The aim of this document is to describe the constraints presented in the function “MES\_MPC.m”, that poses and solves the optimization problem over the horizon T.

Particularly you can find a correspondence between the lines of code and the mathematical formulation in the file “IFAC14\_firstsubmission.pdf”.

%%%%%%CONSTRAINTS BUILDING%%%%%%

* Interaction with the utility grid - eq. (14)

constraints=constraints+set(E1g\*dg + E2g\*Cg <= E3g\*Pg + E4g);

* Ramp up and ramp down rates – eq. (17d)

constraints=constraints+set(-R <= P - Pprev <= R);

* CHP Jenbacher+Assorbitore Sanyo power levels considered not modulable (it can be 0 or maximum value)

constraints=constraints+set(P\_opt\_Jen(k)== P\_max\_Jen\*d\_opt\_Jen(k));

constraints=constraints+set(C\_opt\_Jen(k)== C\_max\_Jen\*d\_opt\_Jen(k));

* CHP Caterpillar+Assorbitore Sanyo power levels take value within a finite set of combination of electrical and cooling power – eq. (5a,5b)

constraints=constraints+set(P\_opt\_Caterp(:,k) == Pnom\_Caterp.\*d\_opt\_Caterp(:,k));

constraints=constraints+set(C\_opt\_Caterp(:,k) == Cnom\_Caterp.\*d\_opt\_Caterp(:,k));

* Chillers power levels take value within a finite set of values too:

constraints=constraints+set(C\_opt\_Chil3(:,k) == Cnom\_Chil3.\*d\_opt\_Chil3(:,k));

constraints=constraints+set(C\_opt\_Chil2(:,k) == Cnom\_Chil2.\*d\_opt\_Chil2(:,k));

constraints=constraints+set(C\_opt\_Chil1(:,k) == Cnom\_Chil1.\*d\_opt\_Chil1(:,k));

* Chiller1 and chiller2 work together (consider all the on/off status variables for each unit)

constraints=constraints+set(d\_opt\_Chil1(1,k)+d\_opt\_Chil1(2,k)+d\_opt\_Chil1(3,k)+d\_opt\_Chil1(4,k)==d\_opt\_Chil2(1,k)+d\_opt\_Chil2(2,k)+d\_opt\_Chil2(3,k)+d\_opt\_Chil2(4,k));

* The on/off status variables of each unit that needs more than one in order to take into account a finite set of operation mode (all the units except the Jenbacher) need to be mutually exclusive – eq.

constraints=constraints+set(d\_opt\_Caterp(1,k)+d\_opt\_Caterp(2,k)+d\_opt\_Caterp(3,k)<=1);

constraints=constraints+set(d\_opt\_Chil3(1,k)+d\_opt\_Chil3(2,k)+d\_opt\_Chil3(3,k)+d\_opt\_Chil3(4,k)<=1);

constraints=constraints+set(d\_opt\_Chil2(1,k)+d\_opt\_Chil2(2,k)+d\_opt\_Chil2(3,k)+d\_opt\_Chil2(4,k)<=1);

constraints=constraints+set(d\_opt\_Chil1(1,k)+d\_opt\_Chil1(2,k)+d\_opt\_Chil1(3,k)+d\_opt\_Chil1(4,k)<=1);

* start up and shut down constraints for each unit – eq.(2)

constraints=constraints+set(SD>=0);

constraints=constraints+set(SU>=0);

constraints=constraints+set(SU>=cSU.\*(d-dprev));

constraints=constraints+set(SD>=cSD.\*(dprev-d));

* Generator operations are constrained by the minimum up/down time that is the minimum amount of time for which a controllable generation unit must be kept on/off – eq. (1)

for i=1:Ng

uptime=0;

downtime=0;

if k==1

uptime=counter(1,i);

downtime=counter(2,i);

end

Td = Tdown(i) - downtime;

Tu = Tup(i) - uptime;

for t=k+1:min(k+Td-1,T)

constraints=constraints+set(dprev - d <= 1 - [d\_opt\_Jen(t); sum(d\_opt\_Caterp(:,t)); sum(d\_opt\_Chil3(:,t)); sum(d\_opt\_Chil2(:,t));sum(d\_opt\_Chil1(:,t))]);

end

for t=k+1:min(k+Tu-1,T)

constraints=constraints+set(d - dprev <= [d\_opt\_Jen(t); sum(d\_opt\_Caterp(:,t)); sum(d\_opt\_Chil3(:,t)); sum(d\_opt\_Chil2(:,t));sum(d\_opt\_Chil1(:,t))]);

end

if (k==1) && (dprev(i)==1)

for t=1:min(Tu,T)

constraints=constraints+set(d(t)==1 );

end

end

end

* Electric and cooling power balance – similar to eq. (15, 16)

constraints=constraints+set(P\_opt\_Jen(k)+P\_opt\_Caterp(1,k)+P\_opt\_Caterp(2,k)+P\_opt\_Caterp(3,k)+ RES + Pg >= demand\_e);

constraints=constraints+set(C\_opt\_Jen(k)+C\_opt\_Caterp(1,k)+C\_opt\_Caterp(2,k)+C\_opt\_Caterp(3,k)+C\_opt\_Chil3(1,k)+C\_opt\_Chil3(2,k)+C\_opt\_Chil3(3,k)+C\_opt\_Chil3(4,k)+C\_opt\_Chil2(1,k)+C\_opt\_Chil2(2,k)+C\_opt\_Chil2(3,k)+C\_opt\_Chil2(4,k)+C\_opt\_Chil1(1,k)+C\_opt\_Chil1(2,k)+C\_opt\_Chil1(3,k)+C\_opt\_Chil1(4,k)>= demand\_c);

* Cost function, taking into account eq. (6) when a unit has discrete generated power levels

objective = objective + Cg + cost\_Jen\_P\*P\_opt\_Jen(k) + cost\_Jen\_C\*C\_opt\_Jen(k) + cost\_Caterp\_P'\*P\_opt\_Caterp(:,k) + cost\_Caterp\_C'\*C\_opt\_Caterp(:,k) + cost\_Chil3'\*C\_opt\_Chil3(:,k) + cost\_Chil2'\* C\_opt\_Chil2(:,k) + cost\_Chil1'\* C\_opt\_Chil1(:,k) + sum(SD) + sum(SU);