基于热传导模型的热防护服设计

摘要

高温作业时，热防护服能够减缓热转移的速度，使热量尽可能少的在人体皮肤表面积聚，保护工作人员的生命安全。在设计热防护服的过程中，进行大量高温环境的性能测试代价高昂，因此，建立合适的热传递模型，为设计热防护服提供理论依据，意义重大。

针对问题一：首先，我们在忽略热防护服空气层中热对流的情况下，建立基于热传导方程组（抛物型）的一维传热模型。将外部 75℃恒温，附件一中假人皮肤温度随时间的变化数据，以及热防护服各层初始温度接近体温 37℃，转化为偏微分方程的初始条件和边界条件。在转化为古典隐格式的差分方法对模型进行求解，得到各层的温度分布情况。我们注意到假人皮肤温度最终达到与时间无关的热稳态（稳定在 48.08℃），并以此为基础，通过热力学中热阻等相关概念推导出各层边界的热稳态温度，与上述模型求解结果的热稳态进行比对，经检验拟合情况良好，证明了上述模型及求解的合理性。

针对问题二：本问题属于单目标优化问题，即在满足使假人体表温度最终不能超过 47℃，且超过 44℃的时长不大于 5 分钟的条件下，尽量使第二层厚度的织物最薄。由于在外界 65℃恒温的情况下，假人皮肤温度分布未知，问题一的偏微分方程组右界边界条件不确定，因此转而采用傅立叶逆变换求解偏微分方程组，得到温度关于时间和距离的变化规律。在外部 65℃恒温的条件下，通过 PSO 智能优化算法，改变第二层织物厚度，得到非线性规划的最优解，最终求出第二层织物最优厚度为 9.223mm，且最终假人皮肤温度稳定在 44.223℃。

针对问题三：本问题属于多目标优化问题。即在满足假人体表温度最终不能超过 47℃，且超过 44℃的时长不大于 5 分钟的条件下，尽量使第二层和第四层的厚度最薄。基于问题二中通过傅里叶逆变换求解得到的温度分布规律，在外部 80℃恒温的条件下，利用 PSO 智能优化算法，改变第二层织物厚度和第四层的空气厚度，得到最优解。最终求解出 II、IV 厚度分别为 x=11.786mm 和 x=5.261mm。

关键词： Fourier’s Law 古典隐式有限差分 PSO 算法 热稳态检验

§1 问题的重述

1.1 背景介绍

1.1.1 作业服介绍

在消防安全工程中，工人在高温下作业会受到空气中热量的伤害，因此佩戴合格的作业服能保证其生命安全。决定作业服阻热能力的因素有作业服织料的层数和理化性质。但是往往受限于时间空间以及经济上的限制往往很难通过实验来测试作业服的质量。通过对特定温度、层数、理化性质下的作业服进行模拟分析收集数据，从数学角度进行建模刻画，可以大大降低设计作业服的成本。

1.1.2 问题引入

织物的厚度增减目的是为了达到优化作业服阻热能力，在既定成本内进行优化的目的，增加织物的厚度，御热能力会有所提升，然而由于不同层有不同的比热、导热系数、密度等参数，全部增厚不一定能带来最好的优化效果；织物厚度的的增加会使费用也增多，衣物厚重也会增加人体的不适感，所以在进行最优解规划的时候应该考虑这些因素。

1.1.3 研究意义

决定作业服阻热能力的因素有作业服织料的层数和理化性质。但是往往受限于时间空间，并且物理实验具有一定的破坏性和偶然性。因此往往很难通过实验来测试作业服的防热能力。

同时通过数学建模模拟实际过程，避免了物理建模造成的实验费用的浪费。制造商能够减少研发新阻燃服时所需要的成本支出。

1.2 问题重述

1.2.1 问题一：

利用附件二中的数据，其在 75℃的环境下，建立数学模型，刻画温度与时间，距离的函数关系，找出温度场与位移-时间的分布关系，求解后，用 MatLab 将生成的数据导入到 Excel 表示。

1.2.2 问题二

改变外界温度，在附件一给出的其他物理性质不改变的情况下，改变第二层织物材料的厚度，使其在 65℃的环境中，确保在 60 分钟内，假人皮肤外侧温度不超过 47℃，且超过 44℃的总时间不超过 5 分钟，并尽量减少衣物厚度的前提下，阻热服的防热能力达到前述条件，使其能够保护工人。

1.3.3 问题三：

继续改变外界温度，在附件一给出的其他物理性质不改变的情况下，改变第二层织物材料的厚度，使其在 80℃的环境中，确保在 30 分钟内，假人皮肤外侧温度不超过 47℃，且超过 44℃的总时间不超过 5 分钟，并尽量减少衣物厚度的前提下，阻热服的防热能力达到前述条件，使其能够保护工人。

§2 问题的分析

2.1 问题总体分析

该题分三问，第一问是外界温度为 75℃的环境下，求温度场的分布，二三问则是对阻燃服的相应的层数厚度的情况下，求解满足题目要求的最优解。

其中涉及到的知识应该有传热学知识，最优化问题，非线性求解规划问题。

2.2 个问分析

2.2.1 问题一

首先题目给出了环境温度 75°C 下、II 层厚度为 6 mm、空气层厚度为 5 mm，90 分钟的实验数据。假人皮肤外侧的温度。我们采用 Excel 分析发现，在 t<1645 时，假人外表温度变化较大，但是在 t>1645 时，假人表面温度变化不大。该现象符合导热偏微分方程传热。因此通过 Fourier’s Law 来刻画温度场和热能流动的关系表达。成功建立以后，我们使用网格划分，按照物理性质的不同将四层防护层划分开来，每一层再层层细分求有限差分。通过有限差分方程遍历求解出温度场方程的分布。最后通过 MatLab 导出 Excel 数据求解。

2.2.2 问题二：

在问题一的模型基础上，问题二对问题提出目标优化，要求求解非稳态下的最优解，在这一点我们需要建立约束条件，题目中要求的是在假人温度不超过 47℃，且对超过的时间有要求。考虑以温度-时间函数来作为约束条件。并且结合实际考虑最优解是否符合生活实际（成本最小化）。

2.2.3 问题三

在第一问和第二问的的基础上，第三问题是多目标优化。题目中对布料 II 和空气层温度的参数提出问题。采用温度-位移函数进行刻画。通过 PSO 算法遍历求出最优解。

§3 模型的假设

假设 Ⅰ： 织物材料是各向同性的。

假设 Ⅱ：传播过程，不考虑织物本身发生的折叠，织物的物理结构基本上保持不变。

假设 Ⅲ：传播过程中只考虑热传导。不考虑水汽，热辐射。

假设 Ⅳ：传播过程中，只是一维传导，对于 $t=x_0$，存在一个垂直与该位置的等温面，在这一点上任意方向的温度是相同的。

假设 Ⅴ：假设加热过程中，随着温度升高，织物的理化性质不发生改变。

假设 Ⅵ：在实验开始时，阻热服每层材料以及空气层温度视为与假人表皮温度相等

§4 名词解释与符号说明

序号	符号	符号说明
1	T_i	点 i 处温度
2	$\alpha=\frac{k}{\rho c}$	导温系数
3	β	感温系数相对值
4	c	比热容
5	J	泛函
6	t	时间
7	n	任意物体边界处外法线方向处方向向量
8	q	热流密度
9	d	网格分割步长
10	τ	时间分割间隔
11	r	网格比

表格 1 符号说明

§5 模型的建立与求解

5.1.模型一 用有限差分法刻画一维热传导温度场

5.1.1 建模思路

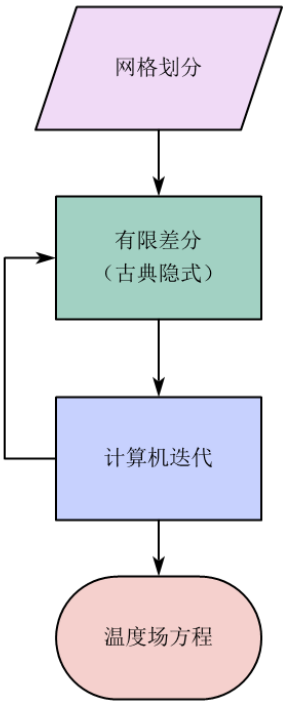

图 1 模型一解题思路

5.1.2 问题一求解

根据假设 I,II,IV,VI 我们将阻热服装的四层防护抽象成上图，热量从外到内，依次进入 I,II,III,IV 层，对应这四层右边的温度为。

$$T_i\ （i=I,II,III，\ IV）\qquad （5.1）$$

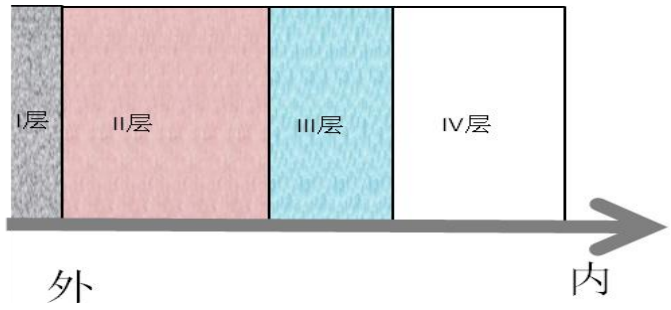

图 2 一维空间的热传导

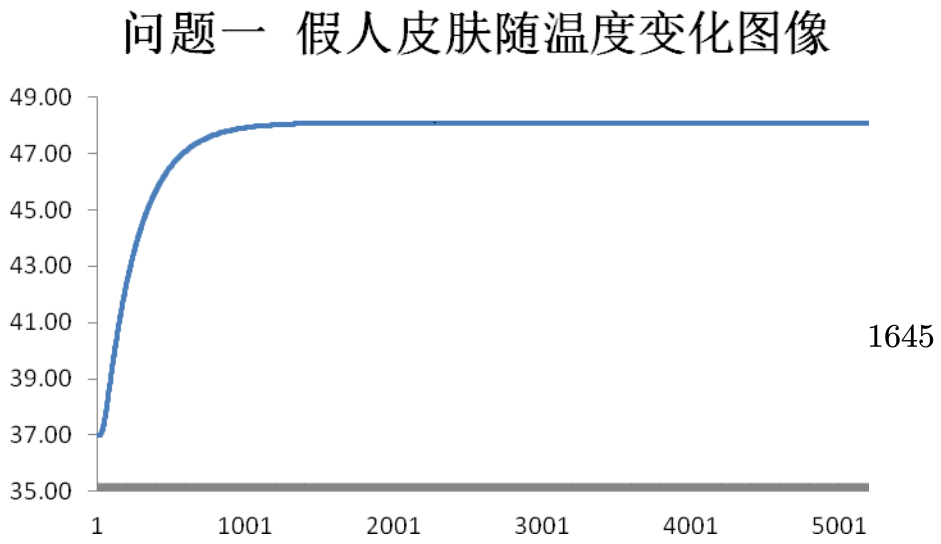

图 3 假人外皮温度模拟

我们将附件二的数据导入 Excel 自带的数据分析程序，我们发现在整个物理变化过程中，存在以下现象：

A 随着温度的变化，的 $T(x,t)$ 导数逐渐减小，后稳定在 48.08℃不变

B 物体相接触的流体介质的温度和物质的物理参数已知

对此我们对应给出以下解释：

a 该题需要一个刻画温度-时间位置三者关系的函数，查阅传热学相关文献

后，发现应该使用 Fourier’s Law。

b $0<t<1645$ 导数为正，当$1645<t<5400$，$T(X,t)$ 为常函数。即该传热物理过程有两阶段，第一阶段为非稳态导热，第二阶段为稳态导热

结合以上两点把$T(X,t)$
表示为

$$\begin{cases} T=T(X,t) & (1645\geqslant t\geqslant 0) \\ T=48.08 & (5400\geqslant t\geqslant 1645) \end{cases}, \tag{5.2}$$

根据物理含意，在 $t\epsilon(0,1645)$ 时，
该系统是无内热源平面非稳态温度场，其微分方程形式如下[1]

$$\begin{cases} \dfrac{\partial T}{\partial t}-\dfrac{k}{c_i\rho_i}\left(\dfrac{\partial^2 T}{\partial t^2}\right)=f \quad t\epsilon[0,5400] & (a) \\ t=o,T=(x) & (b) \\ -k\dfrac{\partial T}{\partial n}|_{\tau}=\alpha(T_I-T_{I-1}) & (c) \end{cases} \tag{5.3}$$

将方程与拉普拉斯方程联立，根据结论：泛函取极值时的极值线 就是拉普拉斯方程在第三类下的解[1]：

$$\begin{cases} \dfrac{\partial^2 T}{\partial x^2}=0 \\ -k\dfrac{\partial T}{\partial x}|_{\tau}=\alpha(T_i-T_{i-1})|_{\tau} \end{cases} \tag{5.4}$$

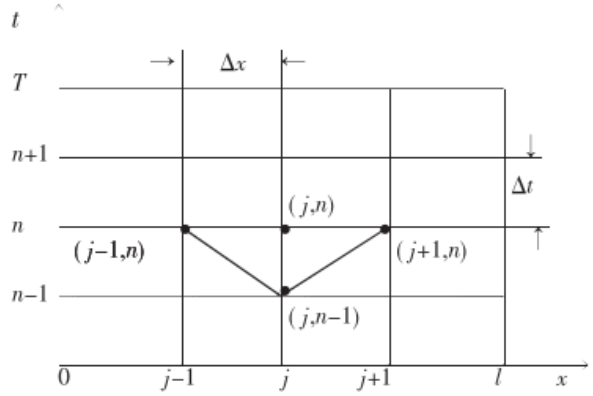

图 4 网格划分

对于这种情况，由于缺少初始边界条件，因此偏微分求解温度场分布方程法思路不可行。故改用有限差法。我们先对不同层材料进行网格划分。

如图 4，将单层材料均匀分成 n 份，则 n=1 时，对应$T_i(x,t)=T_1$。使用古典隐格式，在$(x=x_0,\ t=t_0)$处我们使用向后差商，得到传热有限差分古典隐格式[2]：

$$\frac{T_{i,k+1}-T_{jk}}{\tau}-\frac{\alpha}{d^2}(T_{j+1,k+1}-2T_{j,k+1}+T_{j-1,k+1})=f_{i,k+1} \tag{5.5}$$

$$j=1,2\cdots,N-1 \tag{5.6}$$

$$r=\frac{\tau}{d^2}, \tag{5.7}$$

$O(\tau+d^2)$为局部截断误差又因古典隐格式对网格局部截断误差没有要求，变形得到表达式如下：

$$T(1,j)=\left(T(i,j+1)-\frac{\alpha\tau}{d^2}\times(T(i,j+1)-2\times T(i,\ j+1)+T(i-1.j+1))\right) \tag{5.8}$$

即为绝对稳定的。通过对于$T(X,t)$，我们使用 MatLab 编程后，我们不难得到差分

$$T_{III}(1,j)=(T(i,j+1)-0.003583\times(T(i,j+1)-2\times T(i,\ j+1)+T(i-1.j+1)))$$
$$T_{IV}(1,j)=(T(i,j+1)-0.00459\times(T(i,j+1)-2\times T(i,\ j+1)+T(i-1.j+1)))$$
$$T_{II}(1,j)=(T(i,j+1)-0.003934\times(T(i,j+1)-2\times T(i,\ j+1)+T(i-1.j+1)))$$
$$T_{I}(1,j)=(T(i,j+1)-0.002883\times(T(i,j+1)-2\times T(i,\ j+1)+T(i-1.j+1)))$$

我们通过 MatLab 模拟后，函数的图像如图

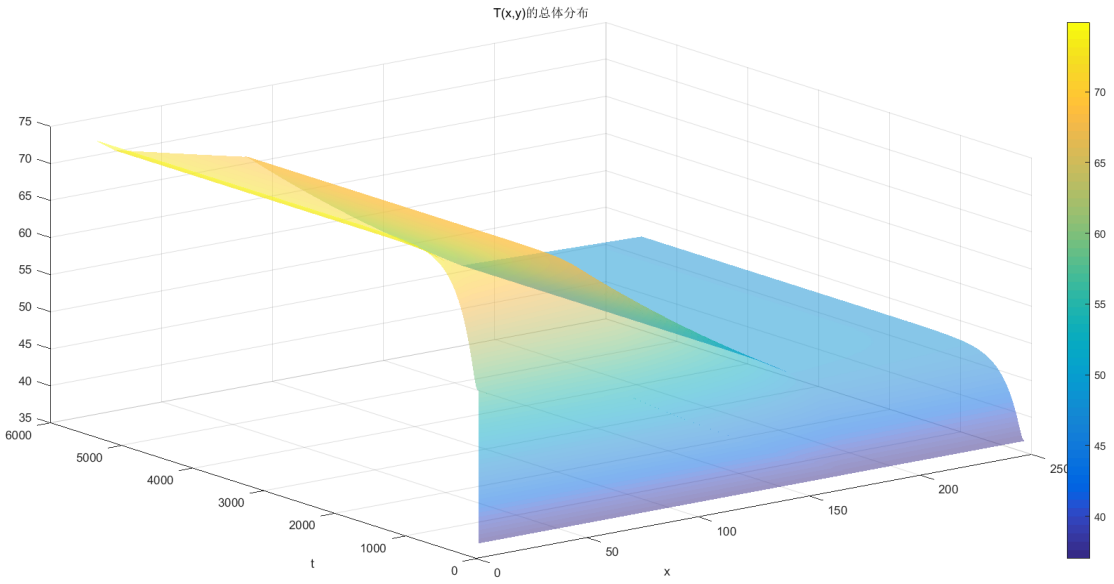

图 5 MatLab 模拟

将四个温度场函数合并成一个分段函数，得上图

将上述温度场函数在 MatLab 中建立图像，如图显示

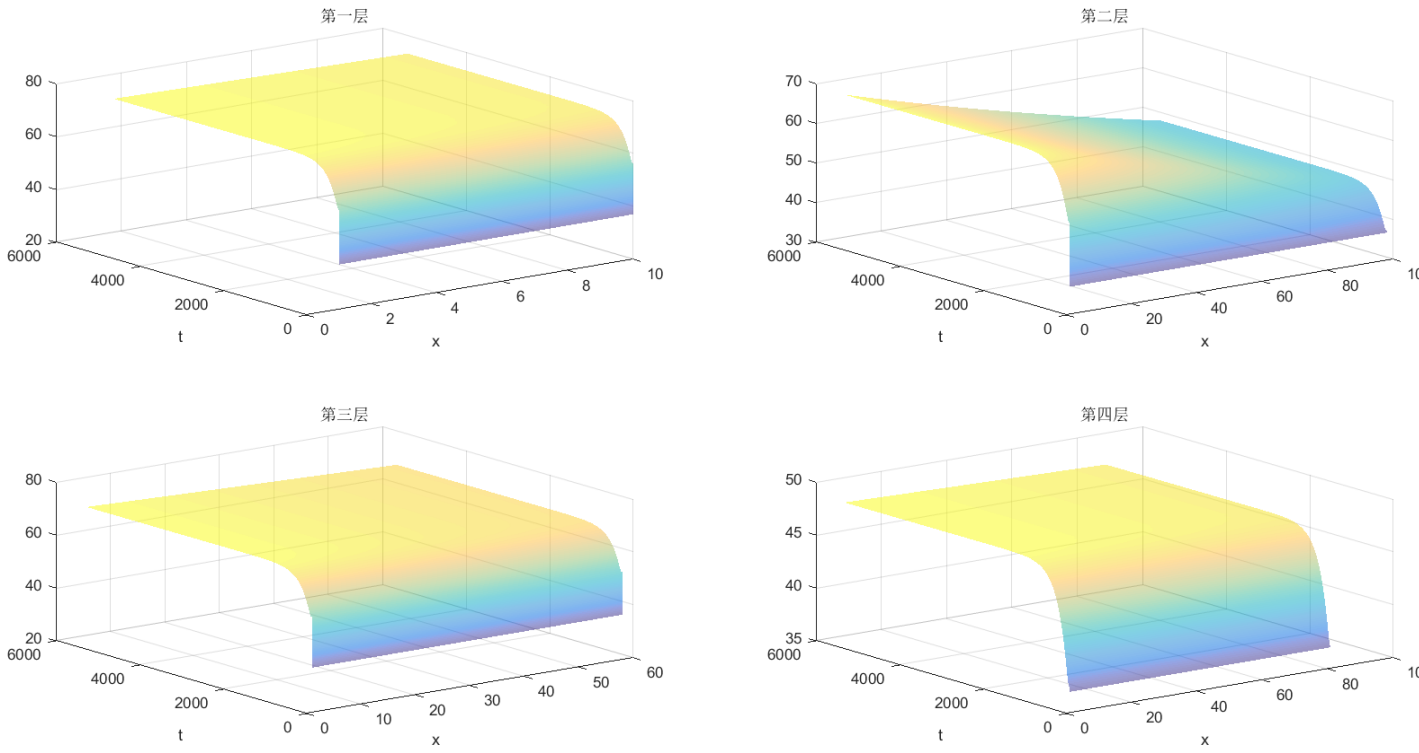

图 6 四层次的 MatLab 图像

5.2 模型二 PSO 求解非线性约束最优解

5.2.1 建模思路

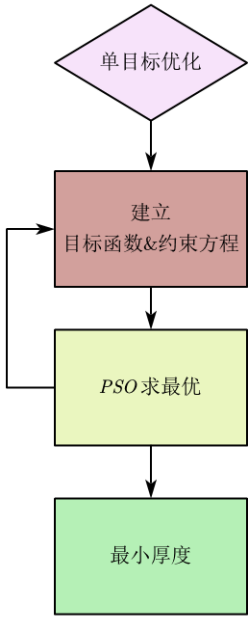

图 7-模型二求解思路

5.2.2 问题二求解

由问题一，我们发现这是一个单目标优化问题，对于

$$T_t = aT_{xx} \quad (t>0, -\infty<x<+\infty)$$

总存在$T(0,x)=\partial(x)$的解。称为 Cauchy 问题一维热传导的基本解。我们使用 Fourier 变换得到[2]:

$$\overline{T}(t,\delta)=F[T(t,x)]=\int_{-\infty}^{+\infty}T(t,\xi)\exp\{i\delta^{\xi}\}d\xi \tag{5.9}$$

$$\begin{gathered} x\to\infty,\ u,u_{\xi}\to 0 \text{解出} \\ \frac{d\overline{T}}{dt}=a\ [(-i\delta)^2]\overline{T}=-a\ x^2\overline{T} \\ \overline{T}|_{t=0}=1 \end{gathered} \tag{5.10}$$

解这个常微分方程为$\overline{T}(t,\delta)=\exp\{-a\ x^2t\}$，作 Fourier 逆变换得[2]:

$$\begin{aligned} u(t,x)&=\frac{1}{2\pi}\int_{-\infty}^{+\infty}\bar{u}\exp\{-i\lambda x\}d\lambda \\ &=\frac{1}{2\pi}\int_{-\infty}^{+\infty}\exp\{-a^2x^2t-i\lambda x\}d\lambda \\ &=\frac{1}{2a\sqrt{\pi t}}\exp\left\{-\frac{x^2}{4a^2t}\right\} \end{aligned} \tag{5.11}$$

根据题目要求，我们可以得到如下

$$T_i(x,t)=\beta_i\sum_{i=}^{IV}\left(-\frac{1}{2\sqrt{ai\pi t}}\right)\times e^{\sum_{i=}^{IV}\left(-\frac{x_i^2}{4ait}\right)}+c \tag{5.12}$$

$$s.t\begin{cases} T(x,t)\big|_{t=3300}<44 \\ T(x,t)<47 \\ i=\ (I,II,III,IV) \end{cases} \quad \{\ \ x\in(0.6,25) \tag{5.13}$$

解出 II 层的材料厚度为 9.223mm。

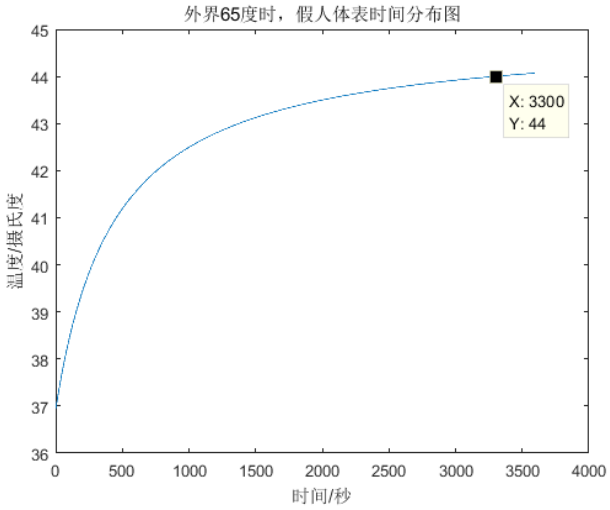

图 9 80℃下假人皮肤温度

求解发现，在 t=3300s 附近刚好为 44℃满足题目要求，说明求解情况良好。

5.3.问题三 双目标优化

5.3.1 建模思路

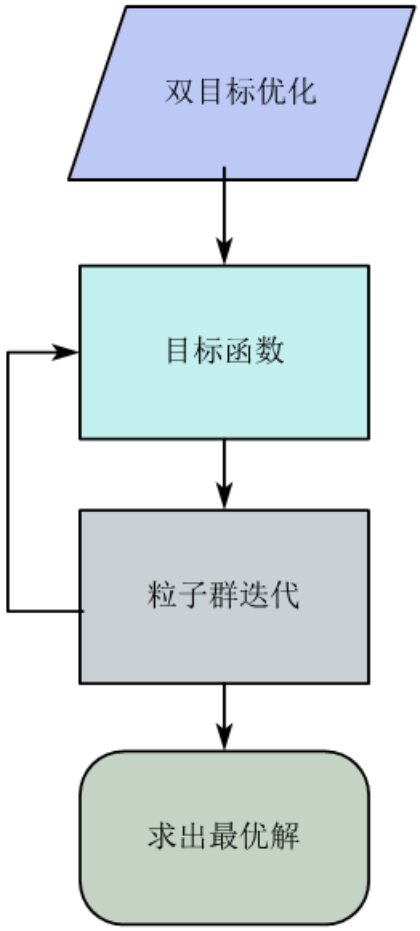

图 9 模型三求解思路

5.3.2.问题求解

在第二问的基础上进行多目标优化，由第二问傅立叶逆变换求解热传导偏微分方程可得到，

$$T(x,t)=\frac{1}{2a\sqrt{\pi t}}\exp\left\{-\frac{x^2}{4a^2t}\right\} \tag{5.14}$$

根据题目要求，可获得目标方程和约束条件

$$T_i(x,t)=\beta_i\sum_{i=}^{IV}\left(-\frac{1}{2\sqrt{ai\pi t}}\right)\times e^{\sum_{i=}^{IV}\left(-\frac{x_i^2}{4ait}\right)}+c \tag{5.15}$$

$$i=\ (I,II,III,IV)$$

$$s.t\begin{cases}T(x,t)\big|_{t=1400}<44\\ T(x,t)<44\\ x_I\epsilon(0.6,25)\ x_{II}\epsilon(0,6.5)\end{cases} \tag{5.16}$$

用 PSO 算法得到结果， 我们选择 II 层的材料厚度为 11.786mm，第IV厚度为 5.261mm。

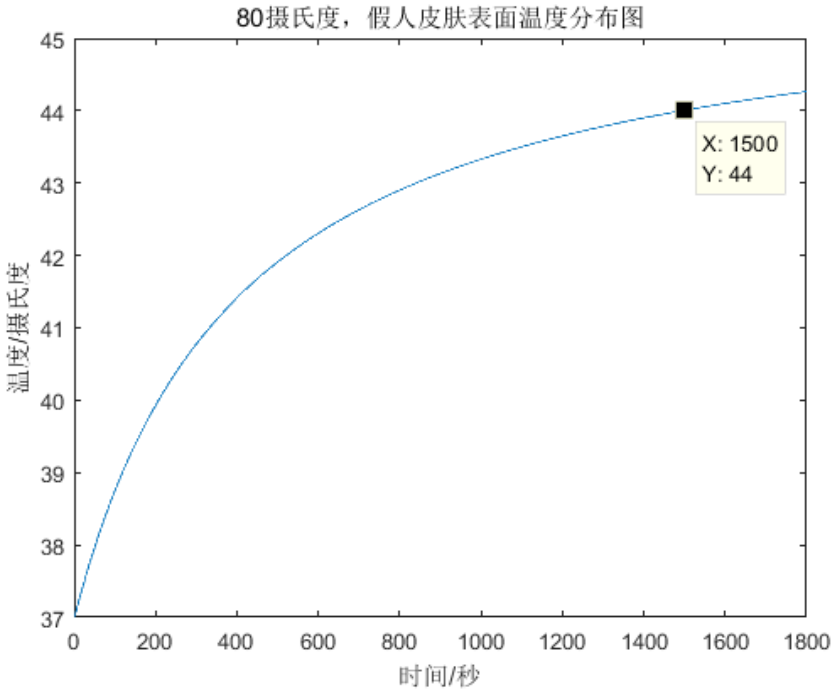

图 10 摄氏度下假人皮肤温度分布

求解发现，在 t=1500s 附近刚好为 44℃满足题目要求，说明求解情况良好。

§6 误差分析与灵敏度分析

查阅文献，该物理过程分为两阶段，第一阶段是非稳态无内热源热传导，第二阶段为稳态屋内热源热传导。现在我们通过物理计算稳态无内热源热传导每一层两侧温度来检验模型是否符合实际[5]:

$$\begin{cases} Q=\lambda_1\dfrac{t_1-t_2}{b1}=\dfrac{\Delta t_1}{b_1/(\lambda_1 A)}=\dfrac{\Delta t_1}{R_1} \\ =\lambda_2 A\dfrac{t_2-t_3}{b_2}=\dfrac{\Delta t_2}{b_2/(\lambda_2 A)}=\dfrac{\Delta t_2}{R_2} \\ =\lambda_3 A\dfrac{t_3-t_4}{b_3}=\dfrac{\Delta t_3}{b_3/(\lambda_3 A)}=\dfrac{\Delta t_3}{R_3} \end{cases} \tag{6.1}$$

将上式整理，得

$$Q=\frac{\Delta t_1+\Delta t_2+\Delta t_3}{b_1/(\lambda_1 A)+b_2/(\lambda_2 A)+b_3/(\lambda_3 A)}=\frac{\Delta t_1+\Delta t_2+t_3}{R_1+R_2+R_3}=\frac{t_1-t_4}{\sum_{i=1}^{3}R_i} \tag{6.2}$$

用稳态热传导公式，我们将结果记为下表：

层数	物理计算温度温度	热阻	每层温降	数学模拟温度
空气层	75			74.91393682
Ⅰ	73.14232294	0.007317073	1.857677064	72.99760897
Ⅱ	69.0253089	0.016216216	4.117014035	68.91110972
Ⅲ	48.71470633	0.08	20.31060257	50.84669826
Ⅳ	48.08	0.0025	0.63470633	48.71062744
合计		0.106033289	26.92	

表二 数学建模结果与物理结果对比

由上表，我们把上述得到的稳态结果与第一问的边界温度进行比较发现。在误差范围内，模型有效。

§7 模型的评价与推广

7.1 模型优缺点讨论

7.1.1 优点

Ⅰ模型使用了有限差分方法，将连续问题离散化，简化了计算量。

II 结论可靠，模型检验使用了物理仿真模拟，从数学和物理两个不同角度建立不同模型检验结论。

Ⅲ,MatLab 求解，用图片的形式表现出了温度场分布，使得 Excel 的数字可视化，避免的数据分析的抽象

7.1.2 缺点

a．论文中的模型将防火服复杂的传导过程简化成热传导过程，在实际传导中还应该包括热对流，热辐射。

b．实际中的传播并不是一维的，而是三维的，应该考虑不同方向上面的传播。

参考文献

[1]孔祥谦. 有限单元法在传热学中的应用[M]. 科学出版社:孔祥谦, 1998. 33-37

[2]曹钢. 一维热传导方程的基本解[J]. 山东轻工业学院学报, 2005, 19(4): 77-79

[3]陆君安. 偏微分方程的 MATLAB 解法[M]. 武汉大学出版社:陆君安, 2001. 147-149

[4]史策. 热传导方程有限差分法的 MATLAB 实现[J]. 咸阳师范学院学报, 2009, 24(4): 28-29

[5]谭天恩. 过程工程原理（化工原理）[M]. 化学工业出版社:谭天恩, 2004. 108-109

附录 代码

第一问

```
%chafenrechuandao1.m
%%
load('data1.mat');
hx=20;  %网格数
ht=413052270.4; %网格数
v1=zeros(ht,hx);
v1(1,:)=69.0253089; %边界值
v1(:,1)=73.14232294; %边界值
%赋最右层边值条件
for i=1:5400
v1(i,10)=data1(i);
end
%给 v2 赋值
v2=zeros(ht,hx);
v2(1,:)=v1(1,:);
v2(:,1)=v1(:,1);
a=0.082/300/1377;
r=(5400/ht)/(0.0152/hx).^2;


%%
%循环计算赋值赋值，提高计算精度
maxt=1;
t=0;
while (maxt>0.1) %精度小于 0.1,跳出循环
       for i=ht-1:-1:2
        for j=hx-1:-1:2

v1(i,j)=(v1(i,j+1)-a*r*(v1(i,j+1)-2*v1(i,j+1)+v1(i-1,j+1)))
;
            %算出 a*r 为 0.003934
            t=v2(i,j)-v1(i,j);
            maxt=0;
              if(t>maxt) %若差值大于 1，继续运算
                maxt=t;
              end
```

```
        end
      end
    v1=v2;
end
%plot
subplot(1,2,1),surf(v1)
axis([0,hx,0,ht,0,75])
% subplot(1,2,2)
% contour(v2);
% hold on;
% x=1:1:hx;
% y=1:1:ht;
% [xx,yy]=meshgrid(x,y);


%chafenrechuandao2.m
%%
load('data2.mat');
hx=20;  %网格数
ht=3117396; %网格数
v1=zeros(ht,hx);
v1(1,:)=73.14232294; %比较边界值
v1(:,1)=69.0253089; %最终边界值
%赋最右层边值条件
for i=1:5400
v1(i,10)=data1(i);
end
%给 v2 赋值
v2=zeros(ht,hx);
v2(1,:)=v1(1,:);
v2(:,1)=v1(:,1);
a=0.082/300/1377;
r=(5400/ht)/(0.0152/hx).^2;


%%
%循环计算赋值赋值，提高计算精度
maxt=1;
```

```
t=0;
while (maxt>0.1) %精度小于 0.1,跳出循环
       for i=ht-1:-1:2
        for j=hx-1:-1:2

v1(i,j)=(v1(i,j+1)-a*r*(v1(i,j+1)-2*v1(i,j+1)+v1(i-1,j+1)))
;
            %算出 a*r 为 0.003934
            t=v2(i,j)-v1(i,j);
            maxt=0;
              if(t>maxt) %若差值大于 1，继续运算
                maxt=t;
              end
         end
        end
    v1=v2;
end
%plot
subplot(1,2,1),surf(v1)
axis([0,hx,0,ht,0,75])
% subplot(1,2,2)
% contour(v2);
% hold on;
% x=1:1:hx;
% y=1:1:ht;
% [xx,yy]=meshgrid(x,y);


%chafenrechuandao3.m
%%
load('data3.mat');
hx=20;  %网格数
ht=16344885.39; %网格数
v1=zeros(ht,hx);
v1(1,:)=69.0253089; %初始边界值
v1(:,1)=48.71470633; %初始边界值
%赋最右层边值条件
for i=1:5400
```

```
v1(i,10)=data1(i);
end
%给 v2 赋值
v2=zeros(ht,hx);
v2(1,:)=v1(1,:);
v2(:,1)=v1(:,1);
a=0.082/300/1377;
r=(5400/ht)/(0.0152/hx).^2;


%%
%循环计算赋值赋值，提高计算精度
maxt=1;
t=0;
while (maxt>0.1) %精度小于 0.1,跳出循环
       for i=ht-1:-1:2
        for j=hx-1:-1:2

v1(i,j)=(v1(i,j+1)-a*r*(v1(i,j+1)-2*v1(i,j+1)+v1(i-1,j+1)))
;
            %算出 a*r 为 0.0035829
            t=v2(i,j)-v1(i,j);
            maxt=0;
              if(t>maxt) %若差值大于 1，继续运算
                maxt=t;
              end
         end
        end
    v1=v2;
end
%plot
subplot(1,2,1),surf(v1)
axis([0,hx,0,ht,0,75])
% subplot(1,2,2)
% contour(v2);
% hold on;
% x=1:1:hx;
% y=1:1:ht;
```

```
% [xx,yy]=meshgrid(x,y);

%chafenrechuandao4.m
%%
load('data4.mat');
hx=20;  %网格数
ht=12522833906; %网格数
v1=zeros(ht,hx);
v1(1,:)=48.71470633; %初始边界值
v1(:,1)=48.08; %初始边界值
%赋最右层边值条件
for i=1:5400
v1(i,10)=data1(i);
end
%给 v2 赋值
v2=zeros(ht,hx);
v2(1,:)=v1(1,:);
v2(:,1)=v1(:,1);
a=0.082/300/1377;
r=(5400/ht)/(0.0152/hx).^2;


%%
%循环计算赋值赋值，提高计算精度
maxt=1;
t=0;
while (maxt>0.1) %精度小于 0.1,跳出循环
       for i=ht-1:-1:2
        for j=hx-1:-1:2

v1(i,j)=(v1(i,j+1)-a*r*(v1(i,j+1)-2*v1(i,j+1)+v1(i-1,j+1)))
;
            %算出 a*r 为 0.0001629
            t=v2(i,j)-v1(i,j);
            maxt=0;
              if(t>maxt) %若差值大于 1，继续运算
                maxt=t;
              end
```

```
        end
       end
    v1=v2;
end
%plot
subplot(1,2,1),surf(v1)
axis([0,hx,0,ht,0,75])
% subplot(1,2,2)
% contour(v2);
% hold on;
% x=1:1:hx;
% y=1:1:ht;
% [xx,yy]=meshgrid(x,y);
```

第二问

```
clc;%清空命令行窗口
clear;%清空工作空间

%% Problem Definition

CostFunction=@(x) targetfun(x);        % Cost Function 目标函数

nVar=1;             % Number of Decision Variables 决策变量数

VarSize=[1 nVar];   % Size of Decision Variables Matrix 决策变量矩阵

VarMin=[0.0006];         % Lower Bound of Variables 变量下限
VarMax=[0.0025];         % Upper Bound of Variables 变量上限
global M1 M2 M3 Vbus Ploss PL QL Pgd

%% PSO Parameters

MaxIt=10;       % Maximum Number of Iterations 最大迭代次数
```

```
nPop=1000;          % Population Size (Swarm Size)种群数

% PSO Parameters

% If you would like to use Constriction Coefficients for PSO,
% uncomment the following block and comment the above set of
parameters.

% % Constriction Coefficients 粒子群算法的一些系数
phi1=1.05;
phi2=1.05;
phi=phi1+phi2;
chi=2/(phi-2+sqrt(phi^2-4*phi));
w=50;           % Inertia Weight
wdamp=1;         % Inertia Weight Damping Ratio
c1=chi*phi1*2;     % Personal Learning Coefficient
c2=chi*phi2*2;     % Global Learning Coefficient

% Velocity Limits
VelMax=0.0001*(VarMax-VarMin);%最大速度
VelMin=-VelMax;%最小速度

%% Initialization

GlobalBest.Cost=inf;%设初始全局最优

for i=1:nPop

       f(i,1) = randi([6,250],1,1)/10000;%生成 0.6mm 到 25mm
随机整数



     particle(i).Position=f(i,:);%将变量赋值到 particle 中

    % Initialize Velocity
    particle(i).Velocity=zeros(VarSize);%初始速度为 0
```

```
    % Evaluation
    particle(i).Cost=CostFunction(particle(i).Position);%计
算种群目标函数值

    % Update Personal Best
    particle(i).Best.Position=particle(i).Position;%更新个
体最优变量
    particle(i).Best.Cost=particle(i).Cost;

    % Update Global Best%更新全局最优变量
    if particle(i).Best.Cost<GlobalBest.Cost

        GlobalBest=particle(i).Best;

    end

end

BestCost=zeros(MaxIt,1);

%% PSO Main Loop

for it=1:MaxIt

    for i=1:nPop


            % Update Velocity 更新速度
        particle(i).Velocity = w*particle(i).Velocity ...

+c1*rand(VarSize).*(particle(i).Best.Position-particle(i).P
osition) ...

+c2*rand(VarSize).*(GlobalBest.Position-particle(i).Positio
n);

        % Apply Velocity Limits 速度是否越限
        particle(i).Velocity =
```

```
max(particle(i).Velocity,VelMin);
        particle(i).Velocity =
min(particle(i).Velocity,VelMax);

        % Update Position 更新变量
        particle(i).Position = particle(i).Position +
particle(i).Velocity;

particle(i).Position(1,:)=round( particle(i).Position(1,:))
;%变量必须是整数
        % Velocity Mirror Effect
        IsOutside=(particle(i).Position<VarMin |
particle(i).Position>VarMax);

particle(i).Velocity(IsOutside)=-particle(i).Velocity(IsOut
side);

        % Apply Position Limits 变量是否越限
        particle(i).Position =
max(particle(i).Position,VarMin);
        particle(i).Position =
min(particle(i).Position,VarMax);

        % Evaluation 计算新一代种群目标函数值
        particle(i).Cost =
CostFunction(particle(i).Position);

        % Update Personal Best 更新个体最优
        if particle(i).Cost<particle(i).Best.Cost

            particle(i).Best.Position=particle(i).Position;
            particle(i).Best.Cost=particle(i).Cost;

            % Update Global Best%更新全局最优
            if particle(i).Best.Cost<GlobalBest.Cost

                GlobalBest=particle(i).Best;
```

```
            end

        end

    end

    BestCost(it)=GlobalBest.Cost;%存储全局最优--迭代过程（方案 2）
%     F123=CostFunction(GlobalBest.Position);
%     BestCost(it)=0.7*M1+0.2*M2+0.15*M3;%方案 1
    disp(['Iteration ' num2str(it) ': Best Cost = ' num2str(BestCost(it))]);%显示

    w=w*wdamp;

end

BestSol = GlobalBest;
```

```
function F=targetfun(x)
F=999;
t=1:4400;
for ii=1:4400

T(ii)=0.988*(-1/(2*0.000044553213*(3.1416926535*t(ii))*exp(-x^2/(4*44553213*t(ii))))+45.491);
if T(ii)==44
    ii
end
if (3600-ii)>300||T(ii)>47    %定义罚函数，若超过 44 度大于 300 秒，或温度大于 47 度就执行
  F=F+0;
else
    F=T(ii);
end
```

end

第三问

```
clc;%清空命令行窗口
clear;%清空工作空间

%% Problem Definition

CostFunction=@(x) targetfun(x);        % Cost Function 目标函数

nVar=2;             % Number of Decision Variables 决策变量数

VarSize=[1 nVar];   % Size of Decision Variables Matrix 决策变量矩阵

VarMin=[0.0006 0.0006];         % Lower Bound of Variables 变量下限
VarMax=[0.0025 0.0064];         % Upper Bound of Variables 变量上限
global M1 M2 M3 Vbus Ploss PL QL Pgd

%% PSO Parameters

MaxIt=10;       % Maximum Number of Iterations 最大迭代次数

nPop=1000;        % Population Size (Swarm Size)种群数

% PSO Parameters

% If you would like to use Constriction Coefficients for PSO,
% uncomment the following block and comment the above set of parameters.
```

```
% % Constriction Coefficients 粒子群算法的一些系数
phi1=1.05;
phi2=1.05;
phi=phi1+phi2;
chi=2/(phi-2+sqrt(phi^2-4*phi));
w=50;           % Inertia Weight
wdamp=1;         % Inertia Weight Damping Ratio
c1=chi*phi1*2;    % Personal Learning Coefficient
c2=chi*phi2*2;    % Global Learning Coefficient

% Velocity Limits
VelMax=0.0001*(VarMax-VarMin);%最大速度
VelMin=-VelMax;%最小速度

%% Initialization

GlobalBest.Cost=inf;%设初始全局最优

for i=1:nPop

       f(i,1) = randi([6,250],1,1)/10000;%生成 0.6mm 到 25mm
随机整数
       f(i,1) = randi([6,64],1,1)/10000;%生成 0.6mm 到 6.4mm
随机整数



     particle(i).Position=f(i,:);%将变量赋值到 particle 中

    % Initialize Velocity
    particle(i).Velocity=zeros(VarSize);%初始速度为 0

    % Evaluation
    particle(i).Cost=CostFunction(particle(i).Position);%计
算种群目标函数值

    % Update Personal Best
    particle(i).Best.Position=particle(i).Position;%更新个
```

```
体最优变量
    particle(i).Best.Cost=particle(i).Cost;

    % Update Global Best%更新全局最优变量
    if particle(i).Best.Cost<GlobalBest.Cost

        GlobalBest=particle(i).Best;

    end

end

BestCost=zeros(MaxIt,1);

%% PSO Main Loop

for it=1:MaxIt

    for i=1:nPop


            % Update Velocity 更新速度
        particle(i).Velocity = w*particle(i).Velocity ...

+c1*rand(VarSize).*(particle(i).Best.Position-particle(i).P
osition) ...

+c2*rand(VarSize).*(GlobalBest.Position-particle(i).Positio
n);

        % Apply Velocity Limits 速度是否越限
        particle(i).Velocity =
max(particle(i).Velocity,VelMin);
        particle(i).Velocity =
min(particle(i).Velocity,VelMax);

        % Update Position 更新变量
        particle(i).Position = particle(i).Position +
```

```
particle(i).Velocity;

particle(i).Position(1,:)=round( particle(i).Position(1,:))
;%变量必须是整数
        % Velocity Mirror Effect
        IsOutside=(particle(i).Position<VarMin |
particle(i).Position>VarMax);

particle(i).Velocity(IsOutside)=-particle(i).Velocity(IsOut
side);

        % Apply Position Limits 变量是否越限
        particle(i).Position =
max(particle(i).Position,VarMin);
        particle(i).Position =
min(particle(i).Position,VarMax);

        % Evaluation 计算新一代种群目标函数值
        particle(i).Cost =
CostFunction(particle(i).Position);

        % Update Personal Best 更新个体最优
        if particle(i).Cost<particle(i).Best.Cost

            particle(i).Best.Position=particle(i).Position;
            particle(i).Best.Cost=particle(i).Cost;

            % Update Global Best%更新全局最优
            if particle(i).Best.Cost<GlobalBest.Cost

                GlobalBest=particle(i).Best;

            end

        end

    end
```

```
    BestCost(it)=GlobalBest.Cost;%存储全局最优--迭代过程（方案 2）
%      F123=CostFunction(GlobalBest.Position);
%      BestCost(it)=0.7*M1+0.2*M2+0.15*M3;%方案 1
    disp(['Iteration ' num2str(it) ': Best Cost = ' num2str(BestCost(it))]);%显示

    w=w*wdamp;

end

BestSol = GlobalBest;

function F=targetfun(x)
F=999;
t=1:4400;
for ii=1:4400

T(ii,j)=1.014*(-1/(2*0.000044553213*(3.1416926535*t(ii))*exp(-x(j)^2/(4*44553213^2*t(ii))))+45.2722);
if T(ii)==44
    ii
end
if (3600-ii)>300||T(ii)>47    %定义罚函数，若超过 44 度大于 300 秒，或温度大于 47 度就执行
  F=F+0;
else
    F=T(ii);
end
end
```