1 Jabr Model

For the OPF model construction we the network as directed graph $(\mathbf{B}, \mathbf{L})$ where $\mathbf{B}$ is the set of Buses and and $\mathbf{L} \subset \mathbf{B} \times \mathbf{B}$ is the set of branches of the network and for each adjacent buses k, m both (k, m) and (m, k) are in $\mathbf{L}$. So the line l adjacent to k, m is modeled by two edges in the arc $\{(k, m), (m, k)\}$. L can be partitioned in $\mathbf{L}_0$ and L_1 with $|L_0| = |L_1|$ where every line l, adjacent to the buses k, m and with a transformer at k, is oriented so that $(k, m) \in L_0$ and $(m, k) \in \mathbf{L}_1$. We also consider a set $\mathcal{G}$ of generators, partitioned into (possibly empty) subsets $\mathcal{G}_k$ for every bus $k \in \mathbf{B}$. We consider the following convex Jabr relaxation of the OPF problem:

$$\inf_{\substack{P_g^G, Q_g^G, c_{km}, \\ s_{km}, S_{km}, P_{km}, Q_{km}}} \quad F(x) \coloneqq \sum_{g \in \mathcal{G}} F_g(P_g^G) \tag{1}$$

Subject to: $\forall km \in \mathbf{L}$

$$c_{km}^2 + s_{mk}^2 \leq c_{kk} c_{mm} \quad \text{Jabr constraint} \tag{2}$$

$$P_{km} = G_{kk} c_{kk} + G_{km} c_{km} + B_{km} s_{km} \tag{3}$$

$$Q_{km} = -B_{kk} c_{kk} - B_{km} c_{km} + G_{km} s_{km} \tag{4}$$

$$S_{km} = P_{km} + j Q_{km} \tag{5}$$

Power balance constraints: $\forall k \in \mathbf{B}$

$$\sum_{km \in L} S_{km} + P_k^L + i Q_k^L = \sum_{g \in \mathcal{G}(k)} P_g^G + i \sum_{g \in \mathcal{G}(k)} Q_g^G \tag{6}$$

Power flow, Voltage, and Power generation limits:

$$P_{km}^2 + Q_{km}^2 \leq U_{km} \tag{7}$$

$$V_k^{\min^2} \leq c_{kk} \leq V_k^{\max^2} \tag{8}$$

$$P_g^{\min} \leq P_g^G \leq P_g^{\max} \tag{9}$$

$$c_{kk} \geq 0 \tag{10}$$

$$c_{km} = c_{mk}, \; s_{km} = -s_{mk}. \tag{11}$$

This relaxation is in general not exact. We can recover exactness thanks to the following result:

Proposition 1. *Model* (1) *with the additional* loop constraint (12) *for every loop in a cycle basis of* $(\mathbf{B}, \mathbf{L})$ *is exact, we refer to this new model as the* Exact Jabr formulation

$$\sum_{k=0}^{\lfloor n/2 \rfloor} \sum_{\substack{A \subset [n] \\ |A| = 2k}} (-1)^k \prod_{h \in A} s_{k_h k_{h+1}} \prod_{h \in A^c} c_{k_h k_{h+1}} = \prod_{k=1}^{n} c_{k_i, k_i}. \tag{12}$$

In cite, auxiliary branches were added to the network, dividing each loop in smaller loops, to decrease the degree of the polynomials defining the loop

constraint. Then Mc Cormick linearization was applied. The problem with this approach is that one exiliary branch is added for every branch in the loop. This result suggests the following approaches to either find a feasible solution or move along the space of feasible solutions.

- Since the loop constraint is multilinear in it's variables, we can consider linear relaxations called *Flower inequalities*, which generalize che classical Mc Cormick relaxations of products of variables.
- Such relaxation is exact on tree Networks (also known as radial networks). Our objective is, given a network $\mathcal{N} = (\mathbf{B}, \mathbf{L})$ which can also not be a tree, consider a radial subnetwork $\mathcal{N}' = (\mathbf{B}, \mathbf{L}')$, with $\mathbf{L}' \subset \mathbf{L}$ and consider the Jabr model on $\mathcal{N}'$. This solution is not necessarily feasible for the original problem $\mathcal{N}$, our objective is to iteratively recover a feasible solution for $\mathcal{N}$. Since the Jabr relaxation is exact on $\mathcal{N}'$ it follows that the constraints

  2 are respected, the constraints which are violated are the flow constraints on the leaves. We can try to recover feasibility my moving along the solution to the Jabr and Loop constraints.
- Given a feasible solution, find feasible directions.

2 Linearization of loop constraints

To find feasible relaxations of the loop constraint we follow cite. It must be noted that a major difference in our approach is that the OPF is not a multilinear optimization problem. So first we show that the same results in cite can be applied to the OPF.

Consider a set of multilinear constraints:

$$\sum_{I \in \mathcal{I}_j} c_I^j \prod_{v \in I_j} x_v \quad \forall I \leq b_j \quad \forall j \in \{1, 2, \dots, m\} \tag{13}$$

$$x_v \in [l_v, u_v] \quad \forall v \in V \tag{14}$$

Where V denotes the variables and $\mathcal{I}_j \in \mathcal{P}(V)$, for $j = 1, \dots, m$ are the variables of the monomials appearing in the j-th homogenous constraint. A straight forward linearization is to introduce a variables z_I for every subset I of variables appearing in the constraints. Thus we obtain the following equivalent problem.

$$\sum_{I \in \mathcal{I}_j} c_I^j z_I \leq b_j \quad \forall j \in \{1, 2, \dots, m\} \tag{15}$$

$$z_I = \prod_{i \in I} x_i \quad \forall I \in \mathcal{E} \coloneqq \cup_{j=1}^{m} I_j \tag{16}$$

$$x_v \in [l_v, u_v] \quad \forall v \in V \tag{17}$$

By affine afformation we can assume the variables x_v to be in the form $c_v \in [0,1]$. Note that such affine transformations need to be handled with care, we will cover this in subsection 2.1. Since contraint (15) is now linear, we are now interested in the linearization of the following set $Pr \coloneqq \{(x,z) \in [0,1]^V \times [0,1]^{\mathcal{E}} \mid z_I = \prod_{i \in I} \forall I \in \mathcal{E}\}$. If such constraints were the only ones, and if the cost was also multilinear, we would know that the solution would be on one of the vertices of the hypercube and the observation that follows would be trivially true. Since in the OPF the cost is not multilinear and there are other types of constraints we show that this is also a relaxation for Pr.

Observation 2 (Standard form relaxation)**.** *The polyhedral PrR defined by the linear constraints* (18)-(21) *is a relaxation of Pr.*

G: fixalignment

$$z_I \leq x_v \quad \forall v \in I \in \mathcal{E} \tag{18}$$

$$z_I + \sum_{v \in I} (1 - x_v) \geq 1 \quad \forall I \in \mathcal{E} \tag{19}$$

$$z_I \geq 0 \quad \forall I \in \mathcal{E} \tag{20}$$

$$x_v \in [0,1] \quad \forall v \in V \tag{21}$$

Proof. We need to show that every element $x = (x_v, z_I)_{v \in V, I \in \mathcal{E}} \in Pr$ satisfies constraints (18)-(21). Constraints (18), (19), (20) obviously hold. For constraint (19), we see that for every choice of $x_v \in [0,1], v \in I$, we have $\prod_{v \in I} + \sum_{v \in I}(1 - x_v) \geq 1$. This is equivalent to showing that the function $a(x_1, \ldots, x_n) \coloneqq \prod_{i=1}^n x_i - \sum_{i=1}^n x_i + n - 1 \geq 0$ on $[0,1]^n$. This is trivially true for $n = 1$. If $n > 1$, let $HC \coloneqq [0,1]^n$ denote the hypercube, Let $F_i \coloneqq \{x \in HC \mid x_i = 0\}$ and $G_i \coloneqq \{x \in HC \mid x_i = 1\}$ denote the faces of the hypercube. We have that

G: This is actually much easier since by fixing all points except one we have an affine functions

$$HC = (HC \cap (F_1 + \langle e_1 - e_2 \rangle)) \cup (HC \cap (G_2 + \langle e_1 - e_2 \rangle)).$$

Thus it is sufficient to show that for every $x \in F_1$ and $y \in G_2$ the function $f_x(u) \coloneqq a(x + u(e_2 - e_1))$ and the funciton $g_y(v) \coloneqq a(y + v(e_1 - e_2))$ are non negative for all $u \in [0, x_2]$ and $v \in [x_1, 1]$ respectively. Expanding the expression of f_x we obtain:

$$f_x(u) = a(u, x_2 - u, x_3, x_4, \ldots, x_n) = -u^2 \prod_{i=3}^n x_i + u \prod_{i=2}^n x_n - sum_{i=2}^n x_i + n - 1$$

The second derivate of $f_x(u)$ respect to u is thus $f_x''(u) = -2 \prod_{i=3} x_i \leq 0$ and is thus concave respect to v. The value of f_x on the boundary points $u = 0, x_1$ is:

$$f_x(0) = a(0, x_2, \ldots, x_n) = \sum_{i=2}^n x_i + n - 1 \geq 0 \tag{22}$$

$$f_x(x_2) = a(x_2, 0, x_3, \ldots, x_n) = \prod_{i=2}^n x_i - \sum_{i=2}^n x_i + n - 2 \geq 0. \tag{23}$$

Where the second inequality is true by induction. Thus for every $x \in F_1$ for every $u \in [x_2, 1]$ we have $f_x \geq 0$. Analogly we have that for every $y \in G_2$ for every $v \in [0, x_1]$ we have $g_y \geq 0$. Thus for all $x \in C$, $a(x) \geq 0$ which concludes the proof. □

The corresponding Standard Form Relaxation for problem with homogeonus cost and constraints if often very weak. As done in cite we augment 2 with *Flower Inequalities*, which are additional inequalities valid for Pr. Again the main difference with cite: McCormick stikes back, is that we cannot restict the hypercuber C to its vertices becase other point could also be optimal for the OPF problem. Se we check the additional flower inequalities are still valid for Pr.

Definition 1 (extendend flower inequalities.)**.** *Let $I \in \mathcal{E}$ and let $J_1, \ldots, J_k \in \mathcal{E} \cup \mathcal{S}$ be such that $I \subset \bigcup_{i=1}^{k} J_i$ and $I \cap J_i \neq \emptyset$ holds for $i = 1, 2, \ldots, k$. The* extended flower inequality *with center I and petals $J_1, \ldots, J_k$ is defined as*

$$z_I + \sum_{i=1}^{k} (1 - z_{J_i}) \geq 1 \tag{24}$$

Proposition 3. *For all $x \in Pr$ and $I \in \mathcal{E}$, $J_1, \ldots, J_k \in \mathcal{E} \cup \mathcal{S}$ such that $I \subset \bigcup_{i=1}^{k} J_i$ and $I \cap J_i \neq \emptyset$ for $i = 1, 2, \ldots, k$. Then* extended flower inequality (24) *with center I and petals $J_1, \ldots, J_k$ holds for x.*

Proof. For $|I| = 1$ this is trivially true. For $|I| > 1$ we want to see that for any $x \in C_{\cup_k J_k} := [0,1]^{\cup_{k=1}^{K} J_K}$ we have $a((x_j)_{j \in \cup J_k}) = \prod_{i \in I} x_i + \sum_{k=1}^{K} (1 - \prod_{j \in J_k} x_j) - 1 \geq 0$. Consider the face $F = \{x \in [0,1]^{\cup_{k=1}^{K} J_K} \mid x_1 = 0\}$. Then $C_{\cup J_k} = C_{\cup J_k} \cap (F + \langle e_1 \rangle)$. Thus we only need to show that for every $x \in F$ the function $f_x(x_1)$ where we keep fixed all the variables except x_0 is positive. If $i : [0,1]^{\cup_{k=1}^{K} J_k \setminus \{1\}} \to F$ then $f_x(x_1) := a(i(x) + x_1 e_1)$. This is an affine function, thus it sufficient to show that it is positive at $x_1 = 0$ and $x_1 = 1$. For $x_1 = 0$ this is trivially true. For $x_1 = 1$, we have $f_x(1) = \prod_{i \in I \setminus \{1\}} x_i + \sum_{k=1}^{K} (1 - \prod_{j \in J_k \setminus \{1\}} x_j) - 1 \geq 0$ by induction on $|I|$. □

2.1 Handling affine trasformations

It must the noted that for each non linear affine trasformation, that is when the lower bound of the corresponding variable is not zero, the number of monomials increases. More precisly, given an monomial defined by $I \in \mathcal{E}$, let $I' \subset I$ be the subset of variables in I for which a nonlinear trasformation is applied. Then the monomial I is split into $2^{|I'|+1}$ new monomials. When the size of such I' is large this greatly increases the number of auxiliary variables z_J which must be introduced. Thus applying many non linear affine transformation can be very costly and complicates the handling of the constraints. For this reason, instead of applying non linear affine transformation, for each variables $v \in V$ such that $x_v \in [l_v, u_v]$ and $l_v * u_v \neq 0$, we split the problem in two new subproblems

having $v_x \in [l_v, 0]$ and $v_x \in [0, u_v]$ respectively. This way linear transformations can be applied in the subproblems. This creates many subproblems, many of which are unfeasible for the OPF, we diminish the number of subproblems we need to solve thanks to some unfeasibility conditions. Then, instead of solving each subproblem in a random order, we rewrite the subproblems as a unique mixed integer programming problem.

3 Feasible directions

Let $x_0 = (P_0, Q_0, c_0, s_0)$ be a feasible solution of the OPF problem (1) with the loop constraints (12). We want to find feasible directions $x_0 = (dP, dQ, dc, ds)$, that is such that $x_1 = (P_0 + dP, Q_0 + dQ, c_0 + dc)$ is still a feasible solution of the OPF problem. We consider each constraint of the Exact Jabr Formulation separately do get feasible directions.

3.1 Jabr Constraint

Since x_0, the jabr equality holds: $c_{ii}c_{jj} = c_{ij}^2 + s_{ij}^2$. Adding the movement dx we want that $(c_{ii} + dc_{ii})(c_jj + dc_{jj}) = (c_{ij} + dc_{ij})^2 + (s_{ij} + ds_{ij})^2$. By expanding the terms and considering that the equality holds for x_0 this is equivalent to:

$$dc_{ii}dc_{jj} + 2c_{ii}dc_{jj} + 2dc_{ii}c_{jj} = dc_{ij}^2 + c_{ij}dc_{ij} + ds_{ij}^2 + s_{ij}ds_{ij} \tag{25}$$

For now we consider movements where dc_{ii} is not zero only on an independent set of nodes in the graph, this way the constraint simplifies to:

$$dc_{ij}^2 + c_{ij}dc_{ij} + ds_{ij}^2 + s_{ij}ds_{ij} - 2dc_{ii}c_{jj} = 0 \tag{26}$$

The solutions to this constraint can be found by minimizing the following minimization problem:

$$\min(dc_{ij}^2 + c_{ij}dc_{ij} + ds_{ij}^2 + s_{ij}ds_{ij} - 2dc_{ii}c_{jj})^2 \tag{27}$$

Which can be solved by gradient descent methods. Since we want to solve this for all $(i, j) \in \mathbf{L}$, we instead solve the following:

$$\min CJ(dc, ds, dP) = \sum_{(i,j)\in\mathbf{L}} (dc_{ij}^2 + c_{ij}dc_{ij} + ds_{ij}^2 + s_{ij}ds_{ij} - 2dc_{ii}c_{jj})^2 \tag{28}$$

G: Anche questo può essere risolto con metodo gradiente? Sappiamo che il minimo globale è zero, ma possono esserci minimi locali? Dobbiamo per forze imporre gli spostamenti solo su nodi indipendenti?

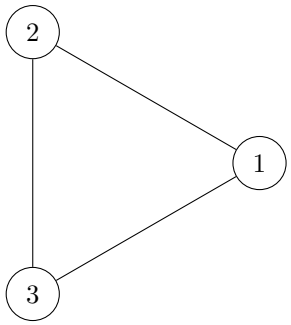
3.2 Loop Constraints

Unfortunately, the number of monomials to computer for checking the various of loop constraints grows factorially with the length of the cycle. Thus trying to

apply a similar method to find feasible directions satisfying the loop constraints as we did for the Jabr constraints would be computationally intractable. We can try the following approaches:

- Find the cycle basis with minimal weight
- Each cycle can be broken down by adding auxiliary edges in the Network, decomposing each loop constraints in loop contraints associated to cycle of the desired length. For each auxiliary edge $e = (e_0, e_1)$ we add the auxiliary variable c_e, s_e, the associated Jabr constraints: $c_e^2 + s_e^2 = c_{e_0} c_{e_1}$. We can now divide the cycles containing e_0 and e_1 in two cycles containing the edge e and thus obtaining smaller associated loop contraints.
- Loop constraints of cycles of length equal to 3 or 4 can be rewritten as 2 polynomial constraints of degree 2.

Let us consider the loop constraints on the following cycle:



This corresponds to the following polynomial constraints:

$$p_3 \coloneqq c_{12}(c_{23}c_{31} - s_{23}s_{31}) - s_{12}(s_{23}c_{31} + c_{23}s_{31}) - c_{11}c_{22}c_{33} \tag{29}$$

We also define the following polynomials associated to the Jabr constraints at each edge:

$$p_{ij} \coloneqq c_{ij}^2 + s_{ij}^2 - c_{ii}c_{jj} \tag{30}$$

and

$$q_3^1 = s_{12}c_{33} + c_{23}s_{31} + s_{23}c_{31} \tag{31}$$

$$d_3^2 = c_{12}c_{33} + c_{23}c_{31} + s_{23}s_{31} \tag{32}$$

Then, we have the following results which let's us to rewrite the constraints associated to a cycle of length three as two polynomial constraints of degree 2.

Proposition 4.

$$\{(c,s) \mid p_3 = p_{12} = p_{23} = p_{31} = 0\} = \{(c,s) \mid q_3^1 = q_3^2 = p_{12} = p_{23} = p_{31} = 0\} \tag{33}$$

If we don't restrict the possible direction $dx = (d_c, d_s, d_P)$, imposing that constraints (31), (32) hold for $x + dx$ would impose polynomial constraints on dx. Alternatively, since for example q_3^1 is a bilinear polynomial respect to the vectors $v_1 = (s_{12}, c_{23}, s_{23})$ and $v_2 = (c_{33}, s_{31}, c_{31})$, we can fix the variables in v_2, that is $dv_2 = 0$ and move along v_1. This way constraint $q_3^1 = 0$ imposes a linear constraint on d_x. We can proceed on a similar way for q_3^2. We note that one can consider instead to move along v_2 and that there are other possible choices for v_1 and v_2.

G: The hope is that even though we are restricting the directions for a single step, that by concatenating steps we can obtain a "good" amount of feasible directions.

G: TODO: think well about possible directions

3.3 Flow constraints

The flow constraint (6) also induce linear constraints on the step dx:

$$dP_{km} = G_{kk}dc_{kk} + G_{km}dc_{km} + B_{km}ds_{km} \tag{34}$$

$$dQ_{km} = -B_{kk}dc_{kk} - B_{km}dc_{km} + G_{km}ds_{km} \tag{35}$$

$$dS_{km} = dP_{km} + jdQ_{km} \tag{36}$$

$$\sum_{km \in L} dS_{km} + dP_k^L + idQ_k^L = \sum_{g \in \mathcal{G}(k)} dP_g^G + i \sum_{g \in \mathcal{G}(k)} dQ_g^G \tag{37}$$

3.4 Other constraints

Magnitude constraints are what limit the step size:

$$V_k^{\min^2} \leq c_{kk} + dc_{kk} \leq V_k^{\max^2} \tag{38}$$

$$P_g^{\min} \leq P_g^G + dP_g^G \leq P_g^{\max} \tag{39}$$

$$c_{kk} + dc_{kk} \geq 0 \tag{40}$$

G: what about this one?

$$P_{km}^2 + Q_{km}^2 \leq U_{km}$$

3.5 Step Problem

Combining the considerations in the previous subsections we obtaing the following problem to calculate a feasible step. Where given a graph, we find a cycle basis, we decompose each cycle in a 3/4-cycles, for each of these smaller cycles we consider their loop constraints and pick which variables do move in the step dx.

G: how?

G: Da riscrivere meglio, i vincoli su q_e^1, q_e^2 sono per ciascun arco in un ciclo in una base di cicli, ma se lo stesso arco è in due basi di cicli diverse, se i due cicli non hanno nodi in comune allora sono due vincoli diversi

$$\min CJ(dx) = sum_{(i,j)\in \mathbf{L}}(dc_{ij}^2 + c_{ij}dc_{ij} + ds_{ij}^2 + s_{ij}ds_{ij} - 2dc_{ii}c_{jj})^2 \tag{41}$$

Subject to:

$$q_e^1 = ds_{12}c_{33} + dc_{23}s_{31} + ds_{23}c_{31} \tag{42}$$

$$q_e^2 = dc_{12}c_{33} + dc_{23}c_{31} + ds_{23}s_{31} \tag{43}$$

$$dP_{km} = G_{kk}dc_{kk} + G_{km}dc_{km} + B_{km}ds_{km} \tag{44}$$

$$dQ_{km} = -B_{kk}dc_{kk} - B_{km}dc_{km} + G_{km}ds_{km} \tag{45}$$

$$dS_{km} = dP_{km} + jdQ_{km} \tag{46}$$

$$\sum_{km\in L} dS_{km} = \sum_{g\in\mathcal{G}(k)} dP_g^G + i\sum_{g\in\mathcal{G}(k)} dQ_g^G \tag{47}$$

$$V_k^{\min^2} \leq c_{kk} + dc_{kk} \leq V_k^{\max^2} \tag{48}$$

$$P_g^{\min} \leq P_g^G + dP_g^G \leq P_g^{\max} \tag{49}$$

$$c_{kk} + dc_{kk} \geq 0 \tag{50}$$

If the optimal solutions of problem (41) are feasible steps if and only if their cost is 0. Leaven like this one obvious solution is $dx = 0$. Possible ideas.

- Impose that the generation cost must decrease, by putting the scalar product of dx with the gradient of the cost function F in x to be smaller than 0. $dx\nabla F(x) < 0$. With this additional constraint if it's an equality then the family of the possible direction doesn't have a decreasing cost direction.

Since the constraints are affine, problem (41) can be solved with a projected gradient descent algorithm.

G: If we don't consider magnitude constraints, we can consider directly the projection of the gradient on the subspace of feasible solutions, this would be easier than considering the projection on a generic poligon (I think)