Excerpt from the Proceedings of the 2011 COMSOL Conference in Stuttgart

Electromagnet Shape Optimization using Improved Discrete Particle Swarm Optimization (IDPSO)

Rhythm S. Wadhwa[*1], Terje Lien[2]
[1]NTNU, [2]NTNU
*Corresponding author: NTNU Valgrinda, Inst. for produksjons- og kvalitetstek., Trondheim, 7491, Norway, rhythmsuren@ieee.org

Abstract: The magnetic field gradient produced by an electromagnet gripper head depends on its design. Stochastic Methods offer certain robustness to the design optimization process. In this paper, Improved Discrete Particle Swarm Optimization (IDPSO) searching technique is applied to the shape and magnetic field gradient optimization of an electromagnet head. The magnetic field and forces are computed using COMSOL. The aim of the optimization is the search of an optimal pole shape geometry leading to a homogeneous magnetic field distribution and the desired holding force in the region of interest.

Keywords: Improved Discrete Particle Swarm Optimization, Electromagnet Design.

1. Introduction

Electromagnet grippers are commonly used for handling ferrous parts in foundries. (1) These grippers offer simple compact construction with no moving parts, uncomplicated energy supply, flexibility in holding complex parts and reduced number of set-ups. (5) For optimal performance of electromagnetic devices, it is necessary to perform design optimization of shape and parameters of their magnetic circuit, size and position of the current windings, magnetic properties of the used magnetic materials, etc. The traditional optimization methods based on trial-and-error procedures are not very suitable, especially for highly complex and multivariable optimization problems because they are time consuming and not accurate. Therefore, the development of new and more efficient methods for inverse optimization and automation of entire optimization process are always desired.

Optimization methods are usually divided into two categories: the gradient-based (deterministic) search methods and non-gradient-based (stochastic) search methods. While former ones need computation of the gradient of the objective function, the latter ones work directly with the values of the objective function, and are more convenient in cases where it is difficult or even impossible to compute exactly the gradient of the objective function. (7)

In electromagnetic device optimization it is a common problem to design devices which will result with desired values of magnetic holding forces with uniform flux density at several certain points. Since the exact expression of the gradient function is impossible, therefore, the usage of deterministic optimization methods is excluded. Hence, for such optimization problems designers utilize stochastic methods. Genetic Algorithms (GA), Evolutionary Strategies (ES) or Particle Swarm Optimization (PSO) are such stochastic methods which have become very popular in the computer aided design of electromagnetic devices. (7)

In order to obtain an approximate solution of an electromagnet parameter optimization, some new concepts have been proposed in recent years. They include applications of Genetic Algorithm approach (8), simulated annealing (9) and adaptive simulated annealing (10)] However, the application of particle swarm optimization (PSO) and their variants to electromagnet parameter design has not been fully explored in the literature.

In this paper we apply an improved particle discrete swarm optimization (IDPSO) searching technique to shape optimization of pole shape geometry of a simple electromagnet. We are searching for optimal pole shape modifications leading to a homogeneous magnetic flux density and holding force in a certain region.

2. Theoretical Background of IDPSO

The initial ideas on particle swarms of Kennedy and Eberhart were essentially aimed at producing computational intelligence by exploiting simple analogues of social interaction, rather than purely individual cognitive abilities (15)(12). The first simulations (12) were influenced by Heppner and Grenander's work (11) and involved analogues of bird flocks searching for corn. These soon developed (12) into a powerful optimization method— Particle Swarm Optimization (PSO).

PSO is an optimization algorithm that is based on swarm intelligence principle (12) which is widely used in application domains such as function optimization, neural network training, fuzzy system control and so on at present (14). It has been proved to be very effective for solving global optimization in various engineering application such as image and video analysis and design and optimization of communication networks. There are also some applications in fault diagnosis and maintenance optimization. However, most applications in this field are using PSO to train ANN or optimize the parameters of FLs. A direct application of PSO variant in maintenance optimization will be shown in this paper.

2.1 Basic PSO Algorithm Description

The Particle Swarm Optimization (PSO) algorithm is a heuristic approach motivated by the observation of social behavior of composed organisms such as birds flocking (Figure 1). A number of simple entities – the particles – are placed in the search space of some problem or function, and each evaluates the objective function at its current location. Each individual in the particle swarm is composed of D dimensional vectors, where D is the dimensionality of the search space.

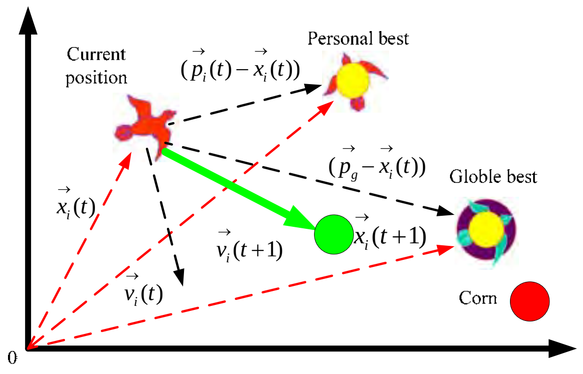

Figure 1. Bird Flocking of PSO

The current position $\vec{x}_i$ can be considered as a set of coordinates describing a point in space. If the current position is better than any that has been found so far, then the coordinates are stored in the vector $\vec{p}_i$. The value of the best function result so far is stored in a variable that can be called $\vec{p}_g$. The objective, of course, is to keep finding better positions and updating $\vec{p}_i$ and $\vec{p}_g$. New points are chosen by adding $\vec{v}_i$ coordinates to $\vec{v}_i$, and the algorithm operates by adjusting $\vec{v}_i$, which can effectively be seen as a step size. The steps of implementing PSO were shown as follows:

1) Initialize a population array of particles with random positions and velocities on D dimensions in the search space.
2) Loop
3) For each particle, evaluate the desired optimization fitness function in D variables.
4) Compare particle's fitness evaluation with that of its $\vec{p}_i$. If current value is better than that of $\vec{p}_i$, then set $\vec{p}_i$ equal to the current coordinates.
5) Identify the particle in the neighborhood with the best success so far, and assign it to the variable $\vec{p}_g$.
6) Change the velocity and position of the particle according to the following equation:

$$\vec{v}_i(t+1) = \omega \cdot \vec{v}_i(t) + c_1 \cdot r_1\big(\vec{p}_i - \vec{x}_i(t)\big) + c_2 \cdot r_2(\vec{p}_g - \vec{x}_i(t)) \qquad (1)$$

$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1) \qquad (2)$$

Where: ω is the inertia weighting; c_1 and c_2 are acceleration coefficients, positive constraint; r_1 and r_2 are the random numbers deferring uniform distribution on [0, 1]; *i* represents *i*th iteration.

7) If a criterion is met (usually a sufficiently good fitness or a maximum number of iterations), exit loop.
8) **End loop**

In PSO, every particle remembers its own previous best value as well as the neighborhood best; therefore it has a more effective memory capability than the GA. PSO is also more efficient in maintaining the diversity of the swarm, since all the particles use some information related to the most successful particle in order to improve themselves, whereas in GA, the worse solutions at every generation are discarded and only the good ones are saved for next generation. Therefore in GA the population evolves around a set of best individuals in every generation. In addition, PSO is easier to implement and there are fewer parameters to adjust compared with GA (12).

2.2 Discrete PSO (DPSO) Algorithm Description

The general concepts behind optimization techniques initially developed for problems defined over real-valued vector spaces, such as PSO, can also be applied to discrete valued search spaces where either binary or integer variables have to be arranged into particles. When integer solutions (not necessarily 0 or 1) are needed, the optimal solution can be determined by rounding off the real optimum values to the nearest integer. DPSO has been developed specifically for solving discrete problems. The new velocity and position for each is determined according to the velocity and position update equations given by (8) and (9).

$$\vec{v}_i(t+1) = round(\omega \cdot \vec{v}_i(t) + c_1 \cdot r_1\big(\vec{p}_i - \vec{x}_i(t)\big) + c_2 \cdot r_2(\vec{p}_g - \vec{x}_i(t))) \qquad (3)$$
$$\vec{x}_i(t+1) = \vec{x}_i(t) + \vec{v}_i(t+1) \qquad (4)$$

In equation (8), the value of velocity is binary or integer because *round ()* function can round off the value.

2.3 Improved DPSO (IDPSO) Algorithm Description

DPSO or PSO performs well in the early iterations, but they have problems approaching a near-optimal solution. If a particle's current position accords with the global best and its inertia weight multiply previous velocity is close to zero, the particle will only fall into a specific position. If their previous velocities are very close to zero, all the particles will stop moving around the near-optimal solution, which may lead to premature convergence of algorithm. All the particles have converged to the best position discovered so far which may be not the optimal solution. So, an improved DPSO is proposed here.

In IDPSO, before updating the velocities and positions in every iteration, the particles are ranked according to their fitness values in descending order. Select the first part of particles (suppose mutation rate is α, fist part is (1-α)) and put them into the next iteration directly. Regenerate the rest part of particles (α) randomly. In this project, we can regenerate the positions and velocities according to the following equation:

$$x_{id} = round\big(rand \cdot \big(S^{\max}(j) - S^{\min}(j)\big) + S^{\min}(j)\big) \qquad (10)$$
$$v_{id}(t) = v_{max} - round \cdot (rand \times 2v_{max}) \qquad v_{id}(t) \in [-v_{max}, v_{max}] \qquad (5)$$

3. Use of COMSOL Multiphysics

COMSOL Multiphysics has been used to study and test multiple possible magnet forms to optimize the magnetic force. To compute and plot the magnetic flux density around the system tip, the model of the electromagnet was implemented in 2D, as well as in 3D.The area of interest experiences a magnetic force according to the formula $F_{mag} = \mu.\nabla B$, where μ is the magnetic moment of a given particle and ∇B is the gradient of the magnetic field.

The involved Maxwell equations are: $\nabla x H = J$ and $\nabla B = 0$, with constitutive relation $B = \mu_0 \mu_r H$. The magnetic vector potential A produces the governing equation $\nabla x(\mu^{-1}\nabla x A - M) = J$ of the Magnetostatics module in COMSOL version 4.2.

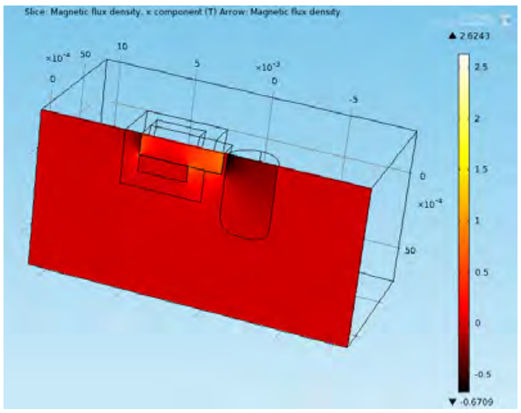
Slice: Magnetic flux density, x component (T) Arrow: Magnetic flux density
▲ 2.6243
▼ -0.6709

Figure 2. Possible Solution (a) of an electromagnet head

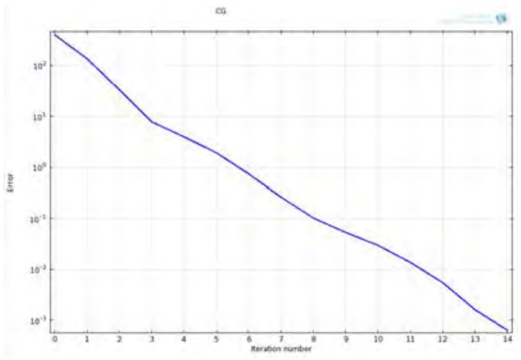

Figure 3. Convergence Plot

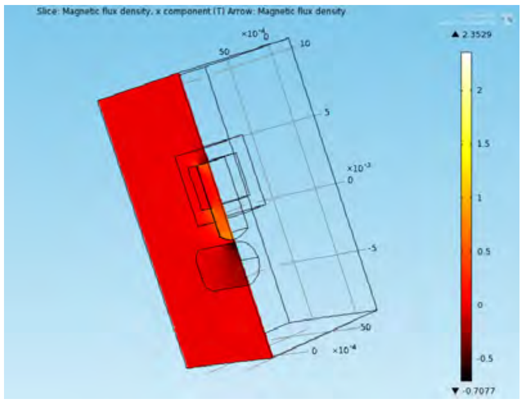
Slice: Magnetic flux density, x component (T) Arrow: Magnetic flux density
▲ 2.3529
▼ -0.7077

Figure 4. Possible Solution (b) of an electromagnet head

4. Simulation Results

The aim of the simulation study is the investigation of the effects of modified stochastic operators on the shape optimization process. We minimized the cost function

$$C_F = \sqrt{\sum_{i=1}^{N_{mp}} \left(\sqrt{F_x^2 + F_y^2 + F_z^2} - F_o\right)^2} \qquad (6)$$

where B_o is the desired constant magnetic flux density in the matching points (N_{mp}). The current density impressed to the coil to get a value of B_o =0.1 T was first estimated in some test runs.

The variation of the number of particles with time to reach global optimum is shown in Figure 5.

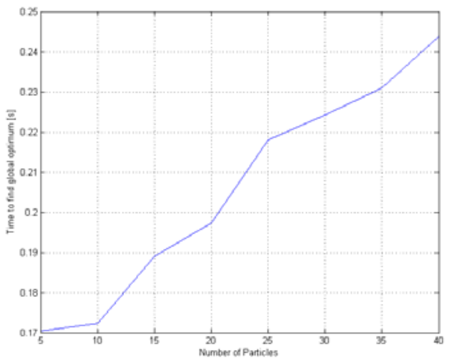

Figure 5. Variation of particles with time

5. Conclusions

In optimization problems where derivatives of the cost functions are not available stochastic methods like Genetic Algorithm (GA), Simulated Annealing (SA), and Particle Swarm Optimization (PSO) can be applied. In this paper an improved version of PSO method was used to optimize the shape of a simple electromagnet configuration. The algorithm known from the literature have been combined with finite element code in COMSOL to study the performance for shape optimization problems.

The true strength of PSO lies in its ability to statistically deliver a true global optimum, but there are no theoretical reasons for assuming it will be more efficient than other stochastic methods. Thus, evaluation of the performance of a certain PSO algorithm always depends on the specific characteristics of the considered problem.

In the near future, it is planned to conduct further comparisons with other Swarm Intelligence (SI) Techniques that will show under which conditions these conclusions can be maintained. It can be expected there are some systems for which one of the SI techniques is better suited than the others.

6. References

1. R.S.Wadhwa, T.Lien and G.J.Monkman Robust Prehension for ferrous metalcasted product families, *Proceedings of MITIP,* 2011

2. Campbell, in *Castings*. Butterworth-Heinemann, 2003

3. S.R.Hoole *Computer Aided Analysis and Design of Electromagnetic Devices*, 1989

4. J.D.Law *Modeling of Field Regulated Reluctance Machines,* PhD Thesis, University of Wisconsin Madison, 1991

5. G.J.Monkman and S.H.Steinmann *Robot Grippers*, Wiley-VCH, 2007

6. D.T.Pham and E.Tacgin An expert system for selection of robot grippers. Expert Systems with Applications,1992, 5, 289-300

7. Zaoui, F., C. Marchand: Using genetic algorithm for the optimization of electromagnetic devices. COMPEL, vol 17, No.1/2/3, 1998, pp.181-185

8. Billinton, R., and Abdulwhab A., (2003) Short-term generating unit maintenance in a deregulated power system using a probabilistic approach, *IEEE Proc., Gener. Transm. Distrib*., Vol. 4, pp. 463-468.
9. Satoh T., and Nara K., (1991) Maintenance scheduling by using simulated annealing method, *IEEE Trans. Power Syst*., Vol. 6, pp. 850-857.
10. Yellen J., Al-Khamis T. M., VERMURI S., LEMONIDIS L., (1992). A decomposition approach to unit maintenance scheduling, *IEEE Trans. Power Syst*., Vol.7, pp. 726–733.
11. Heppner, H., and Grenander, U., (1990) A stochastic non-linear model for coordinated bird flocks, *The ubiquity of chaos,* pp. 233–238. Washington: AAAS.
12. Kennedy, J., and Eberhart, R. C. (1995) Particle swarm optimization. *Proceedings of the IEEE international conference on neural networks IV*, pp. 1942–1948.
13. Eberhart, R. C., and Kennedy, J., (1995) A new optimizer using particles swarm theory. *Proceedings of Sixth International Symposium on Micro Machine and Human Science*, pp. 39-43.
14. Pan Hongxia, and Wei Xiuye, (2009) Particle Swarm Optimization Algorithm with Adaptive Velocity and its Application to Fault Diagnosis, *2009 IEEE Congress on Evolutionary Computation*, pp. 3075-3079.
15. Riccardo Poli, James Kennedy, and Tim Blackwell (2007) Particle swarm optimization, *Swarm Intelligence*, Vol. 1, pp. 33-57.

7. Appendix

Table 1: Magnetic Constants

Magnetic Constants	Value
Relative Permeability (μ_r)	4e3 (Iron)
External Current density ($J_o\varphi$)	1.79 A/m^2

Table 2: Magnetostatic Equations

Magnetostatic Equations	Value
Magnetic Insulation	$A_{\varphi} = 0$
Continuity	$nx(H_1 - H_2) = 0$
Relative Permeability	Isotropic in each subdomain

Table 3: Parameters of the Coil

Diameter of the Copper Wire	$d = 1.22mm$
Cross-section of the wire	$A_L = 1.13mm^2$
Average length of the winding	$l_m = 34.56cm$
Number of windings	$N = 3714$
Length of the coil	$l = 1283.56m$
Mass of the coil	$m = 12.95kg$
External current density	$J = 1.79e6A/$
Output voltage	$U = 41.14V$
Output current	$I = 2.04A$