参赛队号：（221178135842177756）

2022 年第五届河北省研究生数学建模竞赛

题目　　连铸保护渣熔化析晶过程序列统计分析

摘　要

保护渣工作环境是在连铸工艺中称之为“心脏”的结晶器内，能否较好的发挥其冶金功能，主要取决于其理化性能是否合理，直接关系到铸坯表面质量及连铸工艺是否顺行。因此哪些因素能使得保护渣熔化结晶过程更好地发挥作用，是我们研究的重点。本文以图像分割与识别的技术和数字图像处理技术为基础，针对保护渣熔化结晶过程进行了量化分析和研究。主要是对保护渣熔化结晶过程对应的差异模型进行介绍。

（1）针对问题一，在对保护渣熔化结晶过程序列图像进行预处理的时候，先利用图像分割算法把图像左上角裁剪后，给出带有时间和温度（1#丝温度和2#丝温度）的子图像，再调用 python 中支持图像识别的第三方库（python + pytesseract+Tesseract-OCR）自动提取每 1 张子图像左上角的 1#丝温度，并自动填入附件 2 对应的表格。

（2）针对问题二，利用皮尔森相关系数和 P 值量化表征保护渣熔化结晶过程中相邻序列图像间的动态差异，利用低阶颜色矩提取保护渣熔化结晶过程图像的 9 维颜色特征，构建小波多子图共生矩阵提取 10 维小波特征，以及利用灰度共生矩阵的方法得到 7 个特征 4 个角度下的 28 维灰度特征，应用这些量化后的差异特征进行时间序列建模，利用 python 编程软件对上述数学模型进行保护渣熔化结晶过程的分析和研究。

（3）针对问题三，基于第二问的研究成果，利用 matlab 编程软件，构造自相关函数，利用多项式拟合获得关于温度、时间变化与保护渣熔化结晶进度之间的函数关系式。

关键词：皮尔森相关系数；灰度特征；小波特征；颜色特征；时间序列

目录

一：问题重述

为了提高板坯的表面质量，保证连铸的顺利进行，良好的连铸结晶器渣应在五个方面起到完善的作用。(1) 绝缘防止散热；(2)绝缘材料放出空气；(3)从钢水中吸收并熔化漂浮在渣界面上的夹杂物来冲洗钢水；(4)起润滑作用；(5)填充夹套与模具之间的气隙，以改善模具的传热。用于连铸的结晶器渣是在结晶器的钢水表面加入并熔化成液态渣，并渗入结晶器壁和铸锭之间[1]。同时，由于铸坯与结晶器壁的温差较大，液态渣冷却析出结晶，但无法直接观察结晶器渣在结晶器内的时间、温度和结晶量。因此，适当地对结晶温度、时间变化和保护渣熔化结晶进度之间的关系进行建模，设计出满足钢种硬化要求的结晶器流动，以满足连铸生产需要。需要解决以下几个问题：

问题一：采用图像分割与识别的技术或其他技术，自动提取每 1 张图像左上角的 1#丝温度，并将其自动导入附件 2 中对应表格里，并制作温度与时间关系曲线图。

问题二：根据图 2 中 6 个节点图像，采用数字图像处理技术，研究保护渣熔化结晶过程中相邻序列图像间的动态差异，并量化表征这些差异。在此基础上，研究应用这些量化后的差异特征进行时间序列建模，获取保护渣熔化结晶进度曲线。

问题三：考虑温度、时间变化，结合第二问的研究成果，构建数学模型， 讨论温度、时间变化与保护渣熔化结晶进度之间的函数关系。

二：问题分析

2.1 问题一分析

先利用图像分割算法把图像左上角裁剪后，再调用 python 中支持图像识别的第三方库（python + pytesseract+Tesseract-OCR）自动识别并提取每 1 张图像左上角的文字部分，给出温度和时间的变化关系图。

2.2 问题二分析

基于问题一的研究，发现温度和时间等因素对保护渣熔化结晶过程有影响。本问题利用皮尔森相关系数（PCC）和 P 值等相似性指标量化表征保护渣熔化结晶过程中相邻序列图像间的动态差异，根据保护渣熔化结晶过程图像提取特征（颜色特征、小波特征、灰度特征），应用这些量化后的差异特征进行时间序列建模分析。

2.3 问题三分析

结合问题二时间序列建模获得的保护渣熔化结晶进度曲线，建立三维数学模型，研究温度、时间变化与保护渣熔化结晶进度之间的关系。

三：模型假设及相关定义

3.1 模型假设

1.脱碳后的保护渣进行研磨过程中，忽略汗渍和水汽的影响。

2.设备本身有自主知识产权的隐私保护，无法自动获取每 1 张图像对应的时刻与温度，只能实验结束后，靠肉眼识别关键节点图像，忽略肉眼识别带来的误差。

3.2 符号说明

$P_{i,j}$	保护渣熔化结晶过程图像第 i 个颜色通道分量中灰度为 j 的像素出现的概率
N	保护渣熔化结晶过程图像中的像素个数
GLCM	灰度共生矩阵
WMCM	小波多子图共生矩阵
2D-WPT	二维小波包变换
HH、LH、HL、LL	高频噪声子图、垂直细节子图、水平细节子图、近似子图

四：模型的建立与求解

4.1 问题一模型建立与求解

4.1.1 数据预处理

保护渣熔化结晶过程图像在计算机中的存储方式为矩阵。将图像矩阵读入计算机中，利用图像分割算法，按照横坐标为 20-60，纵坐标为 50-100 的标准将图像左上角带有文字的子图像识别并裁剪出。裁剪出的子图像如图 1 所示：（以第 110 秒为示例）

图 1 第 110 秒裁剪出的子图像

4.1.2 模型构建与求解

将图像分割算法裁剪出的子图像调用到 python 中支持图像识别的第三方库（python + pytesseract+Tesseract-OCR），自动识别并提取每 1 张子图像左上角的

1#丝温度，并将其自动导入到附件 2 中的表格中，部分时刻下的 1#丝温度如图 2 所示。具体程序见附录 1。

序号	时间	1#丝温度
1	110	900
2	111	904
3	112	910
4	113	914
5	114	918
6	115	923
7	116	927
8	117	932
9	118	935
10	119	938
11	120	942
12	121	947
13	122	951
14	123	953
15	124	959
16	125	965
17	126	970
18	127	974
19	128	979
20	129	987

图 2 部分时刻下 1#丝温度

由附件 2 中的表格可知：1#丝温度呈现升高--平稳--下降的趋势，如图 3 所示：随着时间的推进，试样开始熔化（110 秒），1#丝温度逐渐升高，直到达到一定温度（1500）饱和时，温度开始呈现下降趋势直至结束结晶（671 秒）。试样开始熔化时的温度高于结束结晶时的温度；而 1#丝温度和 2#丝温度的平均温度和时间的关系如图 4 所示：随着时间的推进，试样开始熔化（110 秒），平均温度逐渐升高，呈现波纹式前进，直到达到一定温度（1300）饱和时，温度开始呈现下降趋势直至结束结晶（671 秒）。整体温度呈现升高--平稳--下降的趋势，试样开始熔化时的温度同样高于结束结晶时的温度。由此可见，1#丝温度和平均温度与时间的关系图整体趋势一致。

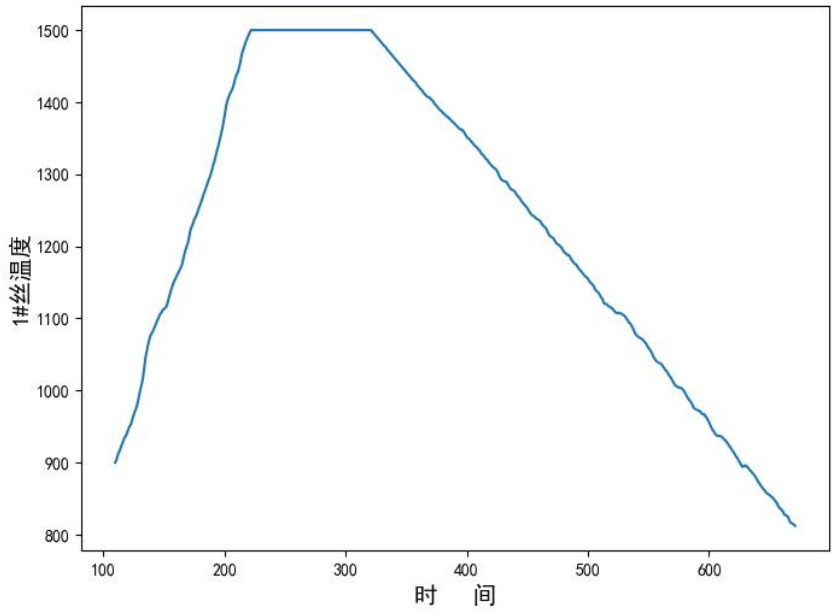

图 3 1#丝温度与时间的关系图

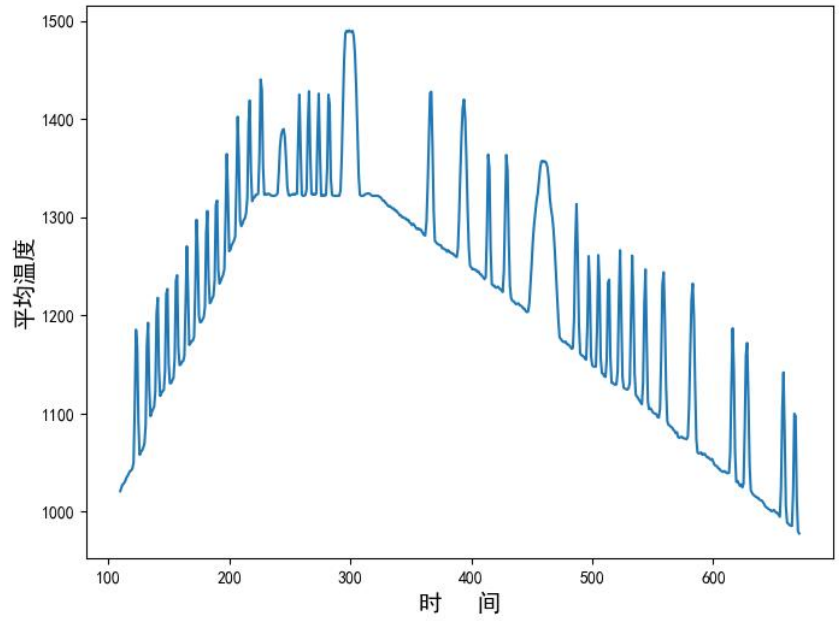

图 4 平均温度与时间的关系图

4.2 问题二模型建立与求解

4.2.1 提取颜色特征

颜色是图像分析的常用特征。颜色矩是一种简单有效的颜色特征表示方法，有一阶矩(均值)、二阶矩(方差)和三阶矩(斜度)等。由于颜色信息主要分布于低阶矩中，所以用低阶矩足以表达图像的颜色分布，且不需要颜色空间量化，特征维数低[2]。本文采用 R、G、B 三个颜色分量的一、二和三阶矩来表示保护渣熔化结晶过程图像的颜色分布，三个颜色矩的定义为：

$$\mu_i = \frac{1}{N}\sum_{j=1}^{N} p_{i,j} \qquad (1)$$

$$\xi_i = \sqrt{\frac{1}{N}\sum_{j=1}^{N}(p_{i,j} - \mu_i)^2} \quad (2)$$

$$S_i = \sqrt[3]{\frac{1}{N}\sum_{j=1}^{N}(p_{i,j} - \mu_i)^3} \quad (3)$$

图像的 3 个分量的前三阶颜色矩组成一个 9 维向量，即得到 9 个颜色特征。

4.2.2 提取小波特征

小波分解、小波（包）变换和灰度共生矩阵(GLCM)是提取图像数据纹理特征的常用工具。小波分解的多分辨率特征使得其能够提取图像频域内的纹理信息。小波变换能够很好地表征一大类以低频（近似）信息为主要成分的信号，但它不能很好地分解和表示包含大量高频（细节）信息（如细小边缘或纹理）的信号，而小波包变换可以对高频信息提供更精细的分解，对包含大量中、高频信息的信号能够进行更好的时频局部化分析[3]。灰度共生矩阵(GLCM) 是一种基于空间的二维方法，从图像强度直方图中提取特征，并且提供随机图像位置的灰度频率，能反映图像灰度关于方向、相邻间隔、变化幅度等方面的综合信息，提取了一些人眼可能看不到的特征[4]。Ma 等人将小波变换和 GLCM 相结合，创建小波多子图共生矩阵(WMCM)，提取超声图像的多尺度多子图纹理特征，用于研究肝纤维化分期[5]。Lizhen 等人融合离散小波分解和灰度共生矩阵提取 X 线图像的纹理特征，用于早期乳腺癌检测[6]。受这些工作的启发，本文采用 WMCM 提取保护渣熔化结晶过程图像的纹理特征，将二维小波包变换(2D-WPT)和灰度共生矩阵(GLCM)相结合，实现保护渣熔化结晶过程图像频域与空间域的纹理特征融合。为计算简便，同时兼顾保护渣熔化结晶过程图像的纹理周期分布特点，本文仅提取了单尺度(空间距离 d=1)单角度(方向$\theta = 0^\circ$)的纹理特征，具体过程如下。

第一步：创建 WMCM

（i）将预处理后的子图像进行一级 2D-WPT 分解，得到 4 个 256*256 像素的子图，即高频噪声子图(High-High, *HH*)、垂直细节子图(Low-High, *LH*)、水平细节子图(High-Low, *HL*)和近似子图(Low-Low, *LL*)。

（ii）计算整体细节子图 *LHL*，其元素 $LHL(i,j) = \sqrt{\left[LH(i,j)\right]^2 + \left[HL(i,j)\right]^2}$；

（iii）将近似子图 *LL* 和整体细节子图 *LHL* 量化到 16 个灰度级，定义 WMCM 的元素 $q(i,j)$ 为 *LL* 和 *LHL* 中同时满足如下条件的像素 (s,t) 的个数：$LL(s,t) = i$，$LHL(s,t) = j$；

（iv）将 WMCM 归一化，即 $q(i,j) = \dfrac{q(i,j)}{\sum_{i=1}^{16}\sum_{j=1}^{16} q(i,j)}$

第二步：提取 WMCM 特征

令 $q_x(i)=\sum_{j=1}^{16}q(i,j)$，$q_y(j)=\sum_{i=1}^{16}q(i,j)$。 定义如下 10 个特征：

(1) 小灰度小细节优势(Small Gray-level Small Detail Advantage, SGSDA)，其值用 w_1 表示，

$$w_1=\sum_{i=1}^{16}\sum_{j=1}^{16}\frac{q(i,j)}{i^2j^2} \tag{4}$$

该特征度量保护渣熔化结晶过程图像的明暗程度及纹理的粗细程度。其值越大，图像越暗，越平滑。

(2) 小灰度大细节优势(Small Gray-level Big Detail Advantage, SGBDA)，其值用 w_2 表示，

$$w_2=\sum_{i=1}^{16}\sum_{j=1}^{16}\frac{q(i,j)j^2}{i^2} \tag{5}$$

该特征度量保护渣熔化结晶过程图像的明暗程度及纹理的粗细程度。

(3) 灰度均值(Gray Level Average, GLA)，其值用 w_3 表示，

$$w_3=\sum_{i=1}^{16}iq_x(i) \tag{6}$$

该特征度量保护渣熔化结晶过程图像整体的明暗程度。其值越大，图像整体越暗。

(4) 细节均值(Detail Level Average, DLA)，其值用 w_4 表示，

$$w_4=\sum_{j=1}^{16}jq_y(j) \tag{7}$$

该特征度量保护渣熔化结晶过程图像整体纹理的粗细程度。其值越大，图像整体纹理越粗。

(5) 灰度标准差(Gray Level Mean Square Error, GLMSE)，其值用 w_5 表示，

$$w_5=\sqrt{\sum_{i=1}^{16}(i-w_3)^2q_x(i)} \tag{8}$$

该特征度量保护渣熔化结晶过程图像灰度的离散程度。其值越大，图像灰度的离散程度越大。

(6) 细节标准差(Detail Level Mean Square Error, DLMSE)，其值用 w_6 表示，

$$w_6=\sqrt{\sum_{j=1}^{16}(j-w_4)^2q_y(j)} \tag{9}$$

该特征度量保护渣熔化结晶过程图像纹理细节的离散程度。其值越大，图像纹理细节的离散程度越大。

(7) 规则度(Regulation)，其值用 w_8 表示，

$$w_8 = \sum_{i=1}^{16}\sum_{j=1}^{16} iq(i,j) \tag{10}$$

该特征度量保护渣熔化结晶过程图像纹理的规则程度。其值越大，图像纹理越规则。

(8) 对比度(Contrast)，其值用 w_9 表示，

$$w_9 = \sum_{i=1}^{16}\sum_{j=1}^{16} (i-j)^2 q(i,j) \tag{11}$$

该特征度量 WMCM 的值的分布以及图像中的局部变化，反映了图像的清晰度和纹理的沟纹深浅。对比度大，则纹理的沟纹深，效果清晰；反之，则沟纹浅，效果模糊。

(9) 逆差分矩(Inverse Difference Moment, WIDM)，其值用 w_{10} 表示，

$$w_{10} = \sum_{i=1}^{16}\sum_{j=1}^{16} \frac{q(i,j)}{(i-j)^2+1} \tag{12}$$

该度量反映了图像纹理的清晰程度和规则程度，其值越大，纹理越清晰、规律性越强。

(10) 熵(Entropy)，其值用 w_{11} 表示，

$$w_{11} = -\sum_{i=1}^{16}\sum_{j=1}^{16} q(i,j)\log(q(i,j)) \tag{13}$$

该度量反映了图像灰度分布的复杂程度，其值越大，图像越复杂。

4.2.3 提取灰度特征

GLCM 能够反映图像灰度关于方向、相邻间隔、变化幅度的综合信息，是分析图像的局部模式及其排列规则的基础[7]。通过统计计算图像上具有特定灰度和一定距离的两个像素的状态确定 GLCM，像素对的“偏移”取自四个不同的方向(0°，45°，90°，135°)，空间距离 d=1，进而计算基于 GLCM 的 7 个统计量，即角二阶矩、对比度、相关性、相异度、熵、同质性、能量（见公式(14)－(20)）。这 7 个统计量在四个方向上的均值和方差构成了图像的 28 维纹理特征，记为灰度特征。

$$\mathrm{ASM} = \sum_i\sum_j G(i,j)^2 \tag{14}$$

$$\mathrm{Contrast} = \sum_i\sum_j (i-j)^2 G(i,j) \tag{15}$$

$$\mathrm{Correlation} = \sum_i\sum_j \frac{(i-\mu_x)(j-\mu_y)G(i,j)}{\sigma_x\sigma_y} \tag{16}$$

$$\mathrm{Dissimilarity} = \sum_i\sum_j |i-j| G(i,j) \tag{17}$$

$$\text{Entropy} = -\sum_i \sum_j G(i,j)\log\left(G(i,j)\right) \tag{18}$$

$$\text{Homogeneity} = \sum_i \sum_j \frac{G(i,j)}{1+|i-j|^2} \tag{19}$$

$$\text{Energy} = \sqrt{\sum_i \sum_j G(i,j)^2} \tag{20}$$

其中 $G(i,j)$ 为灰度共生矩阵的元素值，

$$\mu_x = \sum_i \sum_j i \cdot G(i,j), \qquad \mu_y = \sum_i \sum_j j \cdot G(i,j), \qquad \sigma_x = \sum_i \sum_j (i-\mu_x)^2 \cdot G(i,j),$$

$$\sigma_y = \sum_i \sum_j \left(j-\mu_y\right)^2 \cdot G(i,j).$$

4.2.4 模型求解

利用低阶颜色矩提取保护渣熔化结晶过程图像的 9 维颜色特征，如图 5 所示；构建小波多子图共生矩阵提取 10 维小波特征，如图 6 所示；以及利用灰度共生矩阵得到 7 个特征 4 个角度下的 28 维灰度特征，如图 7 所示。根据提取出来的三组特征，利用箱线图量化表征保护渣熔化结晶过程中相邻序列图像间的差异。

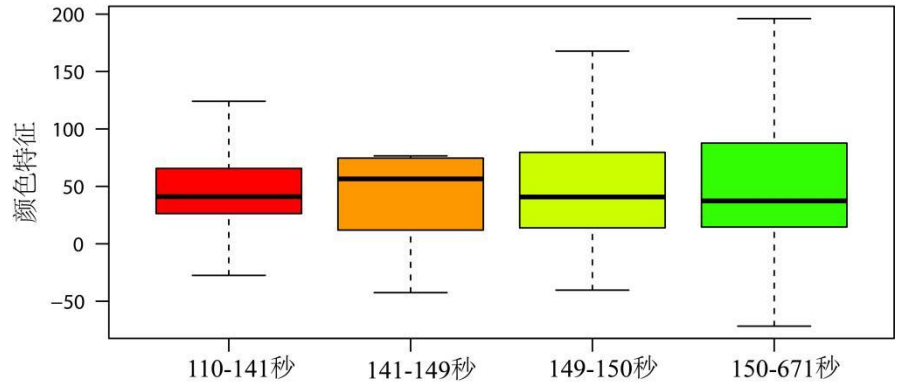

图 5 保护渣熔化结晶过程相邻序列图像的颜色特征

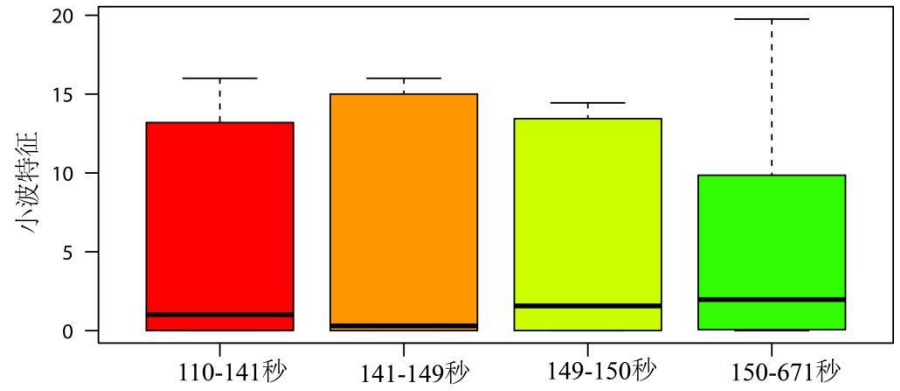

图 6 保护渣熔化结晶过程相邻序列图像的小波特征

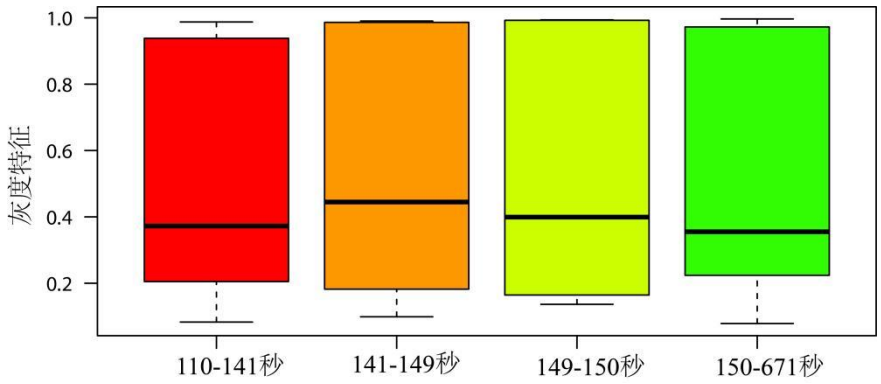

图 7 保护渣熔化结晶过程相邻序列图像的灰度特征

由图 5、7 可直观看出保护渣熔化结晶过程的颜色特征和灰度特征的趋势为升高--平稳--下降，在试样完全熔化到试样熔清（141-149 秒）时达到饱和，这与温度和时间的关系图趋势保持一致。由图 6 可知小波特征的波动幅度较为平缓。

另外，利用提取的三组特征，计算保护渣熔化结晶过程中相邻序列图像间的相似性评价指标 PCC 和 P 值。PCC 值越大，说明保护渣熔化结晶过程中相邻序列图像间的差异越小；P 值越大，说明保护渣熔化结晶过程中相邻序列图像间的差异越大。保护渣熔化结晶过程相邻序列间的差异曲线分别如图 8、9 所示，图中标红的点为保护渣熔化结晶过程的 6 个节点。

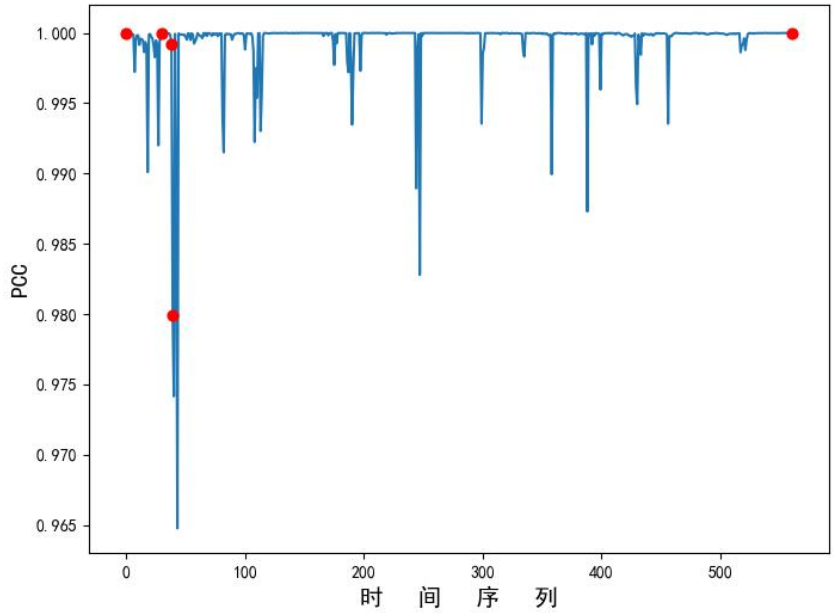

图 8 以 PCC 为衡量指标的保护渣熔化结晶过程差异曲线

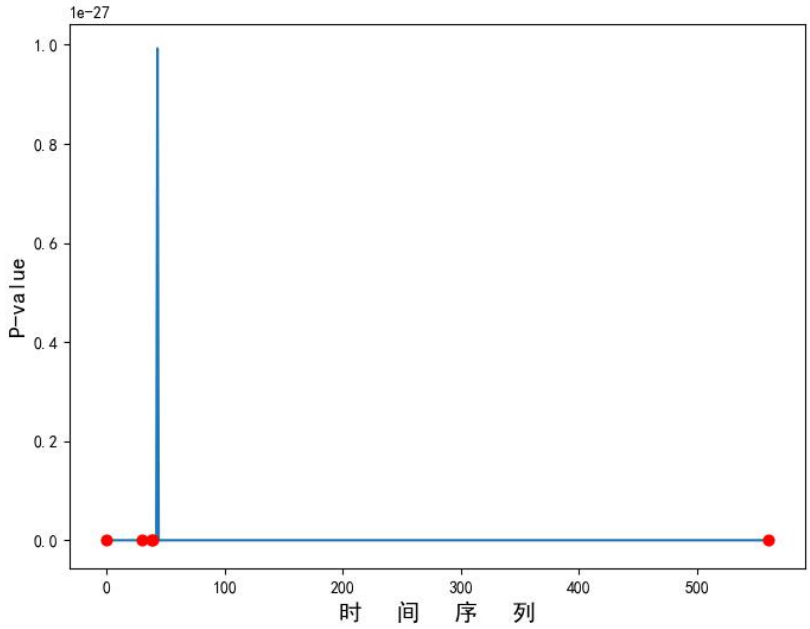

图 9 以 P 值为衡量指标的保护渣熔化结晶过程差异曲线

由图 8、9 可知：无论是以 PCC，还是 P 值作为评价指标，相邻序列保护渣熔化结晶过程差异曲线整体趋势保持一致，均在节点处量化的特征基本没有差异。在保护渣熔化结晶过程中，随着试样开始熔化，直至结束结晶，每个节点处的特征差异最小。

在此基础上，通过自相关系数构造自相关函数（ACF），利用平衡性来检验提取的 47 个特征中有 30 个特征可以建立时间序列模型。这里选取图像颜色特征里的 R-一阶矩(均值)特征作为代表来构建时间序列模型。R-一阶矩(均值)的自相关函数（ACF）如图 10 所示。

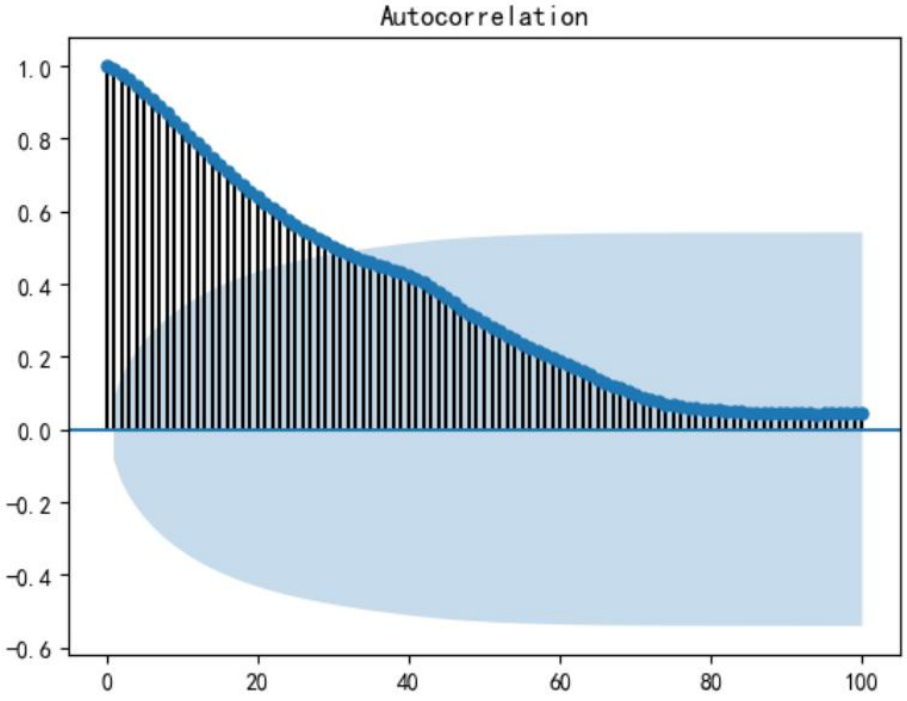

图 10 R-一阶矩(均值)的自相关函数（ACF）

根据 R-一阶矩(均值)的自相关函数（ACF）做多项式拟合构建时间序列模型获得多项式拟合曲线如图 11 所示，然后通过归一化后得到保护渣熔化结晶进度曲线如图 12 所示。

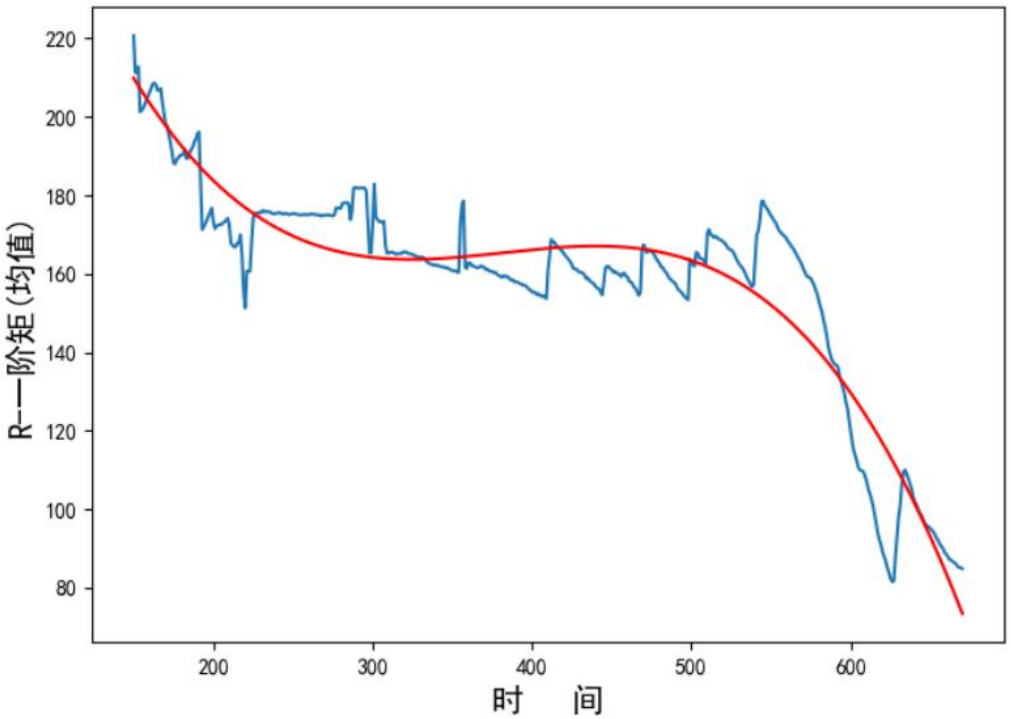

图 11 多项式拟合曲线

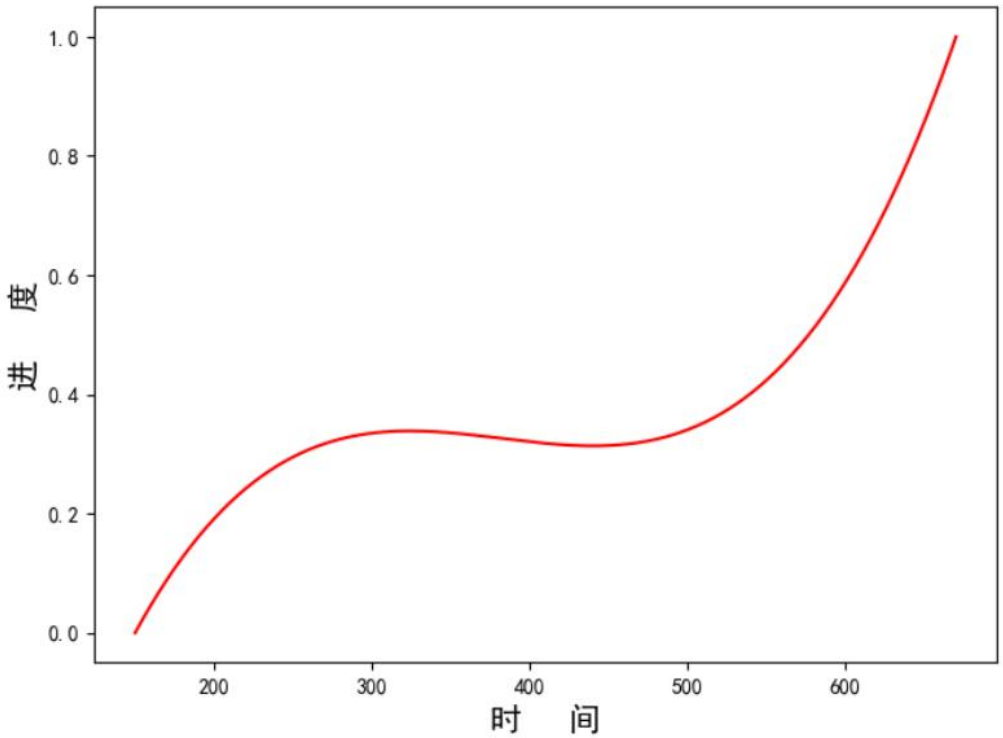

图 12 保护渣熔化结晶进度曲线

4.3 问题三模型建立与求解

考虑温度、时间变化，结合第二问的研究成果，构建数学回归模型，讨论温度、时间变化与保护渣熔化结晶进度之间的函数关系。

根据 R-一阶矩(均值)特征的保护渣熔化结晶进度曲线做多项式拟合，得到1#丝温度 y、时间变化 x 与保护渣熔化结晶进度之间的函数 $f_1(x,y)$ 以及 2#丝温度 y、时间变化 x 与保护渣熔化结晶进度之间的函数 $f_2(x,y)$，并根据函数 $f_1(x,y)$ 和 $f_2(x,y)$ 画出其图像，如图 12 和图 13 所示：

$$f_1(x,y)=p_{00}+p_{10}x+p_{01}y+p_{20}x^2+p_{11}xy+p_{02}y^2+p_{30}x^3+p_{21}x^2y+p_{12}xy^2+p_{03}y^3 \quad (21)$$

其中，

$p_{00}=-6.386e-13$，$p_{10}=0.004864$，$p_{01}=1.32e-15$，$p_{20}=-2.236e-05$，
$p_{11}=1.822e-18$，$p_{02}=-8.844e-19$，$p_{30}=3.212e-08$，$p_{21}=-2.835e-21$，
$p_{12}=-4.154e-22$，　$p_{03}=1.932e-22$.

$$f_2(x,y)=p_{00}+p_{10}x+p_{01}y+p_{20}x^2+p_{11}xy+p_{02}y^2+p_{30}x^3+p_{21}x^2y+p_{12}xy^2+p_{03}y^3 \quad (22)$$

$p_{00}=3.098e-14$，　$p_{10}=0.004864$，　$p_{01}=-8.312e-17$，　$p_{20}=-2.236e-05$，
$p_{11}=-7.495e-20$，　$p_{02}=7.554e-20$，　$p_{30}=3.212e-08$，　$p_{21}=-4.63e-23$，
$p_{12}=-3.837e-23$，　$p_{03}=-2.283e-23$.

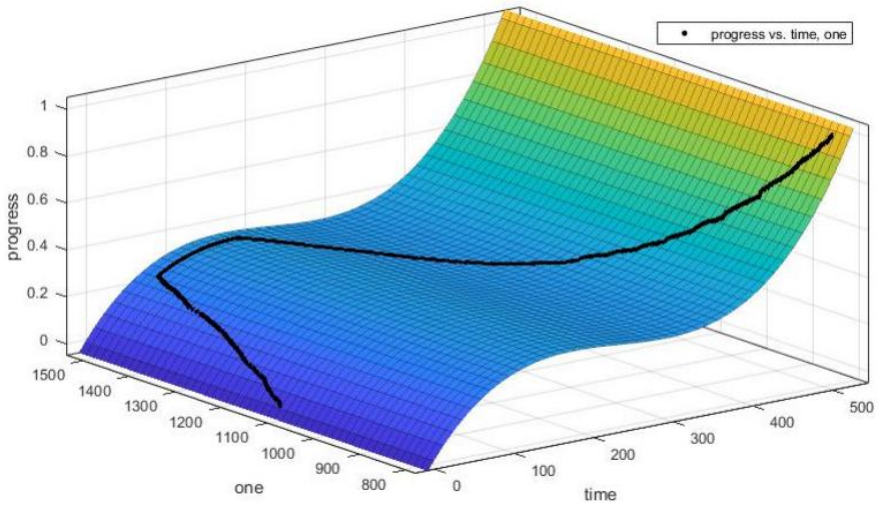

图 12 1#丝温度、时间变化与保护渣熔化结晶进度图

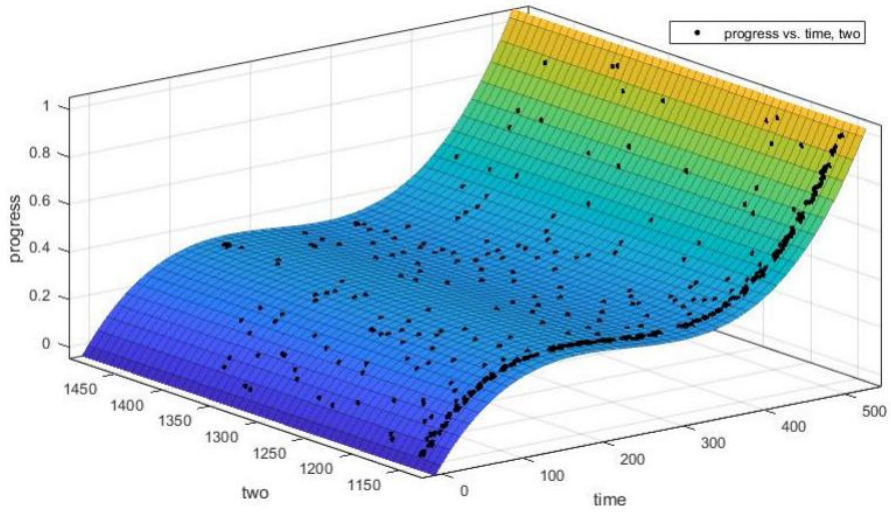

图 13 2#丝温度、时间变化与保护渣熔化结晶进度图

五：模型的评价

1. 由于脱碳后的保护渣进行研磨过程中空气比较复杂，是否有水汽，不能给出明确的定义，就会影响保护渣熔化结晶过程的时间节点以及进程，比较复杂。对此，我们认为空气内部没有水汽，没有客观全面地看待问题。

2. 设备本身有自主知识产权的隐私保护，无法自动获取每 1 张图像对应的时刻与温度，只能实验结束后，靠肉眼识别关键节点图像，忽略肉眼识别带来的误差。

六：参考文献

[1] Jung Wook Cho, Hiroyuki Shibata. Effects of Solidiffication of Mold Fluxes on the Heat Transfer in Casting Mold[J]. Journal of Non-Crystalline Solids, 2002, 282(1):110-117.

[2] Stricker M A, Orengo M. In *Similarity of color images*, Storage and retrieval for image and video databases III, International Society for Optics and Photonics: 1995; pp 381-392.

[3] Yoshida H, Casalino D D, Keserci B, et al. Wavelet-packet-based texture analysis for differentiation between benign and malignant liver tumours in ultrasound images. *Physics in Medicine & Biology* 2003, 48 (22): 3735.

[4] M., Hall-Beyer *GLCM Texture: A Tutorial v. 3.0 March 2017*. 2017.

[5] 马蕊香 基于超声图像的肝纤维化分期研究. *哈尔滨工业大学* 2016.

[6] Shen L, He M, Shen N, et al. Optimal breast tumor diagnosis using discrete wavelet transform and deep belief network based on improved sunflower optimization method. *Biomedical Signal Processing and Control* 2020, 60:101953.

[7] Gómez W, Pereira W C A, Infantosi A F C. Analysis of co-occurrence texture stat istics as a function of gray-level quantization for classifying breast ultrasound[J]. IEE E transactions on medical imaging, 2012, 31(10): 1889-1899.

七：附录

1.问题一：裁剪图像、自动识别并提取每 1 张子图像左上角的 1#丝温度，并将其自动导入到附件 2 中的表格中。

```
from PIL import Image
import pytesseract
import numpy as np
import pandas as pd
import cv2
import re
import os

img_path = 'D:\\python code\\Dti'
list = os.listdir(img_path)
si = []
for ls in list:
        iimg = os.path.join(img_path, ls)
        img = cv2.imread(iimg)
        cropped = img[20:60, 50:100]
        cv2.imwrite(r"D:\\python code\\Dtifen\\{}".format(ls), cr
opped)

img_path = 'D:\\python code\\Dtifen'
list = os.listdir(img_path)
si = []
for ls in list:
        iimg = os.path.join(img_path, ls)
        pytesseract.pytesseract.tesseract_cmd = r'D:\\Program File
s\\Tesseract-OCR\\tesseract.exe'
        image = Image.open(iimg)
        text = pytesseract.image_to_string(image)
        wendu = [float(s) for s in re.findall(r'-?\d+\.?\d*',
text)]
        si.append(wendu)
```

```
time  =  pd.DataFrame(si)
time.to_csv('D:\\python  code\\time.csv')
```

2.问题二：提取颜色、小波、灰度特征，求出 PCC 和 P 值与时间序列的关系，通过自相关系数构造自相关函数（ACF）进而做多项式拟合。

```
#颜色
import the necessary packages
import numpy as np
import cv2

def color_moments(filename):
    img = cv2.imread(filename)
    if img is None:
        return
    # Convert BGR to HSV colorspace
    hsv = cv2.cvtColor(img, cv2.COLOR_BGR2HSV)
    # Split the channels - h,s,v
    h, s, v = cv2.split(hsv)
    # Initialize the color feature
    color_feature = []
    # N = h.shape[0] * h.shape[1]
    # The first central moment - average
    h_mean = np.mean(h)  # np.sum(h)/float(N)
    s_mean = np.mean(s)  # np.sum(s)/float(N)
    v_mean = np.mean(v)  # np.sum(v)/float(N)
    color_feature.extend([h_mean, s_mean, v_mean])
    # The second central moment - standard deviation
    h_std = np.std(h)  # np.sqrt(np.mean(abs(h - h.mean())**2))
    s_std = np.std(s)  # np.sqrt(np.mean(abs(s - s.mean())**2))
    v_std = np.std(v)  # np.sqrt(np.mean(abs(v - v.mean())**2))
    color_feature.extend([h_std, s_std, v_std])
    # The third central moment - the third root of the skewness
    h_skewness = np.mean(abs(h - h.mean())**3)
    s_skewness = np.mean(abs(s - s.mean())**3)
    v_skewness = np.mean(abs(v - v.mean())**3)
    h_thirdMoment = h_skewness**(1./3)
```

```
    s_thirdMoment = s_skewness**(1./3)
    v_thirdMoment = v_skewness**(1./3)
    color_feature.extend([h_thirdMoment, s_thirdMoment,
v_thirdMoment])
    return color_feature

    # 小波
    sampling_rate = 1024
    t = np.arange(0, 1.0, 1.0 / sampling_rate)
    f1 = 100
    f2 = 200
    f3 = 300
    f4 = 400
    data = np.piecewise(t, [t < 1, t < 0.8, t < 0.5, t < 0.3],
                        [lambda t: 400*np.sin(2 * np.pi * f4 * t),
                         lambda t: 300*np.sin(2 * np.pi * f3 * t),
                         lambda t: 200*np.sin(2 * np.pi * f2 * t),
                         lambda t: 100*np.sin(2 * np.pi * f1 * t)])
    wavename = 'cgau8'
    totalscal = 256
    fc = pywt.central_frequency(wavename)
    cparam = 2 * fc * totalscal
    scales = cparam / np.arange(totalscal, 1, -1)
    [cwtmatr, frequencies] = pywt.cwt(data, scales, wavename, 1.0 /
sampling_rate)
    plt.figure(figsize=(8, 4))
    plt.subplot(211)
    plt.plot(t, data)
    plt.xlabel("t(s)")
    plt.title('shipinpu',  fontsize=20)
    plt.subplot(212)
    plt.contourf(t, frequencies, abs(cwtmatr))
    plt.ylabel(u"prinv(Hz)")
    plt.xlabel(u"t(s)")
    plt.subplots_adjust(hspace=0.4)
```

```
    plt.show()
    import pywt
    import matplotlib.pyplot as plt
    import numpy as np

    fs = 1000
    N = 200
    k = np.arange(200)
    frq = k*fs/N
    frq1 = frq[range(int(N/2))]

    aa = []
    for i in range(200):
        aa.append(np.sin(0.3*np.pi*i))
    for i in range(200):
        aa.append(np.sin(0.13*np.pi*i))
    for i in range(200):
        aa.append(np.sin(0.05*np.pi*i))
    y = aa

    wavename = 'db5'
    cA, cD = pywt.dwt(y, wavename)
    ya = pywt.idwt(cA, None, wavename, 'smooth')  # approximated
component
    yd = pywt.idwt(None, cD, wavename, 'smooth')  # detailed component
    x = range(len(y))
    plt.figure(figsize=(12, 9))
    plt.subplot(311)
    plt.plot(x, y)
    plt.title('original signal')
    plt.subplot(312)
    plt.plot(x, ya)
    plt.title('approximated component')
    plt.subplot(313)
    plt.plot(x, yd)
```

```
    plt.title('detailed component')
    plt.tight_layout()
    plt.show()


    # 图像单边谱
    plt.figure(figsize=(12, 9))
    plt.subplot(311)
    data_f = abs(np.fft.fft(cA))/N
    data_f1 = data_f[range(int(N/2))]
    plt.plot(frq1, data_f1, 'red')

    plt.subplot(312)
    data_ff = abs(np.fft.fft(cD))/N
    data_f2 = data_ff[range(int(N/2))]
    plt.plot(frq1, data_f2, 'k')



    plt.xlabel('pinlv(hz)')
    plt.ylabel('amplitude')

    plt.show()
#灰度
import cv2
import numpy as np

def gray_features(img):
      hist = cv2.calcHist([img],[0],None,[256],[0,255])#得到全局直方图
统计数据
      h,w = img.shape
      hist = hist/(h*w)#将直方图归一化为 0-1，概率的形式

      grayFeature = []
      #灰度平均值
      mean_gray = 0
```

```
    for i in range(len(hist)):
        mean_gray += i*hist[i]
    grayFeature.append(mean_gray[0])

    #灰度方差
    var_gray = 0
    for i in range(len(hist)):
        var_gray += hist[i]*((i - mean_gray)**2)
    grayFeature.append(var_gray[0])

    #能量 sum(hist[i])**2
    ##归一化
    max_ = np.max(hist)
    min_ = np.min(hist)
    histOne = (hist - min_)/(max_ - min_)
    ##求解能量
    energy = 0
    for i in range(len(histOne)):
        energy += histOne[i]**2
    grayFeature.append(energy[0])

    #熵
    he = 0
    for i in range(len(hist)):
        if hist[i] != 0:#当等于 0 时，log 无法进行计算，因此只需要计算非 0 部分的熵即可
            he += hist[i]*(np.log(hist[i])/(np.log(2)))
    he = -he
    grayFeature.append(he[0])#因为返回的是含有一个元素的数组,所以通过取值操作将其取出来再加入到列表中去

    #灰度对比度
    con = np.max(img)-np.min(img)
    grayFeature.append(con)
    return grayFeature
```

```
for i in range(1,101):
      img =
cv2.imread(r'E:\...\PYTHON\vvision_detection\bpPackage\PositiveImg\Ob
jImg{}.jpg'.format(i),0)
      img1 =
cv2.imread(r'E:\...\PYTHON\vvision_detection\bpPackage\NegativeImg\No
ObjImg{}.jpg'.format(i),0)
      grayFeas = gray_features(img)
      grayFeas1 = gray_features(img1)
      print(grayFeas)
      print(grayFeas1)
      print("-------------------------------------------------------
------------------------------")
```

3.问题三：根据第二问的研究成果，利用 matlab 编程软件的工具箱做多项式拟合构造温度、时间变化与保护渣熔化结晶进度之间的函数。

4.图像的代码

```
import pandas as pd
import numpy as np
import matplotlib
import matplotlib.pyplot as plt

matplotlib.rcParams['font.sans-serif']=['SimHei']
matplotlib.rcParams['axes.unicode_minus']=False

data = pd.read_excel('附件 2.xlsx',index_col='序号')

X=data['时间']
Y1=data['1#丝温度']
Y2=data['2#丝温度']

import numpy as np
import pandas as pd
from scipy.stats import pearsonr
import os
```

```
import matplotlib.pyplot as plt
import matplotlib
from seaborn import tsplot
matplotlib.rcParams['font.sans-serif']=['SimHei']
matplotlib.rcParams['axes.unicode_minus']=False

data = pd.read_csv('D:\\python code\\Dfeture\\feture.csv',
index_col='Unnamed: 0')
# wendu = pd.read_excel('附件 2.xlsx',index_col='序号')

# X = wendu['时间']
# Y1 = wendu['1#丝温度']
# Y2 = wendu['2#丝温度']
# plt.figure(figsize=(10, 4))
# plt.plot(data.iloc[40:-1, 28])
# plt.show()

python 的多项式拟合
from sklearn.linear_model import LinearRegression
from sklearn.pipeline import Pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import PolynomialFeatures

poly_reg = Pipeline([
    ("poly", PolynomialFeatures(degree=3)),
    ("std_sclar", StandardScaler()),
    ("lin_reg", LinearRegression())
])
x = np.array([i for i in range(150, 671, 1)])
X = x.reshape(-1, 1)
y = np.array(data.iloc[40:-1, 28])

poly_reg.fit(X, y)
y_predict = poly_reg.predict(X)
plt.figure(figsize=(10, 4))
```

```
plt.plot(x, y)
plt.plot(np.sort(x), y_predict[np.argsort(x)], color='r')
plt.figure(figsize=(10, 4))
plt.plot(np.sort(x),
1-(y_predict[np.argsort(x)]-np.min(y_predict))/(np.max(y_predict)-np.
min(y_predict)), color='r')
plt.xlabel('时　间', fontsize=15)
plt.ylabel('进　度', fontsize=15)
plt.show()
y_pre=1-(y_predict[np.argsort(x)]-np.min(y_predict))/(np.max(y_predic
t)-np.min(y_predict))
pd.DataFrame(y_pre).to_csv('进度曲线数据.csv')

##ACF 和 PACF
from statsmodels.graphics.tsaplots import plot_acf, plot_pacf
def ACF_and_PCAF(data):
    #plt.figure(figsize=(10, 4))
    #plt.plot(data)
    # plt.title('Ice Cream Sales', fontsize=18)
    # plt.ylabel('Sales')
    acf_plot = plot_acf(data, lags=100)
    plt.xlabel('时　间', fontsize=15)
    plt.ylabel('进　度', fontsize=15)
    #pacf_plot = plot_pacf(data)
    plt.show(block=True)
    return 0
ACF_and_PCAF(data.iloc[:, 28])
# for i in range(47):
#     x = data.iloc[:, i]
#     ACF_and_PCAF(x)
# #ACF_and_PCAF(data.iloc[:, 28])

# from statsmodels.tsa.stattools import acf,pacf
# plt.figure(figsize=(10, 4))
# lag_acf = acf(data.iloc[:, 28], nlags=100)
```

```
# plt.plot(-np.log(lag_acf)/np.max(-np.log(lag_acf))*100)
# #plt.plot(X, Y1)
# plt.show()
```

白噪声检验结果

```
from statsmodels.stats.diagnostic import acorr_ljungbox
print(u'白噪声检验结果：',acorr_ljungbox(data.iloc[:, 28], lags=2))#
返回统计量和 p 值  lags 为检验的延迟数

# PCC = []
# p_value = []
# for i in range(len(data)-1):
#     x = data.iloc[i, :]
#     y = data.iloc[i+1, :]
#     PCC.append(pearsonr(x, y)[0])
#     p_value.append(pearsonr(x, y)[1])
#
# ACF_and_PCAF(PCC)

# 绘制特征值图形
# for i in range(47):
#     x = data.iloc[:, i]
#     plt.figure(figsize=(8, 6))
#     plt.plot(np.array(x))
# plt.show()

# #绘制特征值图形
# PCC = []
# p_value = []
# for i in range(len(data)-1):
#     x = data.iloc[i, :]
#     y = data.iloc[i+1, :]
#     PCC.append(pearsonr(x, y)[0])
#     p_value.append(pearsonr(x, y)[1])
```

```
#
#
# PCC_1 = []
# p_value_1 = []
#
# temp = [0, 30, 38, 39, len(data)-2]
#
# for i in temp:
#     x = data.iloc[i, :]
#     y = data.iloc[i+1, :]
#     PCC_1.append(pearsonr(x, y)[0])
#     p_value_1.append(pearsonr(x, y)[1])
#
#
# plt.figure(figsize=(8, 6))
# plt.plot(PCC, zorder=1)
# plt.scatter(temp, PCC_1, c='red', s=40, zorder=2)
# plt.xlabel('时  间  序  列', fontsize=14)
# plt.ylabel('PCC', fontsize=14)
#
# plt.figure(figsize=(8, 6))
# plt.plot(p_value, zorder=1)
# plt.scatter(temp, p_value_1, c='red', s=40, zorder=2)
# plt.xlabel('时  间  序  列', fontsize=14)
# plt.ylabel('P-value', fontsize=14)
# plt.show()

# temp = [0, 30, 38, 39, -2]
#
# for i in temp:
#     x = data.iloc[i, :]
#     y = data.iloc[i+1, :]
#     PCC.append(pearsonr(x, y)[0])
#     p_value.append(pearsonr(x, y)[1])
#
```

```
# plt.figure(figsize=(8,6))
# plt.plot(PCC)
#
# plt.figure(figsize=(8,6))
# plt.plot(p_value)
# plt.show()
```