DOI: 10.7641/CTA.2013.20502

多目标分解随机粒子群优化算法及其在直线电机优化设计中的应用

王光辉, 陈　杰, 蔡　涛†, 李　鹏

(北京理工大学 自动化学院, 北京 100081; 复杂系统智能控制与决策教育部重点实验室, 北京 100081)

摘要: 本文提出了一种多目标分解随机粒子群优化算法(MDSPSO). 该算法优化过程中, 所有粒子按各自固定的权重向量, 采用改进Tchebycheff分解方法, 将求解多目标非支配解问题转化为求解多个单目标最优解问题; 而后每个粒子在以自身位置、个体历史最优参考位置及群体最优参考位置的几何中心为中心, 以中心到自身位置为半径的区域内, 随机生成一个新的起始位置, 并参考当前的速度更新下一时刻的位置. 通过对测试函数多次计算得到的数据进行统计分析, 表明MDSPSO的收敛性和多样性均优于另外3种对比算法. 最后针对直线电机磁路复杂、有限元计算费时的问题, 使用神经网络拟合直线电机结构参数与性能的关系作为优化设计的模型, 应用MDSPSO算法, 优化结构参数. 实际测试结果表明, 优化后的直线电机推力大、效率高, 同时有效控制了其推力波动和生产成本.

关键词: 多目标优化; 改进Tchebycheff分解方法; 随机粒子群优化算法; 直线电机

中图分类号: TP273　　**文献标识码**: A

A multi-objective decomposition-based stochastic particle swarm optimization algorithm and its application to optimal design for linear motor

WANG Guang-hui, CHEN Jie, CAI Tao†, LI Peng

(School of Automation, Beijing Institute of Technology, Beijing 100081, China;
Key Laboratory of Complex System Intelligent Control and Decision, Ministry of Education, Beijing 100081, China)

Abstract: This article proposes a multi-objective decomposition stochastic particle swarm optimization (MDSPSO) algorithm. In MDSPSO, every particle has a weighted vector constantly. Then, an improved Tchebycheff decomposition method is applied to decompose the multi-objective problem into some single-objective problems. The reference position of every particle is uniformly generated in the zone with the center which is the geometrical center of its current position, the best previous reference position as well as the swarm best reference position. The radius of this zone is the distance from the center to its current position. Then the particle is updated to the new position according to the reference position and its current velocity. The comparisons with the decomposition-based multi-objective particle swarm optimizer (dMOPSO), a multiobjective evolutionary algorithm based on decomposition (MOEA/D), and nondominated sorting genetic algorithm II (NSGA–II) show that the solutions of MDSPSO can be dominated at least with the best diversity. To reduce the computational time by finite element analysis for optimizing the structure parameters of linear motor, artificial neural network is used as the model to evaluate the performance. Finally, MDSPSO is applied to optimize four objectives simultaneously. The practical result is shown that the optimized linear motor has an increased thrust, improved efficiency, reduced fluctuation and manufacturing cost.

Key words: multi-objective optimization; improved Tchebycheff decomposition method; stochastic particle swarm optimization; linear motor

1　引言(Introduction)

许多实际问题是由相互冲突和影响的多个目标组成, 当需要使多个目标在给定的区域尽可能同时最佳时, 也就需要求解多目标优化问题. 智能进化算法具有并行高效、鲁棒性、通用性强等优点, 被广泛应用于求解多目标优化问题[1−2]. 其中粒子群算法以其算法机制简单、性能好、容易实现等优点获得青睐[3]. 文献[4]提出了一种改进多目标粒子群算法优化电弧炉供电过程, 减少了电极消耗, 缩短了冶炼时间, 延长了炉衬使用寿命, 但算法机制复杂; 文献[5]提出了基于分解的多目标粒子群算法, 将分解方法和粒子群算法相结合并取得了良好的性能. 在

收稿日期: 2012−05−13; 收修改稿日期: 2013−02−19.

†通信作者. E-mail: caitao@bit.edu.cn.

基金项目: 国家杰出青年科学基金资助项目(60925011); 国家自然科学基金国家重大国际(地区)合作研究项目(61120106010).

混合有源滤波器多目标优化设计[6]、多目标柔性作业车间调度[7]等实际问题中都采用了改进的粒子群算法算法来作为求解方法.

直线电机已广泛应用于工业设备、交通运输等直线运动场合, 但相对于旋转电机, 直线电机的气隙较大、动定子两端开断, 使得气隙磁场波形畸变、磁路复杂、推力波动明显、损耗增加[8–9]. 直线电机优化设计问题的目的就是优化电机结构参数以提高电机性能, 文献[10–11]中在优化电机设计时, 是根据一个固定的综合指标来评价电机性能, 但实际问题中, 电机性能的各指标事先很难进行判定, 且指标之间是相互矛盾的, 因而需要使用多目标优化算法对各指标同时进行优化, 并获得多个解供设计者参考.

本文首先提出一种多目标分解随机粒子群算法, 随后通过测试函数对算法性能进行验证. 在直线电机优化设计中, 考虑了直线电机的模型特点, 采用神经网络对直线电机进行建模, 最后采用多目标分解随机粒子群优化算法, 对直线电机结构参数进行优化, 提高了电机的推力和效率并降低推力波动和生产成本.

2 多目标分解随机粒子群优化算法(Multi-objective decomposition - based stochastic particle swarm optimization algorithm)

2.1 多目标优化问题(Multi-objective optimization problem)

多目标优化问题的数学形式可以如下描述:

$$\begin{aligned}&\min\ f(\boldsymbol{x})=(f_1(\boldsymbol{x}),\cdots,f_m(\boldsymbol{x}))\,,\\&\text{s.t. }\boldsymbol{x}\in\Omega^n,\end{aligned}\tag{1}$$

其中: $\boldsymbol{x}$为n维决策变量, Ω^n为决策变量的可行解空间, $f(\boldsymbol{x})$为m个目标函数($f_j\colon\Omega^n\to\mathbb{R}^m$, $j=1,\cdots,m$)组成, $\mathbb{R}^m$为目标空间.

通常情况下, 单目标优化问题可以得到一个最优解, 使得目标函数值最小. 但多目标优化问题中, 各个目标之间是矛盾的, 很难有一个解能同时使几个目标达到最优, 为此, 多目标优化问题通常是寻找Pareto最优. 首先定义如下:

定义 1 已知向量$\boldsymbol{u}=(u_1,\cdots,u_m)$, $\boldsymbol{v}=(v_1,\cdots,v_m)$, 当且仅当对$\forall j\in\{1,\cdots,m\}$, $u_j\leqslant v_j$且$\boldsymbol{u}\neq\boldsymbol{v}$时, 则称$\boldsymbol{u}$支配$\boldsymbol{v}$, 记为$\boldsymbol{u}\prec\boldsymbol{v}$.

定义 2 已知问题(1)的可行解$\boldsymbol{x}^*\in\Omega^n$, 当且仅当$\nexists\boldsymbol{y}\in\Omega^n$使得$f(\boldsymbol{y})\prec f(\boldsymbol{x}^*)$, 则可行解$\boldsymbol{x}^*$为Pareto最优解. 所有Pareto最优解的集合称为Pareto解集(PS), 即

$$\text{PS}=\{\boldsymbol{x}\in\Omega^n|\nexists\boldsymbol{y}\in\Omega^n,f(\boldsymbol{y})\prec f(\boldsymbol{x})\}\,.$$

Pareto解集(PS)所对应的目标空间中目标函数值为Pareto前沿(PF), 即

$$\text{PF}=\{f(\boldsymbol{x})|\boldsymbol{x}\in\text{PS}\}.$$

多目标优化问题的求解目标是尽可能全面地找到Pareto最优解, 同时使得当前Pareto前沿靠近并充分地覆盖真实的Pareto前沿.

2.2 多目标分解(Multi-objective decomposition)

文献[12]提出一种Tchebycheff分解方法, 将多目标优化问题(1)转化为多个单目标优化问题进行求解, 本文在此基础上提出一种改进Tchebycheff分解方法, 如式(2)所示:

$$\begin{aligned}&\min T_{\text{s}}(f(\boldsymbol{x})|(\boldsymbol{\Lambda},\boldsymbol{z}^*,\boldsymbol{z}^\dagger))=\\&\rho\sum_{j=1}^{m}\{\lambda_j(\frac{f_j(\boldsymbol{x})-z_j^*}{z_j^\dagger-z_j^*})\}+\max_{1\leqslant j\leqslant m}\{\lambda_j(\frac{f_j(\boldsymbol{x})-z_j^*}{z_j^\dagger-z_j^*})\},\end{aligned}\tag{2}$$

其中: $\boldsymbol{\Lambda},\boldsymbol{z}^*,\boldsymbol{z}^\dagger$为已知常数; $\boldsymbol{\Lambda}=(\lambda_1,\cdots,\lambda_m)$为权重向量, $\forall\lambda_j\geqslant 0,\sum\limits_{j=1}^{m}\lambda_j=1$; $\boldsymbol{z}^*=(z_1^*,\cdots,z_m^*)$, $\boldsymbol{z}^\dagger=(z_1^\dagger,\cdots,z_m^\dagger)$分别为优化参考点及归一化参考点, 对$\forall j$, $z_j^*=\min\limits_{\boldsymbol{x}\in\Omega^n}\{f_j(\boldsymbol{x})\}$, $z_j^\dagger=\max\limits_{\boldsymbol{x}\in\Omega^n}\{f_j(\boldsymbol{x})\}$, 可见对于最小化优化问题, PF是离优化参考点$\boldsymbol{z}^*$最近的可行解目标空间边界, 而归一化参考点$\boldsymbol{z}^\dagger$用来将不同范围的$m$维目标空间, 压缩到同一范围内再进行比较; ρ为较小的非零正实数, 用于保证解的支配关系.

引理 1 对$\forall\boldsymbol{y}^1,\boldsymbol{y}^2\in\mathbb{R}^m$, 若对$j=1,\cdots,m$有$y_j^1\leqslant y_j^2$且$\boldsymbol{y}^1\neq\boldsymbol{y}^2$, 则$T_{\text{s}}(\boldsymbol{y}^1)<T_{\text{s}}(\boldsymbol{y}^2)$.

显而易见, 此引理1成立.

定理 1 若$\boldsymbol{x}^*\in\Omega^n$是单目标优化问题(2)的最优解, 则$\boldsymbol{x}^*$是多目标优化问题(1)的Pareto最优解.

证 反证法. 若$\boldsymbol{x}^*$不为Pareto最优解, 由定义2可知$\exists\boldsymbol{x}'$使得$f(\boldsymbol{x}')\prec f(\boldsymbol{x}^*)$. 由定义1可知, 对$\forall j\in\{1,\cdots,m\}$, $f_j(\boldsymbol{x}')\leqslant f_j(\boldsymbol{x}^*)$且$f_j(\boldsymbol{x}')\neq f_j(\boldsymbol{x}^*)$. 由引理1可知, $T_{\text{s}}(f_j(\boldsymbol{x}'))<T_{\text{s}}(f_j(\boldsymbol{x}^*))$, 即$\boldsymbol{x}^*$不是单目标优化问题(2)最优解, 与假设矛盾.

定理 2 若$\boldsymbol{x}^*\in\Omega^n$是多目标优化问题(1)的Pareto最优解, 则$\exists\boldsymbol{\Lambda}^*=(\lambda_1^*,\cdots,\lambda_m^*)$, 使得$\forall\boldsymbol{x}\in\Omega^n$, $T_{\text{s}}(f(\boldsymbol{x}^*))\leqslant T_{\text{s}}(f(\boldsymbol{x}))$, 其中$\forall\lambda_j^*\geqslant 0,\sum\limits_{j=1}^{m}\lambda_j^*=1$, $j=1,\cdots,m$.

证 假定$\nexists\boldsymbol{\Lambda}^*$, 有$\forall\boldsymbol{x}\in\Omega^n$, $T_{\text{s}}(f(\boldsymbol{x}^*))\leqslant T_{\text{s}}(f(\boldsymbol{x}))$成立, 则对$\forall\boldsymbol{\Lambda}=(\lambda_1,\cdots,\lambda_m)$, 有$\exists\boldsymbol{x}'\in\Omega^n$, $T_{\text{s}}(f(\boldsymbol{x}'))<T_{\text{s}}(f(\boldsymbol{x}^*))$成立, 其中$\forall\lambda_j\geqslant 0,\sum\limits_{j=1}^{m}\lambda_j=1$. 由式(2)可知对$\forall j$, $f_j(\boldsymbol{x}')<f_j(\boldsymbol{x}^*)$, 即$f(\boldsymbol{x}')\prec f(\boldsymbol{x}^*)$, $\boldsymbol{x}^*$可以被$\boldsymbol{x}'$支配, 与假设矛盾.

注 1 定理1指出单目标优化问题(2)的最优解即为多目标优化问题(1)的Pareto最优解. 定理2表明求解多目标优化问题(1)的Pareto最优解总可以转化为某个单目标优化问

题(2)来进行求解. 多目标分解优化算法就是将求解多目标优化问题(1)转化为同时求解多个单目标优化问题(2), 从而使得算法一次运行即可获得多个Pareto最优解.

2.3 随机粒子群优化算法(Stochastic particle swarm optimization algorithm)

经典粒子群优化算法首先在可行解空间Ω^n随机初始化一粒子群体, 每个粒子每一维有具有一个初始随机位置和随机速度, 然后粒子通过参考个体历史最优和群体最优, 不断迭代更新粒子的位置, 直到满足条件时停止并输出最优解.

$$\underbrace{p_j^i(t+1)}_{\text{下一刻位置}} = \underbrace{p_j^i(t)}_{\text{当前位置}} + \underbrace{\omega\cdot v_j^i(t)}_{\text{当前速度参考}} + \underbrace{c_1\cdot r_1\cdot (pb_j^i(t)-p_j^i(t))}_{\text{个体历史最优参考}Gb_j^i} + \underbrace{c_2\cdot r_2\cdot (pg_j^i(t)-p_j^i(t))}_{\text{群体最优参考}Gg_j^i}, \tag{3}$$

其中: 上标i对应于第i个个体($i=1,\cdots,N$, N为种群规模), 下标j对应于粒子的第j维($j=1,\cdots,n$), t表示当前迭代的代数; $\boldsymbol{P}^i=[p_1^i\ \cdots\ p_n^i]^{\mathrm{T}}$和$\boldsymbol{V}^i=[v_1^i\ \cdots\ v_n^i]^{\mathrm{T}}$分别表示第$i$个粒子的位置和速度, 其中$|v_j^i|\leqslant V_{\max}$; ω为惯性因子; $\boldsymbol{Pb}^i=[pb_1^i\ \cdots\ pb_n^i]$表示第$i$个粒子的个体历史最优位置; $\boldsymbol{Pg}^i=[pg_1^i\ \cdots\ pg_n^i]^{\mathrm{T}}$表示第$i$个粒子的群体最优位置; r_1和r_2为[0, 1]内的随机数; c_1和c_2分别作为个体历史最优与群体最优的加速因子来计算个体历史最优参考Gb_j^i和群体最优参考Gg_j^i.

当确定了参考信息$\boldsymbol{Gb}^i=[Gb_1^i\ \cdots\ Gb_n^i]^{\mathrm{T}}$和$\boldsymbol{Gg}^i=[Gg_1^i\ \cdots\ Gg_n^i]^{\mathrm{T}}$, 结合当前的速度参考信息, 粒子$i$下一刻的位置$\boldsymbol{P}^i(t+1)$只能按式(4)更新到确定位置. 为增加多目标优化问题的Pareto解集中解的多样性, 并提高算法的搜索能力, 本文采用随机随机粒子群优化算法[13], 该算法根据参考信息随机更新下一刻位置, 其粒子更新规则如图1所示.

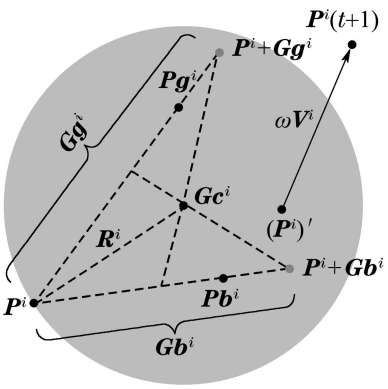

图 1 粒子位置更新示意图

Fig. 1 Updating rule of particle's position

首先根据式(4)计算当前位置$p_j^i(t)$、参考位置$p_j^i+Gb_j^i$和$p_j^i+Gg_j^i$三者的中心位置Gc_j^i:

$$Gc_j^i=\left(p_j^i+(p_j^i+Gb_j^i)+(p_j^i+Gg_j^i)\right)/3, \tag{4}$$

而后计算当前位置$p_j^i(t)$与中心位置Gc_j^i的欧氏距离$R_j^i=d(Gc_j^i,P_j^i)$. 然后在以Gc_j^i为中心以R_j^i为半径的空间内等概率$u(Gc_j^i,R_j^i)$生成一样新的位置$(p_j^i)'$:

$$(p_j^i)'\sim\mathcal{U}(Gc_j^i,R_j^i). \tag{5}$$

若R_j^i为零, 则在可行解空间内等概率随机生成新的位置$(p_j^i)'$, 最后按式(6)更新下一刻粒子的位置$p_j^i(t+1)$. 若粒子位置超出可行解空间, 重置该粒子位置到最近的可行解空间边界, 同时速度置为零.

$$p_j^i(t+1)=\omega\cdot v_j^i(t)+(p_j^i)'. \tag{6}$$

粒子的速度更新公式如下:

$$v_j^i(t+1)=\omega\cdot v_j^i(t)+(p_j^i)'-p_j^i. \tag{7}$$

2.4 多目标分解随机粒子群优化算法(Multiobjective decomposition stochastic particle swarm optimization algorithm)

多目标分解粒子群优化算法(MDSPSO)中, 每个粒子不仅具有速度和位置信息, 同时具有一个固定的权重向量$\boldsymbol{\Lambda}$. 为保证权重向量分布均匀, 所有权重向量的各维均从集合$\{\frac{0}{H},\frac{1}{H},\cdots,\frac{H}{H}\}$中选择. 根据权重和为1要求, 共可以组成$N=\mathrm{C}_{H+m-1}^{m-1}$个不同的权重向量. MDSPSO算法操作步骤如下:

步骤 1　初始化.

1) 随机生成粒子群体$\mathcal{S}=\{\boldsymbol{P}^1,\cdots,\boldsymbol{P}^N\}$, 计算粒子适应值, 并随机初始化每个粒子的速度; 令每个粒子历史最优$\boldsymbol{Pb}^i=\boldsymbol{P}^i(i=1,\cdots,N)$.

2) 粒子群体最优集合$\mathcal{G}=\{\boldsymbol{P}^1,\cdots,\boldsymbol{P}^N\}$, 并随机乱序分配给$N$个粒子, 作为各自群体最优位置.

3) 更新$\boldsymbol{z}^*$与$\boldsymbol{z}^\dagger$. $\forall j, z_j^*=\min\limits_{\boldsymbol{P}\in\mathcal{S}}\{f_j(\boldsymbol{P})\}, z_j^\dagger=\max\limits_{\boldsymbol{P}\in\mathcal{S}}\{f_j(\boldsymbol{P})\}$;

4) 为每个个体指定一个固定的权重向量$\boldsymbol{\Lambda}^i$.

步骤 2　粒子当前位置更新.

对粒子群体$\mathcal{S}$中所有个体, 按式(4)−(7)更新个体的位置与速度, 并计算粒子适应值.

步骤 3　个体历史最优位置更新.

1) 更新$\boldsymbol{z}^*$与$\boldsymbol{z}^\dagger$. $\forall j, z_j^*=\min\{z_j^*,\min\limits_{\boldsymbol{P}\in\mathcal{S}}\{f_j(\boldsymbol{P})\}\}$, $z_j^\dagger=\max\limits_{\boldsymbol{P}\in\mathcal{S}}\{f_j(\boldsymbol{P})\}$.

2) 对$\forall i$, 若$T_{\mathrm{s}}(f(\boldsymbol{P}^i)|(\boldsymbol{\Lambda}^i,\boldsymbol{z}^*,\boldsymbol{z}^\dagger))<T_{\mathrm{s}}(f(\boldsymbol{Pb}^i)|(\boldsymbol{\Lambda}^i,\boldsymbol{z}^*,\boldsymbol{z}^\dagger))$, 则$\boldsymbol{Pb}^i=\boldsymbol{P}^i$, 否则按步骤4操作, 直至完成更新$\mathcal{S}$和$\boldsymbol{z}^*$.

步骤 4　重置停滞粒子位置.

1) 从$\mathcal{G}$任意选择两个不同个体i_1, i_2, 重新生成一个服从高斯分布$N(\frac{\boldsymbol{P}^{i_1}-\boldsymbol{P}^{i_2}}{2}, |\boldsymbol{P}^{i_1}-\boldsymbol{P}^{i_2}|)$的新个体$\boldsymbol{P}^t$, 计算粒子适应值$f(\boldsymbol{P}^t)$.

2) 更新$\boldsymbol{z}^*$. $\forall j,\ z_j^* = \min\{z_j^*, f_j(\boldsymbol{P}^t)\}$.

3) 若$T_{\rm s}(f(\boldsymbol{P}^t)|(\boldsymbol{\Lambda}^i, \boldsymbol{z}^*, \boldsymbol{z}^\dagger)) < T_{\rm s}(f(\boldsymbol{P}^i)|(\boldsymbol{\Lambda}^i, \boldsymbol{z}^*, \boldsymbol{z}^\dagger))$, 则返回$\boldsymbol{Pb}^i = \boldsymbol{P}^i = \boldsymbol{P}^t$, 否则直接返回.

步骤 5 群体最优位置更新.

1) 令$\mathcal{T} = \mathcal{S}\cup\mathcal{G}$, $\mathcal{G} = \varnothing$.

2) 更新$\boldsymbol{z}^\dagger$. $\forall j, z_j^\dagger = \max\limits_{\boldsymbol{P}\in\mathcal{S}\cup\mathcal{G}}\{f_j(\boldsymbol{P})\}$.

3) 对$\forall i, \mathcal{G} = \mathcal{G}\cup\{\boldsymbol{P}^j | \min\limits_{\boldsymbol{P}^j\in\mathcal{T}} T_{\rm s}(f(\boldsymbol{P}^j)|(\boldsymbol{\Lambda}^i, \boldsymbol{z}^*))\}$, $\mathcal{T} = \mathcal{T}\backslash\boldsymbol{P}^j$, 直至完成更新$\mathcal{G}$.

4) 随机打乱集合$\mathcal{G}$中的N个位置乱序随机分配给N个粒子作为它们的群体最优位置.

步骤 6 如果满足终止条件, 则输出$\mathcal{G}$, 否则返回步骤2.

注 2 MDSPSO算法按每个粒子的权重向量采用改进Tchebycheff分解方法, 将求解多目标非支配解问题转化为同时求解多个单目标最优解问题. 粒子位置采用随机生成的策略以增加粒子个体的多样性, 从而提高算法对单目标优化问题的搜索能力. 同时MDSPSO算法将所有粒子的权重向量所对应的单目标优化问题的最优位置作为群体最优集合, 而每个粒子的群体最优位置将从群体最优集合中随机选择, 以保证非支配解的多样性. 采用上述策略, MDSPSO算法一次运行即可获得多个Pareto最优解, 以实现多目标优化问题的求解目标.

3 算法性能测试(Algorithm performance verification)

3.1 评价指标(Performance indexes)

定义 3 定义两集覆盖度$\mathcal{C}(A,B)$表示集合B中能被集合A中元素支配的元素数目占集合B中所有元素数目的比例.

$$\mathcal{C}(A,B) = \frac{|\{\boldsymbol{u}\in B | \exists \boldsymbol{v}\in A : \boldsymbol{v}\prec\boldsymbol{u}\}|}{|B|}.$$

若集合A中的解都能支配集合B中的所有解, 则$\mathcal{C}(A,B)$为1, 相反则为零. 由于两个集合中存在互相无法支配的解, 所以采用两集覆盖度$\mathcal{C}$评价的时候, 需要同时计算$\mathcal{C}(A,B)$和$\mathcal{C}(B,A)$, 且$\mathcal{C}(A,B)+\mathcal{C}(B,A)$不一定为1.

定义 4 定义逆代距$\mathcal{I}$来衡量当前PF集合P^*与真实PF样本集合P的距离, 同时反映了当前PF集合P^*对真实PF样本集合P的覆盖程度.

$$\mathcal{I} = \frac{\sum\limits_{\forall \boldsymbol{v}\in P} d(\boldsymbol{v}, P^*)}{|P|},$$

其中$d(\boldsymbol{v}, P^*) = \min\limits_{\forall \boldsymbol{u}\in P^*} \|\boldsymbol{v}, \boldsymbol{u}\|$.

对测试函数的真实PF进行均匀大样本采样, 构成集合P^*. 逆代距$\mathcal{I}_{\rm GD}$越小, 表明集合P距离真实PF越近, 覆盖度越好.

3.2 测试函数(Test functions)

本文选择测试函数及其特征[14]如表1所示, 其中WFG系列二维测试函数具有复杂的特征, 较其他二维测试函数更加复杂. 在实际问题中, 不同目标的取值范围可能是不同的. 尽管这些测试函数涵盖了常见实际优化问题中可能存在的特征, 但其各目标的取值范围相同或比较接近. 因而保持测试函数第一维不变, 对于二维测试函数, 第二维乘以10作为新的测试函数; 对于三维测试函数, 后两维分别乘以5, 10作为新的测试函数; 对于四维测试函数, 后三维分别乘以3, 6, 10作为新的测试函数.

表 1 测试函数

Table 1 Test functions

函数	m/n	典型特征(具体概念参见文献[14])
WFG2(F1)	2/10	PF非连续, 第二维为多模态
WFG5(F2)	2/10	两维均存在欺骗性模态
WFG7(F3)	2/10	参数相互依赖
WFG9(F4)	2/10	参数相互依赖, 存在欺骗性多模态
DTLZ2(F5)	3/12	三维PF, 存在多对一映射
DTLZ6(F6)	3/12	PF为三维曲线, 存在多对一映射
DTLZ7(F7)	3/22	PF非连续, 最后一维为多模态
DTLZ2(F8)	4/12	四维DTLZ2
DTLZ6(F9)	4/12	四维DTLZ6
DTLZ7(F10)	4/22	四维DTLZ7

3.3 测试结果比较(Comparison of test results)

文献[12]提出了基于差分进化的MOEA/D算法(简记为ME), 并在CEC2009上荣获最佳多目标优化算法; 文献[15]提出了NSGA–II算法(简记为NS), 因算法机制简单, 效果很好, 成为多目标优化问题的经典算法. 为了验证MDSPSO算法(简记为DS)的性能, 本文选用NSGA–II, MOEA/D及dMOPSO算法[5](简记为dM)作为对比算法, 在同一计算机的MATLAB 7.12环境下对表1中测试函数进行20次优化计算, 而后对优化结果进行统计分析. 对二维测试函数, 4种算法的个体数目均为200个($H = 199$), 三维均为300个($H = 23$), 四维均为455个($H = 12$). 基于粒子群算法的DS算法和dM算法设置相同, $r_1 = r_2 = 2.0$, $\omega = 0.7298$. 改进Tchebycheff分解方法中, $\rho = 0.001$. dM, ME和NS的设置均按各自文献进行最优配置. 最终统计结果如表2与表3所示, 其中表2为两集覆盖度对比数据, 表3为逆代距对比数据及时间统计信息, 表中非括号中的数值为均值, 括号中的数值为均方差.

表 2 两集覆盖度

Table 2 $\mathcal{C}$ performance

函数	$\mathcal{C}$(DS, dM)	$\mathcal{C}$(dM, DS)	$\mathcal{C}$(DS, ME)	$\mathcal{C}$(ME, DS)	$\mathcal{C}$(DS, NS)	$\mathcal{C}$(NS, DS)
F1	1.00000(0.00000)	0.00000(0.00000)	0.66325(0.28750)	0.11625(0.13370)	0.06575(0.06898)	0.49275(0.06088)
F2	0.37175(0.32997)	0.00570(0.03242)	0.37900(0.34972)	0.06150(0.02843)	0.40025(0.39868)	0.05500(0.00008)
F3	0.99975(0.00112)	0.00000(0.00000)	0.40200(0.07181)	0.07925(0.01801)	0.32525(0.11845)	0.00125(0.00222)
F4	0.98850(0.00875)	0.00225(0.00472)	0.71625(0.36686)	0.12900(0.20421)	0.31025(0.25293)	0.30350(0.18387)
F5	0.31500(0.03441)	0.01383(0.00595)	0.00117(0.00196)	0.09750(0.01895)	0.34950(0.36266)	0.00083(0.00148)
F6	0.66950(0.03073)	0.00000(0.00000)	0.00400(0.00503)	0.00000(0.00000)	0.95883(0.01028)	0.00000(0.00000)
F7	0.48183(0.02771)	0.00867(0.01051)	0.36017(0.09332)	0.00717(0.01476)	0.73767(0.07205)	0.00150(0.00597)
F8	0.40242(0.01657)	0.00000(0.00000)	0.03154(0.00745)	0.00169(0.02142)	0.48198(0.06021)	0.00000(0.00000)
F9	0.62539(0.02820)	0.00000(0.00000)	0.00857(0.00996)	0.00099(0.00442)	0.81231(0.03115)	0.00000(0.00000)
F10	0.20769(0.02110)	0.01923(0.00921)	0.03692(0.06010)	0.00560(0.01443)	0.84703(0.04525)	0.00539(0.00915)

表 3 逆代距与时间代价

Table 3 $\mathcal{I}_{\mathrm{GD}}$ performance and time cost

函数	DS		dM		ME		NS	
	$\mathcal{I}_{\mathrm{DS}}$	用时/s	$\mathcal{I}_{\mathrm{dM}}$	用时/s	$\mathcal{I}_{\mathrm{ME}}$	用时/s	$\mathcal{I}_{\mathrm{NS}}$	用时/s
F1	0.12271(0.00688)	44.72	0.41561(0.04036)	75.71	1.55382(1.12979)	42.88	0.29362(0.35278)	46.14
F2	0.11729(0.01008)	45.55	0.14474(0.03316)	75.76	0.75688(0.30224)	42.51	0.13975(0.03237)	46.67
F3	0.06807(0.00012)	48.19	0.21328(0.00857)	75.76	0.39505(0.09018)	43.87	0.07634(0.00801)	46.59
F4	0.08080(0.00538)	46.88	0.11298(0.01107)	77.16	0.33791(0.29297)	44.05	0.17687(0.08629)	48.08
F5	0.17378(0.00114)	71.53	0.26672(0.01228)	124.9	0.19086(0.00379)	63.76	0.27099(0.01312)	103.7
F6	0.03707(0.00003)	71.93	0.07308(0.00021)	125.1	0.03483(0.00098)	63.69	4.78374(0.83856)	116.7
F7	0.31108(0.00522)	73.91	0.32090(0.00664)	128.3	1.54117(1.89982)	63.94	1.25173(0.68624)	113.6
F8	0.51847(0.00418)	126.0	0.59042(0.05910)	231.5	0.60595(0.06717)	99.45	0.61878(0.03307)	247.5
F9	0.10752(0.00149)	149.8	0.17605(0.02140)	241.1	0.26720(0.09470)	109.2	8.68658(1.02340)	290.1
F10	0.77037(0.02324)	154.7	1.03551(0.01465)	256.3	2.48086(3.07523)	109.2	2.14478(0.68424)	314.1

为形象化地描述上述数据, 图2−5直观地对比显示了4种算法某次得到的二维测试函数F1−F4的PF.

从图2中中间的放大效果图可见, dM解(dM算法得到的解, 下同)距离真实PF较远, 完全被DS解支配, 因而$\mathcal{C}(\mathrm{DS},\mathrm{dM})=1$; 尽管ME解有少数能支配DS解, DS解依然能支配大部分ME解, 同时ME解并没有接近真实PF上两段, 因而$\mathcal{C}(\mathrm{DS},\mathrm{ME})>\mathcal{C}(\mathrm{ME},\mathrm{DS})$; NS解距离真实PF最近, 能被支配的最少, 同时能支配的其他算法解最多, 但DS很好地接近了全部真实PF, 而NS解只很好地逼近了真实PF上3段, 故NS解能支配的也是DS解上3段中的解, 则$0.67>\mathcal{C}(\mathrm{NS},\mathrm{DS})>\mathcal{C}(\mathrm{DS},\mathrm{NS})$. 虽然$\mathcal{C}(\mathrm{NS},\mathrm{DS})>\mathcal{C}(\mathrm{DS},\mathrm{NS})$, 但NS解距离下3段真实Pareto最优解较远, 从而使得$\mathcal{I}_{\mathrm{NS}}<\mathcal{I}_{\mathrm{DS}}$. 可见两集覆盖度$\mathcal{C}(A,B)$越大, 表明前者对后者的支配能力越强, 越接近真实PF; 若要使逆代距小, 解不仅要接近真实PF, 同时要尽可能覆盖真实PF.

图3−5说明了同样的道理. 由于智能算法具有随机性, 每次求解结果未必接近, 均方差越大表明算法的不确定性越大, 比如图5所示, $\mathcal{I}_{\mathrm{ME}}=0.96285$, 大于表3中所列的平均水平.

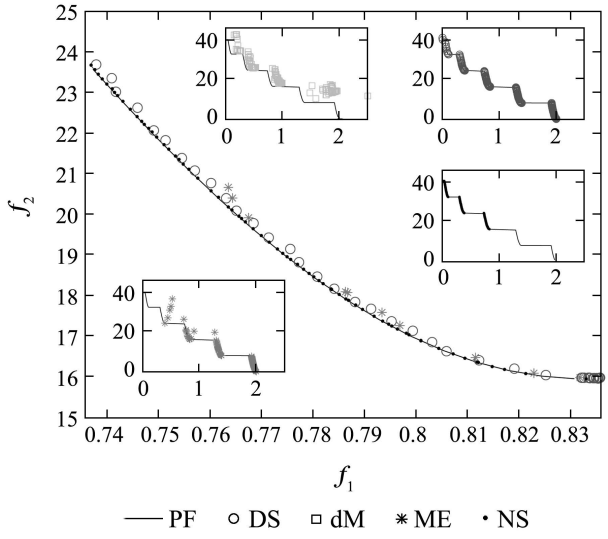

图 2 F1函数PF

Fig. 2 PF of F1 test function

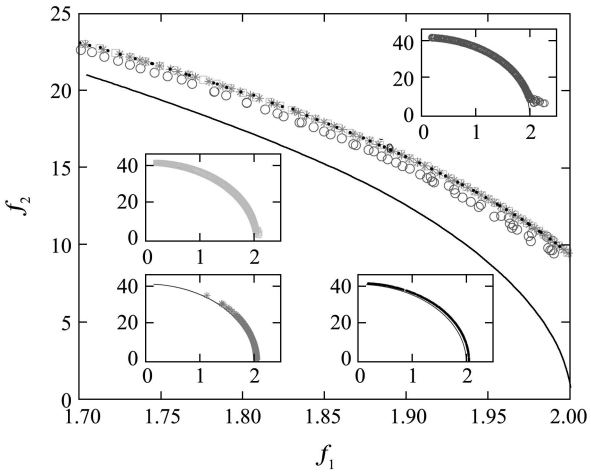

图 3 F2函数PF

Fig. 3 PF of F2 test function

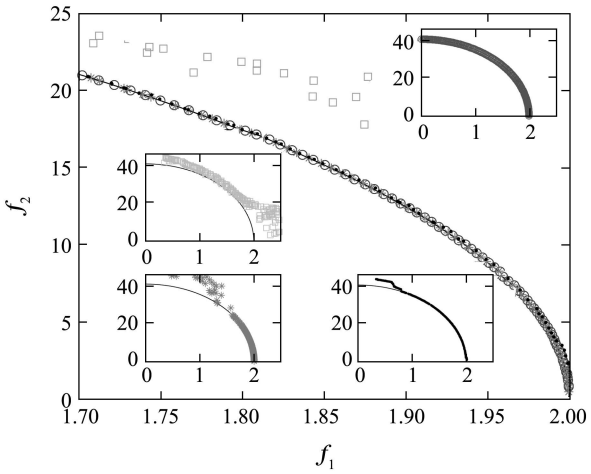

图 4 F3函数PF

Fig. 4 PF of F3 test function

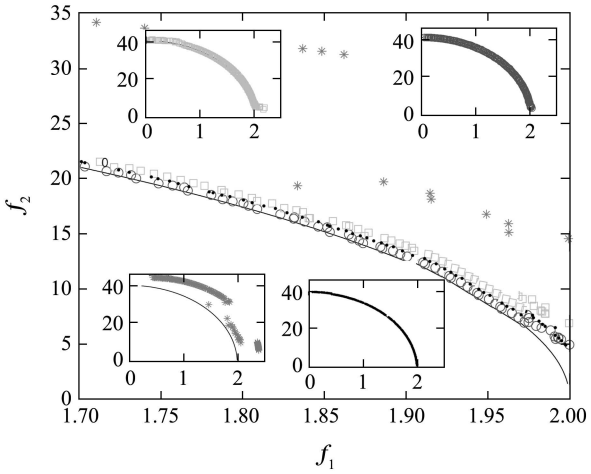

图 5 F4函数PF

Fig. 5 PF of F4 test function

综合可以看出, DS算法具有更优异的收敛性和多样性, 算法也更加稳定. 与dM算法相比, 随机粒子群算法具有更强的随机性, 有助于提高多目标算法的搜索能力; 而改进的Tchebycheff分解方法, 引入求和项, 使得引理1中$T_{\rm s}(\boldsymbol{y}^1)$一定小于$T_{\rm s}(\boldsymbol{y}^2)$, 从而保证了单目标问题最优解一定是多目标问题的Pareto最优解. 整体而言, 基于分解的多目标优化算法性能要高于基于非支配解排序方法.

ME算法由于机制简单, 因此计算时间上始终略胜一筹. DS算法中粒子个体搜索能力更强, 较dM较少地对粒子进行重置, 因此用时也较dM少. 基于分解的多目标优化算法无需判断支配关系, 当目标维数增加时, 时间的增加量也较基于非支配解排序的NS算法增加量少. 但基于分解方法的优化过程中, 若某个体解没有收敛到单目标问题最优时, 此解可能会被其他更优的解支配, 如图2−5所示.

4 直线电机优化设计(Optimal design of linear motor)

4.1 电机描述(Motor description)

如图6所示的直线电机(linear motor, LM)由圆筒型的外部电枢动子和内部永磁定子构成. 电枢铁心采用硅钢片叠压而成, 共有24槽, 每个槽安放一个线圈元件, 各相元件连接次序如表4所示. A, B, C三相采用Y接法, 构成电机的三相绕组. 三相绕组和电枢铁心一起, 构成电机的动子. 沿径向充磁的永磁体NS极交替分布, 等间隔贴放在导磁性的定子杆上, 并在永磁体之间使用不导磁铝环隔开, 整体组成电机的长定子. 沿轴线径向半剖, 如图7所示, 表5列出了图中标注参数的名称与设置.

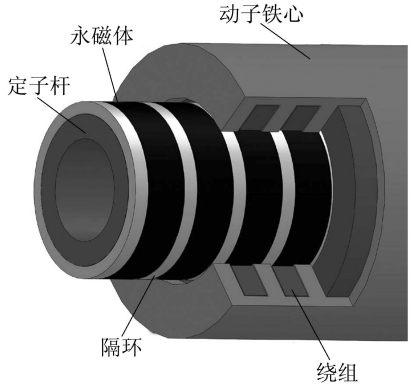

图 6 3D模型部分剖视图

Fig. 6 3D view with cutting

和旋转同步电机工作原理类似, 当动子电枢绕组中通以三相交流电, 在气隙中产生行波磁场, 与永磁体产生的气隙磁场相互作用, 产生轴向力, 使动子沿轴向机械运动, 且动子的运动速度和行波

磁场移动速度相同, 从而产生稳定的同步直线运动. 而行波磁场速度取决于三相电流相位变化速度, 所以电流相位变化与动子机械运动引起的电角度变化相同.

表 4 元件连接次序

Table 4 Winding connection order

相位	元件连接次序(上标 “′” 表示反绕)
A相	$1 \to 2' \to 7' \to 8 \to 13 \to 14' \to 19' \to 20$
B相	$3' \to 4 \to 9 \to 10' \to 15' \to 16 \to 21 \to 22'$
C相	$5 \to 6' \to 11' \to 12 \to 17 \to 18' \to 23' \to 24$

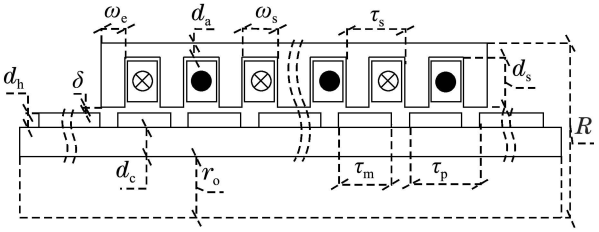

图 7 半剖面断裂视图

Fig. 7 The halved cross section

表 5 直线电机模型参数

Table 5 Model parameters of the LM

参数	值/mm	参数	值/mm
电机外半径R	60	气隙δ	[1, 3]
永磁体宽度τ_m	[10, 24.3]	极距τ_p	24.3
永磁体厚度d_h	[2, 6]	槽距τ_s	20.25
定子内半径r_0	21	槽宽ω_s	[10, 16]
定子杆厚度d_c	[8, 12]	槽深d_s	[13, 23]
端部齿宽度ω_e	[4, 12]	外壳厚d_a	5

有限元分析方法可用于求解复杂区域的电磁场分布问题, 能精确计算电机的电磁分布、推力等性能. 在确定起始位置电流相位后, 通过瞬时有限元分析计算此位置电机的推力大小, 而后沿推力方向动子移动指定步长Δ, 即动子电角度变化$\Delta \times \pi/\tau_p$, 调整电流相位增加$\Delta \times \pi/\tau_p$, 再一次分析新位置的电机推力大小. 在一对极距范围内不同位置连续计算, 则得到电机在一个周期内推力值. 不同位置推力的平均值, 作为电机的平均推力, 记为F_0; 不同位置推力的均方差, 作为电机的推力波动, 记为F_r.

气隙δ大小决定气隙中的磁压降, 对电机性能有决定性影响; 永磁体宽度τ_m与厚度d_h决定的永磁体体积及气隙磁场分布; 槽宽ω_s和槽深d_s决定了线圈体积及电机出力大; 端部齿产生电机端部效应, 其宽度ω_e设置对电机推力波动有重要影响; 同时永磁体和线圈的内外径大小受定子杆厚度d_c影响. 因而选择上述7个参数对作为待决策参数变量, 优化设计电机的性能. 由图2可知, $d_s = R - (r_0 + d_c + d_h + \delta + d_a)$, 因而可将电机的平均推力$F_0$、推力波动$F_r$描述如下:

$$\begin{cases} F_0 = f_0(\delta, \tau_m, d_h, \omega_s, \omega_e, d_c), \\ F_r = f_r(\delta, \tau_m, d_h, \omega_s, \omega_e, d_c). \end{cases} \tag{8}$$

4.2 模型建立(Model building)

对于智能优化方法, 需要循环计算不同参数变量下电机的性能. 若每次计算均采用有限元分析计算, 其计算量过大. 平均一次模型分析大概需要430 s, 优化迭代过程中, 若需要计算模型10000万次, 则需要用时近50天.

鉴于神经网络对非线性函数的拟合能力, 可通过神经网络拟合式(8), 用来计算电机的性能. 神经网络结构如图8所示: 神经网络的决策变量为$(\delta, \tau_m, d_h, \omega_s, \omega_e, d_c)$, 输出为$(F_0, F_r)$, 隐层包含为20个神经元. 输入层采用Sigmoid, 输出采用线性函数, 使用改进的Levenberg-Marquardt训练算法, 使得网络输出的均方误差最小.

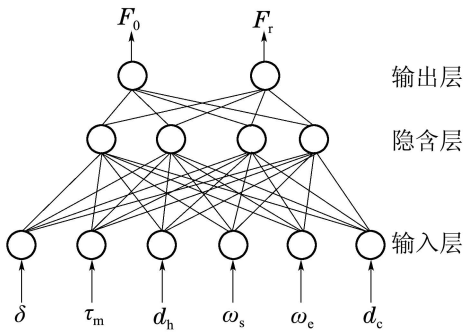

图 8 神经网络结构

Fig. 8 The architecture of artificial neural network

在决策变量的可行域内, 采样500个样本, 使用有限元计算电机的性能(F_0, F_r), 而后将数据用于训练神经网络. 为验证神经网络的拟合精度, 重新采样50个样本, 输入到神经网络中, 将输出与有限元计算结果进行对比, F_0误差绝对值的均值为0.8157 N, 最大为2.6244 N, F_r误差绝对值的均值为1.1089 N, 最大为3.3458 N. 可见, 神经网络达到较好的拟合精度. 而500样本计算总用时约2.5天, 有效节省了直接采用有限元方法进行计算的时间.

4.3 多目标优化设计(Multi-objective optimal design)

直线电机优化设计的目标是提高电机的推力F_0(N), 降低推力的波动F_r(N), 减少绕组体积

$V_{\rm c}(\rm cm^3)$以降低绕组损耗, 同时要控制磁钢使用量$(\rm cm^3)$来平衡电机的成本. 其中:

$$V_{\rm p}=\pi\times[(r_0+d_{\rm c}+d_{\rm h})^2-(r_0+d_{\rm c})^2]\times(20\tau_{\rm m})/1000,$$

$$V_{\rm c}=\pi\times[(R-d_{\rm a})^2-(R-d_{\rm a}-d_{\rm s})^2]\times(24\omega_{\rm s})/1000.$$

设置4个优化目标为

$$f_1=\frac{2000}{F_0},\ f_2=F_{\rm r},\ f_3=\frac{V_{\rm c}}{V_{\rm cmin}},\ f_4=\frac{V_{\rm p}}{V_{\rm pmin}}.$$

注 3 f_3与f_4可以通过结构参数直接计算获得, 而计算f_1与f_2需要求解电机的推力F_0与波动$F_{\rm r}$. 采用神经网络模型来计算F_0与$F_{\rm r}$解决了有限元方法耗时过大的问题, 从而大大节省了计算时间, 降低了设计周期.

采用和测试函数中相同的设置, 分别采用DS, dM, ME, NS 4种算法优化求解上述4个目标20次. DS算法平均用时142.3 s, 比ME算法的107.2 s高, 但低于dM的239.6 s和NS的271.0 s. 由于实际工程问题中真实PF未知, 因此无法通过逆代距来衡量算法性能, 这里仅采用两集覆盖度来比较4种算法的性能, 如表6所示.

表 6 直线电机优化设计结果的两集覆盖度

Table 6 $\mathcal{C}$ performance of LM optimization designs

$\mathcal{C}$(DS, dM)	$\mathcal{C}$(dM, DS)	$\mathcal{C}$(DS, ME)
0.307554(0.02612)	0.00810(0.01143)	0.02588(0.03967)
$\mathcal{C}$(ME, DS)	$\mathcal{C}$(DS, NS)	$\mathcal{C}$(NS, DS)
0.00324(0.00749)	0.40665(0.22701)	0.00000(0.00000)

表7列举了DS算法得到的4个Pareto最优解作为例子, 采用有限元方法分析表7中的4个设计结果, 推力曲线如图9所示.

表 7 直线电机优化设计结果

Table 7 LM optimization designs

	符号	例1	例2	例3	例4
参数	δ	2.92	1.93	1.74	1.70
	$\tau_{\rm m}$	23.16	22.80	22.93	21.50
	$d_{\rm h}$	5.66	5.12	3.61	5.00
	$\omega_{\rm s}$	12.88	13.22	14.03	13.40
	$\omega_{\rm e}$	11.99	11.65	10.52	10.74
	$d_{\rm c}$	10.26	10.73	10.05	10.50
性能	F_0	2351.75	3011.46	3018.02	3072.21
	$F_{\rm r}$	5.55	50.24	59.41	59.52
	$V_{\rm c}$	1395.86	1516.12	1799.87	1581.94
	$V_{\rm p}$	562.11	502.68	341.52	459.30

例1中推力波动$F_{\rm r}$小, 但推力F_0不足. 另外3个例子推力F_0与波动$F_{\rm r}$符合要求, 但例4中绕组$V_{\rm c}$体积和磁钢体积$V_{\rm p}$的组合更合理, 因而按例4中的结构作为电机的优化设计结果, 进行加工生产, 实物如图10所示. 电机的实际推力测试曲线如图9中的实线所示.

从图9中可知, 例4的推力结果与实测推力的最大误差为2.09%, 为有限元数值分析误差, 工程上是允许的. 优化后的电机, 推力大, 波动小, 效率高, 造价合理, 符合电机的设计目标.

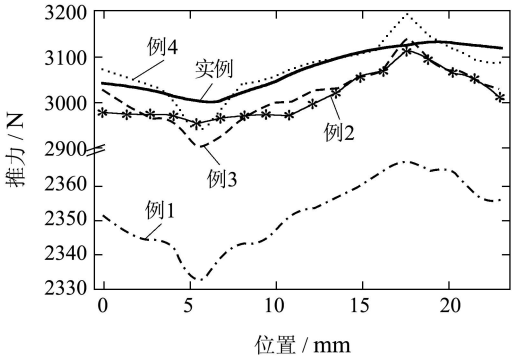

图 9 电机优化设计

Fig. 9 Optimal motor designs

图 10 实际电机照片

Fig. 10 Picture of manufactured motor

5 结论(Conclusions)

本文提出一种多目标分解粒子群优化算法(MDSPSO), 其中改进Tchebycheff分解方法将多目标优化问题转化为单目标优化问题, 并保证了单目标问题的最优解一定是多目标问题的Pareto最优解; 而随机粒子群优化算法具有更强的随机性, 增强了算法的搜索能力. 通过多个测试函数进行仿真对比, 统计表明MDSPSO的收敛性和多样性上优于所对比的其他3种算法. 针对直线电机磁路复杂、有限元计算费时等特点, 建立了直线电机的神经网络模型. 选择直线电机推力、波动、效率、造价作为优化目标, 应用多目标分解随机粒子群优化算法优化设计了直线电机结构参数. 实际结果表明, MDSPSO优化后的直线电机, 推力大, 波动小, 效率高, 造价低.

参考文献(References):

[1] 雷德明, 严新平. 多目标智能优化算法及其应用 [M]. 北京: 科学出版社, 2009.
(LEI Deming, YAN Xinping. *Multi-objective Intelligent Optimiaztion Algorithm and Its Applications* [M]. Beijing: Science Press, 2009.)

[2] ZHOU A M, QU B Y, LI H, et al. Multi-objective evolutionary algorithms: a survey of the state of the art [J]. *Swarm and Evolutionary Computation*, 2011, 1(2011): 32 – 49.

[3] REYES-SIERRA M, COELLO COELLO C A. Multi-objective particle swarm optimizers: a survey of the state-of-the-art [J]. *International Journal of Computational Intelligence Research*, 2006, 2(3): 287 – 308.

[4] 冯琳, 毛志忠, 袁平. 改进多目标粒子群优化算法及其在电弧炉供电优化中的应用 [J]. 控制理论与应用, 2011, 28(10): 1455 – 1460.
(FENG Lin, MAO Zhizhong, YUAN Ping. Improved multi-objective particle-swarm algorithm and its application to electric arc furnace in steelmaking process [J]. *Control Theory & Applications*, 2011, 28(10): 1455 – 1460.)

[5] MARTINEZ S Z, COELLO COELLO C A. A multi-objective particle swarm optimizer based on decomposition [C] //*Proceedings of the 13th Annual Conference on Genetic and Evolutionary Computation*. New York: ACM, 2011, 7: 69 – 76.

[6] 江友华, 廖代发, 唐忠. 混合有源滤波器多目标优化设计 [J]. 控制理论与应用, 2010, 27(7): 916 – 922.
(JIANG Youhua, LIAO Daifa, TANG Zhong. Multi-objective optimal design of hybrid active power filter [J]. *Control Theory & Applications*, 2010, 27(7): 916 – 922.)

[7] 张静, 王万良, 徐新黎, 等. 混合粒子群优化算法求解多目标柔性作业车间调度问题 [J]. 控制理论与应用, 2012, 29(6): 715 – 722.
(ZHANG Jing, WANG Wanliang, XU Xinli, et al. Hybrid particle-swarm optimization for multi-objective flexible job-shop scheduling problem [J]. *Control Theory & Applications*, 2012, 29(6): 715 – 722.)

[8] 叶云岳, 卢琴芬, 范承志, 等. 直线电机技术手册 [M]. 北京: 机械工业出版社, 2003.
(YE Yunyue, LU Qinfen, FAN Chengzhi, et al. *Technical Manual of Linear Motor* [M]. Beijing: China Machine Press, 2003.)

[9] GIERAS J F, PIECH Z J, TOMCZUK B. *Linear Synchronous Motors: Transportation and Automation Systems* [M]. 2nd edition. New York: CRC Press, 2011.

[10] ASHABANI M, MOHAMED Y, MILIMONFARED J. Optimum design of tubular permanent-magnet motors for thrust characteristics improvement by combined taguchi-neural network approach [J]. *IEEE Transactions on Magnetics*, 2010, 46(12): 4092 – 4100.

[11] LI L, MA M, CHEN Q. Analysis and design of moving-magnet-type linear synchronous motor for electromagnetic launch system [J]. *IEEE Transactions on Plasma Science*, 2011, 39(1): 121 – 126.

[12] LI H, ZHANG Q F. Multi-objective optimization problem with complicated Pareto sets, MOEA/D and NSGA-II [J]. *IEEE Transaction on Evolutionary Computation*, 2009, 13(2): 284 – 302.

[13] CLERC M. Standard particle swarm optimisation from 2006 to 2011 [BD/OL]. http://clerc.maurice.free.fr/pso/SPSO_descriptions.pdf.

[14] HUBAND S, HINGSTON P, BARONE L, et al. A review of multiobjective test problems and a scalable test problem toolkit [J]. *IEEE Transactions on Evolutionary Computation*. 2006. 10(5): 477 – 506.

[15] DEB K, PRATAP A, AGARWAL S, et al. A fast and elitist multiobjective genetic algorithm: NSGA-II [J]. *IEEE Transactions on Evolutionary Computation*, 2002, 6(2): 182 – 197.

作者简介:

王光辉　(1987−), 男, 博士研究生, 主要研究方向为伺服系统、直线电机、人工智能与优化计算;

陈　杰　(1965−), 男, 博士, 教授, 博士生导师, 主要研究方向为复杂系统优化与决策、智能控制、约束系统非线性控制;

蔡　涛　(1971−), 男, 博士, 副研究员, 主要研究方向为伺服系统、智能控制、非线性控制, E-mail: caitao@bit.edu.cn;

李　鹏　(1982−), 男, 博士研究生, 目前研究方向非线性系统控制、自适应控制、神经网络、新能源系统.