混合可再生能源发电系统仿真与优化

3190103700 孙懿萱

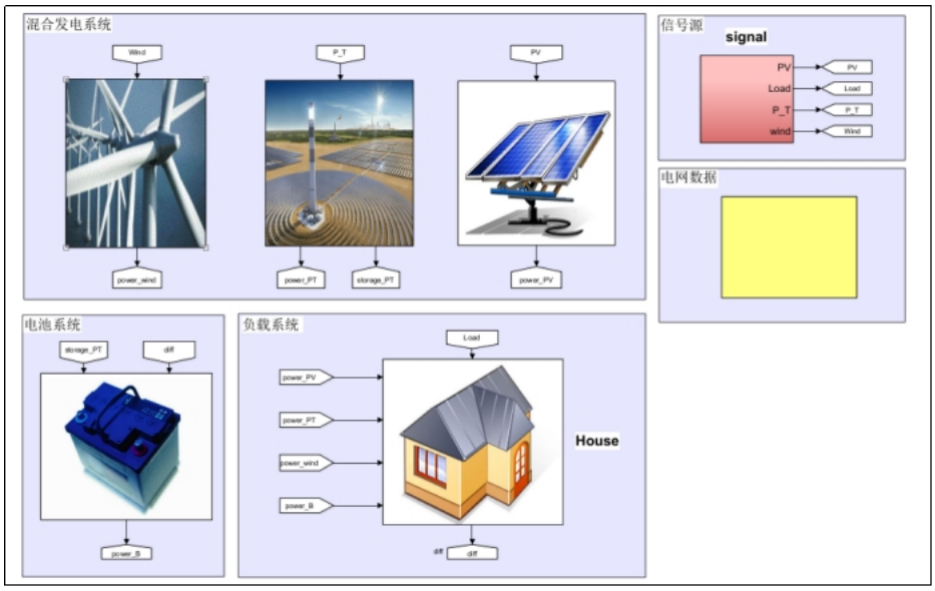
1摘要

以课程中搭建的光伏发电、光热发电和风力发电为基础，外加蓄电池和熔盐电池储能结构，搭建了混合可再生能源发电系统，并以已给的风力、光照条件和需求为输入数据，用遗传算法的方式找出基于最优成本的资源配置方案。

2模型架构

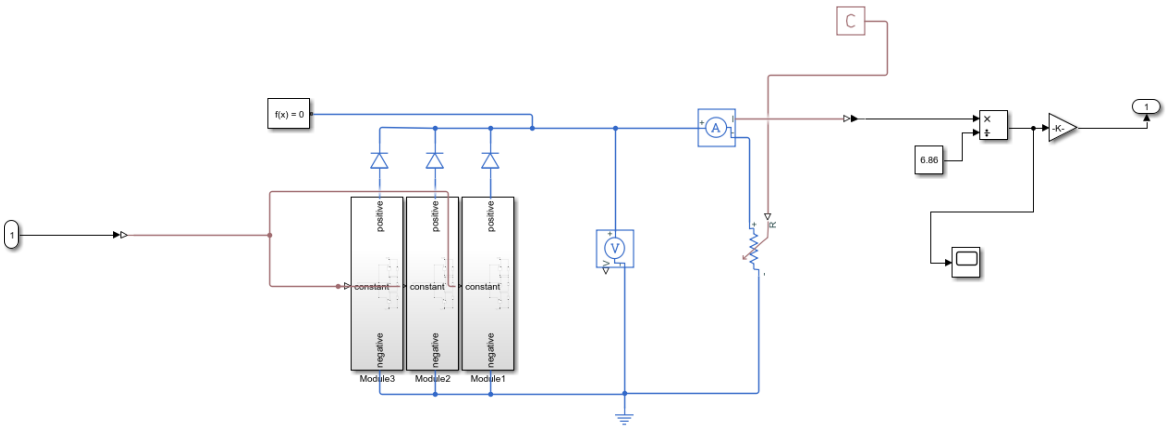
2-1总体架构

模型主要可以分为以下几部分：混合发电系统（包括风力、光热和光伏发电系统）、电池系统（包含蓄电池和熔盐电池）、负载系统、信号源和电网数据。

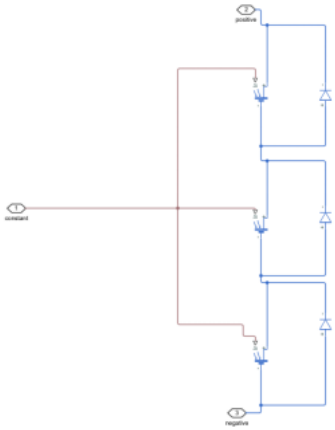


2-2混合发电系统

2-2-1光伏发电系统

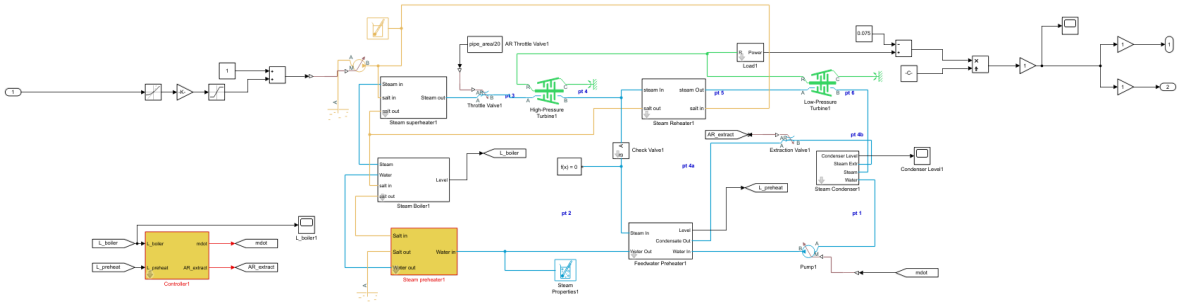


光伏发电部分包括含隔离二极管的三个并联光伏模组，每个光伏模组包含三个串联的光伏电池，光伏电池利用solar cell元件，将光照信息转化为电流信息输出，电池内部结构如下图：



经过优化算法（后续提及）设计，将输出的电流设计一定增益作为混合能源系统的功率设置。

2-2-2光热发电系统

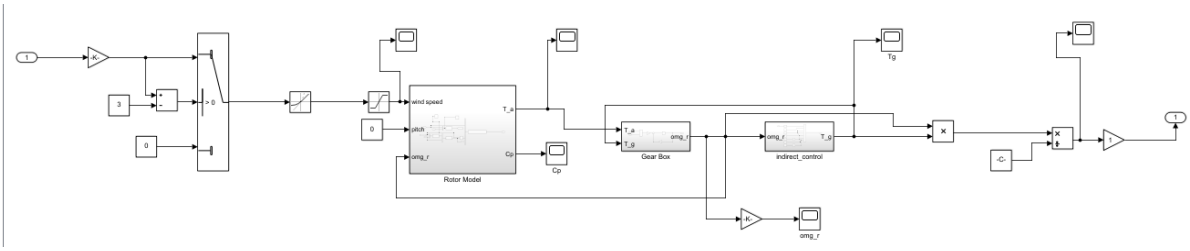


光热发电部分的搭建基于matlab已有的Rankine循环结构，包含吸热器、蒸汽发生器、预热器、过热器、汽轮机、冷却器。光热部分的输出分为两个部分，一部分用于白天和风电、光伏一起满足供电需求，若有多余则存入蓄电池；另一部分用于储能，存储于熔盐电池中。考虑实际情况，熔盐储热时通过发电机进行热电转换，因此在设计该系统时，直接将熔盐储热系统看作为电池，输出也以电能的形式；两部分的比例情况由优化算法得出。

2-2-3风力发电系统

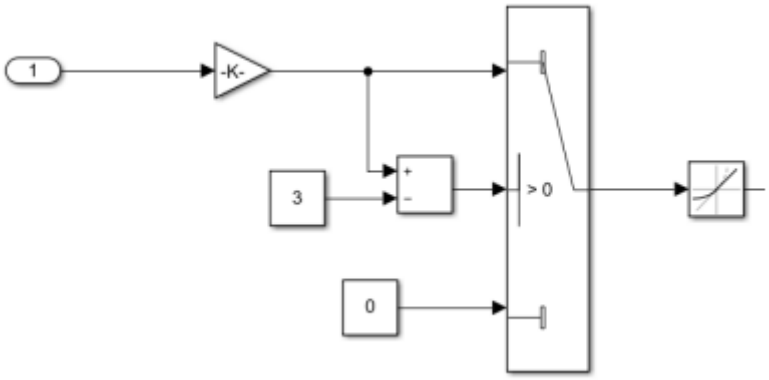
搭建思路基于下图，考虑风轮空气动力装置和传动装置。

风力总体架构如下所示：

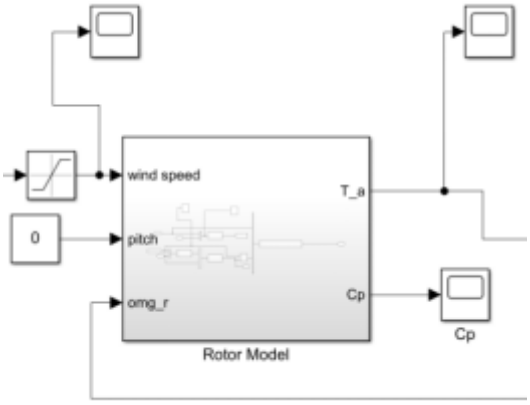


从左至右结构作用分别为：

1、筛选风速值，大于3可视为可利用风力资源，对经筛选后的风速进行平滑处理



2、风力机气动模型



该部分负责风能向机械能转化，根据空气动力学分析可得，风轮从风中获得的机械功率为：

$$P_a = \frac{1}{2}\rho\pi R^2 C_p(\lambda,\beta)v^3$$

$$\lambda = \frac{R\omega_r}{v}$$

具有如下关系式。

$$C_p(\lambda,\beta) = 0.22(\frac{116}{m} - 0.4\beta - 5)\exp\left(-\frac{12.5}{m}\right)$$

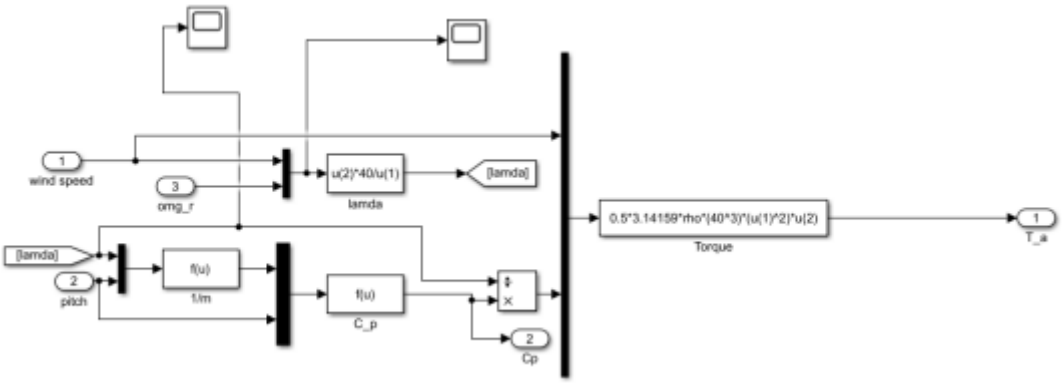
$$\frac{1}{m} = \frac{1}{\lambda + 0.08\beta} - \frac{0.035}{\beta^3 + 1}, 其中\beta = 0$$

进一步可获得空气动力学力矩

$$T_a = \frac{1}{2}\rho\pi R^3 C_q(\lambda)v^2$$

其中定义力矩系数$C_q(\lambda,\beta) = \frac{C_p(\lambda,\beta)}{\lambda}$

由上述数值关系式可建立如下等效数值模型，得到气动转矩



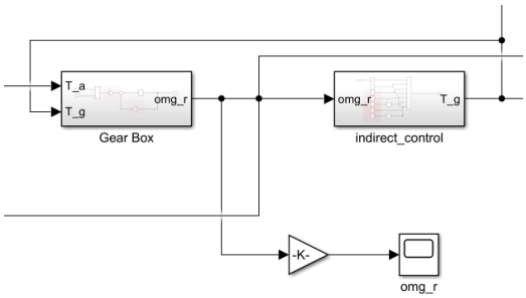
3、求解ω_r和Tg

根据机械传动轴系的微分方程*

*$J_t\dot{\omega}_r = T_a - K_t\omega_r - T_g$

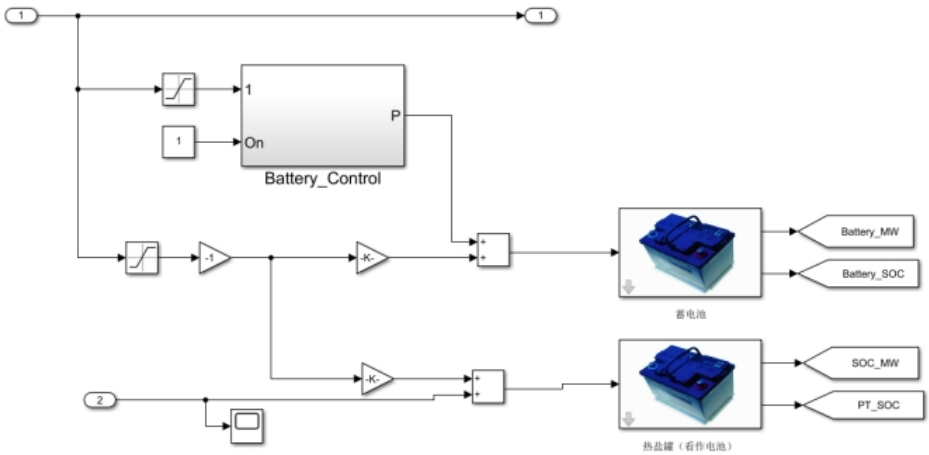
，求解ω_r，同时将ω_r返回至Rotor Model结构。

转矩控制方式采用间接速度控制，将Tg返回至Gear box中$T_g = \frac{\frac{1}{2}\pi\rho R^5 C_p}{\lambda^3}$



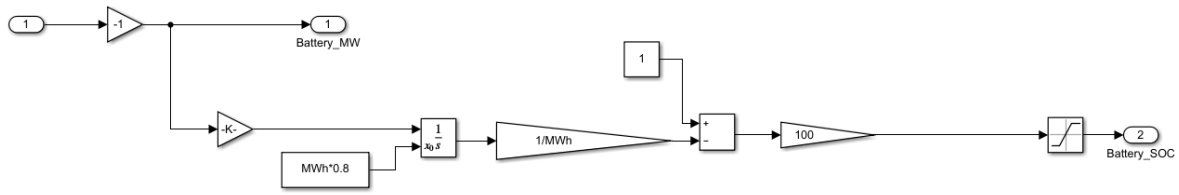
最终将机械能转化为电能

2-3电池系统

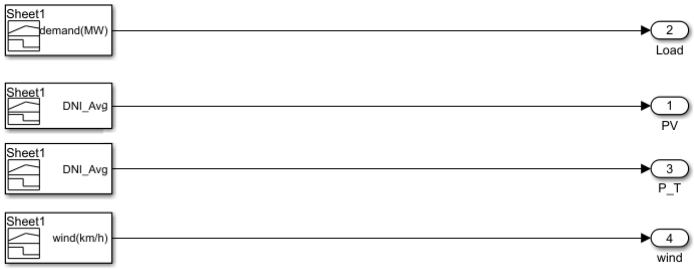


输入信号由风光和光热发电功率减去demand得到，若该值小于0，则代表电池处于放电状态；若大于0，则处于充电状态。利用saturaion判断正负值，若正，则向蓄电池内充电，若为负，则由蓄电池和熔盐电池按一定比例放电，比例由优化算法得到。同时，光热产生的总功率除去供电的部分，其余存入熔盐电池中。

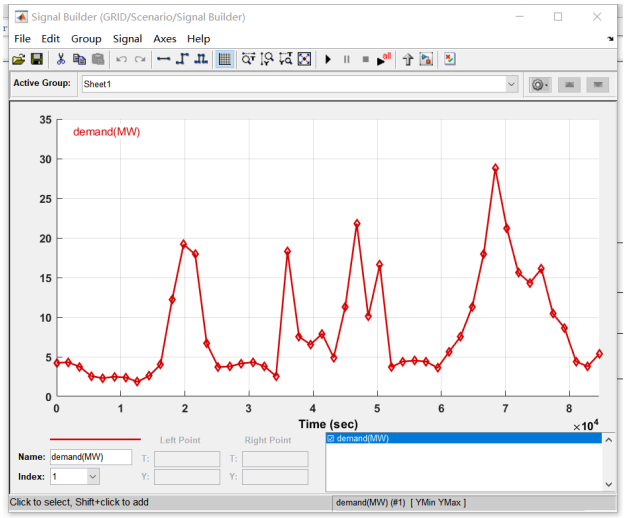
电池内部结构如下，参考了matlab的Simplified Model of a Small Scale Micro-Grid的蓄电池结构。



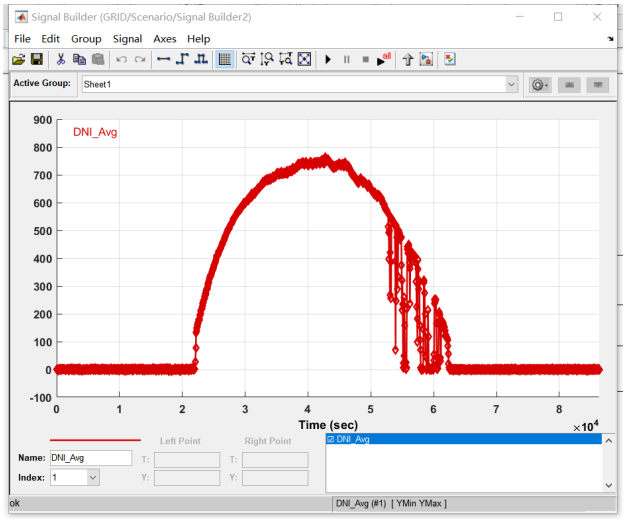
2-4 信号源



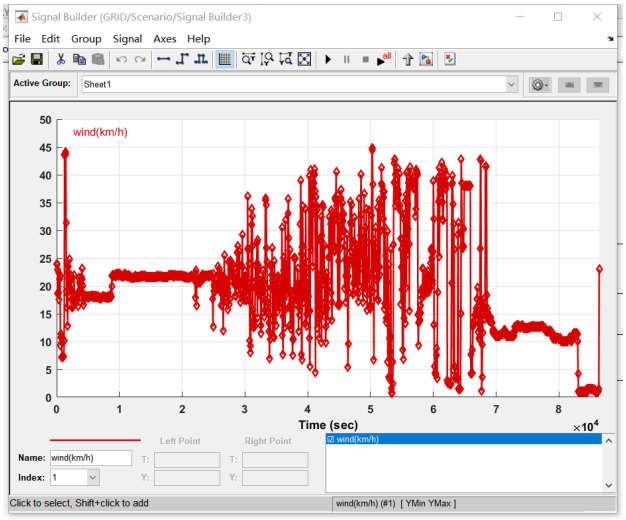
demand



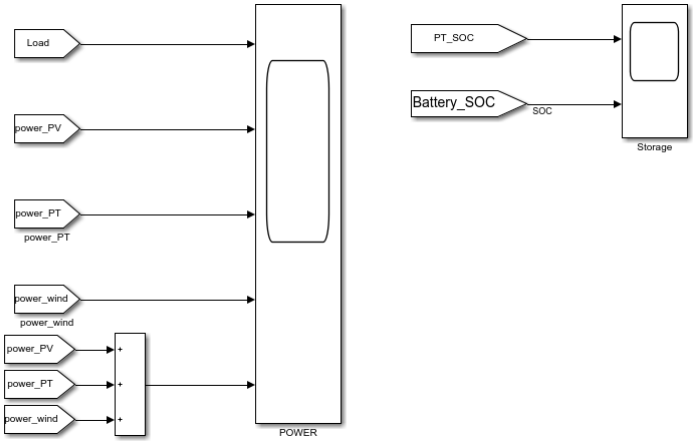
光照



风力



2-5 电网数据采集



3 优化方案设计

3-1 总体思路

1、根据给出的光照条件和风力数据可以分别求出光伏、光热和风力三种能源的各时刻发电量。由于三种能源的发电和储能成本不同，为了成本最优化，设计不同比例因子计算三种能量的配比。遍历一天数据，得出三种能量最大的发电率，以此算出装机容量和装机成本

2、 当配比后的发电量满足demand时，将多余发电量存入蓄电池中；若配比后的发电量不满足demand时，需要电池发电，考虑蓄电池和熔盐电池发电的不同成本，设定不同比例因子计算两个电池发电量的配比。遍历一天数据可得出每时刻的充放电量和总充放电量，可算出两个电池建设成本。

3、总成本即为光伏光热风力的装机成本和蓄电池、熔盐电池的建设成本

4、 由遗传算法多次寻优得出最有成本。其中考虑三个违背事实的情况：蓄电池和熔盐电池充电量小于放电量、总供电量供不应求，针对以上情况在进行遗传算法求解时，加入罚函数限制。

note：虽然系统成本计算案例按50MW设计，但是根据最终求解结果可得30MW左右已经可以满足本题所给demand，因此考虑系统设计的经济性，未采取50MW的设计方案。

3-2 具体算法

参数初始化

```
import numpy as np
from sko.GA import GA
import matplotlib.pyplot as plt

K = np.zeros(7)  #光伏功率、光热功率、风力功率、光热储能比、蓄电池用电比、蓄电池容量、热盐容量
P_solar = np.zeros(48) # 光伏发电功率
P_CPS = np.zeros(48)   # 光热总功率
P_T = np.zeros(48)     # 光热发电功率
P_CPSbattery = np.zeros(48)  # 熔盐电池充电功率
P_wind = np.zeros(48)   # 风力发电功率
P_battery = np.zeros(48)  # 蓄电池和光热电池总体状态
P_demand = np.zeros(48)   #需求量
P_solar_sum = 0 #白天光伏发电
P_CPS_sum = 0  #白天光热发电
P_wind_sum = 0 # 总风力发电量
P_battery_in = 0   # 双电池总充电量
P_CPSbattery_in=0  # 熔盐电池总充电量
P_battery_out = 0  # 双电池总放电量
P_CPSbattery_out=0 # 熔盐电池总放电量

P_demand = [4.235, 4.305, 3.745, 2.555, 2.31, 2.485, 2.38, 1.855, 2.59, 4.025,
12.18, 19.18, 17.92, 6.685, 3.71, 3.78, 4.165, 4.305, 3.815, 2.555, 18.305,
7.525, 6.545, 7.91, 4.9, 11.27, 21.805, 10.115, 16.59, 3.71, 4.41, 4.515, 4.41,
3.605, 5.635, 7.56, 11.305, 17.99, 28.77, 21.245, 15.61, 14.315, 16.1, 10.43,
8.645, 4.41, 3.78, 5.355]
wind = [1.517, 0.983, 0.692, 0.68, 0.646, 0.804, 1.224, 1.164, 1.161, 1.167,
1.210, 1.198, 1.161, 1.151, 0.98, 0.883, 1.087, 1.035, 0.957, 0.824, 0.74, 0.64,
0.852, 1.208, 0.977, 1.740, 1.747, 1.825, 1.639, 2.038, 1.473, 1.8978, 3.352,
1.631, 1.624, 1.915, 1.611, 1.922, 1.15, 0.449, 0.134, 0.208, 0.215, 0.223,
0.123, 0, 0, 0, 0]
PV = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1.236, 2.865, 4.404, 5.088, 5.563,
5.949, 6.201, 6.354, 6.504, 6.615, 6.726, 6.821, 6.684, 6.441, 6.132, 5.743,
5.288, 4.644, 3.437, 3.31, 1.815, 1.332, 0.228, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
0, 0]
DSP = [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5,
0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0,
0, 0, 0,0, 0, 0, 0, 0, 0, 0, 0]
```

定义函数计算各时刻不同能源的发电功率和电池的状态

```
def cal_func(K):

    for i in range(48):
        P_solar[i] = K[0]*PV[i]  # 光伏发电功率
        P_CPS[i] = K[1]*PV[i] # 光热总功率
        P_T[i] = K[1]*(1-K[3])*PV[i] # 光热发电功率
        P_wind[i] = K[2]*wind[i] # 风力发电功率
        # 判断当前时刻总发电功率是否满足demand，满足处于充电状态
        P_battery[i] =  P_solar[i] + P_wind[i] + P_T[i] - P_demand[i]
        P_CPSbattery[i] =  K[1]*K[3]*PV[i] # 熔盐电池充电功率
```

定义目标函数

该函数主要目的是首先判断风光热发电能否满足demand，若满足，则向蓄电池中充电；若不满足，则根据需求差值按一定比例进行蓄电池和熔盐电池放电；记录不同时刻蓄电池和熔盐电池分别的累计充放电量，以此可算出蓄电池和熔盐电池的容量。

```
def real_demo_func(K):
    P_battery_in = 0
    P_CPSbattery_in=0
    P_battery_out = 0
    P_CPSbattery_out=0
    max_battery=0
    min_battery=0
    max_CSP=0
    min_CSP=0

    MIN = 0

    for i in range(48):
        if P_battery[i] < 0:
        # 表示光伏光热风力发电无法满足demand需求，需要电池放电以满足需求
            # 蓄电池的放电量
            P_battery_out += K[4]*abs(P_battery[i])
            # 熔盐电池的放电量
            P_CPSbattery_out += (1-K[4])*abs(P_battery[i])
        else:
        # 此时三种混合能源发电满足需求，可考虑向双电池中充电
            # 记录蓄电池充电量
            P_battery_in += P_battery[i]
        # 记录熔盐电池充电量
        P_CPSbattery_in += P_CPSbattery[i]

        max_battery=max(max_battery,(P_battery_in-P_battery_out))#记录最高电池状态
        min_battery=min(min_battery,(P_battery_in-P_battery_out))#记录最低电池状态
        max_CSP=max(max_CSP,(P_CPSbattery_in-P_CPSbattery_out))#记录最高电池状态
        min_CSP=min(min_CSP,(P_CPSbattery_in-P_CPSbattery_out))#记录最低电池状态

    P_solar_max = max(P_solar)   #总光伏发电
    P_CPS_max = max(P_CPS)       #总光热
    P_wind_max = max(P_wind)     # 总风力


    MIN = 800*P_solar_max + 920*P_wind_max + 2000*P_CPS_max +45*(max_CSP-
min_CSP) + 250* (max_battery-min_battery)  #成本
    K[5]=max_battery-min_battery
    K[6]=max_CSP-min_CSP
    return MIN
```

定义寻优函数

该函数在目标函数的基础上考虑蓄电池、熔盐电池和总发电量供不应求的的情况，加入罚函数限制，总约束即为成本和罚函数之和

```
def demo_func(K):
    P_battery_in = 0
    P_CPSbattery_in=0
    P_battery_out = 0
```

```
    P_CPSbattery_out=0
    max_battery=0
    min_battery=0
    max_CSP=0
    min_CSP=0
    MIN = 0
    con1 = 0
    con2 = 0
    con3 = 0
    cal_func(K)

    for i in range(48):
        if P_battery[i] < 0:
            P_battery_out += K[4]*abs(P_battery[i])
            P_CPSbattery_out += (1-K[4])*abs(P_battery[i])
        else:
            P_battery_in += P_battery[i]

        P_CPSbattery_in += P_CPSbattery[i]

        max_battery=max(max_battery,(P_battery_in-P_battery_out))#记录最高电池状态
        min_battery=min(min_battery,(P_battery_in-P_battery_out))#记录最低电池状态
        max_CSP=max(max_CSP,(P_CPSbattery_in-P_CPSbattery_out))#记录最高电池状态
        min_CSP=min(min_CSP,(P_CPSbattery_in-P_CPSbattery_out))#记录最低电池状态
    # 以上部分原理同目标函数

    P_solar_sum = sum(P_solar)   #总光伏发电
    P_CPS_sum = sum(P_CPS)       #总光热发电
    P_wind_sum = sum(P_wind)     #总风力发电
    P_demand_sum = sum(P_demand) #总需求

    P_solar_max = max(P_solar)   #总光伏发电
    P_CPS_max = max(P_CPS)       #总光热发电
    P_wind_max = max(P_wind)
    P_demand_sum = sum(P_demand)
    MIN = 800*P_solar_max + 920*P_wind_max + 2000*P_CPS_max +45*(max_CSP-
min_CSP) + 250* (max_battery-min_battery)  #成本

    # 罚函数设计
    if (P_battery_in-P_battery_out)<0:
        # 蓄电池充电量小于放电量，不符合事实，加入罚函数约束
        con1 += 10000*abs(P_battery_in-P_battery_out)
    if (P_CPSbattery_in-P_CPSbattery_out)<0:
        # 容颜电池充电量小于放电量，不符合事实，加入罚函数约束
        con2 += 10000*abs(P_CPSbattery_in-P_CPSbattery_out)
    for i in range(48):
        # 供不应求，加入罚函数限制
        a = P_T[i] + P_solar[i] + P_wind[i] - P_battery[i]
        if a < P_demand[i]:
            con3 += 100000*abs(a-P_demand[i])

    cons = MIN + con1 + con2 + con3 # 总约束即为成本+罚函数
    return cons
```

遗传算法求解最优情况

```
###设置初始值
best_cons = np.inf
K = [2, 0.5, 3, 0.2, 0.5, 5, 10]
best_K = np.zeros(4)

###多次寻优求最小值
for i in range(20):
    ga = GA(func=demo_func, n_dim = 7,size_pop = 200, max_iter = 200,prob_mut =
0.001, lb = [0]*7, ub = [10, 10, 10, 1, 1, 1, 1])
    K, cons = ga.run()
    print(cons)
    real_y = real_demo_func(K)
    print(K,real_y)
    if cons < best_cons:
        best_K = K
        y = real_y
        best_cons = cons
print('Final parameters are :',best_K)
print('Final result is :',y)

cal_func(best_K)
P_real = np.zeros(48)
for i in range(48):
    P_real[i] = P_CPS[i] + P_solar[i] + P_wind[i] - P_battery[i]
```

4 优化方案结果

经过多次寻优，得到以下几种基于成本优化的结果：

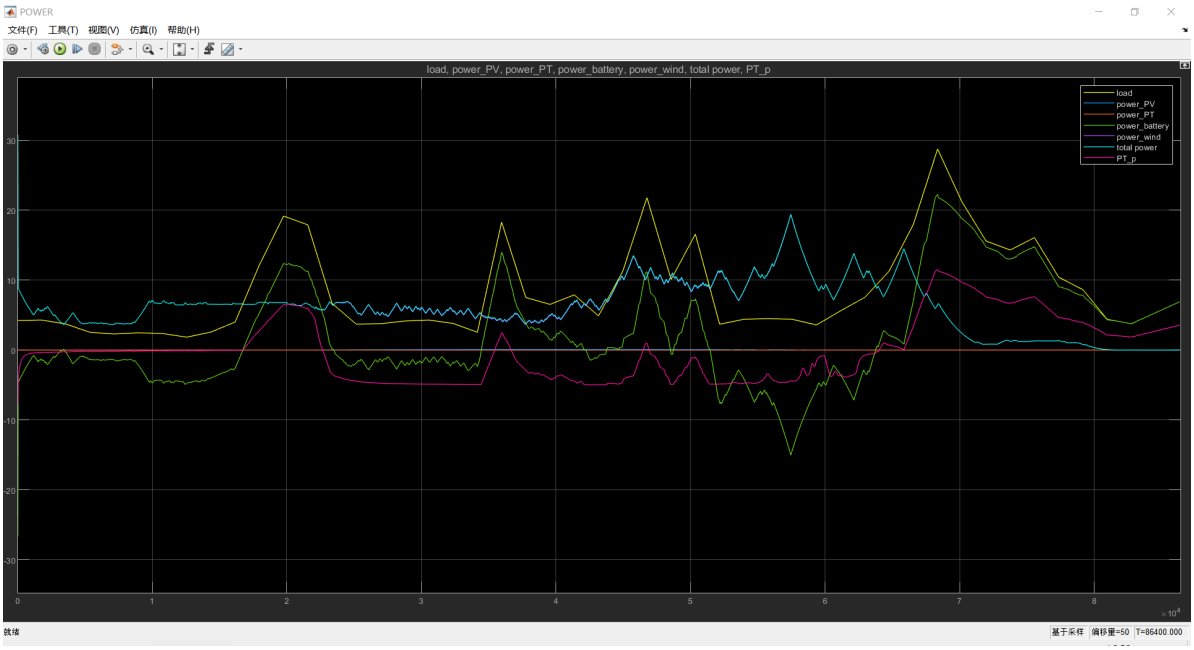
4-1 风力供电为主，光热以储热为主

*根据占比，乘以一天中风力/光照最大值，即为装机容量						
光伏占比	光热总产能比	风力占比	光热储热比例	蓄电池放电比	蓄电池容量	熔盐电池容量
2.38E-02	9.21E-01	5.82E+00	1.00E+00	4.70E-01	5.77E+01	9.01E+01
光伏装机	光热装机	风力装机				总成本
0.1	6.2	19.5				48689.96

上述数据表明，该方案供电以风力发电为主，几乎不采用光伏发电；光热产生的能量主要存储于熔盐电池之中，不考虑与风力一起满足供电需求。

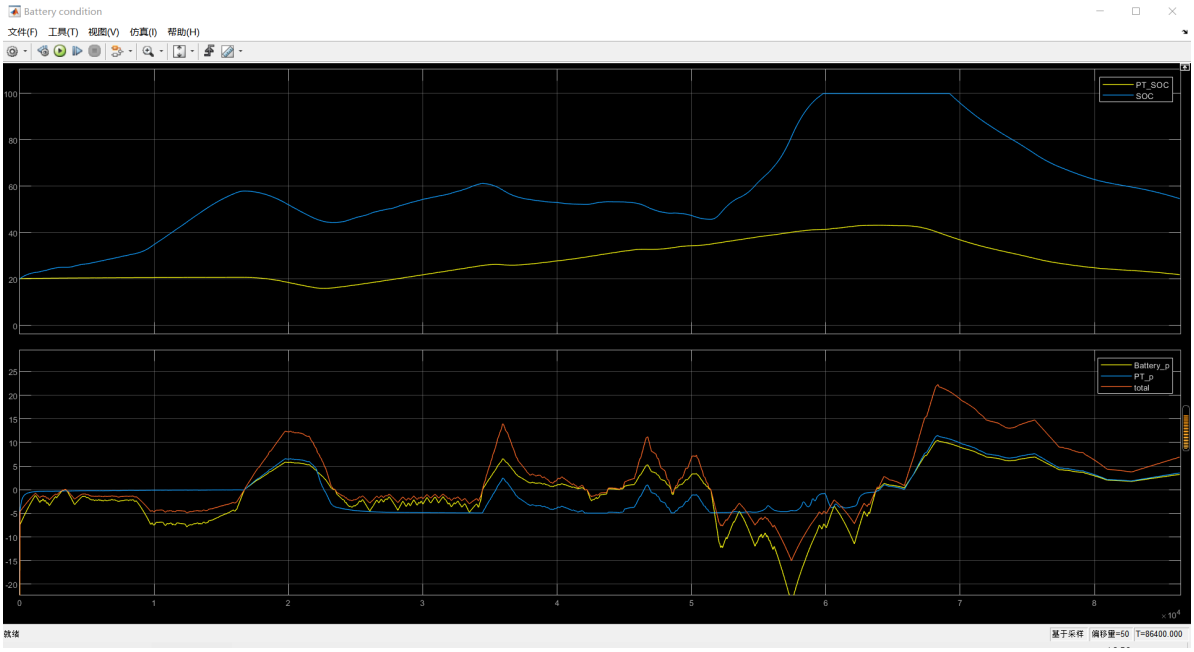
simulink模拟结果如下：

（黄-demand，深蓝-光伏发电（基本为0），红-光热产能用于供电的部分（基本为0），绿-蓄电池与熔盐电池总状态（即供电是否满足demand，电池处于供还是充），紫-风力发电，浅蓝-混合能源总发电功率，粉-熔盐电池释放功率）



（上：黄-熔盐电池充电量状态，蓝-蓄电池充电量状态

下：黄-双电池充放电状态，蓝-蓄电池功率状态，红-熔盐电池功率状态）



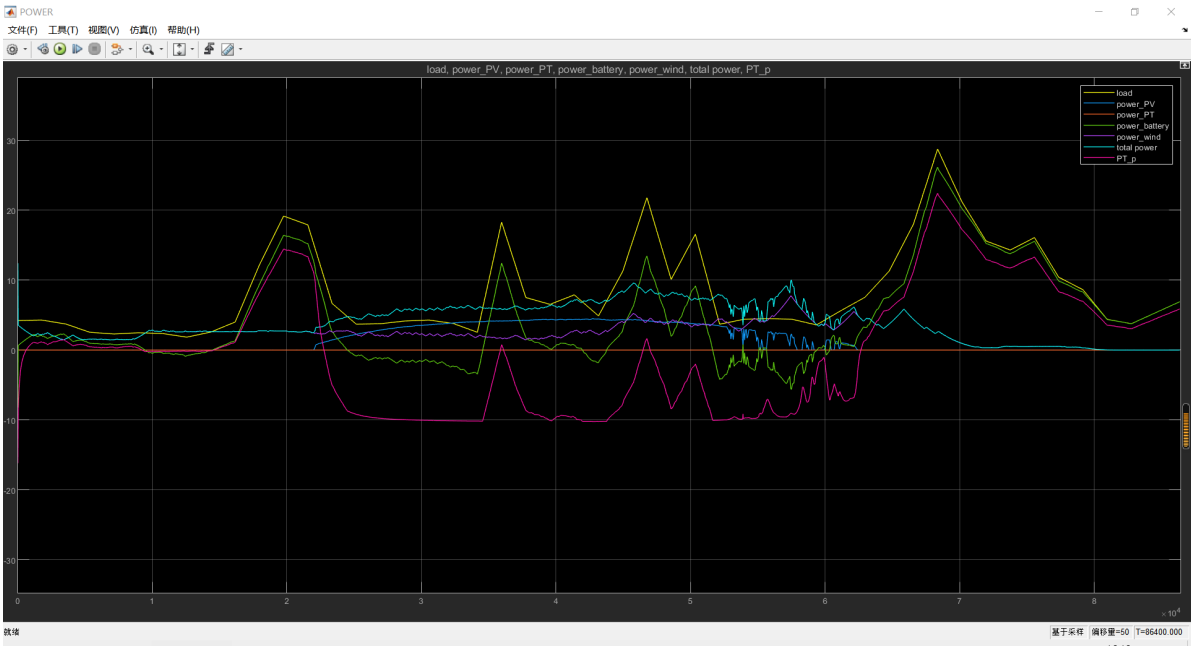
4-2 风力和光伏供电，光热以储热为主

光伏占比	光热总产能比	风力占比	光热储热比例	蓄电池放电比	蓄电池容量	熔盐电池容量
1.0733	1.9014	2.35	0.9949	0.132	25	170
光伏装机	光热装机	风力装机				总成本
4.51	12.8	7.86				50996

上述数据表明，该方案供电以风力发电和光伏发电为主；光热产生能量仍然主要存储于熔盐电池中。

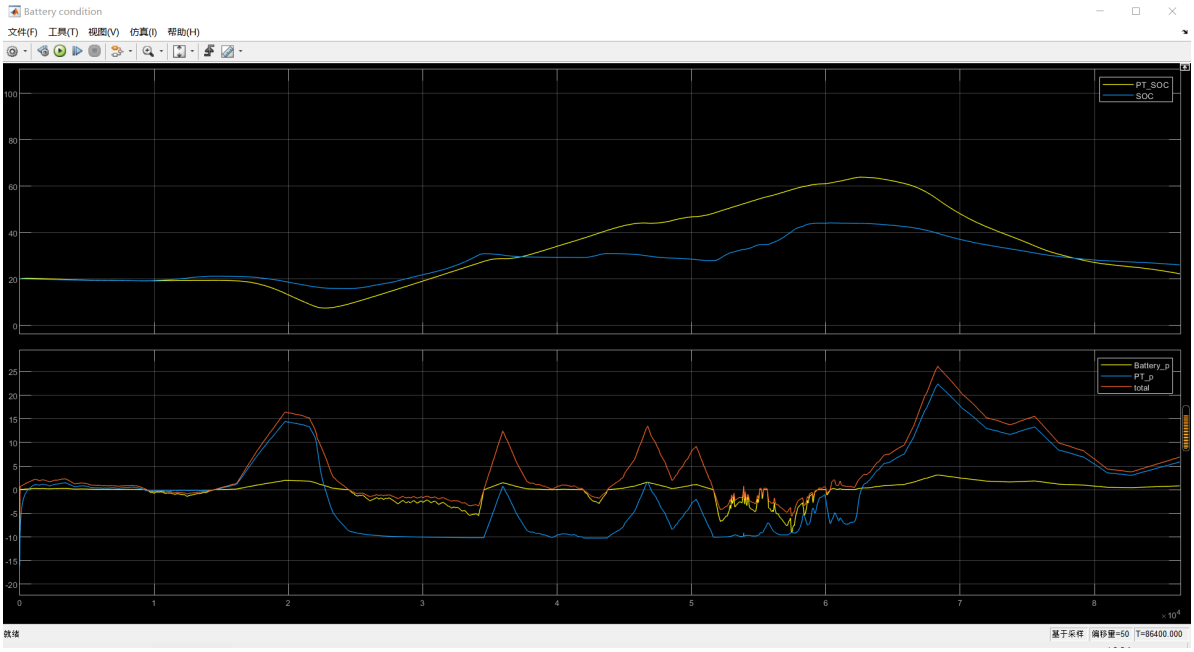
仿真结果如下：

（黄-demand，深蓝-光伏发电，红-光热产能用于供电的部分（基本为0），绿-蓄电池与熔盐电池总状态（即供电是否满足demand，电池处于供还是充），紫-风力发电，浅蓝-混合能源总发电功率，粉-熔盐电池释放功率）



（上：黄-熔盐电池充电量状态，蓝-蓄电池充电量状态

下：黄-双电池充放电状态，蓝-蓄电池功率状态，红-熔盐电池功率状态）



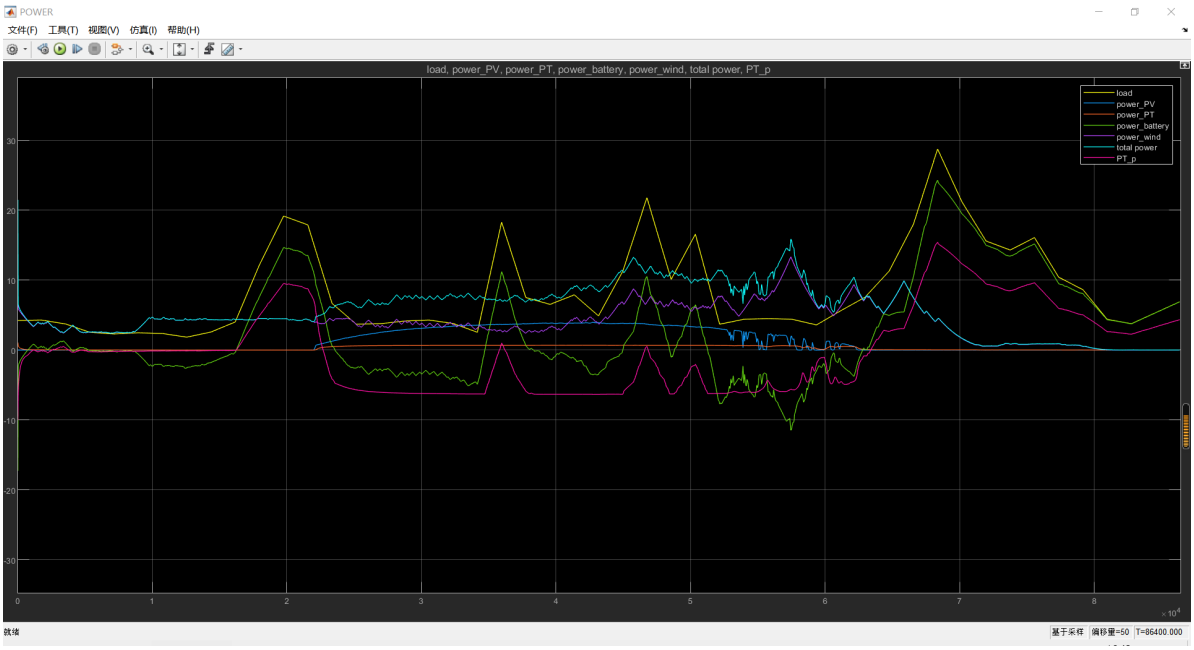
4-3风力为主，光伏光热辅助供电

光伏占比	光热总产能比	风力占比	光热储热比例	蓄电池放电比	蓄电池容量	熔盐电池容量
0.94	1.28	3.86	0.93	0.35	51	111
光伏装机	光热装机	风力装机				总成本
3.97	8.597	12.92				50438.7

上述数据表明，供电主要以风力发电为主，风力无法满足demand的部分由光伏和光热发电满足；光热产生的能量同时用于供电和储热。蓄电池和熔盐电池放电比为0.35:0.65。

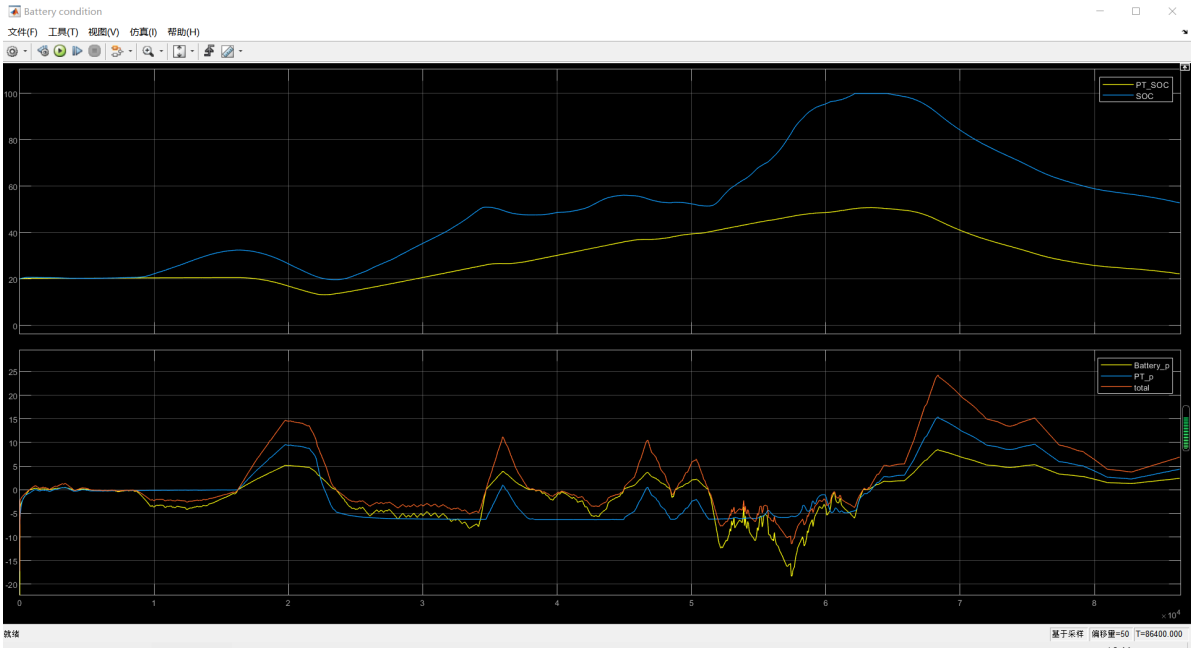
仿真结果如下：

（黄-demand，深蓝-光伏发电，红-光热产能用于供电的部分，绿-蓄电池与熔盐电池总状态（即供电是否满足demand，电池处于供还是充），紫-风力发电，浅蓝-混合能源总发电功率，粉-熔盐电池释放功率）



（上：黄-熔盐电池充电量状态，蓝-蓄电池充电量状态

下：黄-双电池充放电状态，蓝-蓄电池功率状态，红-熔盐电池功率状态）



5 小结

本次大作业搭建了包含风力、光热和光伏发电，及蓄电池和熔盐电池储能结构的混合可再生能源发电系统。根据一天内风力、光照和demand数据，实现了基于遗传算法的成本优化设计，并根据优化方案分别进行了仿真