Engineering Optimization
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/geno20

Decomposition-based multi-objective differential evolution particle swarm optimization for the design of a tubular permanent magnet linear synchronous motor

Guanghui Wang [a b], Jie Chen [a b], Tao Cai [a b] & Bin Xin [a b c]

[a] School of Automation, Beijing Institute of Technology, Beijing, 100081, PR China

[b] Key Laboratory of Complex System Intelligent Control and Decision, Ministry of Education, Beijing, 100081, PR China

[c] Decision and Cognitive Sciences Research Centre, Manchester Business School, University of Manchester, Manchester, M15 6PB, UK

Version of record first published: 03 Oct 2012.

To cite this article: Guanghui Wang, Jie Chen, Tao Cai & Bin Xin (2012): Decomposition-based multi-objective differential evolution particle swarm optimization for the design of a tubular permanent magnet linear synchronous motor, Engineering Optimization, DOI:10.1080/0305215X.2012.720682

To link to this article: http://dx.doi.org/10.1080/0305215X.2012.720682

Engineering Optimization
iFirst, 2012, 1–21

Decomposition-based multi-objective differential evolution particle swarm optimization for the design of a tubular permanent magnet linear synchronous motor

Guanghui Wang[a,b], Jie Chen[a,b], Tao Cai[a,b]*, and Bin Xin[a,b,c]

[a]*School of Automation, Beijing Institute of Technology, Beijing, 100081, PR China;* [b]*Key Laboratory of Complex System Intelligent Control and Decision, Ministry of Education, Beijing, 100081, PR China;* [c]*Decision and Cognitive Sciences Research Centre, Manchester Business School, University of Manchester, Manchester M15 6PB, UK*

(*Received 4 February 2012; final version received 16 July 2012*)

This article proposes a decomposition-based multi-objective differential evolution particle swarm optimization (DMDEPSO) algorithm for the design of a tubular permanent magnet linear synchronous motor (TPMLSM) which takes into account multiple conflicting objectives. In the optimization process, the objectives are evaluated by an artificial neural network response surface (ANNRS), which is trained by the samples of the TPMSLM whose performances are calculated by finite element analysis (FEA). DMDEPSO which hybridizes differential evolution (DE) and particle swarm optimization (PSO) together, first decomposes the multi-objective optimization problem into a number of single-objective optimization subproblems, each of which is associated with a Pareto optimal solution, and then optimizes these subproblems simultaneously. PSO updates the position of each particle (solution) according to the best information about itself and its neighbourhood. If any particle stagnates continuously, DE relocates its position by using two different particles randomly selected from the whole swarm. Finally, based on the DMDEPSO, optimization is gradually carried out to maximize the thrust of TPMLSM and minimize the ripple, permanent magnet volume, and winding volume simultaneously. The result shows that the optimized TPMLSM meets or exceeds the performance requirements. In addition, comparisons with chosen algorithms illustrate the effectiveness of DMDEPSO to find the Pareto optimal solutions for the TPMLSM optimization problem.

Keywords: tubular permanent magnet linear synchronous motor; multiobjective optimization; decomposition; differential evolution; particle swarm optimization

1. Introduction

A motor can be considered as an electromechanical system, which can be described as functions of several structure variables. Skilled designers can design highly effective motors by utilizing their knowledge, experience and judgment to specify these variables. However, due to the complexity of multiple tasks, even the most skilled designers are unable to take into account all of the tasks simultaneously. It is necessary to apply multi-objective optimization to assist the designers in optimizing the motor's performance, reliability, and/or cost, etc.

*Corresponding author. Email: caitao@bit.edu.cn

ISSN 0305-215X print/ISSN 1029-0273 online

http://dx.doi.org/10.1080/0305215X.2012.720682
http://www.tandfonline.com

Due to the multi-criteria nature of most real-world problems, multi-objective optimization problems (MOPs), which are ubiquitous, particularly throughout engineering applications, involve multiple objectives that should be optimized simultaneously. An MOP can be mathematically formulated as

$$\begin{aligned} \text{minimize} \quad & \mathbf{F}(\mathbf{x}) = (f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))^{\mathrm{T}} \\ & \text{s.t. } \mathbf{x} \in \Omega, \end{aligned} \tag{1}$$

where $\mathbf{\Omega}$ is the decision space and $\mathbf{x} \in \mathbf{\Omega}$ is a decision vector. $\mathbf{F}(\mathbf{x})$ consists of m objective functions $f_j : \mathbf{\Omega} \rightarrow \mathbf{R}^m, j = 1, \ldots, m$, where $\mathbf{R}^m$ is the objective space.

Ashabani *et al.* (2010) proposed a multi-objective design optimization to improve force ripple, developed thrust, and permanent magnet (PM) volume of an axially magnetized tubular linear motor simultaneously by simplifying multi-objective fitness functions to a single one. For optimizing the thrust and ripple of a double sided permanent magnet linear motor with respect to its PM width, PM thickness and airgap length, Li *et al.* (2011a) used the thrust ripple ratio ρ to transform the two-objective problem into a single-objective problem. For the shape optimization of surface-mounted permanent magnet motors, a weighted combination of torque ripple, RMS value of phase current, and total harmonic distortion of phase current was minimized as a single-objective by a fast hybrid genetic-particle swarm optimization (Sarikhani and Mohammed 2011).

The common method in the above optimization applications is transforming an MOP into a single-objective optimization problem (SOP, minimize $f(\mathbf{x}) = y(f_1(\mathbf{x}), \ldots, f_m(\mathbf{x}))$, y is an aggregation method). Generally, there only exists a single solution for an SOP. However, because the multiple objectives often conflict with each other, no single solution can optimize all objectives simultaneously. Instead of a single solution for an SOP, the Pareto optimality concept is formally defined for MOPs as follows (Deb 2001).

DEFINITION 1.1 *A vector* $\mathbf{u} = (u_1, \ldots, u_m)^{\mathrm{T}}$ *is said to dominate another vector* $\mathbf{v} = (v_1, \ldots, v_m)^{\mathrm{T}}$*, denoted as* $\mathbf{u} \prec \mathbf{v}$*, iff* $\forall j \in \{1, \ldots, m\}$*,* $u_j \leq v_j$ *and* $\mathbf{u} \neq \mathbf{v}$.

DEFINITION 1.2 *A feasible solution* $\mathbf{x}^* \in \mathbf{\Omega}$ *of problem (1) is called a Pareto optimal solution, iff* $\nexists \mathbf{y} \in \mathbf{\Omega}$ *such that* $\mathbf{F}(\mathbf{y}) \prec \mathbf{F}(\mathbf{x}^*)$*. The set of all the Pareto optimal solutions is called the Pareto set* (PS)*, denoted as*

$$\mathrm{PS} = \left\{ \mathbf{x} \in \mathbf{\Omega} | \nexists \mathbf{y} \in \mathbf{\Omega}, \mathbf{F}(\mathbf{y}) \prec \mathbf{F}(\mathbf{x}) \right\}.$$

The image of the PS in the objective space is called the Pareto front (PF)

$$\mathrm{PF} = \{ \mathbf{F}(\mathbf{x}) | \mathbf{x} \in \mathrm{PS} \}.$$

In SOPs, all solutions can be compared based on their objective function values, and the task of optimization is often to find one single optimal solution. In MOPs, the aim of optimization is to produce a number of Pareto optimal solutions based on domination. All the solutions should be selected to be as diverse as possible for representing the whole PF. Therefore, a selection mechanism should be applied effectively for MOPs. There are three most possible mechanisms (Wagner *et al.* 2007, Coello Coello 2011a), Pareto-based selection, indicator-based selection and aggregating-functions. The decomposition approach was first presented by Zhang and Li (2007) as an aggregating-functions method. It was shown to be a very promising way to address MOPs (Coello Coello 2011a) and applied to some other engineering problems.

A specialized MOEA (Konstantinidis *et al.* 2010) based on decomposition (MOEA/D) uses an M-tournament selection, a window crossover and a mutation operator to find optimal mobile agent routes for minimizing the data path loss and the energy consumption of the sensors as well as maximizing the data accuracy. Simulation results have shown that the specialized MOEA/D performs better than the MOEA/D (Zhang and Li 2007) and the EMOCA (evolutionary multi-objective crowding algorithm, see Rajagopalan *et al.* 2008). Then Konstantinidis and Yang (2011a) applied the algorithm to solve the K-connected deployment and power assignment problem in wireless sensor networks (WSNs). Furthermore, Konstantinidis and Yang (2011b) proposed a generalized subproblem-dependent heuristic to be incorporated into MOEA/D to improve the performance for the WSNs. Cheng *et al.* (2011) proposed a multi-objective ant colony optimization algorithm based on decomposition to solve the bi-objective travelling salesman problem.

Owing to their population-based heuristic nature, evolutionary algorithms (EAs) are the most widely used technique to deal with MOPs. By 1 February 2012, more than 6700 publications had been published on evolutionary multi-objective optimization (Coello Coello 2011b), most in the last eight years. As two instances of EAs used for MOPs, particle swarm optimization (PSO) and differential evolution (DE), two powerful population-based metaheuristics algorithms, were widely used.

Reyes-Sierra and Coello Coello (2006) presented the main issues to be considered when extending PSO to solve MOPs, and summarized some methods to handle them. Recently, an enhanced multi-objective particle swarm optimization approach based on Pareto dominance (dos Santos Coelho *et al.* 2010) was used to optimize 10 design variables of a brushless DC wheel motor considering two objectives simultaneously, *i.e.* the minimization of the mass and the maximization of the efficiency. Kotinis (2011) implemented co-evolution and parallelization into multi-objective particle swarm optimization, and the effectiveness of the multi-objective PSO (Nourbakhsh *et al.* 2011) was proved by comparison with NSGA-II in centrifugal pump applications. Gas turbine engine fuel controller gain tuning can be modelled as an MOP, and solved by PSO (Montazeri-Gh *et al.* 2011). Kulturel-Konak and Konak (2011) proposed PSO with flexible bay structure to solve the unequal area facility layout problem with two objectives, the material handling cost and the layout area requirement.

Villarreal-Cervantes *et al.* (2010) used differential evolution to optimize the structural parameters of a non-redundant parallel robot and the control parameters simultaneously for a five-bar parallel robot. A self-adaptive differential evolution algorithm (Wang *et al.* 2010) with elitist archive and entropy-based diversity preservation strategy was presented to solve multi-objective optimization based on Pareto dominance. Basu (2011) used a two-objective differential evolution to optimize both fuel cost and emission level for an economic environmental dispatch problem. Ramesh *et al.* (2011) proposed an improved generalized differential evolution (I-GDE3) based on the concepts of simulated binary crossover based recombination and dynamic crowding distance. It can optimize the reactive power dispatch problem with multiple and competing objectives, real power loss and voltage deviation.

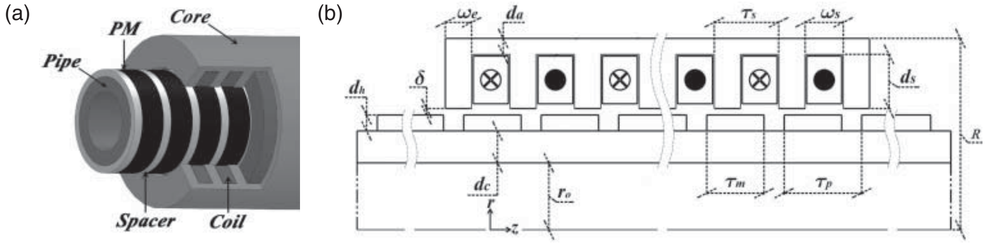
For both PSO and DE, hybridization with other approaches can contribute to their performance improvement for solving practical engineering problems (Kulturel-Konak and Konak 2011, Li *et al.* 2011b). Moreover, the hybridization between DE and PSO is a very promising way to create more efficient optimizers, as surveyed by Xin *et al.* (2011) recently. Chen *et al.* (2009 pointed out that a reasonable tradeoff between exploration and exploitation makes the hybridization more powerful. Therefore, in this article, for the design optimization of a tubular permanent magnet linear synchronous motor (TPMLSM) with multiple objectives, a decomposition-based multi-objective differential evolution particle swarm optimization (DMDEPSO) approach is proposed by hybridizing PSO and DE reasonably. The remainder of this article is organized as follows. In Section 2, the structure and the analysis methods of the TPMLSM are presented, then an artificial neural network response surface is established for calculating the performance with respect to each combination of design variables, instead of FEA to reduce time-consumption. In Section 3, hybridizing DE and PSO, DMDEPSO is proposed to deal with the multi-objective TPMLSM optimization. Section 4 provides the design of TPMLSM optimized by DMDEPSO. Conclusions are drawn in Section 5.

2. Tubular permanent-magnet linear synchronous motor

2.1. *Motor structure*

Tubular permanent magnet linear synchronous motors (TPMLSMs), which show a significant advantage in terms of direct thrust, high efficiency, good reliability and high accuracy positioning, have been widely applied in industrial robots, machine tools, packaging equipment, etc. The schematic model of the present TPMLSM to be optimized is shown in Figure 1.

The TPMLSM consists of a long stator (internal assembly) and a short mover or runner (exterior cylinder). The mover consists of a armature core and a three-phase armature winding. The winding is assembled from 24 coils. As shown in Figure 1(a), each coil is located separately in an armature slot. The stator consists of a ferromagnetic pipe, permanent magnets (PMs) and non-ferromagnetic spacers. These PMs located on an inner pipe are separated from each other by the non-ferromagnetic spacers. The pipe is made of high strength stainless steel with good rigidity for bearing external mechanical forces. The armature core is composed of laminated silicon–steel sheets with high permeability. Figure 2 shows the nonlinearities of the ferromagnetic pipe and armature core (Song 2003). Generally, since the coercive force H_c of the neodymium–iron–boron (NdFeB) PM is nearly $-927\,\mathrm{kA\,m^{-1}}$, material linearity is assumed. Its relative permeability μ_r is close to 1.05.



3D view with cutting.

The halved cross section.

Figure 1. The TPMLSM model.

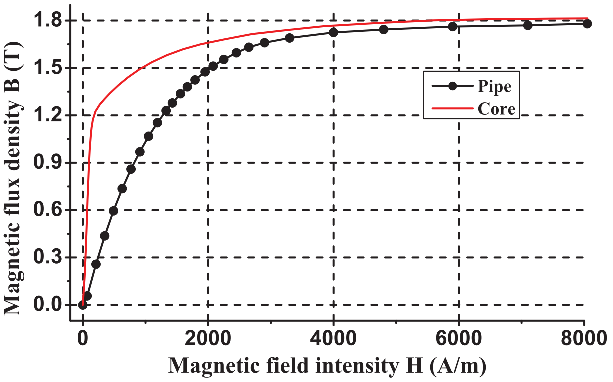


Figure 2. The B–H characteristic of the pipe and armature core.

Table 1. Parameters of the TPMLSM model (all values in millimetres).

Parameter	Symbol	Value	Parameter	Symbol	Value
Airgap	δ	[1, 3]	Pipe inner radius	r_0	21
PM width	τ_m	[10, 24.3]	Slot depth	d_s	[13, 23]
PM height	d_h	[2, 6]	Slot width	ω_s	[10, 16]
Pole pitch	τ_p	24.3	Pipe thickness	d_c	[8, 12]
Motor outer radius	R	60	Slot bottom	d_a	5
End tooth width	ω_e	[4, 12]	Slot pitch	τ_s	20.25

The pole pitch τ_p is set to 24.3 mm. Due to the 24/20 pole/slot ratio (Wang 2011), the slot pitch τ_s is 20.25 mm. The other key geometric parameters shown in Figure 1(b) are listed in Table 1. Only the parameters with non-fixed values are chosen to be optimized. The mover outer radius $R = r_0 + d_c + d_h + \delta + d_s + d_a$ is set to 60 mm to meet design requirement.

2.2. *Electromagnetic analysis*

Because of no significant magnetic effect at a low frequency, the displacement current can be neglected. Thus, omitting the eddy current generated by the motion of the permanent magnets, the simplified form of the Maxwell's equation for TPMLSM electromagnetic analysis can be written as

$$\nabla \times \mathbf{H} = \mathbf{J}, \tag{2}$$

where **H** is the magnetic field intensity and **J** is the current density. Because the magnetic flux density **B** satisfies $\mathbf{B} = \mu(\mathbf{B})\mathbf{H}$ and the magnetic vector potential **A** satisfies $\mathbf{B} = \nabla \times \mathbf{A}$, Equation (2) is converted to the Poisson differential equation (3):

$$\nabla \times \left(\frac{1}{\mu(\mathbf{B})} \nabla \times \mathbf{A} \right) = \mathbf{J}, \tag{3}$$

where $\mu(\mathbf{B})$ is the magnetic permeability.

Because of the non-existence of the radial component A_r and axial component A_z in the cylindrical system, only the angular component A_φ of the magnetic vector potential **A** governs the domain. Thus, Equation (3) can be formulated in the generalized form

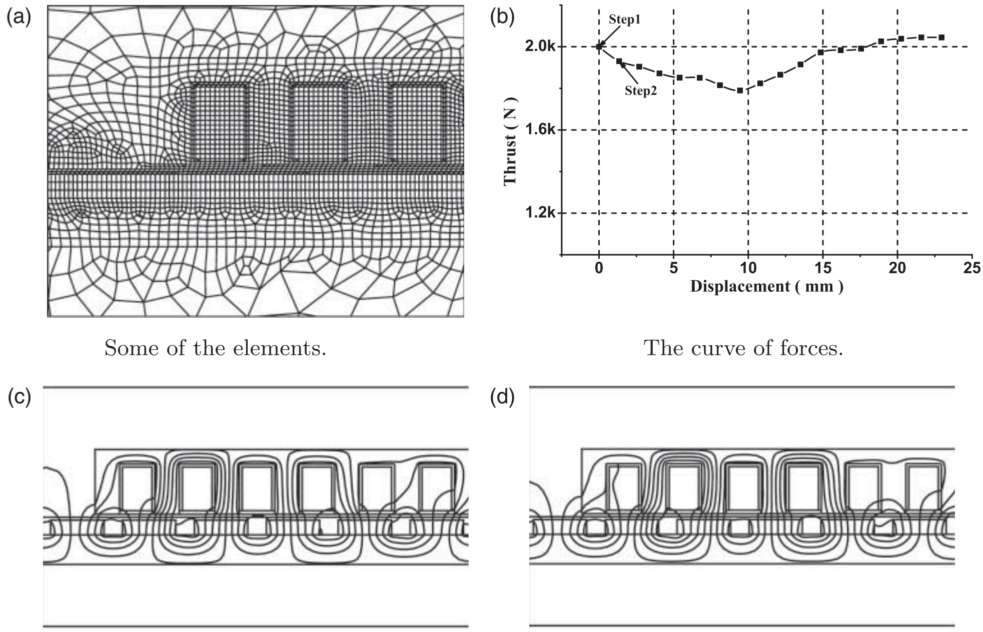
$$\frac{\partial}{\partial r}\left[\frac{1}{\mu(\mathbf{B})}\left(\frac{\partial A_\varphi}{\partial r} + \frac{A_\varphi}{r}\right)\right] + \frac{\partial}{\partial z}\left(\frac{1}{\mu(\mathbf{B})}\frac{\partial A_\varphi}{\partial z}\right) = -J_\varphi, \tag{4}$$

where J_φ is the angular component of the current density **J**. Over the region of the slots, the current density **J** is the excitation input current density $\mathbf{J_c}$ of the coils. Over the region of the permanent magnets, the current density **J** is set to the equivalent current density $\mathbf{J_m} = \nabla \times \mathbf{M}$.

By solving Equation (4), the distribution of magnetic vector potential **A** can be obtained, then the magnetic flux density **B** can be calculated by Equation (5):

$$\mathbf{B} = \nabla \times \mathbf{A} = -\frac{\partial A_\varphi}{\partial z}\mathbf{e}_r + \left(\frac{\partial A_\varphi}{\partial r} + \frac{A_\varphi}{r}\right)\mathbf{e}_z. \tag{5}$$

The magnetic flux density **B** in the 3-D cylindrical coordinate system (r, φ, z) only has the components $B_r(r, \varphi, z)$ in the radial direction $\mathbf{e}_r$ and $B_z(r, \varphi, z)$ in the axial direction $\mathbf{e}_z$. Finally,



Some of the elements.

The curve of forces.

Flux contour lines in Step 1.

Flux contour lines in Step 2.

Figure 3. Finite element analysis of the TPMLSM.

the force **F** can be deduced as follows:

$$\mathbf{F} = \frac{1}{\mu_0} \oint_s \begin{bmatrix} B_r^2 - \frac{1}{2}|\mathbf{B}|^2 & B_r B_z \\ B_r B_z & B_z^2 - \frac{1}{2}|\mathbf{B}|^2 \end{bmatrix} \begin{bmatrix} \mathbf{e}_r \\ \mathbf{e}_z \end{bmatrix} \mathrm{d}s, \tag{6}$$

where μ_0 is the permeability of free space and s is the integration surface, which is the contact surface between the mover and free space. Free space herein means the space the motor runs in, such as air or oil.

2.3. *Finite element analysis*

Because Equation (4) can be solved only when all key effects are considered, such as material nonlinearity, the natural boundary condition, and the continuity of the complicated contact surfaces between different components, it is hard to solve the electromagnetic problem of the TPMLSM by solving Equation (4) directly. Instead, finite element analysis (FEA) can be used to evaluate the motor performance numerically with good accuracy.

Symmetry is exploited to reduce the problem domain to half of the axial cross section of the TPMLSM shown in Figure 1. The calculation area consists of the halved cross section and its surrounding air, with natural boundary conditions. The area is first discretized into finite element sub-domains, such as the TPMLSM elements shown in Figure 3(a). Then the equation of each element is represented by a set of linear equations using the Lagrange method. A global matrix is formed from the assembly of linear equations of all the elements in accordance with the continuity of magnetic potentials. The distribution of **A** can be obtained by solving the global matrix finally.

The three-phase AC input current density **J** RMS is about $4.8 \times 10^6\ \mathrm{A\,m^{-2}}$, arranged in A–A′–B′–B–C–C′–A′–A–B–B′–C′–C order twice for 24-slot 20-pole TPMLSM. Since one pole-pitch distance equal to 180 electrical-angle degrees is divided into 18 sections, it is defined that the moving distance of each step covers 10 electrical-angle degrees. Meanwhile, the current phase turns 10 degrees per step. To analyse the force along the moving direction, the air between mover and stator is divided into two parts. The contact elements between these two parts should be coupled correctly at each step (ANSYS Inc. 2009).

To solve nonlinear electromagnetic analysis problems, the preconditioned conjugate gradient (PCG) method is applied based on the magnetic vector potential (ANSYS Inc. 2009). The convergence tolerance is set to 0.001. Then magnetic forces during the moving processing over one pole pitch can be sampled at 18 steps and calculated one by one. For example, at the first step, the magnetic field distribution is reduced, and its flux linkage is shown in Figure 3(c). According to the distribution, the force can be calculated by a virtual work method, shown as the coordinates that the text arrow 'Step 1' pointed in Figure 3(b). Then the coupling of the contact elements is deleted. After the mover moves one eighteenth pole-pitch distance for the next step, the contact elements are recoupled. The magnetic field density is resolved and its flux linkage is shown in Figure 3(d), and the force is shown as the coordinates that the text arrow 'Step 2' points to in Figure 3(b). The processing will be repeated until the moving distance achieves one pole pitch. Finally, the forces are obtained after 18 steps, as shown in Figure 3(b).

Obviously, the forces along the moving direction fluctuate, which is caused by both the end effect and the cogging force. To quantify the motor performance, the average of the forces is defined as the thrust F_0 and the standard deviation is defined as the ripple F_r.

Under the same conditions of exciting current and structural materials, the thrust F_0 and the ripple F_r are determined by the structural dimensions simultaneously. δ determines the magnetic density; the permanent magnet width τ_m and the height h determine the magnetic field and PM volume; ω_s and d_s determine the winding volume; τ_e affects the end effect; and d_c affects both the PM volume and the winding volume. Because $d_s = R - (r_0 + d_c + d_h + \delta + d_a)$, the thrust F_0 and the ripple F_r can be described by six variables as follows:

$$\left.\begin{aligned} F_0 &= f_0(\delta, \tau_m, d_h, \omega_s, \omega_e, d_c) \\ F_\mathrm{r} &= f_r(\delta, \tau_m, d_h, \omega_s, \omega_e, d_c). \end{aligned}\right\} \tag{7}$$

FEA can solve the problem with high accuracy; however, it always takes too much time for each calculation, about 430 seconds using a Xeon 2.8 GHz CPU with 4 Gb memory workstation. Because thrust the F_0 and the ripple F_r should be calculated repeatedly for each iteration during the procedure of optimization, it will take an extremely long time. So it is very useful to establish a response surface to replace FEA.

2.4. *Artificial neural network response surface*

Response describes the performance measure or quality characteristic of a system. Response surface (RS), which means the relationship between the system input variables and responses, has been widely used for developing, improving and optimizing the processes of structural design.

The polynomial-based response surface method (RSM) can be used to generate a response surface from the fitting of a polynomial equation by using experimental data. The commonly used second order response surface can be written as follows:

$$y = \sum_{j=1}^{k} \beta_{jj} x_j^2 + \sum_{i \neq j}^{k} \beta_i \beta_j x_i x_j + \sum_{j=1}^{k} \beta_j x_j + \beta_0 + \varepsilon, \tag{8}$$

where y is the true response with variables x_i $(i = 1, \ldots, k)$, β is the regression coefficient, and ε denotes the statistical error, which should be minimized by estimating the unknown coefficients (Myers *et al.* 2009).

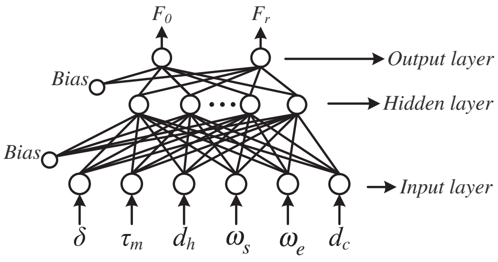


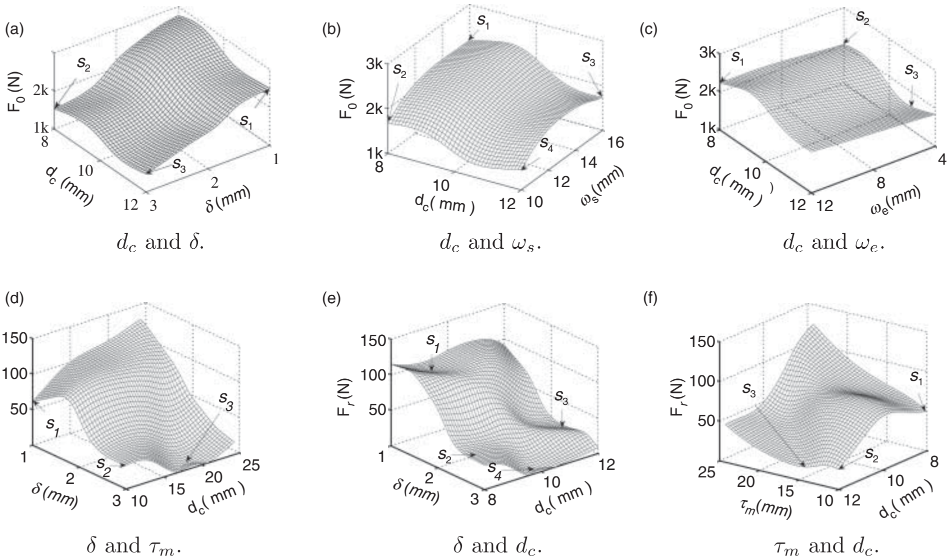
Figure 4. The architecture of the artificial neural network.

However, the polynomial-based method for the response surface suffers from complexity and unacceptable error for high-dimensional problems. Owing to the advantage of approximating nonlinear functions, an artificial neural network (ANN) is adopted to overcome these disadvantages. Among different ANN models, the multi-layer perception (MLP) model, which is capable of learning a rich variety of nonlinear decision surfaces, is adopted in this application. The architecture is shown in Figure 4. The inputs of the ANN are the variables δ, τ_m, d_h, ω_s, ω_e and d_c. The logistic sigmoid transfer function is used to transfer the values of the six input layer neurons to the twenty hidden layer neurons, whereas the linear transfer function is adopted to transfer the values from the hidden layer to the output layer with two neurons. Finally the output, which consists of the thrust F_0 and the ripple F_r is deduced. An improved Levenberg–Marquardt training algorithm is applied to minimize the root-mean-square error of the network output (Beale *et al.* 2011).

Samples are generated first to train ANN with respect to each combination of design variables. 300 latin hypercube samples are generated by *lhsdesign* according to the *maximin* criterion (The MathWorks Inc. 2011). Moreover, the other 2^6 boundary vertices are also chosen as samples. The thrust F_0 and the ripple F_r of each sample are calculated by FEA. Then the ANNRS is trained. To validate its accuracy, another 50 test samples are generated by *lhsdesign*. These test samples are simulated with the ANNRS to calculate the thrusts and ripples, which are also calculated by FEA at the same time. For ANNRS, in comparison with FEA, its average absolute error for the thrust F_0 is 0.4651 N with maximum 1.7336 N, and its average absolute error for the ripple F_r is 0.5306 N with maximum 1.0435 N. This means that ANNRS can achieve reasonable accuracy as a substitute for FEA.

It takes about 43.48 hours to calculate the 364 training samples, about 4 minutes to train the ANNRS. For population-based metaheuristics algorithms, for example 100 particles of a PSO swarm updating for 500 iterations, one needs to evaluate the objectives 50 thousand times. It will take only 23.28 seconds to evaluate the objectives by ANNRS for these times. However, about 8.29 months will be needed by FEA. Obviously, ANNRS is easily the more acceptable for engineering projects.

Given the values of the design variables, ANNRS outputs the thrust F_0 and the ripple F_r, *i.e.* its implementation is ‘opaque’. ANNRS can be viewed solely in terms of transfer characteristics without any knowledge of its internal workings. So ANNRS is a black-box model. To study the transfer characteristics of ANNRS, some variations of thrust F_0 and ripple F_r as functions of different variables are shown in Figure 5 as examples. In all subfigures, the other four unstated variables are set to the median values within their ranges. F_0 has three minima with respect to d_c and δ, four with respect to d_c and ω_s, and three with respect to d_c and ω_e. F_r has three minima with respect to δ and τ_m, four with respect to δ and d_c, and three with respect to τ_m and d_c. Consequently, ANNRS is equivalent to a multi-modal nonlinear function with respect to dependent design variables.



d_c and δ.

d_c and ω_s.

d_c and ω_e.

δ and τ_m.

δ and d_c.

τ_m and d_c.

Figure 5. The variation of the ripple F_r as a function of different variables.

3. Decomposition-based multi-objective differential evolution particle swarm optimization

3.1. *Decomposition-based selection mechanism*

A new decomposition-based approach was proposed by Zhang and Li (2007) to decompose an MOP into a number of different single-objective optimization subproblems (SOSs) and define neighbourhood relations among these SOSs. In this application, the Tchebycheff approach is applied as the decomposition method on an MOP, then an SOS with a weight vector $\mathbf{\Lambda}$ is deduced as below:

$$\begin{aligned} \text{minimize} \quad & g(\mathbf{x}|\mathbf{\Lambda}, \mathbf{z}^*) = \max_{1\le j\le m} \left\{\lambda_j |f_j(\mathbf{x}) - z_j^*|\right\} \\ & \text{s.t. } \mathbf{x} \in \mathbf{\Omega}, \end{aligned} \tag{9}$$

where $\mathbf{\Lambda} = (\lambda_1, \ldots, \lambda_m)$, $\lambda_j \ge 0$ for all $j = 1, \ldots, m$, and $\sum_{j=1}^m \lambda_j = 1$. $\mathbf{z}^* = (z_1^*, \ldots, z_m^*)$ is the reference point, *i.e.* z_j^* is the smallest objective value of f_j that the swarm has found so far.

It is known that for each Pareto optimal solution $\mathbf{x}^*$, there exists a weight vector $\mathbf{\Lambda}$ such that $\mathbf{x}^*$ is the optimal solution of SOS (9), and the optimal solution of each SOS (9) is a Pareto optimal solution of MOP (1). Let $\{\mathbf{\Lambda}^1, \ldots, \mathbf{\Lambda}^i, \ldots, \mathbf{\Lambda}^N\}$ be a set of weight vectors. Correspondingly, there exist N SOSs. In the optimization process, the ith SOS is associated with the ith weight vector $\mathbf{\Lambda}^i$ consistently. Finally, if N is reasonably large and the set $\{\mathbf{\Lambda}^1, \ldots, \mathbf{\Lambda}^N\}$ is properly selected, the N optimal solutions of these N SOSs (9) will provide a good approximation to the Pareto solutions of MOP (1).

Because there exist disparately scaled objectives for many real problems, $g(\mathbf{x}|\mathbf{\Lambda}, \mathbf{z}^*)$ is redefined to normalize the objectives as follows:

$$g(\mathbf{x}|\mathbf{\Lambda}, \mathbf{z}^*) = \max_{1\le j\le m} \left\{\lambda_j \left|\frac{f_j(\mathbf{x}) - z_j^*}{z_j^{\text{nad}} - z_j^*}\right|\right\}, \tag{10}$$

where z_j^{nad} is the nadir point in the objective space, approximated to the largest value of f_j evaluated by the current positions of all the particles in the swarm.

3.2. *Particle swarm optimization*

Since the original version of the particle swarm optimization (PSO) algorithm was introduced by Kennedy and Eberhart (1995) for SOPs, PSO has gained wide application as an optimization tool to facilitate engineering design (Panigrahi *et al.* 2011). The particle swarm optimizer used here is a variant of PSO with the constriction factor (PSO-cf) (Clerc and Kennedy 2002) shown as follows:

$$\left.\begin{aligned} v_j^i(k+1) &= \chi \cdot [v_j^i(k) + c_1 \cdot r_1 \cdot (pb_j^i(k) - p_j^i(k)) + c_2 \cdot r_2 \cdot (nb_j^i(k) - p_j^i(k))] \\ p_j^i(k+1) &= p_j^i(k) + v_j^i(k+1), \end{aligned}\right\} \tag{11}$$

where the superscript i indicates the ith particle ($i = 1, 2, \ldots, PS$; PS is the population size); the subscript j indicates the jth dimension ($j = 1, 2, \ldots, m$; m is the particle dimension, *i.e.* the dimension of the search space); and the index k indicates the kth generation. $\mathbf{p}^i = \left[p_1^i, p_2^i, \ldots, p_m^i\right]^{\mathrm{T}}$ and $\mathbf{v}^i = \left[v_1^i, v_2^i, \ldots, v_m^i\right]^{\mathrm{T}}$ represent the position and the velocity of the ith particle respectively. $\mathbf{pb}^i = \left[pb_1^i, pb_2^i, \ldots, pb_m^i\right]^{\mathrm{T}}$ is the personal best position found so far by the ith particle. $\mathbf{nb}^i = \left[nb_1^i, nb_2^i, \ldots, nb_m^i\right]^{\mathrm{T}}$ is the best position found by the particles in the neighbourhood of the ith particle. r_1 and r_2 are two random numbers in the range [0,1]. c_1 and c_2 are two acceleration coefficients, and both of them are set to $c_1 = c_2 = 2.05$; χ is the constriction factor, and a common setting of this parameter is $\chi = 0.7298$ (Clerc and Kennedy 2002).

For any particle, all dimensions of its position will be confined between corresponding lower bound p_j^l and upper bound p_j^u. In cases of bound violation, the position and the velocity are modified by

$$\left.\begin{array}{ll} p_j^i = p_j^l \text{ and } v_j^i = 0 & \text{if } (p_j^i < p_j^l) \\ p_j^i = p_j^u \text{ and } v_j^i = 0 & \text{if } (p_j^i > p_j^u). \end{array}\right\} \tag{12}$$

3.3. *Differential evolution*

Differential evolution proposed by Storn and Price (1997) has been proved very efficient and robust in function optimization and applied to solve problems in many scientific and engineering fields (Qing 2009), and a recent survey on DE is provided by Das and Suganthan (2011).

The most classical DE variant is DE/rand/1/bin. In this DE variant, for the mutation of the ith individual in the DE population $\{\mathbf{p}^i | i = 1, 2, \ldots, PS\}$, three different individuals $\mathbf{p}^{r_1}$, $\mathbf{p}^{r_2}$ and $\mathbf{p}^{r_3}$ with $r_1 \neq r_2 \neq r_3 \neq i$ will be chosen randomly from the population to generate a new vector. The new vector can be expressed as follows:

$$\mathbf{z}^i = \mathbf{p}^{r_1} + F \cdot (\mathbf{p}^{r_2} - \mathbf{p}^{r_3}), \tag{13}$$

where F is the so-called scaling factor and a general setting for this factor is $F \in (0.4, 0.95)$. For optimization with the nonlinear multi-modal black-box ANNRS, a good initial choice of F is 0.9 (Ronkkonen *et al.* 2005). After mutation, a binominal (bin) crossover operates on the vector $\mathbf{z}^i$ and the target vector $\mathbf{p}^i$ to generate the final vector $\mathbf{u}^i$ in the following way:

$$u_j^i = \begin{cases} z_j^i & \text{if } (\mathrm{rand}_j^i \leq C_r \text{ or } j = rn^i) \\ p_j^i & \text{otherwise,} \end{cases} \tag{14}$$

where p^i_j, z^i_j and u^i_j are the jth dimension components of the vectors $\mathbf{p}^i$, $\mathbf{z}^i$, and $\mathbf{u}^i$, respectively; rn^i is a number that is randomly selected from the index set $\{1, 2, \ldots, m\}$, ensuring that the trial vector $\mathbf{u}^i$ is different from the target solution $\mathbf{p}^i$. C_r is the predefined crossover probability. Ronkkonen *et al.* (2005) also pointed out that C_r should lie in $(0, 0.2)$ when the function is separable, while in $(0.9, 1)$ when the function's parameters are dependent. Due to the multi-modal ANNRS with respect to the dependent design variables, a good choice of C_r is 0.9.

Finally, $\mathbf{u}^i$ will be compared with $\mathbf{p}^i$, and the better one will be selected to be a member of the DE population for the next generation.

3.4. *Decomposition-based multi-objective differential evolution particle swarm optimization*

Because the optimal solution of each SOS is a Pareto optimal solution of the MOP, decomposition-based multi-objective differential evolution particle swarm optimization (DMDEPSO) attempts to optimize the N SOSs decomposed from an MOP simultaneously. In DMDEPSO, the neighbourhood of $\mathbf{\Lambda}^i$ is defined as the T closest weight vectors from $\{\mathbf{\Lambda}^1, \ldots, \mathbf{\Lambda}^N\}$ to a weight vector $\mathbf{\Lambda}^i$. The SOSs with these T closest weight vectors constitute the neighbourhood of the ith SOS with the weight vector $\mathbf{\Lambda}^i$. An underlying assumption for this approach is that the optimal solutions of neighbouring SOSs should be close to each other in the objective space. The current position of a particle can be evolved by DMDEPSO, according to the information from itself, its neighbourhood and the whole swarm.

Zhang and Li (2007) claimed that the decomposition-based algorithm is not very sensitive to the setting of T, and suggested that neighbourhood size T should be much smaller than the population size N. So $N/20$ is rounded to its nearest integer, and T is set to the integer.

Each element of a weight vector takes its value from $\{0/H, 1/H, \ldots, H/H\}$, and the sum of the elements of the weight vector must equal to one. Therefore, there are $N = C^{m-1}_{H+m-1}$ different weight vectors for an MOP with m objectives. Each vector will be associated with one particle consistently, for evaluating the particle's SOS fitness by Equation (10). At each iteration, the DMDEPSO algorithm maintains the following data structures:

- a swarm of N particles, where the ith particle has a position vector $\mathbf{p}^i$, a velocity vector $\mathbf{v}^i$, a father position vector $\mathbf{h}^i$ used for determining whether it stagnates or not, a personal best position vector $\mathbf{pb}^i$, and a neighbourhood best position vector $\mathbf{nb}^i$. N is also the number of SOSs;
- a reference point $\mathbf{z} = (z_1, \ldots, z_m)^{\mathrm{T}}$, a nadir point $\mathbf{z}^{\mathrm{nad}} = (z^{\mathrm{nad}}_1, \ldots, z^{\mathrm{nad}}_m)^{\mathrm{T}}$;
- an external archive ExA, which is used to store non-dominated solutions found during the search process;
- a counter to record the number of continuously stagnating iterations $\mathbf{S}$, and a trigger value s^* used for determining whether the particle is evolved by DE or not.

The DMDEPSO algorithm in pseudocode follows.

```
Step 1: Initialization
        1.1 Set ExA = $\emptyset$, $\mathbf{S} = 0$. Assign proper $H$, then $N = C^{m-1}_{H+m-1}$.
        1.2 Initialize the weight set $\{\mathbf{\Lambda}^1, \ldots, \mathbf{\Lambda}^N\}$, and compute the Euclidean distances between
            any two weight vectors. For each $i = 1, \ldots, N$, set $B(i) = \{i_1, \ldots, i_T\}$ where
            $\mathbf{\Lambda}^{i_1}, \ldots, \mathbf{\Lambda}^{i_T}$ are the $T$ closest weight vectors to $\mathbf{\Lambda}^i$. $T$ is called the neighbourhood
            size.
        1.3 Initialize the position set $\Omega = \{\mathbf{p}^1, \mathbf{p}^2, \ldots, \mathbf{p}^N\}$ of the swarm with $N$ particles being
            randomly generated between the lower bound $\mathbf{p}^l$ and the upper bound $\mathbf{p}^u$.
        1.4 Initialize the velocity of particle $i$ $(i = 1, 2, \ldots, N)$ randomly between $\mathbf{p}^l - \mathbf{p}^i$ and
            $\mathbf{p}^u - \mathbf{p}^i$.
```

1.5 Set the reference point $\mathbf{z} = \min_{1\leq i\leq N}\left\{\mathbf{f}(\mathbf{p}^i)\right\}$ and the nadir point $\mathbf{z}^{\text{nad}} = \max_{1\leq i\leq N}\left\{\mathbf{f}(\mathbf{p}^i)\right\}$.

1.6 Set the initial father position $\mathbf{h}^i$, the personal best $\mathbf{pb}^i$ and neighbourhood best $\mathbf{nb}^i$ to $\mathbf{p}^i$ for each $i = 1, \ldots, N$.

Step 2: Evolve with PSO
For $i = 1, \ldots, N$, according to the personal best $\mathbf{pb}^i$ and the neighbourhood best $\mathbf{nb}^i$, calculate the velocity $\mathbf{v}^i$ and update the position $\mathbf{p}^i$ with Equation (11), and then modify the position $\mathbf{p}^i$ according to Equation (12).

Step 3: Update $\mathbf{z}$, $\mathbf{z}^{\text{nad}}$
For $j = 1, \ldots, m$, set $z_j = \min\left\{z_j, \min\limits_{1\leq i\leq N}\left\{f_j(\mathbf{p}^i)\right\}\right\}$, $z_j^{\text{nad}} = \max\limits_{1\leq i\leq N}\left\{f_j(\mathbf{p}^i)\right\}$.

Step 4: Evolve with DE
For $i = 1, \ldots, N$,

4.1 If $g(\mathbf{p}^i|\mathbf{\Lambda}^i, \mathbf{z}) < g(\mathbf{h}^i|\mathbf{\Lambda}^i, \mathbf{z})$, then set $s_i = 0$, else $s_i = s_i + 1$.

4.2 If $s_i > s^*$, set $r_1 = i$, and randomly select other two different r_2 and r_3 between 1 and N, then generate a solution $\overline{\mathbf{p}}^i$ according to Equation (13), and then a new solution $\mathbf{p}^i$ is produced from $\overline{\mathbf{p}}^i$ with Equation (14).

Step 5: Update $\mathbf{z}$, $\mathbf{z}^{\text{nad}}$
For $j = 1, \ldots, m$, set $z_j = \min\left\{z_j, \min\limits_{1\leq i\leq N}\left\{f_j(\mathbf{p}^i)\right\}\right\}$, $z_j^{\text{nad}} = \max\limits_{1\leq i\leq N}\left\{f_j(\mathbf{p}^i)\right\}$.

Step 6: Update positions
For $i = 1, \ldots, N$,

6.1 (Personal best) If $g(\mathbf{p}^i|\mathbf{\Lambda}^i, \mathbf{z}) \leq g(\mathbf{pb}^i|\mathbf{\Lambda}^i, \mathbf{z})$, then set $\mathbf{pb}^i = \mathbf{p}^i$.

6.2 (Neighbourhood best) For each index $j \in B(i)$, if $g(\mathbf{p}^i|\mathbf{\Lambda}^i, \mathbf{z}) \leq g(\mathbf{nb}^i|\mathbf{\Lambda}^i, \mathbf{z})$, then set $\mathbf{nb}^i = \mathbf{p}^i$.

6.3 (Father position) Set $\mathbf{h}^i = \mathbf{p}^i$.

6.4 (ExA). Remove all the vectors dominated by $\mathbf{F}(\mathbf{p}^i)$ from EA, and add $\mathbf{F}(\mathbf{p}^i)$ into EA if no vectors in EA dominate it.

Step 7: If the stopping criteria are satisfied, then stop and output ExA. Otherwise go to Step 2.

During the process of DMDEPSO, PSO and DE are hybridized for solving all the subproblems which are decomposed from the multi-objective problem. The ith SOS ($i = 1, \ldots, N$) is associated with a constant weight vector $\mathbf{\Lambda}^i$, and its solution is represented by the position of the ith particle of the swarm. In PSO, the ith particle searches for the optimal point of the ith SOS in the solution space by using the best information from itself and its neighbourhood.

Because PSO has the disadvantage of premature convergence, particles are easily trapped into local minima of the nonlinear multi-modal ANNRS. Then, to avoid this phenomenon, DE uses two different particles randomly selected from the whole swarm to evolve the particle if stagnated continuously for s^* iterations. The evolved particle not only can be updated by the DE approach, but also can provide new information from the swarm for the neighbourhoods which include this particle. The hybridization between PSO and DE makes the trade-off between exploration and exploitation more reasonable, so that it is conducive to the search capability of DMDEPSO. Since all the particles search the optimal solutions simultaneously, the PS/PF will be obtained after the algorithm terminates in a single run.

4. Optimization design

In this section, the values of the TPMLSM structural parameters are optimized simultaneously by means of four tasks: enhancing the thrust F_0 (N), restraining the ripple f_r (N), as well as reducing the active PM volume V_{p} (cm^3) and the winding volume V_{c} (cm^3). For the 20-pole 24-slot TPMLSM, the active permanent magnet volume V_{p} is governed by $\pi \times \left[(r_0 + d_c + d_h)^2 - (r_0 + d_c)^2\right] \times (20\tau_m)/1000$, the winding volume V_{c} is approximated to $\pi \times \left[(R - d_a)^2 - (R - d_a - d_s)^2\right] \times (24\omega_s)/1000$. According to the geometric parameters shown in Table 1, it can be deduced that the minimum of V_{p} ($V_{p\ \min}$) is 75.3982 and the maximum ($V_{p\ \max}$) is 659.5837, the minimum of V_{c} ($V_{c\ \min}$) is 950.7716 and the maximum ($V_{c\ \max}$) is 2413.9495.

To illustrate the effectiveness of the hybridization with DE, two other techniques are employed to update the stagnated particles respectively. One adopts a polynomial mutation as shown by Li and Zhang (2009), denoted by DMPMPSO. The other directly selects three particles from the whole swarm to update the stagnated particle by DE which is shown in Section 3.3, denoted by DMWDEPSO.

Also, three other algorithms have been chosen in order to compare their performance with that of the DMDEPSO: dMOPSO (Martínez and Coello Coello 2011), MOEA/D-DE (Li and Zhang 2009), and NSGA-II (Deb *et al.* 2002). dMOPSO and MOEA/D-DE, both based on decomposition, are evolved by PSO and DE respetively. NSGA-II is based on non-dominated sorting and the GA strategy.

The set coverage (Zitzler and Thiele 1999) is often used as a performance measure in real PF unknown situations. For two approximations of the PF, A and B, the set coverage $\zeta(A, B)$ is defined as the percentage of the solutions in B that are dominated by the solutions in A, *i.e.*

$$\zeta(A,B) = \frac{|\{\mathbf{u} \in B | \exists \mathbf{v} \in A : \mathbf{v} \prec \mathbf{u}\}|}{|B|}.$$

$\zeta(A,B)$ is not necessarily equal to $1 - \zeta(A,B)$. $\zeta(A,B) = 1$ means that all solutions in B are dominated by some solutions in A, while $\zeta(A,B) = 0$ implies that no solution in B is dominated by a solution in A. Denote the approximation of the Pareto front that DMDEPSO obtains by **A**, DMPMPSO by **B**, DMWDEPSO by **C**, dMOPSO by **D**, MOEA/D-DE by **E** and NSGA-II by **F**.

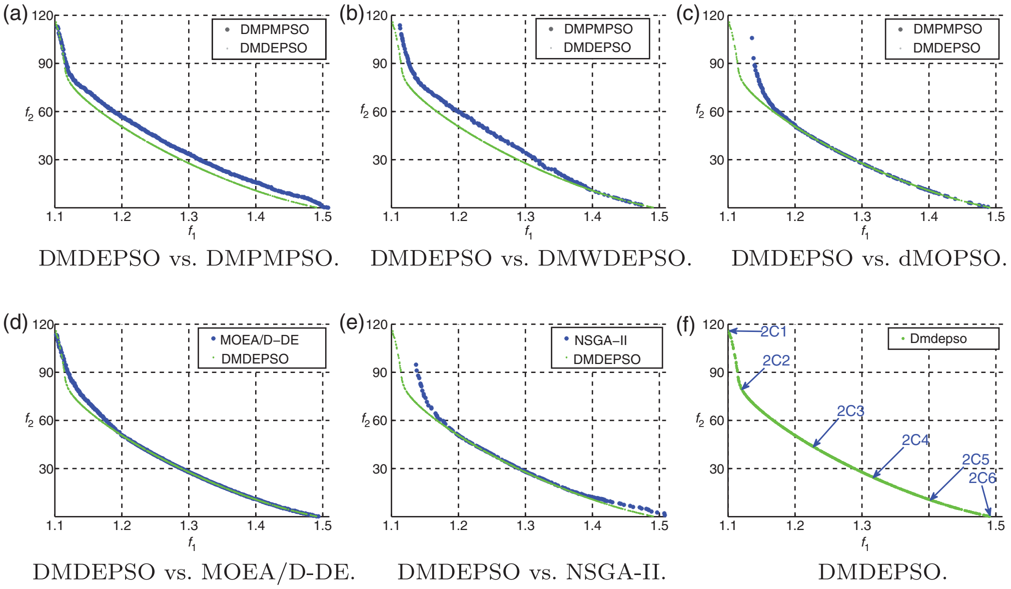
In what follows, optimizations with two objectives and three objectives are first presented separately to show and compare the Pareto fronts visually. Then four-objective optimization is presented. For all the optimizations, each algorithm runs 500 iterations before termination. For DMDEPSO, DMPMPSO and DMWDEPSO, if a particle stagnates continuously for 10 iterations ($s^* = 10$), updating techniques will be applied respectively. To compare the performance of these algorithms, 30 runs are carried out for each algorithm and problem independently.

4.1. *Two-objective optimization*

In order to optimize four tasks simultaneously, two objectives of this optimization are shown in Equation (15). For the objective f_1, V_{p} is multiplied by seven and then added with V_{c} to trade off the V_{c} and V_{p}. The purpose of this operation is to ensure equal weighting on V_{c} and V_{p}. Therefore, F_0, V_{c} and V_{p} are integrated into the objective function f_1. For each algorithm, 200 individuals ($N = 200$) are selected, then the two-objective PFs are obtained.

$$f_1 = \frac{V_{\mathrm{c}} + 7V_{\mathrm{p}}}{F_0}; \quad f_2 = F_{\mathrm{r}}. \tag{15}$$

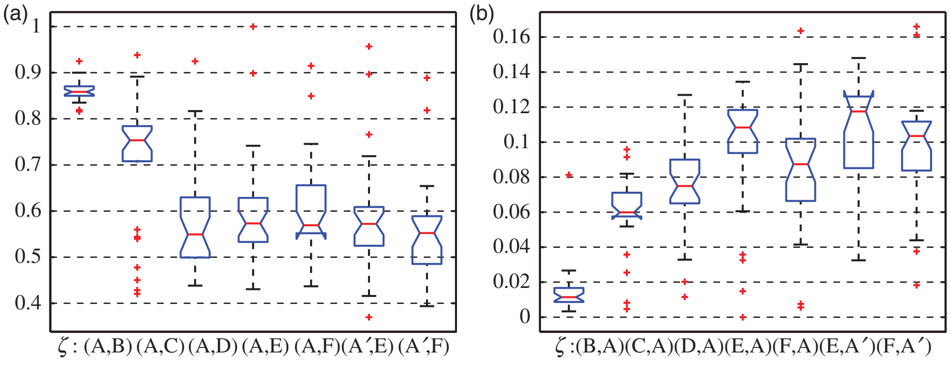
Figure 6 shows the PFs obtained by the first run. The PF indicated by the (light) green dots is obtained by DMDEPSO. It is shown that DMDEPSO has a better performance than the other algorithms for the two-objective optimization problem. And Figure 7 illustrates the statistical results for the set coverage measure of PFs via box-plots. Taking dMOPSO as an example, 54.93% median (the third column shown in Figure 7(a), 43.83% lower adjacent, 49.91% 25th percentile, 63.97% 75th percentile, 81.69% upper adjacent, and one outlier 92.49%) of the final solutions generated



DMDEPSO vs. DMPMPSO. DMDEPSO vs. DMWDEPSO. DMDEPSO vs. dMOPSO.

DMDEPSO vs. MOEA/D-DE. DMDEPSO vs. NSGA-II. DMDEPSO.

Figure 6. Pareto fronts with two objectives obtained by the first run.



DMDEPSO vs. others. Others vs. DMDEPSO.

Figure 7. Set coverage of the two-objective problem.

by dMOPSO are dominated by those generated by DMDEPSO, and only 7.49% median (the third column shown in Figure 7(b), 3.28% lower adjacent, 6.50% 25th percentile, 9.01% 75th percentile, 12.69% upper adjacent, and two outliers, 2.02% and 1.16%) of the final solutions generated by DMDEPSO are dominated by those generated by dMOPSO. This means that $\zeta(\mathbf{A},\mathbf{D}) > \zeta(\mathbf{D},\mathbf{A})$ for each run, and the null hypothesis of equal medians for $\zeta(\mathbf{A},\mathbf{D})$ and $\zeta(\mathbf{D},\mathbf{A})$ is rejected at the 5% significance level according to the Wilcoxon rank sum test. This also applies to all the other comparisons. Generally, the final PF obtained by DMDEPSO is better than that obtained by the other algorithms.

However, updating the stagnated particles increases the ANNRS evaluations. The average of ANNRS evaluations is 105,725.6 times for DMDEPSO, less than 106,217.3 times for DMPMPSO, 106,165.2 times for DMWDEPSO, and 110,215.9 times for dMOPSO, but more than 100,200 times for MOEA/D-DE and NSGA-II consistently. However, it is worth increasing by 5.51% the ANNRS evaluations of MOEA/D-DE or NSGA-II to improve the optimal designs by DMDEPSO.

Table 2. Six design schemes obtained by two-objective optimization.

	Symbol	2C1	2C2	2C3	2C4	2C5	2C6
Parameters	δ (mm)	1.00	1.55	1.85	1.97	2.06	2.17
	τ_m (mm)	23.33	23.33	23.33	23.33	23.33	23.33
	d_h (mm)	2.00	2.00	2.00	2.00	2.00	2.00
	ω_s (mm)	12.39	12.19	12.71	12.66	12.89	13.28
	τ_e (mm)	12.00	11.98	12.00	12.00	12.00	12.00
	d_c (mm)	9.18	9.68	9.69	9.73	9.79	9.92
Tasks	F_0 (N)	2793.00	1675.66	2489.36	2299.91	2187.81	2083.62
	F_{r}(N)	115.69	78.08	43.61	23.57	10.54	0.03
	V_{c} (cm^3)	1797.51	1703.75	1755.64	1737.40	1760.00	1796.54
	V_{p} (cm^3)	182.82	185.73	185.82	186.05	186.39	187.22

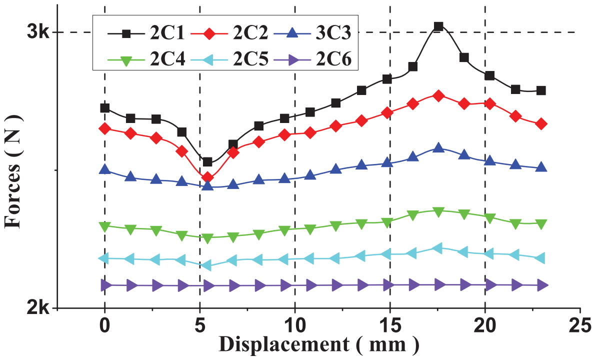


Figure 8. Forces of the six selected optimal designs with two objectives.

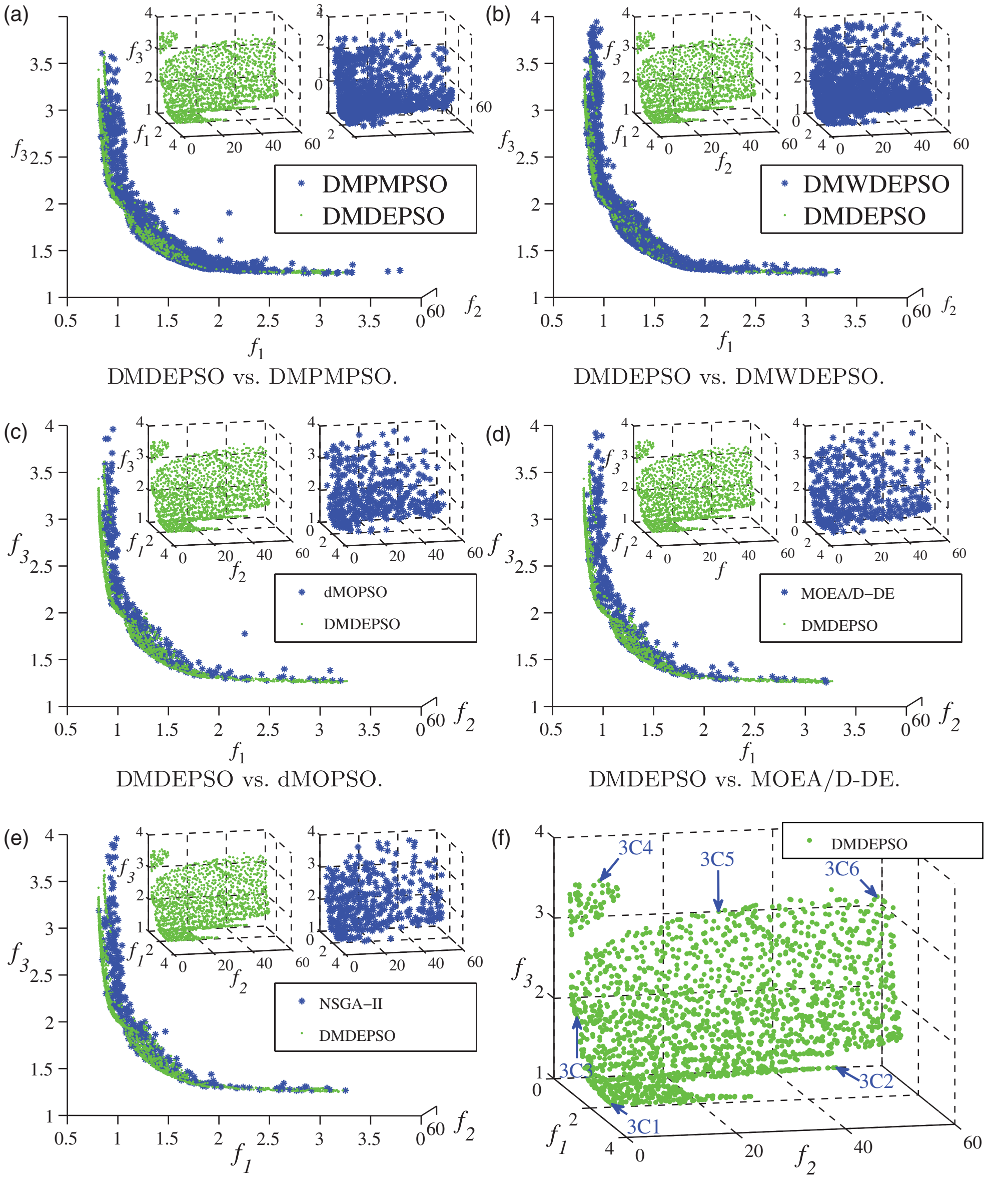
For comparing the performance of DMDEPSO, MOEA/D-DE and NSGA-II by the same number of evaluations, DMDEPSO immediately breaks out of the searching loop and outputs ExA when ANNRS evaluates 100,200 times. Denote the Pareto front that DMDEPSO outputs by $\mathbf{A}'$. DMDEPSO also runs 30 times, then the statistical results on set coverage are deduced and shown in Figure 7. The results show that DMDEPSO performs better than MOEA/D-DE and NSGA-II for the two-objective optimization problem with the same number of ANNRS evaluations.

Six optimal designs from the PF shown in Figure 6(f) are listed in Table 2. Then the performance of these optimal designs is evaluated by FEA, and the forces are shown in Figure 8 respectively.

As shown in Figure 6, the largest F_{r} is as much as 115.69 N, exceeding the upper boundary of the design demand. The range of f_1 is between 1.10 and 1.49. This means that the Pareto solutions are crowded within a small range. Optimization with the aggregative objective f_1 cannot guarantee that F_0, V_{c} and V_{p} satisfy the demands independently. For example, V_{c} and V_{p} have little difference among these Pareto optimal solutions listed in Table 2. Moreover, these solutions are very close to each other except for δ. To enhance the diversity of solutions, the thrust and volumes should be considered separately.

4.2. *Three-objective optimization*

As shown in Table 2, F_0 is near 2000 N. Therefore, for the three-objective optimization problem presented in Equation‘(16), F_0 and F_{r} are optimized simultaneously with the total volume of V_{c} and V_{p}. For each algorithm, 351 individuals ($N = 351$) are selected. Finally, the three-objective



DMDEPSO vs. DMPMPSO.

DMDEPSO vs. DMWDEPSO.

DMDEPSO vs. dMOPSO.

DMDEPSO vs. MOEA/D-DE.

DMDEPSO vs. NSGA-II.

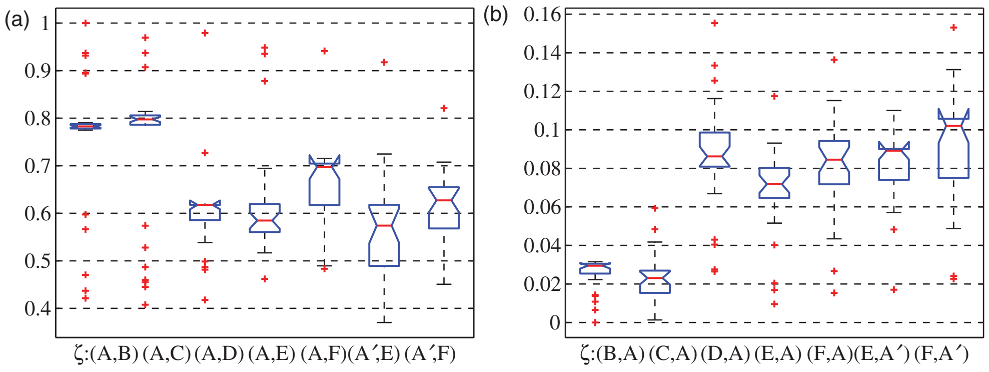
DMDEPSO.

Figure 9. Pareto fronts with three objectives obtained by the first run.

PFs are obtained, and the solutions for which F_r is smaller than 60 N are presented in Figure 9:

$$f_1 = \frac{2000}{F_0};\quad f_2 = F_r;\quad f_3 = \frac{V_c + 7V_p}{V_{c\,\min} + 7V_{p\,\min}}. \tag{16}$$

Figure 9 shows the PFs obtained by the first run. The PF indicated by the (light) green dots is obtained by DMDEPSO. Figure 10 illustrates the statistical results for the set coverage measure of PFs via box-plots. From them, it is known that DMDEPSO has a better performance than the other algorithms for the three-objective optimization problem. The average of ANNRS evaluations is 182,238.9 times for DMDEPSO, 188,218.3 times for DMPMPSO, 186,812.1 times for



DMDEPSO vs. others.

Others vs. DMDEPSO.

Figure 10. Set coverage of three-objective problem.

Table 3. Six design schemes obtained by three-objective optimization.

	Symbol	3C1	3C2	3C3	3C4	3C5	3C6
Parameters	δ (mm)	2.80	1.35	2.62	2.92	2.13	1.95
	τ_m (mm)	10.94	10.94	22.01	23.16	23.23	23.33
	d_h (mm)	2.00	2.00	2.09	5.66	4.39	5.28
	ω_s (mm)	10.00	10.00	12.97	12.88	13.64	13.18
	τ_e (mm)	4.73	12.00	7.37	11.99	12.00	11.84
	d_c (mm)	12.00	11.99	11.95	10.26	10.65	10.48
Tasks	F_0 (N)	643.53	1221.40	1482.67	2351.75	2686.12	3060.82
	F_r(N)	0.14	46.16	0.02	5.55	27.67	56.91
	V_c (cm^3)	1203.24	1285.78	1571.34	1395.86	1612.58	1516.99
	V_p (cm^3)	93.48	93.44	196.16	562.11	433.41	528.16

DMWDEPSO, and 190,029.4 times for dMOPSO, but only 175,851 times for MOEA/D-DE and NSGA-II consistently.

The Pareto front denoted by $\mathbf{A}'$ is immediately output by DMDEPSO when ANNRS evaluates 175,851 times. The statistical results for set coverage are shown in Figure 10. The results also show that DMDEPSO performs better than MOEA/D-DE and NSGA-II for the three-objective optimization problem with the same number of ANNRS evaluations.

Table 3 lists six optimal designs from the PF shown in Figure 9(f). Then the performance of these optimal designs is evaluated by FEA, and the forces are shown in Figure 11 respectively.

From Table 3, it is known that the values of F_0 cover a wide range while F_r is smaller than 60 N. Because f_3 is integrated from V_c and V_p, V_c and V_p cannot be known directly. So in the next section, V_c and V_p are analysed separately to design more reasonable sizes of coils and PMs.

4.3. *Four-objective optimization*

To optimize four tasks simultaneously with four objectives, the optimization problem is presented in Equation (17). Let $N = 455$ for each algorithm. After all algorithms are executed 30 times, several PFs are obtained:

$$f_1 = \frac{2000}{F_0}; \quad f_2 = F_r; \quad f_3 = \frac{V_c}{V_{c\,\min}}; \quad f_4 = \frac{V_p}{V_{p\,\min}}. \tag{17}$$

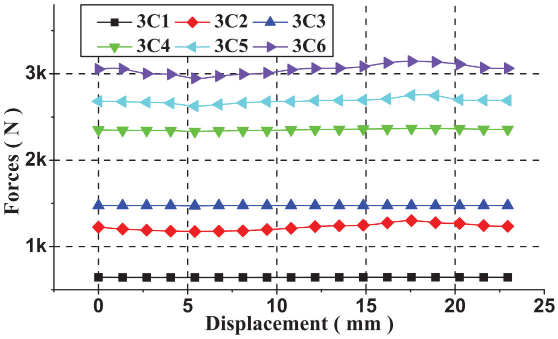


Figure 11. Forces of the six selected optimal designs with three objectives.

Figure 10 illustrates the statistical results for the set coverage measure of PFs via box-plots. It also shows that DMDEPSO has the best performance compared with the other algorithms for the four-objective optimization problem. The average of ANNRS evaluations is about 234,431 times for DMDEPSO, 240,981.7 times for DMPMPSO, 238,981.1 times for DMWDEPSO, and 243,531.5 times for dMOPSO, but only 227,955 times for MOEA/D-DE and NSGA-II consistently.

The Pareto front denoted by $\mathbf{A}'$ is immediately output by DMDEPSO when ANNRS evaluates 227,955 times. Figure 12 shows the statistical results for set coverage. It is known that DMDEPSO also outperforms MOEA/D-DE and NSGA-II for the four-objective optimization problem with the same number of ANNRS evaluations.

There exist eleven solutions with F_0 larger than 3000 N and F_r smaller than 60 N from the PF obtained by DMDEPSO, and six of them are listed in Table 4. Then performance of these optimal designs is evaluated by FEA, and the forces are shown in Figure 13 respectively.

As listed in Table 4, 4C1 and 4C4 have larger V_p and smaller V_c, while 4C2, 4C3 and 4C6 are just the opposite. But the combination of V_c and V_p, presented as f_3 in Equation (16), cannot distinguish the differences of these volumes. So it is necessary to separate them for simultaneous optimization. The larger is the PM volume, the higher the cost of manufacture. Moreover, a large motor winding volume will have the disadvantages of heavy mover mass, high electrical energy cost and extra generated heat. Therefore, 4C5 is chosen as the the final design with proper PM and winding volume.

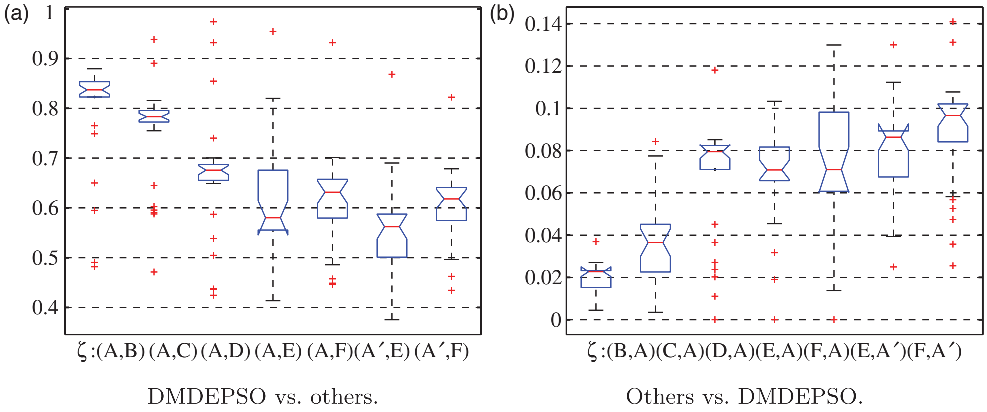


Figure 12. Set coverage of four-objective problem.

Table 4. Six design schemes obtained by four-objective optimization.

	Symbol	4C1	4C2	4C3	4C4	4C5	4C6
Parameters	δ (mm)	1.93	1.74	1.71	2.12	1.70	1.62
	τ_m (mm)	22.80	22.93	22.97	23.33	21.50	22.95
	d_h (mm)	5.12	3.61	3.66	5.27	5.00	3.59
	ω_s (mm)	13.22	14.03	13.48	13.20	13.40	13.46
	τ_e (mm)	11.65	10.52	10.80	11.87	10.74	10.92
	d_c (mm)	10.73	10.02	10.05	10.23	10.50	10.19
Tasks	F_0 (N)	3011.46	3018.02	3020.75	3055.70	3072.21	3102.59
	F_{r}(N)	50.24	59.41	57.31	55.90	59.52	59.09
	V_{c} (cm^3)	1516.12	1799.87	1726.73	1525.61	1581.94	1725.95
	V_{p} (cm^3)	502.68	341.52	346.94	522.77	459.30	341.14

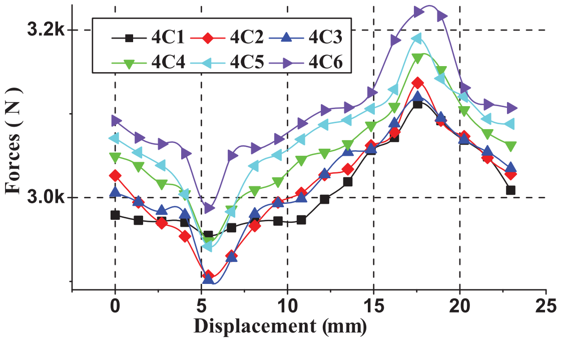


Figure 13. Forces of the six selected optimal designs with four objectives.

5. Conclusions

In this article, a hybrid optimizer based on particle swarm optimization and differential evolution, named DMDEPSO, is proposed for TPMLSM design taking into account multiple objectives including thrust, ripple, PM volume and winding volume. The main ideas in the proposed DMDEPSO can be summarized by the following four aspects.

(i) ANNRS, which is equivalent to a nonlinear multi-modal black-box model, is adopted to evaluate the objectives for reducing the time consumption.
(ii) A decomposition method is used to decompose the MOP into a number of SOSs with normalized objectives. Each SOS is associated with a constant weight vector.
(iii) All the SOSs are simultaneously optimized by population-based evolutionary algorithm, which can find several Pareto optimal solutions simultaneously in a single run.
(iv) The hybridization between PSO and DE contributes to a better tradeoff between exploration and exploitation in which PSO utilizes the information of the particle and its neighbourhood, and DE evolves any continually stagnated particle by two particles randomly chosen from the whole swarm.

TPMLSM is optimized based on DMDEPSO taking account of the different of objectives. Finally, the optimized TPMLSM has desirable PM volume and winding volume, with a thrust larger than 3000 N, which exceeds the requirement, and a ripple lower than 60 N, which meets the requirement. Also, according to the performance comparisons with chosen algorithms, DMDEPSO has shown its excellence for finding the Pareto solutions to the TPMLSM optimization problem. Future work will involve multidisciplinary design optimization taking into consideration multi-physical field coupled phenomena.

Acknowledgements

This work was supported by the National Science Fund for Distinguished Young Scholars under Grant 60925011 and the Projects of Major International (Regional) Joint Research Program NSFC (61120106010).

References

ANSYS Inc., 2009. Low-frequency electromagnetic analysis guide. *In*: *Release 12.0 Documentation for ANSYS*. Available from: http://www1.ansys.com/customer/content/ documentation/120/ans_lof.pdf [Accessed 20 November 2011].

Ashabani, M., Mohamed, Y.A.-R.I., and Milimonfared, J., 2010. Optimum design of tubular permanent-magnet motors for thrust characteristics improvement by combined Taguchi-neural network approach. *IEEE Transactions on Magnetics*, 46 (12), 4092–4100.

Basu, M., 2011. Economic environmental dispatch using multi-objective differential evolution. *Applied Soft Computing*, 11 (2), 2845–2853.

Beale, M.H., Hagan, M.T., and Demuth, H.B., 2011. *Neural network toolbox^{TM} 7 user's guide*, Natick, MA: The MathWorks Inc.

Chen J., *et al.*, 2009. Optimal contraction theorem for exploration–exploitation tradeoff in search and optimization. *IEEE Transactions on Systems, Man, and Cybernetics—Part A: Systems and Humans*, 39 (3): 680–691.

Cheng, J., *et al.*, 2011. Multi-objective ant colony optimization based on decomposition for bi-objective traveling salesman problems. *In*: *Soft computing—a fusion of foundations, methodologies and applications*. Berlin: Springer, 1–18.

Clerc, M. and Kennedy, J., 2002. The particle swarm—explosion, stability, and convergence in a multidimensional complex space. *IEEE Transactions on Evolutionary Computation*, 6 (1), 58–73.

Coello Coello, C.A., 2011a. Evolutionary multi-objective optimization. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 1 (5), 444–447.

Coello Coello, C.A., 2011b. *List of references on evolutionary multiobjective optimization* [online]. México: Instituto Politécnico Nacional. Available from: http://delta.cs.cinvestav.mx/~ccoello/EMOO/EMOObib.html [Accessed 20 November 2011].

Das, S. and Suganthan, P.N., 2011. Differential evolution: a survey of the state-of-the-art. *IEEE Transactions on Evolutionary Computation*, 15 (1), 27–54.

Deb, K., 2001. *Multi-objective optimization using evolutionary algorithms*. New York: Wiley.

Deb, K., *et al.*, 2002. A fast and elitist multiobjective genetic algorithm: NSGA-II. *IEEE Transactions on Evolutionary Computation*, 6 (2), 182–197.

dos Santos Coelho, L., Barbosa, L.Z., and Lebensztajn, L., 2010. Multiobjective particle swarm approach for the design of a brushless DC wheel motor. *IEEE Transactions on Magnetics*, 46 (8), 2994–2997.

Kennedy, J. and Eberhart, R.C., 1995. Particle swarm optimization. *In*: *Proceedings of the 1995 IEEE international conference on neural networks*, 27 November–1 December 1995, Perth, Australia. Vol. 4. Washington, DC: IEEE Computer Society, 1942–1948.

Konstantinidis, A. and Yang, K., 2011a. Multi-objective K-connected deployment and power assignment in WSNs using a problem-specific constrained evolutionary algorithm based on decomposition. *Computer Communications*, 34 (1), 83–98.

Konstantinidis, A. and Yang, K., 2011b. Multi-objective energy-efficient dense deployment in wireless sensor networks using a hybrid problem-specific MOEA/D. *Applied Soft Computing*, 11 (6), 4117–4134.

Konstantinidis, A., *et al.*, 2010. Multi-objective mobile agent-based sensor network routing using MOEA/D. *In*: *Proceedings of the 2010 IEEE congress on evolutionary computation*, 18–23 July 2010, Barcelona, Spain. Piscataway, NJ: IEEE Press, 1–8.

Kotinis, M., 2011. Implementing co-evolution and parallelization in a multi-objective particle swarm optimizer. *Engineering Optimization*, 43 (6), 635–656.

Kulturel-Konak, S. and Konak, A., 2011. A new relaxed flexible bay structure representation and particle swarm optimization for the unequal area facility layout problem. *Engineering Optimization*, 43 (12), 1263–1287.

Li, H., Jiao, Y., and Zhang, L., 2011b. Hybrid differential evolution with a simplified quadratic approximation for constrained optimization problems. *Engineering Optimization*, 43 (2), 115–134.

Li, H. and Zhang, Q., 2009. Multiobjective optimization problem with complicated Pareto sets, MOEA/D and NSGA-II. *IEEE Transactions on Evolutionary Computation*, 13 (2), 284–302.

Li, L., Ma, M., and Chen, Q., 2011a. Analysis and design of moving-magnet-type linear synchronous motor for electromagnetic launch system. *IEEE Transactions on Plasma Science*, 39 (1), 121–126.

Martínez, S.Z. and Coello Coello, C.A., 2011, A multi-objective particle swarm optimizer based on decomposition. *In*: *Proceedings of the 13th annual conference on genetic and evolutionary computation*, 12–16 July 2011, Dublin, Ireland. New York: Association for Computing Machinery, 69–76.

Montazeri-Gh, M., Jafari, S., and Ilkhani, M.R., 2011. Application of particle swarm optimization in gas turbine engine fuel controller gain tuning. *Engineering Optimization*, 44 (2), 225–240.

Myers, R.H., Montgomery, D.C., and Anderson-Cook, C.M., 2009. *Response surface methodology: product and process optimization using designed experiments*. 3rd ed. New York: Wiley.

Nourbakhsh, A., Safikhani, H., and Derakhshan, S., 2011. The comparison of multi-objective particle swarm optimization and NSGA II algorithm: applications in centrifugal pumps. *Engineering Optimization*, 43 (10), 1095–1113.

Panigrahi, B.K., Shi, Y., and Lim, M.H., 2011. *Handbook of swarm intelligence: concepts, principles and applications*. Berlin: Springer.

Qing, A., 2009. *Differential evolution: fundamentals and applications in electrical engineering*. Hoboken, NJ: Wiley.

Rajagopalan, R., *et al.*, 2008. EMOCA: an evolutionary multi-objective crowding algorithm. *Journal of Intelligent Systems*, 17 (1–3), 107–124.

Ramesh, S., Kannan, S., and Baskar, S., 2011. An improved generalized differential evolution algorithm for multi-objective reactive power dispatch. *Engineering Optimization*, 44 (4), 391–405.

Reyes-Sierra, M. and Coello Coello, C.A., 2006. Multi-objective particle swarm optimizers: a survey of the state-of-the-art. *International Journal of Computational Intelligence Research*, 2(3), 287–308.

Ronkkonen J., Kukkonen S., and Price, K.V., 2005. Real parameter optimization with differential evolution. *In*: *Proceedings of the 2005 IEEE congress on evolutionary computation (CEC 2005)*, 2–4 September 2005, Edinburgh. Piscataway, NJ: IEEE Press, 506–513.

Sarikhani, A., and Mohammed, O.A., 2011. Multiobjective design optimization of coupled PM synchronous motor-drive using physics-based modeling approach. *IEEE Transactions on Magnetics*, 47 (5), 1266–1269.

Song, K., 2003. *Magnetic feature curve instant reference of common steel (in Chinese)*. Beijing: China Machine Press.

Storn, R. and Price, K., 1997. Differential evolution—a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization*, 11 (4), 341–359.

The MathWorks Inc., 2011, *Statistics toolbox^{TM} user's guide*. Available from: http:// www.mathworks.com/help/pdf_doc/stats/stats.pdf [Accessed 20 November 2011].

Villarreal-Cervantes, M.G., *et al.*, 2010. Differential evolution techniques for the structure-control design of a five-bar parallel robot. *Engineering Optimization*, 42 (6), 535–565.

Wagner, T., Beume, N., and Naujoks, B., 2007. Pareto-, aggregation-, and indicator-based methods in many-objective optimization. *In*: *Evolutionary multi-criterion optimization*. Lecture notes in computer science 4403. Berlin: Springer, 742–756.

Wang, G., 2011. Slot/pole ratio design of tubular permanent magnet linear synchronous motor. *Materials Science and Engineering*, 179–180 (1), 1303–1308.

Wang, Y., Wu, L., and Yuan, X., 2010. Multi-objective self-adaptive differential evolution with elitist archive and crowding entropy-based diversity measure. *Soft Computing*, 14 (3), 193–209.

Xin, B., *et al.*, 2011. Hybridizing differential evolution and particle swarm optimization to design powerful optimizers: a review and taxonomy. *IEEE Transactions on Systems, Man, and Cybernetics—Part C: Applications and Reviews*, 42 (5), 744–767.

Zhang, Q. and Li, H., 2007. MOEA/D: a multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11 (6), 712–731.

Zitzler, E. and Thiele, L., 1999. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3 (4), 257–271.