

基于蒙特卡洛方法及贪心算法的定日镜场发电功率的优化设计

摘要

本文研究了塔式太阳能光热发电站定日镜场的参数设置以及最大年平均出热功率，根据**反射定律**和**光线追迹法**及定日镜场的不同参数以及不同时间条件下太阳辐射辐照强度，对整个定日镜场中的 1750 个定日镜建立三维物理模型，通过对太阳高度角，方位角，法相直接辐射照强度 DNI，定日镜场输出热功率表以及定日镜的光学效率的计算求解定日镜场的年均输出功率，根据定日镜场部分参数如吸收塔位置坐标和定日镜尺寸数目等是否恒定，采取**贪心算法**求解在一定约束条件下的最优解，并在数学上利用数学归纳法和递归式证明问题具有最优子结构，从而求得全局条件下最大单位镜面面积年平均输出热功率以及相关参数。

问题一，首先根据题目所给定日镜场的经纬度信息和海拔信息，求解出每月 21 日不同时点的法向直接辐射辐照强度 DNI 共 60 个，然后再依据给定的定日镜尺寸，安装高度以及定日镜中心位置，建立以吸收塔为中心的圆形定日镜场三维物理模型，并依据太阳高度以及相邻定日镜几何关系确定阴影遮挡损失以及余弦损失，从而计算得到定日镜场的年平均光学效率为 29.48%，年平均输出热功率 18.24MW 以及单位镜面面积年平均输出热功率为 $0.29kW/m^2$ 。其中，我们将阴影遮挡分为定日镜入射遮挡，反射遮挡以及吸收塔投影遮挡三种情况，利用**反射定律**并在三维空间中利用矩阵变换等方法进行计算。具体的模型以及其它数据详见正文。

问题二，基于问题一的模型基础，在要求定日镜场的额定年平均输出热功率为 60MW 等条件下，我们建立**优化模型**进行功率求解。因为定日镜数量作为离散值本身是可变参数之一，我们选择**贪心算法**为基本优化策略，以**蒙特卡罗方法**搜索出一系列符合条件的位置点取最优结果加入现有定日镜场序列，在当前定日镜场的情况下依据**迭代优化算法**不断更新的定日镜以及相关参数直到达到单位镜面面积年平均输出热功率的最大值，从而保证找到最优解，求解得到在达到额定功率的条件下，使用的定日镜数目为 6963 片，单位镜面面积平均输出热功率为 $0.26kW/m^2$ 。

问题三，在问题二的基础上，引入定日镜尺寸及安装高度等变量进一步进行优化，根据问题一和问题二的结果，我们在求解最优情况下的参数时对定日镜的光学参数进行一定的优化，从而相较于问题二有更快的迭代速度以及更好的迭代效果，再次利用**贪心算法**求得使用的定日镜数目为 6551 片，单位镜面面积平均输出热功率为 $0.21kW/m^2$ 。

关键字： 塔式太阳能光热发电站 贪心算法 确定性有约束优化 迭代优化算法 光线追迹法

一、问题重述

1.1 问题背景

塔式太阳能光热发电是一种低碳环保的新型清洁能源技术。以该技术为基础技术的塔式太阳能光热发电站，是我国构建新能源为主体的新型电力系统、实现“碳达峰，碳中和”目标的重要建设项目。定日镜作为其核心组件，在塔式电站中起着关键作用。定日镜的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，其方位角、俯仰角可分别通过纵向、水平转轴调节。两转轴的交点即定日镜中心的高度，被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集热器上，加热导热介质，并将太阳能以热能形式储存，然后通过热交换实现从热能到电能的转化。

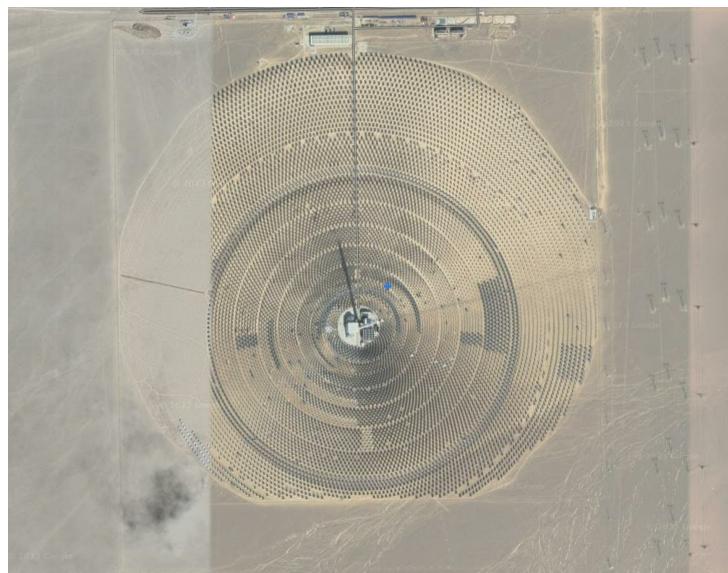


图1 塔式电站定日镜场卫星鸟瞰照片

1.2 问题要求

现计划于东经 98.5° ，北纬 98.5° ，海拔 $3000m$ 处建立圆形定日镜场，其中，规划吸收塔高度为 $80m$ ，集热器采用高 $8m$ 、直径 $7m$ 的圆柱形外表受光式集热器，其周围 $100m$ 范围内建造厂房， $100m$ 至 $350m$ 范围内安装若干定日镜。

问题一：在前提条件定日镜场经纬度，海拔高度，形状以及吸收塔高度参数下，若吸收塔位于圆形定日镜场的中心，给定定日镜尺寸、安装高度及所有定日镜中心的位置，计算当前条件下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。

问题二：在问题一的模型基础上，设定定日镜场的额定年平均输出热功率（以下简称额定功率）为 $60MW$ ，假定所有定日镜尺寸及安装高度相同。设计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情况下单位镜面面积年平均输出热功率达到最大值。

问题三：在问题二的模型基础上，若定日镜尺寸与安装高度可以不同，重新设计定日镜场的各个参数，使得定日镜场在达到问题二中额定年平均输出热功率的条件下单位镜面面积年平均输出热功率达到最大值。

二、问题分析

2.1 对问题一的分析

我们采用**光线追迹法**对光热电站建立聚光模型，即通过模拟跟踪光线从太阳产生到反射到吸热器表面的全过程，来进行分析计算。首先对整个定日镜场的日照辐射进行分析，根据附录的公式依次计算每月 21 日太阳赤纬角，太阳时角，太阳高度角以及太阳方位角，依据三维空间中太阳方向角和高度角确定太阳的方位，将太阳的入射光线的反方向（便于后续计算）用方向向量 \mathbf{v}_{sun} 表示，进一步利用太阳高度角求得法向直接辐射辐照度 DNI ，并将计算结果留存以便后续计算。

因为定日镜场在发电时不可避免会产生各种形式的损失，我们建立了定日镜的光学效率模型，其中考虑的影响定日镜光学效率的主要有五个参数：阴影遮挡效率 η_{sb} ，余弦效率 η_{cos} ，大气透射率 η_{at} ，集热器截断效率 η_{trunc} 以及镜面反射率 η_{ref} 。其中阴影遮挡损失主要包括三部分：吸收塔阴影对镜场造成的损失，后排定日镜在接受太阳光线时被其它定日镜遮挡造成的阴影损失以及反射太阳光时被其它定日镜遮挡而未能聚焦于吸收塔的挡光损失；余弦损失即太阳光入射方向与镜面采光法线方向不平行引起的接收能量损失；大气投射损失为太阳光传播过程中受空气中粉尘、颗粒等影响造成的衰减；集热器截断损失为定日镜反射的光斑不能完全照射到吸收器而产生部分溢出的损失；镜面反射损失为反射镜本身性质反射率以及干净程度的综合损失。

然后对每一个定日镜进行分析，因为问题一中已知 1750 面定日镜的坐标位置以及吸收塔的坐标位置，我们很容易根据反射定律求得每一面定日镜的法向量以及余弦效率 η_{cos} 。其中，考虑到实际生产生活中定日镜反射光斑简单聚焦在吸热器中心会有超温风险，我们采用了**交替偏移聚焦策略**，即每面定日镜反射光斑聚焦在吸热器上的不同位置。在确定定日镜的聚焦位置后，我们可以求得定日镜中心与聚焦中心的直线距离， d_{HR} ，从而确定定日镜的大气损失 η_{at} 。随后，我们计算了每个定日镜与其余定日镜的距离，并以当前定日镜为中心，以定日镜与太阳连线为轴线考虑 90° 范围内定日镜对其可能造成的遮挡损失。

其中，**计算阴影遮挡损失是本题模型中的核心部分**，其基本思路为计算当前研究的

定日镜的任意一点顺着入射光线或者反射光线是否会落入其它定日镜的区域内。具体的计算过程就是利用坐标系变换与矩阵计算根据前述确定的定日镜法向量确定矩形四个顶点在底面坐标系的坐标，同时建立镜面坐标系，求得地面坐标系与不同镜面坐标系的转换矩阵 T ，则为了计算当前研究镜面上某一点在入射光线或者反射光线的方向上是否有被遮挡，我们进行以下计算：先将当前研究定日镜的一点通过转换为地面坐标系上一点，再转换为其它可能产生遮挡的定日镜坐标系之中的坐标，判断直线是否穿过定日镜矩形，即是否会产生遮挡，计算得到遮挡的总面积。关于吸收塔投影产生的遮挡即将塔投影到 XOY 平面上探究与定日镜是否会产生遮挡，这部分遮挡面积计算时较为简单。

最后，根据求得的每面定日镜的光学效率，法向直接辐射辐照度以及固定尺寸下的定日镜面积可以求得定日镜场的输出热功率。

2.2 对问题二的分析

在问题一的模型基础上，当定日镜参数都可变的情况下，在功率和几何约束的条件下，尝试求解定日镜单位镜面面积的年平均输出热功率尽量大的情况，即转化为一个确定性有约束优化。在问题一物理模型确定的基础之上，问题主要集中于算法层次。我们将该优化模型与神经网络进行类比，即定日镜单位镜面面积的年平均输出热功率确定为不同参数的一个函数，在一个高维的离散与连续同时存在的空间中求得极值。对于离散类型的参数如定日镜的数目，我们采取了基于蒙特卡洛随机算法的贪心策略，即每次随机生成一部分新的定日镜的坐标，在其中计算出光学效率最大的定日镜添加到当前已存在的定日镜的集合之中；随后进一步探究连续型参数的取值时，我们考虑在高维空间中选择增长速率最快的方向，即迭代优化算法（随机梯度下降法），同时检验是否满足各个约束条件，得到极值。

2.3 对问题三的分析

问题三在问题二的基础上，引入定日镜尺寸以及安装高度等更多可变变量，在分析了前述问题的数据以及为了避免计算复杂度成几何量级增长，我们将定日镜的部分光学损耗取定为先前数据的均值，随后在上述模型的基础上继续求解得到结果。

三、 模型假设

1. 假设定日镜场天气晴朗，不考虑阴雨天气对 DNI 造成的影响
2. 假设假设在能量转换过程中不考虑热损失，即所有太阳能都被完全转化为其他形式的能量
3. 假设所有定日镜的镜面形状都是矩形，没有凹凸不平或者其它缺陷

四、 符号说明

表 1 符号说明表

符号	说明	单位
α_s	太阳高度角	°
γ_s	太阳方位角	°
φ	当地纬度	°
δ	太阳赤纬角	°
ω	太阳时角	rad/s
H	定日镜场的海拔高度	km
G_0	太阳常数	kW/m ²
DNI	法向直接辐射辐照强度	kW/M ²
E_{field}	定日镜场输出热功率	kW
η	定日镜光学效率	无
η_{at}	大气透射率	无
η_{cos}	余弦效率	无
η_{sb}	阴影遮挡效率	无
η_{ref}	镜面反射率	无
η_{trunc}	集热器截断效率	无
d_{HR}	定日镜中心与聚焦中心的直线距离	m
\mathbf{v}_{sun}	阳光传播方向方向向量	
$\mathbf{v}_{collect}$	反射光传播方向的方向向量	
N	定日镜场定日镜总数量	面
A_i	第 i 个定日镜的反射镜反射面面积	m ²
x	当前研究的定日镜的横坐标	m
y	当前研究的定日镜的纵坐标	m
z	当前研究的定日镜的安装高度	m
x_{target}	当前研究的集热器中心的横坐标	m
y_{target}	当前研究的集热器中心的纵坐标	m
z_{target}	当前研究的集热器中心的安装高度	m
$x(0)$	当前研究的定日镜的初始横坐标	m
$y(0)$	当前研究的定日镜的初始纵坐标	m
$z(0)$	当前研究的定日镜的初始安装高度	m
h_{mirror}	当前研究的定日镜的长度	m

表1 符号说明表 (续)

符号	说明	单位
d_{mirror}	当前研究的定日镜的宽度	m

五、模型的建立与求解

5.1 问题一：求解定日镜场的平均光学效率与平均热功率

5.1.1 模型准备

1) 太阳高度角与方位角的计算：

太阳的高度角与方位角决定了相邻定日镜之间以及定日镜与吸收塔之间的遮挡关系以及太阳辐射能量的大小。计算并存储题目要求的各个时间点下太阳的高度角与方向角数值，方便在后续的问题求解中直接进行调用。

根据题目给定当地经纬度与测量时间数据，可计算出太阳高度角 α_s ，与太阳方位角 γ_s ：

$$\begin{aligned}\sin \alpha_s &= \cos \delta \cos \varphi \cos \omega + \sin \delta \sin \varphi \\ \cos \gamma_s &= \frac{\sin \delta - \sin \alpha_s \sin \varphi}{\cos \alpha_s \cos \varphi}\end{aligned}$$

其中， φ 为当地纬度，北纬为正； ω 为太阳时角

$$\omega = \frac{\pi}{12}(ST - 12),$$

其中 ST 为当地时间， δ 为太阳赤纬角

$$\sin \delta = \sin \frac{2\pi D}{365} \sin \left(\frac{2\pi}{360} 23.45 \right),$$

其中 D 为以春分作为第 0 天起算的天数。

2) 法向直接辐射辐照度 DNI 的计算：

法向直接辐射辐照度（单位： kW/m^2 ）是指地球上垂直于太阳光线的平面单位面积上、单位时间内接收到的太阳辐射能量。法向直接辐射辐照度的计算，是后续定日镜热功率的计算中必不可少的环节。计算并存储题目要求的各个时间点下法向直接辐射辐照度数值，方便在后续的问题求解中直接进行调用。

根据所有要求时间点下太阳高度角 α_s 与当地海拔高度 H ，可算出该时间点的法向直接辐射辐照度 DNI：

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

其中, $G_0 = 1.366 \text{ kW/m}^2$, 为太阳常数; H 为海拔高度 (单位 km)。

5.1.2 模型建立

由于定日镜场中不同定日镜与集热器的相对位置不同, 定日镜将阳光反射到集热器上所需的角度、阳光从定日镜到集热器的传播距离均有不同, 因而不同定日镜的光学效率不同。因此, 需将定日镜安装位置作为参数, 分别考虑单个定日镜对阳光的反射汇聚情况以及其与相邻定日镜的遮挡、截断关系, 再求取平均值。

5.1.2.1 单个定日镜反射模型

定日镜的光学效率的计算中, 包含定日镜阴影遮挡效率、余弦效率、大气透射率、集热器截断效率与镜面反射率的计算。其中, 在题给条件下, 定日镜的余弦效率只与定日镜将阳光反射至集热器时光线与反射镜所成角余弦值大小有关; 大气透射率只与镜面中心到集热器中心的距离有关; 镜面反射率可取为一常数。故建立只考虑某安装位置下单个定日镜的反射模型, 计算其反射镜朝向的法向量, 其反射阳光的余弦效率与大气透射率。

1) 大气透射率的计算

大气透射率指光线从地球大气中穿过的能量比例, 在题给条件下的计算式为

$$\eta_{at} = 0.99321 - 0.0001176d_{HR} + 1.97 \times 10^{-8} \cdot d_{HR}^2 \quad (d_{HR} \leq 1000)$$

其中 d_{HR} 表示镜面中心到集热器中心的距离 (单位: m), η_{at} 为大气透射率。

则只需算出镜面中心到集热器中心的距离, 即可求出该定日镜的大气透射率。

$$d_{HR} = \sqrt{(x - x_{target})^2 + (y - y_{target})^2 + (z - z_{target})^2}$$

其中, x 、 y 、 z 分别为定日镜中心的横、纵、高度坐标, x_{target} 、 y_{target} 、 z_{target} 为集热器中心的横、纵、高度坐标。题目条件下, 有 $x_{target} = y_{target} = 0$, 且恒有 $z_{target} = 76\text{m}$ 。

将定日镜中心坐标代入公式即可求得大气透射率。

2) 反射镜朝向及余弦效率的计算

规定反射镜平面法向量垂直反射镜平面指向其朝向方向, 则计算出光线折射前后的方向向量, 相加单位化后即反射镜平面法向量:

$$\mathbf{v}_{sun} = \begin{bmatrix} \sin \alpha_s \cos \gamma_s \\ \cos \alpha_s \cos \gamma_s \\ \sin \gamma_s \end{bmatrix}$$

$$\mathbf{v}_{collect} = \begin{bmatrix} x - x_{target} \\ y - y_{target} \\ z - z_{target} \end{bmatrix} / d_{HR}$$

其中， \mathbf{v}_{sun} 为反射前光线传播方向的反向量， $\mathbf{v}_{collect}$ 为反射后光线传播方向向量。

由光线方向向量与反射镜平面法向量，可计算反射镜的余弦效率

$$\eta_{cos} = \left(\frac{1}{2} \times (\mathbf{v}_{sun} \cdot \mathbf{n}_{reflect} + 1) \right)^{\frac{1}{2}}$$

其中， η_{cos} 为定日镜的余弦效率。

5.1.2.2 包含遮挡截断关系的定日镜三维模型

定日镜的光学效率的计算中，要计算定日镜的阴影遮挡效率与集热器极端效率，均需考虑定日镜与其相邻定日镜之间的位置关系。故利用算出的反射镜朝向的法向量和题给定日镜安装位置坐标，建立模型计算定日镜之间的遮挡截断关系。

1) 定日镜的遮挡关系

对于阳光接收过程的遮挡，因为遮挡过程中的两平面镜距离较近，其中锥形光线导致的误差可以忽略不计，故在此过程中我们将光束视为平行光。我们以入射光为基准，即建立以入射方向向量为 z 轴的坐标系，利用线性代数方法将地面坐标系的坐标信息转换到入射光坐标系，随后利用同一组平面图形在不同坐标表示、变换下相交部分面积与各自面积之比不变的性质（即线性变换的不变性），将前后两定日镜面积投影至入射光坐标系中过原点的平面，计算相交面积与被遮挡定日镜变换后面积之比，即得到入射光遮挡损失。而对于反射光遮挡损失，则沿用上述思路，建立反射光坐标系，投影后得到反射光遮挡损失。而对于收集塔阴影造成的损失，则将一定时刻各定日镜面积投影到地面，计算其与收集塔地面阴影相交面积，再计算损失。

2) 定日镜的截断关系

对于截断损失，由于定日镜距离收集塔距离较远，故必须考虑锥形光束带来的影响。采用与遮挡关系类似的思路，我们以反射光方向向量为基准建立反射光坐标系，将收集器和定日镜投影在此坐标系下的收集塔垂直平面。与此同时，计算距离与光锥角产生的投影面积增广，修正上述结果。最后得到相交面积，进而得到截断损失。

5.1.3 模型求解

本文关于定日镜场平均光学效率与平均热效率的计算步骤如下：

Step1：数据的导入与初始化

从附件导入所有定日镜的位置坐标。设置 $\eta_{ref} = 0.92$, $z_{target} = 76m$, $x_{target} = y_{target} = 0$ 略去不写。

Step2：计算太阳角与 DNI

计算要求时间点的太阳方位角 α_s 、高度角 γ_s ，计算其法向直接辐射辐照度 DNI。

Step3：计算光学效率

关于每一日期所有时刻对所有定日镜进行遍历，利用公式依次求出定日镜大气透射率 η_{at} 、余弦效率 η_{cos} 。同时，将剩余定日镜关于与当前定日镜的距离递增进行排序，计算阴影遮挡效率 η_{sb} 、集热器截断效率 η_{trunc} 。

依据附录光学效率 η 计算公式

$$\eta = \eta_{sb}\eta_{cos}\eta_{at}\eta_{trunc}\eta_{red}$$

求出所有定日镜在每一时间点的光学效率。

Step4：求平均值

对定日镜余弦效率、遮挡效率、截断效率、光学效率关于每一日期中的所有测量时刻求平均值并填入表 2。

Step5：求定日镜场热效率

依据附录定日镜场的热效率公式计算

$$E_{field} = DNI \cdot \sum_i^N A_i \eta_i$$

计算单位面积镜面平均输出热效率：

$$\bar{E} = \frac{E_{field}}{\sum_i^N A_i}$$

其中， E_{field} 为定日镜场的热效率； N 为定日镜总数（单位：面）； A_i 为第 i 面定日镜采光面积（单位： m^2 ）； η_i 为第 i 面镜子的光学效率； \bar{E} 为单位面积镜面平均输出热效率。

Step7：求年平均值

关于所有测量日期计算定日镜场年平均输出热功率，并填入表 3 中。

5.1.4 结果分析

由上述步骤可算出结果如表 2 与表 3 所示：

表 2 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/m^2)
1月 21 日	0.302435351	0.778065316	0.995842124	0.439517758	0.300137334
2月 21 日	0.306170097	0.790214368	0.99686866	0.437653422	0.301466679
3月 21 日	0.306745884	0.79125255	0.997093591	0.43780238	0.299870597
4月 21 日	0.30670101	0.791301242	0.997063416	0.437724645	0.300304847
5月 21 日	0.305525628	0.788309129	0.996673507	0.437873434	0.3016993
6月 21 日	0.299935044	0.768860438	0.99505366	0.44145213	0.298027875
7月 21 日	0.288432233	0.714257346	0.992350523	0.458220374	0.286083045
8月 21 日	0.285531958	0.696298873	0.990395767	0.466230497	0.280673554
9月 21 日	0.281469034	0.677563264	0.990883249	0.472072505	0.274299049
10月 21 日	0.281468619	0.67771504	0.990873412	0.471970772	0.274532639
11月 21 日	0.285985342	0.699690463	0.99105989	0.464395862	0.28156309
12月 21 日	0.289350392	0.7172844	0.990878647	0.458419034	0.287230742

表 3 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)	单位面积年平均输出热效率 (kW/m^2)
0.294979216	0.740901036	0.993753037	0.451944401	18.24862761	0.290490729

由结果可以看出，定日镜场的每月各平均光学效率与平均输出热功率随日期有细微波动。其中，定日镜场光学效率年平均值为 29.48%，年平均输出热功率为 $18.24MW$ ，单位镜面面积年平均输出热功率为 $0.29kW/m^2$ 。

在问题一题给条件下，定日镜场的光学平均光学效率较低。太阳辐射总量由当地地理位置与时间决定，为不可调变量。通过调整定日镜数目、尺寸、安装高度、位置分布等参数，可使得定日镜场的平均光学效率得到优化，进而提高其单位面积年平均热功率，使得资源的利用效率得到优化。

5.2 问题二：定日镜尺寸和高度相同情况下单位镜面面积年平均输出热功率

5.2.1 模型准备

在前述光线追迹法建立起的定日镜场聚光模型的基础上，我们将定日镜光学效率封装为一个函数，并且以定日镜的各种参数如位置坐标，安装高度以及定日镜的长度和宽度为参数，便于后续基于贪心算法的优化模型的实现。首先，我们确定每个定日镜面的初始高度 z_0 为 $6m$ ，初始长度 $h_{mirror0}$ 为 $8m$ ，初始宽度 $d_{mirror0}$ 为 $4m$ ，并且确定贪心算

法的第一个初始点的坐标为 $(101m, 0m)$ 。在贪心算法的更新过程中，我们采用蒙特卡洛随机算法更新，即随机生成 20 个测试点，并分别计算每个测试点的光学效率，选择最大的那个点更新到定日镜集合之中，并且更新当前定日镜场的年平均输出热功率。

5.2.2 模型建立与求解

首先进行相关参数的初始化，其中包括定日镜场中定日镜的安装高度，长度宽度等参数，随后确定一个初始点并进入判断年平均输出热功率的循环之中。以初始点为基准，随机生成一组 20 个的蒙特卡洛采样点，并判断每个点是否会与当前已存在的定日镜产生冲突（即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的距离比镜面宽度多 $5m$ 以上，随后求出测试集中每个点对应的光学效率，其中的阴影损耗基于当前已经存在的定日镜进行计算，随后选择光学效率最高的定日镜加入到已存在的定日镜集合之中。

当确定新添加的定日镜的位置之后，再对定日镜的长宽尺寸以及安装高度重新进行优化，即在镜面宽度不小于镜面高度，镜面边长在 $2m$ 到 $8m$ 之间，安装高度在 $2m$ 到 $6m$ 之间的范围中根据 SGD 随机梯度下降法进行迭代调优，以单位镜面面积年平均输出热功率尽量大为目标变量，最后当功率确定并且优化到极值时停止循环。

将上述思路编写成 Matlab 代码并进行运算得到结果。

5.2.3 结果分析

由上述步骤计算、优化定日镜场的参数设计，得到参数列表如表 6 所示。计算该参数设计下定日镜场的月、年平均光学效率及输出工具，结果如表 4 与表 5 所示：

表 4 问题二每月 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/m^2)
1月 21 日	0.272084206	0.730637727	0.983786636	0.441927056	0.270016809
2月 21 日	0.274126852	0.736111947	0.983732275	0.441958065	0.269915686
3月 21 日	0.273981277	0.735749267	0.983698676	0.441956201	0.267840364
4月 21 日	0.274078677	0.735977952	0.983685087	0.441982047	0.268362843
5月 21 日	0.273850034	0.735462546	0.983718767	0.441907682	0.270420403
6月 21 日	0.270351887	0.725922815	0.983792338	0.44196288	0.268632827
7月 21 日	0.259755619	0.697565084	0.983720388	0.441935424	0.257639993
8月 21 日	0.255753957	0.686701178	0.98382499	0.441964097	0.251402233
9月 21 日	0.251742422	0.676022192	0.983740954	0.441941691	0.245329676
10月 21 日	0.251819597	0.67619676	0.983806625	0.441933544	0.245614232
11月 21 日	0.256478664	0.688707584	0.983763905	0.441952673	0.25251268
12月 21 日	0.260419635	0.699268439	0.983753716	0.441970908	0.258511919

表 5 问题二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)	单位面积年平均输出热效率 (kW/m^2)
0.264536902	0.710360291	0.98375203	0.441949356	60.04727536	0.260516639

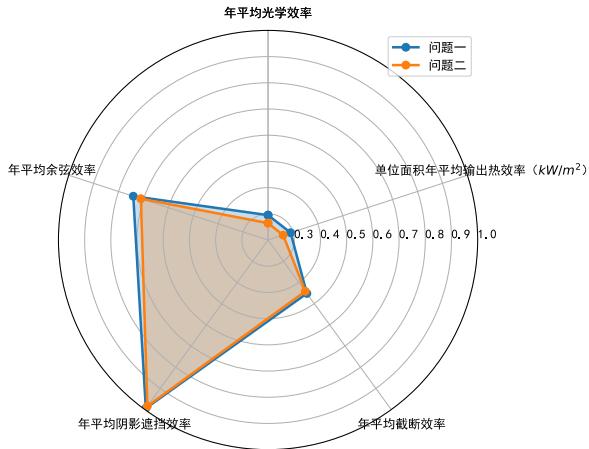
表 6 问题二设计参数表

吸收塔位置坐标	定日镜尺寸 (宽 × 高)	定日镜安装高度 (m)	定日镜总面数	定日镜总面积 (m^2)
(0m,0m)	8m × 4m	6m	6963	22816

由结果可以看出，在将定日镜数目增加到 6963 面后，定日镜场的年平均输出功率升至能够实现达到额定功率 60MW 的水平。但是，定日镜场的平均光学效率与单位面积平均输出热功率均有轻微下降，如图 2 所示。

经过分析，本文对于平均光学效率以及单位面积平均输出热功率降低的现象给出如下推断：由于额定功率的要求，定日镜阵列中定日镜数目增多，加之场地的限制，使得有相当一部分定日镜无法找到合适的位置从而达到足够高的光学效率，使得平均光学效率下降，进而使得单位面积平均热效率下降。

关于该推断，基于计算数据绘出问题二定日镜单位面积输出功率随定日镜坐标分布三维散点图（图 3）与问题二定日镜单位面积输出功率与定日镜中心距集热器中心距离



分布散点图（图 4）。从图中可较为直观地看出，在相同距离或相似坐标位置下，定日镜单位面积输出功率有所参差。若除去额定功率的约束，减少定日镜数目，则除去光学效率与平均热功率相对较低的定日镜，即可提高定日镜的平均光学效率与平均单位面积输出功率，但同时，定日镜场的总平均功率可能将有所下降。

因此，可认为效率的下降是与定日镜场总功率平衡的结果。在本题的设计中，本文在保证满足定日镜场额定的情况下，将单位面积平均输出功率提高至极大值，为最优的一种设计方案，具体设计细节见附件。

值得注意的是，若不规定所有定日镜尺寸相同且安装高度相同且确定，则模型将多出几个可以调节的参数，经过对应的优化后，平均光学效率和单位面积平均输出效率或将有进一步提升，具体情况将在问题三中进行分析。

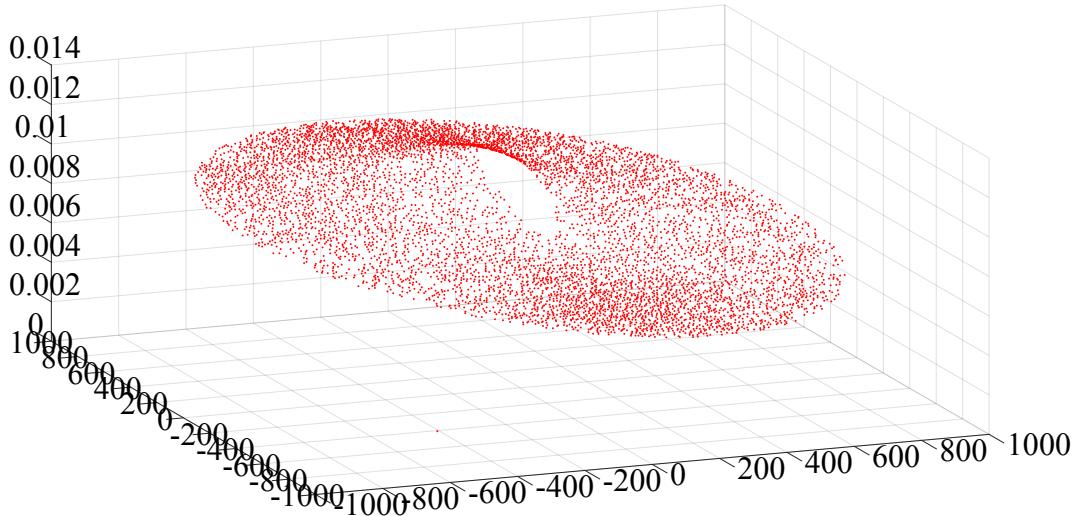


图3 问题二定日镜单位面积输出功率随定日镜坐标分布三维散点图

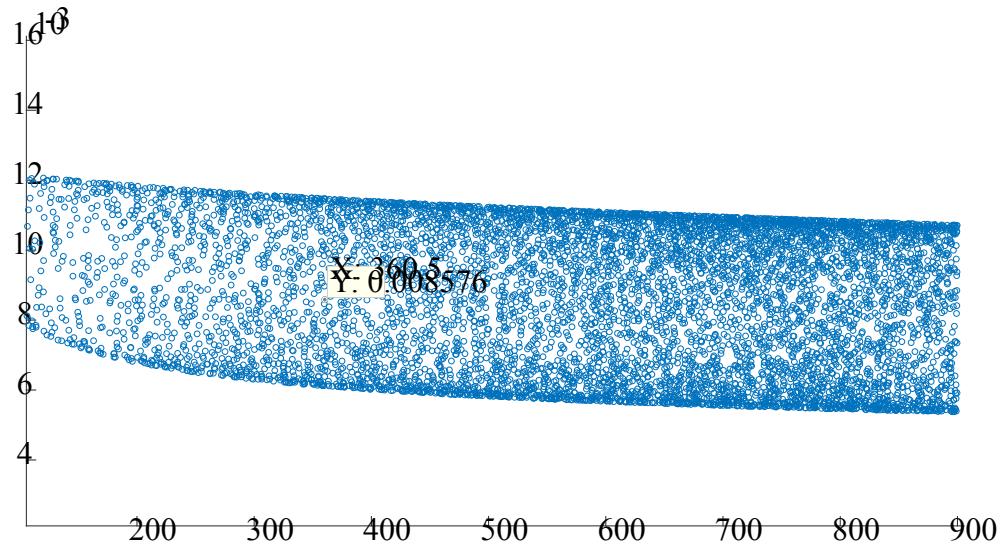


图4 问题二定日镜单位面积输出功率与定日镜中心距集热器中心距离分布散点图

5.3 问题三：定日镜尺寸可不同情况下单位镜面面积年平均输出热功率

5.3.1 模型准备

在问题二的基础上，将定日镜尺寸以及安装高度引进为新的自由变量，即提高每个定日镜的参数维度，同时仍和问题二一样采用封装好的物理模型，将解决问题的重心放在优化模型的算法上。首先我们确定目标变量仍为单位定日镜面面积，可变参数为吸收塔的位置坐标，定日镜的数目，每块定日镜的长度宽度，位置坐标和安装高度。并引入

一定的约束条件如相邻定日镜的距离约束，定日镜长宽的约束进行求解。

5.3.2 模型建立

定义目标函数：将单位镜面面积年平均输出热功率作为目标函数。这个函数将取决于定日镜场的各个参数，包括定日镜的尺寸和安装高度。确定约束条件：将问题一中的功率和几何约束条件转化为数学约束条件，确保设计的定日镜场满足这些约束。

5.3.3 模型求解

采用蒙特卡洛随机算法的贪心策略：在离散类型的参数（如定日镜的数目）上，使用蒙特卡洛随机算法生成一部分新的定日镜坐标，并计算其中光学效率最大的定日镜。将该定日镜添加到当前已存在的定日镜集合中，重复这个过程，直到满足额定功率条件。迭代优化算法（如随机梯度下降法）：对于连续型参数（如定日镜尺寸和安装高度），在高维空间中选择增长速率最快的方向，并检验是否满足约束条件。使用迭代优化算法，例如随机梯度下降法，逐步优化这些参数，以求得极值。

5.3.4 结果分析

由以上算法对模型进行计算优化，得到参数列表如表 9 所示，由该参数出发计算每月 21 日、年平均光学效率与输出热功率，得出结果如表 7，表 8 所示。根据每一个设计定日镜数据绘制单位面积输出功率随定日镜坐标分布三维散点图如图 5 所示，绘制定日镜单位面积输出功率与定日镜中心距集热器中心距离如图 6 所示。

问题三中所用定日镜数目 6551 面，相较问题二的 6963 面少，而比较问题三数据表 8 与问题二数据表 5 可以看出，问题三设计中定日镜的平均光学效率与平均输出效率均较低，然而定日镜总输出功率相较于问题二中高。通过图 5 分析其原因，可以看出存在大量定日镜，其单位面积平均输出热功率明显较其它定日镜低，进而导致了定日镜场平均效率的低下。只需从系统进一步筛选除去或调整这些平均输出效率低的镜，即可提高定日镜场平均光学效率与平均输出热效率，以达成对模型的进一步优化。

同时，值得注意的是，从图 5，以及表 5 与表 8 的对比数据中可以看出，定日镜场中仍有大量定日镜的单位面积平均输出热效率明显较问题一、二情况高，这说明，在对定日镜尺寸与安装高度进行调整后，定日镜的平均效率仍得到了较大的优化。

表 7 问题三每月 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/m^2)
1月 21 日	0.200880108	0.678646684	0.987728121	0.352155271	0.196750297
2月 21 日	0.232934648	0.690562894	0.995488666	0.398174044	0.230507387
3月 21 日	0.203375887	0.706889003	0.991242493	0.341072524	0.202155878
4月 21 日	0.171611849	0.72270881	0.986073803	0.282978155	0.169948628
5月 21 日	0.222484759	0.724379255	0.994722448	0.36283632	0.218417475
6月 21 日	0.205779132	0.723763627	0.992117067	0.336759664	0.201042903
7月 21 日	0.20528797	0.72438855	0.991507116	0.33587254	0.201563485
8月 21 日	0.221139935	0.722429242	0.993588051	0.36202946	0.219075894
9月 21 日	0.236308023	0.706140338	0.997294486	0.39431422	0.234878682
10月 21 日	0.21693428	0.689575422	0.992451598	0.372490744	0.214457092
11月 21 日	0.203467189	0.675510821	0.988331253	0.358127735	0.199061563
12月 21 日	0.216667381	0.674495424	0.991709264	0.380634865	0.210924827

表 8 问题三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)	单位面积年平均输出热效率 (kW/m^2)
0.21140593	0.703290839	0.991854531	0.356453795	61.45651157	0.208232009

表 9 问题三设计参数表

吸收塔位置坐标	定日镜尺寸 (宽 × 高)	定日镜安装高度 (m)	定日镜总面数	定日镜总面积 (m^2)
(0m,0m)	-	-	6551	213817.1

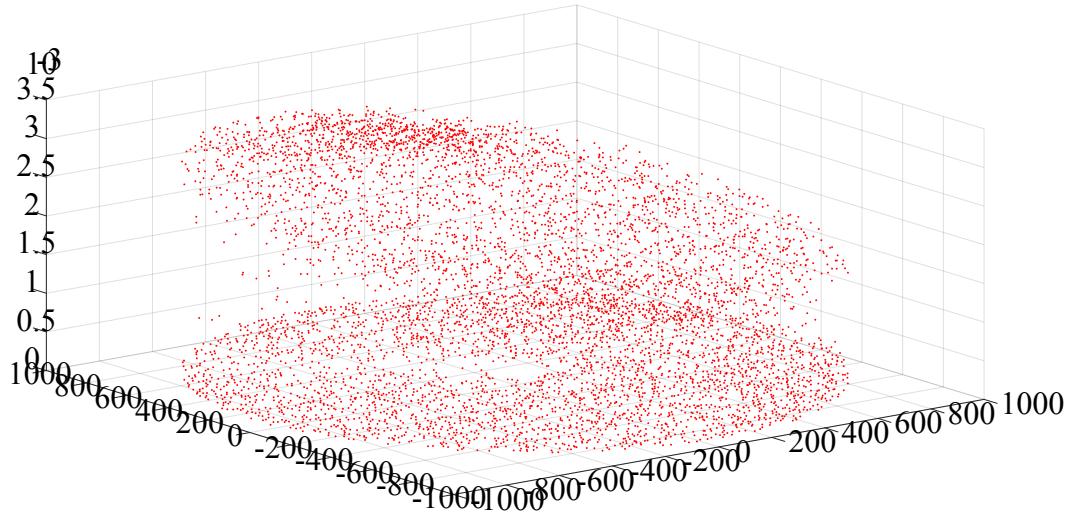


图 5 问题三定日镜单位面积输出功率随定日镜坐标分布三维散点图

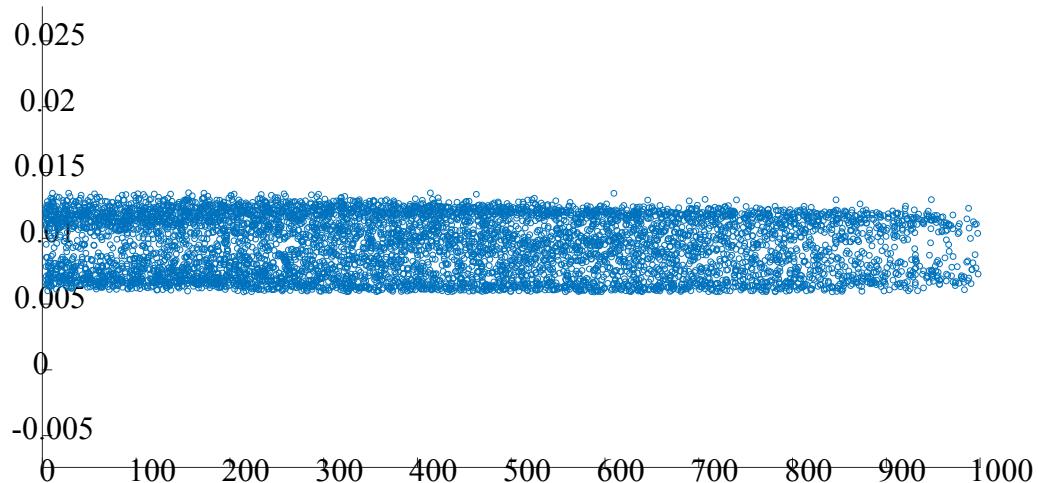


图 6 问题三定日镜单位面积输出功率与定日镜中心距集热器中心距离分布散点图

六、模型评价

6.1 模型的优点

本文中所建立的模型可以分为物理模型以及优化模型，其中物理模型尽可能地去描述真实世界中的定日镜场运行情况，我们从通过经纬度和时间海拔计算出了太阳光线的方向，同时依据实际生产生活需要在聚焦的时候采用了交替便宜聚光策略，并且在计算三维空间的时候建立的定日镜场聚光模型很好地从折射定律出发，依据三维空间中坐标

系矩阵变换的线性代数知识求解了定日镜场三种不同的遮光面积以及锥形太阳光线的影响，最后我们将物理模型很好地集成为一个简单的计算定日镜光学效率的函数，与后面重心在算法部分的优化模型进行了很好的区分。

优化模型的优点在于本题在处理多个变量有约束的优化模型时，先根据变量的类型进行区分，针对离散型的变量譬如定日镜的数目时采用基于蒙特卡洛算法的贪心策略，即将增加定日镜数目的步骤视作在每一步更新的过程中随机生成不同位置的定日镜并根据光学效率选择效率最高的加入到已有的定日镜集合中，随后针对其它连续的参数如定日镜长度宽度，安装高度等进行迭代优化算法，并探究是否会破坏相关的约束条件，得到最优解，并且，我们考虑到多个多维参数的优化过程时间复杂度的问题，在问题三引进新的变量的时候我们选择性地基于前述数据地分析取定部分光学效率的均值，在一定程度上提高了运行效率并且保证了计算结果的准确性。

6.2 模型的缺点

在该模型中，存在一定的缺点，具体体现在

1. 因为缺失实地数据，没有考虑到天气状况对定日镜场发电的影响，所有的日照辐射强度都是基于晴天的条件下计算的。
2. 定日镜场阴影遮挡损失计算过程较为复杂，在多参数优化的情况下时间开销过大。
3. 在贪心算法和迭代优化算法的初始值确定的时候没有进行一定的探究，随机选取一个点或取定中值即开始迭代

参考文献

- [1] 张平, 奚正稳, 华文瀚, 王娟娟, 孙登科. 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] 杜宇航, 刘向民, 王兴平, 蒋志浩. 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.
- [3] 高维东. 塔式太阳能电站定日镜场调度优化研究 [D]. 华北电力大学, 2021.
- [4] O. Farges , J.J. Bezian, M. El Hafi , Global optimization of solar power tower systems using a Monte Carlo algorithm: Application to a redesign of the PS10 solar thermal power plant [J], Renewable Energy, 2018, 119:345-353.

附录 A 附件清单

附件中的程序文件

1	DNIdata.m
2	problem01.m
3	problem02.m
4	calculate.m
5	problem03.m

附录 B 问题一源程序

Listing 1: DNIdata.m

```
1 GO = 1366;
2 latitude = 39.4;
3 longitude = 98.5;
4 altitude = 3000;
5 H = altitude / 1000;
6
7 % 定义一个包含月份的一维数组
8 months = {'一月', '二月', '三月', '四月', '五月', '六月', '七月', '八月', '九月', '十月',
    '十一月', '十二月'};
9
10 % 初始化结果数组
11 days_to_march = zeros(1, 12);
12
13 % 日期: 3月21号
14 march_21 = datenum('21-Mar', 'dd-mmm');
15
16 % 循环计算每个月21号距离3月21号的天数
17 for i = 1:12
18     % 构造日期: 当前月份的21号
19     date_str = sprintf('21-%s', months{i});
20
21     % 将日期字符串转换为日期数值
22     date_num = datenum(date_str, 'dd-mmm');
23
24     % 计算天数差
25     days_diff = date_num - march_21;
```

```

26
27 % 如果结果为负数，表示3月之前的月份，将其视为与12月连续
28 if days_diff < 0
29     days_diff = days_diff + 365; % 假设一年有365天
30 end
31
32 % 将结果保存在数组中
33 days_to_march(i) = days_diff;
34 end
35
36 % 显示结果
37 for i = 1:12
38     fprintf('%s距离3月21号的天数: %d\n', months{i}, days_to_march(i));
39 end
40
41
42 time_points = [9, 10.5, 12, 13.5, 15];
43
44 data = zeros(60, 6);
45
46 for month = 1:12
47     for i = 1:length(time_points)
48
49         D = days_to_march(month);
50
51         ST = time_points(i);
52         omega = pi / 12 * (ST - 12);
53
54         delta = asin(sin(2 * pi * D / 365) * sin(2 * pi / 365 * 23.45));
55
56         alpha_s = asin(cos(delta) * cosd(latitude) * cos(omega) + sind(delta) * sind(latitude));
57         gamma_s = acos((sin(delta) - sin(alpha_s) * sind(latitude)) / (cos(alpha_s) *
58                         cosd(latitude)));
59
60         a = 0.4237 - 0.00821 * (6 - H)^2;
61         b = 0.5055 + 0.00595 * (6.5 - H)^2;
62         c = 0.2711 + 0.01858 * (2.5 - H)^2;
63         DNI = G0 * (a + b * exp(-c / sin(alpha_s)));
64
65         row = (month - 1) * 5 + i;
66         data(row, 1) = month;
67         data(row, 2) = D;
68         data(row, 3) = ST;
69         data(row, 4) = alpha_s;
70         data(row, 5) = gamma_s;
71         data(row, 6) = DNI;

```

```

72     end
73 end
74
75 data_table = array2table(data, 'VariableNames', {'Month', 'Day', 'SolarTime', 'SolarAltitude',
76                           'SolarAzimuth', 'DNI'});
77 writetable(data_table, 'solar_data.xlsx');

```

Listing 2: problem01.m

```

1 data=xlsread('solar_data.xlsx');
2 location01= xlsread('data.xlsx');
3 h_tower=80;
4 h_collector=8;
5 d_collector=7;
6 z0=4;
7 data_size=60;
8 h0_target=h_tower-h_collector/2;
9 delta_h=h_collector/4;
10 h_mirro=6;
11 d_mirro=6;
12 ang_extend=4.65e-3;
13 eta_ref=0.92;
14
15 cudown01=zeros(length(location01(:,1)),4,60);
16 for i = 1:data_size
17     fprintf('range=%d\n',i);
18     for j=1:length(location01(:,1))
19         target=(-1)^j*delta_h+h0_target;
20         x=location01(j,1);
21         y=location01(j,2);
22         locate=[x y z0];
23         distance=(x^2+y^2+(target-z0)^2)^(1/2);
24         eta_at=0.99321-0.0001176*distance+1.97*10^(-8)*distance^2;
25         cudown01(j,1,i)=eta_at;
26
27         sun_altitude=data(i,4);
28         sun_azimuth=data(i,5);
29         DNI=data(i,6);
30         sun_vec=[sin(sun_azimuth)*cos(sun_altitude) cos(sun_azimuth)*cos(sun_altitude)
31                   sin(sun_azimuth)];
32         collect_vec=-[x y z0-target]/distance;
33         coll_azimuth=acos(dot([0 -1],(-[x y])/norm([x y])));
34         cos_2_beta=dot(sun_vec,collect_vec);
35         cos_beta=(1/2*(cos_2_beta+1))^(1/2);
36         eta_cos=cos_beta;
            cudown01(j,2,i)=eta_cos;

```

```

37
38 d_unsort=abs(location01-[x y]).^2,2).^(1/2);
39 ang_unsort=acos((location01-[x y])*[0;-1]./d_unsort)-sun_azimuth;
40 mat_unsort=[d_unsort ang_unsort location01];
41 mat_sorted=sortrows(mat_unsort,1,"ascend");
42 for k=2:length(mat_sorted(:,1))
43 if(mat_sorted(k,2)<pi/4)
44 cover01=[mat_sorted(k,3:4) z0];
45 break;
46 end
47 end
48 mirro0_vec=(sun_vec+collect_vec)/norm(sun_vec+collect_vec);
49 cov01_coll_vec=-cover01/norm(cover01);
50 mirro1_vec=(sun_vec+cov01_coll_vec)/norm(sun_vec+cov01_coll_vec);
51 T1=[sin(sun_azimuth) -cos(sun_azimuth) 0;cross(sun_vec,[sin(sun_azimuth)
52 -cos(sun_azimuth) 0]);sun_vec];
53 T1=T1';
54 mirro0_extend_x=cross([mirro0_vec(1:2) 0],[0 0 1])/norm([mirro0_vec(1:2) 0])*h_mirro/2;
55 mirro0_A1=locate+mirro0_extend_x+cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
56 mirro0_A2=locate+mirro0_extend_x-cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
57 mirro0_A3=locate-mirro0_extend_x-cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
58 mirro0_A4=locate-mirro0_extend_x+cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
59 mirro1_extend_x=cross([mirro1_vec(1:2) 0],[0 0 1])/norm([mirro1_vec(1:2) 0])*h_mirro/2;
60 mirro1_A1=cover01+mirro1_extend_x+cross(mirro1_extend_x,mirro1_vec)/h_mirro*d_mirro;
61 mirro1_A2=cover01+mirro1_extend_x-cross(mirro1_extend_x,mirro1_vec)/h_mirro*d_mirro;
62 mirro1_A3=cover01-mirro1_extend_x-cross(mirro1_extend_x,mirro1_vec)/h_mirro*d_mirro;
63 mirro1_A4=cover01-mirro1_extend_x+cross(mirro1_extend_x,mirro1_vec)/h_mirro*d_mirro;
64 locate_sun=locate*T1;
65 cover01_sun=cover01*T1;
66 mirro0_A1_sun=mirro0_A1*T1;
67 mirro0_A2_sun=mirro0_A2*T1;
68 mirro0_A3_sun=mirro0_A3*T1;
69 mirro0_A4_sun=mirro0_A4*T1;
70 mirro1_A1_sun=mirro1_A1*T1;
71 mirro1_A2_sun=mirro1_A2*T1;
72 mirro1_A3_sun=mirro1_A3*T1;
73 mirro1_A4_sun=mirro1_A4*T1;
74 mirro0_A1_sun=mirro0_A1_sun(1:2);
75 mirro0_A2_sun=mirro0_A2_sun(1:2);
76 mirro0_A3_sun=mirro0_A3_sun(1:2);
77 mirro0_A4_sun=mirro0_A4_sun(1:2);
78 mirro1_A1_sun=mirro1_A1_sun(1:2);
79 mirro1_A2_sun=mirro1_A2_sun(1:2);
80 mirro1_A3_sun=mirro1_A3_sun(1:2);
81 mirro1_A4_sun=mirro1_A4_sun(1:2);
82 poly0=polyshape([mirro0_A1_sun(1) mirro0_A2_sun(1) mirro0_A3_sun(1)
83 mirro0_A4_sun(1)], [mirro0_A1_sun(2) mirro0_A2_sun(2) mirro0_A3_sun(2)

```

```

    mirro0_A4_sun(2)]);
82 poly1=polyshape([mirro1_A1_sun(1) mirro1_A2_sun(1) mirro1_A3_sun(1)
83     mirro1_A4_sun(1)], [mirro1_A1_sun(2) mirro1_A2_sun(2) mirro1_A3_sun(2)
84     mirro1_A4_sun(2)]);
85 poly_out1=intersect(poly0,poly1);
86 if(poly_out1.area==0)
87     sb01=0;
88 else
89     sb01=poly_out1.area/poly0.area;
90 end
91
92 ang_unsort=abs(acos((location01-[x y])*[0;-1]./d_unsort)-coll_azimuth);
93 mat_unsort=[d_unsort ang_unsort location01];
94 mat_sorted=sortrows(mat_unsort,1,"ascend");
95 for k=2:length(mat_sorted(:,1))
96     if(mat_sorted(k,2)<pi/4)
97         cover01=[mat_sorted(k,3:4) z0];
98         break;
99     end
100 end
101 cov01_coll_vec=-cover01/norm(cover01);
102 mirro1_vec=(sun_vec+cov01_coll_vec)/norm(sun_vec+cov01_coll_vec);
103 T1=[sin(coll_azimuth) -cos(coll_azimuth) 0;cross(collect_vec,[sin(coll_azimuth)
104     -cos(coll_azimuth) 0]);collect_vec];
105 T1=T1';
106 mirro0_extend_x=cross([mirro0_vec(1:2) 0],[0 0 1])/norm([mirro0_vec(1:2) 0])*h_mirro/2;
107 mirro0_A1=locate+mirro0_extend_x+cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
108 mirro0_A2=locate+mirro0_extend_x-cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
109 mirro0_A3=locate-mirro0_extend_x-cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
110 mirro0_A4=locate-mirro0_extend_x+cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
111 mirro1_extend_x=cross([mirro1_vec(1:2) 0],[0 0 1])/norm([mirro1_vec(1:2) 0])*h_mirro/2;
112 mirro1_A1=cover01+mirro1_extend_x+cross(mirro1_extend_x,mirro1_vec)/h_mirro*d_mirro;
113 mirro1_A2=cover01+mirro1_extend_x-cross(mirro1_extend_x,mirro1_vec)/h_mirro*d_mirro;
114 mirro1_A3=cover01-mirro1_extend_x-cross(mirro1_extend_x,mirro1_vec)/h_mirro*d_mirro;
115 mirro1_A4=cover01-mirro1_extend_x+cross(mirro1_extend_x,mirro1_vec)/h_mirro*d_mirro;
116 locate_coll=locate*T1;
117 cover01_coll=cover01*T1;
118 mirro0_A1_coll=mirro0_A1*T1;
119 mirro0_A2_coll=mirro0_A2*T1;
120 mirro0_A3_coll=mirro0_A3*T1;
121 mirro0_A4_coll=mirro0_A4*T1;
122 mirro1_A1_coll=mirro1_A1*T1;
123 mirro1_A2_coll=mirro1_A2*T1;
124 mirro1_A3_coll=mirro1_A3*T1;
125 mirro1_A4_coll=mirro1_A4*T1;
126 mirro0_A1_coll=mirro0_A1_coll(1:2);
127 mirro0_A2_coll=mirro0_A2_coll(1:2);

```

```

125     mirro0_A3_coll=mirro0_A3_coll(1:2);
126     mirro0_A4_coll=mirro0_A4_coll(1:2);
127     mirro1_A1_coll=mirro1_A1_coll(1:2);
128     mirro1_A2_coll=mirro1_A2_coll(1:2);
129     mirro1_A3_coll=mirro1_A3_coll(1:2);
130     mirro1_A4_coll=mirro1_A4_coll(1:2);
131     poly0=polyshape([mirro0_A1_coll(1) mirro0_A2_coll(1) mirro0_A3_coll(1)
132                     mirro0_A4_coll(1)], [mirro0_A1_coll(2) mirro0_A2_coll(2) mirro0_A3_coll(2)
133                     mirro0_A4_coll(2)]);
134     poly1=polyshape([mirro1_A1_coll(1) mirro1_A2_coll(1) mirro1_A3_coll(1)
135                     mirro1_A4_coll(1)], [mirro1_A1_coll(2) mirro1_A2_coll(2) mirro1_A3_coll(2)
136                     mirro1_A4_coll(2)]);
137     poly_out2=intersect(poly0,poly1);
138     if(poly_out2.area==0)
139         sb02=0;
140     else
141         sb02=poly_out2.area/poly0.area;
142     end
143
144     poly0=polyshape([mirro0_A1(1) mirro0_A2(1) mirro0_A3(1) mirro0_A4(1)], [mirro0_A1(2)
145                     mirro0_A2(2) mirro0_A3(2) mirro0_A4(2)]);
146     sun_vec_down=[sun_vec(1:2) 0]/norm([sun_vec(1:2) 0]);
147     s1=cross(sun_vec_down,[0 0 1])*d_collector/2;
148     s2=-s1;
149     k=dot(sun_vec,sun_vec_down)/dot(sun_vec,[0 0 1]);
150     s3=s2-sun_vec_down*h_tower*k;
151     s4=-sun_vec_down*(d_collector/2+h_tower*k);
152     s5=s1-sun_vec_down*h_tower*k;
153     poly1=polyshape([s1(1) s2(1) s3(1) s4(1) s5(1)], [s1(2) s2(2) s3(2) s4(2) s5(2)]);
154     poly_out3=intersect(poly0,poly1);
155     if(poly_out3.area==0)
156         sb03=0;
157     else
158         sb03=poly_out3.area/poly0.area;
159     end
160
161     eta_sb=1-sb01-sb02-sb03;
162     if(eta_sb<0)
163         eta_sb=0;
164     end
165     cudown01(j,3,i)=eta_sb;
166
167     collect_vec_down=[collect_vec(1:2) 0]/norm([collect_vec(1:2) 0]);
168     mirro0_extend_x=cross([mirro0_vec(1:2) 0],[0 0 1])/norm([mirro0_vec(1:2)
169                     0])*(h_mirro/2+distance*tan(ang_extend));
170     mirro0_A1=locate+mirro0_extend_x+cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
171     mirro0_A2=locate+mirro0_extend_x-cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;

```

```

166     mirro0_A3=locate-mirro0_extend_x-cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
167     mirro0_A4=locate-mirro0_extend_x+cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
168     collect_vec_down_add=collect_vec_down*d_collector/2;
169     s1=cross(collect_vec_down_add,[0 0 1])+[0 0 h_tower-h_collector];
170     s2=-cross(collect_vec_down_add,[0 0 1])+[0 0 h_tower-h_collector];
171     s3=-cross(collect_vec_down_add,[0 0 1])+[0 0 h_tower];
172     s4=cross(collect_vec_down_add,[0 0 1])+[0 0 h_tower];
173     T2=[0 0 1;cross(collect_vec_down,[0 0 1]);collect_vec_down];
174     T2=T2';
175     mirro0_A1_coll=mirro0_A1*T2;
176     mirro0_A2_coll=mirro0_A2*T2;
177     mirro0_A3_coll=mirro0_A3*T2;
178     mirro0_A4_coll=mirro0_A4*T2;
179     s1_coll=s1*T2;
180     s2_coll=s2*T2;
181     s3_coll=s3*T2;
182     s4_coll=s4*T2;
183     poly0=polyshape([mirro0_A1_coll(1)+target mirro0_A2_coll(1)+target
184                     mirro0_A3_coll(1)+target mirro0_A4_coll(1)+target],[mirro0_A1_coll(2)
185                     mirro0_A2_coll(2) mirro0_A3_coll(2) mirro0_A4_coll(2)]);
186     poly1=polyshape([s1_coll(1) s2_coll(1) s3_coll(1) s4_coll(1)],[s1_coll(2) s2_coll(2)
187                     s3_coll(2) s4_coll(2)]);
188     poly_out4=intersect(poly0,poly1);
189     if(eta_sb==0)
190         eta_trunc=0;
191     else
192         eta_truc=poly_out4.area/(poly0.area*eta_sb);
193     end
194     if(eta_truc>1)
195         eta_trunc=1;
196     end
197     cudown01(j,4,i)=eta_truc;
198
199     end
200
201     result_temp=zeros(60,4);
202     for i=1:data_size
203         result_temp(i,:)=sum(cudown01(:,:,i),1)/length(location01(:,1));
204     end
205
206     result01=zeros(12,5);
207     for i=0:11
208         eta_temp=sum(result_temp(5*i+1:5*i+5,:),1)/5;
209         DNI_temp=sum(data(5*i+1:5*i+5,6))/5;
210         result01(i+1,:)=[eta_ref*prod(eta_temp) eta_temp(2) eta_temp(3) eta_temp(4)
211                         eta_ref*prod(eta_temp)*DNI_temp];

```

```

209 end
210
211 result02_temp=sum(result01,1)/12;
212 result02=[result02_temp(1:4) result02_temp(5)*h_mirro*d_mirro*length(location01(:,1))
213 result02_temp(5)];
214
215 result01(:,5)=result01(:,5)/1e+3;
216 result02(6)=result02(6)/1e+3;
217 result02(5)=result02(5)/1e+6;
218 writematrix(result01, 'problem01_table01_.xlsx');
219 writematrix(result02, 'problem01_table02_.xlsx');

```

附录 C 问题二源程序

Listing 3: problem02.m

```

1 data=xlsread('solar_data.xlsx');
2 z0=6;
3 data_size=60;
4 test_size=20;
5 h_mirro0=8;
6 d_mirro0=4;
7 eta_ref=0.92;
8 d_pass=5;
9 total=0;
10 line0=10000;
11 DNI_aver=sum(data(:,6),1)/data_size;
12 engage=1e+5;
13 set=200;
14 base=700;
15
16
17 location01=[0 101 z0 0];
18 %add function
19 eta=[0.9784 0.6618 1.0000 0.9557];
20 location01(1,4)=DNI_aver*eta_ref*prod(eta)*1e-6*h_mirro0*d_mirro0;
21 total=total+location01(1,4);
22
23 while(total<=60)
24     fprintf('total=%d\n',total);
25     test_d0=(total/60)^engage*base+set;
26     i=1;
27     best=zeros(test_size,4);
28     x0=location01(length(location01(:,1)),1);

```

```

29 y0=location01(length(location01(:,1)),2);
30 x=x0;
31 y=y0;
32 test_set=zeros(test_size,2);
33 while(i<=test_size)
34     x=x0;
35     y=y0;
36     test_d=rand(1,1)*test_d0;
37     ang=rand(1,1)*2*pi;
38     while(test_d<d_mirro0+d_pass)
39         test_d=rand(1,1)*test_d0;
40     end
41     lines=line0;
42     if(length(location01(:,1))<lines)
43         lines=length(location01(:,1));
44     end
45     while sum((sum(abs([x
y] .*ones(size(location01(:,1:2))-location01(:,1:2)).^2,2).^(1/2)<=(d_mirro0+d_pass)*
...
ones(size(location01(:,1))))||(norm([x y])<100)||norm([x y])>base+set)
%||(ang<=pi/6)|| (ang>=5*pi/6)
r=test_d+d_pass;
ang=rand(1,1)*2*pi;
x=x0+r*cos(ang);
y=y0+r*sin(ang);
end
test_set(i,:)=[x y];
i=i+1;
end
for i=1:test_size
    %add function
    eta=[0 0 0 0];
    eta=calculate(eta,x,y,h_mirro0,d_mirro0,z0,data);
    best(i,1:2)=test_set(i,:);
    best(i,3)=z0;
    best(i,4)=DNI_aver*eta_ref*prod(eta)*1e-6*h_mirro0*d_mirro0;
end
best=sortrows(best,4,'descend');
location01=[location01;best(1,:)];
total=total+best(1,4);

end
aver=total/length(location_t(:,1))/d_mirro0*h_mirro0*1000;
scatter(location01(:,1),location01(:,2));

location_t=location01(:,1:2);
total=0;

```

```

74 z_result=0;
75 for z=z0:8
76     total_temp=0;
77     for i=1:length(location_t(:,1))
78         %add function
79         eta=[0 0 0 0];
80         eta=calculate(eta,x ,y,h_mirro0,d_mirro0,z0,data);
81         total_temp=total_temp+DNI_aver*eta_ref*prod(eta)*1e-6*h_mirro0*d_mirro0;
82     end
83     if(total_temp>total)
84         total=total_temp;
85         z_result=z;
86     end
87 end
88
89 total=0;
90 h_mirro=h_mirro0;
91 d_mirro=d_mirro0;
92 for h=h_mirro0:8
93     for d=d_mirro0+1:8
94         total_temp=0;
95         for i=1:length(location_t(:,1))
96             %add function
97             eta=[0 0 0 0];
98             eta=calculate(eta,x,y,h_mirro0,d_mirro0,z0,data);
99             total_temp=total_temp+DNI_aver*eta_ref*prod(eta)*1e-6*h_mirro0*d_mirro0;
100        end
101        if(total_temp>total)
102            total=total_temp;
103            h_mirro=h;
104            d_mirro=d;
105        end
106    end
107 end
108
109 fprintf('average=%d\n',aver);
110
111 %add function:final arguments

```

Listing 4: calculate.m

```

1 function eta=calculate(eta,x,y,h_mirro,d_mirro,z0,data)
2 h_tower=80;
3 h_collector=8;
4 d_collector=7;
5 data_size=60;
6 h0_target=h_tower-h_collector/2;

```

```

7 delta_h=h_collector/4;
8 ang_extend=4.65e-3;
9 eta_ref=0.92;
10
11 cudown01=zeros(60,4);
12 for i = 1:data_size
13
14     target=(-1)^round(rand(1,1)*10) *delta_h+h0_target;
15     distance=(x^2+y^2+(target-z0)^2)^(1/2);
16     eta_at=0.99321-0.0001176*distance+1.97*10^(-8)*distance^2;
17     cudown01(i,1)=eta_at;
18     sun_altitude=data(i,4);
19     sun_azimuth=data(i,5);
20     DNI=data(i,6);
21     sun_vec=[sin(sun_azimuth)*cos(sun_altitude) cos(sun_azimuth)*cos(sun_altitude)
22             sin(sun_azimuth)];
23     collect_vec=-[x y z0-target]/distance;
24     coll_azimuth=acos(dot([0 -1],[-x y])/norm([x y]));
25     cos_2_beta=dot(sun_vec,collect_vec);
26     cos_beta=(1/2*(cos_2_beta+1))^(1/2);
27     eta_cos=cos_beta;
28     cudown01(i,2)=eta_cos;
29
30     eta_sb=0.993753037;
31     cudown01(i,3)=eta_sb;
32
33     locate=[x y z0];
34     mirro0_vec=(sun_vec+collect_vec)/norm(sun_vec+collect_vec);
35     collect_vec_down=[collect_vec(1:2) 0]/norm([collect_vec(1:2) 0]);
36     mirro0_extend_x=cross([mirro0_vec(1:2) 0],[0 0 1])/norm([mirro0_vec(1:2)
37                 0])*(h_mirro/2+distance*tan(ang_extend));
38     mirro0_A1=locate+mirro0_extend_x+cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
39     mirro0_A2=locate+mirro0_extend_x-cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
40     mirro0_A3=locate-mirro0_extend_x-cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
41     mirro0_A4=locate-mirro0_extend_x+cross(mirro0_extend_x,mirro0_vec)/h_mirro*d_mirro;
42     collect_vec_down_add=collect_vec_down*d_collector/2;
43     s1=cross(collect_vec_down_add,[0 0 1])+[0 0 h_tower-h_collector];
44     s2=-cross(collect_vec_down_add,[0 0 1])+[0 0 h_tower-h_collector];
45     s3=-cross(collect_vec_down_add,[0 0 1])+[0 0 h_tower];
46     s4=cross(collect_vec_down_add,[0 0 1])+[0 0 h_tower];
47     T2=[0 0 1;cross(collect_vec_down,[0 0 1]);collect_vec_down];
48     T2=T2';
49     mirro0_A1_coll=mirro0_A1*T2;
50     mirro0_A2_coll=mirro0_A2*T2;
51     mirro0_A3_coll=mirro0_A3*T2;
52     mirro0_A4_coll=mirro0_A4*T2;
53     s1_coll=s1*T2;

```

```

52 s2_coll=s2*T2;
53 s3_coll=s3*T2;
54 s4_coll=s4*T2;
55 poly0=polyshape([mirro0_A1_coll(1)+target mirro0_A2_coll(1)+target
56                 mirro0_A3_coll(1)+target mirro0_A4_coll(1)+target],[mirro0_A1_coll(2)
57                 mirro0_A2_coll(2) mirro0_A3_coll(2) mirro0_A4_coll(2)]);
58 poly1=polyshape([s1_coll(1) s2_coll(1) s3_coll(1) s4_coll(1)],[s1_coll(2) s2_coll(2)
59                 s3_coll(2) s4_coll(2)]);
60 poly_out4=intersect(poly0,poly1);
61 if(eta_sb==0)
62     eta_trunc=0;
63 else
64     eta_truc=poly_out4.area/(poly0.area*eta_sb);
65 end
66 if(eta_truc>1)
67     eta_trunc=1;
68 end
69 cudown01(i,4)=eta_truc;
70
71
72
73 eta=sum(result_temp,1)/length(result_temp(:,1));

```

附录 D 问题三源程序

Listing 5: problem03.m

```

1 data=xlsread('solar_data.xlsx');
2 z0=4.2;
3 z_end=6;
4 data_size=60;
5 test_size=20;
6 h_mirro0=8;
7 d_mirro0=8;
8 h_mirro_start=2;
9 d_mirro_start=2;
10 eta_ref=0.92;
11 d_pass=5;
12 total=0;
13 line0=10000;
14 DNI_aver=sum(data(:,6),1)/data_size;
15 engage=1e+1;

```

```

16 set=300;
17 base=700;
18
19
20 location01=[0 101 z0 0 h_mirro0 d_mirro0];
21 %add function
22 eta=[0.9784 0.6618 1.0000 0.9557];
23 location01(1,4)=DNI_aver*eta_ref*prod(eta)*1e-6*h_mirro0*d_mirro0;
24 total=total+location01(1,4);
25
26 while(total<=62)
27     fprintf('total=%d\n',total);
28     test_d0=(total/62)^engage*base+set;
29     i=1;
30     best=ones(test_size,6);
31     best(:,5)=best(:,5)*h_mirro0;
32     best(:,6)=best(:,6)*d_mirro0;
33     x0=location01(length(location01(:,1)),1);
34     y0=location01(length(location01(:,1)),2);
35     x=x0;
36     y=y0;
37     test_set=zeros(test_size,2);
38     while(i<=test_size)
39         x=x0;
40         y=y0;
41         test_d=rand(1,1)*test_d0;
42         ang=rand(1,1)*2*pi;
43         while(test_d<d_mirro0+d_pass)
44             test_d=rand(1,1)*test_d0;
45         end
46         lines=line0;
47         if(length(location01(:,1))<lines)
48             lines=length(location01(:,1));
49         end
50         while sum((sum(abs([x
y].*ones(size(location01(:,1:2)))-location01(:,1:2)).^2,2).^(1/2)<=(d_mirro0+d_pass)*
...
51             ones(size(location01(:,1))))|| (norm([x y])<100)|| (norm([x y])>base+set)
52             ||(ang<=pi/6)|| (ang>=5*pi/6)
53             r=test_d+d_pass;
54             ang=rand(1,1)*2*pi;
55             x=x0+r*cos(ang);
56             y=y0+r*sin(ang);
57         end
58         test_set(i,:)=[x y];
59         i=i+1;
60     end

```

```

61   for i=1:test_size
62     %add function
63     eta=[0 0 0 0];
64     eta=calculate(eta,x,y,h_mirro0,d_mirro0,z0,data);
65     best(i,1:2)=test_set(i,:);
66     best(i,3)=z0;
67     best(i,4)=DNI_aver*eta_ref*prod(eta)*1e-6*h_mirro0*d_mirro0;
68   end
69   best=sortrows(best,4,'descend');
70   temp_result=best(1,4);
71   temp=0;
72   for z=z0:0.1:z_end
73     for h=h_mirro_start:0.1:h_mirro0
74       for d=d_mirro_start:0.5:d_mirro0
75         eta=calculate(eta,best(1,1),best(1,2),h,d,z,data);
76         temp=DNI_aver*eta_ref*prod(eta)*h*1e-6*d;
77         if(temp/(h*d)>temp_result/(best(1,5)*best(1,6)))
78           tmepr_result=temp;
79           best(1,3)=z;
80           best(1,5)=h;
81           best(1,6)=d;
82         end
83       end
84     end
85   end
86   best(1,4)=temp_result;
87   location01=[location01;best(1,:)];
88   total=total+best(1,4);
89
90 end
91 scatter(location01(:,1),location01(:,2));

```