1 Introduction

This document describes the automatic MLCP formulation from a circuit. It consists in adapting the M.N.A. to manage non-smooth model.

2 Hypothesis

The M.N.A. assumes smooth branches are explicit functions of current or voltage. It means each smooth branch is Voltage Defined (V.D.) or Current Defined (C.D.)

With the same assumption, all branches can be divided into following classes:

1. The C.D. branches
2. The V.D. branches
3. The non smooth branches
4. The dynamical capacitor branches
5. The dynamical inductors branches

3 Smooth equations

1. The Kirchhoff Current Law (KCL).
   NB : The current in the capacitor branches is written with :
   $$I = C * \frac{d(V_i - V_j)}{dt}$$
2. The V.D. equation (VDE) :
   A V.D. branch give an equation :
   $$V_i - V_j = \sum_i a_i V_i + \sum_{j \in J} b_j I_j + \sum_i c_i V_i' + source$$
   This equation comes from the branch constitution, not from Kirchoff. (ex : U = R*I)
3. The inductor law (IL) : n_i equations.
   $$V_i - V_j = L * \frac{dI}{dt}$$

4 Non smooth branches

4.1 Local formulation

For each non smooth branch, the complementarity condition is:

$$\begin{pmatrix} \beta = z_i = B_i l + A_i X + a_i \\ y = C_i X + D_i l + e_i \\ 0 \leq y \perp l \geq 0 \end{pmatrix}$$

Where z_i is a voltage and currents vector. X is the vector of unknowns.

5 Matrices formulation

Before go head, we define the variables.
x contains only the dynamic variables(currents in inductor and tensions from capacitor branches)
Z_s contains only the non dynamic variables(Voltage nodes,...).
Z_{ns} contains the useful currents and tensions from the non smooth components.

We obtain the following system:

$$\begin{pmatrix} x' = A_{1x}x + A_{1zs}Z_s + A_{1ns}Z_{ns} + A_{1s} \\ 0 = B_{1x}x + B_{1zs}Z_s + B_{1ns}Z_{ns} + B_{1s} \\ Z_{ns} = C_{1x}x + C_{1zs}Z_s + C_{1\lambda}\lambda + C_{1s} \\ Y = D_{1x}x + D_{1zs}Z_s + D_{1ns}Z_{ns} + D_{1\lambda}\lambda + D_{1s} \\ 0 \leq Y \perp \lambda \geq 0 \end{pmatrix}$$

Substitute Z_{ns}:

$$\begin{pmatrix} R = A_{1ns}C_{1\lambda} & (eq1) \\ x' = (A_{1x} + A_{1ns}C_{1x})x + (A_{1zs} + A_{1ns}C_{1zs})Z_s + R\lambda + A_{1s} + A_{1ns}C_{1s} & \\ x' = A_{2x}x + A_{2zs}Z_s + R\lambda + A_{2s} & (eq2) \\ 0 = (B_{1x} + B_{1ns}C_{1x})x + (B_{1zs} + B_{1ns}C_{1s})Z_s + B_{1ns}C_{1\lambda}\lambda + B_{1s} + B_{1ns}C_{1s} & \\ 0 = B_{2x}x + B_{2zs}Z_s + B_{2\lambda}\lambda + B_{2s} & (eq3) \\ Y = (D_{1x} + D_{1ns}C_{1x})x + (D_{1zs} + D_{1ns}C_{1s})Z_s + (D_{1\lambda} + D_{1ns}C_{1\lambda})\lambda + D_{1s} + D_{1ns}C_{1s} = & \\ Y = D_{2x}x + D_{2zs}Z_s + D_{2\lambda}\lambda + D_{2s} & (eq4) \\ 0 \leq Y \perp \lambda \geq 0 & (\perp) \end{pmatrix}$$

A_{2s}, B_{2s} and D_{2s} are vectors.

5.1 A time discretisation

$$\left(\begin{array}{cc} R = A_{1ns}C_{1\lambda} & (eq1) \\ x' = A_{2x}x + A_{2zs}Z_s + R\lambda + A_{2s} & (eq2) \\ 0 = B_{2x}x + B_{2zs}Z_s + B_{2\lambda}\lambda + B_{2s} & (eq3) \\ Y = D_{2x}x + D_{2zs}Z_s + D_{2\lambda}\lambda + D_{2s} & (eq4) \\ 0 \le Y \perp \lambda \ge 0 & (\perp) \end{array}\right)$$

eq1 discretisation:

$$x(t_{i+1}) - x(t_i) = h\theta A_{2x}x(t_{i+1}) + h(1-\theta)A_{2x}x(t_i) + h\theta A_{2zs}Z_s(t_{i+1}) +$$

$$h(1-\theta)A_{2zs}Z_s(t_i) + hR\lambda(t_{i+1}) + h\theta' A_{2s}(t_{i+1}) + h(1-\theta')A_{2s}(t_i)$$

We assume $(I - h\theta A_{2x})$ is regular,$W(I - h\theta A_{2x}) = I$.

$$x(t_{i+1}) = Wx_{free} + h\theta WA_{2zs}Z_s(t_{i+1}) + hWR\lambda(t_{i+1})(eq5)$$

With:

$$x_{free} = (I + h(1-\theta)A_{2x})x(t_i) + h(1-\theta)A_{2zs}Zs(t_i) + h\theta' A_{2s}(t_{i+1}) + h(1-\theta')A_{2s}(t_i)$$

eq3 discretisation:

$$0 = B_{2x}x(t_{i+1}) + B_{2zs}Z_s(t_{i+1}) + B_{2\lambda}\lambda(t_{i+1}) + B_{2s}(t_{i+1})$$

With eq5:

$$0 = B_{2x}Wx_{free} + B_{2s}(t_{i+1}) + (h\theta B_{2x}WA_{2zs} + B_{2zs})Z_s(t_{i+1}) + (hB_{2x}WR + B_{2\lambda})\lambda(t_{i+1})$$

Rename the matrices:

$$0 = q_{free} + B_{3zs}Z_s(t_{i+1}) + B_{3\lambda}\lambda(t_{i+1})$$

eq4 discretisation:

$$Y(t_{i+1}) = D_{2x}x(t_{i+1}) + D_{2zs}Z_s(t_{i+1}) + D_{2\lambda}\lambda(t_{i+1}) + D_{2s}(t_{i+1})$$

With eq5 and rename the matrices:

$$Y(t_{i+1}) = D_{2x}Wx_{free} + D_{2s}(t_{i+1}) + (D_{2zs} + h\theta D_{2x}WA_{2zs})Z_s(t_{i+1}) + (D_{2\lambda} + hD_{2x}WR)\lambda(t_{i+1})$$

Rename the matrices:

$$Y(t_{i+1}) = p_{free} + D_{3zs}Zs(t_{i+1}) + D_{3\lambda}\lambda(t_{i+1})$$

MLCP description:

$$W = \begin{pmatrix} W_1 \\ 0 \end{pmatrix}$$

$$Z = \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix}$$

$$W = MZ + q$$

$$0 \le W_1 \perp Z_1 \ge 0$$

MLCP instance We identify a MLCP:

$$W_1 = Y(t_{i+1})$$
$$Z_1 = \lambda(t_{i+1})$$
$$Z_2 = Z_s(t_{i+1})$$
$$M = \begin{pmatrix} D_{3\lambda} & D_{3zs} \\ B_{3\lambda} & B_{3zs} \end{pmatrix}$$
$$q = \begin{pmatrix} p_{free} \\ q_{free} \end{pmatrix}$$

6 Unknowns

This section describes how chose the unknowns. A way could be to add all voltages, tensions and currents (STA). The result is big number of unknowns. We chose to add only the necessary unknowns to describe the circuit.

6.1 The unknowns are:

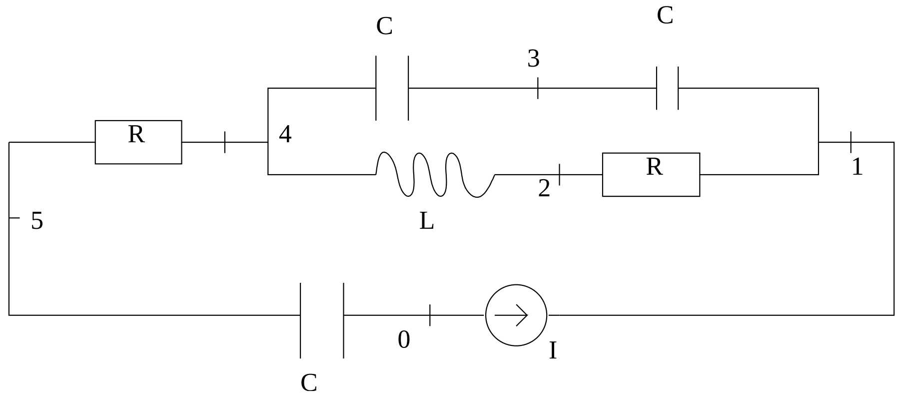
1. Nodal voltages
2. Currents in V.D. branches
3. Currents in non smooth branches
4. Currents in inductor branches
5. Tension in capacitor branches
6. Current in capacitor branches if necessary only(see following section).

7 How get $x' = Ax + BZ_s + CZ_{ns}$?

7.1 Example 1

Add currents and tensions from the capacitor $x =^t (I_{42}, U_{43}, U_{31}, U_{50}), Z_s =^t (V_1, V_2, V_3, V_4, V_5, I_{43}, I_{31}, I_{50})$
We obtain following equation:

$$\begin{array}{c} \begin{array}{cccc} I'_{42} & U'_{43} & U'_{31} & U'_{50} \end{array} \\ \begin{pmatrix} L & 0 & 0 & 0 \\ 0 & C & 0 & 0 \\ 0 & 0 & C & 0 \\ 0 & 0 & 0 & C \end{pmatrix} \end{array} x' = 0x + \begin{array}{c} \begin{array}{cccccccc} V_1 & V_2 & V_3 & V_4 & V_5 & I_{43} & I_{31} & I_{50} \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{pmatrix} \end{array} Z_s$$



$$
\begin{pmatrix} KCL(1) \\ KCL(2) \\ KCL(3) \\ KCL(4) \\ KCL(5) \\ U_{43} \\ U_{31} \\ U_{50} \end{pmatrix}
\left(\begin{array}{cccccccc}
V_1 & V_2 & V_3 & V_4 & V_5 & I_{43} & I_{31} & I_{50} \\ \hline
-\frac{1}{R} & \frac{1}{R} & 0 & 0 & 0 & 0 & 1 & 0 \\
\frac{1}{R} & -\frac{1}{R} & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & -1 & 1 & 0 \\
0 & 0 & 0 & -\frac{1}{R} & \frac{1}{R} & 1 & 0 & 0 \\
0 & 0 & 0 & \frac{1}{R} & -\frac{1}{R} & 0 & 0 & 1 \\
0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 \\
-1 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 1 & 0 & 0 & 0
\end{array}\right) Z_s +
\left(\begin{array}{cccc}
I_{42} & U_{43} & U_{31} & U_{50} \\ \hline
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{array}\right) x =
\begin{pmatrix} I \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}
$$

$$x' = BZ_s$$

$$Cx + BZ_s = s$$

$N_I = N_E = 12$

Add only tensions from the capacitor $x =^t (I_{42}, U_{43}, U_{31}, U_{50})$, $Z_s =^t (V_1, V_2, V_3, V_4, V_5)$
We obtain following equation:

$$
\begin{pmatrix} KCL(4) \\ KCL(3) \\ KCL(5) \end{pmatrix}
\underline{\left(\begin{array}{cccc}
I'_{42} & U'_{43} & U'_{31} & U'_{50} \\ \hline
L & 0 & 0 & 0 \\
0 & C & 0 & 0 \\
0 & C & C & 0 \\
0 & 0 & 0 & C
\end{array}\right)} x' =
\left(\begin{array}{cccc}
I_{42} & U_{43} & U_{31} & U_{50} \\ \hline
0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0
\end{array}\right) x +
\left(\begin{array}{ccccc}
V_1 & V_2 & V_3 & V_4 & V_5 \\ \hline
0 & 1 & 0 & -1 & 0 \\
0 & 0 & 0 & \frac{1}{R} & -\frac{1}{R} \\
0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & -\frac{1}{R} & \frac{1}{R}
\end{array}\right) Z_s
$$

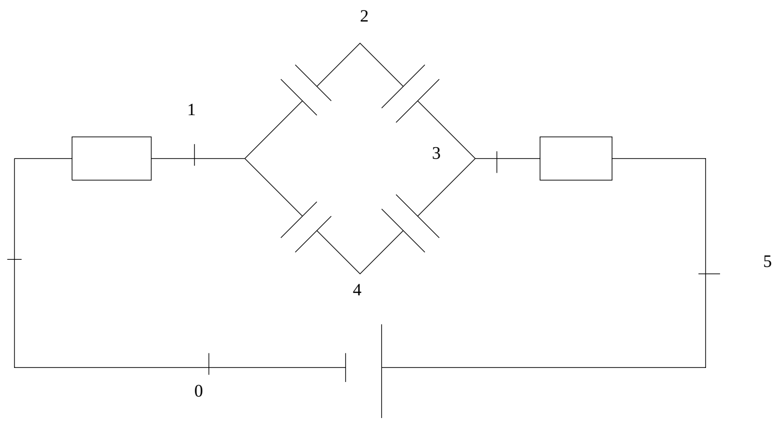
So, we get $x' = Ax + BZ_s$. Therefore, all currents in the capacitor branch are known.
$I_{43} = I_{42} + \frac{V_4}{R} - \frac{V_5}{R}$, $I_{31} = I_{42} + \frac{V_4}{R} - \frac{V_5}{R}$, $I_{50} = \frac{V_4}{R} - \frac{V_5}{R}$
Use these equations to fill following matrices:

$$
\begin{pmatrix} KCL(1) \\ KCL(2) \\ U_{43} \\ U_{31} \\ U_{50} \end{pmatrix}
\left(\begin{array}{ccccc}
V_1 & V_2 & V_3 & V_4 & V_5 \\ \hline
-\frac{1}{R} & \frac{1}{R} & 0 & \underline{\frac{1}{R}} & \underline{-\frac{1}{R}} \\
\frac{1}{R} & -\frac{1}{R} & 0 & 0 & 0 \\
0 & 0 & -1 & 1 & 0 \\
-1 & 0 & 1 & 0 & 0 \\
0 & 0 & 0 & 0 & 1
\end{array}\right) Z_s +
\left(\begin{array}{cccc}
I_{42} & U_{43} & U_{31} & U_{50} \\ \hline
\underline{1} & 0 & 0 & 0 \\
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 1
\end{array}\right) x =
\begin{pmatrix} I \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix}
$$

$N_I = N_E = 9$

7.2 Example 2

This example shows we have to manage the capacitor cycle.



$x =^t (U_{12}, U_{23}, U_{34}, U_{41}) Z_s = (V_1, V_2, V_3, V_4, V_5, I_{50})$
Start to write Ax'=...

$$\begin{pmatrix} KCL(1) \\ KCL(2) \\ KCL(3) \\ KCL(4) \end{pmatrix} \begin{pmatrix} \begin{array}{cccc} U'_{12} & U'_{23} & U'_{34} & U'_{41} \\ \hline C & 0 & 0 & -C \\ -C & C & 0 & 0 \\ 0 & -C & C & 0 \\ 0 & 0 & -C & C \end{array} \end{pmatrix} x' =$$

This matrix is not regular because of the cycle {1-2,2-3,3-4,4-1}. A solution could be to use the Minimum Spanning Tree {1-2,2-3,3-4} to write the KCL law. About the last tension, U_{41}, there are tow ways:

1. add a unknown $I_{41} in Z_S$ and write CU'=I
2. Find the linear relation $U'_{41} = \sum_{jk} a_{jk} U'_{kj}$, and replace U'_{ki}.

The matrices become:
Start to write Ax'=...

$$\begin{pmatrix} KCL(1) \\ KCL(2) \\ KCL(3) \\ I_{41} \end{pmatrix} \begin{pmatrix} \begin{array}{cccc} U'_{12} & U'_{23} & U'_{34} & U'_{41} \\ \hline C & 0 & 0 & -C \\ -C & C & 0 & 0 \\ 0 & -C & C & 0 \\ 0 & 0 & 0 & C \end{array} \end{pmatrix} x' = 0x + \begin{pmatrix} \begin{array}{ccccccc} V_1 & V_2 & V_3 & V_4 & V_5 & I_{50} & I_{41} \\ \hline -\frac{1}{R} & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & \frac{1}{R} & -\frac{1}{R} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{array} \end{pmatrix} Z_s$$

So, we obtain $x' = BZ_s$. Therefore all capacitor's currents are known:$I_{12} = I_{23} = I_{41} + \frac{V1}{R}, I_{43} = I_{41} + \frac{V1}{R} + \frac{V5}{R} - \frac{V3}{R}$

$$\begin{pmatrix} KCL(4) \\ KCL(5) \\ U_{12} \\ U_{23} \\ U_{34} \\ U_{41} \\ VD_{50} \end{pmatrix} \begin{pmatrix} \begin{array}{cccc} U_{12} & U_{23} & U_{34} & U_{41} \\ \hline 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 \end{array} \end{pmatrix} x + \begin{pmatrix} \begin{array}{ccccccc} V_1 & V_2 & V_3 & V_4 & V_5 & I_{50} & I_{41} \\ \hline \frac{1}{R} & 0 & -\frac{1}{R} & 0 & \frac{1}{R} & 0 & \underline{-1+1} \\ 0 & 0 & \frac{1}{R} & 0 & -\frac{1}{R} & -1 & 0 \\ -1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & -1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & -1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{array} \end{pmatrix} Z_s = \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \\ E \end{pmatrix}$$

$N_I = N_E = 11$

7.3 conclusion

The vector x contains inductor's currents and capacitor's tensions. Derivate inductor's current is equal to a nodal voltage difference.

About the capacitor's tensions, we use the Minimum Spanning Tree of the capacitor' tension to avoid cycle.

Algorithm 1 fill the matrices : $x' = A_{1x}x + A_{1s}Z_s + A_{1ns}Z_{ns}$

Require: Init_I_in_x : initialize internal data structure to get all I from x.
Require: Next_I_in_x : return the next available I from x. If there are not available x, return 0.
Require: Minimum Spanning Tree of the capacitor' tension graph
Require: Init_MST : initialize internal data structure to get all u from x.
Require: Next_u_in_MST : return a available U's neighbour from MST if possible. If there are not available u in MST, return 0.
Require: Next_U_in_x : return the next available U from x. If there are not available x, return 0.

```
0
//About current
//get first current
Init_I_in_x()
I_kj = Next_I_in_x()
while I_kj do
    use LI' = V_j − V_k to fill I_kj's line.
end while
//About tension
//get a first capacitor tension
Init_MST()
U_kj = Next_u_in_MST()
while U_kj do
    l=j or k with KCL(k) available.
    Use CU'=I and KCL(k) to fill U_ki's line.
    enable KCL(k)
    U_kj= Next_u_in_MST ()
end while
U_kj = Next_U_in_x()
while U_kj do
    Add an unknown I_kj, and use it to fill the matrices. Write I=CU'.
    U_kj = Next_U_in_x()
end while//reverse A : Ax' = Bx + CZ_s + DZ_ns
A^-1=Inv(A)
x' = A_1x x + A_1s Z_s + A_1ns Z_ns
```
