

**北京科技大学**

**课程设计(自动控制原理)**

**报告**

设计题目:**基于****遺傳算法PID永磁同步电动机控制系统设计** \_

班 级： **自232** \_

姓 名： **黄旨鹏** \_

学 号： **U202342197** \_

指导教师： **任莹莹** \_

日 期： 2025 年 7 月 8 日

**基于遺傳算法PID永磁同步电动机控制系统设计**

## 摘要

本研究针对永磁同步电机（PMSM）高阶系统控制难题，提出基于遗传算法（GA）优化的PID控制策略。传统PID整定方法（如Ziegler-Nichols）在PMSM六阶系统中难以实现稳定控制（调节时间达248秒）。通过设计遗传算法，将PID参数（Kp, Ki, Kd）作为染色体基因进行选择、交叉与变异操作，以积分平方误差（ISE）为适应度函数迭代优化。仿真结果表明：相较于传统PID，遗传算法PID将调节时间缩短至3.42秒（提升7251.46%），超调量降低至47.3%，且有效抑制震荡。尽管存在计算复杂度高与早熟收敛风险，该方法为高阶永磁电机系统提供了高精度控制新路径，对高铁牵引、精密制造等领域具工程应用价值。

另外可见不同的PID参数范围对于遗传算法学习性能有决定性影。不合理的P ID范围参数不但不能提升系统性能，反而会加重运行负担。如合理的PID参数范围，即可以快速到达最大值，亦有可能会使得错失全局最大值，落入局部最大值。

ITAE相比于ISE更可以消除其稳定后的等幅震荡，但是**ITAE**会令时间初开始的误差积分近似为零，有可能会增加超调量。

关键词：遗传算法，高阶永磁电机系统，PID。

**Design of a Permanent Magnet Synchronous Motor Control System Based on Genetic Algorithm-Optimized PID**

## Abstract

This study addresses the control challenges of high-order permanent magnet synchronous motor (PMSM) systems by proposing a genetic algorithm (GA)-optimized PID control strategy. Conventional PID tuning methods (e.g., Ziegler-Nichols) fail to achieve stable control in sixth-order PMSM systems (settling time: 248 s). the designed genetic algorithm optimizes PID parameters (Kp, Ki, Kd) encoded as chromosomal genes through selection, crossover, and mutation operations, with the integral square error (ISE) serving as the fitness function for iterative refinement.

Simulation results demonstrate that compared to traditional PID, the GA-PID reduces settling time to 3.42 s (improvement of 7,251.46%) and lowers overshoot to 47.3%, while effectively suppressing oscillations. Despite risks of high computational complexity and premature convergence, this method provides a novel high-precision control pathway for high-order PMSM systems, offering engineering value for high-speed rail traction and precision manufacturing.

The selection of different PID parameter ranges has a decisive impact on The learning performance of genetic algorithms. Unreasonable PID parameter ranges not only fail to enhance system performance but also increase the computational burden. Even a seemingly reasonable range might facilitate rapid convergence to a local optimum, potentially causing the algorithm to miss the global optimum.

Compared to ISE (Integral of Squared Error), ITAE (Integral of Time-weighted Absolute Error) more effectively suppresses sustained oscillations after the system stabilizes. However, ITAE tends to minimize error contributions during the initial period, which may lead to increased overshoot.

**Keywords：**Genetic algorithm, high-order PMSM system, PID control

目录

[摘要 I](#_Toc202949771)

[Abstract II](#_Toc202949772)

[1 引言 1](#_Toc202949773)

[2 文献综述 1](#_Toc202949774)

[3 研究方案设计与说明 1](#_Toc202949775)

[3.1 遗传算法PID的基本概念 2](#_Toc202949776)

[3.2 遗传算法PID的基本](#_Toc202949777)**[原理](#_Toc202949777)** [2](#_Toc202949777)

[3.3遗传算法优化PID控制器的缺点和注意事项 4](#_Toc202949779)

[4基于 MATLAB 的仿真及分析 5](#_Toc202949780)

[4.1 常用函数 5](#_Toc202949781)

[4.2PID永磁同步电动机控制系统设计 13](#_Toc202949782)

[4.3编制程序 15](#_Toc202949783)

[4.4 仿真研究及分析 16](#_Toc202949785)

[5 结束语 19](#_Toc202949788)

[参考文献 19](#_Toc202949789)

[附錄 20](#_Toc202949790)

## 1 引言

永磁同步电动机作为现代高端装备的核心驱动技术，其高功率密度、高效率及低噪声特性在新能源汽车、高速列车和工业伺服领域发挥着关键作用。尤其在中国高铁的跨越式发展中，永磁同步牵引电机已成为实现列车高速化的重要支撑[2][5]，例如复兴号列车时速已达350公里，同时显著提升能效并减轻系统重量[2]。

值得关注的是，永磁直线同步电动机（PMLSM）作为旋转电机的延伸形态，集成了高推力密度、高加速度、高速及高精度等优势[5]，已成功应用于高铁磁悬浮系统、高端光刻设备、3C制造装备及智能物流系统。随着电力电子技术的进步，永磁电机系统成本持续下降，应用场景已从工业领域延伸至民生设备，如童怀等[4]研究的洗衣机驱动系统。

随着现代控制理论和计算机技术的持续快速发展，控制工程师面临着越来越严峻的挑战：选择适合的控制器结构然后优化其参数以满足特定实际应用的性能要求。实际上，控制系统的建模和设计都是在具有噪声情况下的多模空间中的多维优化任务．由于在实际应用当中，传统的解析和数值方法需要的目标函数的优良特性已经不再存在，因而传统优化方法都不能轻易或准确地进行这一多维多模优化任务。

从大自然的杰作——生物进化论中，我们可以得到灵感与启迪。地球上的生物在漫长的进化过程中，逐渐从最简单的低级生物一直发展到万物之灵的人类。这是一个绝妙的优化过程。达尔文的进化论说明，生物进化经历了突变、自然选择和隔离等过程的渐次分化，最终得以形成新种。这一奇迹并非来自神创，而是“物竞天择，适者生存，不适者淘汰”这一“自然选择”过程的必然结果。

那么，能否将“自然选择”这一法则，用于科学研究和工程实际中的种种搜索和优化问题中呢？研究遗传算法PID在的控制效果十分重要一种为高复杂度高精度深化整定PID在永磁电机控制中的应用研究，不仅关乎国家高端装备自主化战略，更为民生领域技术升级提供核心驱动力。

## 2 文献综述

当前永磁同步电机控制研究主要围绕矢量控制展开。基础控制策略如直轴电流为零控制虽结构简洁，却因忽视凸极电机磁阻转矩而导致能效损失[2]；最大转矩电流比控制虽提升转矩输出效率，但需实时求解非线性方程，存在显著计算负担。在高速弱磁控制领域，电压闭环方案因反馈延迟导致动态响应不足，而移相弱磁技术可拓展转速范围，却诱发电流畸变与转矩脉动问题。

智能优化PID方法虽能改善动态性能，其计算复杂度难以满足高铁牵引系统的毫秒级实时响应需求。相较之下，Ziegler-Nichols整定PID展现出显著优势[1]：在此背景下,通过频域响应模型直接获取控制器参数，无需依赖精确数学模型即可实现快速整定[1][6]，大幅降低算法复杂度。相较于遗传算法、粒子群优化等智能优化方法，其计算效率更高且硬件成本更低[1]，更契合工业规模化应用需求。Ramanaiah团队通过建立永磁电机传递函数模型，结合该方法实现参数优化，实验数据表明系统超调量降至3.1%、调节时间缩短至0.18秒，Bode图验证的21.4度相位裕度更凸显其稳定性优势。

同传统的基于微积分的方法和穷举法等算法相比，遗传算法（ GA ） 是一种成熟的具有极高鲁棒性和广泛适用性的全局优化方法．由于 GA 具有不受问题性质（ 如连续性，可微性） 的限制，能够处理传统优化算法难以解决的复杂问题等优点，显示了它在解决控制系统优化方面的巨大潜力，因而引起了控制领域的极大关注．近年来在自动控制领域，遗传算法在 PID 控制、线性和非线性、最优、鲁棒、自适应、滑模、模糊逻辑、神经网络、参数估计和系统辨识、模型线性化和控制器降阶、机器人手臂控制和轨迹规划等方面得到了广泛的应用[9]两者各执一词 ,難分勝負。

## 3 研究方案设计与说明

### 3.1 遗传算法PID的基本概念

遗传算法是基于自然选择和基因遗传学原理的随机搜索算法。它将适者生存这一基本的进化理论引入串结构，使用复制、交叉和变异等基因操作，在串之间进行有组织但又随机的信息交换．伴随着算法的运行，优良的品质被逐渐不断地继承下来，坏的特性被逐渐淘汰。新一代个体中既包含着上一代个体的大量信息，又不断地在总体特性上胜过旧一代，从而使整个群体向前进化发展．对于 GA 所优化的问题，也就是不断地接近于最优解．遗传算法（Genetic Algorithm, GA）的提出，正是源于这一疑问。早在20世纪60年代，美国密歇根大学的John Holland教授就提出了遗传算法的基本思想。其数学框架是在70年代末期形成的，并在David Goldberg于1989年出版的专著《Genetic Algorithms in Search, Optimization, and Machine Learning》中予以系统介绍。随后，遗传算法开始吸引大量的研究者和探索者，并在许多工程领域得到了应用，例如：管道线路优化、机器学习、模型识别、神经网络结构参数优化及权重学习、精调模糊逻辑控制器、飞船控制系统优化等。

### 3.2 遗传算法PID的基本**原理**

遗传算法（Genetic Algorithm, GA） 是一种建立在自然选择和群体遗传学原理基础上的随机、迭代、进化式搜索方法，具有广泛的适用性。

所有自然物种都通过适应环境得以生存，这种自然适应性是遗传算法的主旋律。遗传算法搜索结合了达尔文的“适者生存”原理和随机信息交换机制。前者淘汰解中的不适应因素，后者则利用了原有解中蕴含的知识，从而显著加速搜索过程。下面举例说明遗传算法的基本方法：

对于一个给定的优化问题，设目标函数为：



不失一般性，假设我们要求最大值。



其中 为自变量，是自变量的定义域，可以是数值或符号，为实数，是解的优劣程度或适应度的一种度量，是从解空间 的映射。那么，使用遗传算法求解该问题的步骤如下：(图 1)



图 1遗传算法的框架

（1）编码

用一定比特数的 0，1 二进制码对自变量进行编码形成基因码链，每一码链代表 一个个体，表示优化问题的一个解.如 x有 16 种可能取值 ，则可以用 4 bit 的二进制码  来表示.将 x，y，z 的基因码组合在一起则形成码链，

（2）产生初始群体

随机产生  个个体，组成初始群体 。该群体代表了优化问题的一些可能解的集合。初始群体的解质量通常较差。遗传算法的任务就是从这些初始群体出发，模拟进化过程，通过择优汰劣，最终得到性能优异的群体和个体，满足优化要求。

**(**3**)** 评价

根据编码规则，将群体 中每一个体的基因串解码，得到对应的自变量取值 。将这些取值代入目标函数，计算出其函数值函数值 越大，表示该个体具有较高的适应度，更适应所定义的“生存环境”。适应度 为群体进化过程中的选择操作提供了依据。

(4) 选择（复制）

以一定的概率（通常与个体的适应度 成正比）从当前群体 中选取 M 对个体作为双亲，用于繁殖后代。

被选中的双亲将其基因复制（复制自身）到下一代群体  中（或用于生成新个体）。适应度高的优良个体有更大的机会被选中并繁殖后代，从而使优良特性得以遗传。选择是遗传算法的关键操作，体现了自然界中“适者生存”的思想。

(5) 交叉

对于被选中用于繁殖的每一对双亲个体，随机选择一个相同的交叉点。在该交叉点处，交换双亲基因串的部分片段，产生两个新的后代个体。它们组合了父辈个体 A 和 B 的特征。

交叉操作体现了自然界中信息交换的思想。

(6) 变异

以较小的概率（变异概率 ，通常很小，例如小于0.01），从群体 中随机选取若干个个体。对于选中的个体，随机选择其基因串中的某一位（基因位），进行“取反”运算（即 0 变 1 或 1 变 0）。如同自然界一样，某一位基因发生变异的概率很小。变异操作模拟了生物进化过程中偶然的基因突变现象。对产生的新一代群体进行重新评价选择、交叉、变异，如此循环往复，使群体中最优个体的适应度和平均适应度不断提高，直至最优个体的适应度达到某一限值或最优个体的适应度和 群体的平均适应度值不再提高，则选代过程收敛，算法结束

### 3.3遗传算法优化PID控制器的缺点和注意事项

使用遗传算法PID控制器有几点值得注意：

1.计算复杂度高：每个个体都需要进行仿真，而仿真通常比较耗时，特别是对于复杂的系统。种群规模大、迭代次数多时，计算时间会很长。

2. 早熟收敛：遗传算法可能过早收敛到局部最优解，而无法找到全局最优解。这主要是由于种群多样性减少过快导致。

3. 参数选择困难：遗传算法本身有多个参数需要设定（如种群大小、交叉概率、变异概率等），这些参数的选择对算法性能影响很大，但没有统一的标准，需要根据经验调整。

4. 随机性：由于遗传算法具有随机性，每次运行的结果可能不一致，因此有时需要多次运行以获取可靠的结果。

5. 编码问题：采用二进制编码时，编码长度和范围的选择会影响精度和搜索空间。过长的编码会增加计算负担，过短则精度不够。

6. 性能指标选择：适应度函数的设计（即性能指标）对优化结果影响很大。不同的性能指标可能导致不同的优化结果，需要根据控制要求仔细选择。

## 4基于 MATLAB 的仿真及分析

### 4.1 常用函数

下面介绍了本论文中要使用遗传算法求解该问题的函数和過程。为帮助理解，本人用Visio 软件制作原理图。程式有简短注释。

1. 编码,产生初始群体

初始设定PID参数都为0到100（无法找到稳定解），由几次仿真后可调整参数范围，直接调整以下参数，可以很快地找到最优解（有可能是局部最大值）。完成以下代码，就完成了二进制矩阵的生成和初始化 (图 2)。而总群大小和染色体总长度越大，其PID参数的精度越大（成指数上升）。相对的，其计算量也会成倍增加。



1. 染色体分割

由于PID有三个参数（Kp, Ki, Kd），所以我们需要同时优化三个参数。因此，每个个体（染色体）应该包含三个参数的编码。我们可以将染色体分成三段，分别表示Kp, Ki, Kd。以 图 2 为例，其三个参数分别对应三个不同颜色的方块，其值取0或1。





图 2遗传算法染色体分割，二进制對应的PID参数影射十进制對应的PID参数关系图

1. 评价

在计算适应度时(图 3)，使用PID参数控制系统的闭环响应，并计算性能指标。性能指标越小越好，而遗传算法通常求最大值，因此我们可以取性能指标的倒数或相反数作为适应度。在此计算 ISE （积分平方误差）為本次的性能指标, 用矩形法数值积分去近似積分計算。







图 3 遗传算法PID参数对应的性能指标图

完成后可得出各种群的适应情况，在由此去遗传操作（选择（复制）, 交叉 , 变异）

1. 选择（复制）

适应度越高的种群会有更高的概率去被选择，把转变成概率，概率又影射至

［0，1］的区间(图 4)，由rand 可公平地选择出更优胜的种群，也就是对应的PID 参数,达尔文的“适者生存”原理机制。淘汰解中的不适应，因素适应度高的优良个体有更大的机会被选中并繁殖后代，从而使优良特性(PID 参数)得以遗传。



图 4遗传算法复制概率影射到［0，1］原理图

1. 交叉

交叉会进行随机生成交叉起点和交叉终点，其中中间的参数会进行，利用循环和randperm生成两个在[a,b]区间的两个数，按位交換。注意，在此操作为在二进制矩阵中(图 5)



图 5遗传算法交换原理图



1. 变异

以较小的概率 取0.1），从群体 中随机选取若干个个体。进行“取反”运算（即 0 变 1 或 1 变 0）。如同自然界一样，某一位基因发生变异的概率很小。变异操作模拟了生物进化过程中偶然的基因突变现象。(图 6)



图 6遗传算法变异原理图



### 4.2PID永磁同步电动机控制系统设计

前人有不少工作[1]，可得系统是一个六阶系统

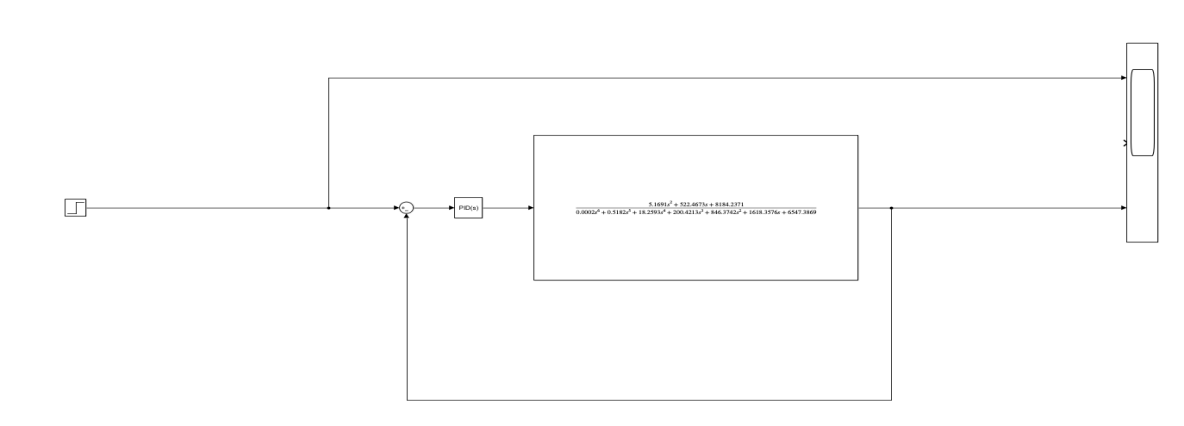


图 7 系统PID控制方框图

其中其传递函数为(图 8):

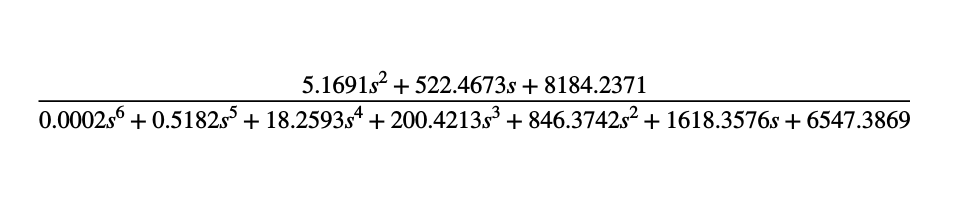


图 8 传递函数

其中系统的复杂性是前所未见，于是利用Z-N PID，去调节，其中只保留P环节，使其临界稳定，调节多次后并未发现稳定解，于是绘画了其系统的根轨迹图:

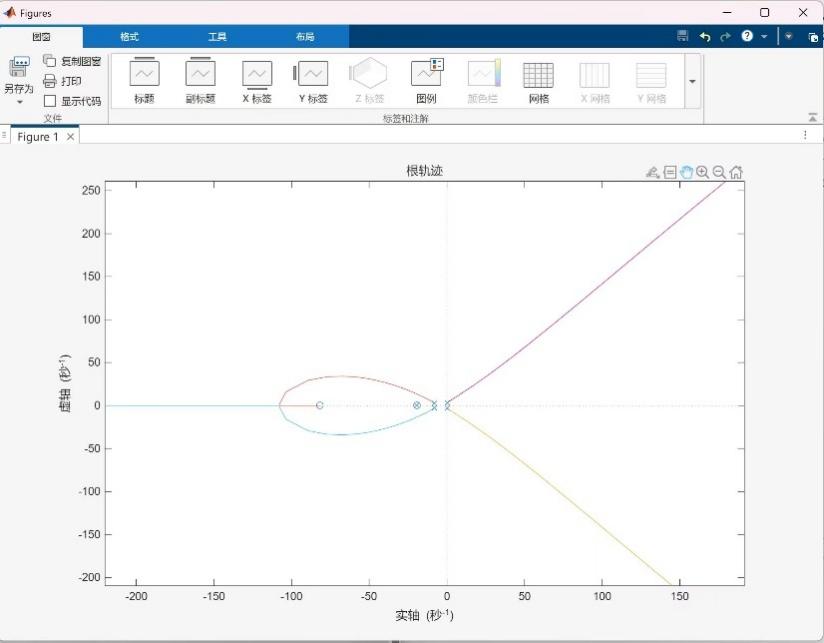


图 9根轨迹图

发现有四个极点都非常接近原点，其中有两个解位于S平面右边。根本无法使期临界稳定。利用Matlab工具箱，自动调节PID参数达到以下效果。其稳定时间達到247 秒(图 10)。

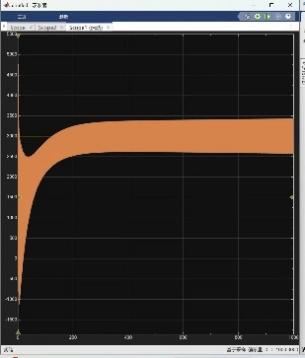


图 10 参数下系统的阶跃响应

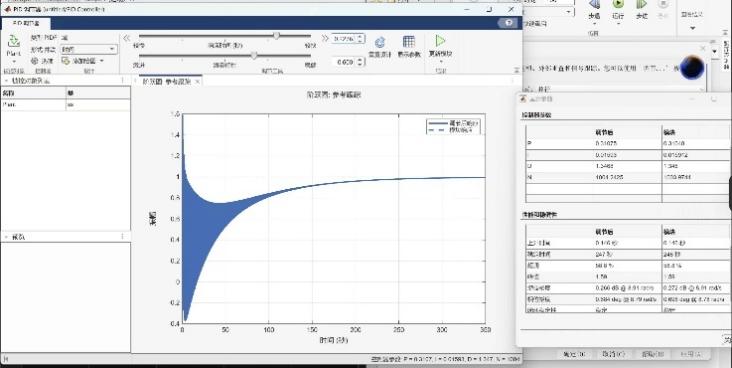


图 11 Matlab工具箱PID调节器

这个为系统的Bode 图和其对应的零极点位置[图 12]:

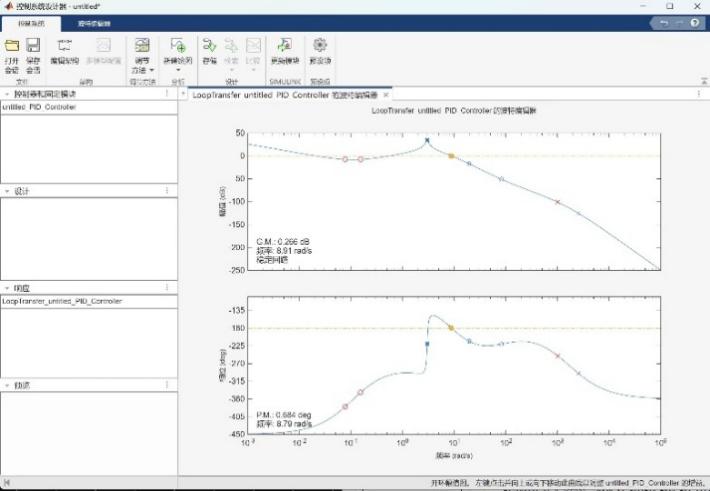


图 12 Matlab工具箱Bode PID调节器

因此急需一个可以对抗高阶系统的方法[9]，发现遗传算法，正是良药之一。

### 4.3编制程序

### 



### 4.4 仿真研究及分析

**4.4.1 PID参数范围的选择:**

对于PID参数范围的选择对于遗传选择PID极为重要，范围过大的PID参数范围会导致一开始性能指标极大，可见下图。因此以下三次分别选用相同的遗传算法参数, 和不同的PID参数范围:

popsize = 40; % 种群大小

chromlength = 120; % 染色体总长度（40位×3参数）

pc = 0.6; % 交叉概率

pm = 0.1; % 变异概率

G = 30; % 迭代次数

% PID参数范围 [Kp\_min, Kp\_max; Ki\_min, Ki\_max; Kd\_min, Kd\_max]

ranges = [0, 100; % Kp范围

0, 100; % Ki范围

0, 100]; % Kd范围

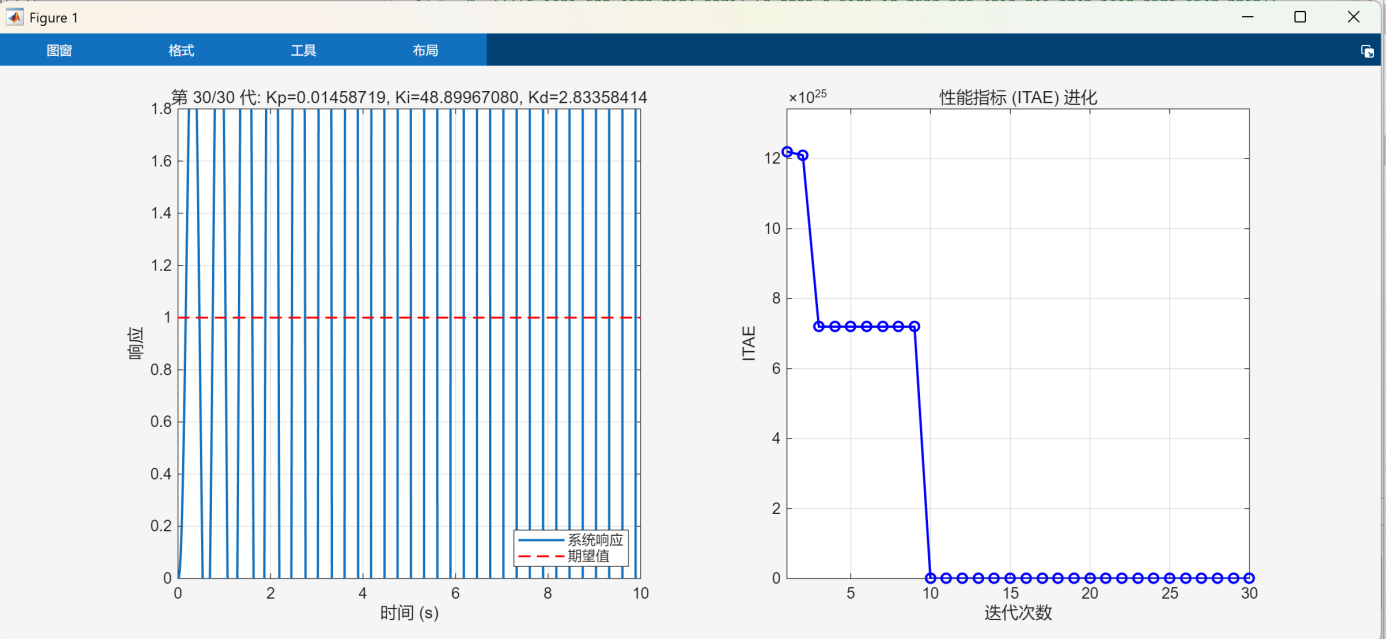


图 13 PID参数范围 [Kp\_min=0, Kp\_max=100; Ki\_min=0, Ki\_max=100; Kd\_min=0, Kd\_max=100]下的遗传算法的仿真。

可见其PID范围过大时，系统找不到稳定解。

PID参数范围 [Kp\_min=0, Kp\_max=100; Ki\_min=0, Ki\_max=100; Kd\_min=0, Kd\_max=100]

ranges = [0, 1.5; % Kp范围

0, 50; % Ki范围

0, 5]; % Kd范围

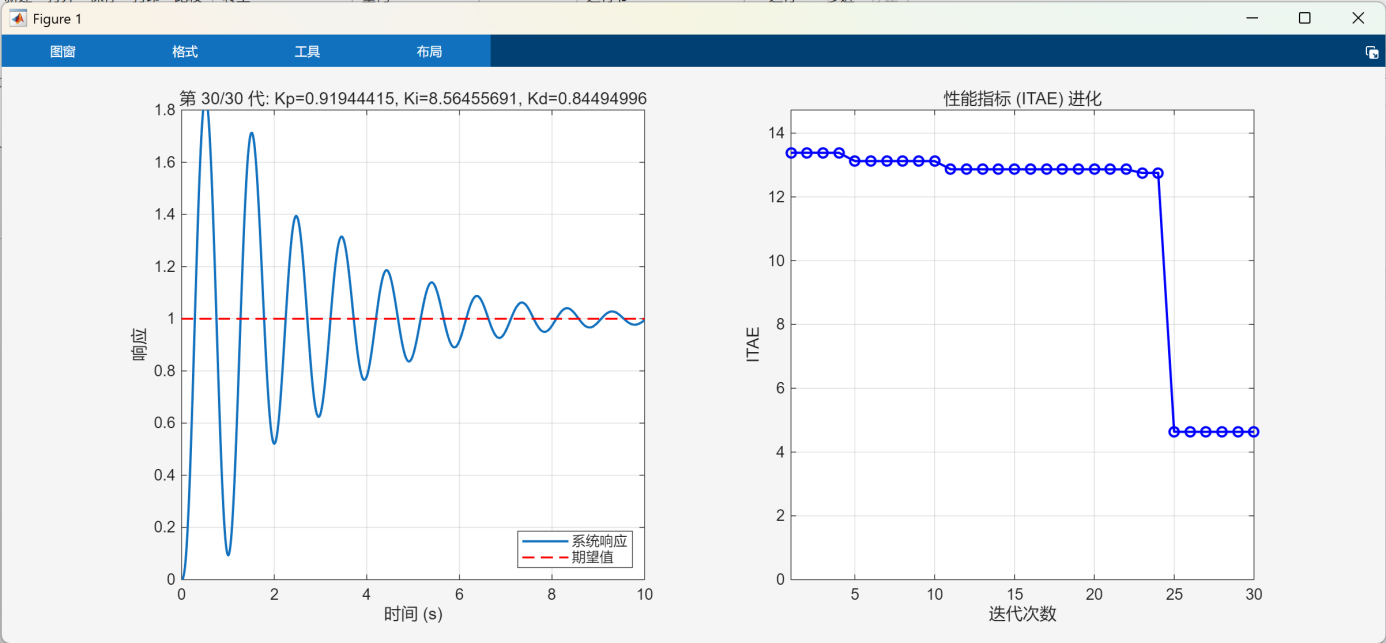


图 14 PID参数范围 [Kp\_min=0, Kp\_max=1.5; Ki\_min=0, Ki\_max=50; Kd\_min=0, Kd\_max=5]下的遗传算法的仿真。

% PID参数范围 [Kp\_min, Kp\_max; Ki\_min, Ki\_max; Kd\_min, Kd\_max]

ranges = [0, 1.5; % Kp范围

0, 10; % Ki范围

0, 5]; % Kd范围

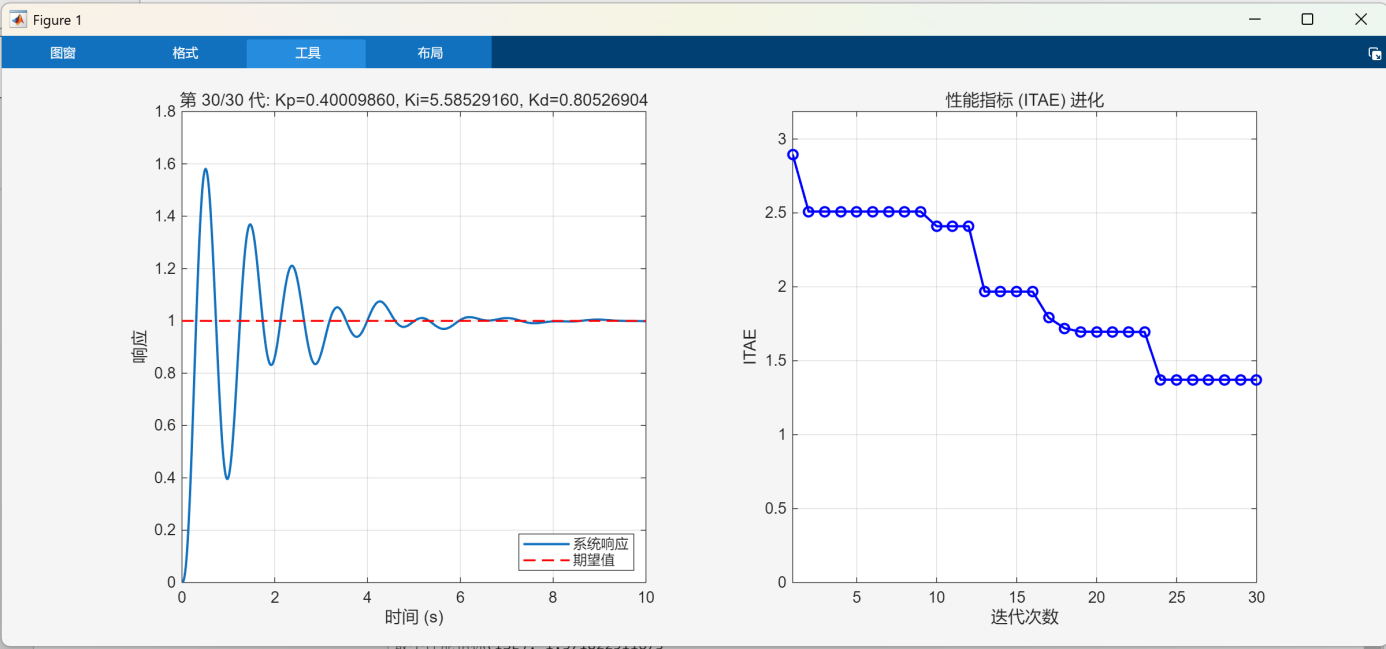


图 15 PID参数范围 [Kp\_min=0, Kp\_max=1.5; Ki\_min=0, Ki\_max=10; Kd\_min=0, Kd\_max=5]下的遗传算法的仿真。

可见不同的PID参数范围对于遗传算法学习性能有决定性影响，从不稳定[图 13]到稳定再到更好的性能。不合理的PID范围参数不但不能提升系统性能，反而会加重运行负担。如合理的PID参数范围，即可以快速到达最大值，亦有可能会使得错失全局最大值，落入局部最大值。

**4.4.2** **隨****迭代次數增加系統性能變化**

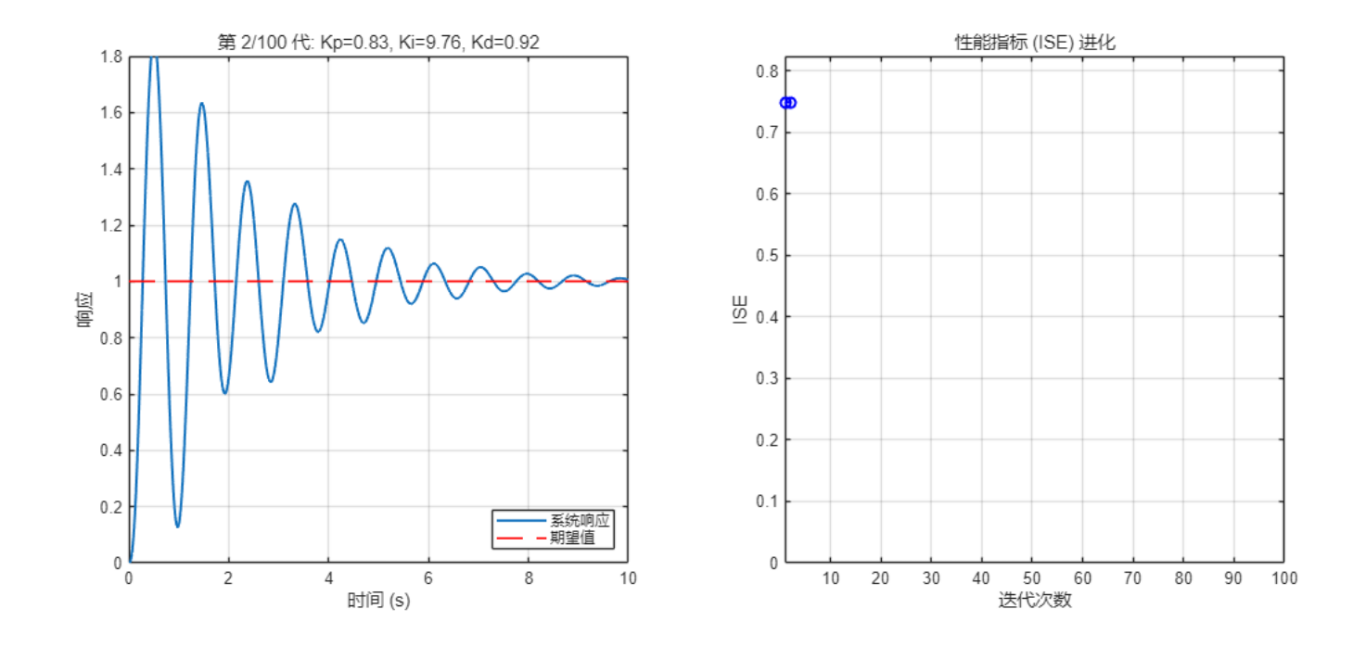
左图为不同迭代次数下的阶跃响应，可见右图的趋势逐渐下降，而不会上升，这是遗传算法的特点，优胜劣汰，每一次迭代都是性能的提升，而不会选择出更坏的参数。而到后来，ISE参数下降不明显，这是因为其族群已经十分相似，难以学习出更好的参数，可以说明他已经进局部最大值。图 17

图 16第2代次数下的阶跃响应(左)，性能指标 (ISE）进化图(右)

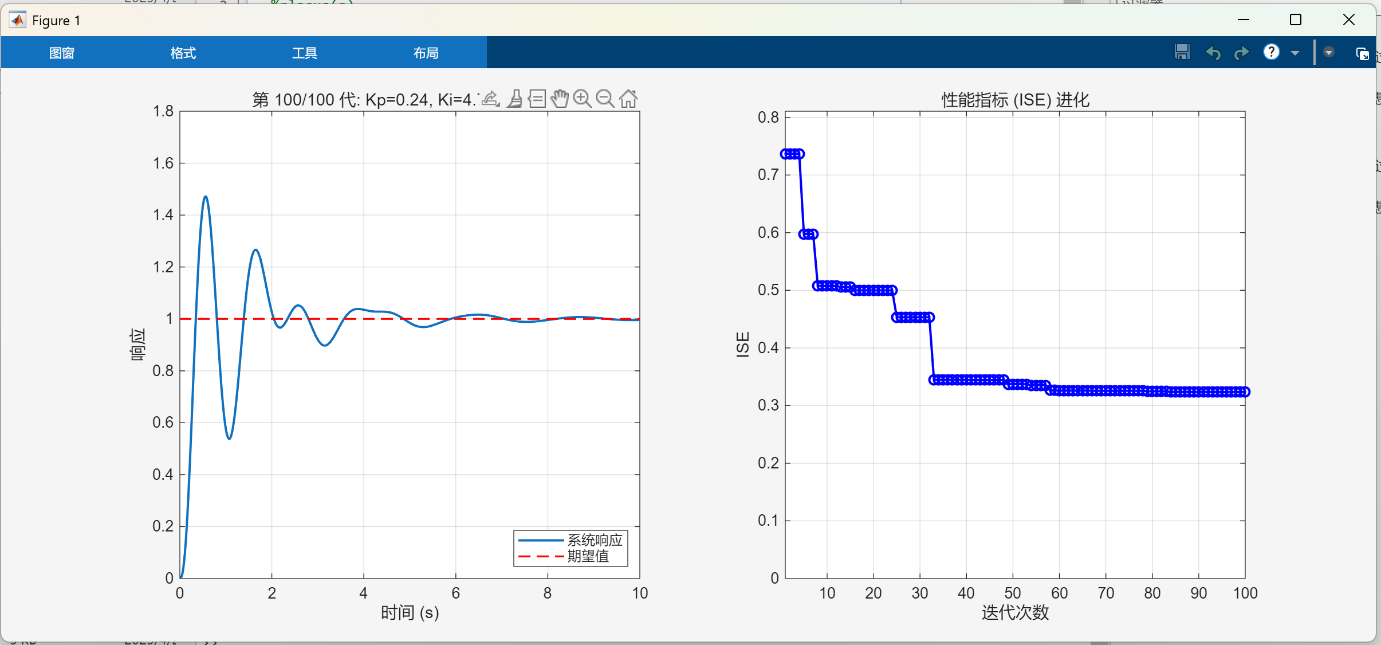


图 17第100代次数下的阶跃响应(左)，性能指标 （ISE) 进化图(右)

可看出第二次代[图 16]和最後一次[图 17]的阶跃响应，明显看出动态性能变好。说明随迭代次数增加系统性能变好，在後期其性能变化越小。

**4.4.3性能评估函數不同對系統的优劣**

而运用ISE作为评价指标时，经常会出现等幅震荡，因此选新的评价指标。除此之外 ITAE也是性能评估的一种，有别于ISE，ITAE其实只是在积分内添加时间t 的乘积，可以消除其稳定后的震荡，因为随着时间的增加，其惩罚项越大（也是添加t乘积的原因）。在此测试ITAE 评价指标和ISE评价指标的优劣之处，其测试结果与上述相等。ITAE相比于ISE更可以消除其稳定后的等幅震荡，但是ITAE会令时间初开始的误差积分近似为零，有可能会增加超调量。

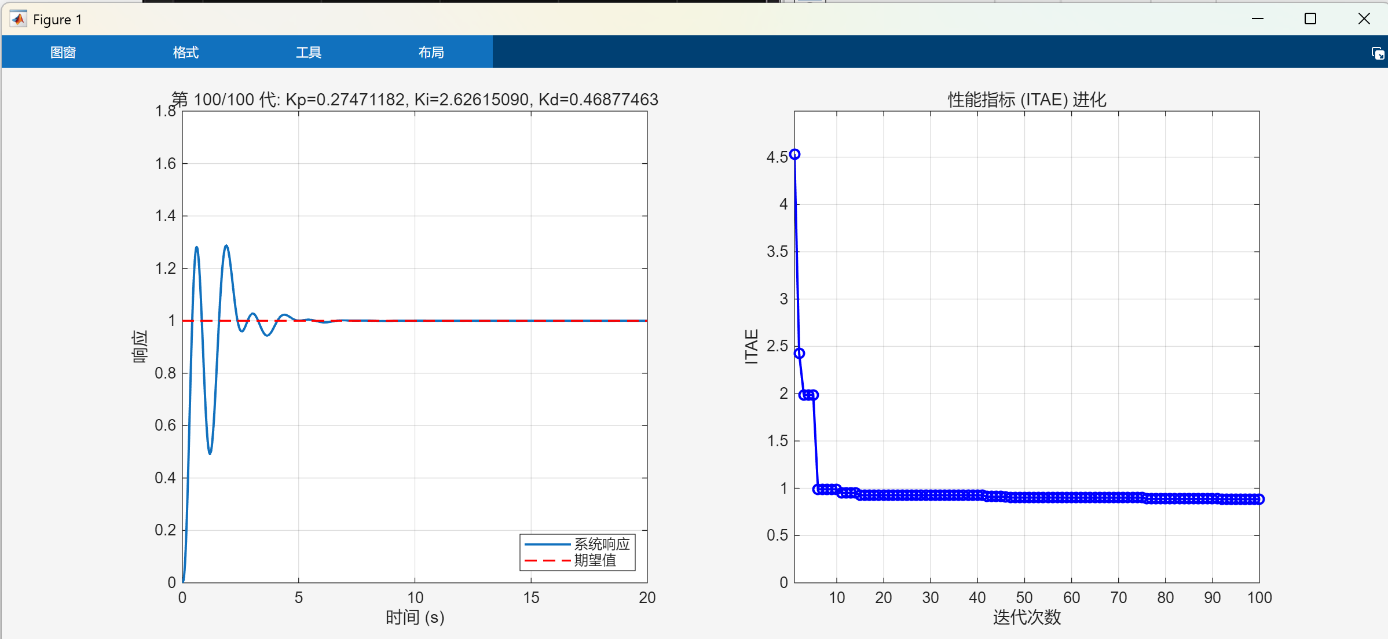


图 18第100代次数下的阶跃响应(左)，性能指标 （ITAE) 进化图(右)

### 

图 19由PID控制下得下系統阶跃响应

### 

### 

图 20遗传PID控制下得下系統阶跃响应

与普通PID调节对比可以发现，性能由248秒的调节时间下降至3.42秒，是极为强劲的调节手段，运行遗传算法PID时，电脑瞬间发出明显的风扇噪音，可以体现出其需要的运行性能。

|  |  |  |
| --- | --- | --- |
|  | **PID控制** | **遗传PID控制** |
| 上升时间 | 0.146秒 | 0.32秒 |
| 稳定时间 | 248秒 | 3.42秒 |
| 超调 | 58.8% | 47.3% |
| 峰值 | 1.59 | 1.47 |
| 震荡程度 | 极度震荡 | 輕微震荡 |
| 值 |  |  |
| 值 |  |  |
| 值 |  |  |

由程式跑出遗传PID 参数之后，导出其参数（精度为12位有效数字），用simulink 生成系统的阶跃响应，成功完成基于遺傳算法PID永磁同步电动机控制系统设计。(图 21)

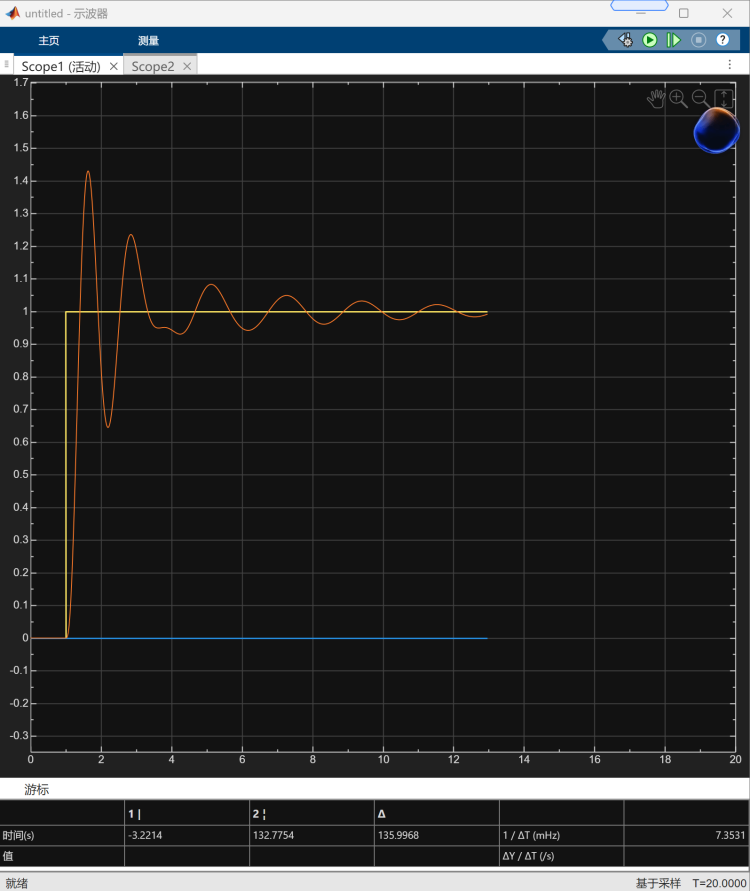


图 21在遗传PID控制下得出最優解後用simulink 的仿真系統阶跃响应

## 5 结束语

永磁同步电机作为高端装备的核心驱动，其控制性能优化关乎国家战略装备自主化与民生技术升级。本研究通过遗传算法重构PID参数整定机制，突破传统方法在六阶电机系统中的局限，显著提升动态响应性能（调节时间从248秒降至3.42秒）。实验验证了GA-PID在解决高阶、非线性控制问题上的有效性，尤其在高精度需求场景中展现潜力。

不同的PID参数范围对于遗传算法的学习表现有重要影响。不合理的PID范围不仅无法提升系统的性能，反而可能增加系统的负担。合理的PID参数范围能够快速达到最大值，但也可能导致错失全局最大值而陷入局部最大值。与ISE相比，ITAE更能够消除系统稳定后的等幅震荡，但ITAE在时间初期的误差积分接近于零，这可能会导致超调量的增加。

在这过程中学到了多方面的知识，包括科研绘图能力，把已经老掉牙的算法拾起来，解决了如此复杂系统的PID控制。我坚信有生物意味的算法可以进一步发展，成就通用人工智能AGI。把人类在繁忙的工作中解放出来，亦希望自己能成为推动时代的一分子，为人类科技发展而努力。

## 参考文献

[1]V Ramanaiah Nippatla, Srihari Mandava.Performance analysis of permanent magnet synchronous motor based on transfer function model using PID controller tuned by Ziegler-Nichols method,Results in Engineering,2025,26:105460,

[2]董凡.基于永磁同步电机的高速列车牵引系统的仿真分析[D].浙江大学,2016.

[3]盛义发,喻寿益,桂卫华,等.轨道车辆用永磁同步电机系统弱磁控制策略[J].中国电机工程学报,2010,30(09):74-79

[4]童怀,刘继辉.永磁同步电动机移相弱磁控制的仿真分析[J].微特电机,2006,(08):17-19.

[5]卢琴芬,沈燚明,叶云岳.永磁直线电动机结构及研究发展综述[J].中国电机工程学报,2019,39(09):2575-2588

[6]刘金琨.先进PID控制MATLAB仿真[M].电子工业出版社:201606.545.

[7]王伟,张晶涛,柴天佑.PID参数先进整定方法综述[J].自动化学报,2000,(03):347-355

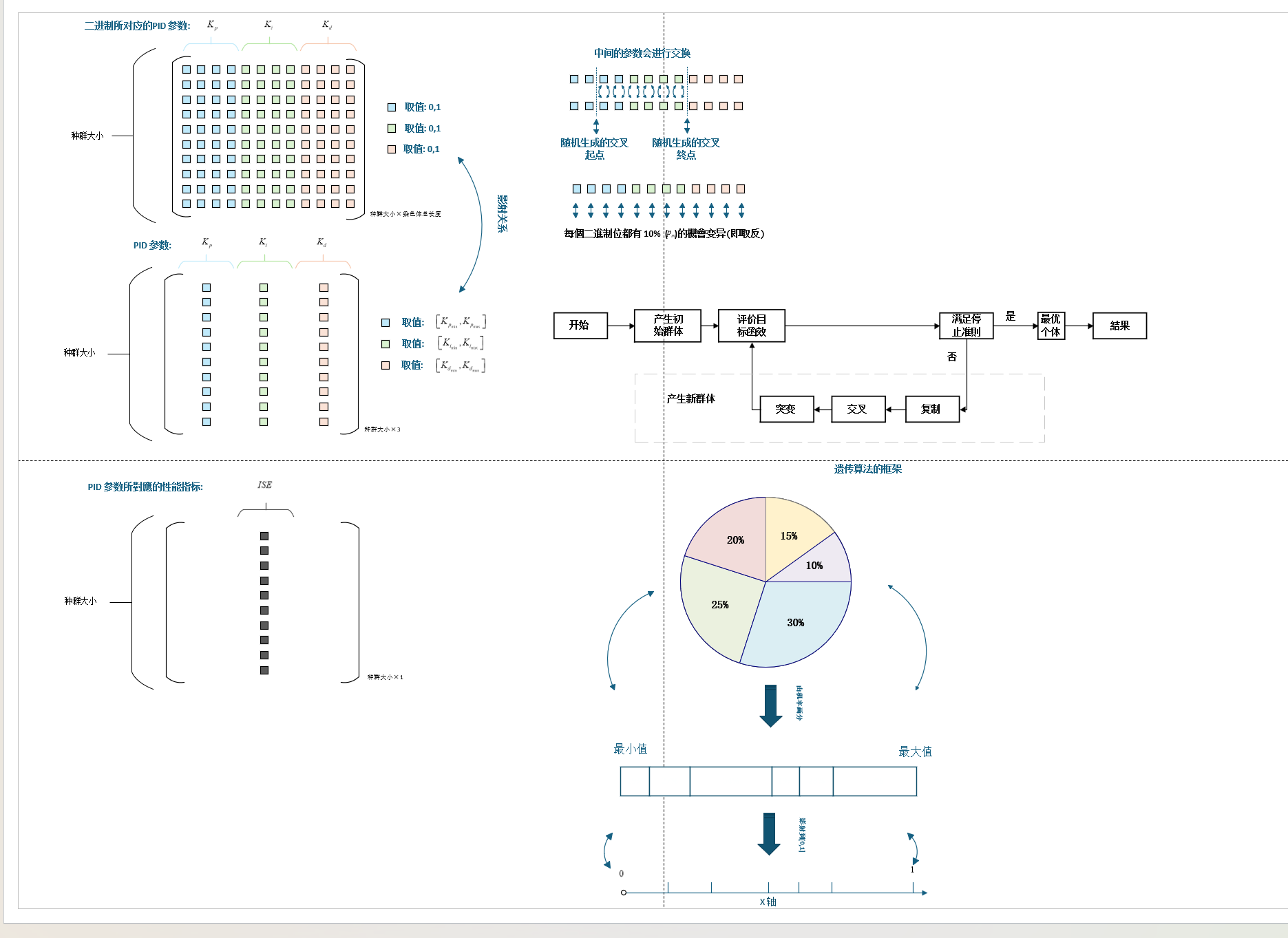
[8] D. Debnath, P. Malla, S. Roy, Position control of a DC servo motor using various

controllers: a comparative study, Mater. Today Proc.,2022,58 :484–488

[9]杨智民,王旭,庄显义.遗传算法在自动控制领域中的应用综述[J].信息与控制,2000,(04):329-339

[10]陈根社,陈新海.遗传算法的研究与进展[J].信息与控制,1994,(04):215-222.

## 附錄



%a=tf([5.1691 522.4673 8184.2371],[0.0002 0.5182 18.2593 200.4213 846.3742 1618.3576 6547.3869]);

%rlocus(a)

%%主程序

function GA()

clear; close all; clc;

% 定义被控系统传递函数

a = tf([5.1691, 522.4673, 8184.2371], [0.0002, 0.5182, 18.2593, 200.4213, 846.3742, 1618.3576, 6547.3869]);

% 遗传算法参数

popsize = 40; % 种群大小

chromlength = 120; % 染色体总长度（40位×3参数）

pc = 0.6; % 交叉概率

pm = 0.1; % 变异概率

G = 100; % 迭代次数

% PID参数范围 [Kp\_min, Kp\_max; Ki\_min, Ki\_max; Kd\_min, Kd\_max]

ranges = [0, 5; % Kp范围

0, 10; % Ki范围

0, 5]; % Kd范围

% 初始化种群

pop = round(rand(popsize, chromlength));

J\_history = zeros(1, G); % 存储每代最小性能指标

best\_params = zeros(G, 3); % 存储每代最优参数[Kp, Ki, Kd]

% 初始种群评估

decpop = bintodec(pop, popsize, chromlength, ranges);

fx = calobjvalue(decpop, a);

[best\_J, idx] = min(fx);

best\_params(1,:) = decpop(idx,:);

J\_history(1) = best\_J;

% 创建图形窗口

figure;

set(gcf, 'Position', [100, 100, 1200, 500]);

% 主循环

for i = 2:G

% 计算适应度

decpop = bintodec(pop, popsize, chromlength, ranges);

fx = calobjvalue(decpop, a);

fitvalue = calfitvalue(fx);

% 遗传操作

newpop = copyx(pop, fitvalue, popsize);

newpop = crossover(newpop, pc, popsize, chromlength);

newpop = mutation(newpop, pm, popsize, chromlength);

% 评估新种群

new\_decpop = bintodec(newpop, popsize, chromlength, ranges);

new\_fx = calobjvalue(new\_decpop, a);

new\_fitvalue = calfitvalue(new\_fx);

% 更新种群（精英保留）

index = new\_fitvalue > fitvalue;

pop(index, :) = newpop(index, :);

% 评估当前种群

decpop = bintodec(pop, popsize, chromlength, ranges);

fx = calobjvalue(decpop, a);

[best\_J, idx] = min(fx);

J\_history(i) = best\_J;

best\_params(i,:) = decpop(idx,:);

% 绘制结果

plot\_results(a, best\_params(i,:), J\_history(1:i), i, G);

end

% 输出最终结果

[opt\_J, gen] = min(J\_history);

opt\_params = best\_params(gen,:);

fprintf('最优参数: Kp=%.4f, Ki=%.4f, Kd=%.4f\n', opt\_params(1), opt\_params(2), opt\_params(3));

fprintf('最小性能指标(ISE): %.4f\n', opt\_J);

% 绘制最终响应曲线

figure;

plot\_final\_response(a, opt\_params);

end

%% 二进制转十进制（支持多参数）

function decpop = bintodec(pop, popsize, chromlength, ranges)

seg\_len = chromlength / 3; % 每段长度

decpop = zeros(popsize, 3); % [Kp, Ki, Kd]

for i = 1:popsize

% 转换Kp

bin\_part = pop(i, 1:seg\_len);

dec\_val = sum(bin\_part .\* (2.^(seg\_len-1:-1:0)));

decpop(i,1) = ranges(1,1) + dec\_val \* (ranges(1,2)-ranges(1,1)) / (2^seg\_len-1);

% 转换Ki

bin\_part = pop(i, seg\_len+1:2\*seg\_len);

dec\_val = sum(bin\_part .\* (2.^(seg\_len-1:-1:0)));

decpop(i,2) = ranges(2,1) + dec\_val \* (ranges(2,2)-ranges(2,1)) / (2^seg\_len-1);

% 转换Kd

bin\_part = pop(i, 2\*seg\_len+1:end);

dec\_val = sum(bin\_part .\* (2.^(seg\_len-1:-1:0)));

decpop(i,3) = ranges(3,1) + dec\_val \* (ranges(3,2)-ranges(3,1)) / (2^seg\_len-1);

end

end

%% 计算目标函数（性能指标）

function fx = calobjvalue(decpop, sys)

fx = zeros(1, size(decpop,1));

for i = 1:size(decpop,1)

Kp = decpop(i,1);

Ki = decpop(i,2);

Kd = decpop(i,3);

% 创建PID控制器

C = pid(Kp, Ki, Kd);

% 构建闭环系统

closed\_loop = feedback(series(C, sys), 1);

try

% 仿真阶跃响应

[y, t] = step(closed\_loop, 0:0.01:10);

e = 1 - y; % 计算误差

% 计算ISE（积分平方误差）

ISE = sum(e.^2) \* 0.01; % 矩形法数值积分

fx(i) = ISE;

catch

% 处理不稳定系统

fx(i) = 1e10;

end

end

end

%% 计算适应度

function fitvalue = calfitvalue(fx)

% 使用倒数形式（最小化问题转为最大化问题）

fitvalue = 1./(fx + 1e-6); % 加小常数避免除零

end

%% 绘制迭代结果

function plot\_results(sys, params, J\_history, gen, max\_gen)

% 获取当前最优PID参数

Kp = params(1);

Ki = params(2);

Kd = params(3);

% 创建控制器和闭环系统

C = pid(Kp, Ki, Kd);

closed\_loop = feedback(series(C, sys), 1);

% 仿真阶跃响应

[y, t] = step(closed\_loop, 0:0.01:10);

% 绘制阶跃响应

subplot(1,2,1);

plot(t, y, 'LineWidth', 1.5);

hold on;

plot([t(1), t(end)], [1, 1], 'r--', 'LineWidth', 1.2);

hold off;

title(sprintf('第 %d/%d 代: Kp=%.2f, Ki=%.2f, Kd=%.2f', gen, max\_gen, Kp, Ki, Kd));

xlabel('时间 (s)');

ylabel('响应');

grid on;

axis([0, 10, 0, 1.8]);

legend('系统响应', '期望值', 'Location', 'Southeast');

% 绘制性能指标进化

subplot(1,2,2);

plot(1:gen, J\_history, 'b-o', 'LineWidth', 1.5);

title('性能指标 (ISE) 进化');

xlabel('迭代次数');

ylabel('ISE');

grid on;

axis([1, max\_gen, 0, max(J\_history)\*1.1]);

drawnow;

end

%% 绘制最终响应曲线

function plot\_final\_response(sys, params)

% 创建控制器和闭环系统

C = pid(params(1), params(2), params(3));

closed\_loop = feedback(series(C, sys), 1);

% 仿真阶跃响应

[y, t] = step(closed\_loop, 0:0.01:10);

% 计算性能指标

e = 1 - y;

ISE = sum(e.^2) \* 0.01;

IAE = sum(abs(e)) \* 0.01;

ITAE = sum(t.\*abs(e)) \* 0.01;

% 绘制响应曲线

plot(t, y, 'LineWidth', 2);

hold on;

plot([t(1), t(end)], [1, 1], 'r--', 'LineWidth', 1.5);

hold off;

title(sprintf('最优PID控制: Kp=%.4f, Ki=%.4f, Kd=%.4f', params(1), params(2), params(3)));

subtitle(sprintf('ISE=%.4f, IAE=%.4f, ITAE=%.4f', ISE, IAE, ITAE));

xlabel('时间 (s)');

ylabel('响应');

grid on;

axis([0, 10, 0, 1.8]);

legend('系统响应', '期望值', 'Location', 'Southeast');

end

% ---------------------- 以下遗传操作函数保持不变 ----------------------

%% 复制操作

function newx = copyx(pop, fitvalue, popsize)

newx = pop;

p = fitvalue / sum(fitvalue);

Cs = cumsum(p);

R = sort(rand(popsize,1));

j = 1;

for i = 1:popsize

while R(i) > Cs(j)

j = j + 1;

end

newx(i,:) = pop(j,:);

end

end

%% 交叉操作

function newx = crossover(pop, pc, popsize, chromlength)

newx = pop;

i = 2;

while i <= popsize-1

if rand < pc

% 选择交叉点

points = sort(randperm(chromlength, 2));

start\_idx = points(1);

end\_idx = points(2);

% 执行交叉

temp = newx(i-1, start\_idx:end\_idx);

newx(i-1, start\_idx:end\_idx) = newx(i, start\_idx:end\_idx);

newx(i, start\_idx:end\_idx) = temp;

end

i = i + 2;

end

end

%% 变异操作

function newx = mutation(pop, pm, popsize, chromlength)

newx = pop;

for i = 1:popsize

if rand < pm

% 随机选择变异位

idx = randi(chromlength);

newx(i, idx) = ~newx(i, idx);

end

end

end

全文完