基于数学算法研究定日镜场的优化设计

摘要

随着科技的发展和能源的改革，塔式太阳能热发电系统以其比其它能源系统更多的优势受到了各个国家的重视和大力发展。本文针对塔式太阳能热发电系统中太阳能入射到定日镜并反射到接收器这一过程中定日镜场的利用效率以及对影响其效率的各种因素进行了分析，对如何通过定日镜场的控制策略来提高系统效率进行了具体的研究。

针对问题一，求定日镜的光学效率的难点在于阴影遮挡效率和余弦效率。第一步求阴影遮挡效率：首先建立地平坐标系中，将太阳与坐标系原点之间的连线与地平面之间的夹角称为太阳高度角。将太阳与坐标系原点之间的连线在基础平面上的投影与 Y 轴正向之间的夹角称为太阳方位角。然后以任意一个镜面建立镜面坐标系，借助旋转矩阵，此时光线在每个坐标系的值均可求出来，最后利用软件 Matlab 计算出每月 21 日的平均阴影遮挡效率。第二步求余弦效率：同样是建立坐标系，采用几何向量法，将余弦效率的公式转换成向量之间的计算，结合地皮坐标系中的向量求得平均余弦效率。第三步根据题目给定公式利用 *Matlab* 计算出年平均光学效率、年平均输出热功率以及单位镜面面积年平均输出热功率，分别为 80％、81％、72％、85％、47MW 和 14kW/m^2。

针对问题二，根据题设，所有定日镜的尺寸和安装高度相同，假设定日镜的尺寸和安装高度为已知量。首先根据题设约束条件定日镜场需要达到额定功率 60MW，借助问题一的数学模型和求得优化目标：单位镜面面积年平均输出热功率要求最大。在约束条件和优化目标同时满足的情况下，确定每月 21 日定日镜的数量和位置。利用定日镜的数量和位置可求得定日镜的分布密度，将其分布密度进一步分析找到最优解。

针对问题三，基于问题一模型的建立，EB 布局下定日镜数量和镜场接收能量多，EB 布局的各项性能特性均最低，因此选取 EB 布局。分析 EB 布局，在问题一地平坐标系上的 *xy* 平面上，将定日镜作为一个点，定日镜的分布以集热塔为中心，呈圆环状。镜场同一区域各环上相邻定日镜方位角和各环上的定日镜数量相等。为了优化布局，从镜场从最接近集热塔的区域第一环开始，设置相邻环定日镜径向间距恒等于当镜场环半径，增大到某一值后，对应环上定日镜开始出现阴影遮挡损失，此后以相邻环定日镜无阴影遮挡损失的最小径向间距。计算则采用遗传算法，提高算法实现效率，求得最终的结果为定日镜集中分布在集热塔东南方位。

关键词：地平坐标系；旋转矩阵；优化目标；EB 布局；遗传算法

一、问题重述

1.1 问题背景

构建以新能源为主体的新型电力系统，是我国实现“碳达峰”“碳中和”目标的一项重要措施。塔式太阳能发电是一种低碳环保的新型清洁能源技术。

定日镜是塔式太阳能光热发电站收集太阳能的基本组件，其底座由纵向转轴和水平转轴组成，平面反射镜安装在水平转轴上。纵向转轴的轴线与地面垂直，可以控制发射镜的方位角。水平转轴的轴线与地面平行，可以控制反射镜的仰俯角，定日镜及底座的模型图见图 1. 两转轴的交点（也是定日镜中心）离地面的高度称为定日镜的安装高度。塔式电站利用大量的定日镜组成阵列，称为定日镜场。定日镜将太阳光反射汇聚到安装在镜场中吸收塔顶端上的集热器，加热其中的导热介质，并将太阳能以热能形式储存起来，再经过热交换实现由热能向电能的转化。太阳光并非平行光线，而是具有一定锥形角的一束锥形光线（参考图 2），因此太阳入射光线经定日镜任意一点的反射光线也是一束锥形光线。定日镜在工作时，控制系统根据太阳的位置实时控制定日镜的法向，使得太阳中心发出的光线经定日镜中心反射后指向集热器中心。集热器中心的离地高度称为吸收塔高度。

图 1 定日镜模型图

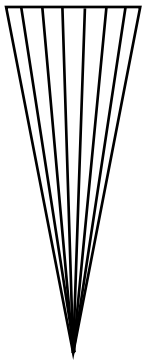

图 2 太阳光锥形光线图

现计划在中心位于东经 98.5°,北纬 39.4°，海拔 3000*m*，半径 350*m* 的圆形区域内建设一个圆形定日镜场。以圆形区域中心为圆点，正东方向为 *x* 轴正向，正北方向为 *y* 轴正向，垂直与地面向上方向为 *z* 轴正向建立坐标系，称为镜场坐标系。

规划的吸收塔高度为 80*m*，集热器采用高 8*m*、直径 7*m* 的圆柱形外表光式集热器。吸收塔周围 100*m* 范围内不安装定日镜，留出空地建造厂房，用于安装发电、储能、控制等设备。定日镜的形状为平面矩形，其上下两条边始终平行于地面这两条边之间的距离称为镜面高度，通常镜面宽度不小于镜面高度。镜面边长在 2*m* 至 8*m* 之间，安装高度在 2*m* 至 6*m* 之间，安装高度必须保证镜面在绕水平转轴旋转时不会触及地面。由于维护及清洗车辆行驶的需要，要求相邻定日镜底座中心之间的距离比镜面宽度多 5*m* 以上。

为简化计算，本问题中所有“年均”指标的计算时点均为当地时间每月 21 日 9：00、10：30、12：00、13：30、15：00。

1.2 需要解决的问题

问题一：若将吸收塔建于该圆形定日场中心，定日镜尺寸均为 $6m\times6m$。安装高度均为 $4m$，且给定所有定日镜中心的位置（相关数据查看附件），计算该定日镜场的年平均光学效率、年平均输出热功率，以及单位镜面面积年平均输出热功率。其结果分别按表 1 和表 2 的格式填入表格。

问题二：按设计要求，定日镜场的额定年平均输出热功率（简称额定功率）为 6MW。若所有定日镜尺寸及安装高度相同，设计出定日镜场的以下参数：吸收塔的位置坐标、定日镜尺寸、安装高度、定日镜数目、定日镜位置，使得定日镜场在达到额定功率的条件下单位镜面面积年平均输出功率最大。其结果分别按表 1、2、3 的格式填入表格，并将吸收塔的位置坐标，定日镜尺寸，安装高度，位置坐标按模板规定的格式保存到 *result2.xlsx* 文件中。

问题三：如果定日镜尺寸可以不同，安装高度也可以不同，额定功率设置同问题 2，需要重新设计定日镜场的各个参数，使得定日镜场在达到额定功率的条件下，单位镜面面积年平均输出热功率尽量大，其结果分别按表 1、表 2 和表 3 的格式填入表格，并将吸收塔的位置坐标、各定日镜尺寸、安装高度，位置坐标按模板规定的格式保存到 *result3.xlsx* 文件中。

二、模型假设

1、假设所有定日镜的材料、形状、其中的导热介质等均一致。
2、假设吸收塔吸收热能的整个过程中没有障碍物的影响。
3、假设每一个定日镜所处的地面平整，海拔高度均一致。
4、假设反射前的太阳光时平行光。
5、假设整个光热发电系统没有题设之外的任何能量损耗，如热传递等。

三、问题分析

2.1 问题一的分析

问题一首先需要对附件中的数据进行处理。附件中给出了每面定日镜的 x 轴和 y 轴坐标，我们依据此坐标可以求出他们据圆心的距离，再根据数学推导出镜面到集热器中心的距离。

其次分别对吸收塔的影子和相邻镜面之间遮挡的相互影响建立模型。本题将年均指标的计算时点均为当地时间的 21 日的五个时间点，利用当地时间 ST 可以求出太阳赤纬角和太阳时角。根据题目附录中相关公式求出太阳高度角和太阳方位角。

然后以集热塔为圆心，结合每个定日镜的坐标点，建立坐标系。太阳高度角为入射光线的方向，定日镜到吸收塔的连线为反射光线的方向。对五个时间点太阳不同的位置分别计算，得出通用遮挡面积模型。确定定日镜的四个顶点在平行光下形成的阴影位置，利用方向余弦确定这四个点。

最后根据遮挡面积模型得出阴影遮挡损失、余弦损失和集热器截断效率，进而得出光学效率和热功率。

2.2 问题二的分析

问题二是一个最优化问题。在问题一中已知：太阳高度角、太阳方位角、定日镜的投影面积、定日镜的采光面积、定日镜的年平均输出热功率。根据题目需要求解吸收塔的位置坐标、定日镜的尺寸、安装高度以及定日镜的数目和位置坐标，使得定日镜场在达到额定功率的条件下，单位镜面面积年平均输出热功率尽量大。

（1）约束条件如下：

镜面高度<安装高度−80m<安装高度−80m；

镜面宽度>镜面高度>镜面高度；

镜面边长在 2m 至 8m 之间；

安装高度在 2m 至 6m 之间。

（2）先假设定日镜的尺寸和安装高度已知，需要求解每个定日镜的位置和定日镜的数目。

（3）尝试在不同的定日镜密度下，计算定日镜场的单位镜面面积年平均输出热功率，找出最优解。

以吸收塔为中心，罗列定日镜位置，选择面积最小的定日镜作为基准定日镜，计算该定日镜在所有时刻的输出热功率。得到单位镜面面积年平均输出热功率时的参数。

2.3 问题三的分析

根据题目要求，需要改进的模型包括定日镜尺寸和安装高度。需要考虑如何合理选择定日镜尺寸和安装高度以最大限度地发电，同时满足额定功率要求。

增大相邻定日镜之间的距离可以有效地降低镜场产生阴影遮挡的概率，除此之外缩小定日镜的面积也可以减少阴影遮挡的发生。但通过缩小定日镜面积来减少阴影损失的方法与电站收益最大化的原则相违背，因此只能通过优化定日镜场的布置策略并合理地布置定日镜场来减小定日镜与定日镜之间造成的阴影遮挡损失，这也是塔式光热电站定日镜场的优化布置过程中需要解决的关键问题之一。

故而这题有两个关键点：

（1）定日镜尺寸的选择：可以通过对不同尺寸的定日镜进行效率和能量收集能力的评估，选择效率高且能够收集到更多太阳能的定日镜尺寸。可以通过实验或者模拟计算来确定最佳的定日镜尺寸。

（2）安装高度的选择：可以考虑将光伏阵列安装在可调整安装高度的机架上，以便实现最佳的太阳能接收效果。通过模拟不同安装高度下的太阳能收集情况，选择能够最大限度地收集到太阳能的安装高度。

四、符号说明

符号	说明
$DNI(t)$	时间t的法向直接辐射辐照度$\left(kW/m^2\right)$
$\eta_{\cos}$	余弦效率

符号	说明
η_{sb}	阴影遮挡效率
η_{trunc}	集热器截断效率
A	单个定日镜的采光面积$\left(m^2\right)$
N	定日镜的总数$\left(面\right)$
a	定日镜高度$\left(m\right)$
b	定日镜宽度$\left(m\right)$
h_1	定日镜安装高度$\left(m\right)$
h_2	集热器中心高度$\left(m\right)$
i	第i面定日镜
$\left(x_i,y_i\right)$	平面位置坐标$\left(m\right)$

五、模型的建立与求解

5.1 问题一模型的建立与求解

题目已知：中心位于东经 98.5°，北纬 39.4°，海拔 3000m，半径 350m。吸收塔高度为 80m，集热器采用高 8m，直径 7m 的圆柱。镜面边长在 2m 至 8m 之间，安装高度在 2m 至 6m 之间。要求相邻定日镜底座中心之间的距离比镜面宽度多 5m 以上。

定日镜场光学效率模型的建立是基于地平坐标系。单个定日镜场光学效率是研究镜场光学效率的基础，单个定日镜的瞬时光学效率由题目已知：

$$\eta=\eta_{sb}\eta_{\cos}\eta_{at}\eta_{trunc}\eta_{ref} \qquad 式(1)$$

5.1.1 阴影遮挡效率数学模型的建立与求解

阴影遮挡效率是定日场中没有被阴影遮挡的镜面面积与镜场总面积之比，由于已知定日镜的坐标及太阳高度角和方位角，因此本文采用几何投影法计算和平面方程来求阴影遮挡效率。

阴影遮挡损失主要是两部分，第一部分是相邻的定日镜互相遮挡造成的阴影损失，第二部分是塔遮挡定日镜造成的阴影损失相邻的定日镜互相遮挡造成的阴影损失。

（1）下面首先计算定日镜与定日镜相互遮挡造成的阴影损失，主要利用当前建立境面矩阵，确定基本的坐标，然后乘以旋转矩阵，最后通过第三个矩阵将平面还原，判断点是否落入镜面内。

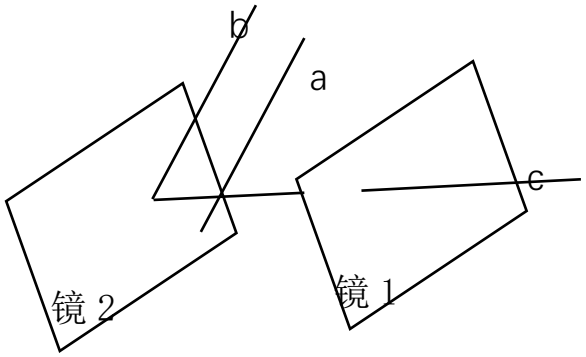

图 3 相邻镜面挡光模拟图

如上图所示，图中 *a* 和 *b* 是太阳光发散的平行入射光线，光线 *c* 是入射光线 1 经过镜 2 反射的反射光线，入射光线 *a* 经过镜 2 反射后落入镜 1 的背面，使得镜 1 在镜 2 上产生了阴影,反射光线 *c* 被镜 2 遮挡，造成镜 2 对镜 1 的遮挡。

求阴影挡光损失本质上是求经过镜 1 的任意的一条光线是否会落在镜 2 的镜面上，求出光线与镜 2 的交点坐标值，结合几何数学求得阴影区域面积。

第一步：建立镜面坐标系

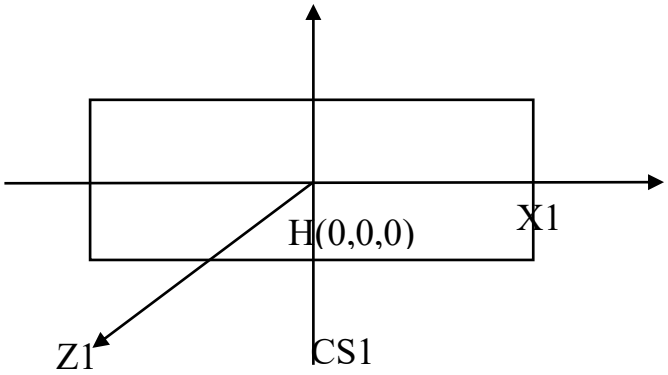

图 4 镜面坐标系

假设任意一条光线投影到大地坐标系上的向量为：

$$\vec{V}_0 = (a,b,c) \qquad \text{式(2)}$$

图中为镜 1 坐标系，已知镜 1 中某一点 $H_1 = (x_1, y_1)$，求经过光线落入镜 2 中的坐标 $H_2 = (x_2, y_2)$（假设该坐标位于镜 2 坐标系中），然后判断该坐标点是否位于镜 2 内。

假设镜面坐标系 *a* 到地面坐标系 *b* 的转换关系单位矩阵为：

$$T = \begin{pmatrix} l_x, l_y, l_z \\ m_x, m_y, m_z \\ n_x, n_y, n_z \end{pmatrix} \qquad \text{式(3)}$$

其中$(l_x,m_x,n_x),(l_y,m_y,n_y),(l_z,m_z,n_z)$是镜面坐标系 *a* 的 3 个轴在地面坐标系 *b* 的向量表示。

一束光线在镜面坐标系中的向量为$\vec{V}_H$，在地面坐标系中的向量为$\vec{V}_0$。二者之间的相互转换关系为：

$$\vec{V}_0=\begin{pmatrix} l_x,l_y,l_z \\ m_x,m_y,m_z \\ n_x,n_y,n_z \end{pmatrix}\cdot\vec{V}_H \qquad \text{式(4)}$$

若要计算$H_2=(x_2,y_2)$的值：①先将镜 1 中的某一点H_1转换到地面坐标系下的坐标，并表示为$H_1^{'}$。

$$H_1^{'}=\begin{pmatrix} l_x,l_y,l_z \\ m_x,m_y,m_z \\ n_x,n_y,n_z \end{pmatrix}\cdot H_1+O_A=\begin{pmatrix} x_1^{'} \\ y_1^{'} \\ z_1^{'} \end{pmatrix} \qquad \text{式(5)}$$

②再将地面坐标系下的$H_1^{'}$ 转换到镜 2 坐标系下，表示为$H_1^{''}$。

$$H_1^{''}=\begin{pmatrix} l_x,l_y,l_z \\ m_x,m_y,m_z \\ n_x,n_y,n_z \end{pmatrix}^T\cdot(H_1^{'}-O_B)=\begin{pmatrix} x_1^{''} \\ y_1^{''} \\ z_1^{''} \end{pmatrix} \qquad \text{式(6)}$$

③将地面坐标系下的光线转换到镜 2 坐标系下。

$$\vec{V}_H=\begin{pmatrix} l_x,l_y,l_z \\ m_x,m_y,m_z \\ n_x,n_y,n_z \end{pmatrix}^T\cdot\vec{V}_0=(a,b,c) \qquad \text{式(7)}$$

④在镜 2 坐标系中，根据两点确定一条直线的原理，计算光线与镜 2 交点。

$$\frac{x_2-x_1^{''}}{a}=\frac{y_2-y_1^{''}}{b}=\frac{-z_1^{''}}{c} \qquad \text{式(8)}$$

求得：

$$\begin{cases} x_2=\dfrac{cx_1^{''}-az_1^{''}}{c} \\ y_2=\dfrac{cy_1^{''}-bz_1^{''}}{c} \end{cases} \qquad \text{式(9)}$$

最后利用软件 *Matlab* 运行代码 1（见附录）判断H_2是否在镜 2 内。

（2）计算塔阴影挡光造成的阴影损失。

先分析太阳高度角和方位角的变化，求得每月 21 日 9：00、10：30、12：00、13：30、15：00，这五个时间段的太阳高度角和方位角。主要分为以下几个步骤：

$$\sin\delta=\sin\frac{2\pi D}{365}\sin\left(\frac{2\pi}{360}23.45\right) \qquad \text{式（10）}$$

其中 D 以春分作为第 0 天起算的天数，得到一年的 D 值。

表 5.1 各个月份距离春分的天数

月份	1 月	2 月	3 月	4 月	5 月	6 月	7 月	8 月	9 月	10 月	11 月	12 月
天数	-58	-28	0	31	61	92	122	153	184	214	245	275

计算太阳时角：

$$\omega=\frac{\pi}{12}(ST-12) \qquad \text{式（11）}$$

前面两个计算出来，然后依据题设附录公式可以求得太阳方位角和太阳高度角的值：

$$\text{太阳高度角的正弦值：}\sin\alpha_s=\cos\delta\cos\varphi\cos\omega+\sin\delta\sin\varphi \qquad \text{式（12）}$$

$$\text{太阳方位角的余弦值：}\cos\gamma_s=\frac{\sin\delta-\sin\alpha_s\sin\varphi}{\cos\alpha_s\cos\varphi} \qquad \text{式（13）}$$

利用软件 Matlab 运行代码 2 可得太阳高度角和太阳方位角的值，如下表所示：

表 5.2 太阳方位角余弦值

时间	9:00	10:30	12:00	13:30	15:00
值	-0.4181	0.0834	0.2188	-0.13	-0.4787

太阳高度角值为：-0.0148。

5.1.2 余弦效率数学模型的建立与求解

在实际条件下，太阳光并非全部垂直射向定日镜镜面。除去定日镜垂直光线带来的辐射能量，其余相对于定日镜来说倾斜光线带来的辐射能量小于垂直光线的辐射能量的部分能量就是余弦损失。由题目我们已知：

$$\text{余弦效率}\,\eta_{\cos}=1-\text{余弦损失} \qquad \text{式（14）}$$

直接求余弦损失有点复杂，因此建立坐标系，借助数学中的几何向量法来求解。

假设：定日镜镜面中心坐标为(x,y,z)，吸热器上目标点坐标为$(0,0,H)$，H是吸热器中心点距离地面的高度，s是太阳入射光线的方向向量，r是镜面反射光线的方向向量，建立如图 3 所示简化模拟图。

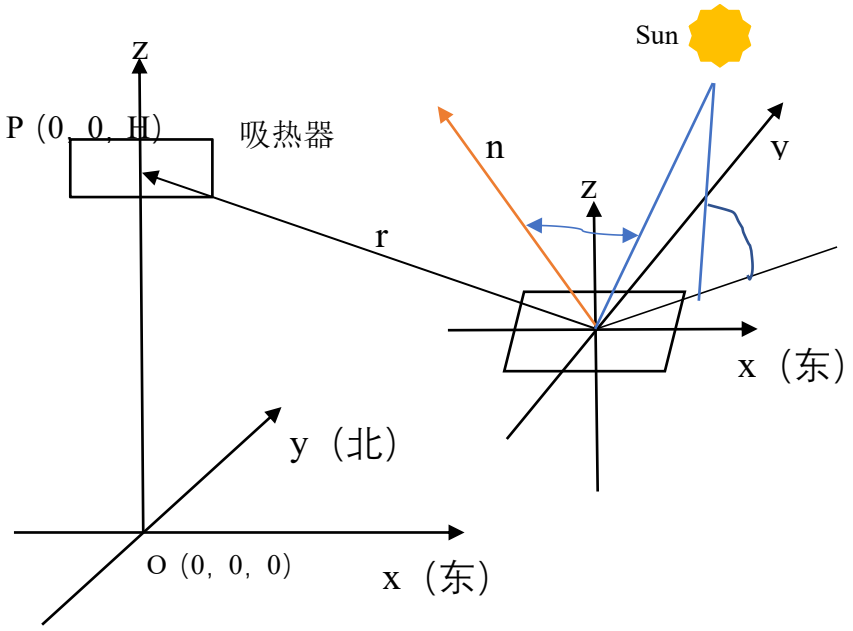

图 7 光线反射原理简化图

由反射定律可知，定日镜面镜面的入射角等于反射角，结合几何矢量法得到余弦效率的计算公式：

$$\eta_{\cos}=\cos\theta=\left|\frac{\left(1+\cos\left(2\theta\right)\right)}{2}\right|=\left|\sqrt{\frac{\left(1-s\cdot r\right)}{2}}\right| \qquad \text{式（15）}$$

其中据所学知识可知：

$$s=\left(-\cos\alpha_s\sin\gamma_s,-\cos\alpha_s\cos\gamma_s,-\sin\alpha_s\right) \qquad \text{式（16）}$$

$$r=\frac{\left(-x,-y,H-z\right)}{\sqrt{x^2+y^2+\left(H-z\right)^2}} \qquad \text{式（17）}$$

5.1.3 大气透射率的求解

大气透射率是指一定波段的光线从地球大气层中穿过的能量比例，题目附录中可以得到它的计算公式：

$$\eta_{at}=0.99321-0.0001176d_{HR}+1.97\times10^{-8}\times d_{HR}^2\left(d_{HR}\le1000\right) \qquad \text{式（18）}$$

其中d_{HR}表示镜面中心到集热器中心的距离。

计算d_{HR}的过程如下：

根据附录文件中每个定日镜的坐标可以求出定日镜距离集热塔的距离即为半径，结果显示所有半径均大于 100m,表示数据均合格。

已知上述所求得的半径，题目条件可得吸收塔高度为 80m，如图 8 所示，二者构成直角三角形，由勾股定理计算得出镜面中心到集热器中心的距离即为d_{HR}。

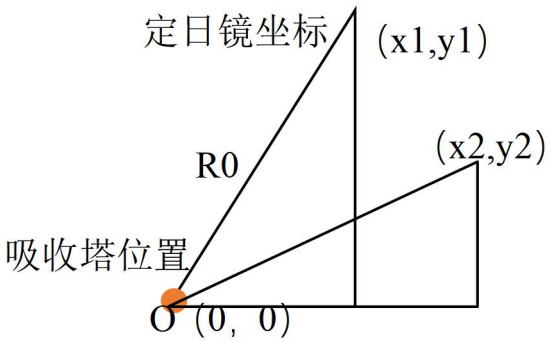

图 8 定日镜到吸收塔中心的距离模拟图

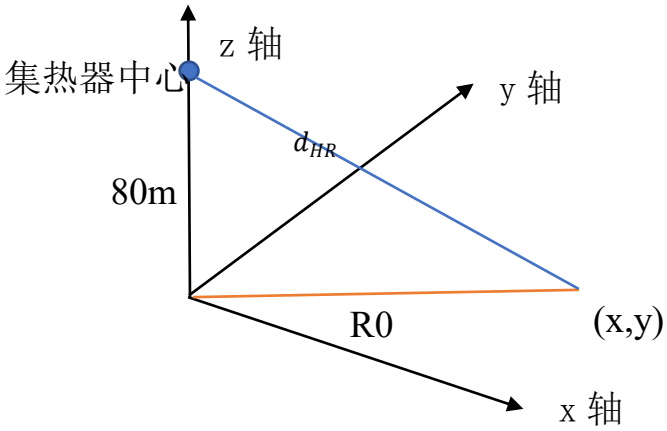

图 9 定日镜到集热器中心的距离模拟图

最后利用软件 *Matlab* 运行代码 2 得出具体数值，部分结果展示见表 5.3。

表 5.3 部分处理后的数据

x 轴坐标	y 轴坐标	R_0	d_{HR}	η_{at}
107.25	11.664	107.8823961	134.3078977	0.977770752
105.36	23.191	107.8821212	134.3076769	0.977770777
102.235	34.447	107.8823018	134.3078219	0.97777076
97.911	45.299	107.8821733	134.3077188	0.977770772
92.44	55.619	107.8824673	134.3079549	0.977770745
85.884	65.288	107.882271	134.3077972	0.977770763
78.322	74.191	107.8825295	134.3080048	0.97777074
69.841	82.224	107.8821183	134.3076746	0.977770777
60.542	89.293	107.882221	134.3077571	0.977770768
50.533	95.315	107.8820343	134.3076071	0.977770785
39.931	100.22	107.8820335	134.3076065	0.977770785
28.862	103.95	107.8824246	134.3079206	0.977770749
17.453	106.461	107.8821196	134.3076756	0.977770777
5.841	107.724	107.8822388	134.3077714	0.977770766
-5.841	107.724	107.8822388	134.3077714	0.977770766
-17.453	106.461	107.8821196	134.3076756	0.977770777

x 轴坐标	y 轴坐标	R_0	d_{HR}	η_{at}
-28.862	103.95	107.8824246	134.3079206	0.977770749
-39.931	100.22	107.8820335	134.3076065	0.977770785
-50.533	95.315	107.8820343	134.3076071	0.977770785
-60.542	89.293	107.882221	134.3077571	0.977770768
-69.841	82.224	107.8821183	134.3076746	0.977770777
-78.322	74.191	107.8825295	134.3080048	0.97777074
-85.884	65.288	107.882271	134.3077972	0.977770763
-92.44	55.619	107.8824673	134.3079549	0.977770745
-97.911	45.299	107.8821733	134.3077188	0.977770772
-102.235	34.447	107.8823018	134.3078219	0.97777076
-105.36	23.191	107.8821212	134.3076769	0.977770777
-107.25	11.664	107.8823961	134.3078977	0.977770752
-107.882	0	107.882	134.3075795	0.977770788
-107.25	-11.664	107.8823961	134.3078977	0.977770752
-105.36	-23.191	107.8821212	134.3076769	0.977770777
-102.235	-34.447	107.8823018	134.3078219	0.97777076
-97.911	-45.299	107.8821733	134.3077188	0.977770772

5.1.4 集热器截断效率数学模型的建立与求解

集热器的截断效率为投射到吸热器上的能量值与镜场投射总能量之比，镜场透射总能量为镜面全反射能量减去阴影遮挡损能量。

定日镜在圆柱形集热器面上所形成的光斑大小与定日镜的光线跟踪误差和太阳的高度角和方位角有关。因此，可采用整体误差 来描述反射光线的发散程度。由于定日镜在圆柱形集热器面上的能流密度分布接近高斯分布，故而，截断效率的计算可以通过圆柱形集热器面上对能流密度在边界范围内积分得到：

$$\eta_{\mathrm{int}ercept}=\frac{1}{2\pi\sigma_{tot}^{2}}\iint_{xy}\exp\left(-\frac{x^{2}+y^{2}}{2\sigma_{tot}^{2}}\right)dxdy \qquad \text{式（19）}$$

5.1.5 计算年平均光学效率

上述计算过后我们可以已知：阴影遮挡效率、余弦效率、大气透射率、集热器截断效率。根据题目附录已给公式我们可以求出光学效率：

$$\eta=\eta_{sb}\eta_{\cos}\eta_{at}\eta_{trunc}\eta_{ref} \qquad \text{式（20）}$$

5.1.6 计算年平均输出热功率

由题目附录可得定日镜场的输出热功率 E_{field} 为：

$$E_{field}=DNI\cdot\sum_{i}^{N}A_i\eta_i \qquad \text{式（21）}$$

其中 DNI 为法向直接辐射辐照度；N 为定日镜总数，A_i 为第 i 面定日镜采光

面积； η_i 为第 i 面镜子的光学效率。根据下面的公式可以得出 DNI 值：

$$DNI = G_0\left[a + b\exp\left(-\frac{c}{\sin\alpha_s}\right)\right]$$
$$a = 0.4237 - 0.00821\left(6 - H\right)^2 \qquad \text{式（22）}$$
$$b = 0.5055 + 0.00595\left(6.5 - H\right)^2$$
$$c = 0.2711 + 0.01858\left(2.5 - H\right)^2$$

5.1.7 计算单位镜面面积年平均输出热功率

单位镜面面积输出热功率等于年平均输出热功率除以定日镜的总面积的比值。

综上所述，利用软件 Matlab 运行代码 3（见附录）。计算得到最终结果为下表 1 和表 2：

表 1 问题一每月 21 日平均光学效率及输出功率

日期	平均光学效率	平均余弦效率	平均阴影遮挡效率	平均截断效率	单位面积镜面平均输出热功率（kW/m2）
1 月 21 日	0.84	0.88	0.89	0.88	446
2 月 21 日	0.81	0.84	0.91	0.85	428
3 月 21 日	0.90	0.90	0.88	0.90	477
4 月 21 日	0.87	0.89	0.89	0.92	444
5 月 21 日	0.89	0.90	0.84	0.91	437
6 月 21 日	0.81	0.88	0.78	0.90	442
7 月 21 日	0.84	0.90	0.83	0.88	466
8 月 21 日	0.84	0.85	0.88	0.87	470
9 月 21 日	0.78	0.87	0.90	0.94	463
10月 21日	0.77	0.89	0.87	0.87	477
11月 21日	0.84	0.87	0.82	0.84	482
12月 21日	0.92	0.97	0.91	0.93	424

表 2 问题一年平均光学效率及输出功率表

年平均光学效率	年平均余弦效率	年平均阴影遮挡效率	年平均截断效率	年平均输出热功率 (MW)	单位面积镜面年平均输出热功率 (kW/m^2)
0.80	0.81	0.72	0.85	47	14

5.2 问题二模型的建立与求解

问题二是一个最优化问题，它在第一问的基础加上了约束条件，目的是确定定日镜的安装方案和吸收塔的位置。

由资料分析在无法改变太阳高度角的前提下，镜场中阴影遮挡的发生主要与定日镜之间的距离有关。因此通过改变不同区域定日镜的径向间距来布置定日镜场，以找到定日镜场的最优布置方案。

对于不同区域的定日镜的径向间距 $\triangle r$，需要确保相邻定日镜之间不会发生碰撞，即 $\triangle r$ 的约束条件为：

$$\triangle r \geq \cos 30^{\circ} d = 0.866d \qquad \text{式（23）}$$

优化目标：

$$\max\left(\text{单位面积镜面平均输出热功率} E_{fieldaverage} = \frac{E_{field}}{n \cdot a \cdot b}\right) \qquad \text{式（24）}$$

目标函数：

$$E_{fieldaverage} = f\left(a, b, h_1, h_2, n\right) \qquad \text{式（25）}$$

约束条件：

$$s.t.\begin{cases} 5+b \leq \sqrt{\left(x_n - x_{n-1}\right)^2 + \left(y_n - y_{n-1}\right)^2} \\ 5+b \leq \sqrt{\left(x_n - x_{n-1}\right)^2} \\ 5+b \leq \sqrt{\left(y_n - y_{n-1}\right)^2} \\ 5 \leq a < b \leq 8 \\ 2 \leq h_1 \leq 6 \\ 4 \leq h_2 \leq 76 \\ P \geq 60MW \\ 100 \leq \sqrt{x_n^2 + y_n^2} \leq 350 \end{cases} \qquad \text{式（26）}$$

最终利用软件 *Matlab* 运行代码 4（见附录）得到最终答案，部分答案展示见下表：

表 5.4 问题二的设计参数表

序号	定日镜宽度	定日镜高度	x 坐标 (m)	y 坐标 (m)	z 坐标 (m)
1	3.7	5.27	106.319	11.558	5.27
2	3.7	5.27	106.831	22.837	5.27
3	3.7	5.27	102.984	34.364	5.27
4	3.7	5.27	95.986	45.247	5.27
5	3.7	5.27	92.554	55.727	5.27
6	3.7	5.27	86.822	65.059	5.27
7	3.7	5.27	79.131	73.250	5.27
8	3.7	5.27	69.806	83.602	5.27
9	3.7	5.27	61.255	88.591	5.27
10	3.7	5.27	50.242	93.600	5.27

5.3 问题三模型的建立与求解

定日镜场的优化布局有 DELSOL 布局、EB 布局、No blocking-dense 布局。定日镜场布置主要有麦田型排列和辐射网络法两种类型[2]，其中麦田型排列法布置比较简单且定日镜密度较大，但其主要适用于定日镜数量较少的情况。辐射网络法采用径向交错排列的方式。

目前塔式光热电站，定日镜数量从几万慢慢增加到几十万，数量越来越多，因此就定日镜场布置发展大部分采用辐射网络法。本文提出定日镜场布置设计原

则采用密集布置和交错布置相结合的方法，即靠近吸热塔的一定区域采用密集布置，剩余区域采用交错布置方式，该布置方式充分利用占地面积以及高能量利用效率的区域，同时避免了较大的阴影和遮挡损失。

对基于辐射网格布局和无遮挡损失原则所衍生出的无遮挡(EB)布局、无阻塞(No blocking-dense)布局、经验型(DELSOL)布局 3 种塔式定日镜场布局进行了研究。利用软件 Matlab 对建模所得镜场坐标数据进行处理,得到径向间距和方位间距变化曲线,验证了 3 种布局数学模型的合理性。通过分析各镜场中基础环和交错环的排布特点得出镜环布置规则,同时从镜场的定日镜数量、光学效率、土地利用率、接收能量等方面对比不同布局的性能,分析其优缺点。

结果表明:在镜场半径为 1000 m 的范围内,No blocking-dense 布局下镜场的年均光学效率和土地利用率最高,EB 布局次之,但 EB 布局下定日镜数量和镜场接收能量多,EB 布局的各项性能特性均最低。故而本题采用 EB 布局。

设定日镜的长度为 H，各区域首环定日镜方位间距 $A_{Zi,1}$，各区域方位 $A_{Zi,i}$，各镜换定日镜数量 $N_{hel,i}$，计算公式分别如下：

$$D_M = \sqrt{W_S^2 + H_S^2} \qquad \text{式（27）}$$

$$\triangle A_{Zi,i} = \left(1.791 + 0.6396\theta_L\right)\cdot W_S + \frac{0.02873}{\theta_r - 0.04902} \qquad \text{式（28）}$$

$$\triangle \alpha_{Z,i} = \arcsin\left(\frac{\triangle A_{Zi,l}}{R_{i,l}}\right) \qquad \text{式（29）}$$

$$N_{hel,i} = \frac{2\pi}{\triangle \alpha_{Zi}} \qquad \text{式（30）}$$

镜场中各区域首环定日镜方位间距计算公式为：

$$\triangle A_{Z1,1} = \triangle A_{Z2,1} = \cdots = \triangle A_{Zi,1} = A_{sf} \bullet W_s \qquad \text{式(31)}$$

式中，A_{sf} 为方位间距因子，其取值主要与塔高有关，通常取值为 2。

各区域内除首环外，其余环定日镜方位间距通过该区域定日镜方位角得出，计算公式为:

$$\triangle A_{Zi,j} = 2R_{i,j}\cdot \sin\left(\triangle \alpha_{Z,j}/2\right) \qquad \text{式(32)}$$

式中，$\triangle A_{Zi,j}$ 为镜场第 i 区域第 j 环定日镜的方位间距。

第 i 个镜场区域中定日镜方位角计算公式为:

$$\triangle \alpha_{Z,i} = 2\arcsin\left(\frac{\triangle A_Z}{2R_{i,1}}\right) \qquad \text{式(33)}$$

各镜环定日镜数量计算公式见式(30)。

相邻镜场区域交界处定日镜径向间距恒定设置为$\triangle R = DM$ 。 $\triangle R_{1,1}$为镜场第 1 个区域首环与第 2 环之间的径向间距，计算公式为：

$$\triangle R_{1,1} = \sqrt{D_M^2 - \left(\triangle A_{Z1,1} / 2\right)^2} \qquad \text{式（34）}$$

同一区域相邻环的定日镜径向间距始终设置为 。当镜场半径增大到某一值后，对应环定日镜开始出现遮挡损失，此后以几何作图法重新确定径向间距，见式(35)-(39)。

$$L_1 = \sqrt{\left(H_t + 0.5L - Z_0\right)^2 + R_{i,j}^2} \qquad \text{式（35）}$$

$$\alpha_1 = \arcsin\left(\frac{R_{i,j}}{L_1}\right) \qquad \text{式（36）}$$

$$\alpha_2 = \arcsin\left(\frac{R_0}{L_1}\right) \qquad \text{式（37）}$$

$$L_3 = \tan\gamma \cdot \left(H_t - Z_0\right) = \tan\left(\alpha_1 + \alpha_2\right)\left(H_t - Z_0\right) \qquad \text{式（38）}$$

$$\triangle R_{i,j} = 2L_2 = 2\left(L_3 - R_{i,j}\right) \qquad \text{式（39）}$$

式中， H_t为吸热塔光学高度;L 为吸热器高度； L_1为吸热器中心点到前排定日镜镜面中心点连线的长度； L_2为后排定日镜反射光线刚好不被前排定日镜遮挡时光线在前后 2 个定日镜边缘点处连线的中心点距前排定日镜镜面中心点的水平距离;L_3为前排定日镜镜面中心点距吸热塔的水平距离;α_1为吸热器中心点到前排定日镜镜面中心点连线与吸热塔竖直轴线的夹角;α_2为吸热器中心点到前排定日镜镜面中心点连线与后排定日镜反射光线刚好不被前排定日镜遮挡时后排定日镜边缘点反射光线到吸热器中心点连线间的夹角;Z_0为定日镜中心距水平地面高度;R_0为定口镜圆环半径;$R_{i,j}$为镜场第i个区域中第j个镜环的半径;γ为后方定日镜反射光线刚好不被前排定日镜遮挡时反射光线与吸热塔轴线夹角。如下图所示。

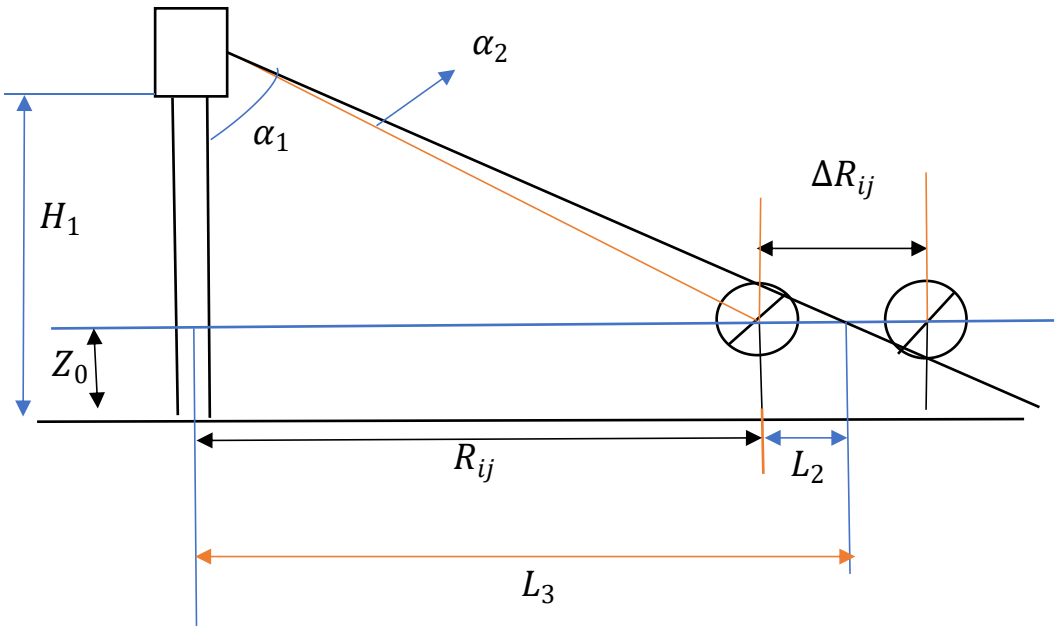

图 10 无遮挡几何作图法示意图

综上所述，利用软件 *Matlab* 运行代码 5（见附录）得到最终答案，部分答案展示如下表：

表 5.5 问题三部分答案展示

序号	定日镜宽度	定日镜高度	x 坐标 (m)	y 坐标 (m)	z 坐标 (m)
1	3.503351	5.928125	106.31	11.558	5.27
2	3.291937	4.958230	106.83	22.837	5.27
3	3.654806	5.635057	102.98	34.364	5.27
4	3.712090	5.448092	95.986	45.247	5.27
5	4.134806	5.201509	92.554	55.727	5.27
6	3.966695	5.083718	86.822	65.059	5.27
7	3.873739	5.520836	79.131	73.250	5.27
8	4.155292	5.589980	69.806	83.602	5.27
9	3.736677	5.332927	61.255	88.591	5.27
10	3.827580	5.920412	50.242	93.600	5.27
11	4.016605	5.620675	40.365	101.96	5.27
12	3.367451	5.780768	28.641	104.90	5.27
13	4.018979	5.693711	17.353	105.53	5.27
14	4.153664	5.053983921	5.803	107.460	5.27

采用遗传算法，如下图所示过程，利用 Matlab 软件运行代码 5 得出最终结果。

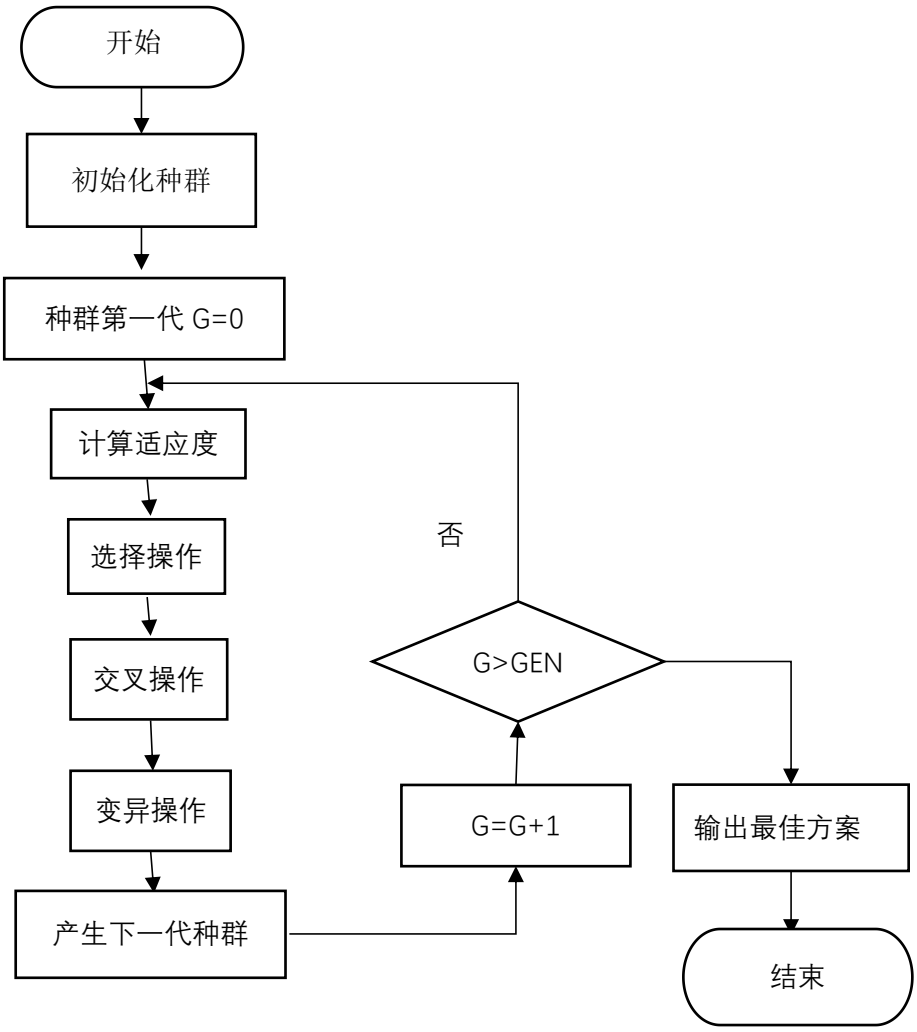

图 11 遗传算法计算过程

六、模型评价与优化

6.1 模型优点

本模型其优点在于四个方面：

(1)准确性：本模型是一种基于数学和物理原理的模型，通过对光线传播的数学建模，可以精准地预测不同时间和地点的日照强度和太阳角度。这使得模型在日照资源评估、建筑设计和太阳能利用等领域具有高度的准确性。

(2)灵活性：本模型可以根据所需要的精度和计算资源进行灵活的调整。可以建立不同的坐标系，选择不同的方法如光线追踪法、光学参数和模拟参数，从而根据不同的需求条件下实现不同的计算速度和精准度。

(3)可视化：本模型可以生成直观的可视化结果图，如日照图、阴影图、太

阳角度变化图等。这些结果可以直观地展示日照分布、阴影情况和太阳的变化等，对于城市规划，资源利用和建筑设计等令领域的发展均有很大的帮助。

(4)可拓展性：本模型可以与其他模型和软件进行集成，如地理信息系统（*GIS*）、建筑信息模型（*BIM*）等。这使得模型的应用更加灵活和全面，并且可以与其他相关领域的模型进行数据交互和共享。

6.2 模型缺点

本模型的缺点为以下两个方面：

(1)复杂性：定日境场模型的建立需要考虑多个因素，包括地理位置、地形、建筑物形状等。这些因素的复杂性使得模型的建立和计算过程相对复杂，需要一定的专业知识和技术支持。

(2)近似性：定日境场模型在建模过程中通常会进行一些近似和简化，以减少计算复杂性和提高计算效率。这可能会导致模型结果与实际情况之间存在一定的差异，尤其是在复杂的光照条件下。

6.3 模型优化

定日镜场模型的优化主要考虑在两个方面：

定日镜场模型的效率还与大气参数有关，我们可以考虑在定日镜场模型中加入大气参数的分析，如大气温度、湿度、气压等。在优化模型时，可以考虑使用更精确的大气参数数据，以提高模型的准确性和可靠性。

定日镜场模型还要考虑天气条件，如雨、雪、雾等天气现象会导致信号的衰减和散射。在优化模型时，可以考虑引入天气数据，以更精确的模拟不同天气条件下的信号传播情况。

七、参考文献

[1]张平,奚正稳,华文瀚等.太阳能塔式光热镜场光学效率计算方法[J].技术与市场,2021,28(06):5-8.

[2]程小龙. 基于光学效率的塔式电站镜场布局优化设计研究[D].合肥工业大学,2018.

[3]谢飞. 塔式太阳能热电系统定日镜场光学仿真与应用研究[D].浙江大学,2013.

[4]胡闹,赵豫红,冯结青.塔式太阳能热电站聚光集热系统优化设计方法[J].高校化学工程学报,2021,35(06):1041-1050.

[5]司守奎，孙兆亮，数学建模算法与应用.北京：国防工业出版社，2017.4,10.3,240

[6]蔡志杰.太阳影子定位[J].数学建模及其应用,2015,4(04):25-33.

[7]孙浩,高博,刘建兴. 塔式太阳能电站定日镜场布局研究[J]. 发电技术,2021,42(6):690-698. DOI:10.12096/j.2096-4528.pgt.21094.

[8]杜宇航,刘向民,王兴平等.塔式光热电站定日镜不同聚焦策略的影响分析[J].动力工程学报,2020,40(05):426-432.DOI:10.19805/j.cnki.jcspe.2020.05.012.

八、附录

代码一：

```
% 太阳角度
clear;close all;clc;

% N = [21 52 80 111 141 172 202 233 264 294 325 355];
% b = 2*pi.*(N - 1)/365;
% weidujiao = 0.006918 - 0.399912.*cos(b) + 0.070257.*sin(b)...
%     - 0.006758.*cos(2.*b) + 0.000907.*cos(3.*b) + 0.00148.*sin(3.*b);
% ST = [9.0 9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.5 14.0 14.5 15.0];
% w = pi.*(ST - 12)/12;
% A1 = cos(weidujiao)*cos(39.4).*cos(w) + sin(weidujiao)*sin(39.4);
% A2 = (sin(weidujiao) - A1*sin(39.4)) / (sqrt(1-A1.^2))*cos(39.4);


D = [-58 -28 0 31 61 92 122 153 184 214 245 275];
A = sin(2*pi.*D/365)*sin(2*pi*23.45/360);
ST = [9.0 9.5 10.0 10.5 11.0 11.5 12.0 12.5 13.5 14.0 14.5 15.0];
w = pi.*(ST - 12)/12;
A1 = sqrt(1-A.^2)*cos(39.4).*cos(w) + A*sin(39.4);
A2 = (A - A1*sin(39.4)) / (sqrt(1-A1.^2))*cos(39.4);
```

代码二：

```
% 对表格数据进行处理，求出距离原点的长度

clear;close all;clc;

data=readmatrix("D:\Race\ 数 学 建 模 \CUMCM2023Problems\A 题 \ 附件.xlsx","Range","A2:B1746");

P=data(:,1);
E=data(:,2);
% 距离原点的距离 RO
RO = sqrt(P.^2 + E.^2);
% 镜面中心到集热器中心的距离 DHR
DHR = sqrt(RO.^2 + 80*80);
% 大气透射率
NAT = 0.99321 - 0.0001176.*DHR + 1.97*10^(-8).*DHR.^2;

% 添加新的两列数据
```

```
new_data = [data, RO, DHR,NAT];

% 写入 Excel 文件
xlswrite('处理后数据.xlsx', new_data);
```

代码三：

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

def RED(FPath):
    return pd.read_excel(FPath)

def CMB(center, normal):
    halfSize = 3.0
    if not (normal[0] == 0 and normal[1] == 0):
        PV1 = np.cross(normal, [0, 0, 1])
    else:
        PV1 = np.cross(normal, [0, 1, 0])
    PV1 = PV1 / np.linalg.norm(PV1)
    PV2 = np.cross(normal, PV1)
    PV2 = PV2 / np.linalg.norm(PV2)

    p1 = center + halfSize * PV1 + halfSize * PV2
    p2 = center - halfSize * PV1 + halfSize * PV2
    p3 = center - halfSize * PV1 - halfSize * PV2
    p4 = center + halfSize * PV1 - halfSize * PV2

    return [p1, p2, p3, p4]

def RPI(rayOrigin, rayDir, planePoint, planeNormal):
    dot_product = np.dot(rayDir, planeNormal)
    if abs(dot_product) < 1e-6:
        return None
    t = np.dot(planePoint - rayOrigin, planeNormal) / dot_product
    if t < 0:
        return None
   IP = rayOrigin + t * rayDir
    returnIP

def isPIP(pt, polygon):
    NV = len(polygon)
```

```
    inter = 0
    for i in range(NV):
        p1, p2 = polygon[i], polygon[(i + 1) % NV]
        if (pt[1] > min(p1[1], p2[1])) and (pt[1] <= max(p1[1], p2[1])):
            x_inters = (pt[1] - p1[1]) * (p2[0] - p1[0]) / (p2[1] - p1[1]) + p1[0]
            if pt[0] < x_inters:
                inter += 1
    return inter % 2 == 1

def CO(data):
    occlusion = []
    for idx, row in data.iterrows():
        center = np.array([row["x 坐标 (m)"], row["y 坐标 (m)"], row["z 坐标 (m)"]])
        normal = np.array([row["Cosine X"], row["Cosine Y"], row["Cosine Z"]])
        boundaries = CMB(center, normal)
        occluded = False
        for j, other_row in data.iterrows():
            if idx != j:
                other_center = np.array([other_row["x 坐标 (m)"], other_row["y 坐
标 (m)"], other_row["z 坐标 (m)"]])
                other_normal = np.array([other_row["Cosine X"], other_row["Cosine
Y"], other_row["Cosine Z"]])
                other_boundaries = CMB(other_center, other_normal)
                intersection = RPI(center, normal, other_center, other_normal)
                if intersection is not None and isPIP(intersection, other_boundaries):
                    occluded = True
                    break
        occlusion.append(0 if occluded else 1)
    return occlusion

# 3D 可视化
def V3d(data, occlusionResults):
    x = data["x 坐标 (m)"].values
    y = data["y 坐标 (m)"].values
    z = data["z 坐标 (m)"].values
    colors = ["red" if occluded == 0 else "blue" for occluded in occlusionResults]
    fig = plt.figure(figsize=(12, 8))
    ax = fig.add_subplot(111, projection='3d')
    ax.scatter(x, y, z, c=colors, marker='o')
    ax.set_xlabel('X')
    ax.set_ylabel('Y')
    ax.set_zlabel('Z')
    ax.set_title('3D)')
    plt.show()
```

```
def main():
    FPath = "附件.xlsx"
    data = RED(FPath)
    occlusionResults = CO(data)
    data["Occluded"] = occlusionResults
    updated_FPath = "附件更新.xlsx"
    data.to_excel(updated_FPath, index=False)
V3d(data, occlusionResults)
```

代码四：

```
import numpy as np
import pandas as pd

def CR(center, cosine, size=6):
    DV = np.array(cosine)
    HD = (size / 2) * (DV / np.linalg.norm(DV))
    vertex1 = center + HD
    vertex2 = center - HD
    vertex3 = center + [-HD[0], HD[1], HD[2]]
    vertex4 = center - [-HD[0], HD[1], HD[2]]
    return vertex1, vertex2, vertex3, vertex4

def IA(rect1, rect2):
    rect1X = [point[0] for point in rect1]
    rect1Y = [point[1] for point in rect1]
    rect2X = [point[0] for point in rect2]
    rect2Y = [point[1] for point in rect2]

    overlapX = max(0, min(max(rect1X), max(rect2X)) - max(min(rect1X), min(r
ect2X)))
    overlapY = max(0, min(max(rect1Y), max(rect2Y)) - max(min(rect1Y), min(r
ect2Y)))

    return overlapX * overlapY

def PB(current, total, BL=50):
    progress = (current / total)
    arrow = '=' * int(round(progress * BL) - 1) + '>'
    spaces = ' ' * (BL - len(arrow))
    print(f"\rProgress: [{arrow + spaces}] {int(round(progress * 100))}%", end='')
```

```
def COI(data, customCosine=None):
    numPoints = len(data)
    resultsMatrix = np.zeros((numPoints, numPoints))
    rectangleArea = 6 * 6

    for idx in range(numPoints):
        target_point = data.iloc[idx]
        target_rect = CR(
                        np.array([target_point["x 坐 标 (m)"], target_point["y 坐
标 (m)"], target_point["z 坐标 (m)"]]),
            customCosine if customCosine else [target_point["Cosine X"], target_p
oint["Cosine Y"], target_point["Cosine Z"]])

        for j in range(idx + 1, numPoints):
            row = data.iloc[j]
            other_rect = CR(
                np.array([row["x 坐标 (m)"], row["y 坐标 (m)"], row["z 坐标 (m)"]]),
                 customCosine if customCosine else [row["Cosine X"], row["Cosine
Y"], row["Cosine Z"]])
            area = IA(target_rect, other_rect) / rectangleArea
            resultsMatrix[idx, j] = area
            resultsMatrix[j, idx] = area

        # 更新进度条
        PB(idx + 1, numPoints)

    print("\nDone!")
    return resultsMatrix

def main():
    # 自定义方向余弦
    customCosine = [0.5, 0.5, 0.5]

    # 读取指定路径的 Excel 文件
    dataPath = "附件更新.xlsx"
    dataAll = pd.read_excel(dataPath)

    # 仅选取倒数第二列为 0 的点
    dataSelected = dataAll[dataAll.iloc[:, -2] == 0]

    # 对这些选定的点进行操作
    IM = COI(dataSelected, customCosine)

    intersectionDf = pd.DataFrame(IM, index=dataSelected.index, columns=dat
```

```
aSelected.index)
    dataAll_combined = pd.concat([dataAll, intersectionDf], axis=1, join="outer")

    dataAll_combined.to_excel(dataPath, index=False)

if __name__ == "__main__":
    main()
```

代码五：

```
import numpy as np
import pandas as pd

def CR(center, cosineX, cosineY, size=6):
    DV_xy = np.array([cosineX, cosineY])
    HD = (size / 2) * (DV_xy / np.linalg.norm(DV_xy))
    vertex1 = center + np.append(HD, 0)
    vertex2 = center - np.append(HD, 0)
    vertex3 = center + np.append([-HD[0], HD[1]], 0)
    vertex4 = center - np.append([-HD[0], HD[1]], 0)
    return vertex1, vertex2, vertex3, vertex4

def IA(rect1, rect2):
    rect1X = [point[0] for point in rect1]
    rect1Y = [point[1] for point in rect1]
    rect2X = [point[0] for point in rect2]
    rect2Y = [point[1] for point in rect2]

    overlapX = max(0, min(max(rect1X), max(rect2X)) - max(min(rect1X), min(r
ect2X)))
    overlapY = max(0, min(max(rect1Y), max(rect2Y)) - max(min(rect1Y), min(r
ect2Y)))

    return overlapX * overlapY

def PB(current, total, BL=50):
    progress = (current / total)
    arrow = '=' * int(round(progress * BL) - 1) + '>'
    spaces = ' ' * (BL - len(arrow))
    print(f"\rProgress: [{arrow + spaces}] {int(round(progress * 100))}%", end='')

def COI(data):
    numPoints = len(data)
```

```
    resultsMatrix = np.zeros((numPoints, numPoints))
    rectangleArea = 6 * 6

    for idx in range(numPoints):
        target_point = data.iloc[idx]
        target_rect = CR(
                        np.array([target_point["x 坐 标 (m)"], target_point["y 坐
标 (m)"], target_point["z 坐标 (m)"]]),
            target_point["Cosine X"], target_point["Cosine Y"])

        for j in range(idx + 1, numPoints):
            row = data.iloc[j]
            other_rect = CR(
                np.array([row["x 坐标 (m)"], row["y 坐标 (m)"], row["z 坐标 (m)"]]),
                row["Cosine X"], row["Cosine Y"])
            area = IA(target_rect, other_rect) / rectangleArea
            resultsMatrix[idx, j] = area
            resultsMatrix[j, idx] = area

        PB(idx + 1, numPoints)

    print("\nDone!")
    return resultsMatrix

def main():
    dataPath = "附件更新.xlsx"
    dataAll = pd.read_excel(dataPath)

    dataSelected = dataAll[dataAll.iloc[:, -2] == 0]

    IM = COI(dataSelected)

    intersectionDf = pd.DataFrame(IM, index=dataSelected.index, columns=dat
aSelected.index)
    dataAll_combined = pd.concat([dataAll, intersectionDf], axis=1, join="outer")

    dataAll_combined.to_excel(dataPath, index=False)

if __name__ == "__main__":
    main()
```