稳态大作业（2）：机组组合

玄松元 吴晨聪 代泽昊

【数据读取】

在上次作业的基础上，读取机组运行成本的一次项和常数项。读取机组启停成本。设定发电机正常爬坡不超过0.1倍的最大出力，开机和关停的幅度不超过0.3倍的最大出力。

```
% 节点类型
bus = mpc.bus(:,1);
gen = mpc.gen(:,1);
gen_index = [1;2;3;4;5;6;7;8;9;10];
bus_index_noGen = find(mpc.bus(:,2) == 1);%节点类型为1的节点无发电机
bus_index_hasGen = [30;31;32;33;34;35;36;37;38;39];%有一个发电机的节点
% 负荷
load = Pd;
% 时长
time = 96;
time_0 = 4;
% 机组出力上下限
gen_Pmax = mpc.gen(:,9);
gen_Pmin = mpc.gen(:,10);
% 线路潮流上下限
branch_Pmax = mpc.branch(:,6);
branch_Pmin = -mpc.branch(:,6);
% 机组运行成本，一次项和常数项
gen_cost_1 = mpc.gencost(:,6);
gen_cost_0 = mpc.gencost(:,7);
% 机组启停成本
gen_cost_startup = repmat(mpc.gencost(:,2),1,time-1);
gen_cost_shutdown = repmat(mpc.gencost(:,3),1,time-1);
% 定义爬坡边界，爬坡为0.1倍最大出力，开机关停为0.3倍最大出力
R = repmat(0.1.*gen_Pmax,1,time-1);
S = repmat(0.3.*gen_Pmax,1,time-1);
```

图 1 基础参数设置

【决策变量】

在上次作业的基础上，增加发电机启停状态（0-1）决策变量和启停成本决策变量。

```
%% 定义决策变量
% 发电机发电量
gen_generate = sdpvar(length(gen(:,1)),time,'full');
% 发电机启停状态（运行为1，停机为0）
gen_state = binvar(length(gen(:,1)),time,'full');
% 节点注入有功
bus_P = sdpvar(length(bus(:,1)),time,'full');
% 将启停成本改写为决策变量构成约束条件
gen_cost_updown = sdpvar(length(gen(:,1)),time-1,'full');
```

图 2 决策变量和启停成本决策变量设置

【目标函数】

优化目标定义为

$$\min \sum_{t=1}^{T}\sum_{i=1}^{N}[C_i^1 g_{i,t} + C_i^0 x_{i,t} + C_{i,t}^{OC}]$$

包括常数项和一次项系数，注意乘以时间比例 0.25。机组启停成本则转化为约束条件在下文中实现，此处只需要求和。

```
%% 计算目标函数
% 由一次项、常数项和启停项构成
object = 0.25*sum(gen_cost_1'*gen_generate) + 0.25*sum(gen_cost_0'*gen_state) + sum(sum(gen_cost_updown));
```

图 3 目标函数

【约束条件】

在上次的基础上，增添和修改以下约束条件：

1. 发电机出力爬坡约束，需要额外考虑开机和关停爬坡。

$$\begin{cases} g_{i,t+1} - g_{i,t} \le S_i^u + (R_i^u - S_i^u)x_{i,t} \\ g_{i,t} - g_{i,t+1} \le S_i^d + (R_i^d - S_i^d)x_{i,t+1} \end{cases}, \forall i, \forall t \in [1, T-1], \begin{cases} \text{上爬坡约束} \\ \text{下爬坡约束} \end{cases}$$

2. 发电机最大出力约束，只需要直接乘以启停状态。

$$g_{i,t} \in [x_{i,t} g_i^{\min}, x_{i,t} g_i^{\max}]$$

3. 机组启停成本约束，为了求目标函数的成本。

$$\begin{cases} C_{i,t}^{OC} \ge z_{i,t} M_i^O \\ C_{i,t}^{OC} \ge -z_{i,t} M_i^C \end{cases}, \forall i, \forall t \in [1, T-1]$$

4. 机组最小持续开停机时间约束，开机不少于 1 小时，关停不少于 1 小时。

$$\begin{cases} \sum_{k=t+1}^{t+T_0} x_{i,k} \ge z_{i,t} T_0 = (x_{i,t+1} - x_{i,t}) T_0, \forall i, \forall t \in [1, T-T_0] \\ \sum_{k=t+1}^{t+T_0} (1 - x_{i,k}) \ge -z_{i,t} T_0 = -(x_{i,t+1} - x_{i,t}) T_0, \forall i, \forall t \in [1, T-T_0] \end{cases}$$

```
%% 列写约束条件
cons = [];
% 系统有功平衡约束，各节点的功率是平衡的
cons = [cons,bus_P(bus_index_noGen,:) == -load(bus_index_noGen,:)];%无发电机节点
cons = [cons,bus_P(bus_index_hasGen,:) == gen_generate(gen_index,:) - load(bus_index_hasGen,:)];%单发电机节点
% 系统有功平衡约束，总的有功供需是平衡的，即每个节点的有功注入之和为0
cons = [cons,sum(bus_P(bus,:)) == zeros(1,time)];
% 线路潮流约束，可以求功率传输分布因子矩阵
matrix_PTDF = makePTDF(mpc);
cons = [cons,matrix_PTDF * bus_P <= repmat(branch_Pmax,1,time)];
cons = [cons,matrix_PTDF * bus_P >= repmat(branch_Pmin,1,time)];
% 发电机出力约束，注意改写为乘以状态
cons = [cons,gen_generate <= gen_state.*repmat(gen_Pmax,1,time)];
cons = [cons,gen_generate >= gen_state.*repmat(gen_Pmin,1,time)];
% 发电机出力爬坡约束，注意同时考虑机组爬坡和启停
cons = [cons,gen_generate(:,2:end) - gen_generate(:,1:end-1) <=  S + (R-S).*gen_state(:,1:end-1)];
cons = [cons,gen_generate(:,1:end-1) - gen_generate(:,2:end) <=  S + (R-S).*gen_state(:,2:end)];
% 将启停成本改写为约束条件，状态差分变量，注意是矩阵逐个对应位置相乘
cons = [cons,gen_cost_updown >= (gen_state(:,2:end) - gen_state(:,1:end-1)).*gen_cost_startup];
cons = [cons,gen_cost_updown >= -1.*(gen_state(:,2:end) - gen_state(:,1:end-1)).*gen_cost_shutdown];
% % 机组最小开停机时间约束
cons = [cons,gen_state(:,2:93) + gen_state(:,3:94) + gen_state(:,4:95) + gen_state(:,5:96) >= 4.*(gen_state(:,2:93) - gen_state(:,1:92))];
cons = [cons,(1-gen_state(:,2:93)) + (1-gen_state(:,3:94)) + (1-gen_state(:,4:95)) + (1-gen_state(:,5:96)) >= 4.*(gen_state(:,2:93) - gen_state(:,1:92))];
```

图 4 约束条件设置

上图中有第一种写法，即展开成向量相加的形式，但可复用性不强。可以引入两个矩阵。这两个矩阵分别右乘在不等式两边。左边负责对 4 个时段求和，右边负责对状态进行差

分。

$$\begin{cases} XU_0 \geq T_0 XU(:,1:T-T_0) \\ (1-X)U_0 \geq -T_0 XU(:,1:T-T_0) \end{cases}$$

$$U_0 = \begin{bmatrix} 0 & 0 & 0 & 0 & \cdots \\ 1 & 0 & 0 & 0 & \cdots \\ 1 & 1 & 0 & 0 & \cdots \\ 1 & 1 & 1 & 0 & \cdots \\ 1 & 1 & 1 & 1 & \cdots \\ 0 & 1 & 1 & 1 & \cdots \\ \vdots & 0 & 1 & 1 & \cdots \\ \vdots & \vdots & 0 & 1 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \cdots \end{bmatrix}, \quad U = \begin{bmatrix} -1 & 0 & 0 & \cdots \\ 1 & -1 & 0 & \cdots \\ \vdots & 1 & -1 & \cdots \\ \vdots & \vdots & 1 & \cdots \\ \vdots & \vdots & \vdots & \cdots \\ \vdots & \vdots & \vdots & \cdots \\ \vdots & \vdots & \vdots & \cdots \end{bmatrix}$$

```
% %% 新写法
U0 = zeros(time,time-time_0);
for i = 1:(time-time_0)
    for j = (i+1):(i+time_0)
        U0(j,i) = 1;
    end
end
U = zeros(time,time-1);
for i = 1:(time-1)
    U(i,i) = -1;
    U(i+1,i) = 1;
end
cons = [cons, gen_state*U0 >= time_0.*gen_state*U(:,1:time-time_0)];
cons = [cons, (1-gen_state)*U0 >= -time_0.*gen_state*U(:,1:time-time_0)];
```

图 5 差分形式表达

【求解与展示结果】

```
%% 求解
options = sdpsettings('solver','gurobi');
result = optimize(cons,object,options);
```

```
%% 展示结果
fprintf('求解时间：%.4f 秒\n',result.solvertime);
fprintf('发电成本：%.2f Dollars\n',value(object));
gen_state_value = value(gen_state);
time_vector = 1:1:96;
% 使用 hsv 颜色图生成 11 种颜色
colors = hsv(10);

% 绘制阶梯图
figure;
hold on;
for i = 1:size(gen_generate, 1)
    stairs(time_vector, value(gen_generate(i,:)), 'LineWidth', 1.5, 'Color', colors(i,:));
end
hold off;

% 设置坐标轴标签
xlabel('时间/15min');
ylabel('发电机出力/MW');
title('各时段各发电机出力');

% 设置图例
legend(num2str(gen), 'Location', 'best');
```

图 6 求解设置

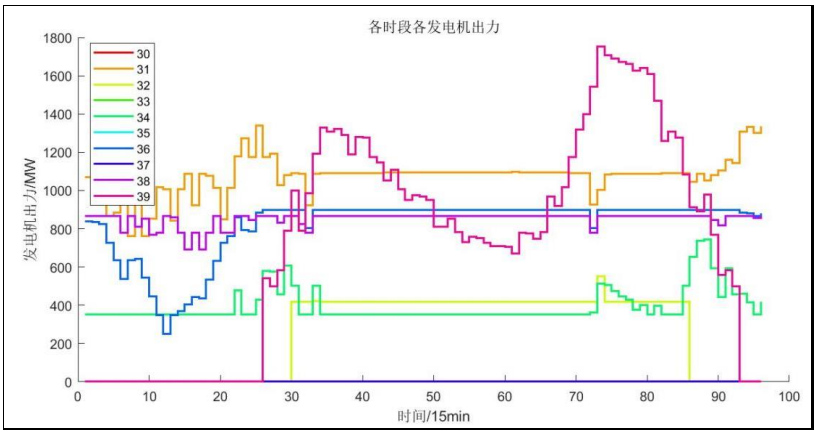


图 7 出力曲线

```
Optimal solution found (tolerance 1.00e-04)
Best objective 2.637398907451e+04, best bound 2.637398907451e+04,
求解时间：17.1073 秒
发电成本：26373.99 Dollars
```

图 8 求解结果

正常情况下的各发电机出力如上图，花费几秒的时间，发电成本为 26373.99 美元。

小组尝试其他代码撰写方式，计算出的发电成本与之相同，但是有一段时间的发电机出力变化与这张图有所不同。原因是 31,36,38 三台发电机在已经正常出力的情况下，发电成本的一次项系数相同，均为 0.25。本模型不考虑各种损耗的成本，计算存在大幅度的简化。因此，在整体约束条件宽松的范围内，无论发电机在何种位置出力，只要单位发电量的发电成本相同，都可能优化出总成本最优解，即最优解对应的调度情况可能并不唯一。

```
%%-----  OPF Data  -----%%
%% generator cost data
%   1   startup shutdown    n   x1  y1  ... xn  yn
%   2   startup shutdown    n   c(n-1)  ... c0
mpc.gencost = [
    2   30  0   3   0.008   0.3 0;
    2   20  0   3   0.011   0.25    2.5;
    2   25  0   3   0.015   0.3 2.7;
    2   23  0   3   0.03    0.35    2.5;
    2   20  0   3   0.018   0.3 2;
    2   25  0   3   0.02    0.3 2.2;
    2   28  0   3   0.015   0.25    2;
    2   20  0   3   0.02    0.3 2;
    2   30  0   3   0.012   0.25    2.8;
    2   32  0   3   0.009   0.28    4.000;
];
```

图 9 发电成本

【启停约束对系统运行状态的影响】

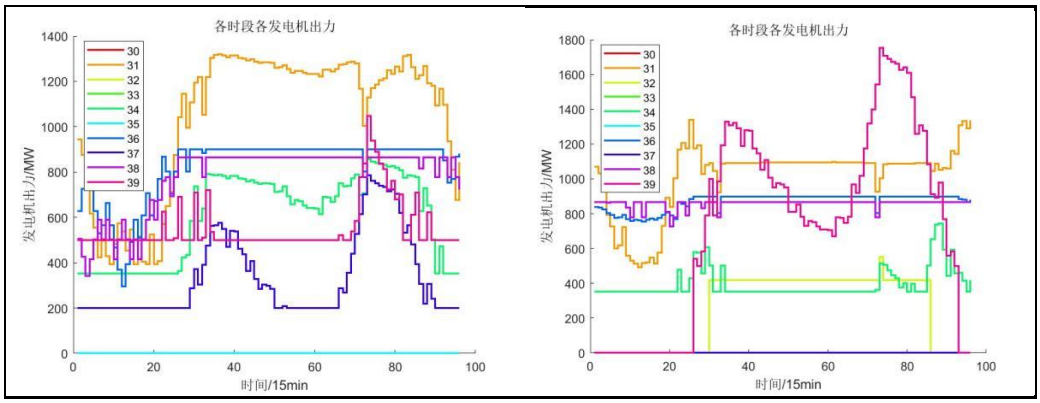
开机和关停爬坡速度的影响

将开机和关停爬坡限度改为 0.2 倍最大出力，爬坡能力大幅下降，引起了成本的上升。

```
Optimal solution found (tolerance 1.00e-04)
Best objective 2.657294216496e+04, best bound 2.657294216496e+04, g
求解时间：0.8481 秒
发电成本：26572.94 Dollars
```

将开机和关停爬坡限度改为 0.4 倍最大出力，爬坡能力增强，成本没法生变化。

```
Optimal solution found (tolerance 1.00e-04)
Best objective 2.637398907451e+04, best bound 2.637398907451e+04,
求解时间：14.5453 秒
发电成本：26373.99 Dollars
```



如图 10 所示，在 T_0=20 时的各机组启停时间是符合要求的。

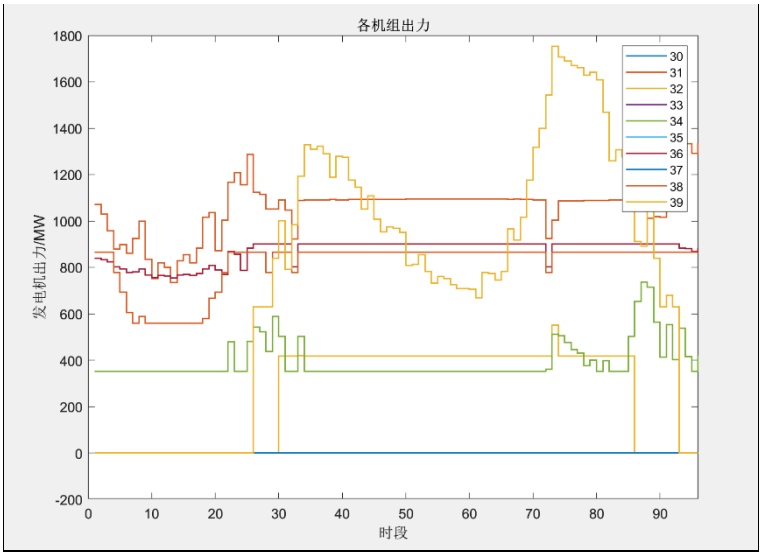


图 10 原方法 T_0=20 时各机组出力

但将 T_0 升高到 75 时，发现出现了不符合启停时间的约束。如图 11 所示，第 32 号节点上的发电机于 T=30 时刻开启，于 T=86 时刻停机，其中间隔时间要小于要求的 75。

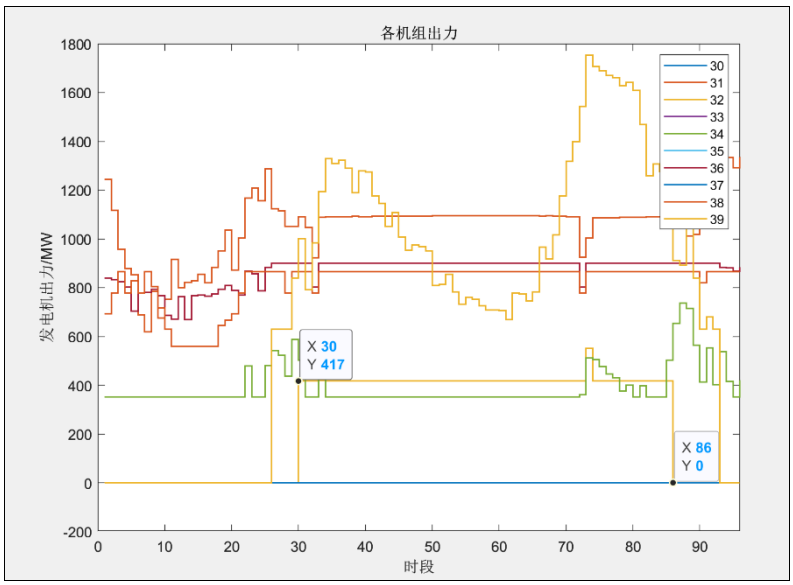


图 11 原方法 T_0=75 时各机组出力

分析其原因，按照课件上的关于机组最小持续开停机时间约束的表达形式，t 的取值范围为[1,96- T_0]，故 Z 的列数等于 96- T_0。也就是说，对于每个机组，只判断 t=96- T_0 之前每个开关动作之后的 T0 时间内是否产生新的开关动作。从逻辑上理解，就是在开启（关闭）之后，其状态 X 在此后 T_0 个值均为 1（0）则说明满足条件。而发生在 t=96- T_0 以后的开关时刻其后面剩下的点的个数要小于 T_0 个，故不考虑。

但是这样的考虑是存在漏洞的，以 T_0=75 为例，在 t=30 时刻发生的开关动作，其后只剩下 65 个点，若全为 1 仍然无法判断是否持续 75 个时刻状态未变，但若剩下 65 个点内存在 0，则肯定能说明其不满足最小关停机的条件。

因此，将原有的最小关停机的限制条件稍作改进：

$$XU_1 \geq T_1 .* XU(:,1:95)$$

其中：U_1 为 96×95 的矩阵，其主对角线及主对角线以下共 T0 条对角线的元素为 1，其余均为 0；

T_1 是 10×95 的矩阵，其每一列的值都是 min(T0,96-i)，i 为列数。

如此修改即可保证如图 11 这样的必然不满足最小启停时间约束的情况不发生。

利用改进后的方法得到总成本随着最小开停机时间 T_0 的变化趋势如图 12 所示。总体来说在 T_0≤56 时，发电机的出力都如图 10 所示，最短的发电机开停间隔是 32 号发电机（t=30 开启，t=86 停止），所以 T_0=56 之前的最优解是一样的，最优值也不变；而 T_0≥72 时，发电机的出力也是一样的，有过开关机动作的发电机只有两台（32 和 39），这两台发电机都只开未关；在 56＜T0＜72 的范围内，总成本会随着最小开停机时间的升高而升高，每种情况下各机组的出力也会不一样。而在所有的范围内，总发电成本最

小为 26377$，最大为 26438$。

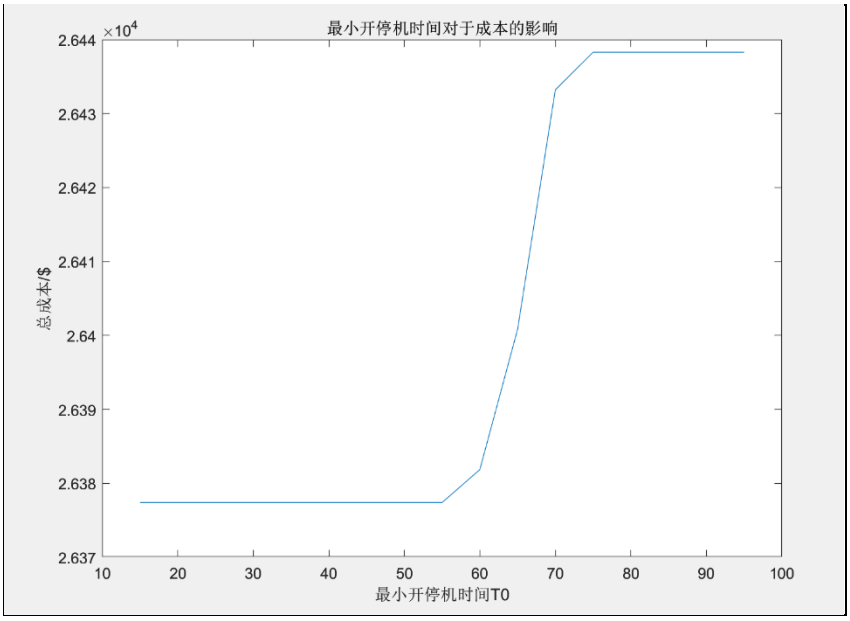


图 12 总成本随最小开停机时间 T0 变化趋势

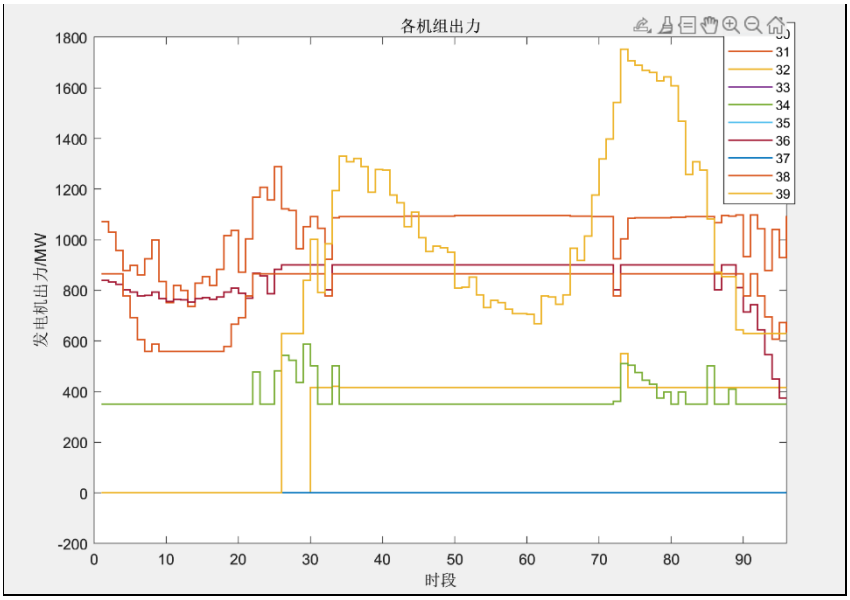


图 13 T0≥72 时各机组的出力曲线