SOLVING ENGINEERING OPTIMIZATION PROBLEMS WITH THE SIMPLE CONSTRAINED PARTICLE SWARM OPTIMIZER

Leticia C. Cagnina and Susana C. Esquivel
LIDIC (Research Group)-Universidad Nacional de San Luis
Ej. de los Andes 950. (D5700HHW) San Luis, Argentina.
lcagnina,esquivel@unsl.edu.ar

Carlos A. Coello Coello*
CINVESTAV-IPN (Evolutionary Computation Group)
Av. IPN 2508. (07300) Mexico D. F., Mexico
ccoello@cs.cinvestav.mx

Abstract This paper introduces a particle swarm optimization algorithm to solve constrained engineering optimization problems. The proposed approach uses a relatively simple method to handle constraints and a different mechanism to update the velocity and position of each particle. The algorithm is validated using four standard engineering design problems reported in the specialized literature and it is compared with respect to algorithms representative of the state-of-the-art in the area. Our results indicate that the proposed scheme is a promising alternative to solve this sort of problems because it obtains good results with a low number of objective function evaluations.

Keywords: Particle Swarm Optimization, Engineering problems, Constrained Optimization

1. Introduction

Engineering design optimization problems are normally adopted in the specialized literature to show the effectiveness of new constrained optimization algorithms. These nonlinear engineering problems have been investigated by many researchers that used different methods to solve

*The third author acknowledges support from CONACyT project no. 45683-Y.

them: Branch and Bound using SQP [15], Recursive Quadratic Programming [13], Sequential Linearization Algorithm [7], Integer-discrete-continuous nonlinear Programming [9], Nonlinear mixed-discrete Programming [8], Simulated Annealing [4], Genetic Algorithms [5], Evolutionary Programming [11] and, Evolution Strategies [6] among many others. These types of problems normally have mixed (e.g., continuous and discrete) design variables, nonlinear objective functions and nonlinear constraints, some of which may be active at the global optimum. Constraints are very important in engineering design problems, since they are normally imposed on the statement of the problem and sometimes are very hard to satisfy, which makes the search difficult and inefficient.

Particle Swarm Optimization (PSO) is a relatively recent bio-inspired metaheuristic, which has been found to be highly competitive in a wide variety of optimization problems. However, its use in engineering optimization problems and in constrained optimization problems, in general, has not been as common as in other areas (e.g., for adjusting weights in a neural network). The approach described in this paper contains a constraint-handling technique as well as a mechanism to update the velocity and position of the particles, which is different from the one adopted by the original PSO.

This paper is organized as follows. Section 2 briefly discusses the previous related work. Section 3 describes in detail our proposed approach. Section 4 presents the experimental setup adopted and provides an analysis of the results obtained from our empirical study. Our conclusions and some possible paths for future research are provided in Section 5.

2. Literature Review

Guo et al. [2] presented a hybrid swarm intelligent algorithm with an improvement in global search reliability. They tested the algorithm with two of the problems adopted here (E02 and E04). Despite their claim that their algorithm is superior for finding the best solutions (in terms of quality and robustness), the solution that they found for E02 is greater than its best known value and for E04 the results obtained are not comparable to ours, because they used more constraints in the definition of that problem.

Shamim et al. [1] proposed a method based on a socio-behavioral simulation model. The idea behind this approach is that the leaders of all societies interact among themselves for the improvement of the society. They tested their using three of the problems adopted here (E01, E02 and E03). The best values reported for these three problems are close

from the optimal known values. The number of fitness function evaluations was 19,259 for E01, 19,154 for E02 and 12,630 for E03. Mahdavi et al. [3] developed an improved harmony search algorithm with a novel method for generating new solutions that enhances the accuracy and the convergence rate of the harmony search. They used three of the problems adopted here (E01, E03 and E04) to validate their approach, performing 300,000, 200,000 and 50,000 evaluations, respectively. For E01 and E02, the best values reported are not the best known values because the ranges of some variables in E01 are different from those of the original description of the problem (x_4 is out of range), which makes such solution infeasible under the description adopted here. The value reported by them for E04 is very close to the best value known.

Bernardino et al. [12] hybridized a genetic algorithm embedding an artificial immune system into its search engine, in order to help moving the population into the feasible region. The algorithm was used to solve four of the test problems adopted here (E01, E02, E03 and E04), using 320,000, 80,000, 36,000 and 36,000 evaluations of the objective functions, respectively. The best value found for E01 is close from the best known; for E02 and E04, the best values obtained are close to the best known, but for E03 the value reported is better than the best known, because one of the decision variables is out of range (x_5). The values in general, are good, although the number of evaluations required to obtain them is higher than those required by other algorithms.

Hernandez Aguirre et al. [24] proposed a PSO algorithm with two new perturbation operators aimed to prevent premature convergence, as well as a new neighborhood structure. They used an external file to store some particles and, in that way, extend their life after the adjustment of the tolerance of the constraints. The authors reference three algoritms which obtained good results for the problems adopted in their study: two PSO-based algorithms and a Differential Evolution (ED) algorithm. One of the PSO-based approaches compared [26] used three of the problems adopted here (E01, E02 and E04), performing 200,000 objective function evaluations. The other PSO-based approach compared [10] was tested with the same set of problems and the best known values were reached for E02 and E04 after 30,000 objective function evaluations. The ED algorithm [27] reported good results with 30,000 evaluations for the four problems. This same number of evaluations was performed by the algorithm proposed by Hernandez et al. [24] and their results are the best reported until now for the aforementioned problems. For that reason, we used these last two algorithms to compare the performance of our proposed approach (the best values reached are listed below). The

ED algorithm will be referenced as "Mezura" and, the PSO by [24] as "COPSO".

3. Our proposed approach: SiC-PSO

The particles in our proposed approach (called *Simple Constrained Particle Swarm Optimizer*, or SiC-PSO), consists of n-dimensional values (continuous, discrete or a combination of both) vectors, where n refers to the number of decision variables of the problem to be solved. Our approach adopts one of the most simple constraint-handling methods currently available. Particles are compared by pairs: 1) if the two particles are feasible, we choose the one with a better fitness function value; 2) if the two particles are infeasible, we choose the particle with lower infeasibility degree; 3) if one particle is feasible and the other is infeasible, we choose the feasible one. This strategy is used when the *pbest*, *gbest* and *lbest* particles are chosen. When an individual is found infeasible, the amount of violation (this value is normalized with respect to the largest violation stored so far) is added. So, each particle saves its infeasibility degree reached until that moment.

As in the basic PSO [18], our proposed algorithm records the best position found so far for each particle (*pbest* value) and, the best position reached by any particle into the swarm (*gbest* value). In other words, we adopt the *gbest* model. But so far, we found that the *gbest* model tends to converge to a local optimum very often [19]. Motivated by this, we proposed a formula to update the velocity, using a combination of both the *gbest* and the *lbest* models [20]. Such a formula (equation 1) is adopted here as well. The *lbest* model is implemented using a ring topology [22] to calculate the neighborhoods of each particle. For a size of neighborhood of three particles and a swarm of six particles (1,2,3,4,5,6), the neighborhoods considered are the following: (1,2,3), (2,3,4), (3,4,5), (4,5,6), (5,6,1) and (6,1,2). The formula for updating particles is the same that in the basic PSO and it is shown in equation 2.

$$v_{id} = w(v_{id} + c_1 r_1(pb_{id} - p_{id}) + c_2 r_2(pl_{id} - p_{id}) + c_3 r_3(pg_d - p_{id})) \quad (1)$$

$$p_{id} = p_{id} + v_{id} \quad (2)$$

where v_{id} is the velocity of the particle i at the dimension d, w is the inertia factor [18] whose goal is to balance the global exploration and the local exploitation, c_1 is the personal learning factor, and c_2, c_3 are the social learning factors, r_1, r_2 and r_3 are three random numbers within the range [0..1], pb_{id} is the best position reached by the particle i, pl_{id} is the best position reached by any particle in the neighborhood of particle

i and, pg_d is the best position reached by any particle in the swarm. Finally, p_{id} is the value of the particle i at the dimension d.

We empirically found that for some difficult functions, a previous version of our algorithm could not find good values. The reason was its diversification of solutions which kept the approach from converging. In previous works [20, 21], we changed the common updating formula (equation 2) of the particles for the update equation presented by Kennedy [23]. In Kennedy's algorithm, the new position of each particle is randomly chosen from a Gaussian distribution with the mean selected as the average between the best position recorded for the particle and the best in its neighborhood. The standard deviation is the difference between these two values. We adapted that formula adding the global best (*gbest*) to the best position of the particle and the best in its neighborhood. We also changed the way in which the standard deviation is determined. We use the *pbest* and, the *gbest* instead of the *lbest* as was proposed by Kennedy. We determined those changes after several empirical tests with different Gaussian random generator parameters. Thus, the position is updated using the following equation:

$$p_i = N\left(\frac{p_i + pl_i + pg}{3}, |p_i - pg|\right) \tag{3}$$

where p_i, pl_i and pg are defined as before and, N is the value retorned by the Gaussian random generator. SiC-PSO used this equation to update particles with a certain probability (a 92.5% probability was adopted to select between equation 3 and, equation 2 the rest of the time). We chosed that probability after conducting numerous experiments.

4. Parameter Settings and Analysis of Results

A set of 4 engineering design optimization problems was chosen to evaluate the performance of our proposed algorithm. The detailed description of the test problems may be consulted in the appendix at the end of this paper. We performed 30 independent runs per problem, with a total of 24,000 objective function evaluations per run. We also tested the algorithm with 27,000 and 30,000 evaluations of the objective function, but no performance improvements were noticed in such cases. Our algorithm used the following parameters: swarm size = 8 particles, neighborhood size = 3, inertia factor $w = 0.8$, personal learning factor and social learning factors for c_1, c_2 and c_3 were set to 1.8. These parameter settings were empirically derived after numerous experiments.

Our results were compared with respect to the best results reported in the specialized literature. Those values were obtained by Hernandez Aguirre et al. [24] and Mezura et al. [27]. We reference those results

into the tables shown next as "COPSO" and "Mezura", respectively. It is important remark that COPSO and Mezura algorithms reached the best values after 30,000 fitness function evaluations, which is a larger value than that required by our algorithm. The best values are shown in Table 1 and, the mean and standard deviations over the 30 runs are shown in Table 2.

Table 1. Best results obtained by COPSO, Mezura and SiC-PSO.

Prob.	Optimal	SiC−PSO	COPSO	Mezura
E01	1.724852	1.724852	1.724852	1.724852
E02	6,059.714335	6,059.714335	6,059.714335	6,059.7143
E03	NA	2,996.348165	2,996.372448	2,996.348094*
E04	0.012665	0.012665	0.012665	0.012689

*Infeasible solution. NA Not avaliable.

Table 2. Mean and St. Dev. for the results obtained.

	Mean			St. Dev.		
Prob.	SiC−PSO	COPSO	Mezura	SiC−PSO	COPSO	Mezura
E01	2.0574	1.7248	1.7776	0.2154	1.2-05	8.8E-02
E02	6,092.0498	6,071.0133	6,379.9380	12.1725	15.1011	210.0000
E03	2,996.3482	2,996.4085	2,996.3480*	0.0000	0.0286	0.0000*
E04	0.0131	0.0126	0.0131	4.1E-04	1.2E-06	3.9E-04

*Infeasible solution.

The three algorithms reached the best known values for E01. For E02, SiC-PSO and COPSO reached the best known, but Mezura reported a value with a precision of only 4 digits after the decimal point, and the exact value reached by them is not reported. For E03, SiC-PSO reached the best value, COPSO reached a value slightly worse than ours, and Mezura reached an infeasible value. SiC-PSO and COPSO reached the best value for E04, although Mezura reported a value that is worse than the best known. In general, COPSO obtained the best mean values, except for E03 for which best mean was found by our algorithm. The lower standard deviation values for E01 and E04 was obtained by COPSO; for E02 and E03, our SiC-PSO found the minimum values.

Tables 3a, 3b, 3c and 3d show the solution vectors of the best solution reached by SiC-PSO as well as the values of the constraints, for each of the problems tested.

Table 3a. SiC-PSO Solution vector for E01 (welded beam).

	Best Solution
x_1	0.205729
x_2	3.470488
x_3	9.036624
x_4	0.205729
$g_1(\vec{x})$	-1.819E-12
$g_2(\vec{x})$	-0.003721
$g_3(\vec{x})$	0.000000
$g_4(\vec{x})$	-3.432983
$g_5(\vec{x})$	-0.080729
$g_6(\vec{x})$	-0.235540
$g_7(\vec{x})$	0.000000
$f(\vec{x})$	1.724852

Table 3b. SiC-PSO Solution vector for E02 (pressure vessel).

	Best Solution
x_1	0.812500
x_2	0.437500
x_3	42.098445
x_4	176.636595
$g_1(\vec{x})$	-4.500E-15
$g_2(\vec{x})$	-0.035880
$g_3(\vec{x})$	-1.164E-10
$g_4(\vec{x})$	-63.363404
$f(\vec{x})$	6,059.714335

Table 3c. SiC-PSO Solution vector for E03 (speed reducer).

	Best Solution
x_1	3.500000
x_2	0.700000
x_3	17
$x4$	7.300000
x_5	7.800000
x_6	3.350214
x_7	5.286683
$g_1(\vec{x})$	-0.073915
$g_2(\vec{x})$	-0.197998
$g_3(\vec{x})$	-0.499172
$g_4(\vec{x})$	-0.901471
$g_5(\vec{x})$	0.000000
$g_6(\vec{x})$	-5.000-16
$g_7(\vec{x})$	-0.702500
$g_8(\vec{x})$	-1.000E-16
$g_9(\vec{x})$	-0.583333
$g_{10}(\vec{x})$	-0.051325
$g_{11}(\vec{x})$	-0.010852
$f(\vec{x})$	2,996.348165

Table 3d. SiC-PSO Solution vector for E04 (tension/compression spring).

	Best Solution
x_1	0.051583
x_2	0.354190
x_3	11.438675
$g_1(\vec{x})$	-2.000E-16
$g_2(\vec{x})$	-1.000E-16
$g_3(\vec{x})$	-4.048765
$g_4(\vec{x})$	-0.729483
$f(\vec{x})$	0.012665

5. Conclusions and Future Work

We have presented a simple PSO algorithm (SiC-PSO) for constrained optimization problems. The proposed approach uses a simple constraint-

handling mechanism, a ring topology for implementing the *lbest* model and a novel formula to update the position of particles. SiC-PSO had a very good performance when applied to several engineering design optimization problems. We compared our results with respect to those obtained by two algorithms that had been previously found to perform well in the same problems. These two algorithms are more sophisticated than our SiC-PSO. Our algorithm obtained the optimal values for each of the test problems studied, while performing a lower number of objective function evaluations. Also, the performance of our approach with respect to the mean and standard deviation is comparable with that shown by the other algorithms. Thus, we consider our approach to be a viable choice for solving constrained engineering optimization problems, due to its simplicity, speed and reliability. As part of our future work, we are interested in exploring other PSO models and in performing a more detailed statistical analysis of the performance of our proposed approach.

References

[1] S. Akhtar, K. Tai and T. Ray. A socio-behavioural simulation model for engineering design optimization. *Engineering Optimization*, 34:341–354, 2002.

[2] C. Guo, J. Hu, B. Ye and Y. Cao. Swarm intelligence for mixed-variable design optimization. *Journal of Zheijiang University SCIENCE*, 5(7):851–860, 1994.

[3] M. Mahdavi, M. Fesanghary and E. Damangir. An improved harmony search algorithm for solving optimization problems. *Applied Mathematics and Computation*, 188(2007):1567–1579, 2007.

[4] C. Zhang and H. Wang. Mixed-discrete nonlinear optimization with simulated annealing. *Engineering Optimization*, 21:277–291, 1993.

[5] S. Wu and T. Chou. Genetic Algorithms for nonlinear mixed discrete-integer optimization problems via meta-genetic parameter optimization. *Engineering Optimization*, 24:137–159, 1995.

[6] G. Thierauf and J. Cai. Evolution Strategies-parallelization and Applications in Engineering Optimization. In *Parallel and Distributed Precessing for Computational Mechanics.*B. H. V. Topping editors, 1997.

[7] H. Loh, and P. Papalambros. A sequential linearization approach for solving mixed-discrete nonlinear design optimization problems. *ASME Journal of Mechanical Design*, 113:325–334, 1991.

[8] H. Li and T. Chou. A global approach of nonlinear mixed discrete programming in design optimization. *Engineering Optimization*, 22:109–122, 1994.

[9] J. Fu, R. Fenton and W. Cleghorn. A mixed integer-discrete-continuous programming method and its applications to engineering design optimization. *Engineering Optimization*, 17:263–280, 1991.

[10] S. He, E. Prempain and Q. Wu. An improved Particle Swarm optimizer for Mechanical Design Optimization Problems. *Engineering Optimization*, 36(5):585–605, 2004.

[11] Y. Cao and Q. Wu. Mechanical Design optimization by Mixed-variable Evolutionary Programming. In *1997 IEEE International Conference on Evolutionary Computation.* Pages 443–446, 1997.

[12] H. Bernardino, H. Barbosa and A. Lemonge. A Hybrid Genetic Algorithm for Constrained Optimization Problems in Mechanical Engineering. In *IEEE Congress on Evolutionary Computation.* Pages 646–653, 2007.

[13] J. Cha, and R. Mayne. Optimization with discrete variables via recursive quadratic programming: part II. *Transaction of ASME*, 111:130–136, 1989.

[14] K. Ragsdell, and D. Phillips. Optimal design of a class of welded structures using geometric programming. *ASME Journal of Engineering for Industries*, 98(3):1021–1025, 1976.

[15] E. Sandgren. Nonlinear integer and discrete programming in mechanical design optimization. *ASME Journal of Mechanical Design*, 112:223–229, 1990.

[16] J. Arora. *Introduction to Optimum Design*. McGraw-Hill,New York, 1989.

[17] A. Belegundu. *A Study of Mathematical Programming Methods for Structural Optimization*. PhD thesis, Department of Civil Environmental Engineering, University of Iowa, Iowa, 1982.

[18] R. Eberhart and Y. Shi. A modified Particle Swarm Optimizer. In *International Conference on Evolutionary Computation, IEEE Service Center*, Anchorage, AK, Piscataway, NJ, 1998.

[19] L. Cagnina, S. Esquivel and R. Gallard. Particle Swarm Optimization for sequencing problems: a case study. In *Congress on Evolutionary Computation.* Pages 536–541, Portland, Oregon, USA, 1994.

[20] L. Cagnina, S. Esquivel and C. Coello Coello. A Particle Swarm Optimizer for Constrained Numerical Optimization. In *9th International Conference - Parallel problem Solving from Nature - PPSN IX.* Pages 910–919, Reykjavik, Island, 2006.

[21] L. Cagnina, S. Esquivel and C. Coello Coello. A Bi-population PSO with a Shake-Mechanism for Solving Constrained Numerical Optimization. In *IEEE Congress on Evolutionary Computation - CEC2007.* Pages 670–676, Singapur, 2007.

[22] J. Kennedy and R. Eberhart. Bores Bones Particle Swarm. In *IEEE Swarm Intelligence Symposium.* Pages 80–89. 2003.

[23] J. Kennedy. The Particle Swarm: social adaptation in Information-Processing Systems. In *New Ideas in Organization.* 1999. D. Corne and M. Dorigo and F. Glover editors.

[24] A. Hernandez Aguirre, A. Muñoz Zavala, E. Villa Diharce and S. Botello Rionda. COPSO: Constrained Optimization via PSO algorithm. *Center for Research in Mathematics (CIMAT).* Technical report No. I-07-04/22-02-2007, 2007.

[25] J. Golinski. An adaptive optimization system applied to machine synthesis. *Mechanism and Machine Synthesis.* 8(1973), pages 419–436, 1973.

[26] X. Hu, R. Eberhart and Y. Shi. Engineering optimization with particle swarm. 2003.

[27] E. Mezura and C. Coello. Useful Infeasible Solutions in Engineering Optimization with Evolutionary Algorithms. In *Proceedings of the 4th Mexican International Conference on Artificial Intelligence, MICAI 2005.* Lecture Notes on Artificial Intelligence No. 3789, pages 652–662. 2005.

Appendix: Engineering problems

Formulating of the engineering design problems used to test the algorithm proposed.

E01: Welded beam design optimization problem. The problem is to design a welded beam for minimum cost, subject to some constraints [14]. Figure A.1 shows the welded beam structure which consists of a beam A and the weld required to hold it to member B. The objective is to find the minimum fabrication cost, considerating four design variables: x_1, x_2, x_3, x_4 and constraints of shear stress τ, bending stress in the beam σ, buckling load on the bar P_c, and end deflection on the beam δ. The optimization model is summarized in the next equation:
Minimize: $f(\vec{x}) = 1.10471{x_1}^2 x_2 + 0.04811 x_3 x_4 (14.0 + x_2)$
subject to:

$$g_1(\vec{x}) = \tau(\vec{x}) - 13,600 \leq 0$$
$$g_2(\vec{x}) = \sigma(\vec{x}) - 30,000 \leq 0$$
$$g_3(\vec{x}) = x_1 - x_4 \leq 0$$
$$g_4(\vec{x}) = 0.10471({x_1}^2) + 0.04811 x_3 x_4 (14 + x_2) - 5.0 \leq 0$$
$$g_5(\vec{x}) = 0.125 - x_1 \leq 0$$
$$g_6(\vec{x}) = \delta(\vec{x}) - 0.25 \leq 0$$
$$g_7(\vec{x}) = 6,000 - Pc(\vec{x}) \leq 0$$

with:

$$\tau(\vec{x}) = \sqrt{(\tau')^2 + (2\tau'\tau'')\frac{x_2}{2R} + (\tau'')^2}$$
$$\tau' = \frac{6,000}{\sqrt{2} x_1 x_2}$$
$$\tau'' = \frac{MR}{J}$$
$$M = 6,000\left(14 + \frac{x_2}{2}\right)$$
$$R = \sqrt{\frac{{x_2}^2}{4} + \left(\frac{x_1 + x_3}{2}\right)^2}$$
$$J = 2\left\{x_1 x_2 \sqrt{2}\left[\frac{{x_2}^2}{12} + \left(\frac{x_1 + x_3}{2}\right)^2\right]\right\}$$
$$\sigma(\vec{x}) = \frac{504,000}{x_4 {x_3}^2}$$
$$\delta(\vec{x}) = \frac{65,856,000}{(30 \times 10^6) x_4 {x_3}^3}$$
$$Pc(\vec{x}) = \frac{4.013(30 \times 10^6)\sqrt{\frac{{x_3}^2 {x_4}^6}{36}}}{196}\left(1 - \frac{x_3\sqrt{\frac{30 \times 10^6}{4(12 \times 10^6)}}}{28}\right)$$

with $0.1 \leq x_1, x_4 \leq 2.0$, and $0.1 \leq x_2, x_3 \leq 10.0$.
Best solution: $x^* = (0.205730, 3.470489, 9.036624, 0.205729)$ where $f(x^*) = 1.724852$.

E02: Pressure Vessel design optimization problem. A compressed air storage tank with a working pressure of 3,000 psi and a minimum volume of 750 ft^3. A cylindrical vessel is capped at both ends by hemispherical heads (see figure A.2). Using rolled steel plate, the shell is made in two halves that are joined by teo longitudinal welds to form a cylinder. The objective is minimize the total cost, including the cost of the materials forming the welding [15]. The design variables are: thickness x_1, thickness of the head x_2, the inner radius x_3, and the length of the cylindrical section of the vessel x_4. The variables x_1 and x_2

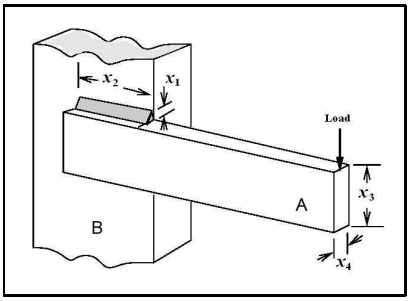

Figure A.1. Weldem Beam.

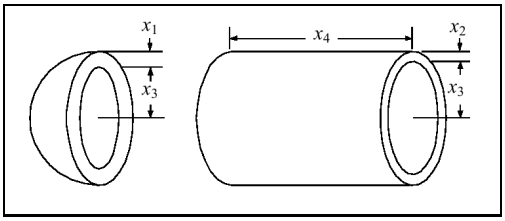

Figure A.2. Pressure Vessel.

are discrete values which are integer multiples of 0.0625 inch. Then, the formal statement is:
Minimize: $f(\vec{x}) = 0.6224x_1x_3x_4 + 1.7781x_2{x_3}^2 + 3.1661{x_1}^2x_4 + 19.84{x_1}^2x_3$
subject to:

$$g_1(\vec{x}) = -x_1 + 0.0193x_3 \leq 0$$
$$g_2(\vec{x}) = -x_2 + 0.00954x_3 \leq 0$$
$$g_3(\vec{x}) = -\pi{x_3}^2{x_4}^2 - \frac{4}{3}\pi{x_3}^3 + 1,296,000 \leq 0$$
$$g_4(\vec{x}) = x_4 - 240 \leq 0$$

with $1 \times 0.0625 \leq x_1$, $x_2 \leq 99 \times 0.0625$, $10.0 \leq x_3$, and $x_4 \leq 200.0$.
Best solution: $x^* = (0.8125, 0.4375, 42.098446, 176.636596)$ where $f(x^*) = 6,059.714335$.

E03: Speed Reducer design optimization problem. The design of the speed reducer [25] shown in figure A.3, is considered with the face width x_1, module of teeth x_2, number of teeth on pinion x_3, length of the first shaft between bearings x_4, length of the second shaft between bearings x_5, diameter of the first shaft x_6, and diameter of the first shaft x_7 (all variables continuous except x_3 that is integer). The weight of the speed reducer is to be minimized subject to constraints on bending stress of the gear teeth, surface stress, transverse deflections of the shafts and stresses in the shaft. The problem is:
Minimize: $f(\vec{x}) = 0.7854x_1{x_2}^2(3.3333x_3^2 + 14.9334x_3 - 43.0934) - 1.508x_1(x_6^2 + x_7^2) + 7.4777(x_6^3 + x_7^3) + 0.7854(x_4x_6^2 + x_5x_7^2)$
subject to:

$$g_1(\vec{x}) = \frac{27}{x_1x_2^2x_3} - 1 \leq 0$$
$$g_2(\vec{x}) = \frac{397.5}{x_1x_2^2x_3^2} - 1 \leq 0$$
$$g_3(\vec{x}) = \frac{1.93x_4^3}{x_2x_3x_6^4} - 1 \leq 0$$
$$g_4(\vec{x}) = \frac{1.93x_5^3}{x_2x_3x_7^4} - 1 \leq 0$$
$$g_5(\vec{x}) = \frac{1.0}{110x_6^3}\sqrt{\left(\frac{745.0x_4}{x_2x_3}\right)^2 + 16.9 \times 10^6} - 1 \leq 0$$
$$g_6(\vec{x}) = \frac{1.0}{85x_7^3}\sqrt{\left(\frac{745.0x_5}{x_2x_3}\right)^2 + 157.5 \times 10^6} - 1 \leq 0$$
$$g_7(\vec{x}) = \frac{x_2x_3}{40} - 1 \leq 0$$
$$g_8(\vec{x}) = \frac{5x_2}{x_1} - 1 \leq 0$$

$$g_9(\vec{x}) = \frac{x_1}{12x_2} - 1 \leq 0$$
$$g_10(\vec{x}) = \frac{1.5x_6 + 1.9}{x_4} - 1 \leq 0$$
$$g_11(\vec{x}) = \frac{1.1x_7 + 1.9}{x_5} - 1 \leq 0$$

with $2.6 \leq x_1 \leq 3.6$, $0.7 \leq x_2 \leq 0.8$, $17 \leq x_3 \leq 28$, $7.3 \leq x_4 \leq 8.3$, $7.8 \leq x_5 \leq 8.3$, $2.9 \leq x_6 \leq 3.9$, and $5.0 \leq x_7 \leq 5.5$.
Best solution: $x^* = (3.500000, 0.7, 17, 7.300000, 7.800000, 3.350214, 5.286683)$ where $f(x^*) = 2,996.348165$.

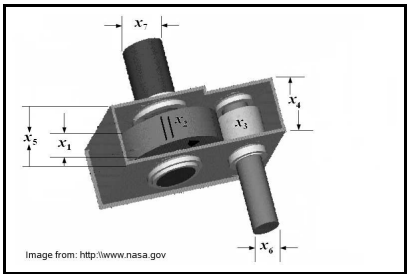

Figure A.3. Speed Reducer

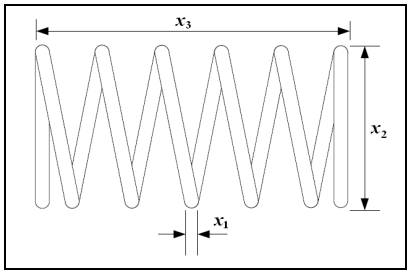

Figure A.4. Tension/Compression Spring.

E04: Tension/compression spring design optimization problem. This problem [16] [17] minimizes the weight of a tension/compression spring (figure A.4), subject to constraints of minimum deflection, shear stress, surge frequency, and limits on outside diameter and on design variables. There are three design variables: the wire diameter x_1, the mean coil diameter x_2, and the number of active coils x_3. The mathematical formulation of this problem is:
Minimize: $f(\vec{x}) = (x_3 + 2)x_2x_1^2$
subject to:

$$g_1(\vec{x}) = 1 - \frac{x_2^3 x_3}{7,178x_1^4} \leq 0$$
$$g_2(\vec{x}) = \frac{4x_2^2 - x_1x_2}{12,566(x_2x_1^3) - x_1^4} + \frac{1}{5,108x_1^2} - 1 \leq 0$$
$$g_3(\vec{x}) = 1 - \frac{140.45x_1}{x_2^2x_3} \leq 0$$
$$g_4(\vec{x}) = \frac{x_2 + x_1}{1.5} - 1 \leq 0$$

with $0.05 \leq x_1 \leq 2.0$, $0.25 \leq x_2 \leq 1.3$, and $2.0 \leq x_3 \leq 15.0$.
Best solution: $x^* = (0.051690, 0.356750, 11.287126)$ where $f(x^*) = 0.012665$.