基于遗传算法的回焊炉温度控制优化设计

摘 要

回焊炉是自动化焊接集成电路板的设备，回焊炉温度控制对产品质量至关重要。本文在一维热传导模型的基础上建立了单目标优化模型与多目标优化模型，针对不同需求优化回焊炉各温区温度，通过最小二乘法、有限差分法、二分法、遗传算法进行求解。

针对问题一：将传热过程视为一维热传导过程，考虑热量从炉内空气向电路板表面的传递与电路板内部的传热，建立**一维热传导模型**。对模型进行**有限差分化**得出隐式差分形式，基于**追赶法**求解。对于各温区间隙，通过**线性插值**得到对应位置的炉内空气温度值。对模型中导热率等参数利用**最小二乘法**进行估计，使各时刻平均相对误差控制在 0.5%，并以此构建后续问题的模型。模型求出问题一参数下小温区 3、6、7 中点温度分别为 129.98 ℃、168.59 ℃、189.71 ℃，小温区 8 结束处的温度为 222.80 ℃，同时得到炉温曲线与每隔 0.5 s 的焊接中心温度值。针对结果进行检验，发现加热过程边缘温度高于中心温度，冷却过程边缘温度低于中心温度，与实际相符。

针对问题二：在一维热传导模型的基础上，将传送带过炉速度视为目标函数与决策变量，将制程限制转化为约束条件，构建**单目标优化模型**。在经过大步长搜索后采用**二分法**进行精细搜索，得出的最大传送带过炉速度为 81.5691 cm/s。分析发现过炉速度高于此值时峰值温度无法达到 240 ℃的最低要求；过炉温度过低时，电路板中心在高温阶段所处时间过长。

针对问题三：将所求面积此作为目标函数求其最小值，视各大温区温度与传送带过炉速度为决策变量，沿用问题二的约束条件，建立**单目标优化模型**。面积的计算采用**梯形法**近似求解，模型通过**遗传算法**进行求解，并以问题二所给参数作为初值进行迭代。求解得到的最小面积为 420.1754 ℃·s，此时的小温区 1~5、6、7、8~9 温度设定值分别为 176.2229 ℃、190.3380 ℃、230.9501 ℃、264.6106 ℃，传送带过炉速度为 90.7978 cm/s。

针对问题四：构建炉温曲线对称性评价指标，将此作为额外目标与问题三模型合并，建立**双目标优化模型**。采用加权和法对两目标加权，转化为单目标优化模型。使用遗传算法求解，将问题三所得最优点作为初值进行迭代。求解得到的最小面积为 421.2770 ℃·s，左右两侧温度平均差异为 2.2102 ℃，此时的小温区 1~5、6、7、8~9 温度设定值分别为 184.1790 ℃、196.1701 ℃、237.5578 ℃、264.6693 ℃，传送带过炉速度为 97.6667 cm/s。

关键词：热传导模型；最小二乘法；有限差分法；二分法；优化模型

一、问题重述

问题研究集成电路板元件焊接时的温度控制与调整。回焊炉是元件自动焊接的设备，内部设有多个小温区。某回焊炉内部有 11 个小分区，按功能分为预热区、恒温区、回流区、冷却区四个大温区。电路板自预热区开始匀速进入。

给定每个小温区长度为 30.5 cm，间隙为 5 cm，炉前、炉后区域长 25 cm，车间温度为 25 ℃。给定温区设定温度、传送带速度与该体系下的炉温曲线。

问题一：建立数学模型，求解传送带速度 78 cm/min、各小温区温度为表 1 值时的炉温曲线。

表 1 问题一温区设定温度

小温区序号	1~5	6	7	8
设定温度	173 ℃	198 ℃	230 ℃	257 ℃

问题二：炉温曲线需满足制程界限，求解各温区温度为表 2 值时的最大传送带过炉速度。

表 2 问题二温区设定温度

小温区序号	1~5	6	7	8
设定温度	173 ℃	198 ℃	230 ℃	257 ℃

问题三：要求炉温曲线超过 217 ℃到峰值温度所覆盖的面积最小，确定最优炉温曲线与各温区温度设定值与传送带过炉速度。

问题四：要求峰值温度为中心线的两侧超过 217 ℃的炉温曲线应尽量对称，给出最优炉温曲线与各温区温度设定值与传送带过炉速度。

二、问题分析

2.1 问题一的分析

问题一需要基于已有数据构建热传导模型，调整模型参数使模型结果与实测值相符。

视热量传导过程为一维热传导过程，构建一维热传导模型。导出描述这一过程的泛定方程，确定初始条件与边界条件。通过有限差分法将偏微分方程转化为差分方程，求出数值解。

在上述热传导模型中，由导热率、比热容、密度等确定的常数是待定的。基

于最小二乘法利用附件数据进行参数确定。将题中数据代入模型，绘出炉温曲线。

2.2 问题二的分析

问题二需要在给定温区温度的基础上，对传送带过炉速度进行优化。

在问题一模型的基础上，构建单因素优化模型，通过二分法求出符合制程界限的最大传送带过炉速度。

2.3 问题三的分析

问题三需要确定各温区设定温度与传送带过炉速度，使 217 ℃到峰值温度所覆盖的面积最小。

采用梯形法对所求面积求数值积分，并以此为目标函数。视各温区温度与传送带过炉速度为决策变量，沿用问题二的约束条件描述制程限制，建立多因素优化模型。由于决策变量多，相互关系复杂，因此考虑使用遗传算法进行求解。

2.4 问题四的分析

问题四在问题三的基础上额外增加一个目标：使峰值温度为轴线的两侧超过 217 ℃部分的炉温曲线应尽量对称。因此，在问题三模型的基础上建立多目标、多因素优化模型。将两目标函数整合，以此将模型转化为单目标优化模型。采用遗传算法进行求解。

三、模型假设

1. 使问题中的热量传导过程为一维热传导过程，仅考虑热量从电路板上下两侧的传导过程，使电路板侧面绝热。
2. 忽略温度的微小变化对物体比热容的影响，考虑热导率在各温区内独立且恒定。

四、符号说明

表 3 符号说明

符号	符号意义	单位
$u(x,t)$	温度 u 关于电路板厚度 x 与时间 t 的函数	℃
T_0	环境温度	℃
d	电路板厚度	mm
Δt	时间差分步长	s
h	空间差分步长	mm

五、模型的建立与求解

5.1 问题一模型的建立与求解

5.1.1 一维热传导模型的建立

1.泛定方程

热流密度 q 是单位时间流过单位面积的热量：

$$q=\frac{Q}{tS}。\tag{1}$$

傅里叶实验定理表明热量密度与温度的下降率成正比，即

$$q=-k\frac{\partial T}{\partial \boldsymbol{n}},\tag{2}$$

其中 k 为热导率，$\boldsymbol{n}$ 为外法线方向的单位矢量。

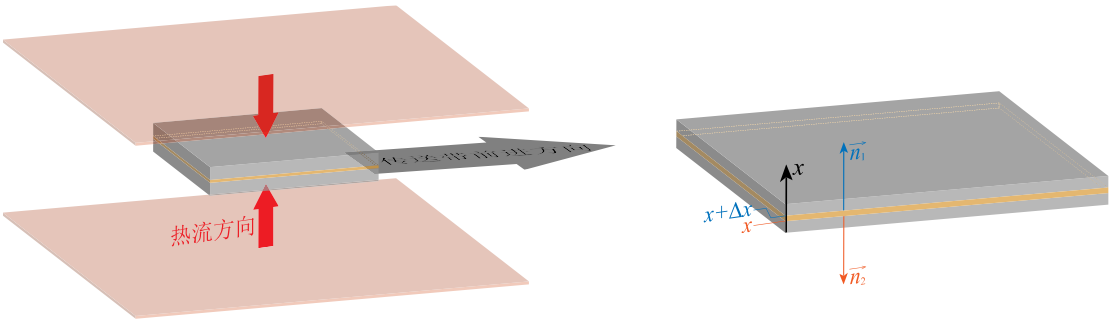

图 1 一维热量传导过程示意图

考虑区间 $[x,x+\Delta x]$ 在时间间隔 Δt 内的热量流动情况，如图 1 所示。联立(1)式与(2)式，流入 x 面的热量为

$$Q_1=-\left(-k\left(\frac{\partial u}{\partial(-x)}\bigg|_x\right)A\Delta t\right)=-k\frac{\partial u}{\partial x}\bigg|_x A\Delta t,$$

流出 $x+\Delta x$ 的热量为

$$Q_2=-k\frac{\partial u}{\partial x}\bigg|_{x+\Delta x}A\Delta t。$$

式中 u 为温度，是关于 x 与 t 的函数；A 为元面积。

升温所需要的热量为

$$Q=cm\Delta T=c(\rho A\Delta x)\left[u(x,t+\Delta t)-u(x,t)\right],$$

式中 c 为电路板比热容，m 为电路板质量，ρ 为电路板密度，均为常量。

由热力学第一定理（能量守恒定律），应有

$$Q = Q_1 - Q_2 ,$$

即

$$c\rho A\Delta x\left[u(x,t+\Delta t)-u\left(x,t\right)\right]=-kA\Delta t\left[u_x(x+\Delta x,t)-u_x\left(x,t\right)\right]。\tag{3}$$

考虑$\Delta t\to 0$ 时，

$$\lim_{\Delta t\to 0} Q = c\rho A\Delta x\Delta t \lim_{\Delta t\to 0}\frac{u\left(x,t+\Delta t\right)-u\left(x,t\right)}{\Delta t} = c\rho A u_t \Delta x\Delta t ,\tag{4}$$

亦有$\Delta x\to 0$，因此

$$\lim_{\Delta x\to 0} Q_1 - Q_2 = -kA\Delta t\Delta x \lim_{\Delta x\to 0}\frac{u_x\left(x+\Delta x,t\right)-u_x\left(x,t\right)}{\Delta x} = -kA u_{xx}\Delta t\Delta x 。\tag{5}$$

(4)(5)两式代入(3)式并整理，得描述一维热传导过程的偏微分方程

$$u_t = Du_{xx} ,\ [1]$$

式中 $D=\dfrac{k}{c\rho}$ 为常数，且有 $D>0$ 。

2.定解条件

①t=0 时，电路板位于炉前区域外，内部温度应为室温，即有初始条件

$$u\left(x,t\right)\big|_{t=0} = 25\,℃。\tag{6}$$

②x=0 与 x=d 处是电路板上下表面处，由牛顿冷却定律[2]得边界条件

$$\left.\frac{\partial u\left(x,t\right)}{\partial x}\right|_{x=0} = \sigma\left[u\left(x,t\right)\big|_{x=0} - U\left(t\right)\right],\tag{7}$$

$$\left.\frac{\partial u\left(x,t\right)}{\partial x}\right|_{x=d} = -\sigma\left[u\left(x,t\right)\big|_{x=0} - U\left(t\right)\right].\tag{8}$$

式中，U 为炉内温度，是关于时间 t 的函数；σ 是表面传热系数。

综上所述，建立一维热传导模型

$$u_t = Du_{xx} ,\tag{9}$$

$$\begin{cases} u(x,t)\big|_{t=0} = T_0, \\ \left.\dfrac{\partial u(x,t)}{\partial x}\right|_{x=0} = \sigma\left[u(x,t)\big|_{x=0} - U(t)\right], \\ \left.\dfrac{\partial u(x,t)}{\partial x}\right|_{x=d} = -\sigma\left[u(x,t)\big|_{x=d} - U(t)\right]. \end{cases} \tag{10}$$

5.1.2 一维热传导模型的求解

1.炉前、炉后与小温区间隙区域的温度处理

对于不做温度控制的位置，假设其某一位置的温度有左右两受控温度线性插值得来，不失一般性，设小温区 1~5、6、7、8~9 的设定温度分别为 T_1~T_4，炉内各位置的温度如表 4 所示。图 2 示实验数据条件下环境温度与位置的关系。

表 4 不同位置的温度

x 范围	位置描述	温度函数
[0,25 cm]	炉前区域	$\dfrac{T_1}{25\ \text{cm}}x$
(25 cm, 197.5 cm]	小温区 1~5	T_1
(197.5 cm, 202.5 cm]	5、6 间隙	$\dfrac{T_2-T_1}{5\ \text{cm}}(x-197.5\ \text{cm})+T_1$
(202.5 cm, 233 cm]	小温区 6	T_2
(233 cm, 238 cm]	6、7 间隙	$\dfrac{T_3-T_2}{5\ \text{cm}}(x-233\ \text{cm})+T_2$
(238 cm, 268.5 cm]	小温区 7	T_3
(268.5 cm, 273.5 cm]	7、8 间隙	$\dfrac{T_4-T_3}{5\ \text{cm}}(x-268.5\ \text{cm})+T_3$
(273.5 cm, 339.5 cm]	小温区 8~9	T_4
(339.5 cm, 344.5 cm]	9、10 间隙	$\dfrac{25\ ℃-T_4}{5\ \text{cm}}(x-339.5\ \text{cm})+T_4$
(344.5 cm, 435.5 cm]	小温区 10、11，炉后区域	25 ℃

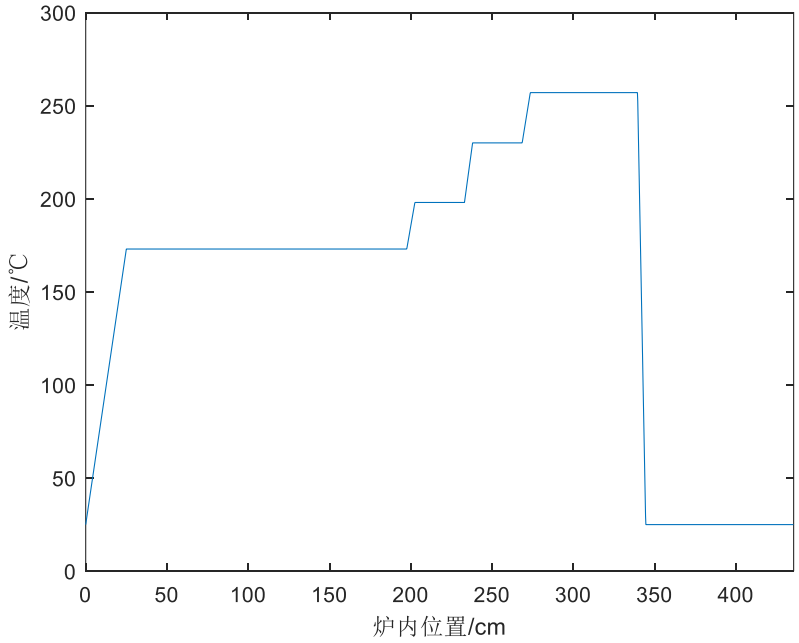

图 2 不同位置下的炉内温度

2. 有限差分化

由于定解条件复杂，因此对(9)(10)两式有限差分化，求出数值解。

（1）泛定方程差分化

对(9)式向后差分，得

$$\frac{u_j^k-u_j^{k-1}}{\Delta t}-D\frac{u_{j+1}^k-2u_j^k+u_{j-1}^k}{h^2}=0\text{。}^{[3]}$$

式中，时间 t 的步长为Δt，位置 x 的步长为 h，记$u(x_j,t_k)=u(jh,k\Delta t)$。

（2）定解条件差分化

初始时刻 k=0，此时

$$u_j^0=T_0, \tag{11}$$

对(7)式向前差分，得

$$\frac{u_1^k-u_0^k}{h}=\sigma\left[u_0^k-U(t)\right], \tag{12}$$

对(8)式向后差分，得

$$\frac{u_{j_{\max}}^k-u_{j_{\max}-1}^k}{h}=-\sigma\left[u_{j_{\max}}^k-U(t)\right]\text{。} \tag{13}$$

综上所述，有限差分化后的模型为

$$\frac{u_j^k-u_j^{k-1}}{\Delta t}-D\frac{u_{j+1}^k-2u_j^k+u_{j-1}^k}{h^2}=0 \tag{14}$$

$$\begin{cases} u_j^0=T_0, \\ \dfrac{u_1^k-u_0^k}{h}=\sigma\left[u_0^k-U(t)\right], \\ \dfrac{u_{j_{\max}}^k-u_{j_{\max}-1}^k}{h}=-\sigma\left[u_{j_{\max}}^k-U(t)\right]. \end{cases}$$

（3）基于追赶法的求解

隐式差分方程递推方式如下，图 3 示该过程。

递推步骤

***step*1.** 对求解区域进行网格分割。创建递推网格：将空间划分为 m 等份，即 $h=\dfrac{d}{m}$；在时间维度划分为 n 等份，即 $\Delta t=\dfrac{l/v}{n}$，其中 l 为回焊炉总长度。

***step*2.** 由(11)式确定初始条件，并将其代入。

***step*3.** 由(14)式完成 k=1 的内部递推。

***step*4.** 由(12)式、(13)式完成 k=1 的边界递推。

***step*5.** k 自增，返回 *step*3，直至 k=n。

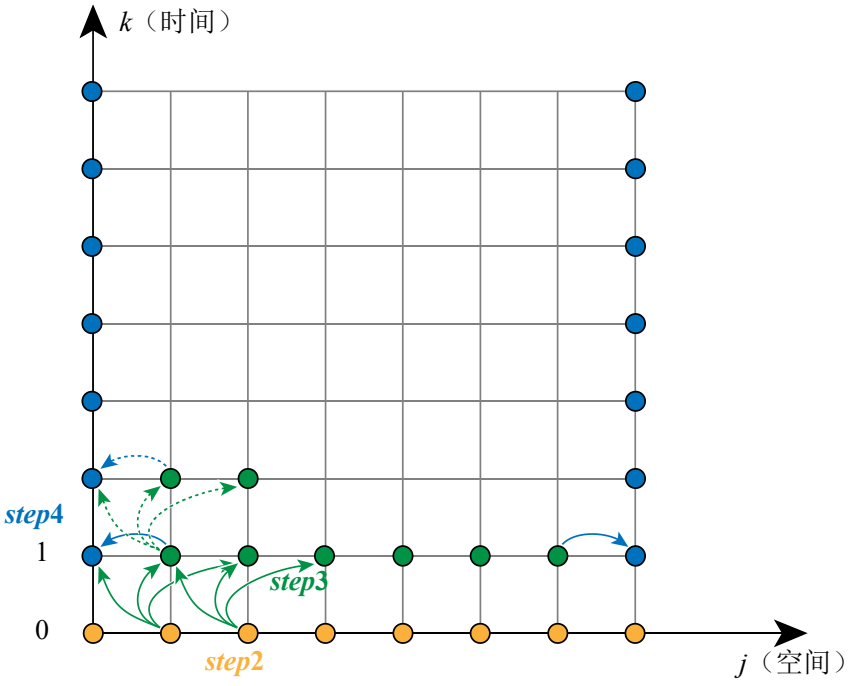

图 3 模型递推过程示意图

在 *step*3 与 *step*4 进行递推时，需要联立相同 k 值的方程组进行求解：

$$\begin{cases} (1+h\sigma)u_0^k - u_1^k = h\sigma U(t), \\ -ru_0^k + (2r+1)u_1^k - ru_2^k = u_1^{k-1}, \\ \vdots \\ -ru_{j-1}^k + (2r+1)u_j^k - ru_{j+1}^k = u_j^{k-1}, \\ \vdots \\ -ru_{j_{\max}-1}^k + (2r+1)u_{j_{\max}}^k - ru_{j_{\max}+1}^k = u_{j_{\max}}^{k-1}, \\ (1+h\sigma)u_{j_{\max}}^k - u_{j_{\max}-1}^k = h\sigma U(t). \end{cases}$$

其中，$r = a\dfrac{\Delta t}{h^2}$，将其写为矩阵乘法形式：

$$\boldsymbol{Au} = \boldsymbol{B},$$

其中，

$$\boldsymbol{A} = \begin{pmatrix} 1+h\sigma & -1 & & & & & & \\ -r & 2r+1 & -r & & & & & \\ & \ddots & \ddots & \ddots & & & & \\ & & -r & 2r+1 & -r & & & \\ & & & \ddots & \ddots & \ddots & & \\ & & & & -r & 2r+1 & -r & \\ & & & & & -1 & 1+h\sigma & \end{pmatrix},$$

$$\boldsymbol{u} = \begin{pmatrix} u_0^k \\ u_1^k \\ \vdots \\ u_j^k \\ \vdots \\ u_{j_{\max}-1}^k \\ u_{j_{\max}}^k \end{pmatrix}, \qquad \boldsymbol{B} = \begin{pmatrix} h\sigma U(t) \\ u_1^{k-1} \\ \vdots \\ u_j^{k-1} \\ \vdots \\ u_{j_{\max}-1}^{k-1} \\ h\sigma U(t) \end{pmatrix}.$$

$\boldsymbol{A}$ 是对角占优的，将矩阵 $\boldsymbol{A}$ 进行 LU 分解：

$$\boldsymbol{A} = \boldsymbol{LU},$$

使 $\boldsymbol{L}$ 为单位下二对角矩阵，$\boldsymbol{U}$ 为上二对角矩阵。则

$$\boldsymbol{LUu} = \boldsymbol{B}。$$

于是有

$$\begin{cases} \boldsymbol{Ly} = \boldsymbol{B}, \\ \boldsymbol{Uu} = \boldsymbol{y}. \end{cases}$$

$\boldsymbol{L}$ 与 $\boldsymbol{U}$ 特别容易求逆，而 $\boldsymbol{A}$ 的逆较为难求。[4]因此，使用追赶法求解能够减少运算次数。

3.参数确定

每个小温区和炉前、炉后区域的参数 D、σ 为待定参数，使用最小二乘法确定参数。

设附件情形下的中心温度在参数 $\boldsymbol{D}$、$\boldsymbol{\sigma}$ 下的估计值为 $\hat{u}^k$，附件所给实际值为 u^k，最小二乘指标

$$\min L=\sum_{k=1}^{k_{\max}}\left(\hat{u}^k-u^k\right)^2 。$$

此时各参数取值如表 5 所示，炉温曲线如图 4 所示。估计值与实际值的平均相对误差降至 0.50%。

表 5 一维热传导模型最优参数

区域	炉前区域	小温区 1~4	小温区 5	小温区 6	小温区 7	小温区 8	小温区 9	小温区 10~11	炉后区域
D （$\times10^{-3}$ mm²/s）	0.14	0.20	0.20	0.20	0.27	0.20	0.20	0.15	0.15
σ（s^{-1}）	0.012	0.021	0.024	0.030	0.034	0.025	0.025	0.010	0.010

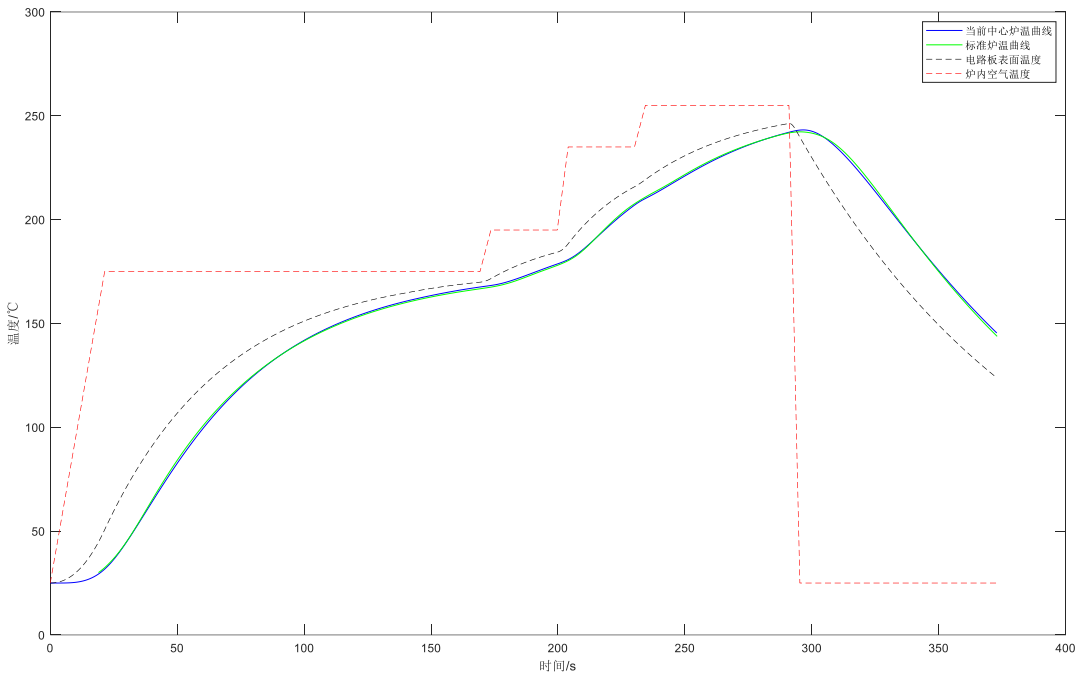

图 4 附件炉温曲线与所求曲线对照图

4.结果

改变模型温度与传送带过炉速度，完成问题一的求解。求解所得炉温曲线如图 5 所示，小温区 3、6、7 中点与小温区 8 结束处温度如表 6 所示。

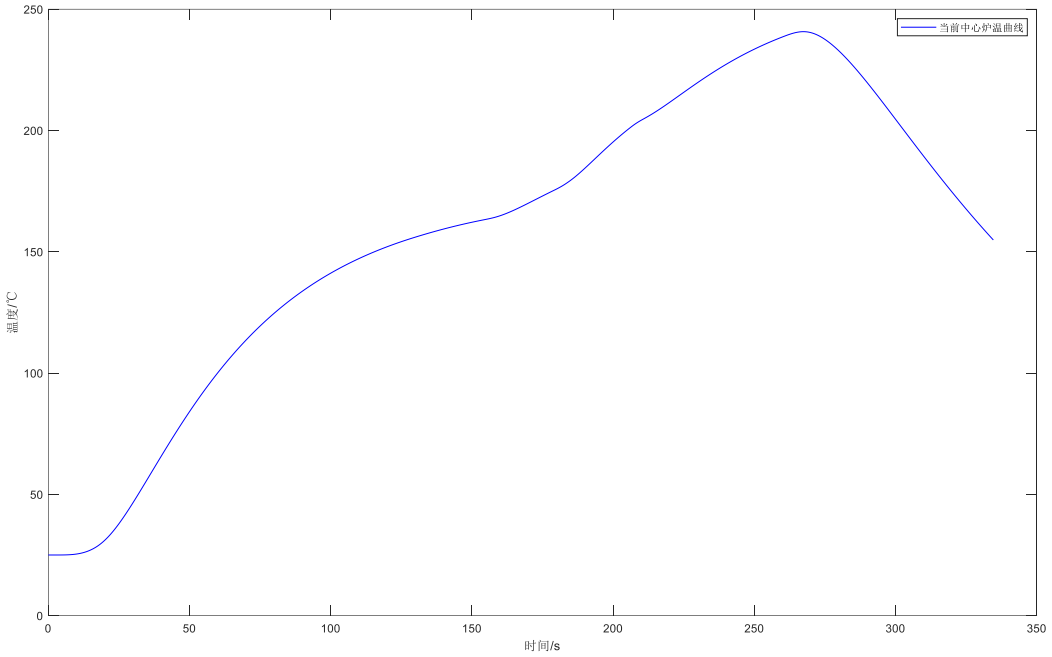

图 5 问题一炉温曲线

表 6 问题一相应位置中心温度

位置	小温区 3 中点	小温区 6 中点	小温区 7 终点	小温区 8 结束
温度	129.98 ℃	168.59 ℃	189.71 ℃	222.80 ℃

5.敏感性分析

调整过炉速度，分析过炉速度的改变对炉温曲线的影响，结果如图 6 所示。

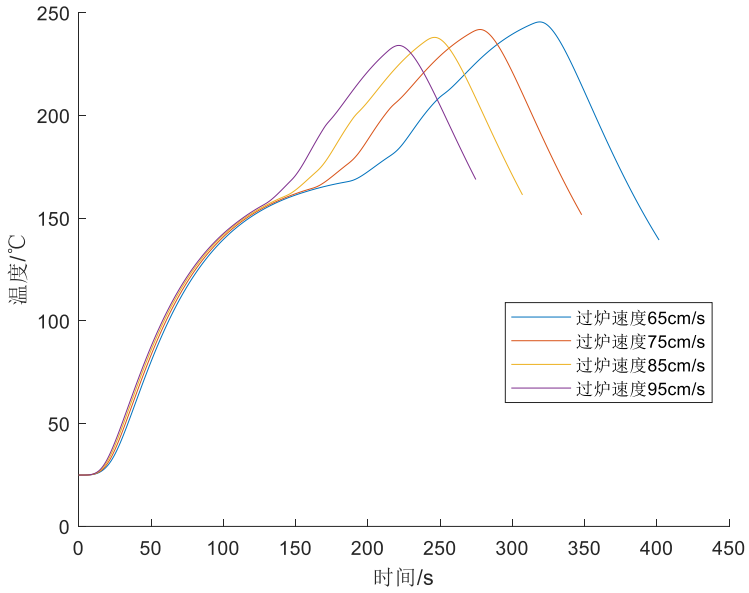

图 6 不同过炉速度下的炉温曲线

由图可见，锅炉速度的改变在小温区 6 之后才对炉温曲线产生显著影响，在温区 1~5 时对炉温曲线的影响较小。过炉速度的提升会使高温区域温度上升的速

率提升。

5.1.3 模型检验

针对问题一所给温度设定值，绘制全过程电路板内部温度变化图（图 7）。

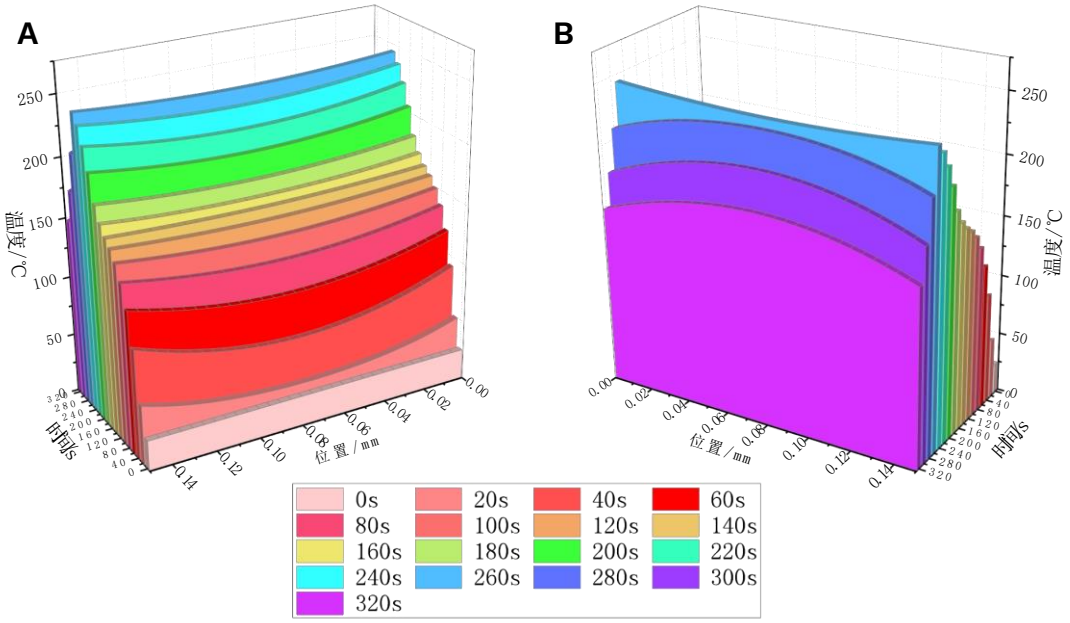

图 7 电路板内部各点温度随时间变化堆积图

由图可见，加热过程中（A），电路板上下两侧温度高于中心温度；冷却过程中（B），电路板表面温度低于中心温度。总体表现为中心温度的变化较表面滞后，且电路板内部温度关于中心轴对称。这与实际情况相符，模型合理。

5.2 问题二模型的建立与求解

5.2.1 单因素优化模型的建立

1.决策变量与目标函数

传送带过炉速度 v 同时是问题二模型的决策变量与目标函数：

$$\max v \text{ 。}$$

2.约束条件

为满足制程限制，模型存在以下约束条件。

①温度改变的速率应在-3~3 ℃/s 之内，即

$$-3 \le \frac{u^{k+1}-u^{k}}{\Delta t} \le 3 \text{ 。}$$

②温度上升过程中在 150 ℃到 190 ℃的时间应在 60~120 s 之内，即

$$60 \le t|_{u=190} - t|_{u=120} \le 120 。$$

③温度大于 217 ℃的时间应在 40~90 s 之内，即

$$40 \le \Delta t|_{u>217} \le 90 。$$

④峰值温度应在 240~250 ℃之内，即

$$240 \le u_{\max} \le 250 。$$

⑤过炉速度有定义域约束：

$$65 \le v \le 100 。$$

综上所述，建立单因素优化模型

$$\max v \tag{15}$$

$$\text{s.t.}\begin{cases} -3 \le \dfrac{u^{k+1} - u^{k}}{\Delta t} \le 3, \\ 60 \le \sum \left| t|_{u=190} - t|_{u=120} \right| \le 120, \\ 40 \le \Delta t|_{u>217} \le 90, \\ 240 \le u_{\max} \le 250, \\ 65 \le v \le 100, \\ u_t = D u_{xx}, \\ x = vt, \\ u(x,t)\big|_{t=0} = T_0, \\ \left.\dfrac{\partial u(x,t)}{\partial x}\right|_{x=0} = \sigma\left[u(x,t)|_{x=0} - U(t) \right], \\ \left.\dfrac{\partial u(x,t)}{\partial x}\right|_{x=d} = -\sigma\left[u(x,t)|_{x=d} - U(t) \right]. \end{cases}$$

5.2.2 单因素优化模型的求解

1.大步长搜索

取速度 v 的步长为 5 cm/s 进行初步搜索，搜索结果如表 7 所示.

表 7 问题二大步长搜索结果

速度（cm/s）	是否符合约束	速度（cm/s）	是否符合约束
65	否	85	否
70	是	90	否
75	是	95	否
80	是	100	否

2.基于二分查找的求解

观察表 7 发现，当速度大于某一值后将不符合约束条件。因此，在速度区间

$$v \in [80\ \text{cm/s}, 85\ \text{cm/s}]$$

进行二分查找以求得更为精确的解，具体步骤如下。

二分查找步骤

***step*1.** 速度左右边界初始化，值分别为 80、85。

***step*2.** 计算速度区间中点是否符合约束条件，若符合，则将中点做为左边界；若不符合，则将中点作为右边界

***step*3.** 循环 *step*2，直至速度区间长度小于 10^{-4} cm/s.

3.结果

查找得到的最大速度为 81.5691 cm/s，此时的炉温曲线如图 8 所示。

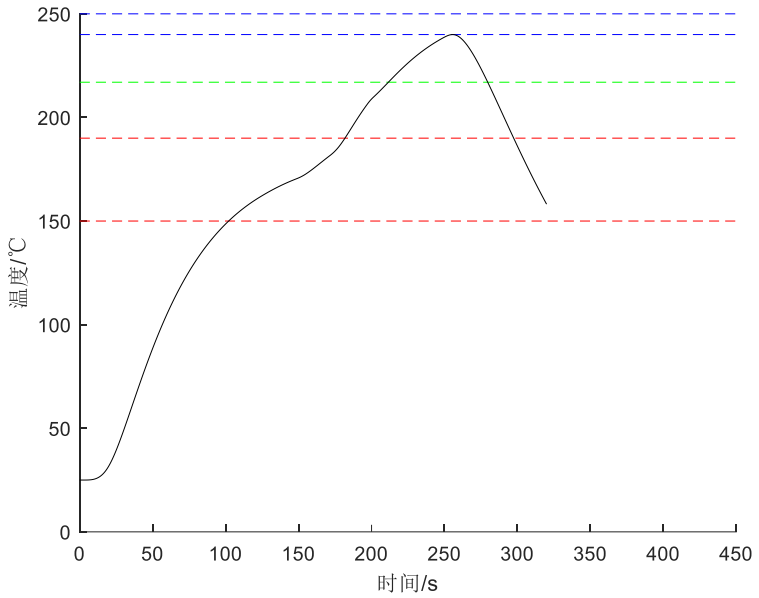

图 8 问题二最优炉温曲线

4.结果分析

针对不同速度分析不符合制程要求的原因，如图 9 所示。

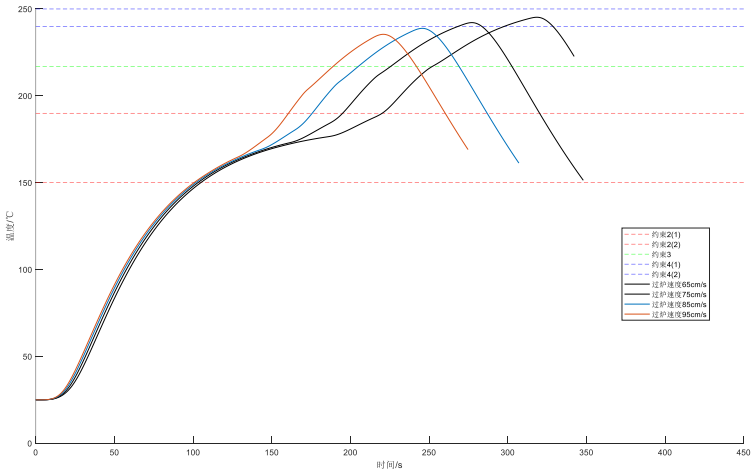

图 9 问题二结果分析图

观察发现：

（1）当**速度较高时**，峰值温度无法达到 240 ℃，无法满足约束条件④的下限；

（2）当**速度较低时**，温度大于 217 ℃的时间过长，温度在 150~190 ℃的时间较长，难以满足约束条件②、③的上限；

（3）在问题二设定的炉温下，约束条件①始终被满足，速度的提升与下降均教平稳。

5.3 问题三模型的建立与求解

5.3.1 多因素优化模型的建立

1.决策变量

各温区设定的温度与传送带过炉速度直接影响炉温曲线，是此模型的决策变量。按温区温度设定要求确定决策变量及其定义域，如表 8 所示。

表 8 问题三决策变量取值范围

决策变量	含义	定义域
T_1	小温区 1~5 的设定温度	[165 ℃, 185 ℃]
T_2	小温区 6 的设定温度	[185 ℃, 205 ℃]
T_3	小温区 7 的设定温度	[225 ℃, 245 ℃]
T_4	小温区 8~9 的设定温度	[245 ℃, 265 ℃]
v	传送带过炉速度	[60 cm/s, 100cm/s]

2.目标函数

炉温曲线中温度 217 ℃界限与峰值垂线形成图形的面积是本模型的目标函数。将面积离散化，采用梯形法求解数值积分，则目标函数

$$\min S=\sum_{k=k|_{u=217℃}}^{k|_{u=u_{\max}}}\left(\frac{u^k+u^{k+1}}{2}-217℃\right)\Delta t\ 。$$

3.约束条件

为满足制程要求，问题三的约束条件与问题二完全一致。

综上所述，建立多因素优化模型

$$\left(T_1,T_2,T_3,T_4,v\right)=\underset{T_1,T_2,T_3,T_4,v}{\arg\min}\,S=\sum_{k=k|_{u=217℃}}^{k|_{u=u_{\max}}}\left(\frac{u^k+u^{k+1}}{2}-217℃\right)\Delta t \tag{16}$$

$$
\text{s.t.}\begin{cases}
-3 \le \dfrac{u^{k+1}-u^{k}}{\Delta t} \le 3, \\
60 \le \sum \left| t|_{u=190} - t|_{u=120} \right| \le 120, \\
40 \le \Delta t|_{u>217} \le 90, \\
240 \le u_{\max} \le 250, \\
65 \le v \le 100, \\
u_t = Du_{xx}, \\
x = vt, \\
u(x,t)\big|_{t=0} = T_0, \\
\left.\dfrac{\partial u(x,t)}{\partial x}\right|_{x=0} = \sigma\left[u(x,t)\big|_{x=0} - U(t)\right], \\
\left.\dfrac{\partial u(x,t)}{\partial x}\right|_{x=d} = -\sigma\left[u(x,t)\big|_{x=d} - U(t)\right].
\end{cases} \tag{17}
$$

5.3.2 多因素优化模型的求解

1.基于遗传算法的求解

由于决策变量数量多，耦合关系复杂，因此使用遗传算法减少求解时间，具体过程如下。

遗传算法步骤

***step*1. 形成初始种群**

使用问题二所给温度参数，在[70 cm/s, 81.5 cm/s]内均匀设定传送带过炉速度，形成 100 个初始种群。由问题三可知如此形成的初始种群必定符合约束条件。

***step*2. 形成下一代**

视-*S* 为种群适应度指标，适应度大的种群有更大的概率进入下一代。

基于轮盘赌选择法挑选个体进入下一代。对每个进入下一代的个体基于概率进行随机交叉与变异。保留精英直接进入下一代。

***step*3. 迭代**

循环 *step*2 到设定的最大迭代次数，或目标函数所求面积稳定超 30 代。

本文采用自适应遗传算法进行求解，迭代过程中，交叉率与变异率随实际情况而改变。当每代精英的适应度趋于恒定时，为跳出局部最优，对交叉率和变异率做适当提升；当某代各种群适应度分布较分散时，适当减少交叉率和变异率。全过程交叉率与变异率将不高于 0.9。

2.结果

求解得到各温区温度设定值与传送带速度如表 9 所示，此时的面积为 *S*=420.1754 ℃·s。

表 9 问题三结果

小温区 1~5	小温区 6	小温区 7	小温区 8~9	传送带速度
176.2229 ℃	190.3380 ℃	230.9501 ℃	264.6106 ℃	90.7978 cm/s

3.结果分析

与问题二对照，绘制两问题炉内温度（图 11）与炉温曲线（图 10）对照图。

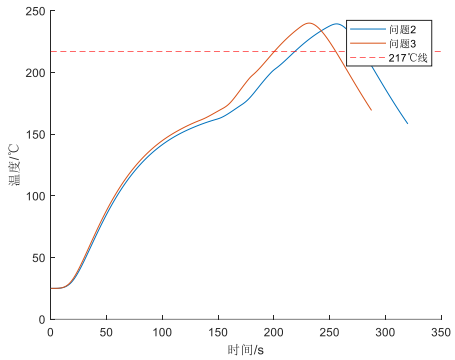

图 10 问题二、三炉温曲线对照图

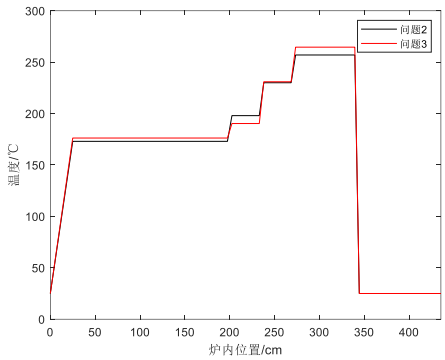

图 11 问题二、三炉内温度对照图

观察图 10 发现问题三所求面积较问题二有所减少。

观察图 11 发现问题三除小温区 6 外，温区设定温度均较问题二有所上升。为减少所求面积，问题三求得最优传送带过炉速度较问题二显著增加，为使电路板温度尽快上升，需要增加温区温度。

5. 4 问题四模型的建立与求解

5. 4. 1 多目标优化模型的建立

在问题三模型的基础上追加一个优化目标，决策变量与约束条件与问题三模型完全一致。

1.曲线对称程度的量化

以峰值温度为中心线，计此时温度为u^m。对左右两侧超过 217 ℃区域节点温度作差取绝对值后求平均：

$$E=\frac{\sum_k |u^{m+k}-u^{m-k}|}{N}。\tag{18}$$

2.模型给出

联立(16)、(17)、(18)，建立多目标优化模型

$$\min_{T_1,T_2,T_3,T_4,v} S=\sum_{k=k|_{u=217℃}}^{k|_{u=u_{\max}}}\left(\frac{u^k+u^{k+1}}{2}-217℃\right)\Delta t$$

$$\min_{T_1,T_2,T_3,T_4,v} E=\sum_k \left(u^{m+k}-u^{m-k}\right)^2$$

$$\text{s.t.}\begin{cases} -3\le \dfrac{u^{k+1}-u^k}{\Delta t}\le 3, \\ 60\le \sum\left|t\right|_{u=190}-t\left|_{u=120}\right|\le 120, \\ 40\le \Delta t\left|_{u>217}\right.\le 90, \\ 240\le u_{\max}\le 250, \\ 65\le v\le 100, \\ u_t=Du_{xx}, \\ x=vt, \\ u(x,t)\big|_{t=0}=T_0, \\ \left.\dfrac{\partial u(x,t)}{\partial x}\right|_{x=0}=\sigma\left[u(x,t)\big|_{x=0}-U(t)\right], \\ \left.\dfrac{\partial u(x,t)}{\partial x}\right|_{x=d}=-\sigma\left[u(x,t)\big|_{x=d}-U(t)\right]. \end{cases}$$

5.4.2 多目标优化模型的求解

1.基于加权合法的模型简化

对两目标进行加权求和。设目标 S 的权值为 α；则目标 E 的权值为$(1-\alpha)$。将多目标优化模型简化为但目标模型，目标函数为

$$\min_{T_1,T_2,T_3,T_4,v} F=\alpha S+(1-\alpha)E \text{ 。}$$

2.基因遗传算法的模型求解

与问题三一致，使用遗传算法进行求解，初始种群基于问题三结果产生。求解所得结果如表10所示，此时的面积S=421.2770 ℃s，对称程度指标E=2.2102 ℃。

表 10 问题四结果

小温区 1~5	小温区 6	小温区 7	小温区 8~9	传送带速度
184.1790 ℃	196.1701 ℃	237.5578 ℃	264.6693 ℃	97.6667 cm/s

3.结果分析

由于增加了一个目标，使所求面积较问题三有所增加，符合实际情况。

与问题三对照，绘制两问题炉内温度（图 13）与炉温曲线（图 12）对照图。

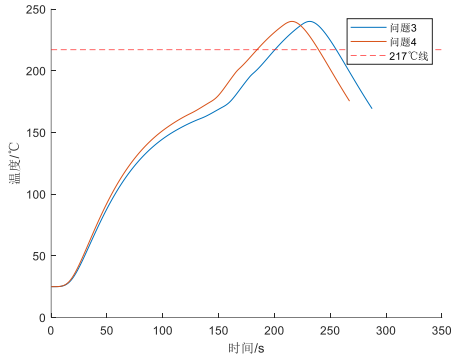

图 12 问题三、四炉温曲线对照图

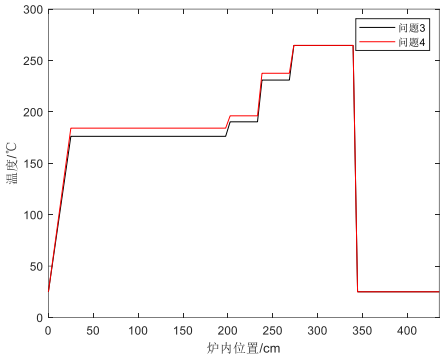

图 13 问题三、四炉内温度对照图

由图 12 可见，问题四所得炉温曲线较问题三对称。传送带过炉速度较问题三有所增加，小温区 7 及前序温区温度有所提升。

六、模型的评价

6.1 模型的优点

1. 通过对偏微分方程进行有限差分化，求出符合精度要求的数值解，简化了求解过程；
2. 通过二分法求解问题二的模型，提高的解的精度；
3. 通过遗传算法对问题三、四模型进行求解，并利用加权和法将多目标问题简化为单目标。

6.2 模型的不足

1. 使用智能算法求解容易使结果陷入局部最优；
2. 模型忽略了热量自侧向的流动。

6.3 模型的推广

模型适用性广，经简单调整即可处理更多温区的温度控制系统优化设计，可将其应用于诸如中央厨房食品产线的温控等领域。

参考文献

[1] 孙昆淼. 数学物理方法[M]. 4 版. 北京: 高等教育出版社, 2010: 115.

[2] 姚端正, 周国全, 贾俊基. 数学物理方法[M]. 4 版. 北京: 科学出版社, 2020: 108.

[3] 张文生. 微分方程数值解: 有限差分理论方法与数值计算[M]. 北京: 科学出版社, 2015: 216.

[4] 戈林鲍姆, 夏蒂埃. 数值方法: 设计、分析和算法实现[M]. 吴兆京, 王国英, 范红军, 译. 北京: 机械工业出版社, 2016: 104.

附　录

附录说明	
附录 1	支撑材料清单
附录 2	源程序代码

附录 1　支撑材料清单

支撑材料清单		
问题一	Que1_1.m	参数估计
	Que1_2.m	问题求解
	Que1_3.m	结果分析
	result.csv	结果
问题二	Que2_1.m	大步长搜索
	Que2_2.m	二分查找
	Que2_3.m	结果分析
问题三	Q3.m	GA 主函数
	Que3_2.m	结果分析
问题四	Q4.m	GA 主函数

支撑材料说明

- 拓展名为.m 的文件为 MATLAB 程序文件，本文的运行环境是 MATLAB R2022b；
- 拓展名为.csv 文件为结果文件，以 ANSI 格式编码。

附录 2　源程序代码

Que1_1.m

参数估计

```
% 问题 1 参数估计
%% 初始化
clear
clc
close all
%% 设置炉温
global U_info
U_info=[ones(1,5)*175,195,235,ones(1,2)*255,ones(1,2)*25];
% 炉前,1-4,5,6,7,8,9,10-11,炉后
D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
```

```
sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
len=30.5;
gap=5;
global x0 y0
x0=[0,25];
y0=[25];
D_ls=[D_info(1)];
sigma_ls=[sigma_info(1)];
for i=1:11
    x0(end+1)=x0(end)+len;
    x0(end+1)=x0(end)+gap;
    y0(end+1)=U_info(i);
    y0(end+1)=U_info(i);
    D_ls(end+1)=D_info(i+1);
    D_ls(end+1)=D_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
end
x0(end)=435.5;
y0(end+1)=25;
D_ls(end+1)=D_ls(end);
sigma_ls(end+1)=sigma_info(end);

%% 绘制炉内温度图
% x_fig=0:0.2:435.5;
% y_fig=[];
% D_fig=[];
% sigma_fig=[];
% for x=x_fig
%     y_fig(end+1)=get_U(x);
%     D_fig(end+1)=get_D(x,D_ls);
%     sigma_fig(end+1)=get_sigma(x,sigma_ls);
% end
% plot(x_fig,y_fig,"k")
% xlabel("炉内位置/cm")
% ylabel("温度/℃")
% xlim([0,435.5])
% figure
% plot(x_fig,D_fig,"b")
% figure
% plot(x_fig,sigma_fig,"g")

%% 导入数据
```

```
aim=xlsread("附件.xlsx");

%% 求解炉温曲线
d=0.15;     % 厚度(mm)
j_step=0.005;   % 厚度步长(mm)
k_step=0.005;     % 时间步长(s)
v=70;   % 传送带前进速度(cm/min)
vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     % (k,j)->(时
间 s,空间 cm);首行/首列为零时刻/位置
[k_max,j_max]=size(data);
data(1,:)=25;   % 初始条件
for k=2:k_max
    t_now=k*k_step; % 当前时间(s)
    DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
    sigma=get_sigma(t_now*vv,sigma_ls);
    for j=2:j_max-1
        data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-1))+data(k-
1,j);
    end
    data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
    data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-get_U(t_now*vv))+data(k-
1,j_max);
end
%% 计算误差平方和
err=sum(abs(aim(1:end-1,2)-
data(aim(1,1)/k_step:0.5/k_step:min(aim(end,1)/k_step,k_max),d/2/j_step))
./aim(1:end-1,2));
errmean=err/(length(aim)-1);
[X,Y]=meshgrid(0:k_step:435/(v/60),0:j_step:d);
mesh(X,Y,data')
xlabel("时间/s")
ylabel("深度/mm")
zlabel("温度/℃")
colorbar()
figure
plot(0:k_step:435/(v/60),data(:,d/2/j_step),"b",aim(:,1),aim(:,2),"g",0:k
_step:435/(v/60),data(:,1),"k--
",0:k_step:435/(v/60),get_U([0:k_step:435/(v/60)].*vv),"r--")
legend("当前中心炉温曲线","标准炉温曲线","电路板表面温度","炉内空气温度")
xlabel("时间/s")
ylabel("温度/℃")
% figure
% plot(0:j_step:d,data(400,:))
```

```
function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    sigma=interp1(x0,sigma_ls,x);
end
```

Que1_2.m

问题求解

```
% 问题 1 求解
%% 初始化
clear
clc
close all
%% 设置炉温
global U_info
U_info=[ones(1,5)*173,198,230,ones(1,2)*257,ones(1,2)*25];
% 炉前,1-4,5,6,7,8,9,10-11,炉后
D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
len=30.5;
gap=5;
global x0 y0
x0=[0,25];
y0=[25];
D_ls=[D_info(1)];
sigma_ls=[sigma_info(1)];
for i=1:11
    x0(end+1)=x0(end)+len;
    x0(end+1)=x0(end)+gap;
```

```
    y0(end+1)=U_info(i);
    y0(end+1)=U_info(i);
    D_ls(end+1)=D_info(i+1);
    D_ls(end+1)=D_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
end
x0(end)=435.5;
y0(end+1)=25;
D_ls(end+1)=D_ls(end);
sigma_ls(end+1)=sigma_info(end);

%% 绘制炉内温度图
% x_fig=0:0.2:435.5;
% y_fig=[];
% D_fig=[];
% sigma_fig=[];
% for x=x_fig
%     y_fig(end+1)=get_U(x);
%     D_fig(end+1)=get_D(x,D_ls);
%     sigma_fig(end+1)=get_sigma(x,sigma_ls);
% end
% plot(x_fig,y_fig,"k")
% xlabel("炉内位置/cm")
% ylabel("温度/℃")
% xlim([0,435.5])
% figure
% plot(x_fig,D_fig,"b")
% figure
% plot(x_fig,sigma_fig,"g")

d=0.15;     % 厚度(mm)
j_step=0.005;   % 厚度步长(mm)
k_step=0.005;     % 时间步长(s)
v=78;   % 传送带前进速度(cm/min)
vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     % (k,j)->(时
间 s,空间 cm);首行/首列为零时刻/位置
[k_max,j_max]=size(data);
data(1,:)=25;   % 初始条件
for k=2:k_max
    t_now=k*k_step; % 当前时间(s)
    DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
    sigma=get_sigma(t_now*vv,sigma_ls);
```

```
    for j=2:j_max-1
        data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-1))+data(k-
1,j);
    end
    data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
    data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-get_U(t_now*vv))+data(k-
1,j_max);
    if any(abs(t_now*vv-[111.25,217.75,253.25,304])<1e-2)
        fprintf("位置%.3f,温度%.6f\n",t_now*vv,data(k,d/2/j_step))
    end
end
[X,Y]=meshgrid(0:k_step:435/(v/60),0:j_step:d);
mesh(X,Y,data')
xlabel("时间/s")
ylabel("深度/mm")
zlabel("温度/℃")
colorbar()
figure
plot(0:k_step:435/(v/60),data(:,d/2/j_step),"b")
legend("当前中心炉温曲线")
xlabel("时间/s")
ylabel("温度/℃")
% figure
% plot(0:j_step:d,data(400,:))
% data(1:0.5/k_step:end,d/2/j_step)
% res=data(1:20/k_step:end,:)

function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    sigma=interp1(x0,sigma_ls,x);
end
```

Que1_3.m

结果分析

```
% 问题 1 结果分析
%% 初始化
clear
clc
close all
%% 设置炉温
global U_info
U_info=[ones(1,5)*173,198,230,ones(1,2)*257,ones(1,2)*25];
% 炉前,1-4,5,6,7,8,9,10-11,炉后
D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
len=30.5;
gap=5;
global x0 y0
x0=[0,25];
y0=[25];
D_ls=[D_info(1)];
sigma_ls=[sigma_info(1)];
for i=1:11
    x0(end+1)=x0(end)+len;
    x0(end+1)=x0(end)+gap;
    y0(end+1)=U_info(i);
    y0(end+1)=U_info(i);
    D_ls(end+1)=D_info(i+1);
    D_ls(end+1)=D_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
end
x0(end)=435.5;
y0(end+1)=25;
D_ls(end+1)=D_ls(end);
sigma_ls(end+1)=sigma_info(end);

d=0.15;     % 厚度(mm)
j_step=0.005;   % 厚度步长(mm)
k_step=0.005;     % 时间步长(s)
figure
hold on
for v=65:10:100
```

```
%     v=78;   % 传送带前进速度(cm/min)
    vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
    data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     %
(k,j)->(时间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
    end
    plot(0:k_step:435/(v/60),data(:,d/2/j_step))
end
legend("过炉速度"+[65:10:100]+"cm/s")
xlabel("时间/s")
ylabel("温度/℃")


function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    sigma=interp1(x0,sigma_ls,x);
end
```

Que2_1.m

大步长搜索

```
% 问题 2 大步长搜索
%% 初始化
clear
clc
close all
%% 设置炉温
global U_info
U_info=[ones(1,5)*182,203,237,ones(1,2)*254,ones(1,2)*25];
% 炉前,1-4,5,6,7,8,9,10-11,炉后
D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
len=30.5;
gap=5;
global x0 y0 D_ls sigma_ls
x0=[0,25];
y0=[25];
D_ls=[D_info(1)];
sigma_ls=[sigma_info(1)];
for i=1:11
    x0(end+1)=x0(end)+len;
    x0(end+1)=x0(end)+gap;
    y0(end+1)=U_info(i);
    y0(end+1)=U_info(i);
    D_ls(end+1)=D_info(i+1);
    D_ls(end+1)=D_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
end
x0(end)=435.5;
y0(end+1)=25;
D_ls(end+1)=D_ls(end);
sigma_ls(end+1)=sigma_info(end);

for v=65:5:100
    res(v)=constraint(v);
end

function res=constraint(v)
% 约束条件
% 输入速度 v(cm/s);输出 1-符合约束,0-不符合约束
    global D_ls sigma_ls
```

```
    d=0.15;     % 厚度(mm)
    j_step=0.005;   % 厚度步长(mm)
    k_step=0.005;     % 时间步长(s)
    vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
    data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     %
(k,j)->(时间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    key=ones(k_max,1)*25; % 中心温度记录
    count1=0;   % 约束 2 计时器
    count2=0;   % 约束 3 计时器
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
        key(k)=data(k,d/2/j_step);

        if key(k)-key(k-1)>3*k_step || key(k)-key(k-1)<-3*k_step    % 约束
(1)
            res=0;
            fprintf("v=%f,约束(1)跳出\n",v)
            return
        end
        if key(k)-key(k-1)>0 && key(k)>150 && key(k)<190
            count1=count1+1;
        elseif key(k)>217
            count2=count2+1;
        end
        if count1*k_step>120 || count2*k_step>90 || key(k)>250   % 约束
(2)(3)(4)上界
%             plt(k,key,t_now,k_step)
            fprintf("v=%f,约束(234)上界跳出\n",v)
            res=0;
            return
        end
    end
```

```
    if count1*k_step<60 || count2*k_step<40 || max(key)<240 % 约束
(2)(3)(4)下界
        fprintf("v=%f,约束(234)下界跳出\n",v)
        res=0;
        return
    end
    fprintf("v=%f,符合条件\n",v)
    res=1;
end

function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    sigma=interp1(x0,sigma_ls,x);
end

function plt(k,key,t_now,k_step)
% 绘图
    figure
    hold on
    plot([0,t_now-k_step],[150,150],"r--", ...
        [0,t_now-k_step],[190,190],"r--", ...
        [0,t_now-k_step],[217,217],"g--", ...
        [0,t_now-k_step],[240,240],"b--", ...
        [0,t_now-k_step],[250,250],"b--")
    plot(0:k_step:t_now-k_step,key(1:k),"k")
end
```

Que2_2.m
二分查找
% 问题 2 二分求解 %% 初始化

```
clear
clc
close all
%% 设置炉温
global U_info
U_info=[ones(1,5)*182,203,237,ones(1,2)*254,ones(1,2)*25];
% 炉前,1-4,5,6,7,8,9,10-11,炉后
D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
len=30.5;
gap=5;
global x0 y0 D_ls sigma_ls
x0=[0,25];
y0=[25];
D_ls=[D_info(1)];
sigma_ls=[sigma_info(1)];
for i=1:11
    x0(end+1)=x0(end)+len;
    x0(end+1)=x0(end)+gap;
    y0(end+1)=U_info(i);
    y0(end+1)=U_info(i);
    D_ls(end+1)=D_info(i+1);
    D_ls(end+1)=D_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
end
x0(end)=435.5;
y0(end+1)=25;
D_ls(end+1)=D_ls(end);
sigma_ls(end+1)=sigma_info(end);


%% 二分
left=80;
right=85;
while right-left>1e-4
    mid=(right+left)/2;
    if constraint(mid)
        left=mid;  % 符合约束，更新左界
    else
        right=mid;   % 不符合约束，更新右界
    end
```

```
end
disp(left)

function res=constraint(v)
% 约束条件
% 输入速度 v(cm/s);输出 1-符合约束,0-不符合约束
    global D_ls sigma_ls
    d=0.15;     % 厚度(mm)
    j_step=0.005;   % 厚度步长(mm)
    k_step=0.005;     % 时间步长(s)
    vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
    data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     %
(k,j)->(时间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    key=ones(k_max,1)*25; % 中心温度记录
    count1=0;   % 约束 2 计时器
    count2=0;   % 约束 3 计时器
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
        key(k)=data(k,d/2/j_step);

        if key(k)-key(k-1)>3*k_step || key(k)-key(k-1)<-3*k_step    % 约束
(1)
            res=0;
            fprintf("v=%f,约束(1)跳出\n",v)
            return
        end
        if key(k)-key(k-1)>0 && key(k)>150 && key(k)<190
            count1=count1+1;
        elseif key(k)>217
            count2=count2+1;
        end
        if count1*k_step>120 || count2*k_step>90 || key(k)>250   % 约束
(2)(3)(4)上界
```

```
%             plt(k,key,t_now,k_step)
            fprintf("v=%f,约束(234)上界跳出\n",v)
            res=0;
            return
        end
    end
    if count1*k_step<60 || count2*k_step<40 || max(key)<240 % 约束
(2)(3)(4)下界
        fprintf("v=%f,约束(234)下界跳出\n",v)
        res=0;
        return
    end
    fprintf("v=%f,符合条件\n",v)
    res=1;
end

function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    sigma=interp1(x0,sigma_ls,x);
end

function plt(k,key,t_now,k_step)
% 绘图
    figure
    hold on
    plot([0,t_now-k_step],[150,150],"r--", ...
        [0,t_now-k_step],[190,190],"r--", ...
        [0,t_now-k_step],[217,217],"g--", ...
        [0,t_now-k_step],[240,240],"b--", ...
        [0,t_now-k_step],[250,250],"b--")
    plot(0:k_step:t_now-k_step,key(1:k),"k")
```

```
end
```

Que2_3.m

结果分析

```
% 问题 2 结果分析
%% 初始化
clear
clc
close all
%% 设置炉温
global U_info
U_info=[ones(1,5)*182,203,237,ones(1,2)*254,ones(1,2)*25];
% 炉前,1-4,5,6,7,8,9,10-11,炉后
D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
len=30.5;
gap=5;
global x0 y0 D_ls sigma_ls
x0=[0,25];
y0=[25];
D_ls=[D_info(1)];
sigma_ls=[sigma_info(1)];
for i=1:11
    x0(end+1)=x0(end)+len;
    x0(end+1)=x0(end)+gap;
    y0(end+1)=U_info(i);
    y0(end+1)=U_info(i);
    D_ls(end+1)=D_info(i+1);
    D_ls(end+1)=D_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
    sigma_ls(end+1)=sigma_info(i+1);
end
x0(end)=435.5;
y0(end+1)=25;
D_ls(end+1)=D_ls(end);
sigma_ls(end+1)=sigma_info(end);

plt_init()
for v=81.5691 %65:10:100
    res(v)=constraint(v);
end
```

```
legend(["约束 2(1)","约束 2(2)","约束 3","约束 4(1)","约束 4(2)","过炉速度
"+[65:10:100]+"cm/s"])
xlabel("时间/s")
ylabel("温度/℃")

function res=constraint(v)
% 约束条件
% 输入速度 v(cm/s);输出 1-符合约束,0-不符合约束
    global D_ls sigma_ls
    d=0.15;     % 厚度(mm)
    j_step=0.005;   % 厚度步长(mm)
    k_step=0.005;     % 时间步长(s)
    vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
    data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     %
(k,j)->(时间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    key=ones(k_max,1)*25; % 中心温度记录
    count1=0;   % 约束 2 计时器
    count2=0;   % 约束 3 计时器
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
        key(k)=data(k,d/2/j_step);

        if key(k)-key(k-1)>3*k_step || key(k)-key(k-1)<-3*k_step    % 约束
(1)
            res=0;
            fprintf("v=%f,约束(1)跳出\n",v)
            return
        end
        if key(k)-key(k-1)>0 && key(k)>150 && key(k)<190
            count1=count1+1;
        elseif key(k)>217
            count2=count2+1;
        end
```

```
        if count1*k_step>120 || count2*k_step>90 || key(k)>250   % 约束
(2)(3)(4)上界
%             plt(k,key,t_now,k_step)
            plot(0:k_step:t_now-k_step,key(1:k),"k")
            fprintf("v=%f,约束(234)上界跳出\n",v)
            res=0;
            return
        end
    end
    if count1*k_step<60 || count2*k_step<40 || max(key)<240 % 约束
(2)(3)(4)下界
        plot(0:k_step:t_now-k_step,key(1:k))
        fprintf("v=%f,约束(234)下界跳出\n",v)
        res=0;
        return
    end
    fprintf("v=%f,符合条件\n",v)
    plot(0:k_step:t_now-k_step,key(1:k),"k")
    res=1;
end

function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    sigma=interp1(x0,sigma_ls,x);
end

function plt_init()
% 绘图
    figure
    hold on
    plot([0,450],[150,150],"r--", ...
```

```
        [0,450],[190,190],"r--", ...
        [0,450],[217,217],"g--", ...
        [0,450],[240,240],"b--", ...
        [0,450],[250,250],"b--")
end
```

Q3.m

GA 主函数

```
%% 初始化
clear
close all
clc

%% GA 参数赋值
NP=50; % 种群数量
G=100;  % 最大遗传次数
L=20;   % 单维二进制数段长度
Dim=5;  % 维度
Max=[185,205,245,265,100];  % 定义域上限
Min=[165,185,225,245,60];    % 定义域下限
max_sta=inf;     % 最大稳定代数，稳定超过此代数将结束算法
max_err=30;     % 最大不符合约束条件次数

%% GA 开始
f=zeros(NP,L*Dim);    % 初始种群
for i=1:NP
    f(i,:)=d2b([182,203,237,254,81.5-(i-1)/10],Min,Max,L);
end
max_fit_rec=zeros(1,G);

for k=1:G
    tic
    %% 开始迭代
    for i=1:NP
        U=f(i,:);
        x(i,:)=b2d(U,Min,Max,L);
        Fit(i)=aim(x(i,:));
    end
    [max_fit,max_fit_idx]=max(Fit);
    f_best=f(max_fit_idx,:);
    Fit1=(Fit-min(Fit))./(max_fit-min(Fit));

    %% 计算稳定代数
    for idx=k-1:-1:1
```

```
        if max_fit_rec(idx)~=max_fit
            break
        end
    end
    sta_num=k-idx;  % 稳定代数

    %% 判断是否稳定超 max_sta 代
    if sta_num>=max_sta
        break
    end

    %% 基于轮盘赌选择法的复制操作
    sum_Fit=sum(Fit1);
    fitvalue=Fit1/sum_Fit;
    fitvalue=(cumsum(fitvalue));
    ms=sort(rand(NP,1));
    fitidx=1;
    newidx=1;
    while newidx<=NP-1
        if ms(newidx) < fitvalue(fitidx)
            nf(newidx,:)=f(fitidx,:);   % 复制
            newidx=newidx+1;
        else
            fitidx=fitidx+1;
            if fitidx>NP
                break;
            end
        end
    end

    %% 基于概率的交叉操作
    for i=1:NP-2
        p=rand;
        if p<get_pc(sta_num)
            t=1;
            for trytime=1:max_err
                tnf=zeros(1,L*Dim);
                q=randi([0,1],1,L*Dim); % 多点交叉
                for j=1:L*Dim
                    tnf(j)=nf(i+q(j),j);
                end
                if judge(b2d(tnf,Min,Max,L))
                    nf(i,:)=tnf(:);
                    break
```

```
                end
            end
        end
    end

    %% 基于概率的变异操作
    for i=1:round((NP-1)*get_pm(sta_num))
        h=randi([1,NP-1],1,1);
        t=1;
        for trytime=1:max_err
            tnf=nf(h,:);
            for j=1:round(L*Dim*get_pm(sta_num))
                g=randi([1,L*Dim],1,1);
                tnf(g)=~tnf(g); % 变异
            end
            if judge(b2d(tnf,Min,Max,L))
                nf(i,:)=tnf(:);
                break
            end
        end
    end

    %% 下一代预备
    f=nf;
    f(NP,:)=f_best; % 保留精英
    max_fit_rec(k)=max_fit;
    usetime=toc;
    fprintf("完成计算第%d 代，最大适应度%.2f，进度%.6f，用时%.2fs，预计还
需%.3fmin\n",k,max_fit,k/G,usetime,usetime*(G-k)/60)
end

%% 收尾
for i=1:NP
    Fit(i)=aim(b2d(f(i,:),Min,Max,L));  % 计算每个样本适应度
end
[max_fit,max_fit_idx]=max(Fit);

%% 显示结果
fprintf("最大值:")
disp(max_fit)
fprintf("最大值点:")
disp(x(max_fit_idx,:))
fprintf('精度：x±')
disp((Max-Min)/(2^L-1));
```

```
function fit=aim(x)
    % 目标函数
    %% 设置炉温
    U_info=[ones(1,5)*x(1),x(2),x(3),ones(1,2)*x(4),ones(1,2)*25];
    % 炉前,1-4,5,6,7,8,9,10-11,炉后
    D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
    sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
    len=30.5;
    gap=5;
    global x0 y0
    x0=[0,25];
    y0=[25];
    D_ls=[D_info(1)];
    sigma_ls=[sigma_info(1)];
    for i=1:11
        x0(end+1)=x0(end)+len;
        x0(end+1)=x0(end)+gap;
        y0(end+1)=U_info(i);
        y0(end+1)=U_info(i);
        D_ls(end+1)=D_info(i+1);
        D_ls(end+1)=D_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
    end
    x0(end)=435.5;
    y0(end+1)=25;
    D_ls(end+1)=D_ls(end);
    sigma_ls(end+1)=sigma_info(end);
    %% 计算
    v=x(5);
    d=0.15;     % 厚度(mm)
    j_step=0.005;   % 厚度步长(mm)
    k_step=0.005;     % 时间步长(s)
    vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
    data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     %
(k,j)->(时间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    key=ones(k_max,1)*25; % 中心温度记录
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
```

```
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
        key(k)=data(k,d/2/j_step);
        if key(k)-key(k-1)<0    % 一下降就停止迭代
            break
        end
    end
    [~,k_idx_umax]=max(key);
    fit=0;
    for k=find(key>217)'
        if k>k_idx_umax
            break
        end
        fit=fit+((key(k)+key(k+1))/2-217)*k_step;
    end
    fit=-fit;
end

function res=judge(x)
    % 约束条件
    % 输入速度 v(cm/s);输出 1-符合约束,0-不符合约束
    %% 设置炉温
    U_info=[ones(1,5)*x(1),x(2),x(3),ones(1,2)*x(4),ones(1,2)*25];
    % 炉前,1-4,5,6,7,8,9,10-11,炉后
    D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
    sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
    len=30.5;
    gap=5;
    global x0 y0
    x0=[0,25];
    y0=[25];
    D_ls=[D_info(1)];
    sigma_ls=[sigma_info(1)];
    for i=1:11
        x0(end+1)=x0(end)+len;
```

```
        x0(end+1)=x0(end)+gap;
        y0(end+1)=U_info(i);
        y0(end+1)=U_info(i);
        D_ls(end+1)=D_info(i+1);
        D_ls(end+1)=D_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
    end
    x0(end)=435.5;
    y0(end+1)=25;
    D_ls(end+1)=D_ls(end);
    sigma_ls(end+1)=sigma_info(end);
    %% 约束
    v=x(5);
    d=0.15;     % 厚度(mm)
    j_step=0.005;   % 厚度步长(mm)
    k_step=0.005;     % 时间步长(s)
    vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
    data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     %
(k,j)->(时间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    key=ones(k_max,1)*25; % 中心温度记录
    count1=0;   % 约束 2 计时器
    count2=0;   % 约束 3 计时器
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
        key(k)=data(k,d/2/j_step);
        if key(k)-key(k-1)>3*k_step || key(k)-key(k-1)<-3*k_step    % 约束
(1)
            res=0;
%             fprintf("v=%f,约束(1)跳出\n",v)
            return
        end
        if key(k)-key(k-1)>0 && key(k)>150 && key(k)<190
```

```
            count1=count1+1;
        elseif key(k)>217
            count2=count2+1;
        end
        if count1*k_step>120 || count2*k_step>90 || key(k)>250   % 约束
(2)(3)(4)上界
%             fprintf("v=%f,约束(234)上界跳出\n",v)
            res=0;
            return
        end
        if key(k)-key(k-1)<0 && key(k)<217  % 下降，减少递推次数
            break
        end
    end
    if count1*k_step<60 || count2*k_step<40 || max(key)<240 % 约束
(2)(3)(4)下界
%         fprintf("v=%f,约束(234)下界跳出\n",v)
        res=0;
        return
    end
%     fprintf("v=%f,符合条件\n",v)
    res=1;
end

function pc=get_pc(sta_num)
    % 交叉率，sta_num-稳定代数
    pc=min(0.2+sta_num*0.05,0.9);   % 防止 pc 超过 0.9
end

function pm=get_pm(sta_num)
    % 变异率，sta_num-稳定代数
    pm=min(0.2+sta_num*0.05,0.9);   % 防止 pm 超过 0.9
end

function d=b2d(str,Min,Max,L)
    %格雷解码并将二进制转为定义域内的十进制
    %b2d(str,Xx,Xs,L),str-二进制行向量，Min-下限，Max-上限，L 单维度二进制串长
度
    Dim=length(Min);
    n=zeros(Dim,L);
    for k=1:Dim
        n(k,1)=str((k-1)*L+1);  % 格雷编码解码
        for j=2:L
            n(k,j)=xor(n(k,j-1),str((k-1)*L+j));    % xor—异或运算
```

```
        end
    end
    for di=1:Dim
        m(di)=0;
        for j=1:L   % 二进制转十进制
            m(di)=m(di)+n(di,L-j+1)*(2^(j-1));
        end
    end
    d=Min+m.*(Max-Min)./(2^L-1);    % 十进制解码至定义域
end

function str = d2b (arr,Min,Max,L)
    %格雷编码
    %d2b(arr,Xx,Xs,L),arr-十进制数据数组（行向量），Min-下限，Max-上限，L 单维度二进制串长度；输出 str 为二进制行向量
    Dim=length(Min);
    str=zeros(1,Dim*L);
    m=round((arr-Min).*(2^L-1)./(Max-Min)); % 分散至二进制定义域
    n=double(boolean(dec2bin(m')-'0'));   % 十进制转二进制
    [~,col]=size(n);
    while col<L     % 首位补 0 至长为 L
        n=[zeros(length(arr),1),n];
        [~,col]=size(n);
    end
    for k=1:Dim % 格雷编码
        str((k-1)*L+1)=n(k,1);  % 首位不变
        str((k-1)*L+2:k*L)=xor(n(k,1:end-1),n(k,2:end));    % 异或运算
    end
end

function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
```

```
    global x0
    sigma=interp1(x0,sigma_ls,x);
end
```

Que3_2.m

结果分析

```
% 问题 3 与问题 2 对照
%% 初始化
clear
clc
close all

%% 设置炉温
U_info2=[ones(1,5)*173,198,230,ones(1,2)*257,ones(1,2)*25];
U_info3=[ones(1,5)*176.2229,190.3380,230.9501,ones(1,2)*264.6106,ones(1,2
)*25];
len=30.5;
gap=5;
x0=[0,25];
y2=[25];
y3=[25];
for i=1:11
    x0(end+1)=x0(end)+len;
    x0(end+1)=x0(end)+gap;
    y2(end+1)=U_info2(i);
    y2(end+1)=U_info2(i);
    y3(end+1)=U_info3(i);
    y3(end+1)=U_info3(i);
end
x0(end)=435.5;
y2(end+1)=25;
y3(end+1)=25;

%% 绘制炉内温度图
x_fig=0:0.5:435.5;
y2_fig=[];
y3_fig=[];
for x=x_fig
    y2_fig(end+1)=interp1(x0,y2,x);
    y3_fig(end+1)=interp1(x0,y3,x);
end
plot(x_fig,y2_fig,"k",x_fig,y3_fig,"r")
xlabel("炉内位置/cm")
ylabel("温度/℃")
```

```
xlim([0,435.5])
legend("问题 2","问题 3")

% 问题 3 炉温曲线绘图
%% 初始化
figure
hold on
U_ls=[U_info2;U_info3];
for idx=[1,2]
    %% 设置炉温
    global U_info
    U_info=U_ls(idx,:);
    % 炉前,1-4,5,6,7,8,9,10-11,炉后
    D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
    sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
    len=30.5;
    gap=5;
    global x0 y0
    x0=[0,25];
    y0=[25];
    D_ls=[D_info(1)];
    sigma_ls=[sigma_info(1)];
    for i=1:11
        x0(end+1)=x0(end)+len;
        x0(end+1)=x0(end)+gap;
        y0(end+1)=U_info(i);
        y0(end+1)=U_info(i);
        D_ls(end+1)=D_info(i+1);
        D_ls(end+1)=D_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
    end
    x0(end)=435.5;
    y0(end+1)=25;
    D_ls(end+1)=D_ls(end);
    sigma_ls(end+1)=sigma_info(end);

    d=0.15;     % 厚度(mm)
    j_step=0.005;   % 厚度步长(mm)
    k_step=0.005;     % 时间步长(s)
    v=[81.5691,90.7978];   % 传送带前进速度(cm/min)
    vv=v(idx)/60;    % 传送带前进速度(cm/s)[不要修改参数]
```

```
    data=zeros(floor(435/vv/k_step)+1,floor(d/j_step)+1);     % (k,j)->(时
间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
    end
    plot(0:k_step:435/vv,data(:,d/2/j_step))
    xlabel("时间/s")
    ylabel("温度/℃")
end
plot([0,350],[217,217],'r--')
legend("问题 2","问题 3","217℃线")


function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    sigma=interp1(x0,sigma_ls,x);
end
```

Q4.m
GA 主函数

```
%% 初始化
clear
close all
clc

%% GA 参数赋值
NP=30; % 种群数量
G=100;  % 最大遗传次数
L=20;   % 单维二进制数段长度
Dim=5;  % 维度
Max=[185,205,245,265,100];  % 定义域上限
Min=[165,185,225,245,60];    % 定义域下限
max_sta=inf;     % 最大稳定代数，稳定超过此代数将结束算法
max_err=30;     % 最大不符合约束条件次数

%% 遗传算法
f=zeros(NP,L*Dim);    % 初始种群
for i=1:NP
    f(i,:)=d2b([182,203,237,254,80.1-(i-1)/10],Min,Max,L);
end
max_fit_rec=zeros(1,G);

for k=1:G
    tic
    %% 开始迭代
    for i=1:NP
        U=f(i,:);
        x(i,:)=b2d(U,Min,Max,L);
        [Fit(i),Fit1(i),Fit2(i)]=aim(x(i,:));
    end
    [max_fit,max_fit_idx]=max(Fit);
    f_best=f(max_fit_idx,:);
    Fit=(Fit-min(Fit))./(max_fit-min(Fit));

    %% 计算稳定代数
    for idx=k-1:-1:1
        if max_fit_rec(idx)~=max_fit
            break
        end
    end
    sta_num=k-idx;  % 稳定代数

    %% 判断是否稳定超 max_sta 代
    if sta_num>=max_sta
```

```
        break
    end

    %% 基于轮盘赌选择法的复制操作
    sum_Fit=sum(Fit);
    fitvalue=Fit/sum_Fit;
    fitvalue=(cumsum(fitvalue));
    ms=sort(rand(NP,1));
    fitidx=1;
    newidx=1;
    while newidx<=NP-1
        if ms(newidx) < fitvalue(fitidx)
            nf(newidx,:)=f(fitidx,:);   % 复制
            newidx=newidx+1;
        else
            fitidx=fitidx+1;
            if fitidx>NP
                break;
            end
        end
    end

    %% 基于概率的交叉操作
    for i=1:NP-2
        p=rand;
        if p<get_pc(sta_num)
            t=1;
            for trytime=1:max_err
                tnf=zeros(1,L*Dim);
                q=randi([0,1],1,L*Dim);
                for j=1:L*Dim
                    tnf(j)=nf(i+q(j),j);
                end
                if judge(b2d(tnf,Min,Max,L))
                    nf(i,:)=tnf(:);
                    break
                end
            end
        end
    end

    %% 基于概率的变异操作
    for i=1:round((NP-1)*get_pm(sta_num))
        h=randi([1,NP-1],1,1);
```

```
        t=1;
        for trytime=1:max_err
            tnf=nf(h,:);
            for j=1:round(L*Dim*get_pm(sta_num))
                g=randi([1,L*Dim],1,1);
                tnf(g)=~tnf(g);
            end
            if judge(b2d(tnf,Min,Max,L))
                nf(i,:)=tnf(:);
                break
            end
        end
    end

    %% 下一代预备
    f=nf;
    f(NP,:)=f_best; % 保留精英
    max_fit_rec(k)=max_fit;
    usetime=toc;
    fprintf("完成计算第%d 代,适应度%f(面积%.3f,对称性指标%.2f),进度%.6f,用
时%.2fs,预计还
需%.3fmin\n",k,max_fit,Fit1(max_fit_idx),Fit2(max_fit_idx),k/G,usetime,us
etime*(G-k)/60)
end

%% 收尾
for i=1:NP
    Fit(i)=aim(b2d(f(i,:),Min,Max,L));  % 计算每个样本适应度
end
[max_fit,max_fit_idx]=max(Fit);

%% 显示结果
fprintf("最大值:")
disp(max_fit)
fprintf("最大值点:")
disp(x(max_fit_idx,:))
fprintf('精度：x±')
disp((Max-Min)/(2^L-1));

function [fit,fit1,fit2]=aim(x)
    % 目标函数
    %% 设置炉温
    U_info=[ones(1,5)*x(1),x(2),x(3),ones(1,2)*x(4),ones(1,2)*25];
    % 炉前,1-4,5,6,7,8,9,10-11,炉后
```

```
    D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
    sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
    len=30.5;
    gap=5;
    global x0 y0
    x0=[0,25];
    y0=[25];
    D_ls=[D_info(1)];
    sigma_ls=[sigma_info(1)];
    for i=1:11
        x0(end+1)=x0(end)+len;
        x0(end+1)=x0(end)+gap;
        y0(end+1)=U_info(i);
        y0(end+1)=U_info(i);
        D_ls(end+1)=D_info(i+1);
        D_ls(end+1)=D_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
    end
    x0(end)=435.5;
    y0(end+1)=25;
    D_ls(end+1)=D_ls(end);
    sigma_ls(end+1)=sigma_info(end);
    %% 计算
    v=x(5);
    d=0.15;     % 厚度(mm)
    j_step=0.005;   % 厚度步长(mm)
    k_step=0.005;     % 时间步长(s)
    vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
    data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     %
(k,j)->(时间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    key=ones(k_max,1)*25; % 中心温度记录
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
```

```
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
        key(k)=data(k,d/2/j_step);
        if key(k)-key(k-1)<0 && key(k)<217  % 下降，减少递推次数
            break
        end
    end
    [~,k_idx_umax]=max(key);
    fit1=0;
    fit2=0;
    dk=0;
    for k=find(key>217)'
        if k>k_idx_umax
            break
        end
        fit1=fit1+((key(k)+key(k+1))/2-217)*k_step;
    end
    while key(k_idx_umax+dk)>217 && key(k_idx_umax-dk)>217
        fit2=fit2+abs(key(k_idx_umax+dk)-key(k_idx_umax-dk));
        dk=dk+1;
    end
    fit2=fit2/dk;
    fit=-0.7*fit1-0.3*fit2;
end

function res=judge(x)
    % 约束条件
    % 输入速度 v(cm/s);输出 1-符合约束,0-不符合约束
    %% 设置炉温
    U_info=[ones(1,5)*x(1),x(2),x(3),ones(1,2)*x(4),ones(1,2)*25];
    % 炉前,1-4,5,6,7,8,9,10-11,炉后
    D_info=    [0.00014,ones(1,4)*0.00020,
0.00020,0.00020,0.00027,0.00020,0.00020,0.00015,0.00015];
    sigma_info=[0.012,  ones(1,4)*0.021,    0.024,  0.030,  0.034,
0.025,0.025,0.010,0.010];
    len=30.5;
    gap=5;
    global x0 y0
    x0=[0,25];
    y0=[25];
    D_ls=[D_info(1)];
    sigma_ls=[sigma_info(1)];
    for i=1:11
```

```
        x0(end+1)=x0(end)+len;
        x0(end+1)=x0(end)+gap;
        y0(end+1)=U_info(i);
        y0(end+1)=U_info(i);
        D_ls(end+1)=D_info(i+1);
        D_ls(end+1)=D_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
        sigma_ls(end+1)=sigma_info(i+1);
    end
    x0(end)=435.5;
    y0(end+1)=25;
    D_ls(end+1)=D_ls(end);
    sigma_ls(end+1)=sigma_info(end);
    %% 约束
    v=x(5);
    d=0.15;     % 厚度(mm)
    j_step=0.005;   % 厚度步长(mm)
    k_step=0.005;     % 时间步长(s)
    vv=v/60;    % 传送带前进速度(cm/s)[不要修改参数]
    data=zeros(floor(435/(v/60)/k_step)+1,floor(d/j_step)+1);     %
(k,j)->(时间 s,空间 cm);首行/首列为零时刻/位置
    [k_max,j_max]=size(data);
    data(1,:)=25;   % 初始条件
    key=ones(k_max,1)*25; % 中心温度记录
    count1=0;   % 约束 2 计时器
    count2=0;   % 约束 3 计时器
    for k=2:k_max
        t_now=k*k_step; % 当前时间(s)
        DD=k_step*get_D(t_now*vv,D_ls)/(j_step*j_step);
        sigma=get_sigma(t_now*vv,sigma_ls);
        for j=2:j_max-1
            data(k,j)=DD*(data(k-1,j+1)-2*data(k-1,j)+data(k-1,j-
1))+data(k-1,j);
        end
        data(k,1)=-k_step*sigma*(data(k-1,1)-get_U(t_now*vv))+data(k-1,1);
        data(k,j_max)=-k_step*sigma*(data(k-1,j_max)-
get_U(t_now*vv))+data(k-1,j_max);
        key(k)=data(k,d/2/j_step);
        if key(k)-key(k-1)>3*k_step || key(k)-key(k-1)<-3*k_step    % 约束
(1)
            res=0;
%             fprintf("v=%f,约束(1)跳出\n",v)
            return
        end
```

```
        if key(k)-key(k-1)>0 && key(k)>150 && key(k)<190
            count1=count1+1;
        elseif key(k)>217
            count2=count2+1;
        end
        if count1*k_step>120 || count2*k_step>90 || key(k)>250   % 约束
(2)(3)(4)上界
%             fprintf("v=%f,约束(234)上界跳出\n",v)
            res=0;
            return
        end
        if key(k)-key(k-1)<0 && key(k)<217  % 下降，减少递推次数
            break
        end
    end
    if count1*k_step<60 || count2*k_step<40 || max(key)<240 % 约束
(2)(3)(4)下界
%         fprintf("v=%f,约束(234)下界跳出\n",v)
        res=0;
        return
    end
%     fprintf("v=%f,符合条件\n",v)
    res=1;
end

function pc=get_pc(sta_num)
    % 交叉率，sta_num-稳定代数
    pc=min(0.7+sta_num*0.05,0.9);   % 防止 pc 超过 0.9
end

function pm=get_pm(sta_num)
    % 变异率，sta_num-稳定代数
    pm=min(0.5+sta_num*0.05,0.9);   % 防止 pm 超过 0.9
end

function d=b2d(str,Min,Max,L)
    %格雷解码并将二进制转为定义域内的十进制
    %b2d(str,Xx,Xs,L),str-二进制行向量，Min-下限，Max-上限，L 单维度二进制串长
度
    Dim=length(Min);
    n=zeros(Dim,L);
    for k=1:Dim
        n(k,1)=str((k-1)*L+1);  % 格雷编码解码
        for j=2:L
```

```
            n(k,j)=xor(n(k,j-1),str((k-1)*L+j));    % xor—异或运算
        end
    end
    for di=1:Dim
        m(di)=0;
        for j=1:L   % 二进制转十进制
            m(di)=m(di)+n(di,L-j+1)*(2^(j-1));
        end
    end
    d=Min+m.*(Max-Min)./(2^L-1);    % 十进制解码至定义域
end

function str = d2b (arr,Min,Max,L)
    %格雷编码
    %d2b(arr,Xx,Xs,L),arr-十进制数据数组（行向量），Min-下限，Max-上限，L 单维
度二进制串长度；输出 str 为二进制行向量
    Dim=length(Min);
    str=zeros(1,Dim*L);
    m=round((arr-Min).*(2^L-1)./(Max-Min)); % 分散至二进制定义域
    n=double(boolean(dec2bin(m')-'0'));   % 十进制转二进制
    [~,col]=size(n);
    while col<L     % 首位补 0 至长为 L
        n=[zeros(length(arr),1),n];
        [~,col]=size(n);
    end
    for k=1:Dim % 格雷编码
        str((k-1)*L+1)=n(k,1);  % 首位不变
        str((k-1)*L+2:k*L)=xor(n(k,1:end-1),n(k,2:end));    % 异或运算
    end
end

function U=get_U(x)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0 y0
    U=interp1(x0,y0,x);
end

function D=get_D(x,D_ls)
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    D=interp1(x0,D_ls,x);
end

function sigma=get_sigma(x,sigma_ls)
```

```
% 输入 x-炉内位置(cm)，输出对应炉内温度
    global x0
    sigma=interp1(x0,sigma_ls,x);
end
```