高温作业专用服装设计

摘要

为解决高温作业专用服装设计的问题，本文由相关理论基础入手，了解了现有隔热服[1]的主要特征，建立了合理的数学模型，解决了温度分布问题与最优的材料厚度问题，同时也达到了低研发成本、相对短研发周期的效果。

针对问题一，分析得到假人皮肤与外界环境存在温度差，热量是由高温部分向低温部分传递，这满足热传导的必要条件。所以 ，高温隔热服的设计可以近似的看作为一个热传导问题来解决。首先基于傅里叶定理[2]，可以列出一个一维非稳态热传导方程，基于提出的假设利用有限差分法[3]对列出的偏微分方程进行差分，使计算简单化表达更直观。确定初始条件和边界条件，将求解区域离散化，同时利用了二阶向前差商，一阶向前差商对差分方程进行推导，得出一个近似的一维热传导方程，再一次对这个近似差分方程进行离散处理，在环境温度为 75°C、II 层厚度为 6 mm、IV 层厚度为 5 mm、工作时间为 90 分钟的条件支撑下，运用 Matlab 软件编程（代码见附录程序 1），得到关于 I、II、III 层以及皮肤表层第 IV 层温度随时间变化的分布图，如图三所示，同时得到了如图 4 所示的三维温度分布图，得到的最终结果见 problem1.xlsx。

针对问题二，分析题目可以了解到，要求出第 II 层最优厚度，可确定为一个优化问题。为了求解优化问题，应用 PSO 算法对本优化问题进行反问题求解。首先，基于问题一列出的热传导方程，列出约束条件，确定边界条件，初始条件为工作时间 60 分钟，外界环境为 65°C，设第 II 层的厚度为 x，求得温度场，再反向求解其厚度。然后进行随机初始化粒子坐标，计算每个粒子适应度，记录全部粒子经历的最佳位置，调整粒子速度公式调整粒子坐标，求出最优解。通过层层迭代选择最优区域，最后选取迭代次数为 20，通过 Matlab 软件编程（代码见附录程序 2）求得第 II 层最优厚度为 8.8mm。

针对问题三，通过对题目的分析，可以明显看出问题三是在问题二的基础上增加了一个未知厚度，外界温度和时间约束条件也相应做了调整，我们需要求出 II 层和 IV 层的最优厚度。因此，我们可以把本题确定为一个多目标优化问题。将问题简单化处理，可以把第 II 层和第 IV 层看做一个整体，将这两个层的厚度进行分权处理，最后求出这个整体最优，然后再分别求两者最优，使问题得到简化效果。根据题目，首先确定第 II 层和第 IV 层的厚度、初始条件、边界条件、约束条件。同问题二，运用 PSO 算法对本优化问题进行反问题求解。最后选取迭代次数为 20，求得 II 层最优厚度为 18.9mm,IV 层最优厚度为 4.6mm。运用 Matlab 软件编程（代码见附录程序 3）

关键字：隔热服 傅里叶定理 有限差分法 PSO 算法 Matlab 软件 目标优化 多目标优化

一、问题重述

1.1 问题背景

对于从事户外高温作业的人员而言，热负荷对其生命安全和身心健康有着巨大的威胁。野外勘探、巡线维修[4]、抢火救灾、建筑施工以及邮政快递等均涉及到高温环境的户外作业[5]。在多种高温作业环境中，作业人员将受到来自太阳的辐射热、来自周围环境物体的辐射热(如被加热地面)，以及由作业者自身行为导致的人体大量产热等作用。这些高温作业者工作时间较长，并常处于高温、强热辐射的工作环境。如在我国海南岛等热带地区，夏季炎热多湿，太阳辐射强，平均气温为 22.5℃～25.6℃，极端气温达到 37℃～39℃，相对湿度可达 85%～98%[6]，给户外作业人员带来了极大的不舒适感以及严重的健康问题，主要表现为热应激、中暑以及皮肤癌等[7]。因此有必要设计合理有效的防护措施，以减少高温、强辐射环境产生的接触热、对流热以及辐射热对人体的伤害。

1.2 问题提出

为了降低研发成本、缩短研发周期,建立了合理的数学模型来确定假人皮肤外侧的温度变化情况，并解决以下问题：

(1) 已知环境温度为 75°C、II 层厚度为 6 mm、IV 层厚度为 5 mm、工作时间为 90 分钟，并测量得到假人皮肤外侧的温度（见附件 2)，专用服装材料的某些参数值（见附件 1)，建立数学模型，计算温度分布，并生成温度分布的 Excel 文件（文件名为 problem1.xlsx)。

(2) 当环境温度为 65°C、IV 层的厚度为 5.5 mm 时，为确保工作 60 分钟时，假人皮肤外侧温度不超过 47°C，且超过 44°C 的时间不超过 5 分钟。试确定 II 层的最优厚度。

(3) 当环境温度为 80 °C 时，为确保工作 30 分钟时，假人皮肤外侧温度不超过 47°C，且超过 44°C 的时间不超过 5 分钟。试确定 II 层和 IV 层的最优厚度。

二、问题分析

2.1 问题一的分析

问题一要求根据已给定的数据，求出温度分布，并生成一份温度分布 excle 文件。针对此问题，根据所提供的已知条件，分析出它满足产生导热的必要条件的，即物体的内部存在温度差，因而热量由高温部分向低温部分传递。我们可以根据傅里叶定理列出偏微分方程，利用差分法对傅里叶公式进行变形，得到新的热传导方程。假设流体密度是恒定的，假设不考虑对流，边界条件为绝热。基于这些假设条件列出初始条件，边界条件。利用差分法把拟定的求解区域离散化，利用二阶向前差商和一阶向前差商，得到一个的近似的导热方程，通过运算求出各层温度分布。

2.2 问题二的分析

问题二要求根据已给的温度时间约束条件，以及第 IV 层材料厚度，来求出第 II 层的最优厚度。针对此问题，首先确定这是一个优化问题。本文可以利用问题一建立的微分方程进行求解，将其中的宽度设置为 x，对该 x 进行求解。本题在求解的过程中涉及到最优厚度，现已知温度随时间的变化（根据附件二），运用反向问题法求其厚度。本问题在求解过程中涉及到非线性问题求解，建立以最优厚度为目标的目标函数，运用粒子群算法建立数学模型，但传统的粒子群算法传统优化算法常常会陷入局部最优问题，现代智能优化算法普遍适用随机全局优化问题，所以我们这里选取 PSO[8][9]（微粒群算法与其他进化类算法相类似，也采用“群体”与“进化”的概念，同样也是依据个体（微粒）的适应值大小进行操作。所不同的是，微粒群算法不像其他进化算法那样对于个体使用进化算子，而是将每个个体看作是在”N 维搜索空间中的一个没有重量和体积的微粒。并在搜索空间中以一定的速度飞行。该飞行速度由个体的飞行经验和群体的飞行经验进行动态调整。）建立数学模型，求出最优解。

2.3 问题三的分析

问题三是在问题二的基础上改变外环境温度、工作时间求得 II 层和 IV 层的最优厚度。针对问题三，首先，确定本问题属于多目标优化问题，我们在模型二的基础上，考虑环境温度变为80℃。并且未知层的厚度变为 II 层和 IV 层，此问题中我们运用模型一中的热传递模型。建立多目标优化模型，以第一层和第二层的厚度为目标函数，给两个层的厚度分权简化成一个整体，求出两者分别的最优解。

三、模型假设

假设热量是垂直于皮肤传导的。
假设温度在变化的过程中是连续的。
假设隔热服的材质是均匀的。
假设流体密度是恒定的。
假设在热传递的过程中不考虑对流。
假设边界条件为绝热。
假设题目中所给的数据都是正确的。

四、符号说明

符号	说明
ρ	材料的密度
c	比热容

x	每层高温防护服材料的厚度
τ	导热传递的时间
t	温度
λ	导热速率
α	热交换系数
$\varphi(x)$	初始条件
$g(x)$	边界条件
h	步长
w	惯性权重
V	粒子运行速度
c	加速因子
r	取值范围
P	粒子所经历的历史位置
p_g	粒子所经历的位置
T	最大迭代次数，最大迭代范围选取为 20

五、模型建立与求解

5.1 问题一模型的建立与求解

5.1.1 模型的建立

对题目分析可知，物体的内部存在温度差，因而热量由高温部分向低温部分传递，满足热传导的必要条件。所以，高温作业服装的设计可以看作为多层热传导形式来求出温度分布。首先，根据假设，可以基于热传学基本定理即傅里叶定理，列出一维非稳态热传导方程为：

$$\rho c\frac{\partial t}{\partial \tau}=\frac{\partial}{\partial x}(\lambda\frac{\partial t}{\partial x}) \tag{1}$$

基于做出的假设，利用差分解法对傅里叶热传导公式变形得到的热传导方程为：

$$\frac{\partial u}{\partial t}=\lambda\frac{\partial^2 u}{\partial x^2}\quad\begin{matrix}0\le t\le T\\0\le x\le l\end{matrix} \tag{2}$$

为了解决$u(x,t)$还需确定初始条件（解存在、唯一并且连续都依赖于初始条件）和边界条件。对于一维导热问题（第一类边界条件），拟定 u 为边界温度，x 为材料厚度，t 为温度变量，设定以 x, t 为变量差分方程，同时设定$\varphi(x)$为初始条件 ，$g_1(t)$、$g_2(t)$为边界条件列出下式：

$$\begin{cases} \dfrac{\partial u}{\partial t}=\lambda\dfrac{\partial^2 u}{\partial x^2} & \\ u(x,0)=\varphi(x) & 0\le x\le l \\ u(0,t)=g_1(t) & 0\le t\le T \\ u(l,t)=g_2(t) & \end{cases} \tag{3}$$

值的解锁在求解区域为$G:\{0\le x\le l,0\le t\le T\}$在此区域内的某些离散点$(x_i,t_i)$上求出$u(x_i,t_i)$足够近似的解。设步长为 h，时间为$\tau$利用差分法把求解区域进行离散化（确定离散点）处理公式如下，

$$\begin{aligned} x_i &= x_0+ih=ih \quad & h&=\frac{l}{N} \quad & i&=0,1,...,N \\ t_k &= t_0+k\tau=k\tau \quad & \tau&=\frac{T}{M} \quad & k&=0,1,...,M \end{aligned} \tag{4}$$

通过离散化以温度和材料厚度为变量，$(x_i,t_k)\Rightarrow(i,k)$ $u(x_i,t_k)\Rightarrow u_{i,k}$ 得到了一个二维离散图如图下图所示：

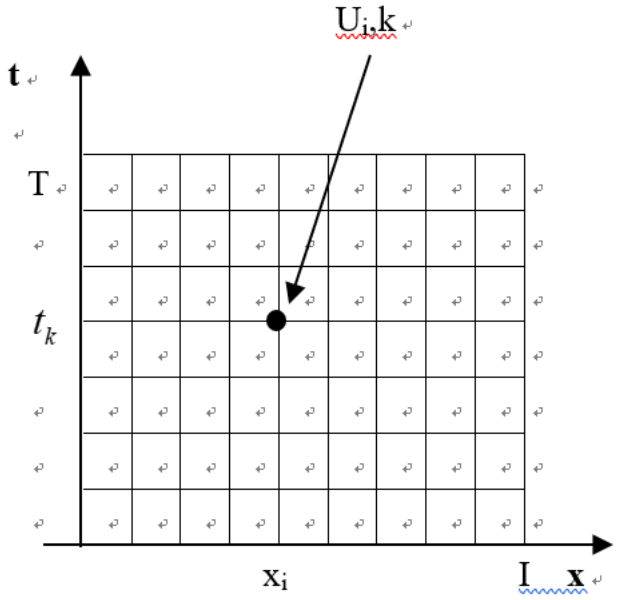

图 1 一维离散图

利用二阶向前差商$O(h^2)$[11]推导差分方程：

$$\frac{\partial^2 u(x,t)}{\partial x^2} \approx \frac{ux+(h,t)-2u(x,t)+u(x-h,t)}{h^2} \tag{5}$$

在节点(x_i,t_k)上

$$\left.\frac{\partial^2 u(x,t)}{\partial x^2}\right|_{\substack{x=x_i \\ t=t_k}} \approx \frac{u(x_i+h,t_k)-2u(x_i,t_k)+u(x_i-h,t_k)}{h^2} \tag{6}$$

$$=\frac{u_{i+1,k}-2u_{i,k}+u_{i-1,k}}{h^2} \tag{7}$$

同样，在节点(x_i,t_k)上转化为一阶向前差商$O(h)$：

$$\frac{\partial(x,t)}{\partial t} \approx \frac{u(x_i+t_k+\tau)-u(x_i,t_k)}{\tau}$$

$$=\frac{u_{i,k+1}-u_{i,k}}{\tau} \tag{8}$$

设λ为导热速率，α为热交换系数。一维热传导方程可以近似为：

$$\frac{u_{i,k+1}-u_{i,k}}{\tau}=\lambda\frac{u_{i+1,k}-2u_{i,k}+u_{i-1,k}}{h^2} \tag{9}$$

$$令\ \alpha=\frac{\tau\lambda}{h^2}$$

$$u_{i,k+1}=\alpha u_{i+1,k}+(1-2\alpha)u_{i,k}+\alpha u_{i-1,k} \tag{10}$$

通过推导出来的近似一维热传导方程同样经过离散处理，得到一个二维离散图如图所示：

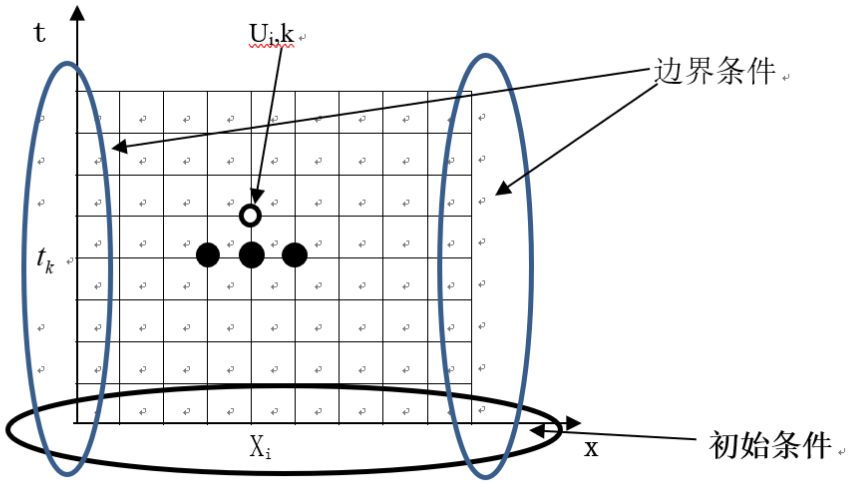

图 2 二维离散图

最后通过计算求出温度分布。

5.1.2 模型的求解

通过建立的非稳态热传递偏微分方程，运用差分法进行求解，求得了如图三所示的温度随时间变化的曲线，由图 3 可以看出随时间变化各层的温度分布。

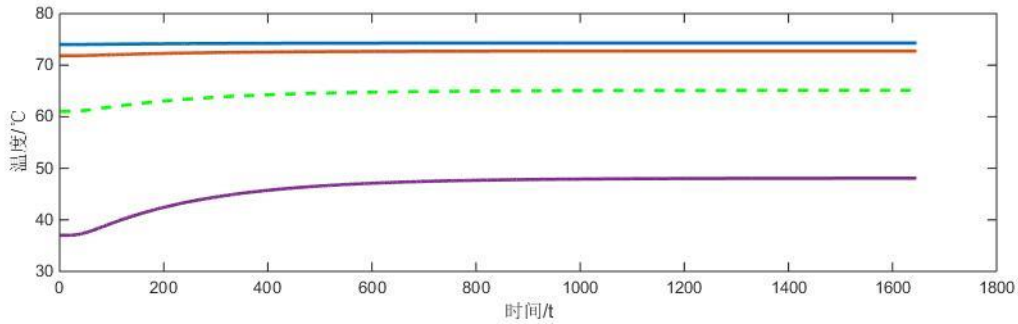

图 3 四层材料温度随时间变化曲线

运用 Matlab 软件，在三维空间下，以时间、厚度、温度为三个控制变量，建立了如图 4 所示的三维温度分布图，可以看出温度达到一个平衡稳定的状态。

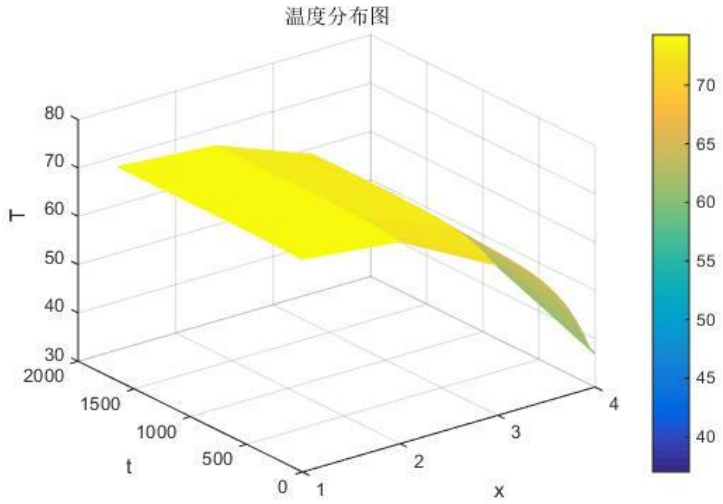

图 4 为温度分布图

得到最终温度分度结果，部分数据如图 5 所示：

时间（s）	温度（℃）		I层	II层	III层	IV层
0	37.00		74.01436	71.83	61.05365	36.99936
1	37.00		74.01436	71.83	61.05365	36.99936
2	37.00		74.01436	71.83	61.05365	36.99936
3	37.00		74.01436	71.83	61.05365	36.99936
4	37.00		74.01436	71.83	61.05365	36.99936
5	37.00		74.01436	71.83	61.05365	36.99936
6	37.00		74.01436	71.83	61.05365	36.99936
7	37.00		74.01436	71.83	61.05365	36.99936
8	37.00		74.01436	71.83	61.05365	36.99936
9	37.00		74.01436	71.83	61.05365	36.99936
10	37.00		74.01436	71.83	61.05365	36.99936

图 5 局部温度分布图

5.2 问题二模型的建立与求解

5.2.1 PSO 算法求解热传导反问题

以最优厚度为目标函数，这样应用目标函数值，可以选择表现优秀的粒子作为局部最优粒子或者全局最优粒子。这样各个粒子在局部最优粒子以及全局最优粒子和自己的飞行速度来更新自己的位坐标。流程图（图 5）是应用 PSO 算法求解热传导反问题的算法描述。

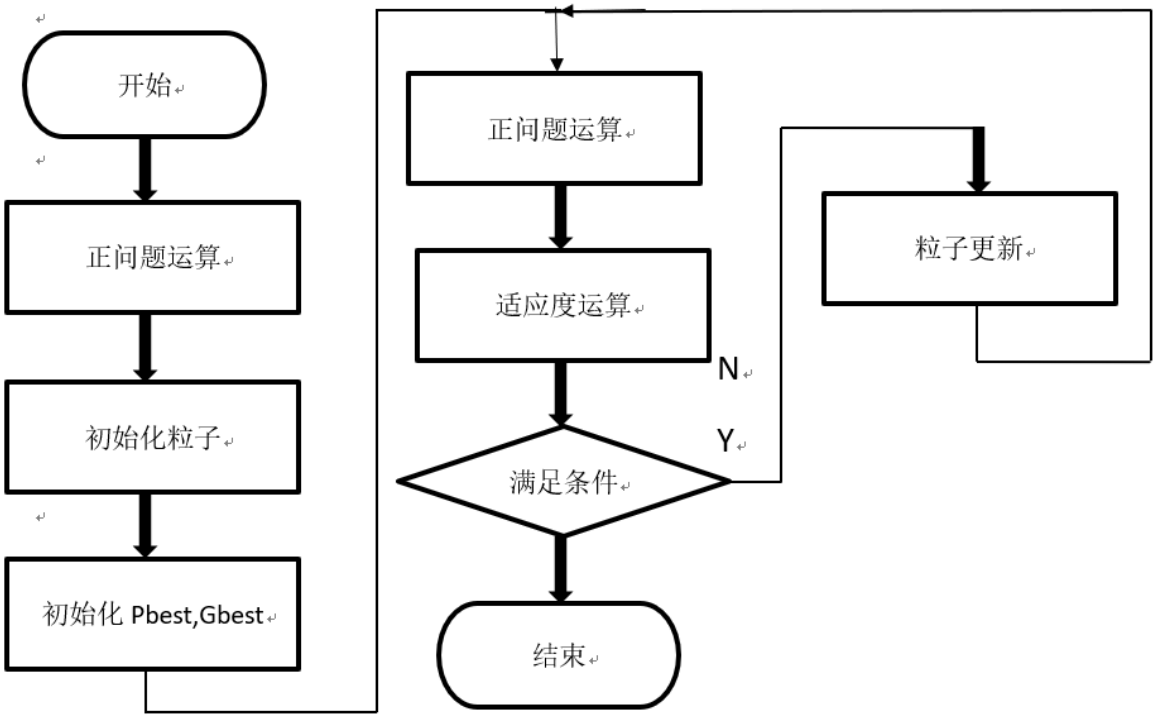

图 6PSO 算法求解热传导反问题的流程图

流程图（图 6）是在该中 PSO 算法中粒子位置坐标更新的描述。

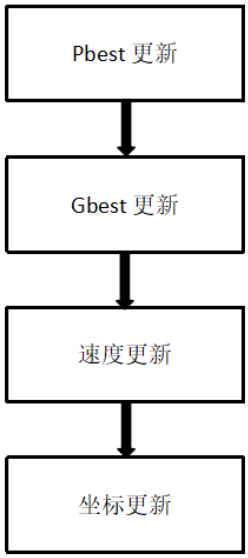

图 7 粒子更新流程图

5.2.2 模型的建立

对题目分析，可知本题具有一定的约束条件，目的是求最优厚度，可以把它规划为一个优化问题 。首先，为解决最优厚度问题，本题将应用 PSO 算法来求解热传导反问题[]。应用 PSO 算法求解热传导反问题。

1、根据实际经验得出的热传导参数值进行正问题求解，因为热传导在不同层之间的热扩散系数不同，a=热传导率/比热*密度，所以温度分布不同，利用问题一建立的微分方程（1）现在已知温度随时间的变化规律（题中所给附件一分析可知），根据题目可以确定厚度为：

$$x_1 = 0.6mm$$
$$x_2 = (L_1 - 0.6)mm$$
$$x_3 = 3.6mm$$
$$x_4 = 5.5mm$$

确定第 II 层的初始条件为：

$$T\left(x_2,0\right)=T\left(L_1-0.6,0\right) \tag{11}$$

第 II 层左边界条件为：

$$-k\frac{\partial T}{\partial x}\Big|_{x=o}=\left(q"\right)\Big|_{x=L_2-0.6} \tag{12}$$

第 II 层右边界条件为：

$$-k\frac{\partial T}{\partial x}\Big|_{x=o}=\left(q"\right)\Big|_{x=L_2-0.6} \tag{13}$$

约束条件为：

$T_{1=}65℃$

$T(L_1\text{-}0.6,1800)<=47℃$

$T(L_1\text{-}0.6,300)<=44℃$

最后求得温度场 u。

2、随机初始化粒子的坐标随机初始化粒子的坐标(即为各粒子各参数赋值)，然后初始化粒子的适应度值,以及局部最优。

3、初始化各粒子的速度依此形成 k 个粒子。初始化各粒子的最大速度,以及惯性权重 w，加速因子 c1、c2。最大迭代次数 T，当前次数 t。(i=0,1,2,3…K，j=0,1,2,3,4,5….)。

4、计算每个粒子的适应度函数值,求得每个粒子的目标函数值.记录目前每个粒子经历的最佳位置。并记录目前全部粒子经历的最佳位置。

5、调整粒子的速度公式如下：

$$V_i\left(t+1\right)=w\cdot V_i\left(t\right)+c_1\cdot r_1\cdot\left(p_i\left(t\right)-X_i\left(t\right)\right)+c_2\cdot r_2\cdot\left(p_g\left(t\right)-X_i\left(t\right)\right) \tag{14}$$

6、调整粒子的坐标，公式如下：

$$x_i\left(t+1\right)=x_i\left(t\right)+V_i\left(t+1\right) \tag{15}$$

在满足约束条件、目标函数的条件下，通过迭代求出最优解。

5.2.3 模型的求解

求解过程如下图所示：

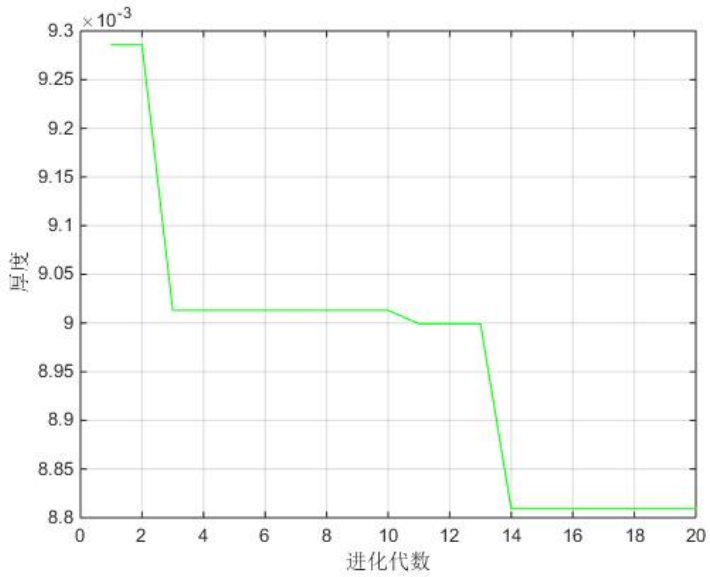

图 8 最优解迭代过程

运用 PSO 算法层层迭代寻求最优解，最后选取迭代次数为 20，通过 Matlab 软件[10]编程（见附录程序 2）由于要满足低成本，短周期的条件，选取 II 层最优厚度为 8.8mm。

5.3 问题三模型的建立与求解

5.3.1 模型的建立

分析问题三，可以知道是在问题二的基础上改变了外界环境温度，与实践限制条件，求 II 层和 IV 层出两层厚度。可见，这是一个多目标优化问题。应该建立多目标优化模型，通过将 II 层和 IV 层看做一个整体，求出整体最优，然后再分别求两者最优，使问题得到简化效果。求最优解还是应用 PSO 算法，由约束条件，已知条件有所变化，改变数值后可带入计算。

此问题我们运用到了问题一的热传递模型：

$$-k\frac{\partial T}{\partial t}\Big|_{x=3.6}=q \tag{16}$$

$$\rho C_{\rho}\frac{\partial T}{\partial t}=\frac{\partial}{\partial x}\left(k\frac{\partial T}{\partial t}\right) \tag{17}$$

根据题目可以确定第二层和第三层的厚度，其中：

$$x_1 = 0.6mm$$

$$x_2 = (L_2 - 0.6)mm$$

$$x_3 = 3.6mm$$

$$x_4 = (\mathrm{L}_4 - 3.6)mm$$

同理确定第二层和第三层初始条件，边界条件

约束条件：
$$s.t\begin{cases} T_I = 80℃ \\ T(L_2 - 0.6, L_4 - 3.6, 1800) \le 47℃ \\ T(L_2 - 0.6, L_4 - 3.6, 300) \le 44℃ \end{cases}$$

求出温度场。

其他步骤同问题二。

5.3.2 模型的求解

求解过程如下：

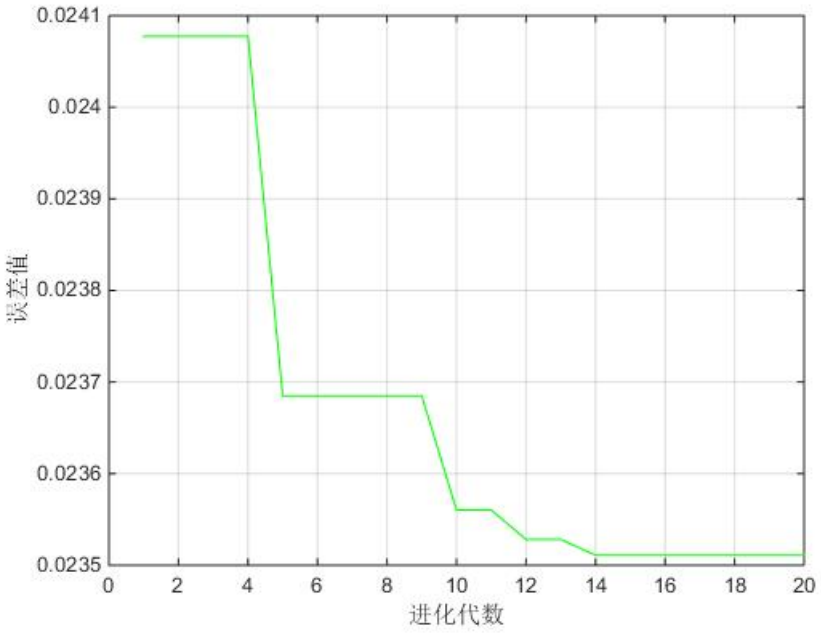

图 9 最优解迭代过程

同问题二一样利用粒子群算法，选取迭代次数为20，求得二层最优厚度。利用Matlab软件（见附录程序 3）求出最优解为 18.9mm，四层最优厚度为 4.6mm.求得最终结果为为18.9mm，四层最优厚度为 4.6mm。

六、模型检验

把求得的四层温度随时间变化函数与所给数据进行比较，然后计算出平均相对误差1.052142897871946e-05，从而证明模型是合理正确的。

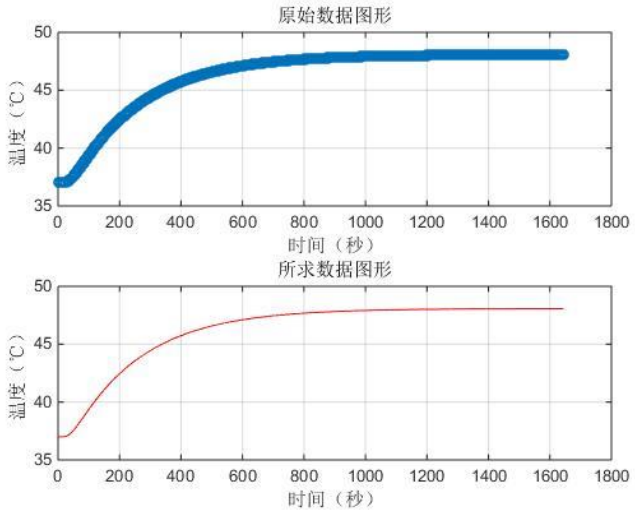

图 10 四层温度变化对比

七、模型评价

7.1 模型的优点

有限差分法可以将方程进行离散化，用有限个网格节点代替连续的求解域，数学概念直观，表达简单。

PSO 算法具有记忆性，粒子群体的历史最好位置可以记忆并传递给其它粒子，而且需要调整的参数较少，结构相对简单，易于工程实现。

7.2 模型的缺点

有限差分法在实际中边界条件比较复杂，两个方向的步长不好取，不太实用。

粒子群算法容易陷入局部最优，导致收敛精度低。

八、模型推广

在本次建模的背景是基于高温作业的工作环境，在建筑、搬运、印染、采掘、冶金、等行业和航空、海下、等舱体动力区或其它军事环境中也广泛存在着高温、强热辐射的工作环境，对于这些作业环境，也可以得到广泛的应用。

九、参考文献

[1]靳会峰.消防装备管理教学片的设计与制作——以《消防员隔热防护服》为例[J].中国教育信息化,2016(18):60-61+65.

[2]李昂,王岳,陶然.傅里叶热传导方程和牛顿冷却定律在流体热学研究中的数学模型应用[J].工业技术创新,2016,03(03):498-502.
[3]张金海,王卫民,赵连锋,姚振兴.傅里叶有限差分法三维波动方程正演模拟[J].地球物理学报,2007(06):1854-1862.
[4]彭辉. 基于 ArcGIS Server 的石油管道巡检管理系统设计与实现[D].湘潭大学,2016.
[5]Relationships between micronutrient losses in sweat and blood pressure among heatexposed steelworkers. TANG Y M,WANG D G,LI J,et al. Industrial Health . 2016
[6]郭甜甜,陈圣波,陆天启.基于 MODIS 数据的多时相海表温度反演——以海南岛西南部近海海域为例[J].热带海洋学报,2017,36(01):9-14.
[7]佚名.高温作业对人体健康的影响[J].安全与健康,2009(07):53.
[8]Amin Safari. A PSO Procedure for a Coordinated Tuning of Power System Stabilizers for Multiple Operating Conditions[J]. Journal of Applied Research and Technology,2013,11(5).
[9]Zeynab Sedaghatjoo,Mehdi Dehghan,Hossein Hosseinzadeh. Numerical solution of 2D Navier–Stokes equation discretized via boundary elements method and finite difference approximation[J]. Engineering Analysis with Boundary Elements,2018,96.
[10]张宏. 基于 QPSO 算法的二维热传导反问题的研究[D].江南大学,2007.
[11]陈亮,顾传青,郑林.非线性方程的数值迭代法及其半局部收敛性[J].数学进展,2014,43(04):481-495.

十、附录

程序 1

```
function heat_conduction() %一维齐次热传导方程
L=input('宽度 L:');
N=input('空间点数 N:');
M=input('时间点数 M:');
alfa=input('扩散系数 alfa:');
lambda=input('稳定条件的值 lambda(取值必须小于 0.5):');%lambda 的值必须小于 0.5
%**************************************************
h=L/N;%空间步长
z=0:h:L;
z=z';
tao=lambda*h^2/alfa;%时间步长
tm=M*tao;%热传导的总时间 tm
t=0:tao:tm;
t=t';
%计算初值和边值
T=zeros(N+1,M+1);
Ti=init_fun(z);
To=border_funo(t);
```

```
Te=border_fune(t);
T(:,1)=Ti;
T(1,:)=To;
T(N+1,:)=Te;
%用差分法求出温度 T 与宽度 L、时间 t 的关系
for k=1:M
    m=2;
    while m<=N
    T(m,k+1)=lambda*(T(m+1,k)+T(m-1,k))+(-2*lambda+1)*T(m,k);
    m=m+1;
    end;
end;
%设置立体网格
for i=1:M+1
    X(:,i)=z;
end;
for j=1:N+1
    Y(j,:)=t;
end
mesh(X,Y,T);
view([1 -1 1]);
xlabel('Z');
ylabel('t');
zlabel('T');

function y=init_fun(z)%初值条件
y=1-z.^2;
return

function y=border_funo(t)%z=0 的边界条件
y=75+t.*0;
return

function y=border_fune(t)%z=L 的边界条件
y=37+t*.0;
return


clc
clear
%导入数据
dataSTS1=xlsread('dataST.xlsx',2);
%时间
S=dataSTS1(:,1);
```

```
%温度
T=dataSTS1(:,2);
subplot(2,1,1);
plot(S,T,'o');
%  图形的一些设置
xlabel('时间（秒）');
ylabel('温度（℃）');
title('原始数据图形')
grid on
%  所求数据图像
T1=dataSTS1(:,3);
subplot(2,1,2);
plot(S,T1,'r')
xlabel('时间（秒）');
ylabel('温度（℃）');
title('所求数据图形');
grid on
%平均相对误差
wucha=(T-T1)./T1;
sw=sum(wucha);
f=sw/length(T)
```

程序 2

```
%% clear
clear
clc

%% pso  算法解决模型寻优问题
maxgen=20;       %进化次数
sizepop=5;     %种群规模
Vmax=[0.001];              %速度最大值
Vmin=[-0.001];             %速度最小值
popmax=[25*0.001];   %粒子最大值
popmin=[7*0.001]; %粒子最小值


    %%  产生初始粒子和速度
    for i=1:sizepop
        %随机产生一个种群
        pop(i,:) = popmin + (popmax-popmin).*rand;
        V(i,:)= Vmin + (Vmax-Vmin).*rand;     %初始化速度
        %计算适应度
        fitness(i)=func2(pop(i,:));         %染色体的适应度
    end
```

```
%% 个体极值和群体极值
[bestfitness bestindex]=min(fitness); %找最小适应度
zbest=pop(bestindex,:);                    %全局最佳
gbest=pop;      %个体最佳
fitnessgbest=fitness;     %个体最佳适应度值
fitnesszbest=bestfitness;     %全局最佳适应度值

%% 迭代寻优
for i=1:maxgen
    i
    for j=1:sizepop
        j
        %计算权重
        weight=1;
        weight=0.9+(0.9-0.5)*exp(-20*(i^6)/(maxgen^6));
        %计算学习因子
        c1=2.5-1.5*i/maxgen;
        c2=1+1.5*i/maxgen;
        %速度更新
        V(j,:) = weight*V(j,:) + c1*rand*(gbest(j,1:length(popmax)) - pop(j,1:length(popmax))) +
c2*rand*(zbest(1:length(popmax)) - pop(j,1:length(popmax)));
        for k=1:size(V,2)
            if V(j,k) > Vmax(k)
                V(j,k) = Vmax(k);
            end
            if V(j,k) < Vmin(k)
                V(j,k) = Vmin(k);
            end
        end

        %种群更新
        pop(j,1:length(popmax))=pop(j,1:length(popmax))+V(j,1:length(popmax));
        for k=1:length(popmax)
            if pop(j,k)>popmax(k)
                pop(j,k)=popmax(k);
            end
        end
        for k=1:length(popmax)
            if pop(j,k)<popmin(k)
                pop(j,k)=popmin(k);
            end
        end
```

```
            %适应度值
            fitness(j)=func2(pop(j,:));

        end

        for j=1:sizepop

            %个体最优更新
            if fitness(j) < fitnessgbest(j)
                gbest(j,:) = pop(j,:);
                fitnessgbest(j) = fitness(j);
            end

            %群体最优更新
            if fitness(j) < fitnesszbest
                zbest = pop(j,:);
                fitnesszbest = fitness(j);
            end
        end
        yy(i)=fitnesszbest;       %保存下每一代的最优值
    end

    figure;
    plot(yy,'g');
    grid on
    xlabel('进化代数');ylabel('厚度');

plotmap2(zbest);




%% func
function yy = func2(x)

%% x
rate = 0.84;
hR = 6.5*0.001;
Ck = 1.28*1e-7;

%%  二层厚度
h2 = x;

M = 12000000/3;
```

```
T = 3600;
deltat = T/M;
N = 100;
h1 = 0.6*0.001;
h3 = 3.6*0.001;
h4 = 5.5*0.001;
L = h1+h2+h3+h4+hR;
deltax = L / N;

r = deltat/(deltax^2);
au = zeros(N+1,1);

for i=1:N+1
    pos = (i-1)*deltax;
    if pos >=0 && pos<h1
        C = 0.082;
        c = 1377;
        rou = 300;
        au(i) = (C/(c*rou))*rate;
    elseif pos>=h1 && pos < h1+h2
        C = 0.37;
        c = 2100;
        rou = 862;
        au(i) = (C/(c*rou))*rate;
    elseif pos>=h1+h2 && pos < h1+h2+h3
        C = 0.045;
        c = 1726;
        rou = 74.2;
        au(i) = (C/(c*rou))*rate;
    elseif pos>=h1+h2+h3 && pos<=h1+h2+h3+h4
        C = 0.028;
        c = 1005;
        rou = 1.18;
        au(i) = (C/(c*rou))*rate;
    elseif pos>=h1+h2+h3+h4 && pos <= L
        au(i) = Ck;
    end
end

%% boundary
u0t = 65;
uNt = 37;
uF = 37;
```

```
%% u
u = zeros(N+1,2);
u(1,:)=u0t;
u(N+1,:)=uNt;
u(2:end,1)=uNt;
CC = r*au;
ref = zeros(1,M+1);
ref(1)=uF;

yy = x;
for j=2:M+1
    if mod(j,2) == 0
        pos1 = 2;
        pos2 = 1;
    else
        pos1 = 1;
        pos2 = 2;
    end
    u(2:N,pos1) = CC(2:N).*u(1:N-1,pos2)+CC(2:N).*u(3:N+1,pos2)+(1-2*CC(2:N)).*u(2:N,pos2);
    ref(j)=u(N+1-floor(hR/deltax),pos1);
    if ref(j) > 47
        yy = yy + inf;
        break;
    end
    if ref(j) > 44 && (M+1-j+1 > 5*60/deltat)
        yy = yy + inf;
        break;
    end
end
```

程序 3：

```
%% clear
clear
clc

%% pso  算法解决模型寻优问题
maxgen=20;       %进化次数
sizepop=5;     %种群规模
Vmax=[0.001,0.0005];              %速度最大值
Vmin=[-0.001,-0.0005];            %速度最小值
popmax=[25*0.001,6*0.001];    %粒子最大值
popmin=[7*0.001,4*0.001]; %粒子最小值
```

```
%% 产生初始粒子和速度
for i=1:sizepop
    %随机产生一个种群
    pop(i,:) = popmin + (popmax-popmin).*rand(1,2);
    V(i,:)= Vmin + (Vmax-Vmin).*rand(1,2);     %初始化速度
    %计算适应度
    fitness(i)=func3(pop(i,:));          %染色体的适应度
end

%% 个体极值和群体极值
[bestfitness bestindex]=min(fitness); %找最小适应度
zbest=pop(bestindex,:);                          %全局最佳
gbest=pop;       %个体最佳
fitnessgbest=fitness;     %个体最佳适应度值
fitnesszbest=bestfitness;     %全局最佳适应度值

%% 迭代寻优
for i=1:maxgen
    i
    for j=1:sizepop
        j
        %计算权重
        weight=1;
        weight=0.9+(0.9-0.5)*exp(-20*(i^6)/(maxgen^6));
        %计算学习因子
        c1=2.5-1.5*i/maxgen;
        c2=1+1.5*i/maxgen;
        %速度更新
        V(j,:) = weight*V(j,:) + c1*rand*(gbest(j,1:length(popmax)) - pop(j,1:length(popmax))) +
c2*rand*(zbest(1:length(popmax)) - pop(j,1:length(popmax)));
        for k=1:size(V,2)
            if V(j,k) > Vmax(k)
                V(j,k) = Vmax(k);
            end
            if V(j,k) < Vmin(k)
                V(j,k) = Vmin(k);
            end
        end

        %种群更新
        pop(j,1:length(popmax))=pop(j,1:length(popmax))+V(j,1:length(popmax));
        for k=1:length(popmax)
            if pop(j,k)>popmax(k)
```

```
                pop(j,k)=popmax(k);
            end
        end
        for k=1:length(popmax)
            if pop(j,k)<popmin(k)
                pop(j,k)=popmin(k);
            end
        end

        %适应度值
        fitness(j)=func3(pop(j,:));

    end

    for j=1:sizepop

        %个体最优更新
        if fitness(j) < fitnessgbest(j)
            gbest(j,:) = pop(j,:);
            fitnessgbest(j) = fitness(j);
        end

        %群体最优更新
        if fitness(j) < fitnesszbest
            zbest = pop(j,:);
            fitnesszbest = fitness(j);
        end
    end
    yy(i)=fitnesszbest;      %保存下每一代的最优值
end

figure;
plot(yy,'g');
grid on
xlabel('进化代数');ylabel('误差值');

plotmap3(zbest);



%% func
function yy = func3(x)

%% x
```

```
rate = 0.86;
hR = 6.5*0.001;
Ck = 1.301e-7;

%% 二层厚度
h2 = x(1);
h4 = x(2);

M = 12000000/5;
T = 1800;
deltat = T/M;
N = 100;
h1 = 0.6*0.001;
h3 = 3.6*0.001;
L = h1+h2+h3+h4+hR;
deltax = L / N;

r = deltat/(deltax^2);
au = zeros(N+1,1);

for i=1:N+1
    pos = (i-1)*deltax;
    if pos >=0 && pos<h1
        C = 0.082;
        c = 1377;
        rou = 300;
        au(i) = (C/(c*rou))*rate;
    elseif pos>=h1 && pos < h1+h2
        C = 0.37;
        c = 2100;
        rou = 862;
        au(i) = (C/(c*rou))*rate;
    elseif pos>=h1+h2 && pos < h1+h2+h3
        C = 0.045;
        c = 1726;
        rou = 74.2;
        au(i) = (C/(c*rou))*rate;
    elseif pos>=h1+h2+h3 && pos<=h1+h2+h3+h4
        C = 0.028;
        c = 1005;
        rou = 1.18;
        au(i) = (C/(c*rou))*rate;
    elseif pos>=h1+h2+h3+h4 && pos <= L
        au(i) = Ck;
```

```
        end
end

%% boundary
u0t = 80;
uNt = 37;
uF = 37;

%% u
u = zeros(N+1,2);
u(1,:)=u0t;
u(N+1,:)=uNt;
u(2:end,1)=uNt;
CC = r*au;
ref = zeros(1,M+1);
ref(1)=uF;

yy = x(1)+x(2);

for j=2:M+1
    if mod(j,2) == 0
            pos1 = 2;
            pos2 = 1;
    else
            pos1 = 1;
            pos2 = 2;
    end
    u(2:N,pos1) = CC(2:N).*u(1:N-1,pos2)+CC(2:N).*u(3:N+1,pos2)+(1-2*CC(2:N)).*u(2:N,pos2);
    ref(j)=u(N+1-floor(hR/deltax),pos1);
    if ref(j) > 47
        yy = yy + inf;
        break;
    end
    if ref(j) > 44 && (M+1-j+1 > 5*60/deltat)
            yy = yy + inf;
            break;
    end
end
```