used to control the arrangement of atoms during their crystallization process. It is a step-wise technique that allows moves to inferior solutions and therefore is able to overcome local minima, as shown in Figure 3.

FIG. 3. Schematic comparison of simulated annealing (SA) and a descent strategy (DS) (modified from Dowsland 1995). f(x) is the objective function.

This process is driven by the function

$$P_{accept} = e^{-\frac{\Delta C}{T}}$$

The change in the objective function, or cost, between two moves is denoted by ΔC. P_{accept} is compared to a randomly generated value between 0 and 1 and the inferior candidate solution is accepted when the random value is less than P_{accept}. In general, the temperature at the beginning of the process is fixed for each problem as a schedule parameter and is then reduced to zero during the optimization process according to an "annealing schedule". During the last section of this schedule, the "freezing" stage, only better moves are accepted and simulated annealing behaves as a descent strategy.

Simulated annealing algorithms are reasonably robust if the parameters controlling the cooling curve are assigned values that reflect the complexity of the solution space. This implementation uses the modified Lam-Delosme annealing schedule to define schedule characteristics (Swartz and Sechen 1990). This schedule operates by assuming an optimal profile for the percentage of candidate solutions that should be accepted at each iteration of the search process and adjusting the temperature over a statistical interval to achieve this target accept rate (Figure 4).

FIG. 4. Example of accept rate and temperature schedules

Starting from an initial temperature, $T_{initial}$, the temperature update is defined as:

$$T^{n+1} = \frac{1-(accept\_rate_{actual} - accept\_rate_{target})}{K} \times T^{n}$$

where K is a constant; a value of 10 has been found to be effective. Other schedule parameters affecting the performance of this schedule include the number of candidate solutions considered in each iteration, the number of iterations in a complete search process, how often the temperature is updated and over what statistical interval.

From the initial state, candidate solutions are generated by selecting a single system variable at random and perturbing it within the allowable variable ranges. This implementation makes use of the concept of Hustin move sets (Hustin 1988) where each "move" is assigned a sub-range of the maximum allowed variable change. Since the move sub-range only defines the upper limit of a variable change, smaller moves are always possible. Each move range, *r*, is then assigned a quality, Q_r:

$$Q_r = \frac{\sum_{acceptrules} |\Delta C(r)|}{number\_of\_attempted\_rules}$$

according to the change in cost of past applications of the rule, ΔC(r). Rule qualities are used to update the probability of selecting a move sub-range in subsequent perturbations of the solution. Generally, larger moves are used in the beginning of the process whereas smaller moves are used towards the end as the solution converges. While newer annealing schedules may be available that are along similar lines to those described, these techniques have been proven successful in the domain of structural topology optimisation (Shea et. al. 1997).

3.2.2 Schedule parameter setting

The schedule parameters were set in order to converge to optimally directed solutions in the least amount of time. For the structural control problem studied, the number of iterations was set to 150 where each iteration consisted of 100 candidate solutions. A guideline given in (Swartz and Sechen 1990) for the number of moves per iteration is

$$10 \times n_{variables}^{\frac{4}{3}}$$

with $n_{variables}$ = number of variables. While this was used as a starting point for setting the number of moves it was found that good convergence was achieved in far less moves for this particular task.

A further 40 "freeze" iterations were performed at the end of the process where only better solutions were accepted. Additional parameters were then set to allow the process to stop if

absolute and relative convergence criteria were met. In the results presented, all processes converged within the first two iterations of the "freeze" process. The initial temperature was set to 20, based on the numeric range of the cost function, and updated every ten moves. Six sub-ranges for variable changes were used to define the move set. As simulated annealing works best for incremental small changes of solutions, the first four ranges cover 50% of the total range, 3 mm, while the remaining two are set at 70% and 100% of the maximum range. All search parameters were held constant across all load cases.

3.3. Probabilistic Global Search Lausanne (PGSL)

3.3.1 An introduction to PGSL

PGSL has been developed at IMAC (Raphael and Smith 2000). It has already been applied to several tasks in the field of structural engineering, such as optimization problems in timber structures (Svanerudh et. al. 2001) and bridge diagnosis (Robert-Nicoud et. al. 2000). PGSL uses the assumption that better points are more likely to be found in the neighborhood of good points and therefore intensifies search in regions which contain good solutions. Gradient techniques are not employed. The algorithm itself consists of four nested loops (Algorithm 1, Figure 5).

ALGORITHM 1. The four nested loops of PGSL

FIG. 5. Example for the development of the probability density function of one optimization variable X_i during the four nested loops of PGSL

A feature that PGSL shares with other random search methods such as adaptive random search and controlled random search is the use of a PDF (Probability Density Function). However, the following differences between PGSL and other random methods are:

1. Other random methods that make use of an explicitly defined PDF, follow a "creep" procedure similar to simulated annealing. They aim for a point-to-point improvement by restricting search to a region around the current point. The PDF is used to search within a

small neighborhood. On the other hand, PGSL works by global sampling. There is no point-to-point movement.

2. The four nested cycles in PGSL are not similar to any features of other algorithms.
3. Representation of probabilities is different. Other methods make use of a mathematical function with a single peak (e.g. gaussian) for the PDF. PGSL uses a histogram - a discontinuous function with multiple peaks. This allows fine control over probabilities in small regions by subdividing intervals.
4. Probabilities are updated differently. The primary mechanism for updating probabilities in other methods is by changing the standard deviation. In PGSL, the entire shape and form of the PDF can be changed by subdividing intervals as well as by directly increasing probabilities of intervals.

The algorithm has been tested on non-linear benchmark problems and compared with results from genetic algorithms applied to the same problems (Raphael and Smith 2000). When no problem-specific knowledge is employed, PGSL performs as well as genetic algorithms.

3.3.2 Adjusting the parameters of PGSL

The parameters to be determined for PGSL are the number of iterations for each one of the four nested loops (see paragraph 3.3.1). For the detection of optimal parameters, the following procedure was employed. Drawing from experiences made with other optimization problems, the number of sampling cycles is set at two and the number of probability updating cycles at one. The number of iterations for the third loop should be fixed at P × number of variables, where P varies from 10 to 20. Values for the number of iterations in the subdomain cycle are determined by experiment. Different sets of parameters have been tested on the control problem. Table 2 presents the different parameters chosen to be tested on the control problem in combination with two representative load cases. Parameter set 1 did not use the empirical procedure described above.

TABLE 2. Test cases for parameter adjustment of PGSL. See Figure 7 for the description of load cases

The results are plotted in a best-so-far curve, where the cost of the objective function has been plotted against the time the algorithm used in Figure 6a) and 6b):

FIG. 6. Parameter study

Parameter set 4 has been used with the tests. The parameter study reveals that the procedure followed for sets 2, 3 and 4 results in the best solutions. Therefore only the values of focusing cycles (NFC) and subdomain cycles (NSDC) need to be adjusted to fix the total number of evaluations of the objective function. This underlines the ease and simplicity of fixing PGSL parameters.

4. Description of the tests

Tests focused on the comparison of two stochastic search techniques applied to the same structural control problem. As described in section 2.1, a tensegrity structure should respond to loads such that a given objective is reached. For a comparative study we are concentrating on a relative height objective. This is related to our working objective to keep the top nodes of the structure at a constant slope. An application of such an objective is to control the slope of a roof. Roofing systems require slope control to avoid ponding during rainfall and melting of snow. The objective function governing this search can be formulated as follows:

Find a set of possible bar strokes ($P_{i,\text{bar stroke}}$; i = number of moveable bars) such that

$$cost = \sqrt{\frac{(Node_{6,z} - Node_{52,z})^2 + (Node_{6,z} - Node_{62,z})^2 + (Node_{52,z} - Node_{62,z})^2}{3}} (mm)$$

is minimized according to a predefined threshold value.

($Node_{6,z}$ = z-displacement of controlled node number 6, see also Figure 7)

The constraints used were:

- Maximum bar stress ≤ 28.5 Mpa

- Maximum cable stress ≤ 901 Mpa (70% of the maximum cable stress allowed by the manufacturer)
- Maximum buckling force ≤ 20 kN

According to the technical specifications of the jacks used, the maximum move of the telescopic bars from their initial position is limited to ± 21mm. The precision range of each move is in steps of ± 0.25mm. Three series of tests have been carried out:

- **Series 1: Constrained scenario**

  Two bars per module were assumed to be telescopic and could be moved by ± 3mm.

  Optimization of seven different load cases.
- **Series 2: Search progression**

  Five bars per module were assumed to be telescopic and could be moved by ± 3mm.

  Optimization of seven different load cases.
- **Series 3: Effect of the number of search runs**

  Five bars per module

  Optimization of two different load cases.

The load cases and the controlled nodes are presented in Figure 7.

FIG. 7. Plan view of the tensegrity structure with load cases (LC) and controlled nodes used for the tests

4.1 Test results

A Pentium III 600 Mhz machine has been used for all tests. Results have been obtained by running each optimization method at least three times.

4.1.1 Series 1

Test results are presented in Table 3.

TABLE 3. Test results for 2 moveable bars per module

Although both algorithms required approximately 3 to 4 hours to complete search, the final solution was found after several minutes as noted in Table 3. The solutions obtained from simulated annealing and PGSL were identical regarding cost and proposed bar strokes. An optimal solution near a cost value of zero could not be obtained, since:

- only two telescopic bars per module were allowed to move,
- movement was limited to ± 3mm, and
- possible bar positions were only in steps of 0.25mm.

Upon examining the proposed bar strokes for the final solution it was observed that these tests identified solutions that are on the edge of the solution space since every bar was moved by its maximum stroke, either +3mm or –3mm. This signifies that the iteration might have been stopped after each bar was extended to its limit. Nevertheless, PGSL identified the best solution more rapidly than simulated annealing.

4.1.2 Series 2

Table 4 presents the number of iterations needed to attain the best state for each combination of load case and search technique.

TABLE 4. Test results for 5 moveable bars per module

The development of the costs during an optimization has been plotted in best-so-far curves as introduced in Section 3.3. The curve of the simulated annealing process shows three peaks. As a point-to-point search technique, it is launched three times from the initial conditions in order to allow three different paths to converge to a near optimal solution. For PGSL, in contrast, this is less advantageous since it is inherently a parallel technique. The number of evaluation cycles has been increased proportionally for each PGSL run.

Figure 8a to 8g provide results from seven different load cases that now can be compared with respect to minimum cost and speed of convergence. PGSL shows faster convergence in load

case 1 (Figure 8a) as simulated annealing. Nevertheless, both algorithms converge to the same best cost. This leads to the conclusion that this solution is most likely the global minima.

Final results of load case 2 (Figure 8b) are close to each other. Although simulated annealing converges to the best result at the end, PGSL converges faster in the beginning of the iterative cycle. For load case 3 (Figure 8c) PGSL performs better than simulated annealing in terms of best cost and speed of convergence. Simulated annealing converges to the best solution in load case 4 (Figure 8d). Observations for load case 5 (Figure 8e) are similar to those made for load case 3. However, a zone between approximately 3000 and 21000 evaluations is present, where simulated annealing outperforms PGSL. Load cases 6 and 7 (Figures 8f and 8g) may be discussed together since these results are analogous. PGSL converges faster for the first approximately 2000 evaluations of the objective function. In the middle of the evaluation simulated annealing provided better solutions than PGSL. Although PGSL found the set of best bar movements, the differences in cost are negligible.

FIG. 9. Best-so-far curves

4.2 Effect of the number of runs

Stochastic search techniques do not necessarily converge to the same solution when started multiple times for the same objective and initialstate. Therefore, multiple runs have been executed for two load cases (load case 2 and load case 6) to evaluate the effect of the number of runs.

Figure 9a shows the results for 25 runs for load case 2. After two runs a solution in the region of the best solution has already been found. Both algorithms converge to the same solution after eleven runs.

Load case 6 (Figure 9b) is a more complex problem. The lowest cost curve for simulated annealing shows a more “staircase-like” behavior. Although after 6 runs no further changes in the lowest cost of simulated annealing can be observed, PGSL finds a better solution close to

the best state of simulated annealing after 15 runs. In all cases, acceptable costs were achieved after two to four runs, considering practical aspects of applying solutions to the structure. This is discussed further in the next section.

FIG. 9. Best-so-far curves for multiple runs
Bar movements proposed differed significantly for almost all runs.

4.3 Discussion

The first two test series showed the ability of both stochastic search methods to find good control solutions. As shown in section 4.1.2, there was little evidence that one method provides more accurate results than the other. No best algorithm for all test load cases can be identified. These results thus support the “no-free-lunch” theorem (Wolpert and Mcready 1997). PGSL converges faster in the approximately first 1000 evaluations of the objective function. During that period, simulated annealing accepts still worse solutions to avoid local minima. This behavior leads to a better end result in some cases. Nevertheless, from a practical viewpoint, both algorithms provide good results.

The success of this search does not necessarily require solutions that are near to 0. As it was observed in section 4.1.1, it may not be possible to counteract completely all deflections within the constraints of this task. Furthermore, the usual inaccuracies between behavioral models and real behavior often do not justify the computational cost of a theoretically better solution. Such tradeoffs help determine the most appropriate values required for accuracy.

Evidence of the stochastic nature of both algorithms has been given with test series 3, presented in section 4.2. Although costs were similar, command characteristics were different for almost each run. The time necessary to evaluate multiple runs for determination of good control moves inhibits the practical use of search methods even for quasi-static control. Solutions with similar values of the objective function propose significantly different bar

movements. This has an impact on their applicability since current actuator positions might lead to different choices.

PGSL has an advantage over simulated annealing in terms of the number of parameters to be fixed before each optimization process. Simple guidelines lead to rapid parameter adaptation for other applications.

As a step-wise method, simulated annealing allows movements to control solutions that violate constraints, which are currently rejected. Soft constraints can be used in order to find an optimum by stepping through a region of invalid solutions. Since PGSL is a probabilistic method that focuses on good solutions in parallel, it does not iterate from invalid solutions. The approach used for finding good solutions thus differs between methods. This difference may determine which algorithm is better suited for a given application.

Once good control commands have been used successfully, they may be stored in a case base for subsequent use. This would increase speed significantly. Furthermore, case adaptation techniques may prove to be useful when previous cases do not exactly match current control tasks. This is a focus of work in progress.

5. Conclusions

Stochastic search results show much potential for controlling deflections in highly coupled tensegrities. Simulated Annealing and PGSL provide rapid convergence to good solutions in this application. PGSL has the advantage that control parameters are fewer and more intuitive. The most rapid technique depends on the desired accuracy of the objective. PGSL usually provides good solutions for high and low required accuracies whereas simulated annealing offers better results for intermediate cases.

Evaluating multiple runs for one control task resulted in gradually more accurate results. However, the time that is necessary for solution diversity in order to perform multi-criteria optimization might limit this advantage.

While no one tuning of an algorithm can be successful for all tasks, as stated by the "no-free-lunch" theorem, simulated annealing and PGSL perform well for highly coupled computational shape control of structures. PGSL has the advantage of fewer search parameters that require setting prior to search.

6. Acknowledgements

This research is partially funded by the Swiss National Science Foundation. We would like to thank Etienne Fest for valuable comments and Raymond Délez for technical suggestions. We would also like to thank Maag Technic for supporting this research.

7. References

Cagan, J., Degentesh, D, Yin, S. (1998). "A simulated annealing-based algorithm using hierarchical models for general three-dimensional component layout", *Computer-Aided Design*, Vol. 30, No. 10, Elsevier science, 781-790.

Calladine, C. R. (1978). "Buckminster Fuller's "Tensegrity" Structures and Clerk Maxwell's Rules for the Construction of Stiff Fames", *Int. J. Solids Structures*, Vol. 14, 161-172.

Ceranic, B., Fryer, C., Baines, R. W. (1999). "An application of simulated annealing to the optimum design of reinforced concrete retaining structures", in: Topping, B.H.V. And Kumar, B. (eds.): *Optimization and Control in Civil and Structural Engineering*, Civil-Comp Press, Edinburgh, 47-53.

Chattopadhyay A., Seely, C. E. (1994)."A simulated annealing technique for multiobjective optimization of intelligent structures", *Smart Materials Structures 3*, IOP Publishing, 98-106.

Deb, K., Gulati, S. (2001). "Design of truss-structures for minimum weight using genetic algorithms", *Finite Elements in Analysis and Design*, Elsevier science, 447-465.

Dowsland, K. (1995). "Simulated Annealing", in: Reeves, C. (ed.): *Modern heuristic techniques for combinatorial problems*, McGraw-Hill, 20-69.

El-Beltagy, M. A., Keane, A. J. (1999). "A comparison of various optimization algorithms on a multilevel problem", *Engineering Applications of Artificial Intelligence 12*, Elsevier science, 639-654.

Fest, E., Shea, K., Domer, B., Smith, I.F.C. (2001). "Adjustable Tensegrity Structures", *Journal of Structural Engineering, American Society of Civil Engineers*, (submitted 2001).

Fuller, B. (1962). "Tensile Integrity Structures", U.S. Pat. 3,063,521.

Goldberg D. (1989). "Genetic Algorithms in Search, Optimization and Machine Learning", Reading, MA: Addison Wesley.

Gremling, J. R., Passino, K. M. (2000). "Genetic adaptative state estimation", *Engineering Applic. Artif. Intell. Vol. 13*, Elsevier science, pp. 611-623.

Grierson, D. E., Khajehpour, S. (1999). “Multi-Criteria Conceptual Design of Office Buildings Using Adaptive Search”, *Proceedings of the 6th EG-SEA-AI workshop*, Wydawnictwa Naukowo-Techniczne, Warszawa, 51-74.

Han, J. H., Lee, I. (1999). “Optimal placement of piezoelectronic sensors and actuators for vibration control of a composite plate using genetic algorithms”, *Smart Materials Structures 8*, IOP Publishing, 257-267.

Hustin, S. (1988). “Tim, A New Standard Cell Placement Program Based on the Simulated Annealing Algorithm,” *Master of Science*, University of California, Berkeley, Department of Electrical Engineering and Computer Science.

Kirkpatrick, S., Gelatt, C. D., Vecchi, P. (1983). “Optimization by Simulated Annealing”, *Science Volume 220*, Number 4598, 671-680.

Khot, N. S. (1998): “Multicriteria Optimization for Design of Structures with Active Control”, *Journal of Aerospace Engineering 11 (2)*, ASCE Publication, pp. 45-51.

Kristinsson, K., Dumont, G. A. (1992). “System Identification and Control Using Genetic Algorithms”, *IEEE Transactions on Systems, Man and Cybernetics*, Vol. 22, No. 5, IEEE Press, pp. 1033-1046.

Masri, S. F., Bekey, G. A., Safford, F. B. (1980). “A global optimization algorithm using adaptive random search”, *Applied mathematics and computation*, Vol. 7, 353-375.

Mayhan, P., Washington, G. (1998). “Fuzzy model referencing learning control: a new control paradigm for smart structures”, *Smart Materials Structures 7*, IOP Publishing, 874-884.

Motro, R. (1992). “Tensegrity Systems: The State of the Art”, *International Journal of Space Structures*, Vol. 7, No. 2, 75-83.

Nishimura, I., Yamada, T., Sakamoto, M., Kobori, T. (1992). “Active tuned mass damper”, *Smart Materials Structures 1*, IOP Publishing, 306-311.

Parmee, I. C., Cvetkovic, D., Bonham, C., Packham, I. (2001). “Introducing prototype interactive evolutionary systems for ill-defined, multi-objective design environments”, *Advances in Engineering Software 32*, Elsevier science, pp. 429-441.

Passino, K. M. (1996). “Toward bridging the gap between conventional and intelligent control”, in: *Intelligent control systems*, M. M. Gupta and N. K. Sinha (eds.), IEEE press, 3-27.

Perelli, Y. (2000). “Comportement des structures Tenségrités”, *Master Thesis*, EPFL, unpublished.

Pugh, A. (1976). “An introduction to tensegrity”, University of California Press Berkeley.

Raphael, B., Smith, I. F. C (2000). “A probabilistic search algorithm for finding optimally directed solutions”, *Proceedings of Construction Information Technology*, Icelandic Building Research Institute, Reykjavik, 708-721.

Reeves, C., R., Beasly, J. E (1995). “Introduction”, in: Reeves, C. (ed.): Modern heuristic techniques for combinatorial problems, McGraw-Hill, 1-19.

Robert-Nicoud, Y., Raphael, B., Smith, I.F.C. (2000). “Decision support through multiple models and probabilistic search, Proceedings: Construction Information Technology 2000, Icelandic Building Research Institute, 765-779.

Rossier, S. (1994). “Optimization of Cable Structures Using Genetic Algorithms”, *Master Thesis*, Herriot Watt University/EPFL, unpublished.

Salama, M., Umland, J., Bruno, R.,Garba, J. (1993). “Shape adjustment of precision truss structures: analytical and experimental validation”, *Smart Materials and Structures 2*, Elsevier science, 240-248.

Swartz W., and C. Sechen (1990). "New Algorithms for the Placement and Routing of Macro Cells", IEEE *proceedings: Cat No. 90CH2924-9*, IEEE Conference on Computer-Aided Design, Santa Clara, CA, November 11-15, 336-339.

Shea, K., Cagan, J. and Fenves, S. J. (1997). “A shape annealing approach to optimal truss design with dynamic grouping of members.”, *ASME Journal of Mechanical Design*, Vol.119, (3), pp388-394.

Shea, K., Fest, E. and Smith, I.F.C. (2001). “Developing Intelligent Structures with Stochastic Search”, *Advanced Engineering Informatics*, Elsevier (accepted).

Shea, K., Smith, I.F.C. (1999). "Applying shape annealing to full-scale transmission tower re-design", *Proceedings of DETC99, 1999 ASME Design Engineering Technical Conference*, September 12-15 1999, Las Vegas, NV.

Svanerudh, P., Raphael, B., Smith, I.F.C. (2001). "Lowering costs of timber shear-wall design using global search", *ASCE Journal of Structural Engineering*, (submitted 2001).

Williamson, A.,Skelton, R. (1998). "A General Class of Tensegrity Systems", *Proceedings: Engineering Mechanics for the 21st Century*, ASCE Conference, La Jolla, California, May 17-20.

Wolpert, D. H., Macready, W. G. (1997). "No Free Lunch Theorems for Optimization", *IEEE Transactions on Evolutionary Computation*, Vol. 1, No. 1, IEEE Press, pp. 67-82.

Youssef, H., Sait, M. S., Adiche, H. (2001). "Evolutionary algorithms, simulated annealing and tabu-search: a comparative study", *Engineering Applications of Artificial Intelligence 14*, Elsevier science, 167-181.

Tables

TABLE 1. Moore's law applied to the control problem

Number of modules	time needed to evaluate all possible solutions		
	Today	In 5 years (10 times faster)	In 20 years (10000 times faster)
	[years]	[years]	[years]
3	4.80E+12	4.80E+11	4.80E+06
7	1.93E+40	1.93E+39	1.93E+34
15	3.12E+95	3.12E+94	3.12E+89
20	9.99E+129	9.99E+128	9.99E+123
25	3.20E+164	3.20E+163	3.20E+158

TABLE 2. Test cases for parameter adjustment of PGSL. See Figure 7 for the description of load cases

Parameter	Set			
	1	2	3	4
Number of sampling cycles (NSC)	5	2	2	2
Number of probability updating cycles (NPUC)	3	1	1	1
Number of focusing cycles (NFC)	150	150	300	150
Number of subdomain cycles (NSDC)	8	10	15	15

TABLE 3. Test results for 2 moveable bars per module

Load Case	Simulated annealing		PGSL	
	Cost [mm]	Time [mm:ss]	Cost [mm]	Time [mm:ss]
1	9,8096	25:14	9,8096	06:16
2	3,3600	17:51	3,3600	04:44
3	6,1594	15:57	6,1594	07:44
4	1,1591	13:28	1,1591	05:32
5	6,3735	08:12	6,3735	06:26
6	1,4448	13:07	1,4448	07:37
7	3,1963	13:06	3,1963	07:29

TABLE 4. Test results for 5 moveable bars per module

Load Case	Simulated annealing		PGSL	
	Cost [mm]	Iteration	Cost [mm]	Iteration
1	1.4101	16.287	1,4101	1.700
2	0.0073	23.781	0.0085	34.428
3	0.0106	51.888	0.0054	30.000
4	0.0024	11.779	0.0075	16.316
5	0.0171	21.000	0.0050	21.628
6	0.0071	1.670	0.0050	37.842
7	0.0051	45.781	0.0029	37.811

Figures

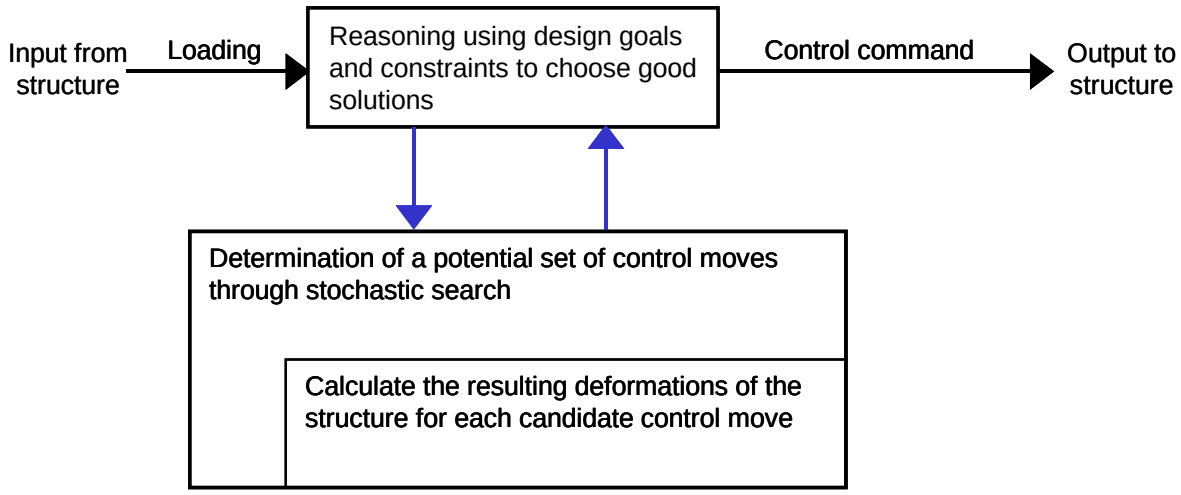


FIG. 1. Finding control commands through stochastic search

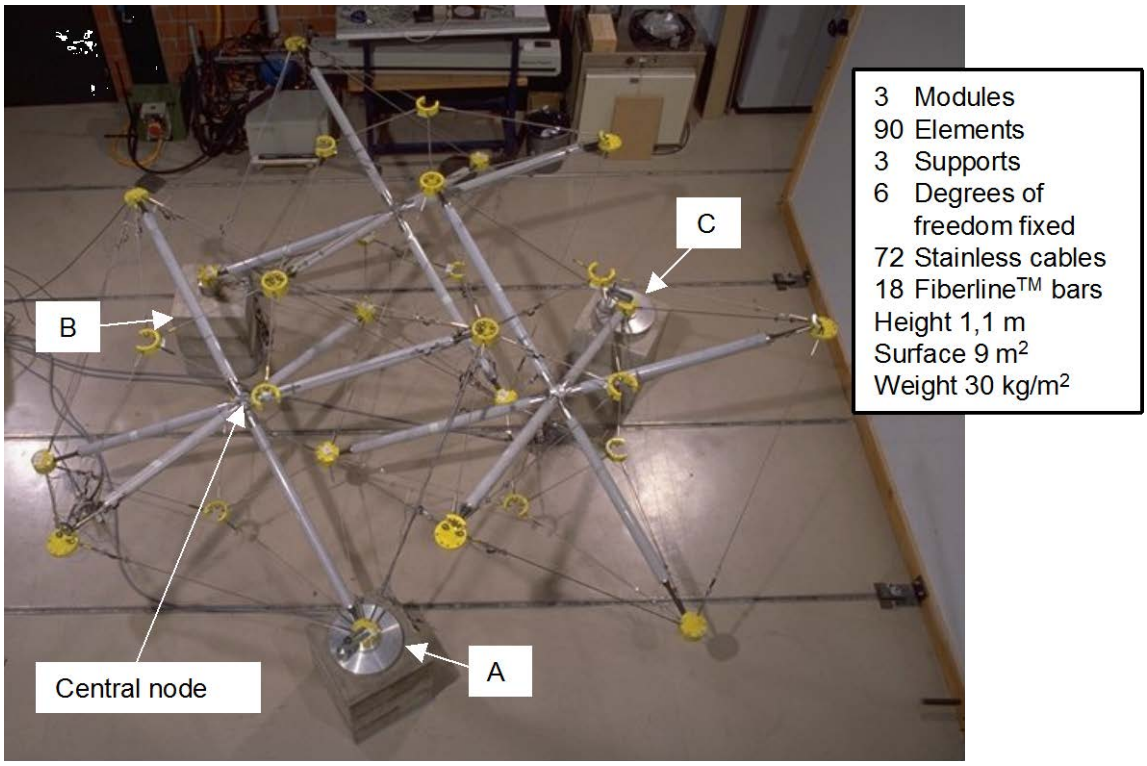


FIG. 2. Tensegrity structure built from three modules (A, B, and C are supports)

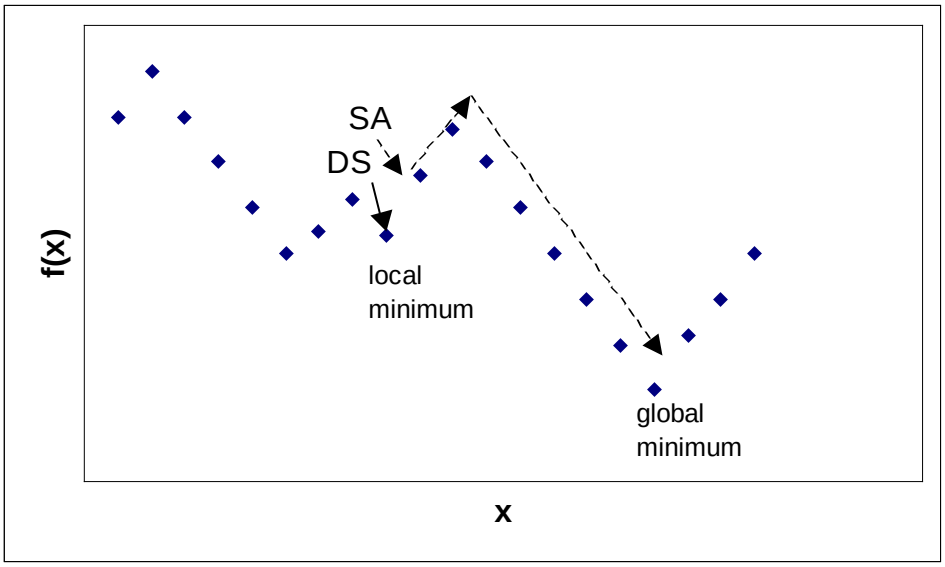


FIG. 3. Schematic comparison of simulated annealing (SA) and a descent strategy (DS) (modified from Dowsland 1995). f(x) is the objective function.

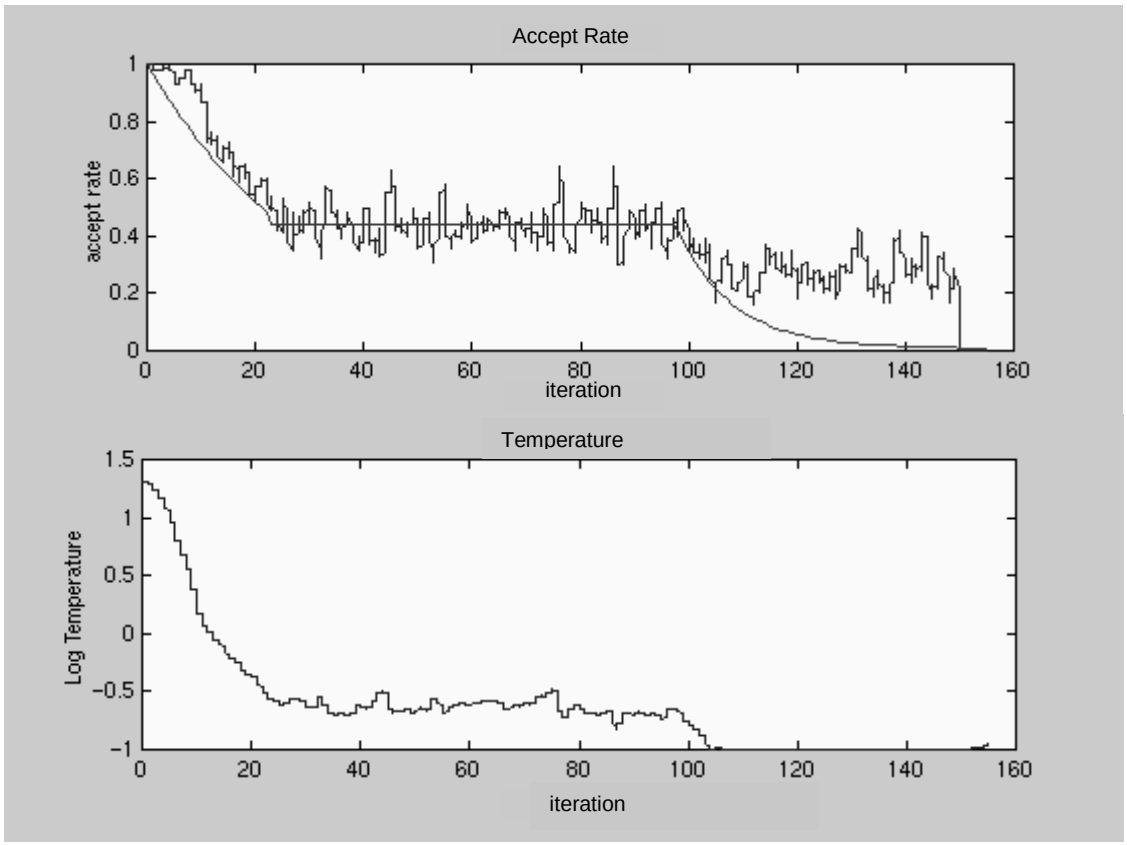


FIG. 4. Examples of accept rate and temperature schedules

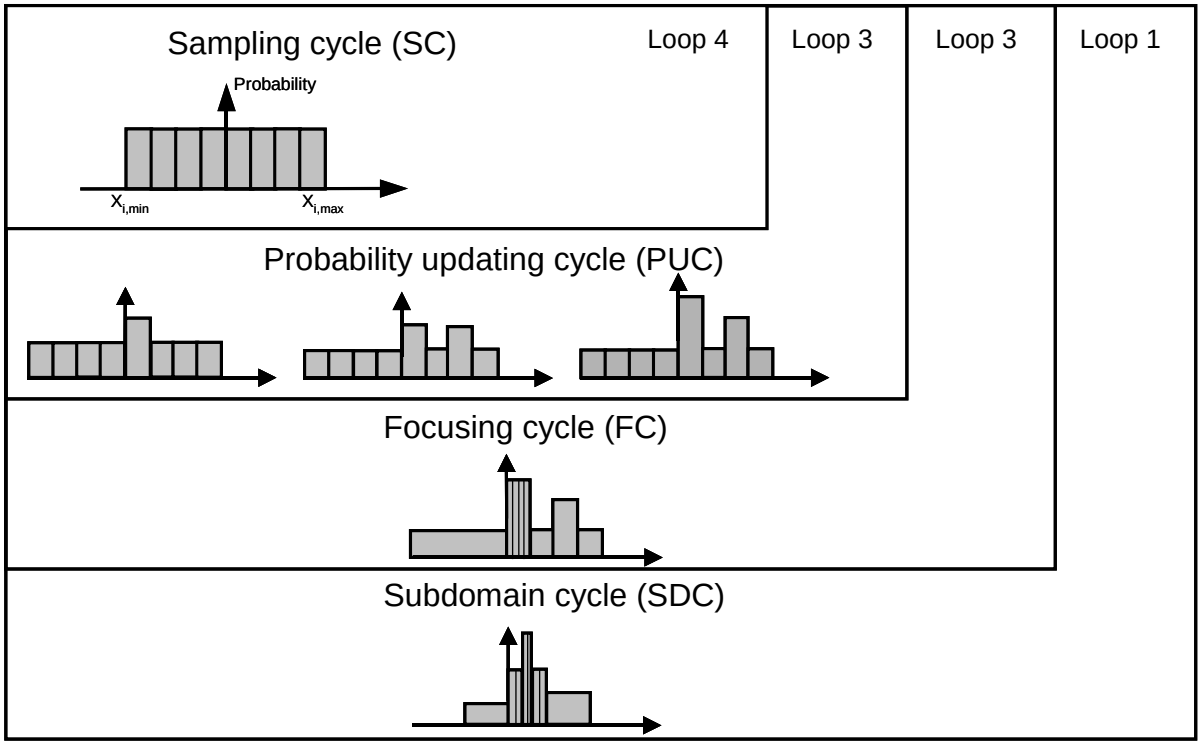
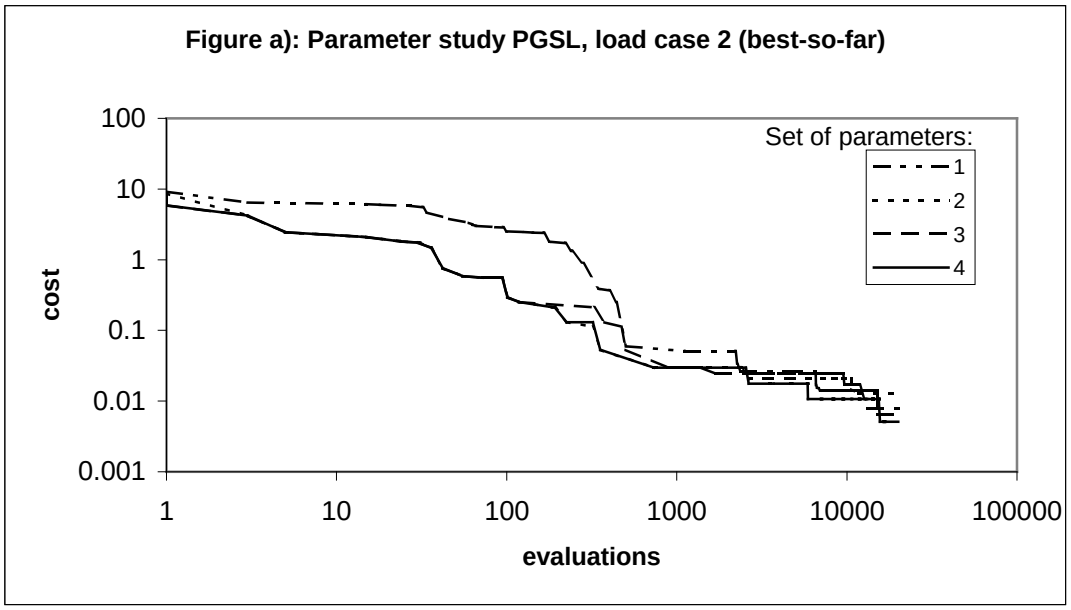


FIG. 5. Example for the development of the probability density function of one optimization variable X_i during the four nested loops of PGSL



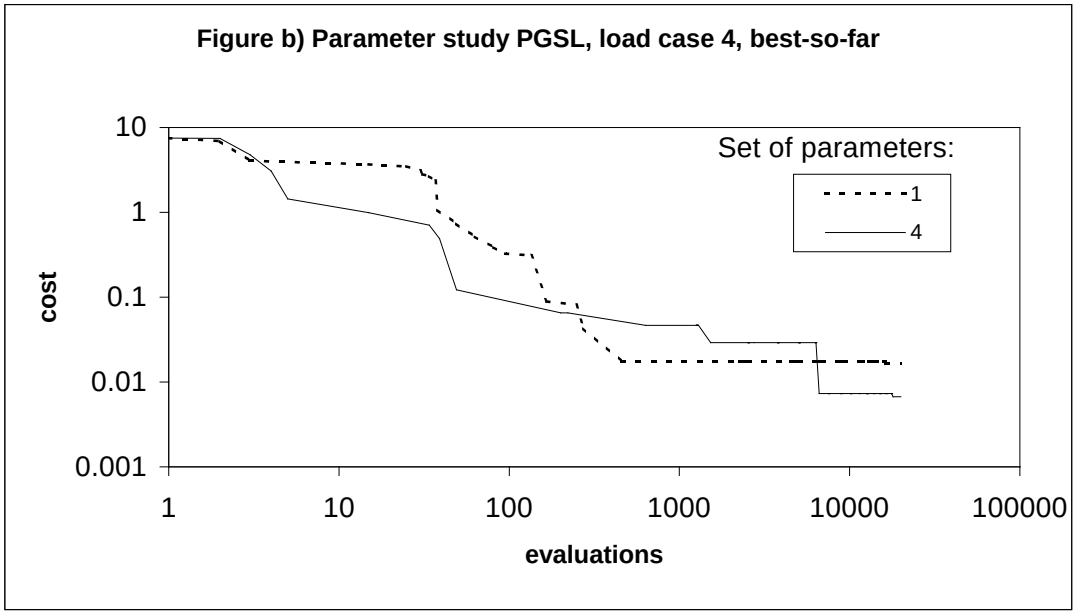


FIG. 6. Parameter study

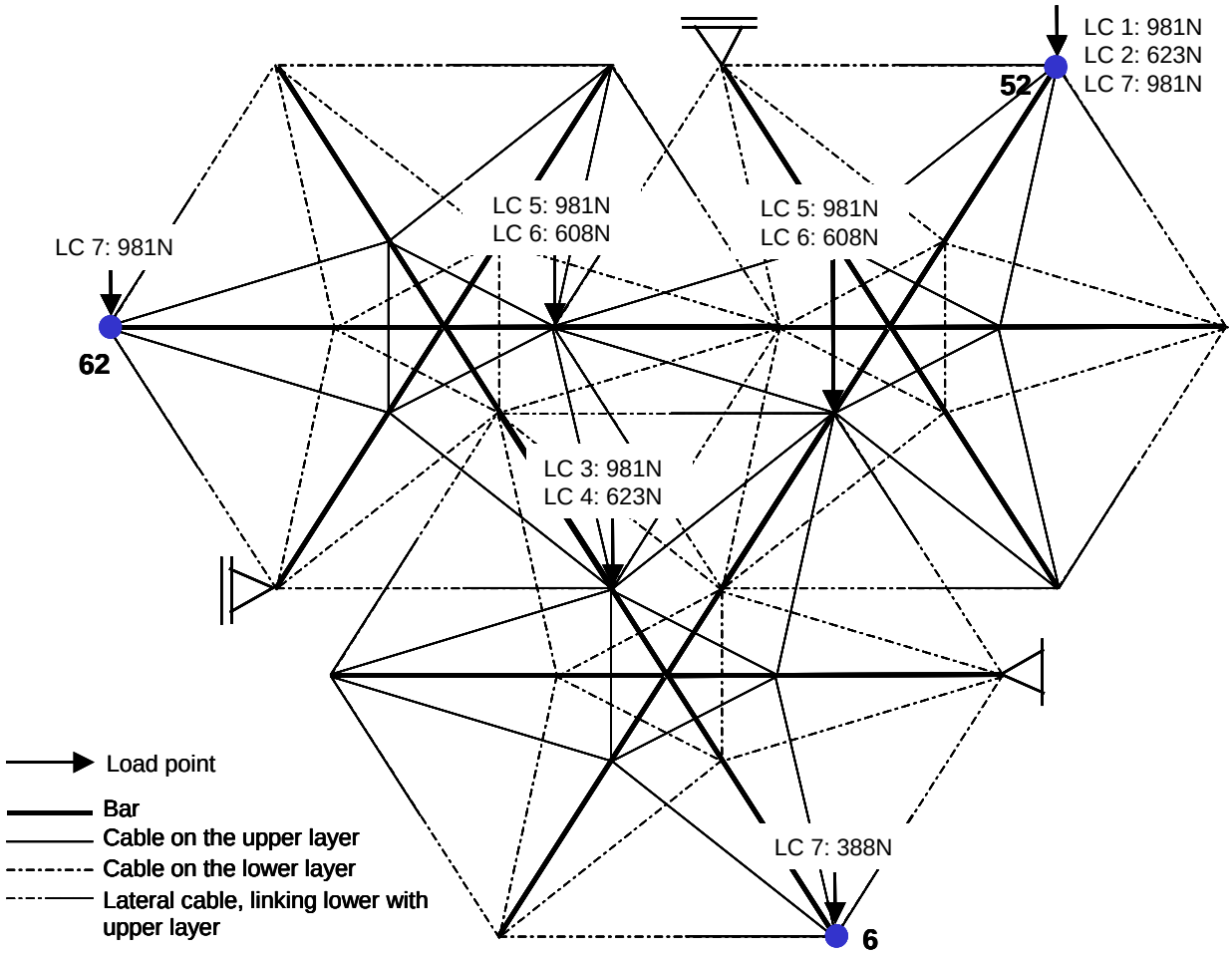


FIG. 7. Plan view of the tensegrity structure with load cases (LC) and controlled nodes used for the tests

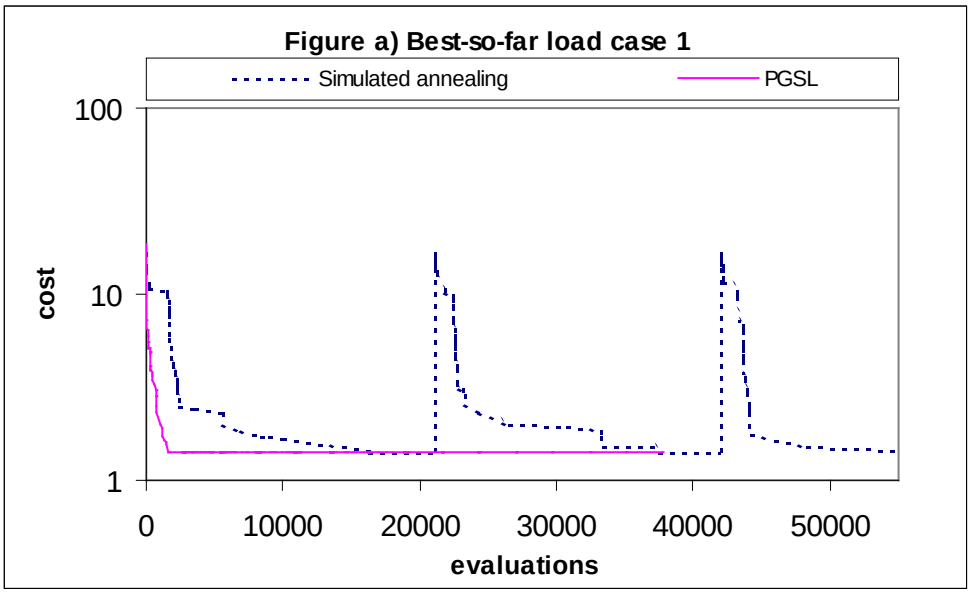


Figure a) Best-so-far load case 1
Simulated annealing
PGSL
100
10
1
cost
0
10000
20000
30000
40000
50000
evaluations

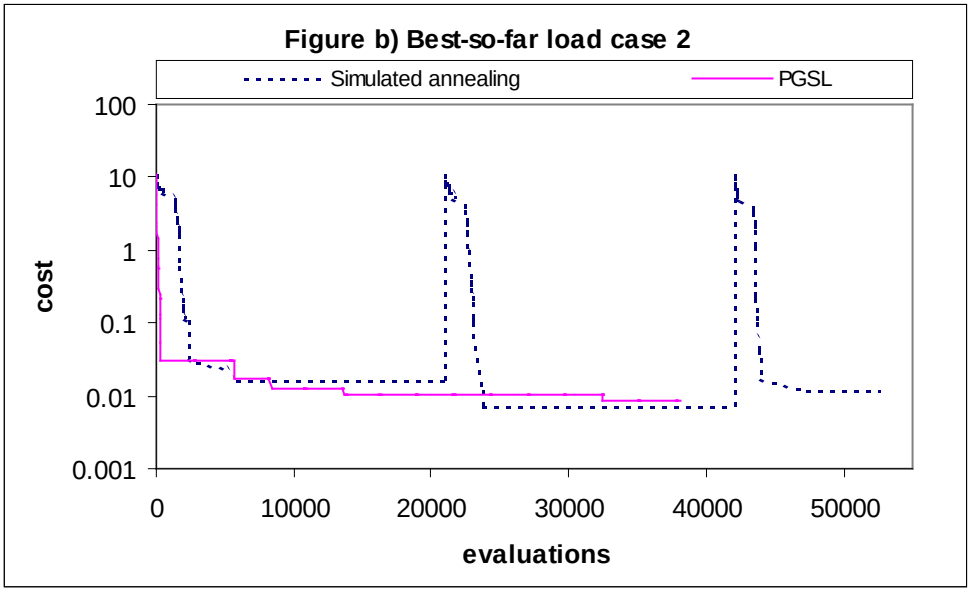


Figure b) Best-so-far load case 2
Simulated annealing
PGSL
100
10
1
0.1
0.01
0.001
cost
0
10000
20000
30000
40000
50000
evaluations

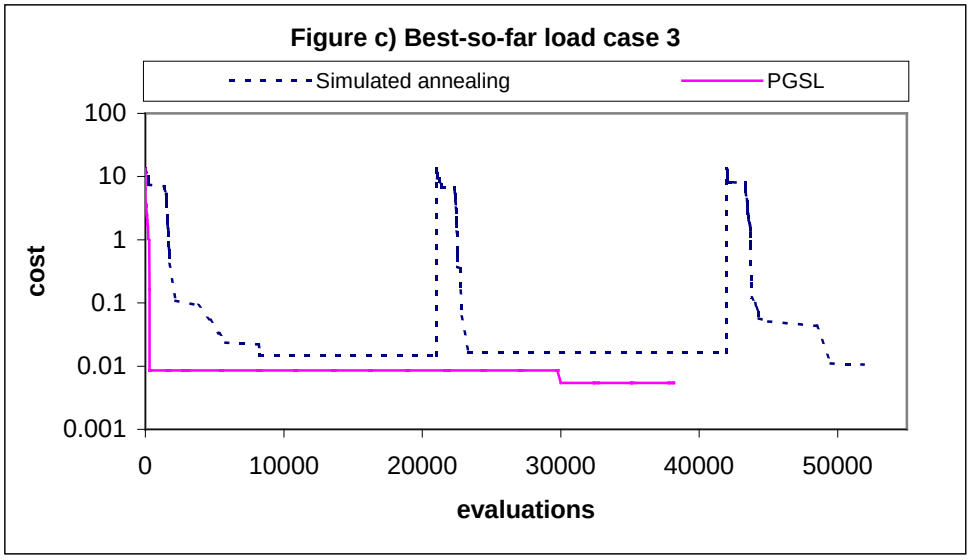


Figure c) Best-so-far load case 3
Simulated annealing
PGSL
100
10
1
0.1
0.01
0.001
cost
0
10000
20000
30000
40000
50000
evaluations

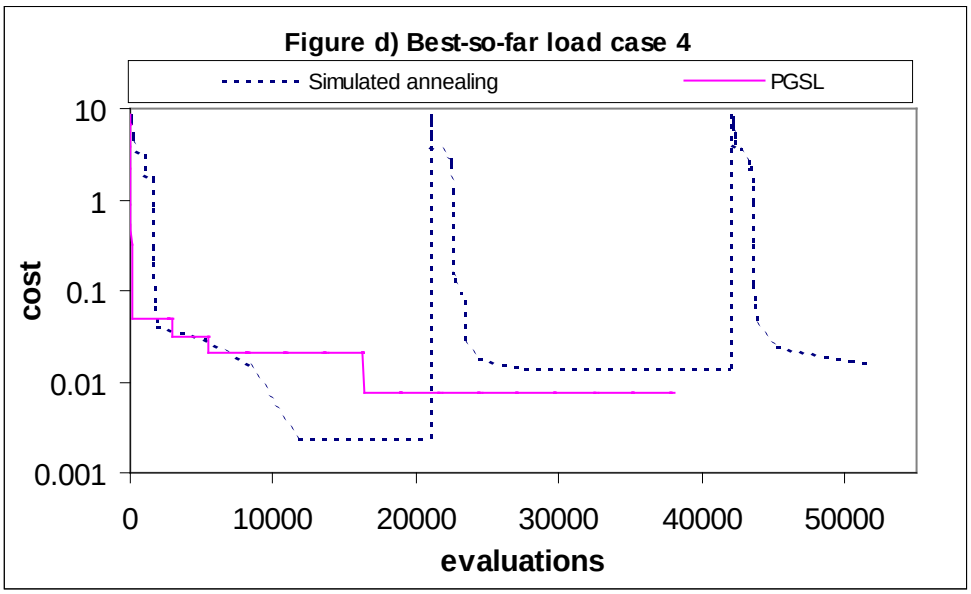


Figure d) Best-so-far load case 4
Simulated annealing
PGSL
10
1
0.1
0.01
0.001
cost
0
10000
20000
30000
40000
50000
evaluations

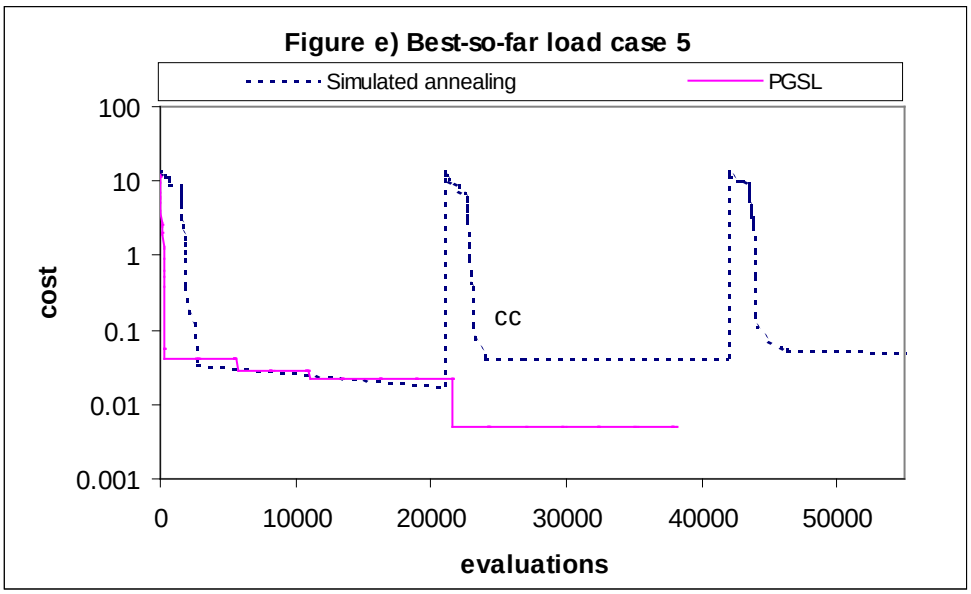
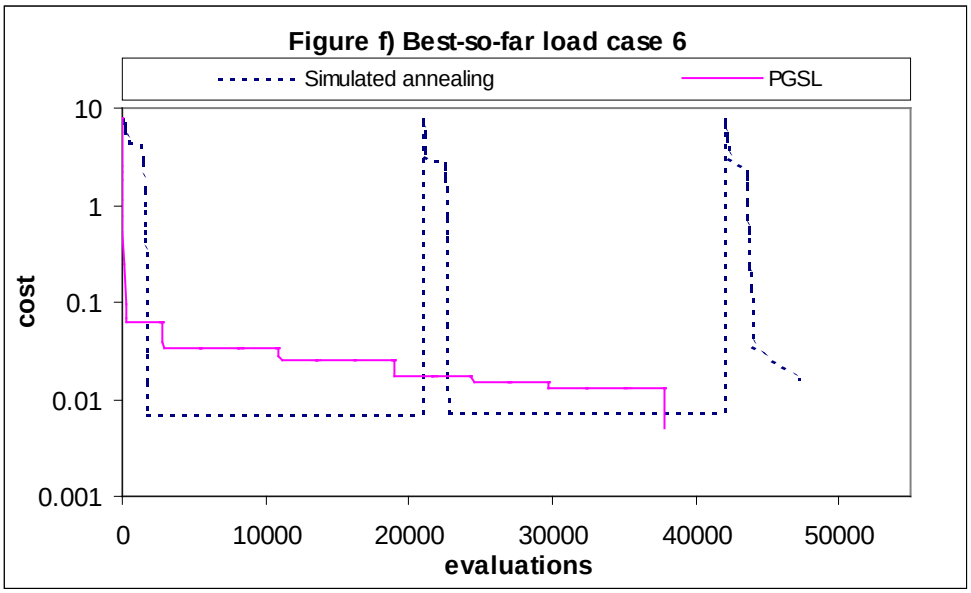


Figure e) Best-so-far load case 5
Simulated annealing
PGSL
100
10
1
0.1
0.01
0.001
cost
cc
0
10000
20000
30000
40000
50000
evaluations



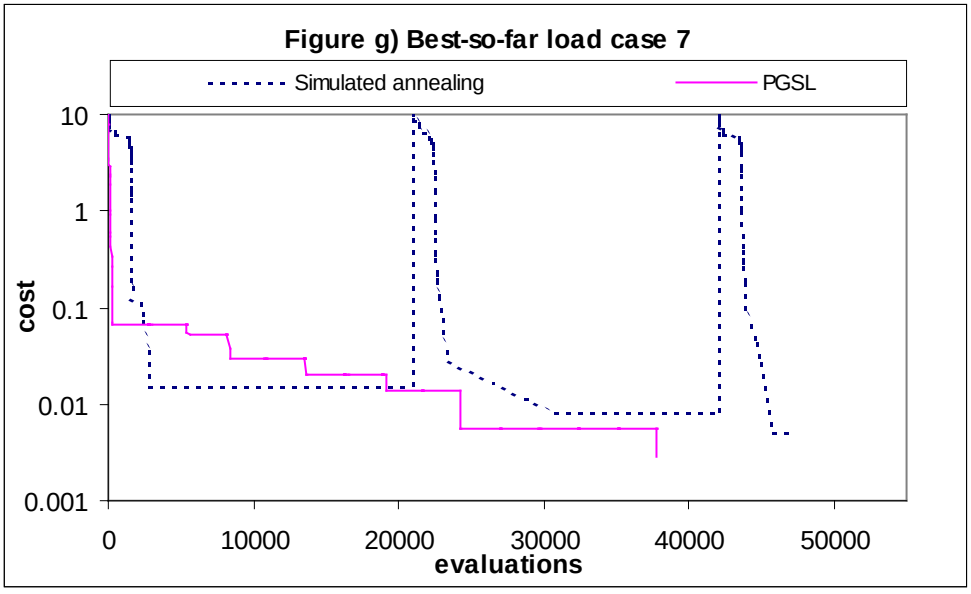


FIG. 8. Best-so-far curves

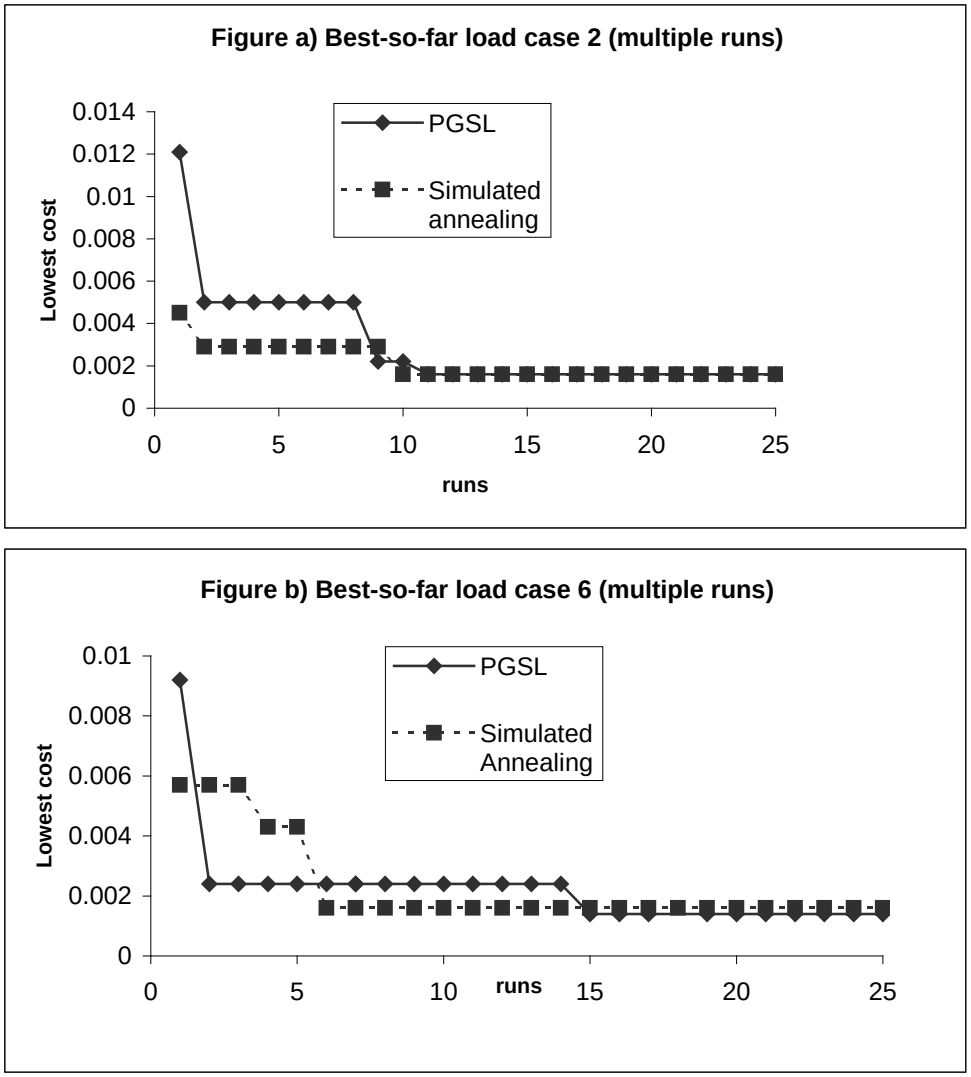


FIG. 9. Best-so-far curves for multiple runs

Algorithms

```
Set the complete search space as the current subdomain
Loop 1: Repeat for NSDC (Number of Sub-Domain Cycles) iterations
        assume a uniform probability density function (PDF) for all
          variables in the current subdomain.

   Loop 2: Repeat for NFC (Number of Focusing Cycles) iterations

      Loop 3: Repeat for NPUC (Number of Probability Updating Cycles)
              iterations

         Loop 4:  Repeat for NSC (Number of Sampling Cycles)iterations
                  Generate a solution using the current PDF
         End of Loop 4

         Select the best solution in Loop 4.
         For each variable, locate the interval containing the best value.
         Increase the probability of this interval.

      End of Loop 3:
      Select the best solution in Loop 3: Subdivide the interval containing
      the best solution.
      Assume a uniform probability within the best interval.
      Assume an exponentially decreasing distribution away from the best
      interval.

   End of Loop 2:

   Select a smaller subdomain centred around the best solution so far.
   The width of this subdomain is chosen after performing certain checks to
   prevent premature convergence (details can be found in [Raphael and
   Smith, 2000])

End of Loop 1.
```

ALGORITHM 1. The four nested loops of PGSL